Masakazu Iwamura
Faisal Shafait (Eds.)

LNCS 7139

Camera-Based Document Analysis and Recognition

4th International Workshop, CBDAR 2011
Beijing, China, September 2011
Revised Selected Papers

Springer

Lecture Notes in Computer Science 7139

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Masakazu Iwamura   Faisal Shafait (Eds.)

Camera-Based Document Analysis and Recognition

4th International Workshop, CBDAR 2011
Beijing, China, September 22, 2011
Revised Selected Papers

Springer

Volume Editors

Masakazu Iwamura
Osaka Prefecture University
Graduate School of Engineering
Dept. of Computer Science and Intelligent Systems
1-1 Gakuencho, Naka Sakai, Osaka 599-8531, Japan
E-mail: masa@cs.osakafu-u.ac.jp

Faisal Shafait
German Research Center for Artificial Intelligence (DFKI GmbH)
Multimedia Analysis and Data Mining Competence Center
Trippstadter Str. 122, 67663 Kaiserslautern, Germany
E-mail: faisal.shafait@dfki.de

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-29363-4 e-ISBN 978-3-642-29364-1
DOI 10.1007/978-3-642-29364-1
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012934970

CR Subject Classification (1998): I.4, I.5, I.7, I.2.7, H.3, H.5, H.4

LNCS Sublibrary: SL 6 – Image Processing, Computer Vision, Pattern Recognition, and Graphics

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The pervasiveness and wide-spread availability of camera phones and hand-held digital still/video cameras have led the community to recognize document analysis and recognition of digital camera images as a promising and growing field of research. Constraints imposed by the memory, processing speed and image quality are leading to new interesting open problems which cannot be directly resolved by traditional techniques.

To cater to the demands of camera-based document processing, the idea of a new satellite workshop of the International Conference on Document Analysis and Recognition (ICDAR) was conceived by Koichi Kise. Together with David Doermann, he took the responsibility of organizing the first workshop on Camera-Based Document Analysis and Recognition as a satellite workshop of ICDAR 2005 in Seoul, Korea. The workshop was very well received by the community and hence it was held again in 2007 (Curitiba, Brazil), and 2009 (Barcelona, Spain) with the corresponding ICDAR conferences. Following the success of the past three workshops, the 4th International Workshop on Camera-Based Document Analysis and Recognition (CBDAR 2011) was held in Beijing, China, quite successfully with 68 participants. The workshop aimed to provide an opportunity to researchers and developers from various backgrounds to exchange their ideas and explore new research directions through presentations of the latest research activities and discussions.

In six years since the first CBDAR was held, the situation surrounding the CBDAR field has evolved. Taking photos/videos and uploading them to photo-sharing sites or one's blog have become more popular. New recognition services of scene text such as Evernote and Google Goggles are now available. Furthermore, a huge number of real images are available in Google Books and Street View. Needless to say that computer performance has been improved. Thus, it is high time to discuss and explore new research directions. This book contains refereed and improved versions of papers presented at CBDAR 2011 and is intended to give a snapshot of the state-of-the-art research in the field of camera-based document analysis and recognition.

The program of CBDAR 2011 was organized in a single-track one-day workshop. It comprised of two oral sessions and one poster session. In addition to that, two keynote talks were held by speakers from industry: Qiong Liu from FXPAL Inc. and Alessandro Bissacco from Google Inc. Finally, a panel discussion on the state of the art and new challenges was organized as the concluding session of CBDAR 2011.

After the workshop, authors of selected papers were invited to submit expanded versions of their papers for this edited volume. The authors were encouraged to include ideas and suggestions that arose in the panel discussions of the workshop. This volume is organized in three sections, reflecting the workshop session topics.

Finally, we would like to sincerely thank those who helped to make CBDAR 2011 a successful event: all paper authors, workshop attendees, Cheng-Lin Liu (ICDAR Executive Chair), Koichi Kise (ICDAR Workshop Chair) and other ICDAR organizers for their generous support, the members of the Program Committee and additional reviewers for reviewing and commenting on all of the submitted papers, and FXPAL, Google and DFKI for their financial support as sponsors of the workshop.

The 5th International Workshop on Camera-Based Document Analysis and Recognition (CBDAR 2013) is planned to be held at Washington DC, USA.

December 2011

Masakazu Iwamura
Faisal Shafait

Table of Contents

Text Detection and Recognition in Scene Images

Camera-Based Systems

Datasets and Evaluation

Multi-script and Multi-oriented Text Localization from Scene Images

Thotreingam Kasar and Angarai G. Ramakrishnan

Medical Intelligence and Language Engineering Laboratory,
Department of Electrical Engineering, Indian Institute of Science
Bangalore, India 560012
{tkasar,ramkiag}@ee.iisc.ernet.in

Abstract. This paper describes a new method of color text localization from generic scene images containing text of different scripts and with arbitrary orientations. A representative set of colors is first identified using the edge information to initiate an unsupervised clustering algorithm. Text components are identified from each color layer using a combination of a support vector machine and a neural network classifier trained on a set of low-level features derived from the geometric, boundary, stroke and gradient information. Experiments on camera-captured images that contain variable fonts, size, color, irregular layout, non-uniform illumination and multiple scripts illustrate the robustness of the method. The proposed method yields precision and recall of 0.8 and 0.86 respectively on a database of 100 images. The method is also compared with others in the literature using the ICDAR 2003 robust reading competition dataset.

Keywords: Text detection, scene text, multi-script documents, multi-oriented text, camera-based document analysis.

1 Introduction

Text provides useful semantic information that may be used to describe the content of a document image. While it is relatively easy to segment characters from clean scanned documents, text extraction from natural scenes is difficult since scene text can appear on any surface, not necessarily on a plane. They are often characterized by arbitrary text layouts, multiple scripts, artistic fonts, colors and complex backgrounds.

The Robust Reading Competition was held at the 7^{th} international conference on document analysis and recognition 2003 to find the system that best reads complete words in camera-captured images. The dataset contains various kinds of degradations such as uneven lighting conditions, complex backgrounds, variable font styles, low resolution and text appearing on shiny surfaces. However, there are no samples of inclined or multi-oriented text. It is also limited to English. In a multi-lingual country like India, many documents, forms and signboards are generally bi-lingual or multi-lingual in nature. Every script has certain distinctive characteristics and may require script-specific processing methods. Therefore, the presence of multiple scripts require a special treatment.

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 1–14, 2012.

There is a felt need for methods to extract and recognize text in scenes since such text is the primal target of camera-based document analysis systems. Unlike the problem of classical object-driven image segmentation, such as separating sky from mountains, pixel-accurate segmentation is required for character recognition. Robust extraction of text is a critical requirement since it affects the whole recognition process that follows.

2 Review of Text Detection

The existing methods for text detection fall under the two broad categories: texture based methods and connected component based methods. Texture based methods exploit the fact that text has a distinctive texture. They use methods of texture analysis such as Gabor filtering, wavelets and spatial variance. Zhong et al. [1] use local spatial variance of gray-scale image and locate text with high variance regions. Li and Doermann [2] use a sliding window to scan the image and classify each window as text or non-text using a neural network. Sabari et al. [3] uses multi-channel filtering with a Gabor filter bank on the grayscale image. The responses of the Gabor filters and color-based CC analysis are merged and text regions are obtained using geometrical and statistical information of the individual components. Wu et al. [4] employ 9 derivative of Gaussian filters to extract local texture features and apply k-means clustering to group pixels that have similar filter outputs. Assuming that text is horizontally aligned, text strings are obtained by aggregating the filtered outputs using spatial cohesion constraints. Clark and Mirmehdi [5] apply a set of five texture features to a neural network classifier to label image regions as text and non-text. Chen and Yuille [6] extract features based on mean intensity, standard deviation of intensity, histogram and edge-linking and classify using an AdaBoost trained classifier. Shivakumara et al. [7] compute median moments of the average sub bands of wavelet decomposition and use k-means clustering (k=2) to classify text pixels. The cluster with the higher mean is chosen as the text cluster. Boundary growing and nearest neighbor concepts are used to handle skewed text.

In CC based methods, the image is segmented into regions of contiguous pixels having similar characteristics like color, intensity or edges. These CCs are then classified using a set of features distinguishing textual and non-textual characteristics followed by grouping of the textual CCs. Robust segmentation of text CCs from the image is the most critical part in the process. Gatos et al. [8] segment the grayscale and inverted grayscale images by rough estimation of foreground and background pixels to form CCs which are further filtered by using height, width and other properties of the CCs. Zhu et al. [9] use Niblack method to segment the grayscale image and each CC is described by a set of low level features. Text components are classified using a cascade of classifiers trained with Adaboost algorithm. Pan et al. [10] propose a method to detect text using sparse representation. CCs are labeled as text or non-text by a two-level labeling process using an over-complete dictionary, which is learned from edge segments of isolated character images. Layout analysis is further applied to verify these text candidates.

CC-based approaches are suitable for camera-based images since they can deal with arbitrary font styles, sizes, color and complex layouts. However, their performance significantly degrades in the presence of complex backgrounds which interfere in the accurate identification of CCs. In this paper, we introduce a novel color clustering technique for robust extraction of CCs from complex images. A set of 'intrinsic' text features are designed from the geometric, boundary, stroke and gradient properties for classifying text CCs. Finally, adjacent text CCs are grouped together making use of the spatial coherence property of text to form words.

3 Color Text Segmentation

Since the boundaries of characters are always closed, potential text candidates are identified from the regions bounded by closed edges. Canny edge detection is performed individually on each channel of the color image and the overall edge map E is obtained by combining the three edge images as follows:

$$E = E_R \vee E_G \vee E_B \tag{1}$$

Here, E_R, E_G and E_B are the edge images corresponding to the three color channels and $\vee$ denotes the logical OR operation. This simple method yields the boundaries of all the characters present in the document image irrespective of its color, size or orientation. However, there are several instances of broken edges in practice. We perform an edge-linking procedure to bridge narrow gaps in the resulting edge image. The co-ordinates of the end points of all the open edges are then determined by counting the number of edge pixels in a 3×3 neighborhood and the center pixels when the count is 2. These pair of edge pixels indicate the direction of the edge and are used to determine the direction for a subsequent edge-following and linking process. Depending on the arrangement of the two edge pixels, there can be 8 possible search directions namely North, North-East, East, South-East, South, South-West, West and North-West. Fig. 1(a) shows two local edge patterns for which the edge-following step is done in the North and North-East directions. The other 6 search directions are given by multiples of 90^o rotated versions of these edge patterns. The edge-following process in continued for a distance λ in the estimated search direction. If an edge pixel is encountered, the traversed path is made 'ON' and is included in the edge map. If on the other hand, no edge pixel is found at the end of λ pixel traversal, the path is ignored.

To close gaps between parallel edge segments, a simple bridging operation is also performed by dilating the pixels at the end points of all the open edges with a structuring element shown in Fig. 1(b). These post-processing steps result in closing all the small gaps that may have arisen during the thresholding step of edge detection.

$$\begin{array}{|c|c|c|}\hline 0&0&0\\\hline 0&1&0\\\hline 0&1&0\\\hline\end{array}\quad\begin{array}{|c|c|c|}\hline 0&0&0\\\hline 0&1&0\\\hline 1&0&0\\\hline\end{array}$$

(a)

$$\begin{array}{|c|c|c|}\hline 0&1&0\\\hline 1&1&1\\\hline 0&1&0\\\hline\end{array}$$

(b)

Fig. 1. (a) Edge patterns for which direction of search for edge-linking process is in the North and North-East directions. The 6 other search directions are given by multiples of 90^o rotated versions of these edge patterns (b) Structuring element for bridging parallel gaps.

The color values of the region bounded by each of the closed edges are then considered for the subsequent color clustering step using COCOCLUST algorithm [11]. Each of these blobs is described by its mean $L^*a^*b^*$ color. A one-pass Leader clustering process is applied on the mean colors of all the segmented components. The obtained color clusters initialize a k-means clustering process. The clusters obtained at the end of convergence of the k-means clustering represent the final segmented output. The pseudo-code for the clustering algorithm is given below.

```
Input:Color Samples,CS = {C_1,C_2,...,C_M}
    Color similarity threshold, $T_s$
Output:Color clusters, CL
Assign CL[1] = C_1 and Count = 1
For i = 2 to M, do
  For j = 1 to Count, do
    If Dist(CL[j],C_i)$\leq$ $T_s$
      CL[j] = Mean({CL[j],C_i})
      Break
    Else
      Count = Count + 1
      CL[Count] = C_i
    EndIf
  EndFor
EndFor
Perform k-means clustering initialized with the obtained
   color clusters CL
```

where $\text{Dist}(\text{C}_1, \text{C}_2)$ denotes the distance between the colors $\text{C}_1 = (L_1^*,\, a_1^*,\, b_1^*)^{\text{T}}$ and $\text{C}_2 = (L_2^*,\, a_2^*,\, b_2^*)^{\text{T}}$ given by:

$$\text{Dist}(\text{C}_1,\, \text{C}_2) = \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2} \tag{2}$$

The threshold parameter T_s decides the similarity between the two colors and hence the number of clusters. Antonacopoulos and Karatzas [12] perform grouping of color pixels based on the criterion that only colors that cannot be differentiated by humans should be grouped together. The threshold below which two colors are considered similar was experimentally determined and set to 20. We use a slightly higher threshold to account for the small color variations that may appear within the text strokes. In our implementation, the threshold parameter T_s is empirically fixed at 35, after experimentation.

It may be observed that unlike that of COCOCLUST, where clustering is performed on each pixel, the clustering is carried out at the component-level in this case. Each CC, described by its mean color, is checked for similarity with that of other CCs and they are grouped if the color distance is within a threshold value. Since each CC is represented by a single color (mean color), it cannot be further split and the above clustering process only allows merging of 'similar' CCs that were pre-segmented by the edge detection process. The result of color segmentation is reasonably robust to non-uniform illumination and shadows since edge detection is largely unaffected by these photometric distortions.

By making some sensible assumptions about the document image, obvious non-text CCs are removed by an area-based filtering. CCs whose area is less than 15 pixels or greater than one-fifth of the entire image area are not considered for further processing. Large CCs whose heights or widths are greater than one-third of the image dimension are also filtered out. This heuristic filtering removes a significant amount of noise without affecting any text CC, thereby reducing the computational load. Fig. 2 illustrates the robustness of the proposed method on an example color image where there is a gradual variation in the text color.

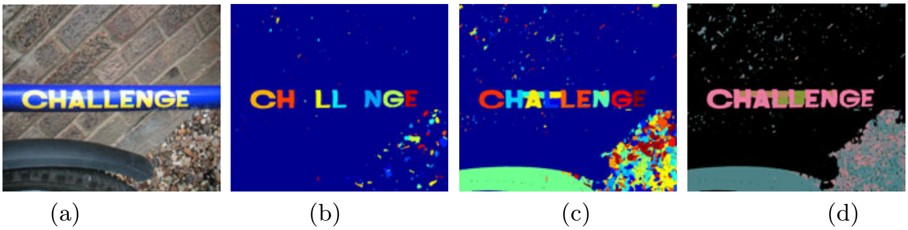


Fig. 2. Robustness of the proposed edge-guided color segmentation method (a) Input image (b) Segmented regions obtained using Canny edge detection (c) Segmented regions obtained after edge-linking (d) Segmented image obtained after color clustering. Note that the outputs shown in (b)-(d) are obtained after the area-based heuristic filtering.

4 Feature Extraction for Text Classification

Each individual color layer is analyzed and text-like CCs are identified employing a set of 12 low-level features derived from the boundary, stroke and gradient-based features.

Geometric Features: Text CCs have geometric properties that can be used to distinguish them from non-text CCs. Their aspect ratios and occupancy ratios tend to cluster around a small range of values. Text CCs are normally much smaller than most of the background clutter. They are also characterized by a small number of convex deficiency regions since text CCs are composed of a few number of strokes. For every CC, we compute the following features:

$$Aspect\,Ratio = \min\left(\frac{W}{H}, \frac{H}{W}\right) \tag{3}$$

$$Area\,Ratio = \frac{Area(CC)}{Area(Input\,Image)} \tag{4}$$

$$Occupancy = \frac{|CC|}{Area(CC)} \tag{5}$$

$$Convex\,Deficiency = \min\left(1, \frac{\#\,Convex\,deficiency}{\alpha}\right) \tag{6}$$

where W and H denote the width and height of the CC being processed and $|\,.\,|$ the number of ON pixels. The parameter α is used to normalize the feature value so that it lies in the range [0,1].

Boundary Features: Text CCs generally have smooth and well-defined boundaries and hence have a higher degree of overlap with the edge image than the non-text components [9]. These characteristics are captured by the following features.

$$Boundary\,Smoothness = \frac{|CC - (CC \circ S_2)|}{|CC|} \tag{7}$$

$$Boundary\,Stability = \frac{|E_{CC} \bigcap \text{Boundary}(CC)|}{|\text{Boundary}(CC)|} \tag{8}$$

Here, E_{CC} denotes the set of edge pixels of the component CC, $\circ$ refers to the morphological opening operation and S_2 a square structuring element of size 2×2.

Stroke Features: Text CCs are also characterized by a uniform stroke width all along the stroke, which is also normally much smaller than its height.

$$SW\ Deviation = \frac{StdDev[StrokeWidth(CC)]}{Mean[StrokeWidth(CC)]} \tag{9}$$

$$SW\,CCHeight\,Ratio = \frac{StrokeWidth(CC)}{H} \tag{10}$$

Assuming that each character is of a uniform color, the original CC and its corresponding binary image B should have a high degree of ‘similarity’ and a small degree of ‘dissimilarity’. Here B is obtained by binarizing the region corresponding to the CC from the gray counterpart of the original color image. The converse holds for non-text regions since they are generally non-homogeneous in

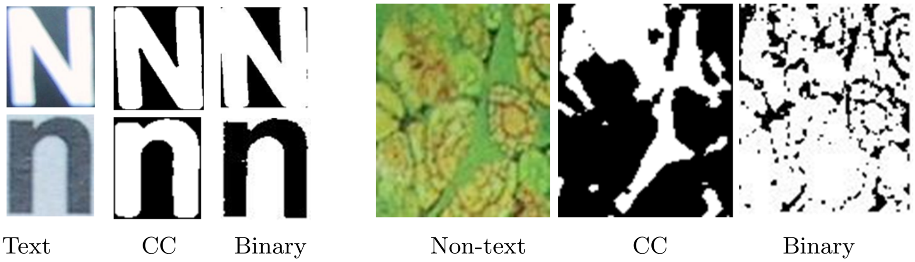


Fig. 3. Illustration of the stroke homogeneity feature. For any text CC, the corresponding binary output is 'similar' to the CC obtained from color segmentation. But, non-text CCs do not exhibit such a property owing to the inhomogeneity.

nature. This characteristic of text is named stroke homogeneity and is computed as follows:

$$\begin{aligned} B &= Binarize(ImagePatch) \qquad (11)\\ \text{If } |CC \cap B| &\geq |CC \cap \text{NOT}(B)|\\ Sim &= |CC \cap B|\\ Dissim &= |\text{XOR}(CC, B)|\\ &\text{Else}\\ Sim &= |CC \cap \text{NOT}(B)|\\ Dissim &= |\text{XOR}(CC, \text{NOT}(B))| \end{aligned}$$

$$Stroke\, Homogeneity = min\left(1, \frac{Dissim}{Sim}\right) \qquad (12)$$

Owing to its simplicity, we choose the block-based Otsu thresholding technique to binarize the gray-scale image patch described by each CC where the image patch is subdivided into 3 × 3 blocks. Due to the presence of inverse text, the binarized outputs may be inverted in some cases. This is illustrated in figure 3. The relation $|CC \cap B| \geq |CC \cap \text{NOT}(B)|$ holds for text brighter than the background. Thus, this test is used in obtaining the 'similarity' and 'dissimilarity' measures appropriately.

Gradient Features: Text regions are characterized by a high density of edges. As pointed out by Clark and Mirmehdi [5], the gradient in text regions exhibit an anti-parallel property. Based on these observations, the following features are computed.

$$Gradient\, Density = \frac{\sum_{(x,y)\in E_{CC}} G(x,y)}{\mid CC \mid} \qquad (13)$$

where $G(x,y)$ denotes the gradient magnitude obtained from the Gaussian derivative of the gray scale image.

$$Gradient\, Symmetry = \frac{\sum_{i=1}^{8}[A(\theta_i) - A(\theta_{i+8})]^2}{\sum_{i=1}^{8} A(\theta_i)^2} \qquad (14)$$

$$Angle\,distribution = \frac{\sum_{i=1}^{8}[A(\theta_i) - \bar{A}]^2}{\sum_{i=1}^{8} A(\theta_i)^2} \tag{15}$$

where $\theta(x,y)$ is the gradient orientation quantized into 16 bins i.e. $\theta_i \in \left[(i-1)\frac{\pi}{8}, \frac{i\pi}{8}\right)$; $i = 1, 2, \cdots, 16$, $A(\theta_i)$ is the magnitude of edges in direction θ_i and $\bar{A}$ is the mean gradient magnitude over all θ_i.

In order to classify the segmented CCs into text and non-text classes, we employ two classifiers namely an SVM and a neural network (NN) classifier trained on the above set of features. A total of 4551 English character CCs and 39599 non-character CCs are extracted from the training images from the ICDAR 2003 robust reading competition dataset [13]. The LIBSVM toolkit [14] is used for implementing the SVM classifier. Radial basis kernel function is used and the optimum parameters for the SVM are determined through a 5-fold validation process. In addition, we use the MATLAB NN Toolbox to implement a two-layer feed-forward neural network with one sigmoidal hidden layer and output neurons. During the testing stage, the input image is first segmented into its constituent CCs which are classified as text or non-text using the two trained classifiers. We declare a test CC as text if it is classified as text by either of the two classifiers.

5 Experiments and Results

The proposed method is tested on our own database of camera-captured images with complex text layouts and multi-script content. Obviously, it is not appropriate to use rectangular word bounding boxes for quantifying the performance of the method as in the ICDAR 2003 text locating competitions. The ICDAR metric is strict and heavily penalizes errors in estimation of word boundaries that drastically affect the detection rate. Pixel-based evaluation of the performance of text detection is preferred since there is no need to consider the number of detected words. Hence, we develop a semi-automatic toolkit [15] for annotating generic scene images, which is available for free download. Using the toolkit, we obtain pixel-accurate ground truth of 100 scenic images containing text in various layout styles and multiple scripts. Figure 4 shows one such ground truth for a multi-script image where the text is laid out in an arc form. The availability of such a groundtruthed data enables us to evaluate the performance using pixel-based precision and recall measures which is also used in the Document Image Binarization Contest 2009.

1. A pixel is classified as true positive (TP) if it is ON in both ground truth and output of text detection.
2. A pixel is classified as false positive (FP) if it is ON only in the output of text detection.
3. A pixel is classified as false negative (FN) if it is ON only in the ground truth image.

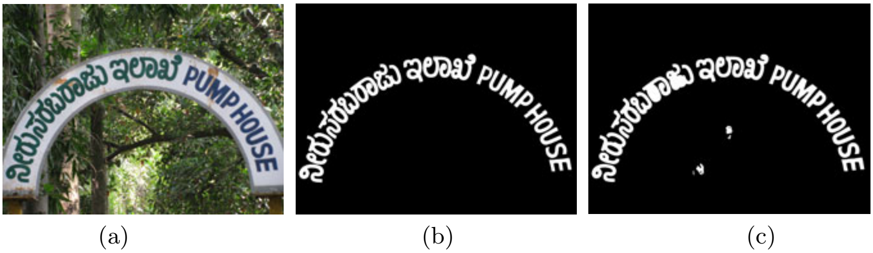


Fig. 4. (a) A sample multi-script image from our database that contains curved text. (b) The corresponding pixel-accurate ground truth (c) Text CCs identified by our method yielding precision and recall value of 0.88 and 0.99 respectively.

The precision and recall measures are then computed as,

$$p = \frac{Number\, of\, TP}{(Number\, of\, FP + Number\, of\, TP)} \tag{16}$$

$$r = \frac{Number\, of\, TP}{(Number\, of\, FN + Number\, of\, TP)} \tag{17}$$

Since the features used for identifying text CCs do not assume any characteristic of the script, the same method can detect text irrespective of the script and text orientation. Figure 5 shows some sample outputs of text detection on our database. Table 1 gives the overall performance of the proposed method, which yields $p = 0.80, r = 0.86$ and $f = 0.83$ respectively. The availability of pixel-accurate ground truth enables us to evaluate the performance of text detection directly without the need to group the text CCs into words.

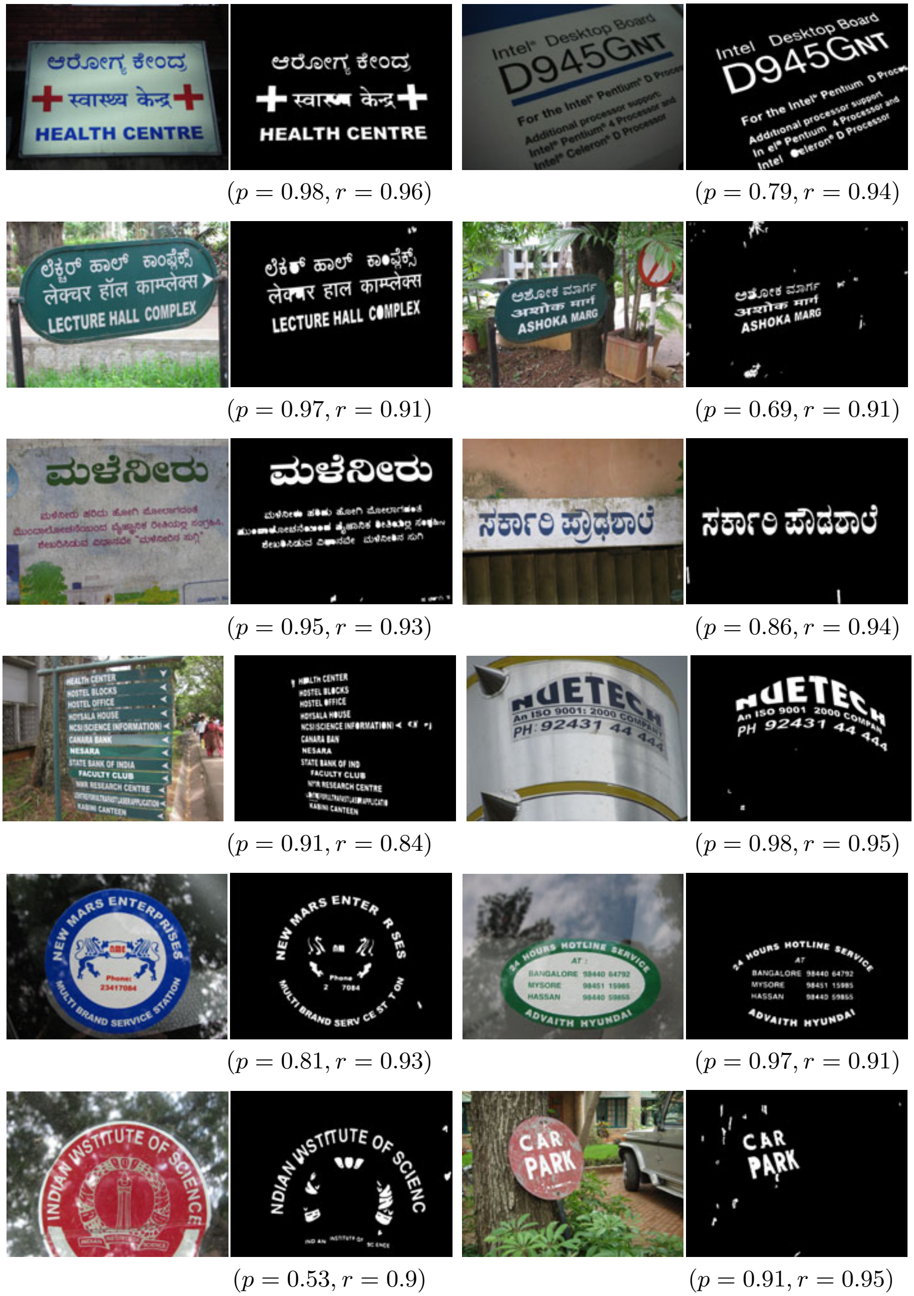
Table 1. Overall result of text localization on our dataset

Precision	**Recall**	**f**
0.8	0.86	0.83

5.1 Estimation of Word Bounding Boxes

In order to make a fair comparison with the results of other existing methods in the literature, we also use the ICDAR 2003 evaluation metrics computed from rectangle-based area matching score to compute the precision and recall. This requires grouping of the detected text CCs into words and obtaining its bounding rectangles for quantifying the result of text detection.

Since the ICDAR dataset contains only horizontally aligned text, we employ Delaunay triangulation to link adjacent CCs together and obtain those CCs that lie in a straight line using simple heuristics such as position, height and area. A link map, $V = \bigcup v_{ij}$ is created, where v_{ij} is an edge of a triangle linking CC_i



($p = 0.98, r = 0.96$) ($p = 0.79, r = 0.94$)

($p = 0.97, r = 0.91$) ($p = 0.69, r = 0.91$)

($p = 0.95, r = 0.93$) ($p = 0.86, r = 0.94$)

($p = 0.91, r = 0.84$) ($p = 0.98, r = 0.95$)

($p = 0.81, r = 0.93$) ($p = 0.97, r = 0.91$)

($p = 0.53, r = 0.9$) ($p = 0.91, r = 0.95$)

Fig. 5. Representative results of text localization on images with multi-script content and arbitrary orientations. The pixel-based precision and recall measures are indicated below each image.

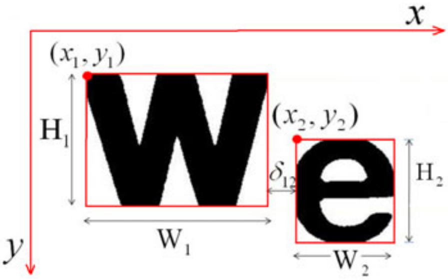


Fig. 6. The parameters that guide the process of grouping of adjacent CCs into text lines

to CC_j obtained by applying Delaunay triangulation to the classified CCs. In the first step, the links are filtered by assigning labels based on height ratio and vertical overlap ratio:

$$\Phi(v_{ij}) = \begin{cases} +1, & [0.5 < (H_i/H_j) < 2] \wedge \\ & [(Y_i + H_i) - Y_j > 0] \wedge \\ & [(Y_j + H_j) - Y_i > 0] \\ -1, & \text{otherwise} \end{cases} \tag{18}$$

where (X_i, Y_i, W_i, H_i) are the attributes of bounding box (See Fig. 6).

The links labeled -1 are filtered out and the remaining links between the characters are used to generate text lines. Text lines containing less than 3 CCs are eliminated since a text line usually contains more than 2 characters. This criterion helps in eliminating false positives. To handle isolated characters, we mark all components with high posteriors for text during the classification step and accept only those CCs whose likelihood exceeds 0.9.

We make use of the spatial regularity in the occurrence of characters in a text line to recover false negatives. A straight line $F(x) = ax + b$ is fitted to the coordinates of the centroids $\{C_{xi}, C_{yi}\}$ of the bounding boxes of all CCs in a text group. A component CC'_k, whose bounding box center (C'_{xk}, C'_{yk}) lies inside the text line, is re-classified as text component if the following criteria are satisfied:

$$\Lambda' < mean(\Lambda) + \beta H_{line} \tag{19}$$

$$(\bar{A}_{CC}/4) < Area(CC_k) < 2\bar{A}_{CC} \tag{20}$$

where $\Lambda = abs(F(X_i) - Y_i)$, $\Lambda' = abs(F(X'_k) - Y'_k)$, β is a control parameter empirically set to 0.2, H_{line} is the height of bounding rectangle of text line and $\bar{A}_{CC}$ is the mean area of all CCs in the string. As illustrated in Fig. 7, this grouping procedure enables the recovery of CCs that were misclassified by the classifiers thereby increasing the performance of the method.

The inter bounding box distance is used to cut each text line into words. The distance between adjacent components CC_j and CC_i is calculated as follows:

$$\delta_{ji} = abs(X^i - (X^j + W^j)) \tag{21}$$

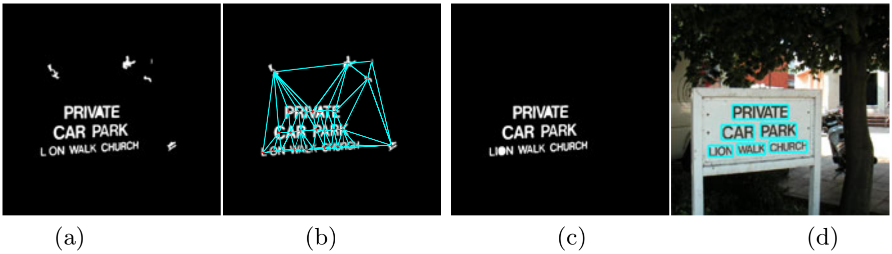


Fig. 7. Grouping of verified CCs into words and recovery of false negatives using the spatial coherence property of text. Notice that the character 'I' in the last line, which was initially misclassified as non-text, is re-classified as text during the grouping stage since its size is comparable to that of the adjacent text CCs.

The text line is cut wherever the following condition occurs:

$$\delta_{ji} > \gamma \times mean(\{\delta_{ji}\}) \tag{22}$$

where γ is a control parameter empirically set to 2.65. Each segment of the text line is considered as a word and its bounding rectangle is computed.

Table 2 lists the performance of our method and compares it with other existing methods. The overall precision, recall and f-measures obtained on the whole test set are 0.63, 0.59 and 0.61 respectively. The performance of our method is comparable to that of the best-performing methods in the ICDAR 2005 text locating competition [16]. It performs marginally worse than a recent method proposed by Epshtein et al. [19], which is designed for locating only horizontal text. Our method, however, works well for generic scene images having arbitrary text orientations.

Table 2. Comparison of the proposed method with performance of other techniques on ICDAR dataset

Method	Precision	Recall	f
ICDAR 2005			
Hinnerk Becker	0.62	0.67	0.62
Chen and Yuille	0.6	0.6	0.58
Recent methods			
Nuemann and Matas [17]	0.59	0.55	0.57
Minetto et al. [18]	0.63	0.61	0.61
Epshtein et al. [19]	0.73	0.6	0.66
Proposed method	**0.63**	**0.59**	**0.61**

6 Conclusions

This paper describes an important preprocessing for scene text analysis and recognition. CC-based approaches for text detection are known for their robustness to large variations in font styles, sizes, color, layout and multi-script content and therefore suitable for processing camera-based images. However, extraction of CCs from complex background is not a trivial task. To this end, we introduce a edge-guided color clustering technique suitable for extracting text CCs from complex backgrounds. We also design a set of 'intrinsic' features of text for classifying each of the identified CC as text or non-text to achieve layout and script independence. The system's performance is enhanced by invoking the spatial regularity property of text that effectively filters out inconsistent CCs which also aid in the recovery of missed text CCs. Experiments on camera-captured images that contain variable font, size, color, irregular layout, non-uniform illumination and multiple scripts illustrate the robustness of the method.

References

1. Zhong, Y., Karu, K., Jain, A.K.: Locating text in complex color images. Patt. Recog. 28(10), 1523–1535 (1995)
2. Li, H., Doermann, D., Kia, O.: Automatic Text Detection and Tracking in Digital Video. IEEE Trans. Image Proc. 9(1), 147–156 (2000)
3. Raju, S.S., Pati, P.B., Ramakrishnan, A.G.: Gabor Filter Based Block Energy Analysis for Text Extraction from Digital Document Images. In: Proc. Intl. Workshop DIAL, pp. 233–243 (2004)
4. Wu, V., Manmatha, R., Riseman, E.M.: TextFinder: an automatic system to detect and recognize text in images. IEEE Trans. PAMI 21(11), 1124–1129 (1999)
5. Clark, P., Mirmehdi, M.: Finding text using localised measures. In: Proc. British Machine Vision Conf., pp. 675–684 (2000)
6. Chen, X., Yuille, A.L.: Detecting and Reading Text in Natural Scenes. In: Proc. IEEE Intl. Conf. CVPR, vol. 2, pp. 366–373 (2004)
7. Shivakumara, P., Dutta, A., Tan, C.L., Pal, U.: A New Wavelet-Median-Moment based Method for Multi-Oriented Video Text Detection. In: Proc. Intl. Workshop on Document Analysis and Systems, pp. 279–286 (2010)
8. Gatos, B., Pratikakis, I., Kepene, K., Perantonis, S.J.: Text detection in indoor/outdoor scene images. In: Proc. Intl. Workshop CBDAR, pp. 127–132 (2005)
9. Zhu, K., Qi, F., Jiang, R., Xu, L., Kimachi, M., Wu, Y., Aizawa, T.: Using Adaboost to Detect and Segment Characters from Natural Scenes. In: Proc. Intl. Workshop CBDAR, pp. 52–59 (2005)
10. Pan, W., Brui, T.D., Suen, C.Y.: Text Detection from Scene Images Using Sparse Representation. In: Proc. ICPR, pp. 1–5 (2008)
11. Kasar, T., Ramakrishnan, A.G.: COCOCLUST: Contour-based Color Clustering for Robust Binarization of Colored Text. In: Proc. Intl. Workshop CBDAR, pp. 11–17 (2009)
12. Antonacopoulos, A., Karatzas, D.: Fuzzy Segmentation of Characters in Web Images Based on Human Colour Perception. In: Lopresti, D.P., Hu, J., Kashi, R.S. (eds.) DAS 2002. LNCS, vol. 2423, pp. 295–306. Springer, Heidelberg (2002)
13. ICDAR Robust reading competition data set (2003), http://algoval.essex.ac.uk/icdar/Competitions.html
14. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines, http://www.csie.ntu.edu.tw/cjlin/libsvm
15. Kasar, T., Kumar, D., Prasad, A.N., Girish, D., Ramakrishnan, A.G.: MAST: Multi-scipt Annotation Toolkit for Scenic Text. In: Joint Workshop on MOCR and AND, pp. 113–120 (2011), software http://mile.ee.iisc.ernet.in/mast

16. Lucas, S.M.: ICDAR 2005 Text Locating Competition Results. In: Proc. ICDAR, pp. 80–84 (2005)
17. Neumann, L., Matas, J.: A Method for Text Localization and Recognition in Real-World Images. In: Kimmel, R., Klette, R., Sugimoto, A. (eds.) ACCV 2010, Part III. LNCS, vol. 6494, pp. 770–783. Springer, Heidelberg (2011)
18. Minetto, R., Thome, N., Cord, M., Fabrizio, J., Marcotegui, B.: SNOOPERTEXT: A multiresolution system for text detection in complex visual scenes. In: Proc. IEEE ICIP, pp. 3861–3864 (2010)
19. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: Proc. IEEE Conf. CVPR, pp. 2963–2970 (2010)

Assistive Text Reading from Complex Background for Blind Persons

Chucai Yi[1,2] and Yingli Tian[1,2]

[1] Media Lab, Dept. of Electrical Engeering, The City College of New York, City Univ. of New York, 160 Convent Avenue, New York, NY, USA, 10031
[2] Dept. of Computer Science, The Graduate Center, City Univ. of New York, 365 Fifth Avenue, New York, NY, USA, 10016
cyi@gc.cuny.edu, ytian@ccny.cuny.edu

Abstract. In the paper, we propose a camera-based assistive system for visually impaired or blind persons to read text from signage and objects that are held in the hand. The system is able to read text from complex backgrounds and then communicate this information aurally. To localize text regions in images with complex backgrounds, we design a novel text localization algorithm by learning gradient features of stroke orientations and distributions of edge pixels in an Adaboost model. Text characters in the localized regions are recognized by off-the-shelf optical character recognition (OCR) software and transformed into speech outputs. The performance of the proposed system is evaluated on ICDAR 2003 Robust Reading Dataset. Experimental results demonstrate that our algorithm outperforms previous algorithms on some measures. Our prototype system was further evaluated on a dataset collected by 10 blind persons, with the system effectively reading text from complex backgrounds.

Keywords: blind person, assistive text reading, text region, stroke orientation, distribution of edge pixels, OCR.

1 Introduction

According to the statistics in 2002 [14], more than 161 million persons suffer visual impairment, in which there are 37 million blind persons. It is a challenging task for blind persons to find their way in unfamiliar environments, for example, independently finding the room they are looking for. Many aid systems have been developed to help blind persons avoid obstacles in all kinds of environments [3]. Some indoor positioning systems modeled global layout of a specific zone and used radio wave analysis to locate the persons wearing signal transceivers [13]. Some systems employed Quick Response (QR) codes to guide blind persons to destinations. However, most of these systems require pre-installed devices or pre-marked QR codes. Also, the blind user needs to consider compatibility of different systems. Therefore, the above systems cannot provide blind users these services in environments without pre-installed devices or markers. However, most blind persons can find nearby walls and doors, where text signage is always placed to indicate the

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 15–28, 2012.

room number and function. Thus blind persons will be well navigated if a system can tell them what the nearby text signage says. Blind persons will also encounter trouble in distinguishing objects when shopping. They can receive limited hints of an object from its shape and material by touch and smell, but miss descriptive labels printed on the object. Some reading-assistive systems, such as voice pen, might be employed in this situation. They integrate OCR software to offer the function of scanning and recognition of text for helping blind persons read print documents and books. However, these systems are generally designed for scanned document images with simple background and well-organized characters rather than packing box with multiple decorative patterns. The OCR software cannot directly handle the scene images with complex backgrounds. Thus these assistive text reading systems usually require manual localization of text regions in a fixed and planar object surface, such as a screen and book.

To more conveniently assist blind persons in reading text from nearby signage or objects held in the hand, we design a camera-based assistive text reading system to extract significant text information from objects with complex backgrounds and multiple text patterns. The tasks of our system are indoor object detection to find out nearby wall, door, elevator or signage, and text extraction to read the involved text information from complex backgrounds. This paper focuses only on the step of text extraction, including 1) text localization to obtain image regions containing text, and 2) text recognition to transform image-based information into text codes [20]. Fig. 1 illustrates two examples of our proposed assistive text reading system. In order to perform text recognition by off-the-shelf OCR software, text regions must be detected and binarized. However, the problem of automatic localization of text regions from camera captured images with complex backgrounds has not been solved. For our application, text in camera captured images is most likely surrounded by various background outliers, and text characters usually appear in multiple scales, fonts, colors, and orientations. In this paper, we propose a novel algorithm of text localization based on gradient features of stroke orientations and distributions of edge pixels.

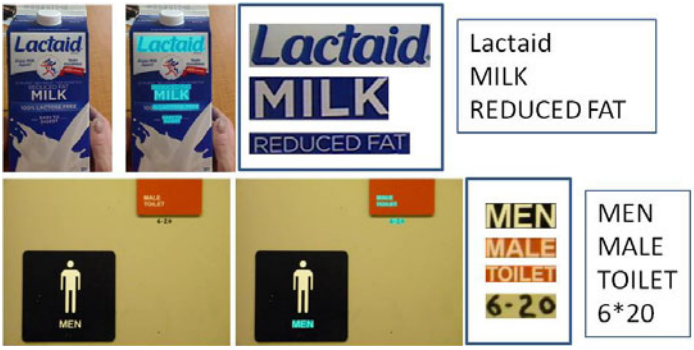


Fig. 1. Two examples of text localization and recognition from camera captured images. Top: a milk box; Bottom: a male bathroom. From left to right: camera-captured images, localized text regions (marked in cyan), text regions, and text codes recognized by OCR.

1.1 Previous Work in Text Localization

Many algorithms were presented to localize text regions in scene images. We divide them into two categories. The first category are rule-based algorithms that applied pixel level image processing to extract text information by predefined text features such as character size, aspect ratio, edge density, character structure, and color uniformity of text string, etc. Phan et al. [12] modeled edge pixel density by Laplacian operator and maximum gradient difference to calculate text regions. Shivakumara et al. [17] used gradient difference map and global binarization to obtain text regions. Epshtein et al. [4] used the consistency of text stroke width and defined stroke width transform to localize text characters. Nikolaou et al. [10] applied color reduction to extract text in uniform colors. This type of algorithms tried to define a universal feature descriptor of text. In [2], color based text segmentation is performed through a Gaussian mixture model for calculating confidence value of text regions. The second category are learning-based algorithms that apply explicit machine learning models on feature maps of training samples to extract robust text features and build text classifiers. Chen et al. [1] presented 5 types of block patterns on intensity based and gradient based feature maps to train classifiers in Adaboost learning model. Kim et al. [6] considered text as specific texture and analyzed the textural features of characters by support vector machine (SVM) model. Kumar et al. [7] used the responses from Globally Matched Wavelet (GMW) filters of text as features and applied SVM and Fisher classifier for image window classification. Ma et al. [9] performed classification of text edges by using HOG and LBP as local features on the SVM model. Shi et al. [16] used gradient and curvature features to model the gray scale curve for handwritten numeral recognition under a Bayes discriminate function. In this paper, we propose a text localization algorithm by defining novel feature maps based on stroke orientations and edge distributions.

2 System and Algorithm Overview

Our prototype system is equipped with a wearable camera attached to a cap or pair of sunglasses, an audio output device such as Bluetooth or earphones, and a mini-microphone for user speech input. This simple hardware structure ensures the portability of the system. A wearable computer/PDA provides the platform for information processing.

Fig. 2 depicts the main components of the prototype system. Blind users wearing cameras capture signage and objects they are facing. The camera captured images are then processed by our novel proposed text localization algorithm to detect text regions. In this text localization method, the basic processing cells are rectangle image patches with fixed aspect ratio, where features of text are extracted from both stroke orientations and edge distributions. In the training process, a feature matrix from the training set is formed as the input of an Adaboost machine learning algorithm to build a text region classifier. In the testing process, an adjacent character grouping algorithm is first applied on camera captured natural scene images to preliminarily localize the candidate image patches [19]. The classifier learned from Adaboost algorithm is employed to classify the text or non-text patches, where neighboring text patches are merged into text regions. Then off-the-shelf OCR software is employed to perform text recognition in the localized text regions. The recognized words are transformed into speech for blind users.

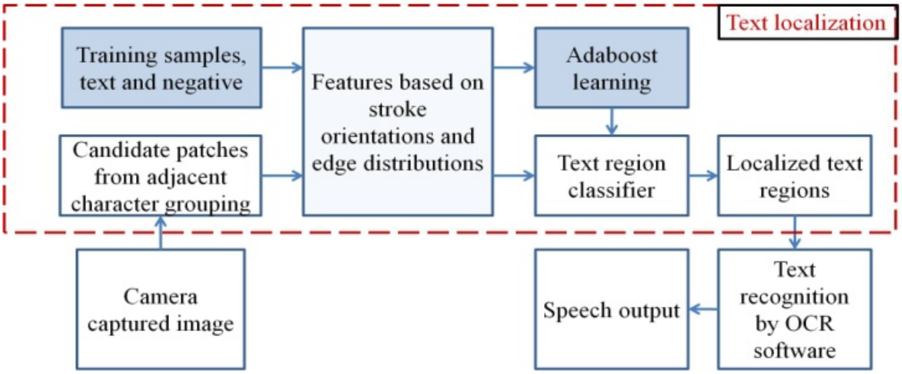

Text localization
Training samples, text and negative
Features based on stroke orientations and edge distributions
Adaboost learning
Candidate patches from adjacent character grouping
Text region classifier
Localized text regions
Camera captured image
Speech output
Text recognition by OCR software

Fig. 2. Flowchart of our system

The main contributions of this paper include: (1) a novel algorithm of automatic text localization to extract text regions from complex background and multiple text patterns; (2) a camera-based assistive prototype system to aid blind persons reading text from signage in unfamiliar environments and other objects; and (3) a dataset of objects and signage captured by blind persons for assistive text reading system evaluations.

3 Automatic Text Localization

We design a learning based algorithm of automatic text localization. In order to handle complex backgrounds, we propose two novel feature maps to extracts text features based on stroke orientation and edge distribution respectively. Here stroke is defined as a uniform region with bounded width and extensive length. These feature maps are combined to build an Adaboost-based text classifier.

3.1 Text Stroke Orientation

Text characters consist of strokes in different orientations as the basic structure. Here, we propose a new type of features, stroke orientations, to describe the local structure of text characters. From the pixel-level analysis, stroke orientation is perpendicular to the gradient orientations at pixels of stroke boundaries, as shown in Fig. 3. To model the text structure by stroke orientations, we propose a new operator to map gradient feature of strokes to each pixel. It extends local structure of stroke boundary into its neighborhood by gradient orientations. It provides a feature map to analyze global structures of text characters.

Given an image patch I, Sobel operators in horizontal and vertical derivatives are used to calculate 2 gradient maps G_x and G_y respectively. The synthesized gradient map is calculated as $G = \left({G_x}^2 + {G_y}^2\right)^{1/2}$. Canny edge detector is applied on I to calculate its binary edge map E. For a pixel p_0, a circular range is set as $R(p_0) = \{p|d(p,p_0) \leq 36\}$, where $\mathrm{d}(.)$ is set as Euclidean distance. In this range we find out the edge pixel p_e with the minimum Euclidean distance from p_0. Then the pixel p_0 is labeled with gradient orientation at the pixel p_e from gradient maps by (1), where $P = \{p|p \in R(p_0), p\ is\ edge\ pixel\}$ and Υ normalizes stroke orientation into the range $(3\pi/2\,,5\pi/2]$, which shifts forward one period from $(-\pi/2\,,\pi/2]$ to avoid the value 0, because $\mathrm{S}(\mathrm{p}_0)$ is set as 0 if and only if no edge pixel is found out within the range of pixel p_0.

$$p_e = \underset{p\in P}{\operatorname{argmin}}\, d(p,p_0)$$
$$S(p_0) = \Upsilon\left(\arctan\left(\, G_y(p_e),\ G_x(p_e)\right)\right) \tag{1}$$

A stroke orientation map $S(p)$ is output by assigning each pixel the gradient orientation at its nearest edge pixel, as shown in Fig. 4(a). The pixel values in stroke orientation map are then quantized into an N bin histogram in the domain $(3\pi/2\,,5\pi/2]$ (see Fig. 4(b)). In feature extraction, strokes with identical or similar orientations are employed to describe structure of text from one perspective. In the N bin histogram, we group the pixels at every d consecutive bins together to generate a multi-layer stroke orientation map, where strokes in different orientations are separated into different layers. Without considering the cyclic shifts of the bins, there are totally $N - d + 1$ layers. In our evaluation, d is set to be 3 and N is set to be 16 respectively. Thus each sample generates 14 layers of stroke orientation maps, where text structure is described as gradient features of stroke orientations. We can extract structural features of text from stroke orientation maps.

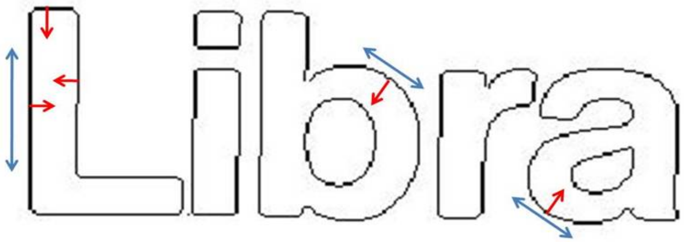


Fig. 3. An example of text strokes and relationship between stroke orientations and gradient orientations at pixels of stroke boundaries. Blue arrows denote the stroke orientations at the sections and red arrows denote the gradient orientations at pixels of stroke boundaries.

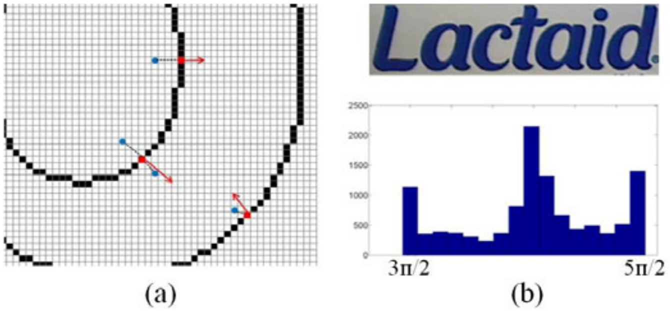


Fig. 4. (a) An example of stroke orientation label. The pixels denoted by blue points are assigned the gradient orientations (red arrows) at their nearest edge pixels, denoted by the red points. (b) A 210 ×54 text patch and its 16-bin histogram of quantized stroke orientations.

3.2 Distributions of Edge Pixels

In an edge map, text character appears in the form of stroke boundaries. Distribution of edge pixels in stroke boundaries also describes the characteristic structure of text. The most commonly used feature is edge density of text region. But edge density measure does not give any spatial information of edge pixels. It is generally used for distinguishing text regions from relatively clean background regions. To model text structure by spatial distribution of edge pixels, we propose an operator to map each pixel of an image patch into the number of edge pixels in its cross neighborhood. At first, edge detection is performed to obtain an edge map, and the number of edge pixels in each row y and each column x is calculated as N_R (y) and N_C (x). Then each pixel is labeled with the product value of the number of edge pixels in its located row and that in its located column. Based on this transform, the feature map of edge distribution is calculated by assigning each pixel weighed sum of the neighborhood centered at it, as (2). In the feature map of edge distribution, pixel value reflects edge density of its located region.

$$D(x,y) = \sum_{n} w_n \cdot N_R(y_n) \cdot N_C(x_n) \tag{2}$$

where (x_n, y_n) is neighboring pixel of (x, y) and w_n denotes the weight value.

3.3 Adaboost Learning of Features of Text

Based on the feature maps of gradient, stroke orientation and edge distribution, a classifier of text is trained from Adaboost learning model. Image patches with fixed size (height 48 pixels, width 96 pixels) are collected from images of ICDAR 2011 robust reading competition [21] to generate a training set for learning features of text. We generate positive training samples by scaling or slicing the ground truth text regions, according to the ratio of width w to height h. If the ratio is $w/h < 0.8$, the region is discarded. If the ratio w/h falls in [0.8,2.5), the ground truth region is scaled to a window of width-to-height ratio $2:1$. If the ratio is $w/h \geq 2.5$, we slice this ground truth region into overlapped training samples with width-to-height ratio 2:1. Then they are scaled into width 96 and height 48 pixels. The negative training samples are generated by extracting the image regions containing edge boundaries of non-text objects. These regions also have width-to-height ratio 2:1, and then we scale them into width 96 and height 48. In this training set, there are total 15301 positive samples and each contains several text characters with compatible accommodation of image patch, and 35933 negative samples without containing any text information for learning features of background outliers. Some training examples are shown in Fig. 5.

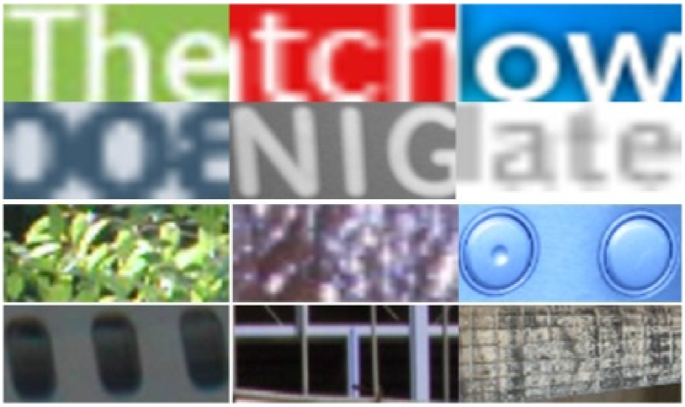


Fig. 5. Examples of training samples with width-to-height ratio 2:1. The first two rows present positive samples and the remaining two rows present negative samples.

To train the classifier, we extract 3 gradient maps, 14 stroke orientation maps, and 1 edge distribution map for each training sample. We apply 6 block patterns [1] on these feature maps of training samples. As shown in Fig. 6, these block patterns are involved in the gradient distributions of text in horizontal and vertical directions. We normalize the block pattern into the same size (height 48 pixels, width 96 pixels) as training samples and derive a feature response f of a training sample by calculating the absolute difference between the sum of pixel values in white regions and the sum of pixel values in black regions. For the block patterns with more than 2 sub-regions (see Fig. 6(a-d)), the other metric of feature response is the absolute difference between the mean of pixel values in white regions and the mean of pixel values in black regions. Thus we obtain $6 + (6 - 2) = 10$ feature values through the 6 block patterns and 2 metrics from each feature map. The "integral image" algorithm is used in these calculations [18]. From the 18 feature maps (3 gradient maps, 14 stroke orientation maps, and 1 edge distribution map), a training sample can generate a feature vector of 180 dimensions as (3). Then we compute feature vectors for all the 51234 samples in training set. By using the feature vector $\boldsymbol{f}^i$ of the i-th sample as the i-th column, a feature matrix $\boldsymbol{F}$ is obtained by (4).

$$\boldsymbol{f}^i = \left[f_1^i, f_2^i, \dots, f_{180}^i\right]^T \quad (3)$$

$$\boldsymbol{F} = [\boldsymbol{f}^1, \boldsymbol{f}^2, \dots, \boldsymbol{f}^t, \dots, \boldsymbol{f}^{51234}] \quad (4)$$

The 180×51234 feature matrix is used for learning a text classifier in cascade Adaboost model. A row of the feature matrix records feature responses of a certain block pattern and a certain feature map on all training samples. In the process of Adaboost learning, weak classifier is defined as $\langle r, T_r, \rho \rangle$. The three parameters denote the r-th row of feature matrix $(1 \le r \le 180)$, a threshold of the r-th row T_r, and polarity of the threshold $\rho \in \{-1,1\}$. In each row r, linearly spaced threshold values are sampled in the domain of its feature values by (5).

$$T_r \in \left\{T \middle| T = f_r^{min} + \frac{1}{N_T}(f_r^{max} - f_r^{min})t\right\} \quad (5)$$

where N_T represents the number of thresholds, f_r^{min} and f_r^{max} represent the minimum and maximum feature value of the r-th row, and t is an integer ranging from 1 to N_T. We set $N_T = 300$ in the learning process. Thus there are in total $180 \times 2 \times 300 = 108000$ weak classifiers. When a weak classifier $\langle r, \rho, T_r \rangle$ is applied on a sample with corresponding feature vector $\boldsymbol{f} = [f_1, \dots, f_r, \dots, f_{180}]^{\boldsymbol{T}}$, if $\rho f_r \ge \rho T_r$, it is classified as positive samples, otherwise it is classified as negative samples.

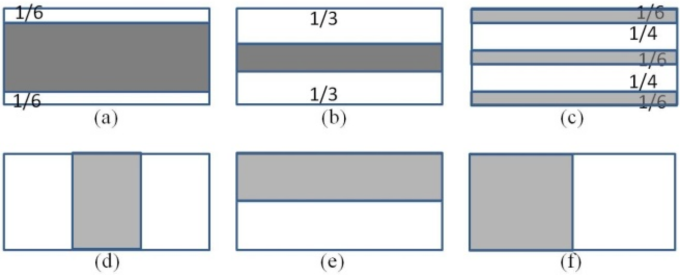


Fig. 6. Block patterns based on [1]. Features are obtained by the absolute value of sum (or mean) of pixel values in white regions minus sum (or mean) of pixel values in black regions.

Cascade Adaboost classifiers proved to be an effective machine learning algorithm in real-time face detection [18]. The training process is divided into several stages. In each stage, based on the feature matrix of all positive samples and the negative samples that are incorrectly classified in previous stages, Adaboost model [5] performs an iterative selection of weak classifiers. The selected weak classifiers are integrated into a strong classifier by weighted combination. The iteration of a stage stops when 99.5% of positive samples are correctly classified while 50% of negative samples are correctly classified by the current strong classifier. The strong classifiers from all stages are cascaded into the final classifier. When a testing image patch is input into the final classifier, it is classified as text patches if all the cascaded strong classifiers determine it is a positive sample, otherwise it is classified as a non-text patch.

3.4 Text Region Localization

Text localization is then performed on the camera captured image. Cascade Adaboost classifier cannot handle the whole image, so heuristic layout analysis is performed to extract candidate image patches prepared for text classification. Text information in the image usually appears in the form of text strings containing no less than three character members. Therefore adjacent character grouping [19] is used to calculate the image patches that possibly contain fragments of text strings. These fragments consist of three or more neighboring edge boundaries which have approximately equal heights and stay in horizontal alignment, as shown in Fig. 7. But not all the satisfied neighboring edge boundaries are text string fragments. Thus the classifier is applied to the image patches to determine whether they contain text or not. Finally, overlapped text patches are merged into a text region, which is the minimum rectangle area circumscribing the text patches. The text string fragments inside those patches are assembled into informative words.

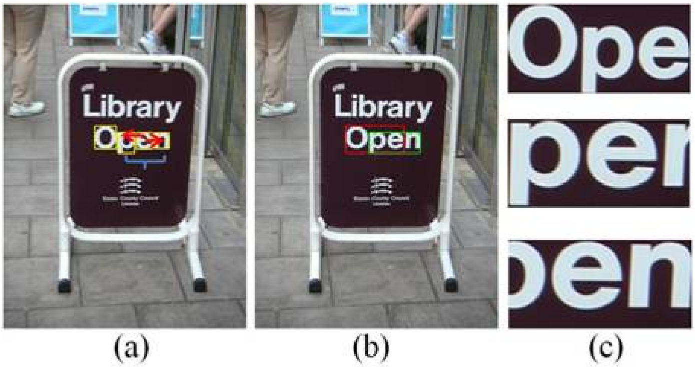


Fig. 7. (a) Character 'e' have adjacent siblings 'p' on the left and 'n' on the right. (b) Adjacent characters are grouped together to obtain two fragments of text strings. (c) Candidate image patches after scaling and slicing, prepared for classification.

4 Text Recognition and Audio Output

Text recognition is performed by off-the-shelf OCR to output the informative words from the localized text regions. A text region labels the minimum rectangular area for the accommodation of characters inside it, so the border of the text region contacts the edge boundary of the text character. However, experiments show that OCR generates better performance if text regions are assigned proper margin areas and binarized to segment text characters from background. Thus each localized text region is enlarged by enhancing the height and width by 10 pixels respectively, and then we use Otsu' method [11] to perform binarization of text regions, where margin areas are always considered as background.

We evaluate two OCR engines, *Tesseract* and *Nuance OmniPage*, on the localized text regions. *OmniPage* shows better performance in most cases, but it is commercial software without open source codes. *Tesseract* is an open-source OCR engine that can be more conveniently integrated into our system.

The recognized text codes are recorded in script files. Then we use Microsoft Speech SDK to load these files and display the audio output of text information. Blind users can adjust speech rate, volume and tone according to their requirements.

5 Experiments

5.1 Datasets

Two datasets are used in our experiments. First, the ICDAR 2003 Robust Reading Dataset is used to evaluate the proposed localization algorithm separately. It contains 509 natural scene images in total. Most images contain indoor or outdoor text signage. The image resolutions range from 640×480 to 1600×1200. Since layout analysis based on adjacent character grouping can only handle text strings with three or more character members, we omit the images containing only ground truth text regions of less than 3 text characters. Thus 488 images are selected from this dataset as testing images to evaluate our localization algorithm.

To evaluate the whole system and develop a user friendly interface, we recruit 10 blind persons to build a dataset of reading text on hand-held objects. They wear a camera attached to a pair of sunglasses and capture the image of the objects in his/her hand, as shown in Fig. 8. The resolution of captured image is 960×720. There are 14 testing objects for each person, including grocery boxes, medicine bottles, books, etc. They are required to rotate each object several times to ensure that surfaces with text captions are captured. These objects are exposed to background outliers and illumination changes. We extract 116 captured images and label 312 text regions of main titles manually.

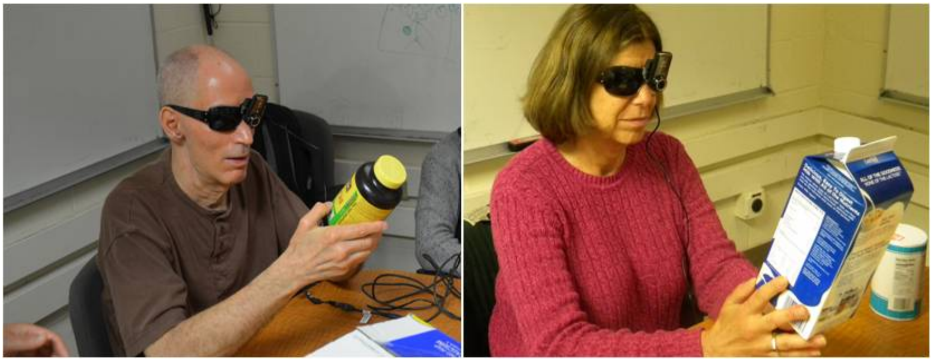


Fig. 8. Blind persons are capturing images of the object in their hands

5.2 Results and Discussions

A localization algorithm is performed on the scene images of Robust Reading Dataset to calculate image regions containing text information. Fig. 9 and Fig. 10(a) depict some results of localized text regions, marked by cyan rectangle boxes. To analyze the accuracy of the localized text regions, we compare them with ground truth text regions by the measures *precision*, *recall* and *f-measure.* For a pair of text regions, match sore is estimated by the ratio between the intersection area and the united mean area of the two regions. Each localized (ground truth) text region generates maximum match score from its best matched ground truth (localized) text region. *Precision* is the ratio between the total match score and the total number of localized regions. It estimates the false positive localized regions. *Recall* is the ratio between the total match score and the total number of ground truth regions. It estimates the missing text regions. *f*-measure combines *precision* and *recall* by harmonic sum. The evaluation results are calculated from average measures on all testing images, which are precision 0.69, recall 0.56, and *f*-measure 0.60. The results are comparable to previous algorithms as shown in Table I. Average processing time on original image resolution is 10.36s. To improve the computation speed, we downsample the testing images into lower resolutions while ensuring that the degradation does not significantly influence the performance. Both the width and the height of downsampled testing image do not exceed 920. Then we repeat the evaluation and obtain precision 0.68, recall 0.54, *f*-measure 0.58, and average process time 1.54s.

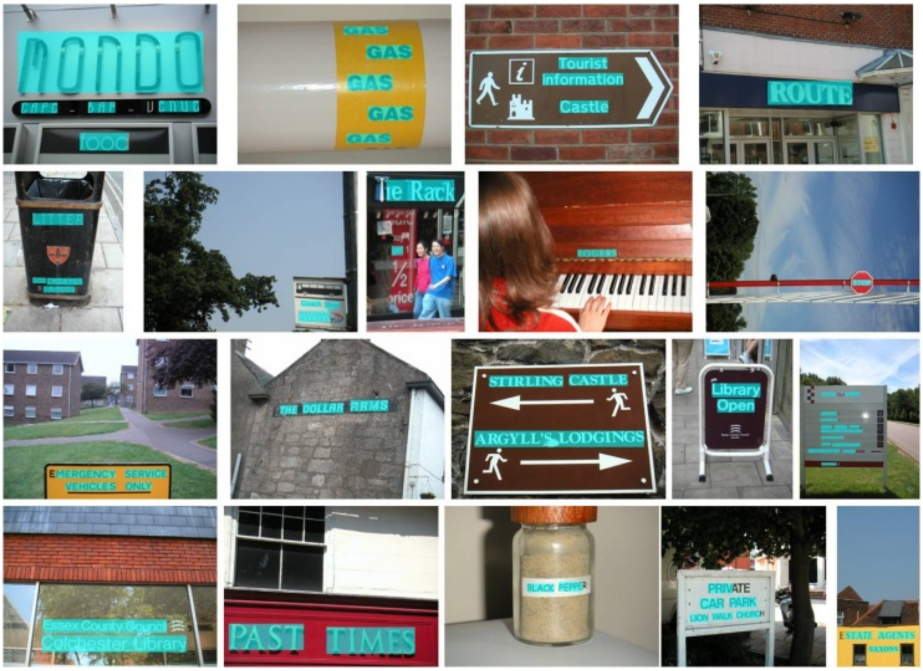


Fig. 9. Some example results of text localization on the robust reading dataset, and the localized text regions are marked in cyan

To evaluate the proposed features of text based on stroke orientations and edge distributions, we can make a comparison with Alex Chen's algorithm [1, 8] because it applies similar block patterns and a similar learning model, but with different feature maps, which are generated from intensities, gradients and joint histograms of intensity and gradient. The evaluation results of Chen's algorithm on the same dataset is precision 0.60, recall 0.60, and *f*-measure 0.58 (Table 1). This demonstrates that our proposed feature maps of stroke orientation and edge distribution give better performance on precision and *f*-measure.

Table 1. The performance comparison between our algorithm and the algorithms presented in [8] on Robust Reading Dataset

Method	Precision	Recall	*f*	time/s
Ours	0.69	0.56	0.60	10.36
Ours(downsample)	0.68	0.54	0.58	1.54
HinnerkBecker	0.62	0.67	0.62	14.4
AlexChen	0.60	0.60	0.58	0.35
Ashida	0.55	0.46	0.50	8.7
HWDavid	0.44	0.46	0.45	0.3

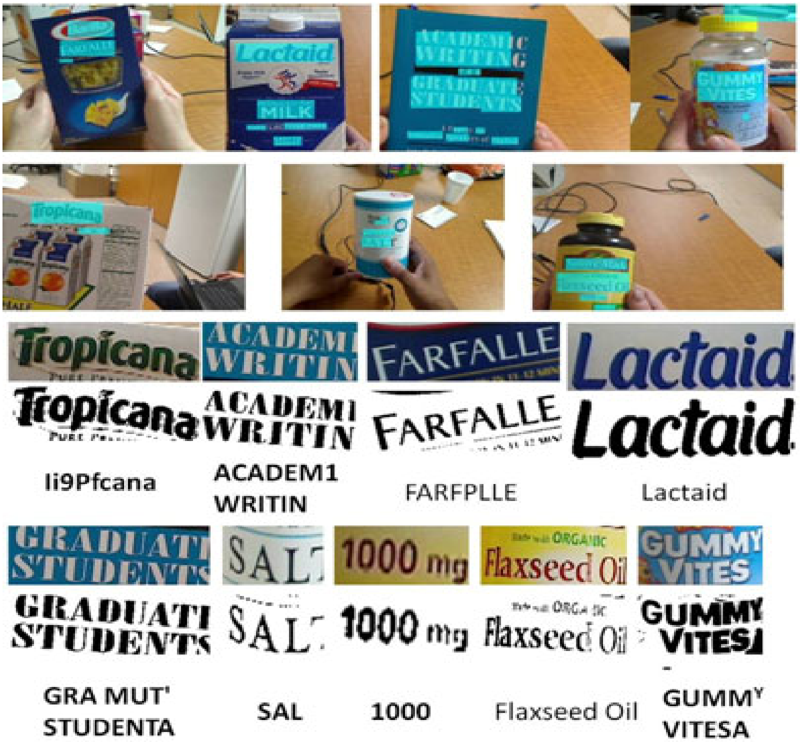


Fig. 10. The top two rows present some results of text localization on the blind-captured dataset, where localized text regions are marked in cyan. The bottom rows show two groups of enlarged text regions, binarized text regions and word recognition results from top to down.

Further, our system is evaluated on the blind-captured dataset of object text. We define that a ground truth region is hit if its three-quarter is covered by localized regions. Experiments show that 225 of the 312 ground truth text regions are hit by our localization algorithm. By using the same evaluation measures as above experiments, we obtain precision 0.52, recall 0.62, and *f*-measure 0.52 on this dataset. The precision is much lower than that on Robust Reading Dataset. We infer that the images in blind-captured dataset of object text have lower resolutions and more compact distributions of text information. Then OCR is applied on the localized regions for character and word recognition rather than the whole images. Fig. 10 shows some examples of text localization and word recognition in the system. Recognition algorithm might not correctly and completely output the words inside localized regions. Additional spelling correction is required to output accurate text information. It takes 1.87 seconds on average in reading text from the normalized blind-captured images with resolution 640×480. In real applications, text extraction and device input/output can be processed in parallel, that is, speech output of recognized text while localization of text regions in the next image.

6 Conclusion

In this paper, we have developed a novel text localization algorithm and an assistive text reading prototype system for blind persons. Our system can extract text information from nearby text signage or object captions under complex backgrounds. Text localization and recognition are significant components of our system. To localize text, models of stroke orientation and edge distribution are proposed for extracting features of text. The corresponding feature maps estimate the global structural feature of text at every pixel. Block patterns are defined to project the proposed feature maps of an image patch into a feature vector. An Adaboost learning model is employed to train classifiers of text based on the feature vectors of training samples. To localize text in camera captured images, adjacent character grouping is performed to calculate candidates of text patches prepared for text classification. The Adaboost-based text classifier is applied to obtain the text regions. Off-the-shelf OCR is used to perform word recognition in the localized text regions and transform into audio output for blind users.

Our future work will focus on extending our localization algorithm to process text strings with less than 3 characters and to design more robust block patterns for text feature extraction. We will also extend our system to extract non-horizontal text strings. Furthermore, we will address the significant human interface issues associated with reading region selection by blind users.

Acknowledgement. This work was supported in part by NIH Grant 1R21EY020990, NSF Grant IIS-0957016 and EFRI-1137172.

References

1. Chen, X., Yuille, A.L.: Detecting and reading text in natural scenes. In: CVPR, vol. 2, pp. II-366 – II-373 (2004)
2. Chen, X., Yang, J., Zhang, J., Waibel, A.: Automatic detection and recognition of signs from natural scenes. IEEE Transactions on Image Processing 13(1), 87–99 (2004)
3. Dakopoulos, D., Bourbakis, N.G.: Wearable obstacle avoidance electronic travel aids for blind: a survey. IEEE Transactions on Systems, Man, and Cybernetics 40(1), 25–35 (2010)
4. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: CVPR, pp. 2963–2970 (2010)
5. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: Int. Conf. on Machine Learning, pp. 148–156 (1996)
6. Kim, K.I., Jung, K., Kim, J.H.: Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. IEEE Trans. on PAMI (2003)
7. Kumar, S., Gupta, R., Khanna, N., Chaudhury, S., Joshi, S.D.: Text Extraction and Document Image Segmentation Using Matched Wavelets and MRF Model. IEEE Trans. on Image Processing 16(8), 2117–2128 (2007)
8. Lucas, S.M.: ICDAR 2005 text locating competition results. In: Proceedings of the ICDAR, vol. 1, pp. 80–84 (2005)
9. Ma, L., Wang, C., Xiao, B.: Text detection in natural images based on multi-scale edge detection and classification. In: The Int. Congress on Image and Signal Processing, CISP (2010)
10. Nikolaou, N., Papamarkos, N.: Color Reduction for Complex Document Images. International Journal of Imaging Systems and Technology 19, 14–26 (2009)
11. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. on System, Man and Cybernetics, 62–66 (1979)
12. Phan, T., Shivakumara, P., Tan, C.L.: A Laplacian Method for Video Text Detection. In: Proceedings of ICDAR, pp. 66–70 (2009)
13. Ran, L., Helal, S., Moore, S.: Drishti: an integrated indoor/outdoor blind navigation system and service. In: Pervasive Computing and Communications, pp. 23–40 (2004)
14. Resnikoff, S., Pascolini, D., Etya'ale, D., Kocur, I., Pararajasegaram, R., Pokharel, G.P., et al.: Global data on visual impairment in the year 2002. Bulletin of the World Health Organization, 844–851 (2004)
15. Schneiderman, H., Kanade, T.: A statistical method for 3D object dection applied to faces and cars. In: CVPR (2000)
16. Shi, M., Fujisawab, Y., Wakabayashia, T., Kimura, F.: Handwritten numeral recognition using gradient and curvature of gray scale image. Pattern Recognition 35(10), 2051–2059 (2002)
17. Shivakumara, P., Phan, T., Tan, C.L.: A gradient difference based technique for video text detection. In: The 10th ICDAR, pp. 66–70 (2009)
18. Viola, P., Jones, M.J.: Robust real-time face detection. IJCV 57(2), 137–154 (2004)
19. Yi, C., Tian, Y.: Text string detection from natural scenes by structure based partition and grouping. IEEE Transactions on Image Processing 20(9), 2594–2605 (2011)
20. Zhang, J., Kasturi, R.: Extraction of Text Objects in Video Documents: Recent Progress. In: IAPR Workshop on Document Analysis Systems (2008)
21. ICDAR 2011 Robust Reading Competition (2011), `http://robustreading.opendfki.de/`

A Head-Mounted Device for Recognizing Text in Natural Scenes

Carlos Merino-Gracia[1], Karel Lenc[2], and Majid Mirmehdi[3]

[1] Neurochemistry and Neuroimaging Laboratory, University of La Laguna, Spain
cmerino@ull.es
[2] Center for Machine Perception, Czech Technical University, Czech Republic
lenckar1@fel.cvut.cz
[3] Visual Information Laboratory, University of Bristol, UK
majid@cs.bris.ac.uk

Abstract. We present a mobile head-mounted device for detecting and tracking text that is encased in an ordinary flat-cap hat. The main parts of the device are an integrated camera and audio webcam together with a simple remote control system, all connected via a USB hub to a laptop. A near to real-time text detection algorithm (around 14 fps for 640×480 images) which uses Maximal Stable Extremal Regions (MSERs) for image segmentation is proposed. Comparative text detection results against the ICDAR 2003 text locating competition database along with performance figures are presented.

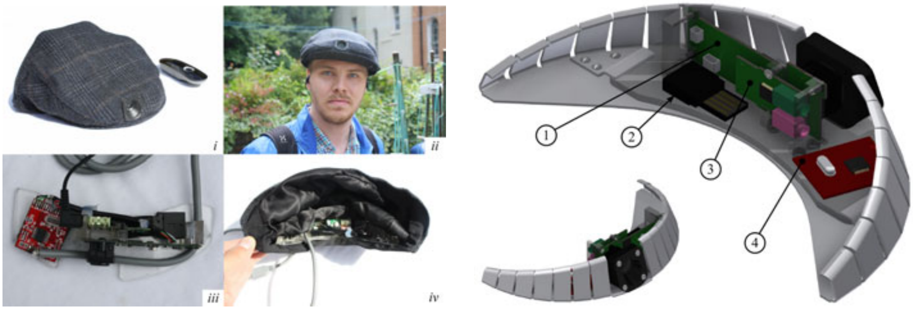
Keywords: wearable device, text detection, text understanding, MSER.

1 Introduction

The area of wearable computing has seen relatively little growth over the last few years after the initial wave of enthusiasm in the area, mainly due to the miniaturisation of personal computing devices, such as mobile phones that need not be worn, but carried, that perform most of our everyday needs. Also, the focus of recent advances in wearable computing have been in specific and specialist areas, e.g. in health monitoring systems. Regardless of this, wearable devices for everyday and general purpose use are still extremely important to help those most in need of it, e.g. disabled users such as the blind, or those incapacitated by language barriers, e.g. tourists!

In this work, we present a simple hat, with embedded camera, speaker, and USB port (see Fig. 1) for an application that involves the real-time detection and tracking of text. The camera provides real-time video, via a discreetly hidden USB cable, to a small laptop (to be carried) where the number crunching occurs. The results of text detection and recognition is returned to the hat via an audio signal on the USB port to a speaker embedded in the hat (which can be used with earphones if necessary). All electronic components are off-the-shelf and are held in a part which is readily removable from the hat. This allows us to easily extend the device in the future just by adapting the removable part, for example

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 29–41, 2012.



(a) (b)

Fig. 1. Developed device together with remote control (a-*i*) and its shape when used (a-*ii*). Removable part (a-*iii*) is placed inside a hat (a-*iv*) in a metal framework which is visualized in image (b). The device comprises a USB camera with auto focus (1), a RC receiver (2) and a USB sound card (3) which are connected to a USB hub (4).

with an embedded computer which will be able to handle all the computation. Since the device does not require units integrated with shades or spectacles, it does not interfere with users who have some residual vision.

Helping visually impaired people to understand the scene in their surrounding environment is a major goal in computer vision, with text detection and its communication to the user a significant aspect of it. One of the earliest approaches can be considered to be the assistive technology approach by Kurzweil's reading machine [8] in 1975 which enabled book reading for blind people using a flat CCD scanner and computer unit with optical character recognition (OCR) and text to speech synthesis (TTS) systems. Several desktop solutions with a similar design are still widely available. This layout was improved using a camera, for example in the iCARE portable reader [7] which made document manipulation less cumbersome. In Aoki et al. [1], a small camera mounted on a baseball cap was used for user navigation in an environment. Chmiel et al. [2] proposed a device comprising glasses with integrated camera and DSP-based processing unit which performed the recognition and speech synthesis tasks. However, this device was directed mainly towards document reading for the blind. The SYPOLE project [21] designed a tool primarily intended for reading text in the user's natural environment by taking snapshots of documents, e.g. banknotes, via a camera mounted on a hand-held PDA device.

In the context of other application areas, detecting and recognizing text is important for translation purposes, e.g. for tourists or robots. This is a subject of interest for the Translation robot [22] which consists of a camera mounted on reading glasses together with a head-mounted display used as the output device for translated text.

Text detection has received increasing attention in recent years, with many works surveyed in [9] and [24]. An example of a recent approach is Pan et al. [20]

who combined classic region-based and connected components-based (CC) methods into a complex text detection system and achieved the best results on the ICDAR dataset yet (used for performance measurement by many text detection algorithms). Their system binarized the image in the first stage based on a text confidence map, calculated from classified gradient features of different sized regions. Segmented CCs were then classified using learned condition random field parameters of several unary and binary component features. Another recent example is Epshtein et al. [5] who used the stroke-width transform to obtain candidate text regions formed of CC pixels of similar stroke widths.

Contrary to the degree of attention enjoyed by text detection, text tracking has been hardly investigated considering that it is very important for reasonable user interaction in any text detection system involving ego or object motion. In our previous work [15], we developed a real-time probabilistic tracker based on particle filtering which is used in the proposed text detection system here. We are only aware of one other work, Myers and Burns [16], who tracked text by feature correspondence across frames by correlating small patches. While we have developed our text tracking application beyond what we previously reported in [15], the focus of the work presented here is on text detection and on the hat-based communication device. Our most recent results on text tracking will be presented in a future work.

In this paper we also examine the use of Maximally Stable Extremal Regions (MSER) [13] for text detection. Originally developed as a method to detect robust image features, the method responds well to text regions. MSER has been used for license plate detection [4] and more recently, Neumann and Matas [17] used MSERs in a supervised learning system for text detection and character recognition using SVM classifiers. Although this method yielded promising results it is computationally expensive. Our approach is based on MSER as a candidate text region detector but we rely on the hierarchical relationship between detected MSERs to quickly filter through them (Section 3). Then a cascade of text classifying filters is applied to candidate text regions. Using much simpler text classification techniques allows us to provide a close to real-time implementation. We present single image text detection performance results evaluated against the standard ICDAR 2003 text locating competition database. Performance figures are provided to illustrate the efficiency of the algorithm (Section 4)[1].

2 Hardware Design

Placing a camera in a hat is a logical choice as it is both an unobtrusive location and an ideal position in reference to where the eyes and head point to. Mayol et al. [14] examined possible positions of wearable cameras and concluded that head mounted cameras provide the best possible link with the user's attention.

[1] Additionally, example videos recorded using the hat can be downloaded from: `http://www.cs.bris.ac.uk/home/majid/CBDAR/`

The hardware proposed here allows its integration into many varieties of hats, here we have used an ordinary fashion accessory – a *flat-cap*.

The hardware was developed with emphasis on robustness, serviceability and visual appearance. Figure 1a shows the appearance of the completed device. It is composed of a fixed part and a removable one. The fixed part is an aluminium plate, bent into a shape that very loosely follows the curves of the hat, while protecting the space used by the removable part. It has an opening in the front side, protected by a glass cover, which fits onto the camera lens. This provides dust and, to some degree, weather insulation. The removable part holds all the electronic devices. It is built out of commodity hardware, with a total cost of all the components under 100€: a high definition web camera with adjustable focus (Logitech Quickcam Pro 9000), an USB sound card used for voice feedback to the user through a pair of connected headphones, a RF transceiver and an USB hub. A view of the inner part of the hat, with the removable part and the electronic parts is shown in Fig. 1b.

The device is controlled by a hand-held remote control which acts like an ordinary USB keyboard. To minimize the number of cables, all the devices are connected to a generic USB hub which allows connecting the hat to any USB-enabled *computing device*, from tablets to fully equipped laptop computers, with a single cable.

3 Proposed System

A simplified schematic of the proposed system is shown in Fig. 2. Initially, we detect candidate text regions in the image input stream using our MSER-based approach. These regions are then tracked in consecutive frames and are eventually analysed using the open source Tesseract OCR[2] engine integrated into our software. Recognized text regions above a significant confidence measure determined by the OCR engine are then sent to a text-to-speech synthesis module (Flite TTS[3], also integrated into our software).

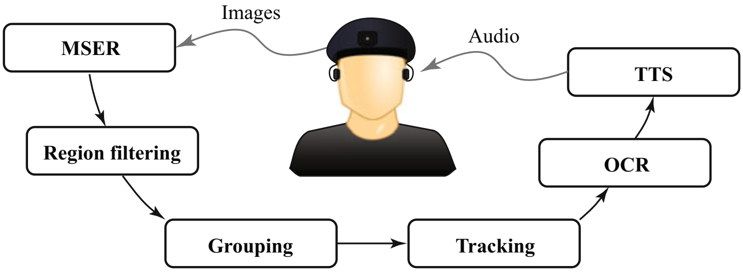


Fig. 2. General structure of the text detector application

[2] Tesseract OCR: `http://code.google.com/p/tesseract-ocr/`

[3] Flite TTS: `http://www.speech.cs.cmu.edu/flite/`

In our previous work on text detection and tracking [15] we used adaptive thresholding to initially binarize the original image. Then a tree was constructed representing the topological relationship between CCs in the binary image. A key step of the algorithm was a hierarchical filtering of the tree nodes, which allowed the rejection of many candidate regions without classification. After that, the remaining tree nodes where filtered using a cascade of text classifiers.

The approach proposed here uses Maximally Stable Extremal Regions [13] for image segmentation along with hierarchical filtering similar to our previous work.

3.1 Image Segmentation

MSERs are regions of interest in an image which present an extremal property of the intensity function around its contour. When applying a varying threshold level to a grey scale image, CC regions in the thresholded image evolve: new regions appear at certain levels, regions grow and eventually join others. Those regions which keep an almost constant pixel count (area) for a range of threshold levels are called MSERs. This technique, originally proposed as a distinguished region detector, also presents very desirable properties when applied to text detection, such as stability and multiscale detection.

MSERs can also be obtained by filtering the *component tree* of the source image, as shown by Donoser et al. [3]. The component tree is a representation of all the CCs which result from applying a varying threshold level to a grey scale image. The CCs are laid out in a hierarchy representing the topological relationship between them. A stability factor – i.e. the rate of change in the area of the components – is computed for each node in the component tree. MSERs are identified as local minima of the stability factor along paths in the tree towards the root.

We use the efficient, linear time MSER algorithm by Nister et al. [18], which crucially also constructs the component tree. We make two passes on the original image. First, MSER+ regions are obtained by applying the MSER algorithm on the image. This produces light regions inside dark ones. Then MSER- regions are obtained by applying the MSER algorithm to the inverse (negative) of the original image which produces dark regions inside light ones. The sets of regions returned by each pass are disjoint and both passes are needed to detect light text on dark backgrounds and dark text on light backgrounds. The algorithm can be easily modified to return a *hierarchical MSER* tree; an example output can be seen in Fig. 4b where blue regions were obtained by the MSER+ pass and the red regions by the MSER- pass. Darker regions represent upper tree nodes (closer to the root), while brighter regions show lower nodes (closer to the leaves). With hierarchical MSER, we have the desirable properties of MSERs as a distinguished region finder applied to text detection. Additionally, we keep the topological relationship of the CCs, which provides context information for later text filtering stages.

The resulting hierarchical MSER tree is then pruned in two stages: (1) reduction of linear segments and (2) hierarchical filtering. The first stage identifies

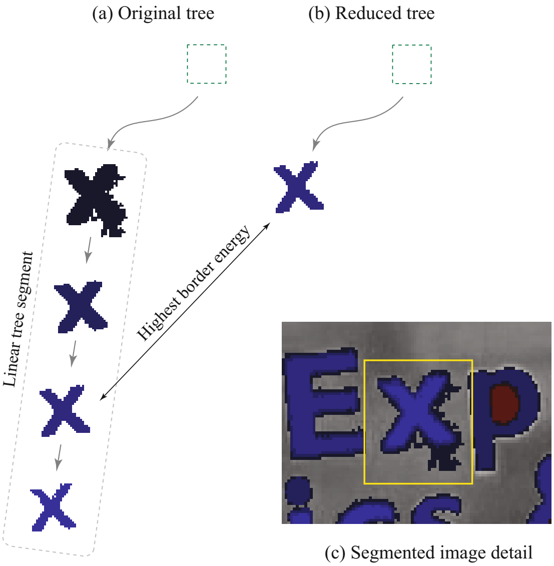
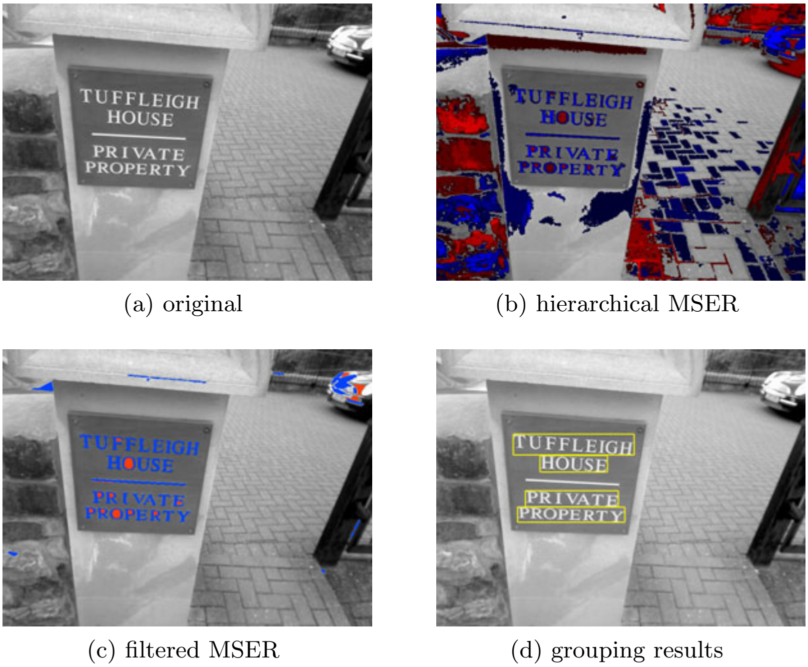


Fig. 3. Linear tree segments removal



(a) original (b) hierarchical MSER

(c) filtered MSER (d) grouping results

Fig. 4. Output from different stages of the text detection algorithm

all the linear segments within the tree where a linear segment is a maximum path between two tree nodes without any branches in between. Likewise, it is a path starting with a node with only one child, and ending with a branch node (a node with more than one child), or a leaf. Each linear segment is then collapsed into the node along the path, as shown in Fig. 3, which maximizes the *border energy* function (see below). In the second stage, the tree is walked depth-first, and a sequence of text classifying filters is applied to leaf nodes. Any non-leaf node without any descendant node classified as text is also tested with the text classifying filters. This stage is similar to the hierarchical tree filtering we originally proposed in [15]. Figure 4 shows the output of several stages of our text detection algorithm.

3.2 Region Filtering

During the tree walk, candidate regions are passed through a cascade of filters, i.e. *size*, *aspect ratio*, *complexity*, *border energy* and *texture*. This arrangement means that most of the regions will be discarded by the simpler filters, thus reducing the number of regions examined by the more complex tests. Thus, for a region i:

Size – the simplest condition filters out regions whose boundary length falls outside an allowed interval $(l_{min}; l_{max})$. The interval limits are fixed as a function of the image size:

$$l_{\min} < |\mathbf{B}_i| < l_{\max} \tag{1}$$

where $\mathbf{B}_i$ is the set of points around the region's boundary.

Aspect Ratio – given width W_i and height H_i of candidate region i, this condition rejects regions that are too wide or too narrow:

$$a_{\min} < \frac{W_i}{H_i} < a_{\max} \tag{2}$$

Complexity – this is a simple measurement of region complexity. It measures the ratio between the region boundary length and its area A_i. This criterion filters out regions with a rough border, which are usually produced by noise:

$$\frac{|\mathbf{B}_i|}{A_i} < c \tag{3}$$

Border Energy – this is a measure of contrast against the background. It filters out regions with low average edge response (from a Sobel operator (S_x, S_y)) around its boundary set of points $\mathbf{B}_i$, i.e. the region is valid only if its border energy exceeds a threshold:

$$\frac{1}{|\mathbf{B}_i|} \sum_{(x,y)\in\mathbf{B}_i} \sqrt{(S_x(x,y)^2 + S_y(x,y)^2)} > e \tag{4}$$

Texture Measure – the last filter in the sequence is a measurement of texture response, as text regions usually contain high frequencies. We found that the LU transform [23] yields good response results when applied to text regions. It is a simple transformation based on LU decomposition of square image sub-matrices A around each interest point.

$$A = P\ L\ U \tag{5}$$

where L and U are lower and upper diagonal matrices and the diagonal elements of L are equal to one. Matrix P is a permutation matrix. In the LU decomposition, the number of zero diagonal elements of U is in direct proportion to the dimensionality of the null-space of A.

The actual texture response $\Omega_p(l,w)$ is calculated as the mean value of the diagonal values of the U matrix.

$$\Omega_p(l,w) = \frac{1}{w-l+1}\sum_{k=l}^{w}|u_{kk}|, \qquad 1 < l < w \tag{6}$$

where w is the window size and l number of skipped lower frequency values. The texture response T_i of a region i is calculated as the mean LU transform value of a sampled set of points (N_i) inside the bounding box of the region.

$$T_i = \frac{1}{|N_i|}\sum_{p\in N_i}\Omega_p(l,w) \qquad T_i > t \tag{7}$$

Figure 5 shows the output of the of LU transform on an example image. In all the filters above, the thresholds were determined empirically and fixed in all our experiments to: $a_{\min} = 0.1$, $a_{\max} = 5$, $c = 1.4$, $e = 40$ and $t = 1.9$.

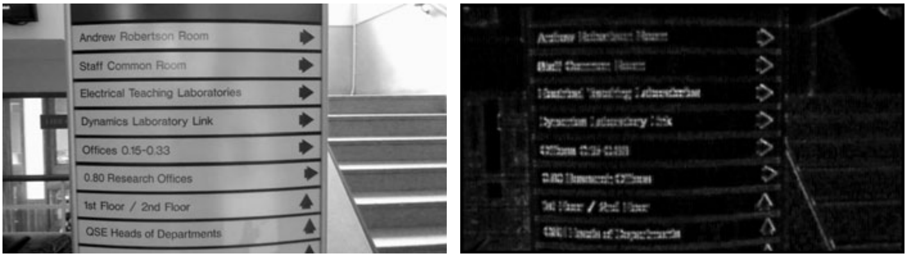


Fig. 5. LU transform output on an example image

3.3 Perceptual Text Grouping

After the image segmentation step, which produces a set of candidate text regions (usually representing isolated letters), a perceptual grouping step is performed to join them into candidate words and phrases. First, a planar Delaunay graph is constructed joining the centre of gravity of every text region. Each vertex of the graph represents a single text region, while the edges represent proximity relationships. Next, each edge e is then filtered using a sequence of tests.

Edge Angle – The first test looks at the angle between edges and the horizontal axis ($\alpha(e)$), such that,

$$-45^\circ < \alpha(e) < 45^\circ \tag{8}$$

This is a strong limitation but the majority of text is horizontal or with a slight slope. The angle of the text is also limited by the capabilities of the OCR engine used, as for now we are not performing any perspective correction.

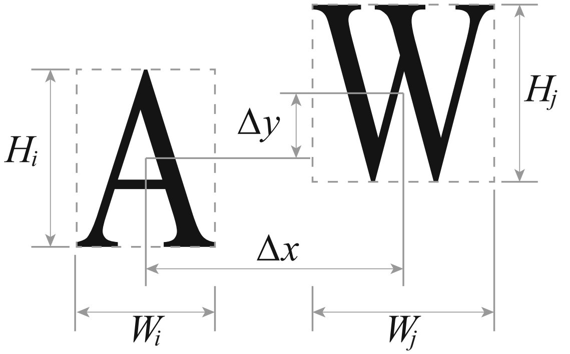


Fig. 6. Variables used for text grouping

Relative Position of Adjacent Tegions – The following criteria were inspired by the work of Ezaki et al. [6]. Two letters appearing on the same text line are usually close together. In this test we limit the allowed distance, relative to their respective sizes.

$$\mathit{\Delta} x < r_x \max(H_i, H_j) \qquad \mathit{\Delta} y < r_y \max(W_i, W_j) \tag{9}$$

where (H_i, W_i) and (H_j, W_j) are the bounding box dimensions of both regions, and $(\mathit{\Delta} x, \mathit{\Delta} y)$ represents the distance between the centres of both regions' bounding boxes (Figure 6). (r_x, r_y) are the *proximity coefficients.*

Size of Adjacent Regions – Similarly to the last test, two letters laying on the same line are assumed to have a similar size. This test limits the variance of adjacent region sizes.

$$\frac{|H_i - H_j|}{|H_i + H_j|} < r_h \qquad \frac{|W_i - W_j|}{|W_i + W_j|} < r_w \tag{10}$$

where (r_h, r_w) are the *size coefficients*, also determined experimentally.

After the edge filtering stage every remaining connected subgraph represents a text group. Text groups are tracked on consecutive frames and sent to the OCR engine for recognition.

4 Results

To facilitate comparative analysis, we measure performance on single image text detection on the ICDAR 2003 text localisation competition ‘TrialTrain’ dataset [10]. The same definitions for *precision* and *recall* were used as defined by the competition. However, given that our algorithm detects whole sentences instead of isolated words, we joined the bounding boxes of the ICDAR database words into sentences, to be able to make fair evaluations. This is the same approach that Pan et al. [19] employed.

The performance result[4] of the proposed method is shown in Table 1 along with the reported detection results from ICDAR 2003 and ICDAR 2005 text location competitions (average, and winning entries), as well as our previous method [15] and three other recent and state-of-the-art algorithms [19,17], and [5].

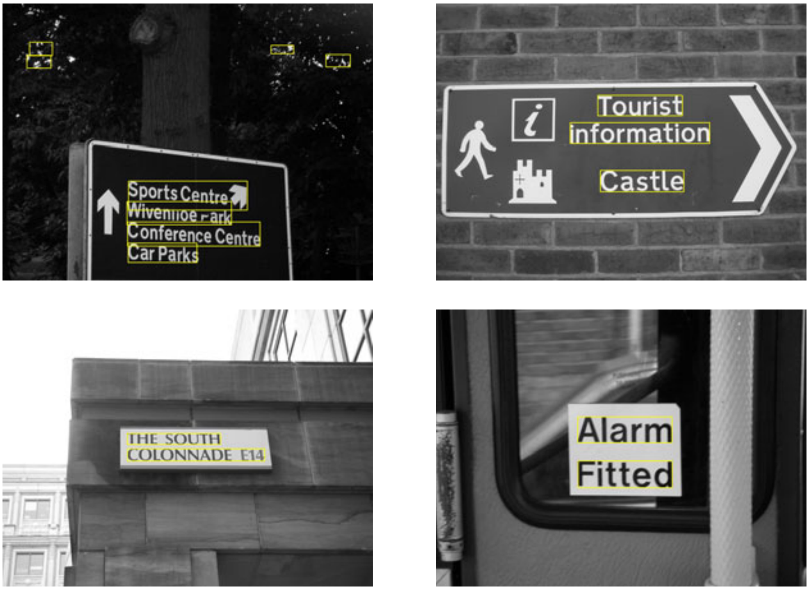


Fig. 7. Example results for some of the ICDAR 2003 database images

The proposed method shows a recall value of 0.67, close to the currently best performing algorithms, e.g 0.71 of [19], while not managing to obtain comparable precision performance. This means that our algorithm overestimates the number of detected regions, but indeed, it is not missing many real text locations. The lower precision rate can be compensated by the OCR engine discarding the unrecognisable regions. The text tracking step can also help in discarding the

[4] All results were obtained using an Intel Core 2 Duo T9300 CPU.

false positives as these non-text regions are unstable, while text regions are more consistently detected across several frames. In fact, by performing registration and super-resolution on tracked text regions [12], recognition accuracy can be increased. This is however beyond the scope of this paper and forms part of our future work. Some example results are shown in Fig. 7.

Table 1. Text detection performance on the ICDAR 2003 database

Text localization	prec.	recall	f	time (s)
Ashida (2003 winner) [10]	0.55	0.46	0.50	8.5
ICDAR 2003 average [10]	0.32	0.32	0.31	5.3
Hinnerk Becker (2005 winner) [11]	0.62	0.67	0.64	14.4
ICDAR 2005 average [11]	0.39	0.46	0.39	4.25
Merino and Mirmehdi [15]	0.44	0.68	0.48	0.1
Neumann and Matas [17]	0.59	0.55	0.57	N/A
Epshtein et al. [5]	0.73	0.60	0.66	0.94
Pan et al. [19]	0.67	0.71	0.69	2.43
Proposed method	0.51	0.67	0.55	0.2

One key advantage of our implementation is its simplicity and speed, which makes it feasible for real-time applications, including those involving text tracking. On the ICDAR database, it takes an average of 156 ms per image, but this is not representative for a real-time video text processor as every ICDAR database image has a different size and they are mostly high resolution still images. For video sequences we are able to process 14fps on 640 × 480 images and 9fps on 800 × 600 images (see Table 2).

Table 2. Time consumptions of different stages of the text locator

	MSER	Filtering	**total**	
ICDAR database	134 ms	16 ms	156 ms	
640 × 480 video	49 ms	10 ms	61 ms	14 fps
800 × 600 video	74 ms	15 ms	95 ms	9 fps

5 Conclusion

We have reported a wearable text recognition tool that employs MSERs as the basis for real-time text detection. The proposed method refines our previous real time algorithm by exploiting hierarchical structure obtained from MSERs to yield more stable regions compared to the previous adaptive threshold method. It outperforms other published approaches computationally while maintaining similar text detection performance on the ICDAR dataset. In our future work, we plan to explore the introduction of a training stage for character recognition without reliance on third-party software, adding more cascading filters, and improving precision and recall results in general.

Acknowledgements. The Hat's construction was carried out at Bristol University based on a previous prototype and work carried out at University of La Laguna by Carlos Merino Gracia under the direction of José Luis González Mora.

Karel Lenc's contribution to this work was carried out at Bristol University as an ERASMUS student.

The Spanish Ministerio de Industria, Turismo y Comercio funds Carlos Merino Gracia through the European Regional Development Fund (project TSI-020100-2009-541).

References

1. Aoki, H., Schiele, B., Pentland, A.: Realtime personal positioning system for wearable computers. In: ISWC 1999, pp. 37–43. IEEE Computer Society, Washington, DC, USA (1999)
2. Chmiel, J., Stankiewicz, O., Switala, W., Tluczek, M., Jelonek, J.: Read IT project report: A portable text reading system for the blind people (2005)
3. Donoser, M., Bischof, H.: Efficient maximally stable extremal region (MSER) tracking. In: CVPR 2006, pp. 553–560 (2006)
4. Donoser, M., Arth, C., Bischof, H.: Detecting, Tracking and Recognizing License Plates. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007, Part II. LNCS, vol. 4844, pp. 447–456. Springer, Heidelberg (2007)
5. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: CVPR 2010, pp. 2963–2970 (2010)
6. Ezaki, N., Kiyota, K., Minh, B., Bulacu, M., Schomaker, L.: Improved text-detection methods for a camera-based text reading system for blind persons. In: ICDAR 2005, pp. 257–261 (2005)
7. Hedgpeth, T., Black, J.A., Panchanathan, S.: A demonstration of the iCARE portable reader. In: ASSETS 2006, pp. 279–280 (2006)
8. Kurzweil, R.: The age of spiritual machines: when computers exceed human intelligence. Viking Press (1998)
9. Liang, J., Doermann, D., Li, H.: Camera-based analysis of text and documents: a survey. IJDAR, 84–104 (2005)
10. Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: ICDAR 2003 robust reading competitions. In: ICDAR 2003, pp. 682–687 (2003)
11. Lucas, S.: ICDAR 2005 text locating competition results. In: ICDAR 2005, pp. 80–84 (2005)
12. Mancas-Thillou, C., Mirmehdi, M.: Super-resolution text using the teager filter. In: CBDAR 2005, pp. 10–16 (2005)
13. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: BMVC 2002 (2002)
14. Mayol, W.W., Tordoff, B.J., Murray, D.W.: Wearable visual robots. Personal and Ubiquitous Computing 6, 37–48 (2002)

15. Merino, C., Mirmehdi, M.: A framework towards realtime detection and tracking of text. In: CBDAR 2007, pp. 10–17 (2007)
16. Myers, G.K., Burns, B.: A robust method for tracking scene text in video imagery. In: CBDAR 2005 (2005)
17. Neumann, L., Matas, J.: A Method for Text Localization and Recognition in Real-World Images. In: Kimmel, R., Klette, R., Sugimoto, A. (eds.) ACCV 2010, Part III. LNCS, vol. 6494, pp. 770–783. Springer, Heidelberg (2011)
18. Nistér, D., Stewénius, H.: Linear Time Maximally Stable Extremal Regions. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 183–196. Springer, Heidelberg (2008)
19. Pan, Y.F., Hou, X., Liu, C.L.: Text localization in natural scene images based on conditional random field. In: ICDAR 2009, pp. 6–10 (2009)
20. Pan, Y.F., Hou, X., Liu, C.L.: A hybrid approach to detect and localize texts in natural scene images. TIP (2011)
21. Peters, J.P., Thillou, C., Ferreira, S.: Embedded reading device for blind people: a user-centred design. In: AIPR 2004, pp. 217–222 (2004)
22. Shi, X., Xu, Y.: A wearable translation robot. In: ICRA 2005 (2005)
23. Targhi, A.T., Hayman, E., Olof Eklundh, J.: Real-time texture detection using the LU-transform. In: CIMCV (2006)
24. Zhang, J., Kasturi, R.: Extraction of text objects in video documents: Recent progress. In: DAS 2008, pp. 5–17. IEEE Computer Society, Washington, DC, USA (2008)

Text Detection of Two Major Indian Scripts in Natural Scene Images

Aruni Roy Chowdhury[1], Ujjwal Bhattacharya[2], and Swapan K. Parui[2]

[1] Department of Information Technology,
Heritage Institute of Technology, Kolkata, India
arunirc@gmail.com

[2] Computer Vision and Pattern Recognition Unit,
Indian Statistical Institute, Kolkata, India
{ujjwal,swapan}@isical.ac.in

Abstract. In this article, we present a robust scheme for detection of Devanagari or Bangla texts in scene images. These are the two most popular scripts in India. The proposed scheme is primarily based on two major characteristics of such texts - (i) variations in stroke thickness for text components of a script are low compared to their non-text counterparts and (ii) presence of a headline along with a few vertical downward strokes originating from this headline. We use the Euclidean distance transform to verify the general characteristics of texts in (i). Also, we apply the probabilistic Hough line transform to detect the characteristic headline of Devanagari and Bangla texts. Further, similarity and adjacency measures are applied to identify text regions, which do not satisfy the verification in (ii). The proposed approach has been simulated on a repository of 120 images taken from Indian roads and the results are encouraging. Also, we have discussed the applicability of the proposed scheme for detection of English texts. Towards this end, we have considered the training and test samples from the image database of ICDAR 2003 Robust Reading Competition.

Keywords: Text detection in natural scenes, text extraction, reading of text in outdoor scene images, ICDAR 2003 Robust Reading Competition.

1 Introduction

Detection of texts in images of natural scenes has enough application potentials. However, related studies are primarily restricted to English and a few other scripts of developed countries. Two surveys of existing methods for detection, localization and extraction of texts embedded in images of natural scenes can be found in [1,2]. A few of the recent studies on the problem include [3,4,5,6,7,8,9].

In the Indian context, there are often texts in one or more Indian script(s) in an image of natural outdoor scenes. Devanagari and Bangla are its two most popular scripts used by around 500 and 220 million people respectively. Thus, studies on detection of Devanagari or Bangla texts in scene images are important. In a

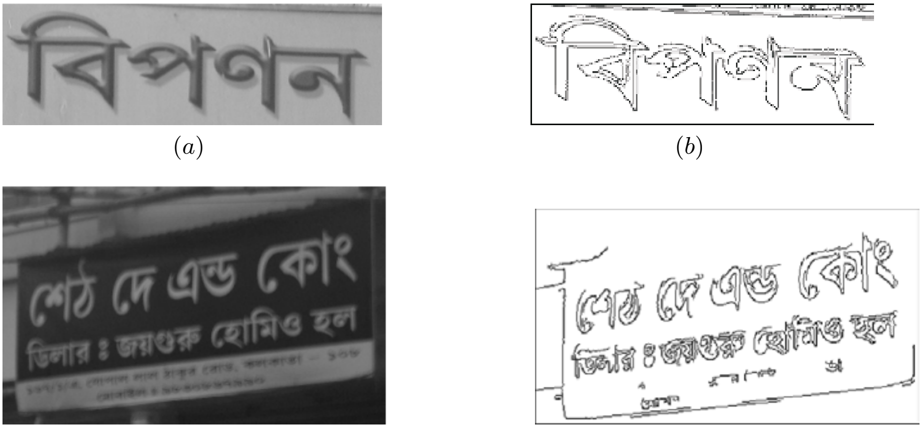
M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 42–57, 2012.

recent study, Bhattacharya et al. [10] proposed a scheme based on morphological operations for extraction of texts of these two scripts from scene images.

Existing approaches for text detection can be broadly categorized into connected component (CC) based and texture based algorithms. The CC based methods are relatively simple, but they often fail to be robust. On the other hand, although texture-based algorithms are more robust, they usually have higher computational complexities.

A well-known feature that text components have approximately uniform stroke widths throughout a character or letter unlike most other components present in a scene image, has been used before [8,9,11]. In [8], an input image is scanned horizontally to identify pairs of sudden intensity changes and the intermediate region is verified for approximate uniformity in color and stroke widths. The limitations of the approach in [8] have been described in [9].

In this later work certain Stroke Width Transform (SWT) was designed based on the Canny image [12] by following rays along the gradient direction of an edge pixel to reach to another edge pixel roughly opposite to the former one. The distance between them was used to assign the stroke width of each pixel along the path of traversal. However, there are instances of embossing and/or shadow-effect when a text component produces more than two edges or broken letter boundaries as shown in Fig. 1. In such situations the SWT [9] provides wrong estimates of stroke widths.



(a) (b)

(c) (d)

Fig. 1. (a) and (c) two original (gray level) images (or its part) from our database of natural scenes; (b) and (d) Canny edges of the images in (a) and (c)

As a solution to this problem, we use the well-known distance transform [13] for detection of candidate text regions and the detail of our strategy for the same is described in Section 3.2. In Section 3.3, we define a set of general rules based on the geometry of text regions for elimination of some of the false positive responses of the scheme described in Section 3.2. At the end of this stage, texts of non-Indic scripts should also get selected. Presence of headline, a characteristic feature of Devanagari and Bangla texts, is verified next and its computation based on probabilistic Hough line transform [14] is presented in Section 3.4.

In the earlier work [10], morphological operations were employed for detection of headline of Bangla and Devanagari texts. However, this approach fails when such texts are sufficiently inclined. In the proposed strategy, the above problem is solved by using probabilistic Hough line transform for the purpose of detection of prominent lines in the image. Subsequent use of script specific characteristics helps to identify the presence of headline in candidate text regions.

2 Devanagari and Bangla Script Characteristics

There are 50 basic characters in the alphabets of both Devanagari and Bangla scripts. For both these scripts, often two or more consonants or one vowel and one or two consonants combine to form different shapes called compound characters. Both Devanagari and Bangla have a large number of such compound characters. Additionally, the shapes of the basic vowel characters (excepting the first one) get modified when they occur with a consonant or a compound character. The shape of a few basic consonant characters also gets modified in a similar situation.

Most of the characters of both scripts have a horizontal line at their upper part. This line is called the headline. In a continuous text of these scripts, the characters in a word often get connected through this headline. A text line of any of these two scripts has three distinct horizontal zones. These are shown in Fig. 2. The portion above the headline is the upper zone and below it but above an imaginary line called the base line, is the middle zone while the part below the base line is called the lower zone. There are many vertical segments in the middle zone of Devanagari and Bangla texts.

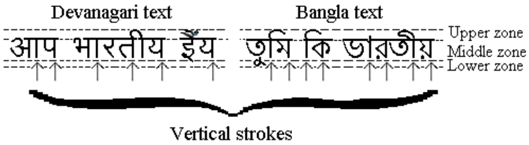


Fig. 2. Three zones of Devanagari and Bangla texts

Here, it may be noted that Gurumukhi and Assamese, two other Indic scripts, also have the headline feature similar to Devanagari or Bangla and no other Indian script has this headline feature. Moreover, Bangla and Assamese scripts have the same character set barring only two characters.

3 Proposed Approach to Text Detection

In a previous study [10], we observed that binarization of scene images often results in partial or complete loss of textual information. However, connected component (CC) analysis based on Canny edge detector has less number of cases of low-contrast regions being missed out. In the present work, we studied a robust scheme for finding CCs from Canny image along with a few rules for detection of Devanagari or Bangla text components. A block diagram of the present scheme is shown in Fig. 3.

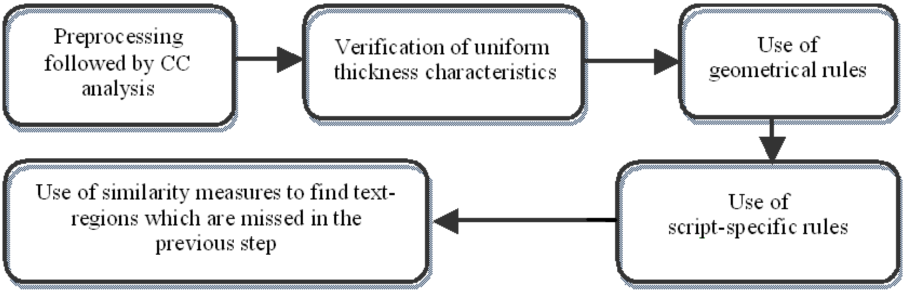

Preprocessing followed by CC analysis
Verification of uniform thickness characteristics
Use of geometrical rules
Use of script-specific rules
Use of similarity measures to find text-regions which are missed in the previous step

Fig. 3. Block diagram of the present scheme for text detection

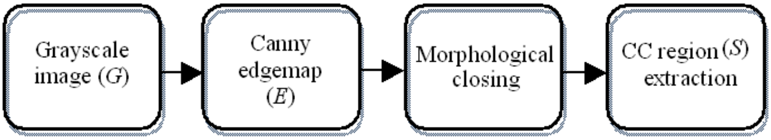


Fig. 4. Block diagram of preprocessing operations

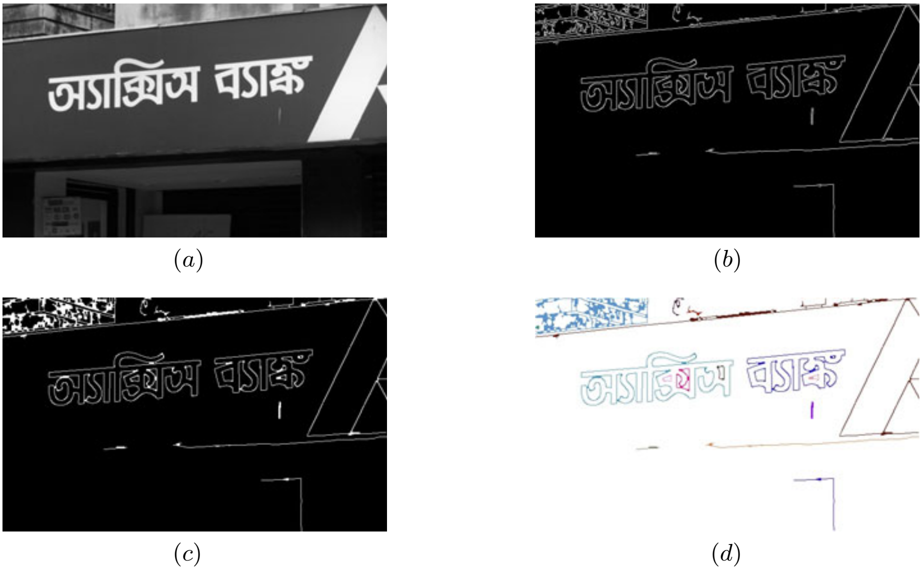
3.1 Preprocessing and Connected Components

An input color image (I) is first converted to 8-bit grayscale image (G). We use Canny operator [12] to get the edgemap (E) from G. This step is perhaps the most critical towards the success of the proposed approach and a brief description of our present implementation is provided. The Canny edge detector in OpenCv has three parameters - $val1$, $val2$ and $val3$. We used $val3 = 3$ for Gaussian smoothing of the input image with 3×3 kernel, the Gaussian being determined using window-size $(w_x = 3, w_y = 3)$

$$\sigma_x = 0.3(\frac{w_x}{2} - 1) + 0.8 \text{ and } \sigma_y = 0.3(\frac{w_y}{2} - 1) + 0.8$$

The larger of $val1$ and $val2$ is used as a threshold for selection of prominent edges and the smaller of these two is used as a distance threshold for linking of nearby edges. On the basis of the training samples of our database of scene images, we selected $val1 = 196$ and $val2 = 53$. This value of $val2$ helped us to avoid linking of edges of text components with edges of background objects. On the other hand, such a choice of $val2$ often leaves edges of a text component segmented into smaller pieces. We solved this problem by applying a morphological closing operation with a 3×3 kernel anchored at center on E as a post-processing operation of the Canny edge detector. This often helps to connect broken edges of the same character or symbol. Also, many erratic edges of background objects merge to form a larger component. A flow chart for these operations is shown in Fig. 4 and their effects are shown in Fig. 5 using a sample image.

For further analysis, we consider the smallest bounding rectangle S in the image G corresponding to each connected component obtained by the above operations.



(a)

(b)

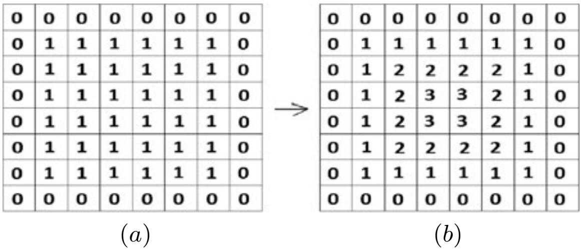
(c)

(d)

Fig. 5. (a) A sample grayscale image, (b) Canny edgemap E of the input image in (a), (c) result of the morphological closing operation on Canny edgemap in (b), (d) connected components in (c) are shown using different random colors - each color corresponds to a separate component in (c)

3.2 Extraction of Strokes of Near-Uniform Thickness

Each sub-image S obtained in Section 3.1 is binarized and subjected to the Euclidean distance transform (DT) [13]. Each pixel in the resulting image is set to a value equal to its distance from the nearest background pixel. Thus, we compute the distance of each object pixel from its edge or boundary as shown in Fig. 6.



0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0

(a)

→

0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0
0	1	2	2	2	2	1	0
0	1	2	3	3	2	1	0
0	1	2	3	3	2	1	0
0	1	2	2	2	2	1	0
0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0

(b)

Fig. 6. (a) a sample sub-image with foreground pixels denoted by 1 and background pixels denoted by 0; (b) the distance transform of the sub-image in (a)

For further analysis we consider the smallest bounding rectangle S in the image G corresponding to each connected component obtained by the above operations.

Determination of Background Color. Texts can appear lighter against dark background or darker against light background. In [9], the distance between edges of opposing gradients was computed along both +ve and -ve gradient directions to account for both the possibilities of lighter or darker texts.

In the proposed scheme, we consider the sub-image S and its inverse S' and compute the DT for each of them as shown below. Let the corresponding transformed images be D and D'. Now, we compute the number of zeros as well as the number of non-zeros along the four boundaries of both D and D'. The number of zeros will be larger for a sub-image with lighter foreground against dark background and the corresponding DT (D or D') is selected as $D^{\star}$. An example is shown in Fig. 7 to justify the above selection procedure.

Some letters may be so aligned that they have majority object pixels present along boundaries, giving a wrong estimation of background color. To deal with this, instead of using the minimum bounding rectangle of each component we increase its size by adding a small integer m (in our implementation, $m = 2$) to its dimensions, taking care of image boundary overflows.

Thus, a larger portion of background pixels is sampled in the bounding rectangle defining the sub-image with fewer chances of foreground pixels being wrongly counted while checking border pixels.

It is to be noted, for the purpose of background color estimation, that even a binarized image would have sufficed. However, as the distance transform is required for subsequent stroke thickness calculation also, we do not perform the extra step of thresholding.

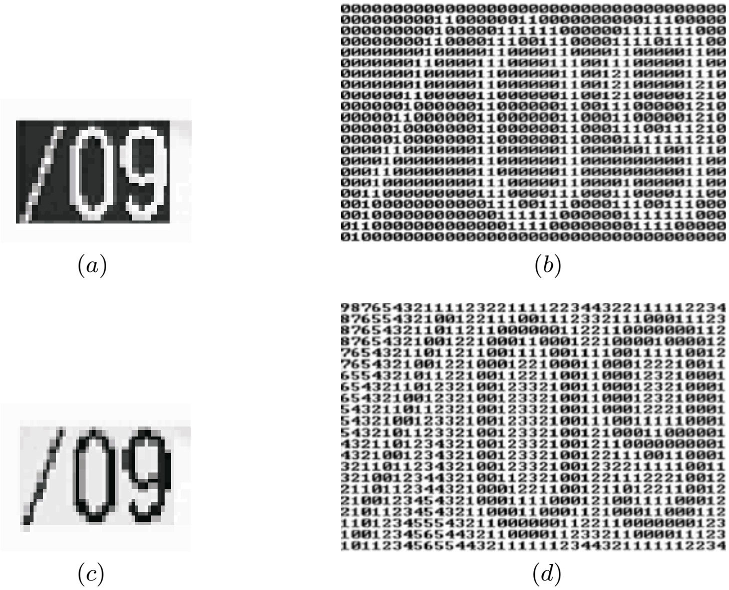


Fig. 7. A sub-image and its inverse are shown in (a) and (c) and the respective DT images (pixel values) are shown in (b) and (d) respectively. Majority of pixels along the boundaries of the DT in (b) are zeros while the same in (d) are non-zeros.

Computation of Stroke Thickness. For each pixel with non-zero $D^\star$, we consider a 3×3 window centered at the pixel. If the $D^\star$ value of the pixel is a local maximum among the nine such values, we store the $D^\star$ value in a list $< T >$ for further processing. In fact, the corresponding pixels form a skeleton like representation of the object as shown in Fig. 8. Such a $D^\star$ value (a local maximum value) is an estimate of half of the local stroke thickness.

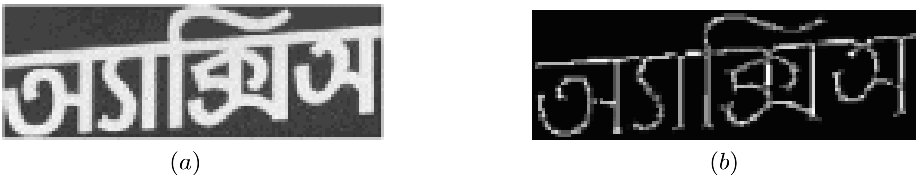


Fig. 8. (a) A sub-image of the image shown in Fig. 5(a); (b) skeleton-like representation of the object in (a) formed by the pixels in its $< T >$

Finally, we compute the mean (μ) and the standard deviation (σ) of the local stroke thickness values stored in $< T >$. If $\mu > 2\sigma$. (well-known 2-σ limit used in statistical process control), we decide that the thickness of the underlying stroke is nearly uniform and select the sub-image S as a candidate text region.

The flow chart of the above scheme for deciding a candidate text region is shown in Fig. 9.

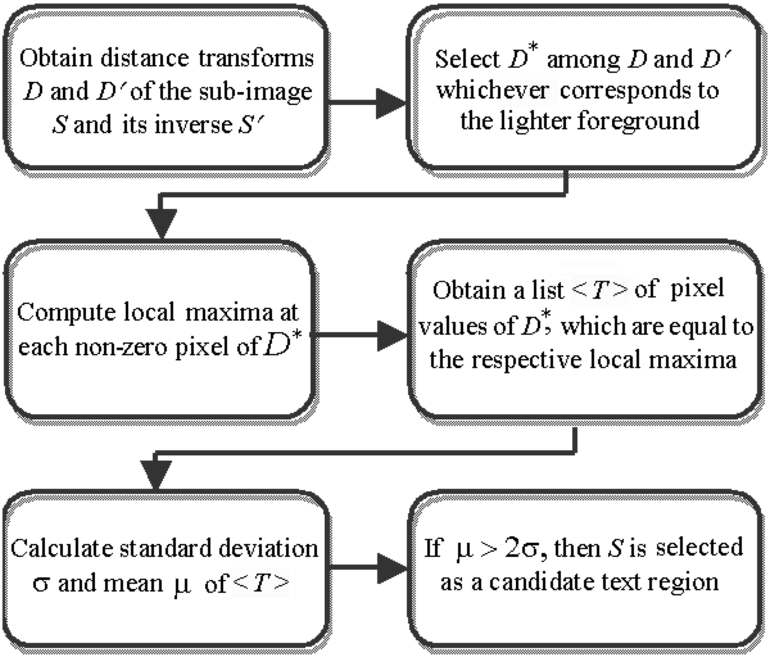

Obtain distance transforms D and D' of the sub-image S and its inverse S'
Select D^* among D and D' whichever corresponds to the lighter foreground
Compute local maxima at each non-zero pixel of D^*
Obtain a list $<T>$ of pixel values of D^*, which are equal to the respective local maxima
Calculate standard deviation σ and mean μ of $<T>$
If $\mu > 2\sigma$, then S is selected as a candidate text region

Fig. 9. Flow chart of the use of nearly uniform stroke thickness characteristics

Candidate text regions selected as in the above are stored in a set $< R >$. These include English texts as well as a few non-text components with near uniform stroke thickness akin to text.

The above approach of candidate text region detection is quite different from the “Stroke Width Transform”-based approach described in [9]. To the best of our knowledge the proposed approach is novel and is simpler than the existing ones.

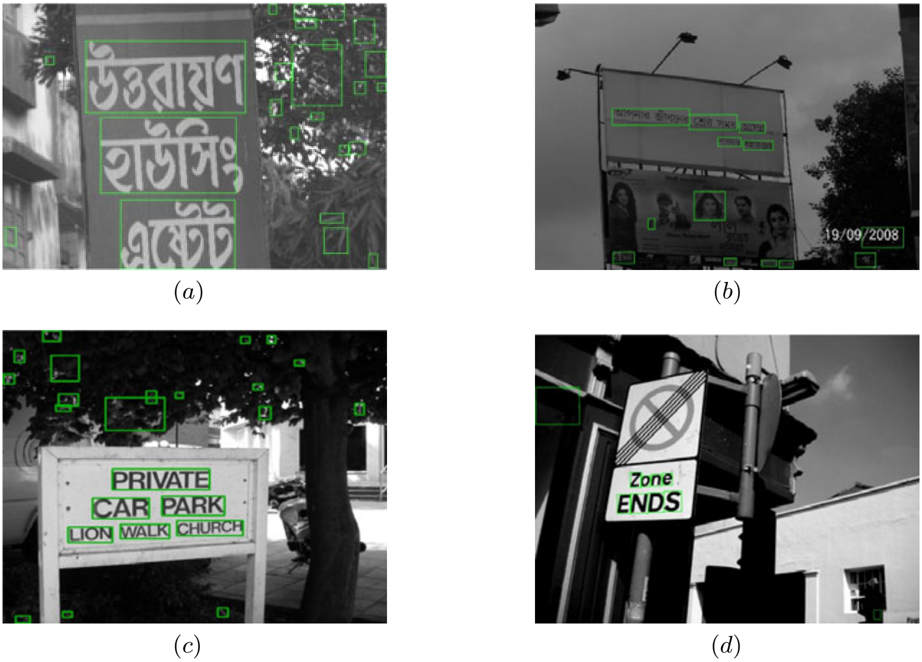
3.3 General Rules Based on the Geometry of Text Regions

Each candidate text region of $< R >$ selected by the above approach is first tested against a small set of rules designed on the basis of geometric characteristics of such regions. These rules are the following:

- The aspect ratio of a text region should vary between 0.1 and 10.
- Both height and width of a candidate region cannot be larger than half of the corresponding size of the input image.
- Height of a candidate text region is larger than 10 pixels.

- The overlap between the rectangular bounding boxes of two adjacent text regions cannot exceed 50% of the area of any one of the two bounding boxes.

The above rules are generally true for most of the scripts. However, the numerical values used in the above rules are selected on the basis of training images consisting of primarily Devanagari and Bangla texts. So, the above numerical values may not be optimal whenever the sample images of the underlying database primarily consist of texts of a different script. At the end of this stage, text regions irrespective of its script get selected. Additionally, a few non-text regions may also get selected. Results on two sample images from our database containing Bangla texts and another two sample images from ICDAR2003 database containing English texts are shown in Fig. 10. Finally, let $< V >$ denote the list of selected regions of a gray level image G at the end of the present stage.



(a) (b)

(c) (d)

Fig. 10. Selected regions are marked by green rectangular boundaries. There are a few situations when non-text regions get selected even after applying the general rules based on their geometric characteristics; (a), (b) two samples from our image database containing texts of Indian scripts; (c), (d) two samples from ICDAR 2003 image database containing English texts.

3.4 Detection of Headline of Bangla and Devanagari Texts

In order to identify regions of Bangla and Devanagari texts from among the regions in the set $< V >$, we compute a few common characteristics features of these two scripts as described below. In each of the above regions we compute the progressive probabilistic Hough line transform (PPHT) [14] to obtain the characteristic horizontal headlines of Devanagari or Bangla texts. This transform usually results in a large number of lines and we consider only the first n prominent (with respect to the number of points lying on them) ones among them. A suitable value of n is selected empirically. Now, the lines with absolute angle of inclination with the horizontal axis less than $\theta°$ (selected empirically to allow significantly tilted words) are considered as horizontal lines. In Figs. 11(a) and 11(d), we have shown two regions corresponding to two connected components of the sample image in Fig. 5. Respective prominent Hough lines are shown in Figs. 11(b) and 11(e) and the selected horizontal lines are shown in Figs. 11(c) and 11(f). A necessary condition for selection of a member of $< V >$ as a text region is that these horizontal lines appear in its upper half. Let $< L >$ denote the set of such horizontal lines corresponding to a region.

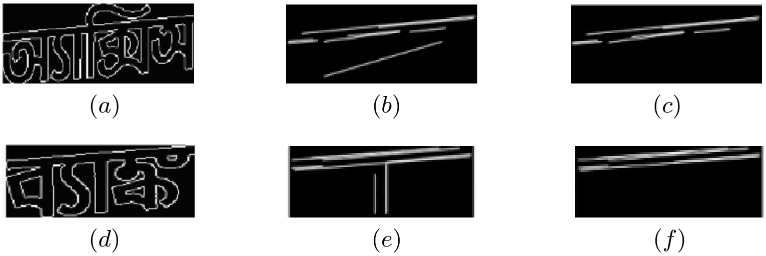


Fig. 11. (a), (d): two separate regions of edgemap E of the sample image in Fig. 5; (b), (e): prominent Hough lines corresponding to the regions in (a) and (d) respectively; (c), (f): selected horizontal lines from (b) and (e) respectively

Final decision of existence of a possible headline among the horizontal lines in $< L >$ is based on computation of vertical Hough lines. We again apply the probabilistic Hough line transform using a smaller threshold value, as these vertical lines need not be as prominent as the horizontal lines of $< L >$. If the majority of these vertical lines (Fig. 12) lie immediately below a member of $< L >$, the corresponding horizontal line in $< L >$ is decided as a headline.

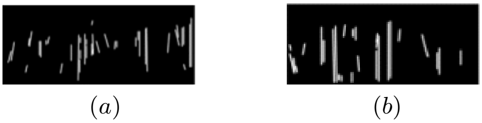


Fig. 12. (a), (b): Vertical Hough lines corresponding to the regions in Figs. 11 (a) and (d) respectively

A sufficient condition that a member in the set $< V >$ is a text region of Devanagari or Bangla is that the region produces a headline as in the above. Let

$< M >$ denote the subset of $< V >$ each member of which satisfies the above sufficient condition. However, a few members of $< V > - < M >$ may still be valid Devanagari or Bangla text.

3.5 Use of "Similarity Measures" for Detecting Missed-Out Text Regions

The main criterion used in the above for selection of texts of Indian scripts is the presence of a headline, which in turn depends on the Hough transform being able to pick up the headline and the vertical strokes immediately below the headline. There are several cases where the headline may be too small and also there are certain situations where it does not occur at all. To detect possible Devanagari or Bangla text regions in $< V > - < M >$, which do not exhibit the headline property as in the above, we recursively loop through the regions of $< M >$ and shift a member of $< V > - < M >$ to $< M >$ provided it has high similarity with one of the current members of $< M >$ with respect to its height, width, relative position and average stroke thickness. We stop when no addition is made to the current list of $< M >$. Values of parameters involved in these similarity measures are decided empirically. In the example shown in Fig.13 a numeral string without any headline has been selected with the help of the above similarity criteria.

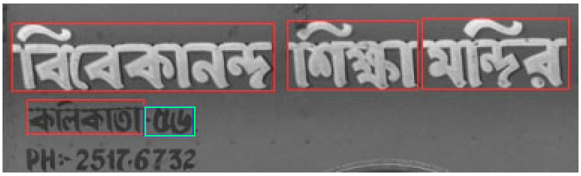


Fig. 13. An instance where part(s) of Bangla text does not exhibit the headline property - numerals in this case. These numerals being adjacent to selected words with headline (shown by red colored rectangular boundaries) are also selected (shown by green colored rectangular boundaries) based on its similarity to the adjacent words.

4 Experimental Results and Discussions

We implemented the proposed scheme on Windows platform, using MSVC++ and the OpenCV computer vision library [15]. Since no standard database of outdoor scene images containing texts of Devanagari and Bangla is publicly available, we are in fact developing one such annotated database captured by (i) a Kodak DX7590 (5.0 MP) still camera and (ii) a SONY DCR-SR85E handy cam used in still mode (1.0 MP). This database will soon be made available for academic research purposes. We used 120 sample images taken from this repository for collection of simulation results of the proposed approach. The training set consists of 20 images and the remaining 100 image samples formed the test set of our simulation. These 100 test images are the same samples used in our previous study [10]. Several of these images contain English texts in addition to Bangla and Devanagari texts. Detected texts in a few of these samples are shown in Fig. 14.

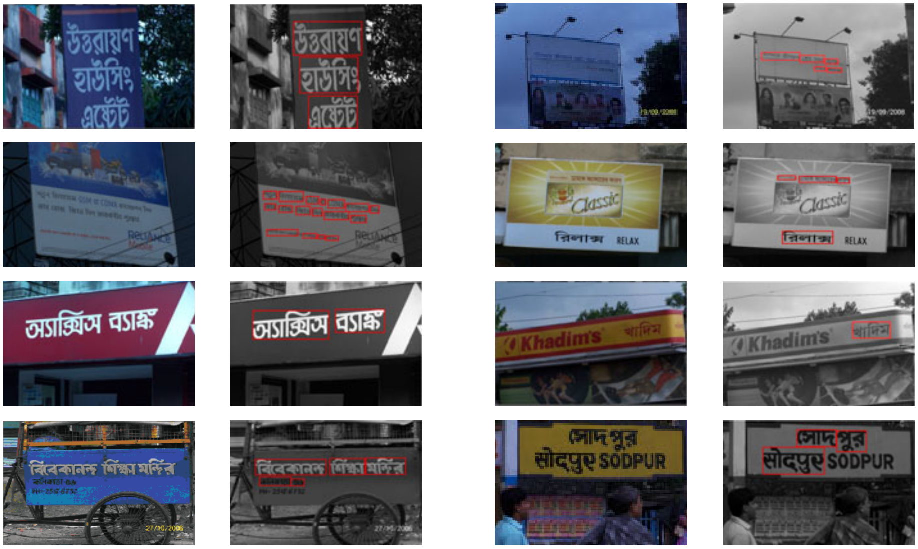


Fig. 14. A few samples from our image database of natural scenes. Identified Bangla and Devanagari texts are marked by red colored rectangular boundaries.

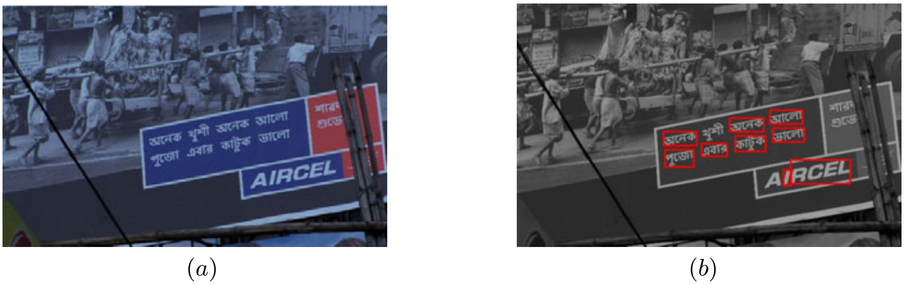
The major drawback of the earlier approach [10] for detection of texts of major Indian scripts was its inability to tackle large tilt or curve in small sized text. The present approach has largely solved this problem. A sample image consisting of tilted texts and results of our text detection approach on the same are shown in Fig. 15. The previous approach [10] failed to detect any text from this sample image. Also, the present approach based on probabilistic Hough transform successfully detects headline even if it is discontinuous and noisy. Another advantage of the proposed approach is that it identifies the word boundaries, which should help the subsequent OCR module.

We summarize the results of our simulation using 100 sample images by providing values of two quantities, recall and precision defined as follows.

$$\text{Precision}(p) = \frac{\text{Number of correctly detected Bangla or Devanagari words}}{\text{Total number of detections}}$$

$$\text{Recall}(r) = \frac{\text{Number of correctly detected Bangla or Devanagari words}}{\text{Total number of Bangla or Devanagari words in the sample images}}$$

The proposed method provided $p = 0.72$ and $r = 0.74$. This improves the results of the earlier report on the same image samples [10] with respect to both the measures. Two of the sample images on which the performance of the proposed



(*a*) (*b*)

Fig. 15. (*a*) A sample image with sufficiently tilted texts; (*b*) detected text regions are shown in red-colored rectangles

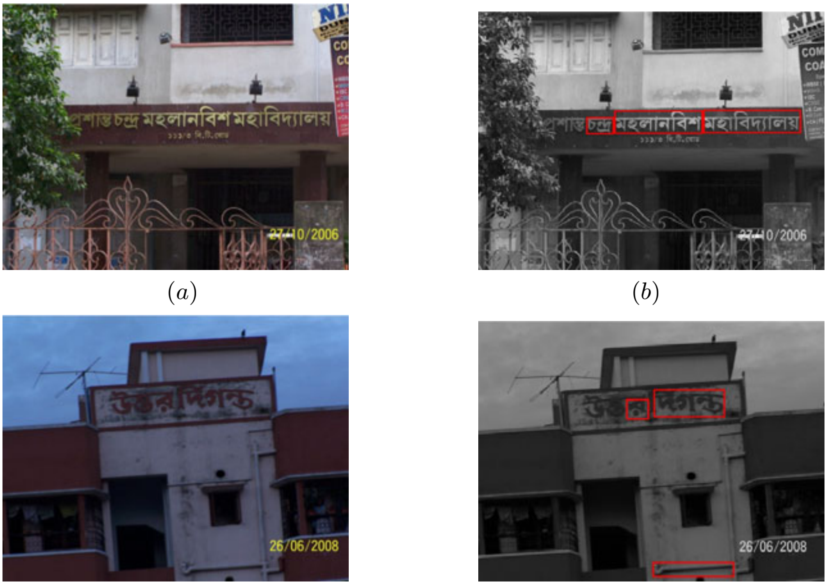
approach was poor are shown in Fig. 16. Comparative results with respect to the algorithms in [9] (which did not use any script specific characteristics) and [10] are provided in Table 1.

Table 1. Comparative performance on 100 test sample images of our database

Algorithm	Precision (p)	Recall (r)
Proposed algorithm	0.72	0.74
Algorithm in [9]	0.59	0.64
Algorithm in [10]	0.69	0.71

Although the present motivation is the development of a methodology for detection of particularly Bangla and Devanagari texts in scene images, the uniform stroke thickness characteristic used at its initial stage does hold good for texts of any script. Moreover, the rules discussed in Section 3.3 are true for general texts irrespective of its script. However, the numerical values of the parameters involved in those rules are needed to be tuned for the particular scripts to ensure acceptable performance. As a part of the present work, we separately tuned these parameters based on the training samples of ICDAR 2003 image database [16] of robust reading competition. However, in this latter attempt we naturally did not apply Indian script specific headline detection module described in Section 3.4. A few samples from the ICDAR 2003 database on which our text detection algorithm performed perfectly are shown in Fig. 17 and another few samples from the same database on which our algorithm did not perform efficiently are shown in Fig. 18.

A close observation of the simulation results presented in Figs. 16 and 18 reveals that the false detections are more frequent in case of ICDAR 2003 sample images compared to the samples of our database. This is justified by the fact that the outputs on samples of our database presented in Fig. 16 passed an additional checking of presence of headline (as described in Section 3.4) which is a very strong feature and effectively removes majority of false detections.



(*a*) (*b*)

(*c*) (*d*)

Fig. 16. (*a*) and (*c*): Two samples from our image database. In (*b*) a few words of (*a*) remain undetected since these have got connected in the Canny edgemap with adjacent non-text components; in (*c*) the text portion is dirty due to fungus. The false detection in (*d*) is due to the presence of a strong linear component.

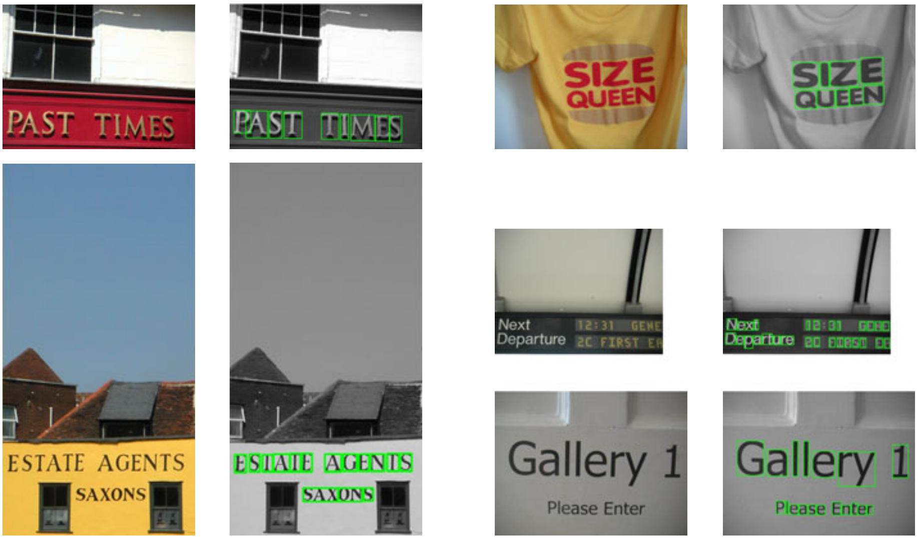


Fig. 17. A few samples from ICDAR 2003 image database of natural scenes on each of which performance of the proposed algorithm is perfect. Portions identified as texts are marked by green colored rectangular boundaries.

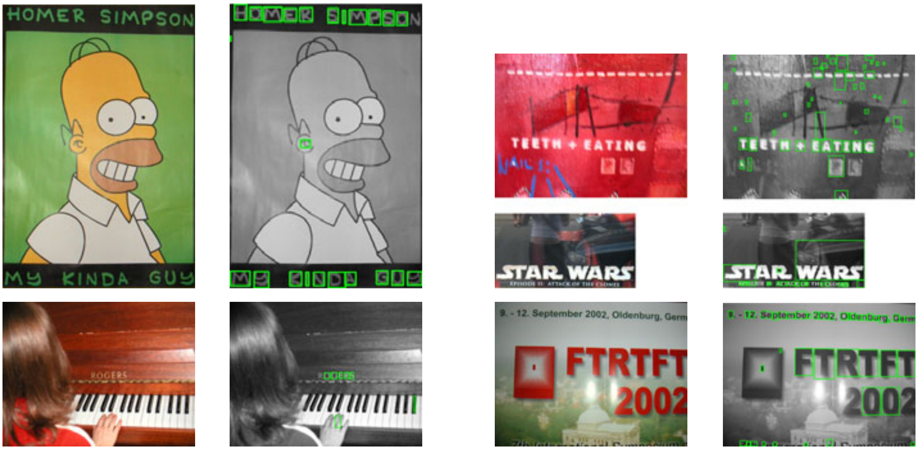


Fig. 18. A few samples from ICDAR 2003 image database of natural scenes on each of which performance of the proposed algorithm is poor. Portions identified as texts are marked by green colored rectangular boundaries.

5 Conclusions

Although the simulation results of the proposed method on our image database of outdoor scenes containing texts of major Indian scripts are encouraging, in several cases, it produced false positive responses or some of the words or a part of a word failed to be detected. Another major concern of the present algorithm is the empirical choice of a number of its parameter values.

We are at present studying the effect of using machine learning strategies to avoid empirical choice of the values of its various parameters. Preliminary results show that this will improve the values of both p and r by several percentages. However, we need more elaborate testing of the same.

In future, we plan to use a combined training set comprising of training samples from both of our and the ICDAR2003 image databases so that the resulting system can be used for detection of texts of major Indian scripts as well as English. Finally, identification of scripts of detected texts is necessary before sending them to the respective text recognition modules. There are a few works [17] in the literature on this script identification problem. Similar studies of script identification for texts in outdoor scene images will be taken care of in the near future.

References

1. Liang, J., Doermann, D., Li, H.: Camera Based Analysis of Text and Documents: A Survey. Int. Journ. on Doc. Anal. and Recog. 7, 84–104 (2005)
2. Jung, K., Kim, K.I., Jain, A.K.: Text Information Extraction in Images and Video: a Survey. Pattern Recognition 37, 977–997 (2004)
3. Li, H., Doermann, D., Kia, O.: Automatic Text Detection and Tracking in Digital Video. IEEE Trans. Image Processing 9, 147–167 (2000)

4. Gllavata, J., Ewerth, R., Freisleben, B.: Text Detection in Images Based on Unsupervised Classification of High Frequency Wavelet Coefficients. In: Proc. of 17th Int. Conf. on Patt. Recog., vol. 1, pp. 425–428 (2004)
5. Saoi, T., Goto, H., Kobayashi, H.: Text Detection in Color Scene Images Based on Unsupervised Clustering of Multihannel Wavelet Features. In: Proc. of 8th Int. Conf. on Doc. Anal. and Recog., pp. 690–694 (2005)
6. Ezaki, N., Bulacu, M., Schomaker, L.: Text Detection From Natural Scene Images: Towards a System for Visually Impaired Persons. In: Proc. of 17th Int. Conf. on Patt. Recog., vol. II, pp. 683–686 (2004)
7. Ye, Q., Huang, Q., Gao, W., Zhao, D.: Fast and Robust Text Detection in Images and Video Frames. Image and Vis. Comp. 23, 565–576 (2005)
8. Subramanian, K., Natarajan, P., Decerbo, M., Castañon, D.: Character-Stroke Detection for Text-Localization and Extraction. In: Proc. of Int. Conf. on Doc. Anal. and Recog., pp. 33–37 (2005)
9. Epshtein, B., Ofek, E., Wexler, Y.: Detecting Text in Natural Scenes with Stroke Width Transform. In: Proc. of IEEE Conf. on Comp. Vis. and Patt. Recog., pp. 2963–2970 (2010)
10. Bhattacharya, U., Parui, S.K., Mondal, S.: Devanagari and Bangla Text Extraction from Natural Scene Images. In: 10th Int. Conf. on Doc. Anal. and Recog., pp. 171–175 (2009)
11. Kumar, S., Perrault, A.: Text Detection on Nokia N900 Using Stroke Width Transform, `http://www.cs.cornell.edu/courses/cs4670/2010fa/projects/final/results/group_of_arp86_sk2357/Writeup.pdf` (last accessed on October 31, 2011)
12. Canny, J.: A Computational Approach to Edge Detection. IEEE Trans. Patt. Anal. and Mach. Intell. 8, 679–714 (1986)
13. Borgefors, G.: Distance Transformations in Digital Images. Comp. Vis., Graph. and Image Proc. 34, 344–371 (1986)
14. Matas, J., Galambos, C., Kittler, J.: Progressive Probabilistic Hough Transform. In: Proc. of BMVC 1998, vol. 1, pp. 256–265 (1998)
15. Bradski, G., Kaehler, A.: Learning OpenCV. O'Reilly Media, Inc. (2008)
16. Lucas, S.M., et al.: ICDAR 2003 Robust Reading Competitions. In: Proc. of 7th Int. Conf. on Doc. Anal. and Recog., pp. 682–668 (2003)
17. Zhou, L., Lu, Y., Tan, C.L.: Bangla/English Script Identification Based on Analysis of Connected Component Profiles. In: Proc. Doc. Anal. Syst., pp. 243–254 (2006)

An Algorithm for Colour-Based Natural Scene Text Segmentation

Chao Zeng, Wenjing Jia, and Xiangjian He

Research Centre for Innovation in IT Services and Applications (iNEXT)
FEIT, University of Technology, Sydney, Sydney, Australia
chaozeng@it.uts.edu.au, {Wenjing.Jia,Xiangjian.He}@uts.edu.au

Abstract. Before the step for text recognition, a text image needs to be segmented into foreground containing only the text area and background. In this paper, a method is proposed for segmenting colour natural scene texts which suffer from a wide range of degradations with complex background. A text image is firstly processed by two 3-means clustering operations with different distance measurements. Then, a modified connected component (CC)-based validation method is used to obtain the text area after clustering. Thirdly, a proposed objective segmentation evaluation method is utilised to choose the final segmentation result from the two segmented text images. The proposed method is compared with other existing methods based on the ICDAR2003 public database. Experimental results show the effectiveness of the proposed method.

Keywords: natural scene text segmentation, k-means clustering, connected component analysis (CCA), segmentation evaluation.

1 Introduction

Text information extraction (TIE) [1] has found a wide range of applications in real life, such as vehicle license plate recognition, video text reading system for blind people and so on. A TIE system mainly consists of three steps: text localisation, text segmentation and text recognition. Particularly, text segmentation refers to the process of separating the text pixels from the background and producing a binarised version of a localised text image. An example of a segmented text image is shown in Fig. 1(d), where text pixels are marked as black colour and background pixels are marked as white colour without loss of generality. The results of text segmentation highly affect the performance of the next step: text recognition. Since it plays such an important role in a TIE system, many publications have devoted to the research of text segmentation. In recent years, there are mainly two types of methods for text segmentation: edge-based methods and colour-based methods.

Edge-based methods usually make use of the edge information of the character strokes of text, and then extract the text in further processing steps. Kasar et al. [2] performed an edge-based connected component analysis and estimated the threshold for each edge connected component with the help of the foreground and the

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 58–68, 2012.

background pixels. In [3], the union of edge information on R, G and B channel was used to generate an edge image. The representative colours in CIE L*a*b* space were obtained along the normal directions of the edge contour. These representative colours served as the initialization of k-means clustering. In each colour cluster, connected component labelling was utilized and several constraints were defined to remove the non-text components. The final binarisation for each component is achieved by the threshold estimation similar to that in [2]. In the work of Yu et al. [4], an improved version of double-edge model based on the method in [5] was proposed to extract character strokes between the predefined minimum and maximum stroke widths. In [6], a stroke edge filter was devised to get the edge information of character strokes. The stroke edges were identified by a two-threshold scheme and the stroke colour is further estimated by inner pixels between edge pairs. The segmentation result was obtained by combining the binary stroke edges and strokes with four heuristic rules. Firstly, the edge of text was detected in [7] to get the text boundary. Secondly, the pixels inside the text boundary were selected with a low/high threshold as the seeds for the modified flood fill algorithm. Lastly, the false edges were removed by a morphological opening operation. However, the edge information could not be correctly achieved for the texts with uneven lighting or highlight, so the segmentation result was not good and it in turn resulted in the recognition failures.

Whereas colour-based methods assume that the characters of text have the same colour information and are different from that of the background. In [8], the colour plane with the maximum breadth of histogram among Cyan/Magenta/Yellow colour space was chosen for adaptive single-character image segmentation. In [9], the binarisation was performed on the optimally selected colour axis, on which it had the largest inter-class separability in the RGB colour space. Thillou et al. [10] pointed out that the selection of clustering distances is the main problem of the degraded natural text segmentation after investigating several colour spaces for clustering,. With the combination of Euclidean Distance and Cosine Similarity [11], the pixels of each natural scene text image is clustered into three categories: textual foreground, background and noise. The final segmentation result was achieved after the implementation of Log-Gabor filters. Colour-based methods describe the chromaticity differences between text and background which can be used for segmentation.

This paper proposes a new connected component-based text validation measure for finding most possible cluster of text after 3-means clustering. In order to choose the better segmentation result obtained from the Euclidean Distance-based and the Cosine Similarity-based 3-means clustering, an objective segmentation evaluation method describing the intra-region homogeneity and the inter-region contrast is also proposed.

The remaining of the paper is organised as follows. In Section 2, an overview of the selective metric-based clustering is given. The proposed method is presented in Section 3. Experimental results are shown in Section 4. Finally, conclusion is presented in Section 5.

2 Introduction of Selective Metric-Based Clustering

In order to get a better performance, both luminance and colour chromaticity were used to process a colour image in [11]. The 3-means clustering algorithm was applied to a natural scene text image with Euclidean Distance D_{eucl} and Cosine Similarity $S_{\cos}$ respectively. The pixels of a natural scene text image were classified into textual foreground cluster, background cluster and noise cluster by 3-means clustering algorithm. The Euclidean Distance D_{eucl} and Cosine Similarity $S_{\cos}$ were defined as:

$$D_{eucl}(\overrightarrow{x_1},\overrightarrow{x_2})=\sqrt{(R_1-R_2)^2+(G_1-G_2)^2+(B_1-B_2)^2}\,. \tag{1}$$

$$S_{\cos}(\overrightarrow{x_1},\overrightarrow{x_2})=1-\left(\frac{\overrightarrow{x_1}\cdot\overrightarrow{x_2}}{\left\|\overrightarrow{x_1}\right\|\cdot\left\|\overrightarrow{x_2}\right\|}\right)\left(1-\frac{\left\|\overrightarrow{x_1}\right\|-\left\|\overrightarrow{x_2}\right\|}{\max\left(\left\|\overrightarrow{x_1}\right\|,\left\|\overrightarrow{x_2}\right\|\right)}\right). \tag{2}$$

In Eq. (1) and Eq. (2), $\overrightarrow{x_1}=\left(R_1,G_1,B_1\right)^T$ and $\overrightarrow{x_2}=\left(R_2,G_2,B_2\right)^T$ are colour vectors in RGB space.

Based on the observation that the sizes of the character regions in foreground were usually more regular than those in background and noise regions, a text validation measure M was proposed in [11] in order to find the most textual foreground cluster, which was defined as:

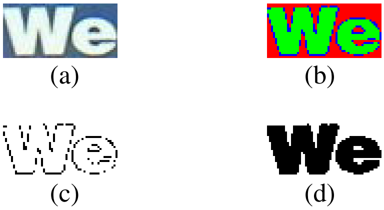
$$M=\sum_{i=1}^{N}\left\|area_i-\frac{1}{N}\left(\sum_{i=1}^{N}area_i\right)\right\|, \tag{3}$$

where N was the number of CCs and $area_i$ referred to the area of the connected component i . The cluster that has the highest pixel occurrence on the image border was chosen as the background. The M values of the remaining two clusters were denoted as M_1 and M_2 respectively. Meanwhile, the M value for the merged cluster of these two clusters was also computed as M_3. The cluster with the smallest value among M_1, M_2 and $M_3$was selected as the text cluster. Between the two binarisation results obtained by 3-means clustering with two distance metrics, the better result was chosen by a step of character segmentation-by-recognition using Log-Gabor filters.

3 The Proposed Method

3.1 Modification of Validation Measure

The text measurement M defined in Eq. (3) provides a method to evaluate the uniformity of different connected components' sizes. However, since it simply sums up the terms each representing the absolute difference between a CC's size and the mean size of all CCs, it does not accurately reflect the uniformity of different sets of CCs' sizes, especially when their sizes are significantly different. Hence, such a definition of M often makes wrong decisions for the selection of text cluster because data in different scales are not comparable. This can be seen in the example shown in Fig. 1.



(a) (b)

(c) (d)

Fig. 1. A comparison between the segmentation result obtained using M metric in [11] and that using the proposed M_{norm} metric with Euclidean Distance. (a) Original image. (b) 3-means clustering result with Euclidean Distance. Here, the green, red and blue colours represent the textual foreground cluster, background cluster and the noise cluster respectively. (c) Segmented binary text (in black) by using M . (d) Segmented binary text (in black) by using M_{norm}.

In Fig. 1(b), the red cluster is the background cluster, the green cluster is the textual foreground cluster and the blue cluster lying around the boundary of the textual foreground cluster is the noise cluster. There are many small CCs in the noise cluster, two big CCs in the textual foreground cluster (in green) and two big CCs in the cluster merging the noise cluster (in blue) and textual foreground cluster (in green). Following the validation measure M and the rules of selecting the text cluster in [11], the noise cluster is judged as the text cluster. Due to the fact that the sizes of small CCs and the sizes of big CCs are not comparable, the sum of the differences as described in Eq. (3) for the cluster having small CCs is less than that in the cluster having big CCs. Therefore, the cluster with small CCs is chosen as the text cluster like the case illustrated in Fig. 1. In this work, after performing 3-means clustering with Euclidean Distance in Eq. (1) and Cosine Similarity in Eq. (2), we modify the definition of the CCA-based validation measure for finding the text cluster and define a new measurement denoted as M_{norm} in Eq. (4).

$$M_{norm} = \sum_{i=1}^{N} \left\| \frac{area_i}{A} - \frac{1}{N}\left(\sum_{i=1}^{N} \frac{area_i}{A} \right) \right\|, \tag{4}$$

where A represents the total area of all CCs in a cluster, N is the number of CCs and $area_i$ refers to the area of the CC i . As defined in Eq. (4), by normalising the areas of each CC, the sizes of the CCs in the clusters with significantly different size become in a same scale. Similar to the selection rule in [11], the cluster with the smallest value of M_{norm} is determined as the text cluster.

3.2 Proposed Segmentation Evaluation Method

A natural scene text image is firstly processed by running 3-means clustering algorithm with two different metrics in Eq. (1) and Eq. (2). Using the proposed measurement M_{norm}, two segmentation results are obtained. Next, objective segmentation evaluation needs to be performed in order to judge the quality of different segmentation results which is feasible in real-time applications. For this purpose, we propose an objective segmentation evaluation method which simultaneously considers intra-region uniformity and inter-region disparity in order to choose the final segmentation result.

The intra-region uniformity refers to the homogeneous property within text or background region, while the inter-region disparity refers to the difference along the border between text and background region. Inspired by [12], we define below a metrics E using the local information of pixels to evaluate the two segmentation results according to the two metrics in Eq. (1) and Eq. (2) respectively.

$$E = \frac{C}{UT + UB}, \tag{5}$$

where C represents the contrast at the border between text and background, UT represents the uniformity of text and UB represents the uniformity of background. C, UT and UB are defined in Eqs. (6), (7) and (8) respectively.

$$C = \frac{\sum_{(i,j)\in C_{bd}} \left(\max\left(D_{(i,j)}^{R}\right) + \max\left(D_{(i,j)}^{G}\right) + \max\left(D_{(i,j)}^{B}\right) \right)}{3 \cdot 255 \cdot N_{bd}}. \tag{6}$$

$$\mathrm{UT} = \frac{\sum_{(i,j)\in C_{t}} \left(\max\left(D_{(i,j)}^{R}\right) + \max\left(D_{(i,j)}^{G}\right) + \max\left(D_{(i,j)}^{B}\right) \right)}{3 \cdot 255 \cdot N_{t}}. \tag{7}$$

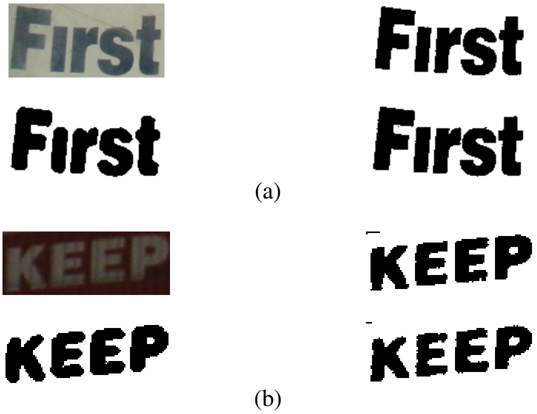
$$\mathrm{UB} = \frac{\sum_{(i,j)\in C_{bk}} \left(\max\left(D_{(i,j)}^{R}\right) + \max\left(D_{(i,j)}^{G}\right) + \max\left(D_{(i,j)}^{B}\right) \right)}{3 \cdot 255 \cdot N_{bk}}. \tag{8}$$

In Eq. (6), C_{bd} is the set of pixels belonging to the border of text and background, and N_{bd} is the number of pixels in C_{bd}. In Eq. (7), C_{t} is the set of pixels belonging to text, and N_{t} is the number of pixels in C_{t}. In Eq. (8), C_{bk} is the set of pixels belonging to background, and N_{bk} is the number of pixels in C_{bk}. In Eq. (6), Eq. (7) and Eq. (8), $\max\left(D_{(i,j)}^{R}\right)$, $\max\left(D_{(i,j)}^{G}\right)$ and $\max\left(D_{(i,j)}^{B}\right)$ refer to the maxima of the differences between the value of pixel at (i, j) and those of its 4-neighbours in R, G and B channels respectively.

Between the two text segmentation results corresponding to the two metrics, the one with the greater E value is chosen as the final segmentation result.

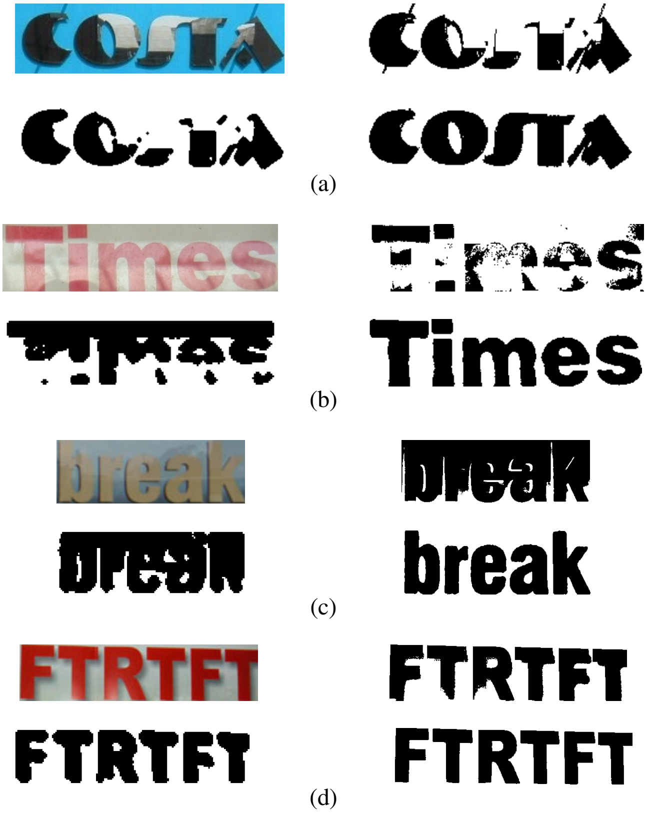
4 Experimental Results

In order to evaluate the performance of the proposed method, the public ICDAR2003 database is used for testing. All test text images are from the folder named "Robust Word Recognition\TrailTest". The classical Otsu's method [13] and a state-of-the-art edge-based binarisation method [7] are implemented for comparing the segmentation results with the proposed method. Some segmentation examples are shown below (from Fig. 2 to Fig. 6).


(a)

(b)

Fig. 2. Comparison of segmentation results in simple cases. Top left: original image; top right: Otsu's method; bottom left: the method in [7]; bottom right: the proposed method.


(a)

(b)

(c)

(d)

Fig. 3. Comparison of segmentation results in uneven lighting cases. Top left: original image; top right: Otsu's method; bottom left: the method in [7]; bottom right: the proposed method.

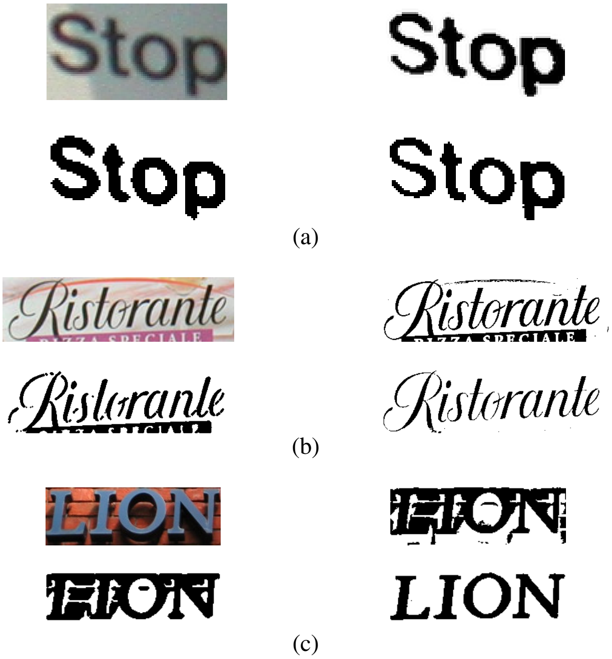


Fig. 4. Comparison of segmentation results in complex background cases. Top left: original image; top right: Otsu's method; bottom left: the method in [7]; bottom right: the proposed method.

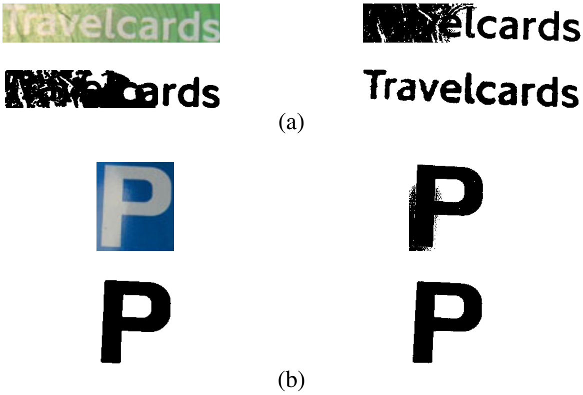
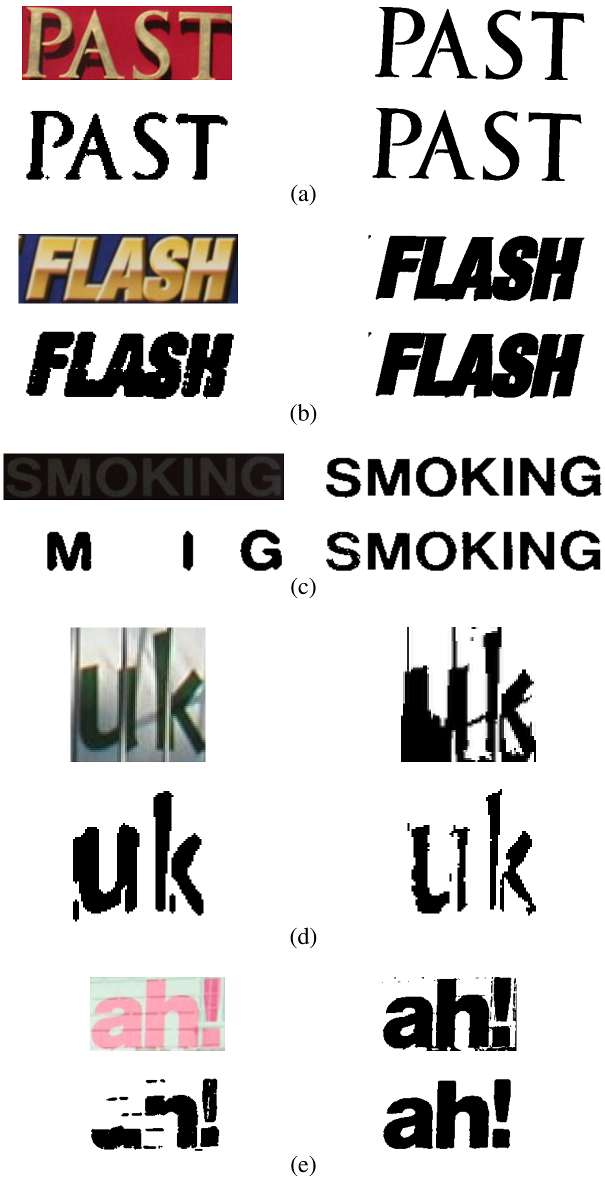


Fig. 5. Comparison of segmentation results in highlight cases. Top left: original image; top right: Otsu's method; bottom left: the method in [7]; bottom right: the proposed method.



(a)

(b)

(c)

(d)

(e)

Fig. 6. Comparison of segmentation results in other cases. Top left: original image; top right: Otsu's method; bottom left: the method in [7]; bottom right: the proposed method.

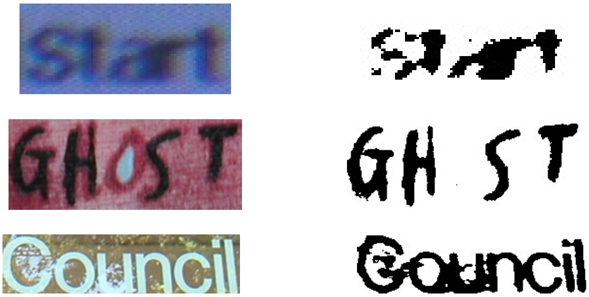


Fig. 7. Some failed results using the proposed method. Left column: original images. Right column: segmentation results by the proposed method.

Fig. 8. Some removed text images from ICDAR2003 dataset

In each example from Fig. 2 to Fig. 6, the top left is the original image, the top right is the segmentation result using Otsu's method, the bottom left is the segmentation result using the edge-based binarisation method [7], and the bottom right is the segmentation result using the proposed method.

As shown in the examples above, the proposed method can deal with uneven lighting (e.g., Fig. 3), complex background (e.g., Fig. 4) and highlight (e.g., Fig. 5), and the performance is better than the two other methods. For the simple cases as shown in Fig. 2, the proposed method gets as good segmentation result as the other two methods. The use of colour information can describe the feature of colour text image and make the segmentation less sensitive to uneven lighting, compared with the edge-based methods. This helps keep the integrity of text characters as shown in Fig. 3(d). The proposed validation measure provides robust segmentation from complex background and shadow as shown in Fig. 4(c). In Fig. 6(b), the text has multiple similar colours in contrast to that of the background, and the clustering operation can group the text pixels into the same cluster based on their colour information. In Fig. 6(c), both the Otsu's method and the proposed method can acquire a good segmentation of the text with low contrast, while the edge-based method misses four letters due to the fact that there is no seed generated within the edges of the missing letters.

Fig. 7 shows some failed segmentation results by using the proposed method. The failures are caused by blurring, colour ununiform of characters and the background pixels with similar colour information of text.

After removing some text images having very small sizes (less than 20 pixels in height or width), very low readability or hollow character strokes, such as the examples in Fig. 8, 1042 text images containing 5112 characters are used for comparing the OCR recognition performances of different methods. OmniPage Professional 17 OCR software is used for recognition. According to our sexperiments, the recognition rate of the original images without segmentation is only 0.548%, which shows the necessity of segmentation. When the segmented text images are obtained by using Otsu's method [13], the edge-based method in [7] and the proposed method respectively, the statistical results are compared and listed in Table 1.

Table 1. Comparison of character recognition rate with state-of-the-art approaches

Methods	Recognition Rate
Otsu's method [13]	73.533%
Edge-based method in [7]	60.446%
Proposed method	80.008%

The comparison of recognition rates in Table 1 shows that the proposed colour-based segmentation method outperforms the classical Otsu's method [13] and a most recent method in [7] for natural scene text segmentation. Using colour information can help deal with the effect of uneven lighting and highlights. Whereas, the edge-based method in [7] cannot extract the edge information of text correctly and hence cannot achieve good segmentation and recognition.

5 Conclusion

This paper presents a colour-based image segmentation method which consists of three steps: 3-means clustering with two distance metrics, modified CCA-based validation measure and a segmentation evaluation method. The clustering makes use of R, G and B information as the feature vector for every pixel to cluster the pixels of the text image into three groups by using both Euclidean Distance and Cosine Similarity as distance metrics. A new validation measure is proposed based on normalized size of connected components instead of their absolute values. Finally, the better one of the two segmentation results is selected using the proposed segmentation evaluation method when taking into account the intra-region uniformity and the inter-region disparity. A better segmentation evaluation method for better segmentation results will be investigated in our future research.

References

1. Jung, K., Kim, K.I., Jain, A.K.: Text Information Extraction in Images and Video: A Survey. Pattern Recognition 37(5), 977–997 (2004)
2. Kasar, T., Kumar, J., Ramakrishnan, A.G.: Font and Background Color Independent Text Binarization. In: 2nd CBDAR, pp. 3–9 (2007)
3. Kasar, T., Ramakrishnan, A.G.: COCOCLUST: Contour-based Color Clustering for Robust Binarization of Colored Text. In: 3rd CBDAR, pp. 11–17 (2009)
4. Yu, J., Huang, L., Liu, C.: Double-edge-model Based Character Stroke Extraction from Complex Backgrounds. In: 19th ICPR 2008, pp. 1–4 (2008)
5. Ye, X., Cheriet, M., Suen, C.Y.: Stroke-model-based Character Extraction from Gray-level Document Images. IEEE Transactions on Image Processing 10(8), 1152–1161 (2001)
6. Li, X., Wang, W., Huang, Q., Gao, W., Qing, L.: A Hybrid Text Segmentation Approach. In: ICME, pp. 510–513 (2009)

7. Zhou, Z., Li, L., Tan, C.L.: Edge Based Binarization for Video Text Images. In: 20th ICPR, pp. 133–136 (2010)
8. Yokobayashi, M., Wakahara, T.: Segmentation and Recognition of Characters in Scene Images Using Selective Binarization in Color Space and GAT Correlation. In: ICDAR, vol. 1, pp. 167–171 (2005)
9. Yokobayashi, M., Wakahara, T.: Binarization and Recognition of Degraded Characters Using A Maximum Separability Axis in Color Space and GAT Correlation. In: 18th ICPR, vol. 2, pp. 885–888 (2006)
10. Thillou, C.M., Gosselin, B.: Color Text Extraction from Camera-based Images: the Impact of the Choice of the Clustering Distance. In: ICDAR, vol. 1, pp. 312–316 (2005)
11. Thillou, C.M., Gosselin, B.: Color Text Extraction with Selective Metric-based Clustering. Computer Vision and Image Understanding 107(1-2), 97–107 (2007)
12. Correia, P.L., Pereira, F.: Objective Evaluation of Video Segmentation Quality. IEEE Transactions on Image Processing 12(2), 186–200 (2003)
13. Otsu, N.: A Threshold Selection Method from Gray-level Histograms. IEEE Transactions on Systems, Man & Cybernetics SMC-9(1), 62–66 (1979)

Recognizing Natural Scene Characters by Convolutional Neural Network and Bimodal Image Enhancement

Yuanping Zhu[1], Jun Sun[2], and Satoshi Naoi[2]

[1] Department of Computer Science, Tianjin Normal University, Tianjin, China
zhuyuanping@mail.tjnu.edu.cn
[2] Fujitsu R&D Center Co. Ltd., Beijing, China
{sunjun,naoi}@cn.fujitsu.com

Abstract. In this paper, a natural scene character recognition method using convolutional neural network(CNN) and bimodal image enhancement is proposed. CNN based grayscale character recognizer has strong tolerance to degradations in natural scene images. Since character image is bimodal pattern image in essence, bimodal image enhancement is adopted to improve the performance of CNN classifier. Firstly, a maximum separability based color-to-gray method is used to strengthen the discriminative power in grayscale image space. Secondly, grayscale distribution normalization based on histogram alignment is performed. Through increasing the data consistency among grayscale training and test samples, it leads to a better CNN classifier. Thirdly, a shape holding grayscale character image normalization is adopted. Based on these measures, a high performance natural scene character recognizer is constructed. The recognition rate of 85.96% on ICDAR 2003 robust OCR dataset is higher than existing works, which verified the effectiveness of the proposed method.

Keywords: natural scene character recognition, convolutional neural network, color-to-gray, bimodal image enhancement.

1 Introduction

Optical character recognition(OCR) has been researched for a long term. Traditional OCR focuses on scanned documents. Since foreground and background are easy to be segmented, high accuracy has been obtained by binary character recognition. Recent years, more and more OCR requests come from camera based images, web images and natural scene images. In these images, characters usually have no clear foreground or clean background. Furthermore, degradations such as low resolution, blurring, distortion, luminance variance, complicate background etc. often appear. All these factors make a great challenge in character recognition of natural scene images. On such degradations, obtaining not bad binarization is difficult, let alone tolerance to distortion. Thus, no good result of traditional OCR on natural scene character recognition is expected.

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 69–82, 2012.

Natural scene character recognition and degraded character recognition become active topics gradually. Most of character recognition methods managed to improve on traditional OCR [4-9]. Weinman et al.[4] proposed a scene text recognition method. His emphasis was to combine lexicon information to improve scene text recognition result. Yokobayashi et al.[5][6] utilized a GAT (Global Affine Transformation) correlation method to improve binary character recognition's tolerance to the distortions of scaling, rotation and shearing. GAT method was much time-consuming. Recognizing degraded character using grayscale feature seems to be a more effective way [9]. Campos [11] utilized a bag-of-visual-word based method to recognize natural scene character. Six local descriptor based character features were compared. Although some of them were better than traditional OCR, the results still showed the difficulty in this field.

CNN has strong tolerance to shift, scale and distortion. It has shown good performance on handwriting digit recognition [2][3] and been applied to other recognition tasks [14][15]. Some researchers applied CNN on degraded character recognition and obtained promising results. Saidane [12] proposed a CNN based scene text recognition method. Through recognizing character on color image directly, it outperformed the binary character recognition method. On a same test set with [5][6], Saidane's method got better result. But its classifier output didn't discriminate upper case and lower case alphabets. Moreover, although classification without many preprocessing of input images made training easy, there was room to improve for CNN based recognition. To recognize camera based documents, Jacobs [13] also proposed a CNN based text recognition method. Its CNN based character recognizer worked on grayscale character images and outperformed the binary character recognition based commercial OCR software on low-resolution documents captured by camera.

Avoiding information loss in binarization, grayscale images and color images contain more helpful information for classification. For low quality character images, recognition on grayscale or color image is necessary. In essence, character is a bimodal pattern image consisting of foreground and background. Enhancing discriminative information in this pattern will be beneficial to recognition and should be considered.

Based on the above motivation, a natural scene character recognition method based on convolutional neural network and bimodal image enhancement is proposed in this paper. The CNN character recognizer works on grayscale images. Since most of natural images are color images, a maximum separability based color-to-gray method is proposed to enhance discriminative information when convert color images to grayscale images. In order to increase data consistency among training and test samples, grayscale distribution normalization based on histogram alignment is proposed. Before grayscale characters are input into CNN classifier, a shape holding grayscale character image normalization is applied. Experiments on the public ICDAR 2003 Robust OCR dataset [1] are used to evaluate the performance of the proposed method.

The rest of paper is organized as follows. Section 2 describes the details of proposed natural scene character recognition method. Section 3 gives the experiment results including the comparison with other existing methods. Finally, some conclusions and discussions are given in section 4.

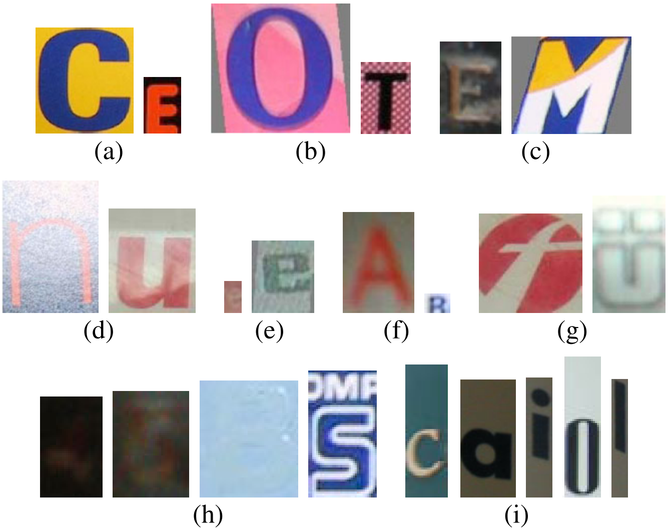
2 Natural Scene Character Recognition

2.1 Framework

ICDAR 2003 Robust OCR dataset [1] is a public dataset for research on natural scene text detection and recognition. Because of the complicate images, it is a very tough task for character recognition. Our work is carried out on this dataset. Its “Train” and “Test” subsets are adopted as our training set and “Sample” subset is adopted as our test set. To describe the degradations, Yokobayashi et al [5][6] classified the samples of ICDAR dataset into seven groups according to the degree of image degradations and background complexity, as shown in Fig. 1(a)-(g). In Fig. 1(h) and (i), some examples of bad samples and inaccurate segmentation are shown. The bad examples are hardly to be recognized even by human. The inaccurate segmentation is a practical problem in natural scene text detection and its influence to grayscale character recognition will be discussed in section 2.

As illustrated in Fig.2, our natural scene character recognition has four major stages:

- Color-to-Gray conversion
- Grayscale distribution normalization
- Grayscale character image normalization
- Grayscale character recognition using CNN classifier



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Fig. 1. Examples in ICDAR 2003 Robust OCR dataset. (a) Clear; (b) Background design; (c) Multi-color character; (d) Non-uniform lighting; (e) Little contrast; (f) Blurring; (g) Serious distortion; (h) Bad samples; (i) Inaccurate character segmentation.

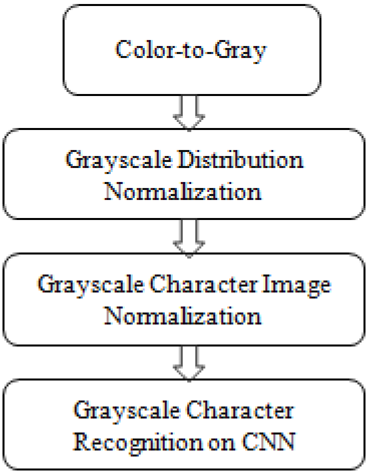

Color-to-Gray
Grayscale Distribution Normalization
Grayscale Character Image Normalization
Grayscale Character Recognition on CNN

Fig. 2. Workflow of the proposed natural scene character recognition

Firstly, color images should be converted to grayscale images. In essence, character is a bimodal pattern image consisting of foreground and background. The distinct difference between foreground and background is beneficial to final character recognition.

Ordinary color-to-gray methods perform conversion on all images under a fixed way like:

$$Gray = (R + G + B)/3 \tag{1}$$

or

$$Gray = 0.301R + 0.586G + 0.113B \tag{2}$$

Not considering the specific situations of characters, they fail to produce distinct difference between foreground and background in many cases. A typical example is that a uniform grayscale image is obtained using (1), when a red character surrounds by blue background and red luminance is equal to blue luminance.

Considering character is a bimodal pattern image, we hope to push foreground and background away from each other after grayscale conversion. Therefore, we utilize a maximum separability criterion to find an optimal color-to-gray conversion which separate foreground and background best in grayscale image.

Secondly, grayscale distribution normalization based on histogram alignment is performed to increase the data consistency or homogeny among training and test samples. As we all know, machine learning based recognition are always closely related to the data distribution. The data consistency among training and test samples has significant influence to the recognition accuracy. For grayscale character recognition, if training and test samples have similar grayscale distribution, better recognition is expected. For this purpose, histogram alignment is applied.

Thirdly, a shape holding grayscale character image normalization method is proposed. The character images need to be normalized to the required size of net classifier. Traditional character image normalization methods are designed for binary images or images with clean and approximate white background. Many natural scene images dissatisfy that condition and will produce the white strips around the characters after normalization. By reversed color judgment and background filling, the proposed method overcomes that problem and preserves the bimodal pattern of foreground and background in normalization.

Finally, CNN classifier accepts the grayscale character images after above processing and produces character recognition results.

2.2 Maximum Separability Color-to-Gray Conversion

The goal of our color-to-gray conversion is to help grayscale character recognition. This is the distinct difference from other tasks. Since character is a bimodal pattern in general, we hope discriminative information between foreground and background can be preserved after color-to-gray conversion. In other words, we should maximize the separability between foreground and background in grayscale images.

Projecting the RGB coordinates to an axis can realize color-to-gray conversion. Then, color-to-gray is converted to the problem of seeking optimal projection axis. Comparing with [6] to decide binary pixels, utilizing maximum separability in color-to-gray has much smaller risk.

Between-class variance in grayscale image represents the separability. If foreground and background are labelled, the axis with maximal between-class variance is easy to be found by Fisher criterion. Because precise labelling is impossible, we just utilize a binarization to approximately separate foreground and background. Here, we adopt Otsu[16]. In fact, between-class variance can be obtained from Otsu method directly.

We sample the axis parameters in spherical coordinates *(ρ,θ,φ)*. The origin is the center of RGB cube. Let *A(θ,φ)* be a projection axis in sphere space. *Gray(θ,φ)* is the converted grayscale image under the projection axis *A(θ,φ)*. $S_b(\theta,\varphi)$ denotes the between-class variance of *Gray(θ,φ)*. The objective function is:

$$A^* = \arg\max_{A} S_b(\theta,\varphi) \tag{3}$$

The full space search is heavy time-consuming. To reduce the search time, we constrain the search in a part of entire space. Defining an initial axis, we just search in the neighbourhood space around the initial axis.

The initial axis is defined by bi-class color clustering axis. However, color clustering cannot separate foreground and background well for many complicate cases. In experiment, we found that the diagonal axis of RGB cube is a good complement of bi-class clustering axis. Therefore, we adopt both of them to construct a dual-initial-axis search as Figure 3 illustrated. It has very close evaluation to full space search while it speeds up more than tens of times. The detail comparison data can be found in experiment result section.

The algorithm is summarized as follows.

Algorithm of maximum separability color-to-gray

```
Step 1: Initial axes location
  RGB cube diagonal axis and bi-class clustering axis
are:
   $A_0 = (\theta_0, \varphi_0); A_1 = (\theta_1, \varphi_1);$
Step 2: Convert RGB to spherical coordinates
Step 3: Optimal project axis search
  The search range is limit to
   $(\theta, \varphi) \in \{(\theta_0 \pm \Delta\theta), (\varphi_0 \pm \Delta\varphi)\}$
            $\bigcup \{(\theta_1 \pm \Delta\theta), (\varphi_1 \pm \Delta\varphi)\}$
Step 4: Perform color-to-gray conversion on optimal axis
```

Figure 4 shows comparative results between proposed method and formula (1). The grayscale images are scaled to fit the classifier, so we can observe the grayscale images to be input into the classifier. The new method yields much clearer characters than normal color-to-gray. It is no doubt that the better recognition results are expected.

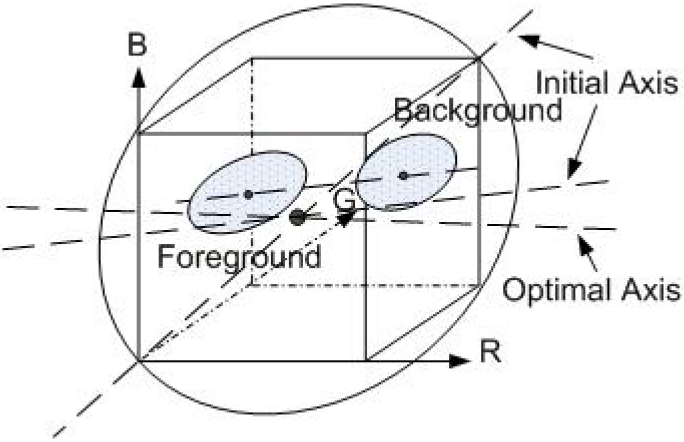
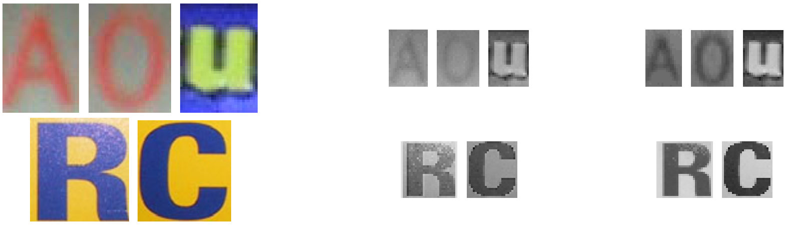


Fig. 3. Maximum Separability based Color-to-Gray



(a) (b) (c)

Fig. 4. Color-to-Gray examples. (a) color images; (b) results of formula (1); (c) results of maximum separability based color-to-gray.

2.3 Grayscale Distribution Normalization

In traditional character recognition, character normalization of size, distortion is effective to improve recognition because it increases the data consistency among training and test samples. As for grayscale character recognition, grayscale distribution is an additional factor. Especially for CNN classifier, grayscale gradient magnitude and distribution is significant.

Histogram represents grayscale distribution of images. To increase the data consistency among training and test samples, sample images are expected to be transformed to a uniform grayscale distribution. Without sample class labels known in advance, only category-independent global distribution information can be used, such as grayscale center, range, and variance and so on. All samples follow the same transformation. Thus, histogram center and range are aligned to approximate to uniform parameters, where center means average grayscale. Histogram alignment transformation is given by:

$$a(x) = s * (x - c) + C \tag{4}$$

Where s is the grayscale range scaling factor calculated with $\gamma = 2.0$:

$$s = \begin{cases} \dfrac{1.0}{2.0/(1+e^{-\gamma(l/L-1)})}, & if\ L/l > 1; \\ \dfrac{2.0}{1+e^{-\gamma(L/l-1)}}, & if\ L/l < 1; \end{cases} \tag{5}$$

The uniform histogram parameters including range L and center C are learned from the training set. Considering the unbalanced sample numbers of categories in ICDAR dataset, the above parameters are computed based on category dependent intermediate parameters.

In image transformation, a sine function based histogram transformation is applied to enhance bimodal histograms of character images. It pushes the histogram peaks of foreground and background away from each other while keeping the grayscale range as illustrated in Fig. 5. The transformation formula is:

$$f(x) = \frac{l}{2} * \{1 + \frac{1}{\sin(\alpha * \pi / 2)} * \sin(\alpha\pi(\frac{x}{l} - \frac{1}{2}))\} \tag{6}$$

Where l is grayscale level range, and α is to control the enhancement strength, 0.9 is set.

Then, final grayscale distribution normalization is:

$$h(x) = a(x) \bullet f(x) \tag{7}$$

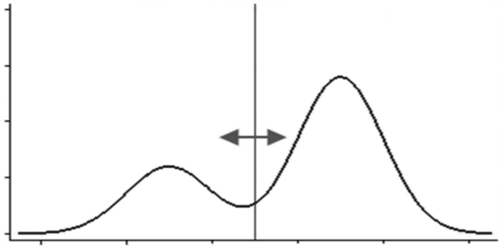


Fig. 5. Bimodal histogram enhancement

2.4 Shape Holding Grayscale Character Image Normalization

Our CNN classifier takes a 29*29 image as the input. All input character images should be normalized to this size firstly. Designed for binary images, traditional character normalization methods cannot work well on natural scene characters.

In natural scene images, characters are not easy to be segmented accurately as examples in Fig. 1(i). In these cases, severe shape distortion happens if normalize aspect ratio to 1.0, so as to those narrow letters like 1, I etc. Shape holding normalization is required. But white strips will appear if we put scaled images into 29*29 boxes directly. The sharp edge effect has negative influence on net classifier training and test. Therefore, background filling should be considered in normalization. Fig. 6 shows some examples and the shape holding grayscale character image normalization method is described as follows:

Algorithm of shape holding grayscale character image normalization

```
Step 1: Reversed color judgment
  Comparing border pixels' gray mean with the whole
image's gray mean, the images with lower light of border
are determined as reversed color images and reverse them.
Step 2: Background estimation
  The largest gray of border pixels is selected as the
background gray.
Step 3: Image normalization with background filling
  After scaled character image is put into the 29*29 box,
white strips are filled by the background gray value.
```

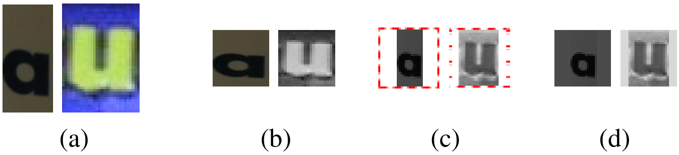


Fig. 6. Grayscale character image normalization. (a) original images; (b) non-shape-holding normalization; (c) normalization without background filling; (d) normalization with background filling.

2.5 Convolutional Neural Network

Due to the ideas of local receptive fields, share weights and sub-sampling, convolutional neural network classifier ensure robustness on shift, scale and distortion. From the view of classifier architecture, it more like a classifier combined with a learning based feature extractor. Convolutional neural network has shown best result of handwriting digit recognition in MNIST benchmark[17]. Its strong robustness on distortion and degradations attracts us to choose it as the natural scene character recognizer.

Simard's CNN[3] is effective and efficient. We adopt its architecture. As shown in Fig. 7, it takes 29*29 grayscale character image as input. A total of 62 characters including 10 numerals and 52 alphabets are used in the classifier construction. In this convolutional neural network, the first two convolutional layers are more like feature extractors working on different resolutions and 5 and 50 feature maps are chosen. The fully connected layer or hidden unit layer contains 150 units.

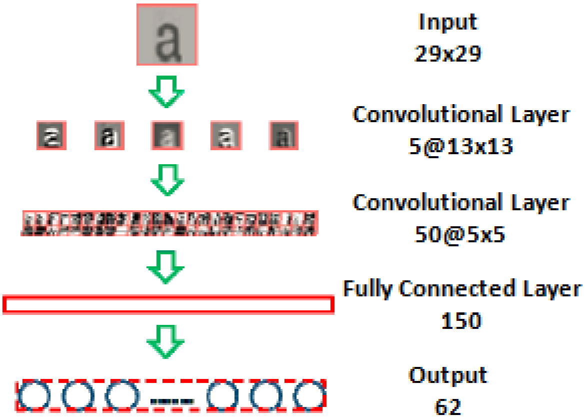


Fig. 7. Convolutional neural network architecture

Our neural network classifier is trained on ICDAR Robust OCR 2003 data. In order to get enough training samples and compare with the work in [12] on same conditions, we choose both the "Train" and "Test" subsets(a total of 11615 images) in network training as [12].

Since natural scene images are usually captured by camera, perspective distortion is one of the most important degradations. Random perspective distortion model is applied on original samples to generate synthetic samples. Moreover, Gaussian noise is added in synthetic sample generation. The final training set is expanded by double size.

3 Experiment Result

As mentioned before, our experiments are based on ICDAR 2003 Robust OCR dataset[1]. The "Sample" subset is used as test set. The entire "Sample" subset has 854 character images, 851 for alphabets and numerals. Several previous works [5][6][12] tested on selected 698 images of "Sample" subset. Bad samples as Fig. 1(h) shown are discarded. The remained 698 images are classified into seven groups as Fig. 1(a)-(g) shown and Table 1 gives image numbers of each group. For method evaluation and comparison, we also test our method on this 698-image set.

Before the recognition experiment, to evaluate the results of maximum separability color-to-gray, Table 2 gives the comparison in training&testing sample set among the proposed method and conventional color-to-gray methods which were labeled as CG1 to CG4. The average variance is used to evaluate discriminative power in grayscale images. The fast search range is limited to [-10, +10] in degree around two initial axes compared with [-90, +90] for full search. One degree is selected as interval. θ and φ use the same search range. The maximum separability color-to-gray is significantly superior to conventional color-to-gray methods. The full space search is heavy time-consuming while the best result can be ensured. The fast version has results very close to the full space search and speeds up about 40 times, whose processing time for each sample is reduced from 4573ms to 119ms. Anyway, its computation cost is much higher than conventional methods CG1 & CG2. The code optimization is required. Further information of their influence to character recognition can be found in below experiment.

Table 3 compares the proposed method with three existing methods[5][6][12] denoted by M1 to M4. Both the methods in [5][6] are binary character recognition methods. The basic binary character recognition method can only obtain recognition rate 70.3%, which shows the difficulty of tradition OCR for this task. GAT (Global Affine Transformation) helps to significantly improve it to 81.9%.

But, with the CNN's robustness to degradations, [12] outperforms the binary character recognition methods easily by 84.53%. Because its classifier runs on color image directly, we label it colorCNN here. Our CNN classifier recognizes grayscale characters, which is labeled as grayCNN. The recognition rate of 85.96% verifies the effectiveness of the proposed method. It also indicates that binary information is not enough for the tough task of natural scene character recognition. Recognizing natural scene characters on grayscale or color image is more appropriate.

The results of our CNN classifiers under other situations(M5 to M7) are also given in Table 3. If rely on grayscale character recognizer of CNN with conventional color-to-gray method CG2, the accuracy of 84.67% is obtained. This result has no obvious difference with colorCNN in [12], which means grayscale information works well for most of cases in natural scene character recognition. With the help of character image enhancement, the accuracy rise to 85.96%, increased by 1.29 percent. If we analyze each technique's contribution separately, dual-initial-axis maximum separability color-to-gray helps to increase 0.86 percent in accuracy while grayscale distribution normalization helps to increase 0.57 percent. These results verify the effectiveness of each part.

Table 1. Classification of images in ICDAR2003 Robust OCR sample dataset

Group	**Number of Images**
Clear	199
Background design	130
Multi-color character	54
Non-uniform lighting	40
Little contrast	37
Blurring	210
Serious distortion	28
Total	698

Table 2. Color-to-Gray result comparison

Color-to-Gray method	**Average variance**		**Time(ms)**
	Training set	**Test set**	
CG1: Formula 1	40.7	34.6	0.26
CG2: Formula 2	42.1	36.5	0.29
CG3: Maximum Separability color-to-gray with dual-initial-axis search	44.9	39.8	119
CG4: Maximum Separability color-to-gray with full space search	45.9	41.3	4573

Table 3. Character recognition result comparison

Method	**#Error**	**Accuracy**
M1: Bin(Method in [5])	207	70.3%
M2: Bin+GAT(Method in [6])	126	81.9%
M3: colorCNN(Method in [12])	108	84.53%
M4: Proposed method (grayCNN+CG3+GDN)	98	85.96%
M5: grayCNN+CG2	107	84.67%
M6: grayCNN+CG3	101	85.53%
M7: grayCNN+GDN	103	85.24%

GDN, GAT. Grayscale distribution normalization and Global affine transformation.

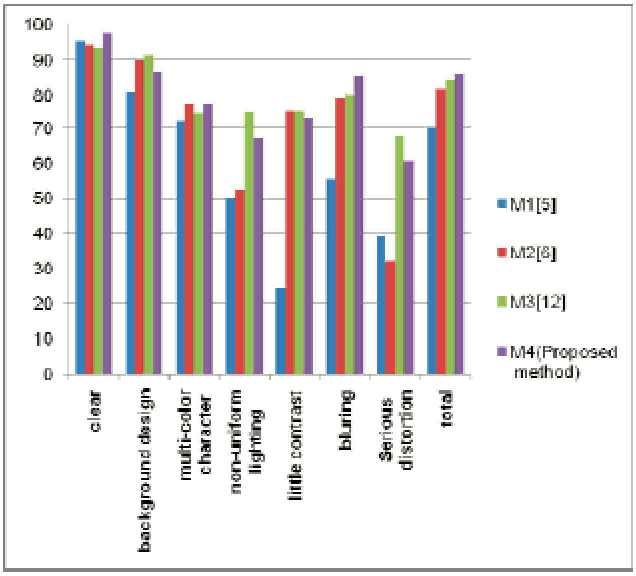


Fig. 8. Character recognition rate comparison

Fig. 8 shows recognition rates for each group of four methods'. Our method's recognition rates range from 60.71% for serious distorted images to 97.49% for clear images. This result is encouraging. Especially for clear images, 97.49% is a high enough accuracy even for some practical applications. It can also be observed that our method yields good results on blurring images. Bimodal image enhancement seems more effective for these two degradations comparing with other methods.

4 Discussions and Conclusions

In this paper, a natural scene character recognition method using convolutional neural network and bimodal image enhancement is proposed. CNN classifier shows robustness on severe degradations of natural scene character samples in ICDAR 2003 dataset. The evaluation results also verify that the enhancement of bimodal character image is beneficial to grayscale character recognition. The proposed maximum separability based color-to-gray conversion method is able to preserve more discriminative information than normal color-to-gray methods, which increases the classification accuracy obviously. By increasing data consistency and enhancing character images, grayscale distribution normalization can also improve CNN classifier, but its contribution is lower than expectation. One of possible reasons is that noises also get strengthened after the normalization in some samples. Another possible reason is too complicated data set. In a word, the recognition rate of 85.96% is higher than previous works in the same dataset, which verify the effectiveness of the proposed method.

As a future work, we will seek a proper way to enhance images and depress their noises at the same time. Furthermore, because accurate character segmentation is hardly to be guaranteed in real application of natural scene text recognition, combining lexicon information to reduce segmentation and recognition errors will be our next work

References

1. Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: ICDAR 2003 robust reading competitions. In: 7th International Conference on Document Analysis and Recognition, Edinburgh, Scotland, vol. 2, pp. 682–687 (2003)
2. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. IEEE 86, 2278–2324 (1998)
3. Simard, P.Y., Steinkraus, D., Platt, J.C.: Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. In: 7th International Conference on Document Analysis and Recognition, pp. 958–962 (2003)
4. Weinman, J.J., Learned-Miller, E., Hanson, A.R.: Scene Text Recognition Using Similarity and a Lexicon with Sparse Belief Propagation. IEEE Trans. on Pattern Analysis and Machine Intelligence 10(31), 1733–1746 (2009)
5. Yokobayashi, M., Wakahara, T.: Segmentation and recognition of characters in scene images using selective binarization in color space and gat correlation. In: 8th International Conference on Document Analysis and Recognition, vol. 1, pp. 167–171 (2005)
6. Yokobayashi, M., Wakahara, T.: Binarization and recognition of degraded characters using a maximum separability axis in color space and gat correlation. In: 18th International Conference on Pattern Recognition, Hongkong, China, vol. 2, pp. 885–888 (2006)
7. Clark, P., Mirmehdi, M.: Recognising text in real scenes. International Journal Document Analysis and Recognition 4(4), 243–257 (2004)
8. Chen, D., Odobez, J., Bourlard, H.: Text detection and recognition in images and video frames. Pattern Recognition 3(37), 595–608 (2004)
9. Kopf, S., Haenselmann, T., Effelsberg, W.: Robust character recognition in low-resolution images and videos. Technical report, Department for Mathematics and Computer Science, University of Mannheim (2005)
10. Sun, J., Hotta, Y., Katsuyama, Y., Naoi, S.: Camera based Degraded Text Recognition Using Grayscale Feature. In: 8th International Conference on Document Analysis and Recognition, pp. 182–186 (2005)
11. de Campos, T., Babu, B., Varma, M.: Character Recognition in Natural Images. In: International Conference on Computer Vision Theory and Applications, Lisbon, Portugal (2009)
12. Saidane, Z., Garcia, C.: Automatic scene text recognition using a convolutional neural network. In: 2nd International Workshop on Camera-Based Document Analysis and Recognition, pp. 100–106 (2007)
13. Jacobs, C., Simard, P.Y., Viola, P., Rinker, J.: Text Recognition of Low-resolution Document Images. In: 8th International Conference on Document Analysis and Recognition (ICDAR 2005), pp. 695–699 (2005)

14. Deng, H., Stathopoulos, G., Suen, C.Y.: Error-Correcting Output Coding for the Convolutional Neural Network for Optical Character Recognition. In: 10th International Conference on Document Analysis and Recognition, Barcelona, Spain, pp. 581–585 (2009)
15. Garcia, C., Delakis, M.: Convolutional Face Finder: A Neural Architecture for Fast and Robust Face Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(11), 1408–1423 (2004)
16. Otsu, N.: A Thresholding Selection Method from Gray-level Histogram. IEEE Transactions on System, Man, and Cybernetics 9(1), 62–66 (1978)
17. LeCun, Y.: The MNIST database of handwriting digits, `http://yann.lecun.com/exdb/mnist`

PaperUI

Qiong Liu and Chunyuan Liao

FX Palo Alto Laboratory
3400 Hillview Avenue, Bldg.4
Palo Alto, California, U.S.A.
{liu,liao}@fxpal.com

Abstract. PaperUI is a human-information interface concept that advocates using paper as displays and using mobile devices, such as camera phones or camera pens, as traditional computer-mice. When emphasizing technical efforts, some researchers like to refer the PaperUI related underlying work as interactive paper system. We prefer the term PaperUI for emphasizing the final goal, narrowing the discussion focus, and avoiding terminology confusion between interactive paper system and interactive paper computer [40]. PaperUI combines the merits of paper and the mobile devices, in that users can comfortably read and flexibly arrange document content on paper, and access digital functions related to the document via the mobile computing devices. This concept aims at novel interface technology to seamlessly bridge the gap between paper and computers for better user experience in handling documents. Compared with traditional laptops and tablet PCs, devices involved in the PaperUI concept are more light-weight, compact, energy efficient, and widely adopted. Therefore, we believe this interface vision can make computation more convenient to access for general public.

Keywords: paper document, camera phone, human-computer interface, human-computer interaction, document recognition, augmented paper, vision-based paper interface.

1 Introduction

Let's think about an ideal reading device without restricting by state-of-the-art technologies. Is it iPad? In our mind, our ideal reading device should be more interactive than iPad. Moreover, it should be much larger than iPad (for comfortable reading), much smaller than iPad (for portability), much lighter, much cheaper, can be merged or separated easily, and have better readability. We believe that PaperUI is a step toward this goal.

After its invention several thousand years ago, paper has become an essential part of our daily life. Paper production provides a cost efficient way to use wood mill wastes. Additionally, paper can be recycled or decomposed much easier than most plastics or electronic-devices. According to the statistics published by woodconsumption.org [9], the world produces approximately 300 million tons of paper each year and every US office employee generates approximately 9,999 more paper sheets each year. All these

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 83–100, 2012.

facts indicate that paper is one of the most widely used communication media and will continue to be used in our daily life for a very long time.

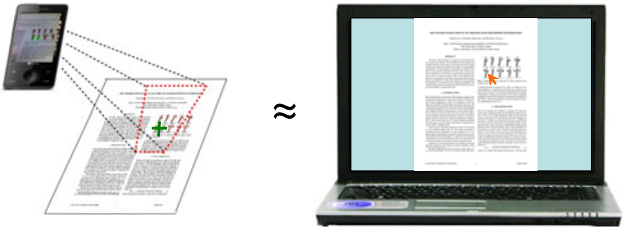


Fig. 1. PaperUI vs. Traditional Laptop UI

On the other hand, technology advances in computer/consumer-electronic industry and chemical industry enable many novel products that can replace or surpass traditional paper products. These new exciting products stimulate people to imagine paperless office from 40 years ago [39]. However, many e-products finally stimulate paper usage and create more e-wastes and plastic wastes beyond paper in the past 40 years.

To make more efficient use of paper products as well as novel electronic devices, new technologies are demanded to combine the merits of both paper and electronic devices. PaperUI is an initial attempt to address this issue. Figure 1 illustrates the PaperUI concept vs. a traditional laptop UI. The PaperUI concept advocates using paper as static content displays for its wide adoption, good readability, flexible display size, flexible display arrangement, robustness, energy efficiency, light-weight, and less long term hazards. It also advocates dynamic human-information interaction via light-weight and energy efficient electronic devices for its portability, fast warm-up time, low energy cost and low long-term e-waste hazards. Through properly balancing information displayed on paper and dynamic mobile display, PaperUI may also help us to reduce hardcopies without sacrificing convenience or changing working habits.

The PaperUI concept is not a concept that comes from nothing. Many digital function designs of this interface come from existing computer interfaces. Therefore, it can provide most digital functions available on state-of-the-art desktop or laptop computers. Different from existing computer interfaces that provides an exclusive working space (paperless working space), the PaperUI interface tries to cooperate with the traditional paper interface to benefit both. This tight cooperation seamlessly merges the physical working space and digital working space. It can save users' efforts for switching back and forth between two different working spaces. It can also save energy by not using power for displaying most static contents.

PaperUI is an add-on interface for existing paper interface widely adopted by people for many centuries. Because it is an add-on interface, anyone can selectively use this interface. When people choose to use the PaperUI interface, they can get state-of-the-art digital functions beyond traditional paper interface. When they choose not to use the PaperUI interface, they still can use the traditional paper interface as usual.

In the following sections, we first present PaperUI overview, then we present emerging technologies that may be helpful to the PaperUI implementation, followed by research prototypes and applications that align with the PaperUI concept. We end the paper with discussions of future work.

2 PaperUI Overview

PaperUI can be situated in the design space of interactive paper systems. Early interactive paper system research such Digital Desk [41], achieves the augmentation of paper by using fixed cameras above a desk to track documents as well as gestures. Because this setup has fixed camera and desk, the mobility of this system is very limited. This fixed camera setup also restricts efficient usage of the camera resolution. Moreover, that system has to use powerful projectors to cover the whole desktop for usable feedback. Different from early interactive paper systems, the PaperUI concept offers more considerations to system mobility, interaction resolution, and energy efficiency.

Different from interactive paper computer [40], which is a film-like electronic device, PaperUI only uses mobile electronic device as a small interactive window, and does not extend electrical wires to the whole paper surface. This setup concept can help the PaperUI interface to reduce energy cost and e-waste. If the interactive paper computer can finally be used as the small interactive window by the PaperUI interface, the energy and e-waste reduction is expected to be more significant.

To improve mobility, researchers use Anoto's digital pen [42] or similar tracking device [43] to interact with paper. Early application explorations of the pen technology use pen path tracking to fill digital forms [44], use pen position on a piece of paper to activate sound corresponding to that position [45], use pen gesture tracking to capture handwritten notes and activate digital functions [48], or even use pen position and pen gesture to manage meeting capture and activate room controls [47]. These explorations may be considered as early prototypes of the PaperUI concept. Even though digital pen and similar tracking devices have better mobility than fixed systems, they do require users to carry a special device for the interaction. This special requirement may hinder wide adoption of this pen based technology. Another major drawback of these early prototypes is that they do not provide much active visual feedback. This drawback severely limits the digital functions that can be offered.

To overcome this extra device barrier and visual feedback barrier, researchers start to develop PaperUI technologies that can enable user-paper interaction via widely adopted smartphones. Beyond the wide adaptability, the smartphone-based interface has an extra display for visual feedback. When the display has a touch input, the PaperUI interface is further enhanced for more accurate user-document interaction.

3 Emerging Technologies for Realizing the PaperUI Concept

Although paper is one of the most widely used viewing devices, it cannot play dynamic media such as video and cannot be used to access the Web. It also lacks digital functions such as copy-paste, or search. Cellphones and other portable electronic devices are commonly used to play video and access Web pages, but do not have the affordances of paper such as high resolution and readability. Paper patch identification technology advances can provide the best of both devices by linking digital media to paper documents. To access digital functions associated with a paper document patch, a mobile device (e.g. camera phone) is used to identify a document patch, and digital functions associated to that patch in the document are enabled on the mobile device. With this approach, paper surface is used as a counter part of a traditional computer display and the display-equipped mobile device is used as counter parts of a traditional computer mouse and the mouse pointed sub-region on a traditional display. By building a PaperUI system like this, most digital functions supported by a traditional PC/laptop interface may be enabled for a PaperUI interface with more portability, less energy cost, and less e-wastes.

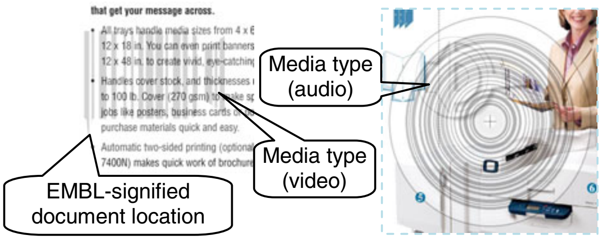
Now, there are mainly seven approaches to identify a document patch. The first approach is to print a barcode or QR code [1] on the image patch for identification. The second approach is to use micro-optical-patterns such as Dataglyph [29] on document patches for identification. The third approach is to modify document content to encode hidden information for identification [5]. The fourth approach is to index underlying paper fingerprint for document identification [4]. The fifth approach is to use OCR or character recognition outputs for identification [6,21]. The sixth approach is to index printed document content features and use these features to identify document [9,10,11,12,13,32,33]. The seventh approach is to put an RFID on the document patch for identification [14,19].

3.1 Barcode

Barcode is an optical marker printed on paper or other object. By changing a series of special patterns' color, shape, thickness and spacing, barcode can encode an ID number or other information associated with its hosting object. Barcode was first invented in 1948 by Bernard Silver [1] and had its first successful commercial use for supermarket checkout in 1974 [8].

Because of its long history, barcode gets used by most people and gets nearly unbeatable robustness and affordance. According to [2], the worst case accuracy for the old UPC code is 1 error in 394 thousand and the worst case accuracy for the new Data Matrix code is one error in 10.5 million. Meanwhile, the cost for providing a barcode is under 0.5 penny. With its unbeatable affordance and robustness, we see more and more barcode printed on document to track document category, price, or even support some basic interaction.

The big disadvantage of using barcode on document is caused by its opaque property. Regular barcodes are visually obtrusive. That makes barcode printing interfere with the document content layout. This fact generates a series of barriers for using barcode to create multiple links to the same document page. First, if we put too many barcodes on the same document page, we will have much less space for real contents. Even though the cost for producing a barcode is cheap, wasting content space may create more cost than producing a barcode. Second, changing original document layout with many obtrusive barcodes also makes the document uglier for many readers. Third, intensive barcode use on document may increase paper waste. Fourth, barcode printing may require changes to a traditional printing process. Besides these disadvantages, traditional barcode is also not very suitable to indicate media link type associated with a barcode. More specifically, if we simply print a barcode on a document page, most users will think it is catalog information or price information. Very few people will think it is a multimedia link. This barcode property may reduce readers' interests to interact with documents.



(a) (b)

Fig. 2. EMBL Examples. Excerpt from Liu et al. [38]

To compensate for these issues, we designed a media awareness-mark, called Embedded Media Barcode Link (EMBL) [38]. Figure2 shows EMBLs printed on paper documents, where the EMBL iconic marks indicate linked video and audio respectively. EMBL is a semi-transparent media-icon-modified barcode overlay on paper document content for linking to associated media. It uses an "EMBL-signified document location" to define the precise location for media association. An EMBL uses semi-transparent form to reduce interference with original document content and get closer to an EMBL signified location. It uses a semitransparent barcode to identify signified document patches, and uses iconic information to reveal associated-media information to a user. EMBL's benefits can hardly be achieved manually. To facilitate EMBL creation, we designed an EMBL authoring tool to arrange EMBL based on barcode blending coefficient optimization in a neighborhood.

3.2 Micro Optical Patterns

To reduce the intrusiveness of barcodes, researchers invented micro optical patterns such as Dataglyph [29], and Anoto dot pattern [42]. Since the encoding mechanisms of these micro optical patterns are very similar to barcodes, they are sometime considered as barcode variations. Because of similar encoding mechanisms, the identification accuracy of these micro patterns are considered comparable to traditional barcodes. Different from traditional barcodes, these micro optical patterns are much smaller in size. According to literature, each Dataglyph pattern consists of a 45-degree diagonal line as short as one over one-hundredth of an inch or less. On the other hand, the Anoto dot pattern divides paper into a grid with a spacing of about 0.3mm. These significant size reductions make these micro patterns much less intrusive than traditional barcodes. The size reduction also makes it possible to print code densely for better "virtual mouse cursor" localization. Additionally, they can encode much more data than traditional barcode on the same size document patch.

There are mainly four disadvantages of using these micro optical patterns technologies for the PaperUI design. First, these patterns need to be printed by high resolution printers. To avoid intrusiveness to printed contents, micro pattern designers have to make the basic pattern units smaller than basic content units (e.g. a stroke). This design strategy demands much better printer resolution. According to literatures, the Anoto pattern printing process needs at least 600-dpi resolution (some claim a required resolution of 1000-dpi). Since the Dataglyph line segments are very small, it also needs high resolution printing. Second, these high resolution patterns require document scanners or very high resolution cameras for image capture. This requirement makes it difficult to use existing camera phones for Dataglyph or Anoto pattern capture. To achieve this requirement, some companies make specific equipment, such as Anoto pens for PaperUI interaction. Third, printing a large amount of dots or dataglyph on paper will make the paper background look grey and reduce the content image contrast. Fourth, this technique also requires printing procedure change that may become a barrier for using the technique in existing printing industry.

3.3 Encode Hidden Information

There are many different approaches to encode hidden information in documents. These approaches are frequently considered as watermarking techniques. Documents include many objects such as figures, lines, words, paragraphs etc. People may change these objects position, size, and contour etc. to encode information in documents [5]. For example, people can shift line upwards or downwards by very small amount to encode information. They may also shift words horizontally to modify the spaces between words for information hiding [3]. There is also a large number of image water marking techniques. Even though these techniques are mainly discussed in the watermarking research field, they do have potential to be used in the PaperUI implementation. One advantage of using these techniques in PaperUI is that most of them are much less intrusive than barcode.

There are also several disadvantages of using these techniques. First, data hiding techniques frequently use content specific knowledge in algorithm design. That makes them less adaptive to a big variety of document contents. For example, it is difficult to use the line shift technology for figures or images in a document. Second, when the host signal for image watermarking is not known, crosstalk between the watermark signal and host signal is a common problem [5]. To suppress crosstalk, many algorithms require original image available for hidden information extraction. This requirement is conflict with the PaperUI procedure, which needs to identify a document patch before getting the digital version of the document patch. Third, these techniques require document providers to change the hardcopy printing process, and this change cannot be separated from content printing.

3.4 Paper Fingerprint

Paper is composed of fine fibers entangled with each other. These entangled patterns are very durable and have very low probability to be identical. Therefore, these patterns can be considered as paper fingerprints [4]. Paper patch identification can be achieved by comparing captured fiber-pattern images with indexed fiber-pattern images. This technique has nearly no disturbance to printed contents. However, since wood fibers are normally much smaller than basic units of micro-optical-patterns, this approach demands a special camera or very high resolution camera for the patch identification task. Moreover, because the fiber pattern does not follow human made rule as the micro-optical-pattern does, performing fiber pattern matching is expected to be slower than micro-optical-pattern decoding.

3.5 Character/Word Recognition

Characters and words are much easier to capture than entangled wood fibers. It is also very intuitive to think about using characters and words for text document patch identification. Commercial optical character recognition (OCR) software is widely used to convert printed books and documents into text for web publication, text-to-speech, text-mining etc. According to [6], most OCR software claims 99% accuracy rates on new good quality clean images. With 45 pages from a digitized newspaper collection 1803-1954, the author found that raw OCR accuracy varied from 71% to 98.02%. These results are collected with scanned documents as inputs. The reported accuracies will decrease further when phone camera instead of scanner is used.

There are several advantages of using existing OCR software for PaperUI implementation. First, because OCR software is considered as an existing module, application development may become much easier than using other self-developed approaches. Second, this approach does not require printing process change. Third, it is not intrusive to original document.

There are also several disadvantages of using the existing OCR software. First, OCR software is language dependent most of the time. If language independent task is demanded, all language models have to be installed on the recognition machine. Second, OCR software normally requires high resolution cameras for image capture. This requirement is only supported by very limited cameras. Third, most OCR software cannot handle angled document capture, or low lighting capture. Fourth, OCR software cannot work on photos or figures that frequently appear in documents. Fifth, most OCR software still cannot work on characters printed with a complex layout.

To overcome some OCR software limitations, researchers proposed methods for layout free and language independent character recognition [21]. We believe research in this direction is a promising direction for document patch identification. On the other hand, we are still not sure if it is possible to improve this method and make it working on figures and natural images that exist in many documents. If document patch identification algorithm cannot work on figures and natural images, we will face difficulties to achieve our final goal - using cell phone cross hair as a mouse cursor over any portion of a document.

3.6 Local Image Features

Using local image features is another promising approach for document patch identification. Recently, researchers invented many different features for cell phone and paper interaction. HotPaper [9] and Mobile Retriever [10] use features based on document text such as the spatial layout of words. Other systems such Bookmarkr [11], MapSnapper [12], EMM [32] use pixel level image features, such as the SIFT [13] and FIT [33] algorithms, to recognize generic document content such as pictures and graphic elements.

Using image local features have many advantages. First, these systems do not require exclusive spaces on paper for marker printing. Second, they do not change document printing procedure. Third, most of these features can work on camera phone captured images. Fourth, some of these algorithms allow us to accurately locate a cellphone crosshair corresponding location on paper so that we can use the cellphone crosshair as a mouse cursor for human-document interaction.

Because HotPaper and Mobile Retriever use word level layout features, they are limited to western text regions in a document. Method described in [31] can work on Japanese characters or other eastern Asia characters via parameter adjustment. However, it is still limited to text regions in a document. FIT and SIFT are tested on document mixtures including, western text regions, eastern Asia characters, figures, and natural images with reasonable recognition results [30] (99+% recognition rate). Encouraged by those test results, several recent PaperUI systems are developed based on these features. On the other hand, because FIT and SIFT work on pixel level, their constructions are normally slower than word level features. This is an issue we should pay more attention for future PaperUI research. Beyond recognition speed, another issue of this local feature based approach is the interaction indication. More specifically, because this patch identification approach does not put any marker on paper, it is difficult for a user to figure out where and how to interact with document contents. To overcome this issue, some products use dedicated text paragraphs to explain the interaction. Researchers also try to add some less intrusive markers to facilitate user's interactions with paper [32].

3.7 RFID-Based Document Recognition

Recent advances in RFID technology make RFID chips small enough to be embedded in sheets of paper [14]. These advances reveal potential of using RFID technology for paper patch identification. RFID technology may allow users to interact with paper at a distance. It also has fast response speed. However, we still need to overcome some issues before we can finally use this technology for the PaperUI development. First, special printer need to be developed for accurately "printing" these RFID devices on paper. Second, we need to develop portable RFID identification devices that can be easily carried by users. Third, we also need to develop technology that can avoid RFID interferences from other pages. Fourth, technology is needed to allow RFID devices to identify the user selected RFID from proximate RFIDs on the same page. Fifth, to facilitate interactions with the whole paper surface, it is better to develop technologies that can accurately estimate the RFID device's pointing direction based on nearby RFIDs.

4 PaperUI Applications

Based on emerging technologies, many research demos have been developed in the PaperUI direction. In this section, we will present applications based on their underlying emerging technologies and interaction resolution.

4.1 Digital Pen Based Applications

After Anoto^{TM} commercialized its digital paper technology, companies such as Logitech^{TM}, Maxell^{TM}, Nokia^{TM}, Leapfrog^{TM}, and Livescribe Connect^{TM} developed digital pen hardware for feeding users' paper inputs to computers. The technology advances on digital pen enabled many interesting applications. Early applications in this direction include medical and bank form filling.

The Leapfrog FLY Fusion Pentop Computer^{TM} [48] is sold in many supermarkets. This device can read things a user writes on Fly Paper and automatically upload the user's notes to a computer. With this UI system, a user can convert some text in notes to document and back up the document; it can also perform basic math tutorials, translations, spell check, games, and Trivia. We believe this kind of computer form is much easier for kids to master than laptops. Compared with desktops and laptops, this pen computer also has less impact to kids' traditional activity.

Recently Livescribe Connect^{TM}'s smart pen enables its users to record audio while taking lecture notes and retrieve audio with pen and notes. It also connects its user to Email, Google Docs, Facebook, and Evernote®. Additionally, it enables users to play music with pen and paper. Compared with heavy and bulky laptops, this light-weight interface is more convenient for students to take notes in classes [45].

For field workers, Capturx^{TM} provides pen-paper interfaces for Excel forms, Microsoft® Office OneNote, ArcGIS, and PDF. Moreover, its disaster response kit is designed to help emergence response teams map, collect, and share data from the scene of wildfires, hurricanes, and floods etc [49]. In a different application scenario, Infomax^{TM} provides a digital pen and paper solution for environmental compliance technicians to fill out various complicated inspection forms, such as restaurant inspection forms, methane producing well maintenance forms, traffic violation forms, and sales forms [50]. Compared with traditional laptop interfaces, this pen interface is much easier to be adopted by field workers.

In research fields, researchers also invented many novel applications with the digital pen technology. Yeh and Liao's ButterflyNet [46] enables biologists to integrate paper notes with information explicitly captured in field sites: digital photographs, sensor data, and GPS etc. With this system, biologist can directly transfer their collected contents to spreadsheet, browse all synchronously created media, and share their work with other colleagues.

Song et al. developed PenLight [52] and MouseLight [51] systems that can visually augment paper documents and give the user immediate access to additional information and computation tools. More specifically, PenLight allows an architect to use a projector and a digital pen on or above paper surface to retrieve additional information and project the information on the printout. It supports copy-paste, section view, 3D model navigation, related computation, and coordination of different versions and different collaborators.

To support meetings in conference rooms, researcher developed a system that allows meeting participants to control services in a meeting environment through a digital pen and an environment photo on digital paper [47]. Figure 3 shows the paper interface and deployment environment of this system. Unlike state-of-the-art device control interfaces that require interaction with text commands, buttons, or other artificial symbols, the photo-on-paper enabled service access is more intuitive. Compared with PC and PDA supported control, this approach is more flexible and cheap. With this system, a meeting participant can initiate a whiteboard on a selected public display by tapping the display image in the photo, or print out a display by drawing a line from the display image to a printer image in the photo. The user can also control video or other active applications on a display by drawing a link between a printed controller and the image of the display. Beyond meeting room device control, Liao et al. also developed a PaperCP system that can facilitate student-instructor communication for active learning. With the PaperCP system, users can enjoy the inherent advantages of paper. Moreover, students can electronically submit their handwritten notes to the instructor, thereby maintaining two-way communication with the instructor [53].

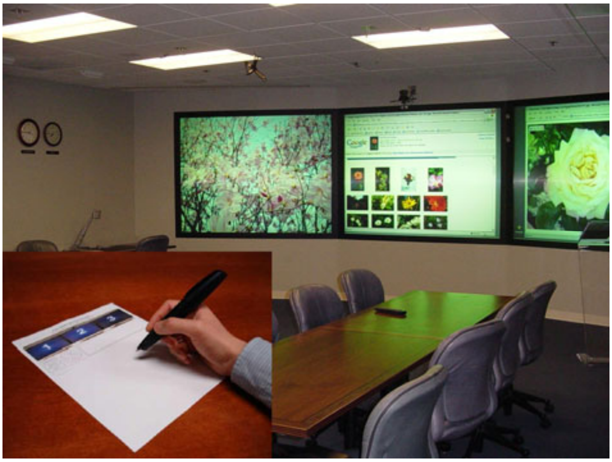


Fig. 3. The deployment environment and interface of POEMS (Paper Offered Environment Management Service) for meeting room control. Excerpt from Hu et al. [47].

4.2 Barcode Based Applications

Because of paper's good physical property, Barcode have been used on paper surface for a long time. Due to reading device limitations, the use of barcode was only limited to business operators in the past. Recently, many barcode readers were released for the fast growing cameraphone market. This trend makes it much easier for general public to use barcode. For example, consumers can use cameraphone captured barcode to compare product price [15], read reviews, acquire coupons, shopping, input a business card, navigate a city guide or map [16], get athletes' videos, pictures and fan data from a poster [17], get updates of a news story, get weather information from a map location [18], or read additional contents linked to an IEEE article. Ricoh’s iCandy [27] also allows kids to select a movie on TV based on a barcode capture. These systems are much more convenient than desktop or laptop systems that require manual input of product information or web addresses. They also overcome typing difficulties encountered by many cell phone users.

4.3 RFID Based Applications

Even though RFID has been invented for a long time, its application for PaperUI is still new. Derek et al. developed Marked-up Maps by setting a RFID grid under a paper map. With this RFID underlying grid, a user can wave a handheld computer equipped with an RFID reader above the region of interest on a paper map and get digital information. In their example, they assume a tourist can use their system to get extra information from a Marked-up Map of the Montreal subway, Nottingham, and Greater Vancouver. This extra information may be restaurant and hotel information near a subway station, shopping centers linked to store directories, theatres to current and upcoming shows, historical sights, or spatially accurate transit, district, and landmark data for navigation. One issue of that system is that a tag placed beneath a landmark that sits in relative isolation compared to other tagged landmarks will have a greater read range than landmarks that are more closely spaced [19]. This inconsistency issue makes it a little difficult for beginners to interact with a map.

4.4 Character/Word Recognition Based Applications

Camera phone based text recognition also has some interesting applications for PaperUI. For example, ScanR® and rivals can convert a camera phone image of a hardcopy into PDF for search, editing, email, text to audio translation etc. Abbyy® [7] and Google Goggles allow users to take photographs of business cards, translate the card into digital information and add that information into address book. With proper OCR software, cameraphone can also be used as a translator for foreign restaurant menus, posters etc. [20, 21]. Google Goggles’ recent release already can translate text restaurant menus from a language to other languages [23].

4.5 Encoding Hidden Information Based Applications

DigiMarcTM and its rivals try to embed digital information in images and use the embedded information to initiate many different applications. These applications include bring videos, interaction widgets, and other multimedia information to a cameraphone via capturing a DigiMarc® encoded paper page. It also claims finding product coupons, comparing product price, and finding product stores via capturing a product package [22]. In theory, DigiMarc® has the same function as barcode and therefore can be used for nearly all applications that a barcode can do. Better than a traditional barcode, DigiMarc® has much less impact to original pictures. This feature makes a DigiMarc® encoded page look nicer than a barcode attached page. On the other hand, because the encoded data has less contrast than barcode in the signal space, it will be less reliable than barcode. Moreover, since embedded data can be invisible to human. That makes it difficult for a user to find out the DigiMarc® encoded page. To solve this problem, DigiMarc® uses a small visible icon to remind users about the extra data existence.

4.6 Original Document Feature Based Applications

Recently, many companies, research labs and universities developed some interesting PaperUI applications based on original document features. For example, Google Goggles can do book search by capturing the book cover; it can recognize some artwork and bring back related information of the artwork; it can also perform product search based on wine marks and spencer [23]. Kooaba's Paperboy can provide interactive storytelling about print ads; it can navigate a consumer to a nearby store based on ads capture; it also allows food makers to provide product ingredients and origins to users via the phone capture [24]. Ricoh's technology allows people to get an updated guidebook via capture an old guidebook; it can automatically associate a http link to its surrounding text arrangement features so that users can be directed to the link by capturing the surrounding features of the http link [25]; it also supports a user to voice-annotate a real-estate brochure[26]. Amazon SnapTell can use the cameraphone photo of any CD, DVD, book, or video game to retrieve the product and find ratings and pricing information online [28]. Mobile Retriever [10] suggests using document identification technology to help visually impaired persons [10]. MapSnapper enables users to query a remote information system based on photos of a paper map taken with a mobile device [12]. Bookmarkr allows users to share photos with friends by taking an image of a photo in a photobook with the mobile phone's camera [11]. Rohs [18] augments pre-defined regions in a printed map with dynamic weather information.

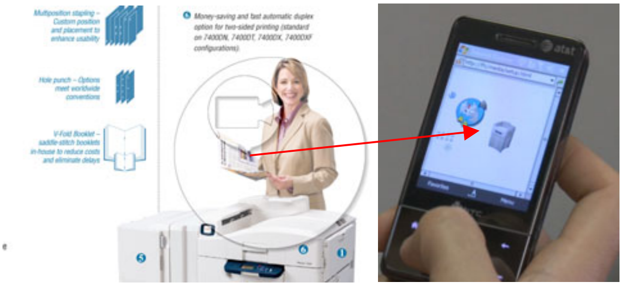

Multiposition stapling – Custom position and placement to enhance usability
Hole punch – Options meet worldwide conventions
V-Fold Booklet – saddle-stitch booklets in-house to reduce costs and eliminate delays
Money-saving and fast automatic duplex option for two-sided printing (standard on 7400DN, 7400DT, 7400DX, 7400DXF configurations)
at&t

Fig. 4. (left) An EMM in a cartridge installation manual (right) The associated step-by-step video tutorial. Excerpt from Liu et al. [32].

When content-based feature are used, there is no on-paper indication at all to the user that there is media linked to the document. As a result, a HotPaper [26] user has to pan a camera phone over the paper document to look for hotspots until feedback such as a red dot or vibration is presented on the cell phone. Moreover, there is no media type indication either. Additionally, because a user does not know where to capture and how to capture, digital links may be missed or large amount of resources has to be used to index all possible captures. This is awkward when digital links are sparsely distributed through many pages.

To solve this problem, Liu et al. augment paper with meaningful awareness-marks, called Embedded Media Markers (EMMs) that indicate the existence, type, and capture guidance of media links. Figure 4 shows an EMM and its application scenario. On seeing an EMM, the user knows where to capture an image of the EMM-signified document patch with a cell phone in order to view associated digital media. This is analogous to Web pages that use underlines, font differences, or image tags to indicate the existence of links that users then click for additional information. Unlike barcodes, EMMs are nearly transparent and thus do not interfere with the document appearance. Unlike Embedded Data Glyphs [29] or Anoto patterns [42], EMMs can be printed with a regular low-resolution printer and identified from an image captured by a normal cell phone camera. Unlike other appearance-based approaches, EMMs clearly indicate signified document patches and locations. The design of EMMs also indicates what type of media (e.g. audio, video, or image) is associated with the EMM-signified document location. Furthermore, by requiring the captured image to cover the whole mark, we can improve feature construction accuracy, matching accuracy, and efficient resource usage [32]. Currently, the EMM system supports links to 5 types of marks including audio, video, web, image, and text.

4.7 Fine-Grained Phone-Paper Interactions

Ideally, we want to use cellphone or pen as a mouse or even better device in a PaperUI system. Applications in the previous session focus on creating digital links to a large paper patch. Operations in these applications are similar to very rough point-and-click mouse-operations. There is still a big gap between these coarse operations and fine-grained mouse operations. For example, these applications do not use gestures such as the marquee, bracket and lasso selectors, which offer more flexibility to manipulate document content and have been widely deployed in GUIs [34]. They also lack resolution for selecting a small region, such as a math symbol region, in a set of adjacent regions.

To overcome these problems, Liao et al. developed PACER [34] that features a camera-touch hybrid interface. Figure 5 illustrates a PACER application scenario. The system recognizes documents based on natural document visual features instead of any special markers on paper or specific end user hardware. More importantly, it allows users to manipulate fine-grained document content with various gestures beyond point-and-click. With this system, a user first aims a camera phone roughly at the region and captures a picture. PACER recognizes the picture, and presents on the screen the corresponding high quality digital version of the document, rather than the raw video frames (We call this design loose registration). The user then operates the phone as an embodied see-through “Magic Lens” [35], with its crosshair center treated like a mouse pointer. To fine tune the starting/ending point of the gesture, the user can also switch from the embodied interaction to the touch interaction by directly touching the screen and moving the pointer in a zoomed and scrollable view.

With this camera-touch hybrid interaction, a PACER user can select, copy and email an interesting region from a paper document; pick the title of a reference from a journal, and then search for its full text on Google Scholar; specify a word or math symbol for text search on paper; check dictionary of a foreign words in the document; snap to a sightseeing drive on a paper map, and browse the street views on the phone while sweeping it along the route; play a music score by moving the phone over the intended sections; using voice or multimedia data to annotate a very small region in the document; perform free-form gestures for document annotation; or discuss a document with a remote user via pointing, drawing, copy-paste, and speaking. Through enabling fine-grained gesture to the paper-cameraphone system, PACER greatly extended the possible application scenarios of the PaperUI concept.

Different from the PACER usage scenario, a user may also want a portable system support for reading with papers on a desk. This kind of scenario is less mobile than the PACER scenario. However, reading with the paper on a desk and a light pen in hand is a more relaxed setup for long time reading. FACT [36] is designed for this need. Figure 6 illustrate the FACT system and one application scenario. FACT consists of a small camera-projector unit, a laptop, and ordinary paper documents. With the camera-projector unit pointing to a paper document, the system allows a user to issue pen gestures on the paper document for selecting fine-grained content and applying various digital functions a traditional computer can issue. FACT can also support various applications in information transfer, association, sharing and synchronous navigation across the paper-computer boundary. For example, a FACT user can select a picture on paper and then copy the digital version of this picture into a Word document on the computer; this operation can also be reversed so that Multimedia annotations created on the laptop can be attached to a specific word, phrase, paragraph or arbitrary shaped region for pen-based retrieval. FACT may also be used as a platform for sharing paper/web information with remote friends. For example, a paper user can select a map region and ask a remote friend for tour suggestions, and the remote friend can attach interesting web contents to the document and project back on the real paper map.

The PaperUI concept is not proposed to completely replace existing computer interfaces. Actually, the PaperUI interface can be used with existing computer interfaces to take both advantages. The recently developed MixPad [37] allows a user drag a picture on a paper page to a nearby laptop with a finger on paper or type text via the laptop keyboard to annotate an illustration in a printout. Figure 7 illustrates the MixPad idea.

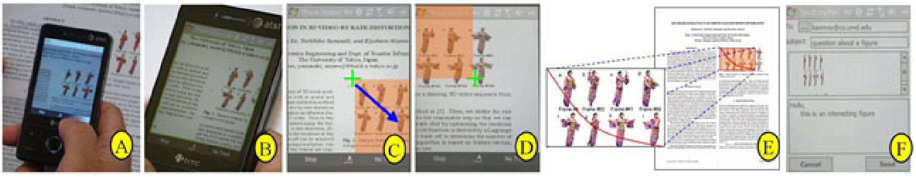


Fig. 5. Copy&Email via PACER. (A) Point the phone crosshair to an interesting area on paper and take a snapshot. (B) Once the snapshot is recognized, the corresponding high-quality version is displayed. (C)-(D) Move the phone (in the arrow direction) over the paper to select a region (highlighted in orange) with a marquee gesture. (E) Overview of the gesture/region within the document. (F) The selected region is sent via email, together with the hyperlink pointing to the original page and document. Excerpt from Liao et al.[34].

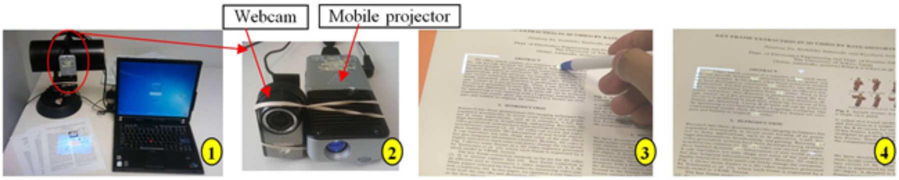


Fig. 6. (1) Interface prototype, (2) Close-up of the camera-projector unit, (3) A word (highlighted by the projector) selected by a pen tip for full-text search, (4) The resulting occurrences of the word highlighted by the projector. Excerpt from Liao et al.[36]

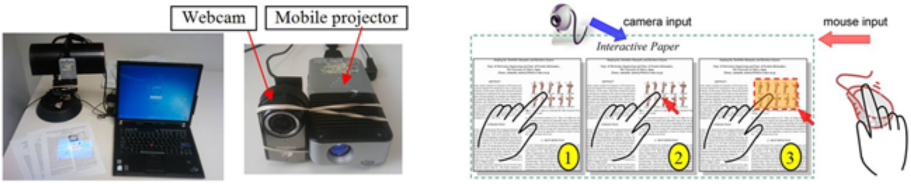


Fig. 7. (left) MixPad interface prototype, (middle) Close-up of the camera-projector unit, and (right) steps to select a picture portion: (1) roughly pointing a finger to a region, (2) mouse cursor being projected to where the fingertip is, and (3) Drawing a mouse marquee gesture to select a region at fine granularity. Excerpt from Liao et al.[37].

5 Concluding Remarks

Emerging technologies open a door for us to explore the PaperUI concept and practices. With many research prototypes and products developed in this field, we believe the prime time for this new interface is coming. Through migrating existing PC applications to the more portable PaperUI interface and developing new applications in this field, we strongly believe that we can find more promising research topics on novel document patch identification methods, mobile device pointing direction estimation, and user interactions etc.

References

1. Wikipedia, Barcode, `http://en.wikipedia.org/wiki/Barcode`
2. IDAUTOMATION, About Barcode Accuracy and Misreads, `http://www.barcodefaq.com/barcode-encoding.html`
3. Maxemchuk, N.F., Low, S.: Marking text documents. In: Proc. IEEE Int. Conf. Image Processing 1997 (ICIP 1997), Santa Barbara, CA, vol. 3, pp. 13–16 (October 1997)
4. Fuji Xerox, Paper Fingerprint Recognition Technology (August 3, 2011), `http://www.fujixerox.com/eng/company/technology/xaya/`
5. Hartung, F., Kutter, M.: Multimedia watermarking techniques. Proceedings of the IEEE 87, 1079–1107 (1999)
6. Holley, R.: How Good Can It Get? D-Lib Magazine (March/April 2009), `http://www.dlib.org/dlib/march09/holley/03holley.html`
7. Jacobi, J.L.: ABBYY FineReader 10 Professional Edition review (August 3, 2011), `http://www.pcadvisor.co.uk/reviews/software/3258293/abbyy-finereader-10-professional-edition-review/`
8. Fox, M.: Alan Haberman, Who Ushered In the Bar Code, Dies at 81. The New York Times (June 15, 2011)
9. Resource Conservation Alliance, Focus On Paper Consumption, `http://www.woodconsumption.org/products/paper.pdf`
10. Liu, X., Doermann, D.: Mobile Retriever: access to digital documents from their physical source. Int. J. Doc. Anal. Recognit. 11(1), 19–27 (2008)
11. Henze, N., Boll, S.: Snap and share your photobooks. In: Proceedings of ACM Multimedia 2008, pp. 409–418 (2008)
12. Hare, J., Lewis, P., Gordon, L., Hart, G.: MapSnapper: Engineering an Efficient Algorithm for Matching Images of Maps from Mobile Phones. In: Proceedings of Multimedia Content Access: Algorithms and Systems II (2008)
13. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. Int. J. Comput. Vision 60(2), 91–110 (2004)
14. FujiSankel, Hitachi Develops RFID Powder, `http://healthfreedoms.org/2009/10/23/hitachi-develops-rfid-powder/`
15. Asaravala, A.: Camera Phones Help Buyers Beware, Tech Biz: IT, `http://www.wired.com/techbiz/it/news/2004/01/61936`
16. Hanlon, M.: ScanBuy - barcode software on your camera phone creates the Physical World Hyperlink, Mobile Technology, `http://www.gizmag.com/go/6701/`
17. Byron Acohido, New '2D barcodes' puts info at the tip of your camera phone, USA TODAY (May 20, 2009)
18. Rohs, M.: Real-World Interaction with Camera Phones. In: Murakami, H., Nakashima, H., Tokuda, H., Yasumura, M. (eds.) UCS 2004. LNCS, vol. 3598, pp. 74–89. Springer, Heidelberg (2005)
19. Reilly, D., et al.: Marked-Up Maps: Combining Paper Maps and Electronic Information Resources. Personal Ubiquitous Computing 10(4), 215–226 (2006)
20. Ellison, J.: How To Make The Cell Phone Into A Portable Scanner. News Week Magazine (October 20, 2007)
21. Iwamura, M., Tsuji, T., Kise, K.: Memory-based recognition of camera-captured characters. In: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (DAS 2010), pp. 89–96. ACM, New York (2010)

22. DigiMarc, Enhanced, Interactive Experiences from Media Using Smartphones (August 3, 2011), http://www.digimarc.com/
23. Google Goggles, Google Goggles in action (August 3, 2011), http://www.google.com/mobile/goggles/#label,
24. Kooaba, Paperboy: Deliver digital extras for print (August 3, 2011), http://www.kooaba.com/
25. Hull, J.J., Erol, B., Graham, J., Ke, Q., Kishi, H., Moraleda, J., Van Olst, D.G.: Paper-based augmented reality. In: Proc. 17th Int. Conf. Artificial Reality and Telexistence
26. Erol, B., Antunez, E., Hull, J.J.: HOTPAPER: multimedia interaction with paper using mobile phones. In: Proceedings of ACM Multimedia 2008, pp. 399–408 (2008)
27. Graham, J., Hull, J.: Icandy: A tangible user interface for itunes. In: Proc. CHI 2008: Extended Abstracts on Human Factors in Computing Systems, Florence, Italy (2008)
28. Toto, S.: SnapTell: Instant Product Lookup From The iPhone. You Want This. Tech Crunch (November 19, 2008), http://techcrunch.com/2008/11/19/snaptell-instant-product-lookup-from-the-iphone-you-want-this/
29. Hecht, D.L.: Embedded Data Glyph Technology for Hardcopy Digital Documents. In: SPIE -Color Hard Copy and Graphics Arts III, vol. 2171, pp. 341–352 (February 1994)
30. Nakai, T., Kise, K., Iwamura, M.: Camera-Based Document Image Retrieval as Voting for Partial Signatures of Projective Invariants. In: Proc. 8th International Conference on Document Analysis and Recognition (ICDAR 2005), pp. 379–383 (September 2005)
31. Nakai, T., Kise, K., Iwamura, M.: Real-Time Retrieval for Images of Documents in Various Languages using a Web Camera. In: Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR 2009), pp. 146–150 (July 2009)
32. Liu, Q., Liao, C., Wilcox, L., Dunnigan, A., Liew, B.: Embedded Media Markers: Marks on Paper that Signify Associated Media. In: Proceedings of ACM IUI 2010, pp. 149–158 (2010)
33. Liu, Q., Yano, H., Kimber, D., Liao, C., Wilcox, L.: High Accuracy and Language Independent Document Retrieval With A Fast Invariant Transform. In: Proceedings of IEEE ICME 2009, pp. 386–389 (2009)
34. Liao, C., Liu, Q., Liew, B., Wilcox, L.: Pacer: fine-grained interactive paper via camera-touch hybrid gestures on a cell phone. In: Proceedings of the 28th International Conference on Human Factors in Computing Systems (CHI 2010). ACM, New York (2010)
35. Rohs, M., Oulasvirta, A.: Target Acquisition with Camera Phones when userd as Magic Lenses. In: Proceedings of CHI 2008, pp. 1409–1418 (2008)
36. Liao, C., Tang, H., Liu, Q., Chiu, P., Chen, F.: 2010. FACT: fine-grained cross-media interaction with documents via a portable hybrid paper-laptop interface. In: Proceedings of the International Conference on Multimedia (MM 2010), pp. 361–370. ACM, New York (2010)
37. Liao, C., Liu, Q.: MixPad: Augmenting Interactive Paper with Mice & Keyboards for Cross-media Interaction with Documents. To be published Proc. ACM UBICOMP 2011, Beijing, China (2011)
38. Liu, Q., Liao, C., Wilcox, L., Dunnigan, A.: Embedded media barcode links: optimally blended barcode overlay on paper for linking to associated media. In: International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction (ICMI-MLMI 2010). ACM, New York (2010)
39. The Office of the Future, Business Week (2387): 48–70 (June 30, 1975)

40. Lahey, B., Girouard, A., Burleson, W., Vertegaal, R.: Paperphone: understanding the use of bend gestures in mobile devices with flexible electronic paper displays. In: Proceedings of CHI 2011. ACM Press (2011)
41. Wellner, P.: Interacting with paper on the DigitalDesk. Communications of the ACM 36(7), 87–96 (1993)
42. Anoto, THE DIGITAL PEN, (August 3, 2011), `http://www.anoto.com/?id=19146`
43. Mackay, W.E., Fayard, A.-L.: Designing Interactive Paper: Lessons from three Augmented Reality Projects. In: Proc. IWAR 1998, pp. 81–90 (1998)
44. easeMD, Digital Pen for Off-site Use by Physicians (August 3, 2011), `http://www.easemd.com/doctors/digital-pen.php`
45. Livescribe, Record and Play Back (August 3, 2011), `http://www.livescribe.com/en-us/smartpen/`
46. Yeh, R.B., Liao, C., Klemmer, S., Guimbretière, F., Lee, B., Kakaradov, B., Stamberger, J., Paepcke, A.: ButterflyNet: A Mobile Capture and Access System for Field Biology Research. In: Proceedings of CHI 2006, pp. 571–580 (2006)
47. Hu, C., Liu, Q., Liu, X., Liao, C., McEvoy, P.: Poems: A Paper Based Meeting Service Management Tool. In: ICME 2007, February 7 (2007)
48. Leapfrog, Pentop Computer User Manual, `http://www.leapfrog.com/etc/medialib/leapfrog/fly/ffls_37735.Par.40053.File.dat//Users/na/Desktop/Parent%20Guides/FFLS_37735_PG.pdf`
49. Capturx, Automate Any Paperwork, `http://www.adapx.com/`
50. Infomax, Digital Writing Slutions Made Simple, `http://www.e-infomax.com/case.html`
51. Song, H., Guimbretiere, F., Grossman, T., Fitzmaurice, G.: MouseLight: bimanual interactions on digital paper using a pen and a spatially-aware mobile projector. In: Proceedings of the 28th International Conference on Human Factors in Computing Systems (CHI 2010). ACM, New York (2010)
52. Song, H., Grossman, T., Fitzmaurice, G., Guimbretiere, F., Khan, A., Attar, R., Kurtenbach, G.: PenLight: combining a mobile projector and a digital pen for dynamic visual overlay. In: Proceedings of the 27th International Conference on Human Factors in Computing Systems (CHI 2009), pp. 143–152. ACM, New York (2009)
53. Liao, C., Guimbretière, F., Anderson, R., Linnell, N., Prince, C., Razmov, V.: PaperCP: Exploring the Integration of Physical and Digital Affordances for Active Learning. In: Baranauskas, C., Abascal, J., Barbosa, S.D.J. (eds.) INTERACT 2007, Part II. LNCS, vol. 4663, pp. 15–28. Springer, Heidelberg (2007)

Decapod: A Flexible, Low Cost Digitization Solution for Small and Medium Archives

Faisal Shafait[1], Michael Patrick Cutter[2], Joost van Beusekom[1],
Syed Saqib Bukhari[2], and Thomas M. Breuel[2]

[1] German Research Center for Artificial Intelligence (DFKI),
Kaiserslautern, Germany
{faisal.shafait,joost.van_beusekom}@dfki.de
[2] University of Kaiserslautern, Germany
cutter@iupr.com, {bukhari,tmb}@informatik.uni-kl.de

Abstract. Scholarly content needs to be online, and for much mass produced content, that migration has already happened. Unfortunately, the online presence of scholarly content is much more sporadic for long tail material such as small journals, original source materials in the humanities and social sciences, non-journal periodicals, and more. A large barrier to this content being available is the cost and complexity of setting up a digitization project for small and scattered collections coupled with a lack of revenue opportunities to recoup those costs. Collections with limited audiences and hence limited revenue opportunities are nonetheless often of considerable scholarly importance within their domains. The expense and difficulty of digitization presents a significant obstacle to making such paper archives available online. To address this problem, the Decapod project aims at providing a solution that is primarily suitable for small to medium paper archives with material that is rare or unique and is of sufficient interest that it warrants being made more widely available. This paper gives an overview of the project and presents its current status.

1 Introduction

Document digitization is not easy. The whole process, from initial image capture to a useful output, is arcane with no guarantee of usable results. Though there has been an immense amount of high quality research in the document engineering field over the past two decades in both academia and industry, little of it has made it into real, deployed systems. Even after capture, in most cases the technology needed to convert the material is expensive and requires expert users to configure it, and to develop workflows to deal with the exceptions that inevitably occur. Existing digitization solutions are well suited for large digitization projects like [16], where expensive equipment and training personnel is economically feasible. However, for small projects these costs present a significant barrier.

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 101–111, 2012.

To assemble a solution an institution must procure and assemble equipment, train operators, procure several pieces of software, and develop exception handling and QA processes and tools. All of these require specialized skills and knowledge that is not readily available. It is beyond the scope of the average institution, and it is expensive. The Decapod project targets just these institutions or collections, ones with modest budgets, with material that is unique or fragile and must remain on-site, either because it is being used locally or there are restrictions on it being removed. Such institutions do not have sufficient material to justify the high set up costs of the overseas solution despite the low unit costs. A capture process is needed that is fast, able to deal gently with diverse materials and resilient to operator error, paper quality, lighting variations and other factors.

Much of the scholarly material that would benefit from Decapod is complex in layout. Journals, with their multi-column layout, illustrations and complex lists and tables, auction catalogs, inventories and records, newspapers and newssheets, manuscripts and so forth contain images, multiple columns or boxes. Moreover, many of these documents are old, fragile, discolored, and in archaic typefaces. If the material is bound then even flat-bed scanning produces distorted images. Off-the-shelf packages such as the OCR packages are not particularly good in dealing with complex layouts and historical documents. The correction process is particularly tedious. This is unlikely to change as the market for OCR is not large, and the investment of the surviving commercial companies such as Abbyy, Nuance (Scansoft) is more oriented towards the commercially more important goal of extending the languages covered than addressing the more esoteric layouts. (It should be noted that they are doing an excellent job of addressing the breadth of languages, where inexpensive software packages can OCR around 200 languages).

Decapod is focused on delivering an affordable and cost effective solution to permit high quality, minimal user intervention solutions to the capture and preparation of small to medium collections. We apply the technology advances (both hardware and software) of the recent decades to remove the usability, cost and quality barriers to such projects. This is now possible thanks to the existence of well understood software and algorithmic approaches to the digitization problem and the emergence of affordable high resolution digital cameras. The project will deliver an out-of-the-box solution that allows local staff with modest training to easily capture their material and convert it to archive quality content suitable for deposition in online archives. The solution will deal with bound material that must be treated gently (and also, of course, single sheet material), and will trim the image down to the page boundaries and remove discolorations and other visual defects so as to deliver page images comparable to those from a flat bed scanner. Our proposed solution will accomplish the following:

- Non-Destructive Scanning: The system will allow the non-destructive scanning of documents, journals, and bound volumes.
- Low Cost: Open source software, standard laptops, consumer-grade digital cameras.

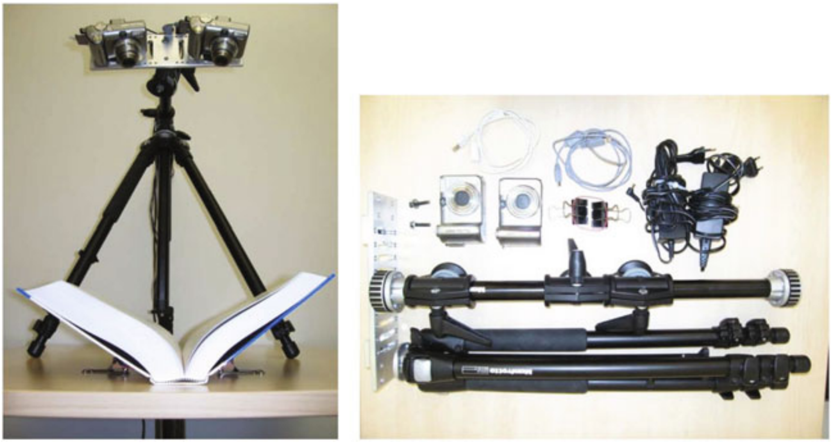


Fig. 1. A prototype scanning rig, consisting of standard tripod hardware and consumer digital cameras. The rig is portable and can be operated anywhere using a laptop computer.

- Competitive Quality: When used with a high-end digital camera and good lighting, the system will be capable of generating images of quality at least as good as that obtained by Google's scanning process.
- Portability: All system hardware components (cameras, tripods, laptop, etc.) will fit into a small suitcase.
- Usability by Non-Experts: The system will require minimal operator training and be usable by non-experts such as local staff and volunteers.
- Real-Time Scan Quality Control: Re-scans can be expensive or impossible; real-time scan quality control catches a high fraction of capture errors while the operator still has access to the document.

2 System Architecture

Decapod will deliver a complete solution for the capture of materials for which current digitization workflows are not appropriate. The deliverables will include software and suggested hardware configurations and hence allow the assembly of a complete system using off-the-shelf hardware components. A prototype of the system hardware is shown in Figure 1. The software components of the proposed system are:

- camera-based document capture using advanced computer vision algorithms to create "Scanner Equivalent" page images.
- A deeply user centered and easy-to-use document capture and quality control system based on state of the art document understanding technology that removes the need for most user interaction and simplifies the interaction when it is necessary.

- A high-quality scan-to-PDF conversion software that emits PDF/A with high fidelity (to the original) typefaces and embedded document layout information to permit reflow and text to speech.
- Integration of all software components into an end-to-end solution.

The overall flow of the system is a series of three steps. First there is the capture process, i.e. the creation of images of the pages from the physical material. The software demands at this point are primarily to ensure that the material is captured in its entirety and to sufficient quality. The next stage, which could take place later, is the generation of archive quality images and document structure information. The final stage is the generation of a usable output, which in this project is reflowable PDF/A documents. Several components from the OCRopus open source OCR system [3] will be employed in the system to achieve the final output. The relationship between the OCRopus system and the additional modules being developed as part of the Decapod project is shown in Figure 2. Different modules of the Decapod system are now described in more detail. Note that all of these modules have been / are being developed as part of the Decapod project.

2.1 Document Capture

Documents are captured using a standard off-the-shelf consumer camera. The camera is connected to a USB port of the PC and `gphoto` library[1] is used to view live (low-resolution) video stream from the camera and to trigger image capture programmatically. We tested cameras from different brands for their support and stability with `gphoto` library. In our experience, Cannon cameras proved to be most stable (in terms of software crashes). Hence, we picked Cannon Powershot G10 due to its high resolution and relatively low cost. To increase throughput, we have also integrated support for a USB foot pedal to trigger image capture so that the user only has to turn pages with his hands while digitizing a book.

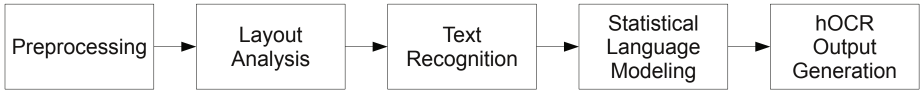
2.2 Dewarping

Once book pages are captured, they need to be dewarped for a better visual impression [12]. Dewarping module of the project is at a starting stage now. We plan to investigate approaches for monocular dewarping [5], stereo dewarping [15,17] as well as dewarping using structured light [4] for their ease of setup, robustness, stability and output image quality.

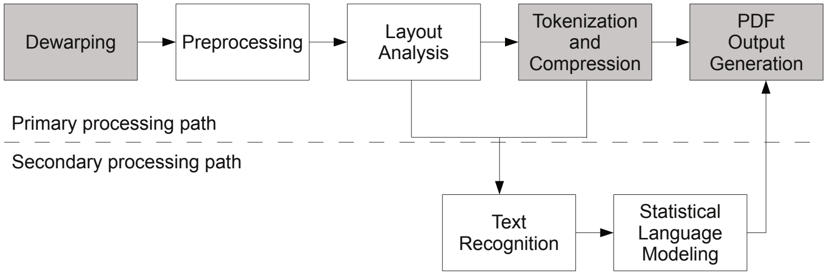
2.3 Preprocessing / Layout Analysis

The flattened book pages returned by the dewaping module can be processed by typical modules for scanned pages like border noise removal [13,11], skew correction [1], text/non-text segmentation [9], and layout analysis [14]. The text/non-text segmentation module is particularly more important in Decapod since we

[1] http://www.gphoto.org/



(a) The standard OCRopus processing pipeline



(b) The Decapod book scanning pipeline (components being developed as part of the Decapod project are shown in gray)

Fig. 2. The relationship between the OCRopus system and the additional modules being developed as part of the Decapod project

need to determine which connected components belong to text for font reconstruction. We have developed a multi-resolution morphology based method [6] within the Decapod project. Its main advantage over our previously published method [9] is that it does not require block segmentation prior to text/non-text classification. It is an extension of Bloomberg's text/image segmentation algorithm [2] that is specifically designed for text and halftone image separation. Bloomberg's method is simple and fast and performs well on text and halftone image segmentation, but it is unable to segment text and non-text components other than halftones, such as drawings, graphs, maps, etc. In our work, we introduced modifications to the original Bloomberg's algorithm for making it a general text and non-text image segmentation approach, where non-text components can be halftones, drawings, maps, or graphs.

2.4 Tokenization

Input to the tokenization algorithm is a text-only image in which non-text components have already been removed. The goal of tokenization is to cluster all the characters in a document into clusters containing the same character in the same font. These clusters are then called tokens. The tokenization is described in [7] in detail. A short summary will be presented here.

As we are interested in obtaining clusters with the same characters only, the label of each character obtained by OCR is used to cluster only characters together having the same label.

Inside this cluster it has then to be distinguished between different fonts. This is done by clustering characters that are visually similar into same clusters. As most font differences will show in the outline of the character - the main shape being the same for the most common fonts - the dissimilarity measure used for clustering will focus on the comparison of the outline of two characters.

First, the two characters are aligned to overlap their centroids. Second, the outline maps of both characters in a cluster are computed using morphological operations. Last, the dissimilarity is computed using the following formulas:

$$\text{error} = \sum_{x=0}^{W}\sum_{y=0}^{H} M(x,y) * \|T(x,y) - I(x,y)\| \tag{1}$$

$$\text{error}_I = \sum_{x=0}^{W}\sum_{y=0}^{H} M(x,y) * I(x,y) \tag{2}$$

$$\text{error}_T = \sum_{x=0}^{W}\sum_{y=0}^{H} M(x,y) * T(x,y) \tag{3}$$

$$\text{final error} = min(\frac{\text{error}}{\text{error}_I}, \frac{\text{error}}{\text{error}_T}) \tag{4}$$

where I is the candidate image, T is the token, and M is the mask. H and W represent the height and width of the token respectively. The mask M is obtained by morphological operations on the image of the character. First the image is dilated and inverted. Then this image is subtracted from a binary eroded image. The result of this last operation is combined with a thinned version of the image by a binary OR operation, finally arriving at the mask M.

If the overall error is lower than a given threshold, the new image I is clustered into the same cluster as T. In the other case, a new cluster is generated. Note that, due to the edge sensitive shape similarity metric, it is unlikely that different letters will be merged together.

The pairwise computation of the dissimilarity measure is computationally expensive. To reduce the amount of comparisons, a preliminary, inexpensive clustering is done based on the following features: height, width, and the number of holes present in the image of the character. Only characters where all of these features are identical, are compared to each other in the clustering.

2.5 Font Reconstruction

Font reconstruction is the inference of a mathematical representation of a digital font, given how it is typeset in a document image, which can then be used to reproduce the font in copies of the document or in new documents. The goal of the font reconstruction model is to capture all necessary characteristics of the original font in order to reproduce the original document in a visually faithful way.

The OCR system, OCRopus, outputs segmented and labeled letters, which become the input to our font reconstruction algorithm. After the token clustering phase the document can now be represented as a sequence of token IDs delimited by spaces. The co-occurrence of these tokens in words is the feature used to infer the candidate font groups. This is based on the assumption that a single latin word is almost always written in the same font. Therefore, for example if a token representing the letter 'a' co-occurs in multiple words then we assume those words were written in the same font. The candidate font selection method is further discussed in [7].

The next phase is to classify the font class of every letter within the document. This is achieved by exploiting locality and shape similarity to the candidate font alphabets. The probability of a token being classified as a particular font is determined by its spatial proximity of tokens which make up a respective font. The influence of font assignment of tokens decays until a maximum distance where then a simple nearest neighbor classifier is used for all remaining token font assignment. Details of the method can be found in [8]. Through our evaluation, we showed that this method is reasonably accurate on multi-font documents scanned at 300 DPI.

Once is a font group is identified, further processing is required in order to capture and output the font in reconstructed documents. We begin with the font group's alphabet of token prototypes and trace each of these prototypes by using `potrace`[2] - an open source polygon approximation tracing algorithm [10]. Since we want an accurate representation of the font, it is critical that the input to our tracing algorithm be at the maximum possible resolution. Since high resolution is already a requirement for high quality OCR this condition is met.

The input to `potrace` is the merged token prototype image (bitmap image) and the output in the vector outline (Bézier curves). `Potrace` approximates the bitmap using polygons to trace the outline of the bitmap. Peter Silinger describes the operation of `potrace` in four steps. In the first step the bitmap is decomposed into a sequence of paths between black and white areas in the image. In the second step the paths from the previous step are approximated by an optimal polygon. In the third step the polygons are transformed into an outline. The final step joins the outlines of each polygon to form Bézier curves representing the image.

Once a vectorized representation is available, the next step is to add parameters to each character in the font that control how it is rendered. This includes:

- **Relative size:** The size of each letter is a function of the aggregated statistics of the bounding box sizes of each instance of the letter classified as belonging to the same font group.
- **Baseline ratio:** Currently the system approximates how much of the letter should be placed above and below the baseline by analyzing the baseline ratio of the same letter in another established font. This aspect of the system can be improved by finding the most similar corresponding font.

[2] http://potrace.sourceforge.net/

- **Left-right padding:** The amount of space between a letter and any other generic symbol can be inferred from either a corresponding font or extracted directly from the original document.
- **Kerning:** Kerning is the specific amount of space specified in some fonts between particular pairs of letters (like 'V' and 'A') to give a more visually pleasing effect. The Decapod system currently does not add specific kerning pairs. This could be added by measuring the spacing between pairs of letters within the document.

Often a document does not contain every letter of the alphabet in every font. Therefore, a distinction is to be made between reconstructing a font that can be used to recreate a specific document and reconstructing a portable font that can be used to author new documents in a similar form to the original. Ultimately, Decapod should achieve the former. To achieve the latter goal, one needs the ability to detect the most similar fonts to the newly reconstructed font in order to complete the alphabet of the font.

If there are more letters of the same font in a document, the corresponding tokens of a large number of these letters will be merged together during tokenization. Hence, result of tracing will be more visually appealing. Besides, the compression ratio will also increase with document size.

The font reconstruction algorithm is robust against possible outliers because letters that occur frequently are the ones that are the most likely to become part of the reconstructed font. After the initial candidate font selection phase all other tokens are classified as instances of one of the present fonts. These instances of a candidate font do not effect the shape of the font and are purely used as labels when reconstructing the document with reconstructed fonts.

2.6 PDF Generation

The PDF generation step converts the dewarped document images into different types of PDF. A representation of the processing pipeline for PDF generation is shown in Figure 3. Depending on the type of PDF that is wanted as output, different steps of the pipeline are run.

- Image only PDF: in this case, the dewarped images are converted into a single PDF. As no textual information is available this form is not searchable. However, this format needs no additional information and can thus even be used if no OCR and no font information is available. As input only the dewarped images are needed.
- Image with overlaid transparent text PDF: in this format, the recognized text will be overlaid transparently on the dewarped image, making the PDF searchable while maintaining the documents original appearance. As input the dewarped images and the character bounding boxes together with their label is needed.
- Tokenized PDF: instead of saving the page as a whole, tokenization is done on the connected components and only one character image per cluster is

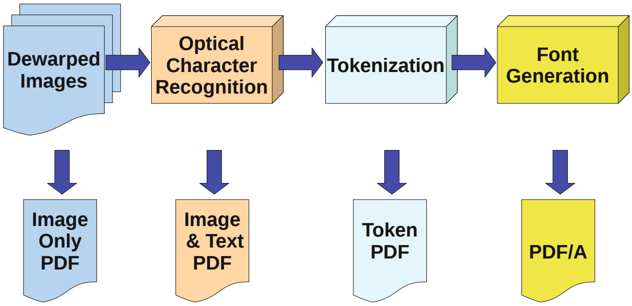


Fig. 3. Overview of the PDF generation steps. Dewarped images can easily be converted into image only PDFs, whereas font reconstructed PDFs need all information of all intermediate processing steps.

saved. This results in a searchable, lossy compressed version of the original input image. For this process OCR output and tokenization is needed. This type of PDF is merely an intermediate format and is not meant to be in the final system. Again, the dewarped images together with the OCR information serve as input.

– Font Reconstructed PDF: in this format, the extracted tokens are not saved as image inside the PDF, but they build the basis for generating a font out of the tokens. The generated fonts are then embedded in the PDF document.

The PDF generation uses the `ReportLab Toolkit` for PDF generation[3] and OCRopus [3] for performing OCR to create the underlying text layer. All data is stored in the *book structure*, a set of files and directories for each set of documents, as proposed by the OCRopus system.

3 Project Status

The Decapod software is available online[4]. The current version implements the full processing pipeline: starting with capturing of documents, loading and saving started projects, reordering the pages and export as PDF.

However, some important modules are still missing: dewarping is not yet available, instead only dewarped (e.g. flat bed scanned) images can be processed. The PDF generation has limited functionality. Currently, image-only PDF and image with overlaid transparent text layer PDF can be generated.

It is difficult to assess the overall performance of the Decapod system at this stage. In our view, the dewarping module will be the most crucial for delivering an overall good quality output. For tokenized and font reconstructed PDFs, it would be important to keep the token clustering errors low. We also plan to

[3] http://www.reportlab.com/

[4] http://code.google.com/p/decapod/

develop an automatic validation algorithm to verify the quality of the font reconstructed PDF. In this way, it will be possible to warn the user of a bad output quality or to automatically fall back to the image with overlaid transparent text PDF.

4 Conclusion

The Decapod project investigates the use of low cost consumer electronics hardware (e.g. a standard tripod stand, consumer digitial cameras, and a PC) for setting up a digitization project at a small scale. Besides, software components are being developed to produce searchable PDFs that are visually similar to the original documents by having the same (reconstructed) fonts and layout. Our solution is open-source, easy to use, and will provide an out-of-the box method for in-situ digitization of small to medium archives where setting up a full production system such as that used by JSTOR or Google is not feasible.

References

1. van Beusekom, J., Shafait, F., Breuel, T.M.: Combined orientation and skew detection using geometric text-line modeling. International Journal on Document Analysis and Recognition 13(2), 79–92 (2010)
2. Bloomberg, D.S.: Multiresolution morphological approach to document image analysis. In: Proc. Int. Conf. on Document Analysis and Recognition, St. Malo, France, pp. 963–971 (1991)
3. Breuel, T.M.: The OCRopus open source OCR system. In: Proc. SPIE Document Recognition and Retrieval XV, San Jose, CA, USA, pp. 0F1–0F15 (January 2008)
4. Brown, M., Seales, W.: Document restoration using 3d shape: A general deskewing algorithm for arbitrarily warped documents. In: Proc. Int. Conf. on Computer Vision, pp. 367–374 (July 2001)
5. Bukhari, S.S., Shafait, F., Breuel, T.M.: Dewarping of document images using coupled-snakes. In: Proc. Int. Workshop on Camera-Based Document Analysis and Recognition, Barcelona, Spain, pp. 34–41 (July 2009)
6. Bukhari, S.S., Shafait, F., Breuel, T.M.: Improved document image segmentation algorithm using multiresolution morphology. In: SPIE Document Recognition and Retrieval XVIII, San Francisco, USA (January 2011)
7. Cutter, M.P., van Beusekom, J., Shafait, F., Breuel, T.M.: Unsupervised font reconstruction based on token co-occurrence. In: 10th ACM Symposium on Document Engineering, Manchester, UK (September 2010)
8. Cutter, M.P., van Beusekom, J., Shafait, F., Breuel, T.M.: Font group identification using reconstructed fonts. In: SPIE Document Recognition and Retrieval XVIII, San Francisco, USA (January 2011)
9. Keysers, D., Shafait, F., Breuel, T.M.: Document image zone classification - a simple high-performance approach. In: 2nd International Conference on Computer Vision Theory and Applications, Barcelona, Spain, pp. 44–51 (March 2007)
10. Selinger, P.: Potrace: a polygon-based tracing algorithm (2003), http://potrace.sourceforge.net

11. Shafait, F., van Beusekom, J., Keysers, D., Breuel, T.M.: Document cleanup using page frame detection. International Journal on Document Analysis and Recognition 11(2), 81–96 (2008)
12. Shafait, F., Breuel, T.M.: Document image dewarping contest. In: Proc. Int. Workshop on Camera-Based Document Analysis and Recognition, Curitiba, Brazil, pp. 181–188 (September 2007)
13. Shafait, F., Breuel, T.M.: The effect of border noise on the performance of projection based page segmentation methods. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(4), 846–851 (2011)
14. Shafait, F., Keysers, D., Breuel, T.M.: Performance evaluation and benchmarking of six page segmentation algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(6), 941–954 (2008)
15. Ulges, A., Lampert, C., Breuel, T.M.: Document capture using stereo vision. In: Proc. ACM Symposium on Document Engineering, pp. 198–200. ACM (2004)
16. Vincent, L.: Google book search: Document understanding on a massive scale. In: Int. Conf. on Document Analysis and Recognition, Curitiba, Brazil, pp. 819–823 (September 2007)
17. Yamashita, A., Kawarago, A., Kaneko, T., Miura, K.: Shape reconstruction and image restoration for non-flat surfaces of documents with a stereo vision system. In: Proc. Int. Conf. on Pattern Recognition, pp. 482–485 (2004)

A Method for Camera-Based Interactive Whiteboard Reading

Szilárd Vajda, Leonard Rothacker, and Gernot A. Fink

TU Dortmund, Department of Computer Science,
Dortmund, Germany
{szilard.vajda,leonard.rothacker,gernot.fink}@udo.edu

Abstract. Recognizing mind maps written on a whiteboard is a challenging task due to the unconstrained handwritten text and the different graphical elements — i.e. lines, circles and arrows — available in a mind map. In this paper we propose a prototype system to recognize and visualize such mind maps written on whiteboards. After the image acquisition by a camera, a binarization process is performed, and the different connected components are extracted. Without presuming any prior knowledge about the document, its style, layout, etc., the analysis starts with connected components, labeling them as text, lines, circles or arrows based on a neural network classifier trained on some statistical features extracted from the components. Once the text patches are identified, word detection is performed, modeling the text patches by their gravity centers and grouping them into possible words by density based clustering. Finally, the grouped connected components are recognized by a Hidden Markov Model based recognizer. The paper also presents a software tool integrating all these processing stages, allowing a digital transcription of the mind map and the interaction between the user, the mind map, and the whiteboard.

Keywords: whiteboard reading, unconstrained document layout analysis, handwriting recognition.

1 Introduction

Nowadays, in the field of handwriting recognition the focus is shifted from classical topics like bank checks or postal documents recognition [14] to more challenging topics like historical documents recognition, personal memos or sketch interpretation [15] and lately to recognition of unconstrained whiteboard notes [11, 13]. The later is in the focus of the attention because it deals with an unconstrained type of documents with no specific writing style, layout, etc.

Doing collaborative work (e.g. brainstormings, discussions, presentations) is quite common a in corporate or academical environment. However, there is just a limited amount of work [11, 13, 18] to embed this whiteboard outcome in a smart environment scenario (e.g. a conference room). To provide not just a digital capture of the whiteboard, but also the recognized content in an interactive software framework, is one of the final goals of such a smart room.

Instead of tackling this issue by some specific (sometimes costly) hardware (e.g. special whiteboard, several cameras, pen, wireless microphone proposed by the e-Learning

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 112–125, 2012.

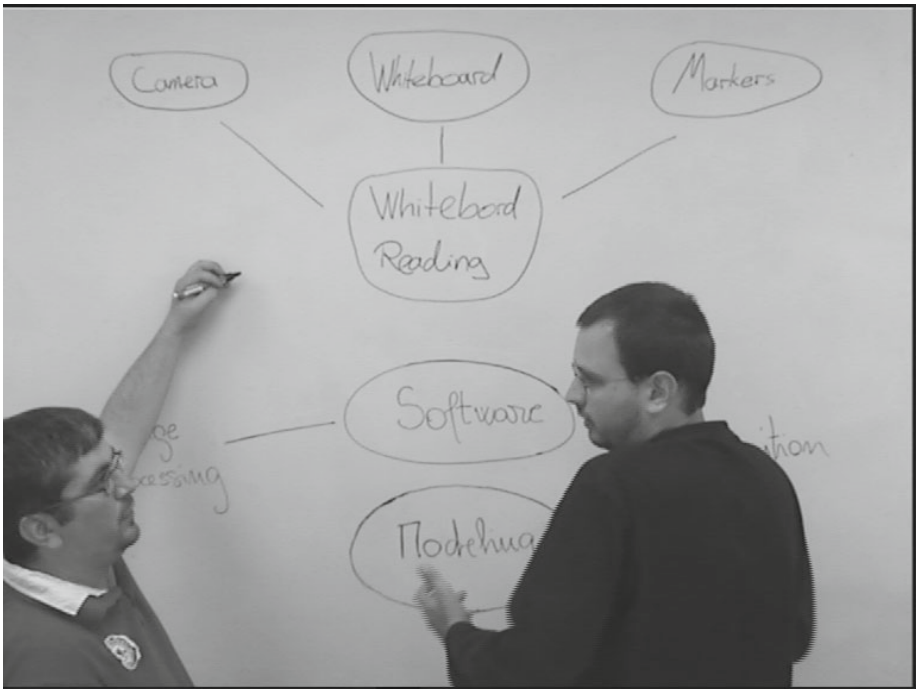


Fig. 1. Scene from a mindmap creation process around the idea of ”Whiteboard reading”

system [18]), we propose a system which uses only regular hardware (a simple whiteboard, markers, a low-resolution active camera and a projector) available in each conference room. Such hardware setup provides us a natural environment to actively support the collaborative mind mapping [3], allowing the users to keep their old habits and writing down their ideas using just the whiteboard markers without bothering about some special equipment. The focus is rather on the content and not on the layout. Such a mind map creation process is depicted in Fig. 1.

The current system focuses on two main aspects. First, we will present the system capable to recognize on-line the different text and non-text components and secondly, we will concentrate on the digital outcome of that recognition process: a digital, editable mind map framework and the interaction between the static whiteboard content, the user and the projected and already recognized mind map. Such an interaction is missing from the currently available systems. The scientific challenges lie in the facts that we analyze the documents without any prior knowledge, no curve tracing is considered, and due to the reduced number of such mind maps, the handwriting recognizer is trained on a completely different data.

The following sections of the paper are organized as follows. Related works concerning the whiteboard recognition will be discussed in the next section. Section 3 describes in detail the complete whiteboard reading system. Section 4 is completely dedicated to the data and the experimental setup. Finally, Section 5 summarizes and highlights the strengths of the presented reading system.

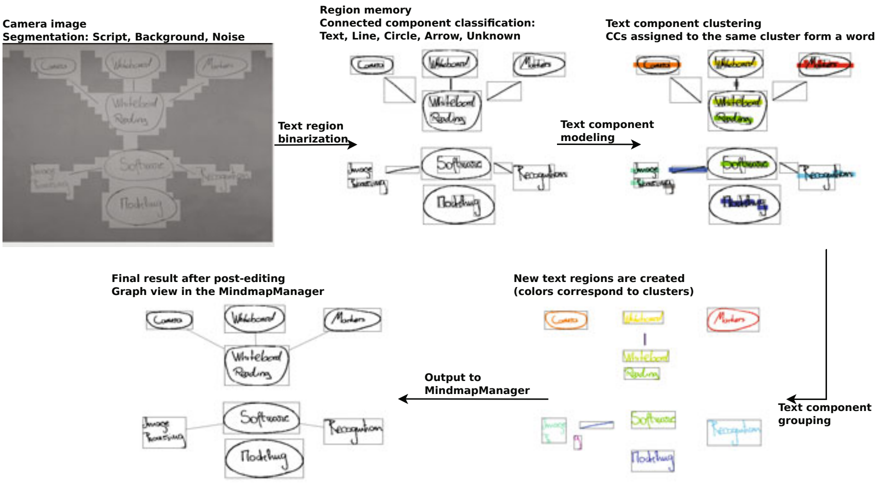

Camera image
Segmentation: Script, Background, Noise
Text region binarization
Region memory
Connected component classification:
Text, Line, Circle, Arrow, Unknown
Text component modeling
Text component clustering
CCs assigned to the same cluster form a word
Text component grouping
New text regions are created
(colors correspond to clusters)
Output to MindmapManager
Final result after post-editing
Graph view in the MindmapManager

Fig. 2. Overview of the proposed whiteboard reading system

2 Related Work

Over the last two decades an impressive progress has been achieved in the field of handwriting recognition, even for large vocabularies [12] and multi-writer scenarios. However, the main constraint was always the same. To produce those sound results clean and well segmented data was necessary. In the whiteboard reading scenario addressed in this paper such presumptions can not hold. In particular, for collaborative works like mind mapping, different layouts, writers, styles and text and non-text mixture should be handled.

The very first attempt to handle such a challenging task was addressed by Wienecke et al. [16], where complete handwritten sentences were recognized using a low-resolution camera. The proposed prototype system was quite promising, however it was able to recognize text only.

A similar type of research was conducted in [7,8], where the data acquisition was performed on-line using an infrared sensor for tracking a special pen. Even though the results are sound, it to be noted, that only clear, well structured text lines were recognized.

To recognize Japanese characters on a whiteboard, the authors in [19] consider a complex hardware scenario including two cameras and a special pen to capture the writing. The text detection is not anymore performed by the system but rather by the software provided by the pen manufacturer. The system is used in an e-Learning scenario.

In a more recent work [11] of ours, we focused on a similar task, recognizing well-structured handwritten whiteboard paragraphs, considering only a camera and no on-line information. The results are really promising, however, the text detection on the whiteboard is based on some connected component (CC) estimation which is very rigid due to the usage of some thresholds.

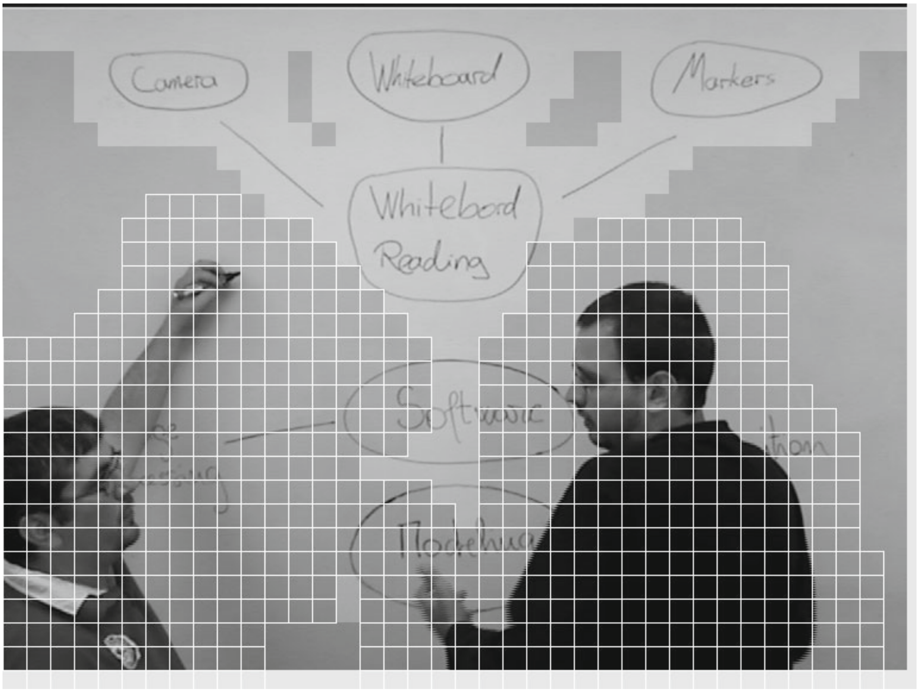


Fig. 3. The segmentation of the Fig. 1 into text, background and noise

Finally, in our recent work [13] we addressed the mind map recognition scenario (see Fig. 1), where beside the text components graphical elements like lines, circles, arrows were detected and a totally unconstrained document layout was analyzed. The goal of the current work is to improve that system by adapting the recognition to the different changing layouts, reconstruct the mind map and introduce a certain type of interaction between user, document and whiteboard.

3 Whiteboard Reading System

In this section we concentrate on the whiteboard reading system, describing the complete process starting from the image acquisition, throughout the different processing steps and finally the recognition and the user interaction with the system. A system overview with its particular processing stages is shown in Fig. 2.

3.1 Image Acquisition and Camera-Projector Calibration

For image acquisition a camera and for user interaction a projector must be directed to the whiteboard. The camera is capturing the whole mind map creation process. Low-resolution gray level camera images are used for further processing (see Section 3.2).

The camera-projector calibration is needed to project content to the whiteboard that is derived from the camera image. In order to project information on the whiteboard for user interaction (see Section 3.7), a mapping between the camera- and the projection image coordinate systems has to be obtained. The projected image contains additional user information and is shown on the whiteboard using the projector. This way the projection image can be seen as an overlay to the mind map drawn with a marker by the user.

For calibration a chessboard is projected on the whiteboard that is captured with the camera. Because the chessboard is rendered in the projection image, its chessboard corner coordinates are known in the projection image coordinate system. By finding chessboard corners in the camera image correspondences between both images are obtained. Finally, a homography can be estimated that maps each point from the camera coordinate system to the projection image coordinate system. The chessboard corner localization and homography estimation that we use are both freely available [1].

3.2 Image Segmentation

The purpose of image segmentation is to separate elements written on the whiteboard with a marker from the whiteboard background and noisy image parts. Noisy parts are for example regions where the user stands (see Fig. 1). Afterwards the regions containing written content are further segmented by categorizing them into different mind map elements (text, line, circle, arrow).

Segmentation of the Camera Image. After image acquisition the objective is to extract only the content written on the whiteboard. This relevant information is then added to a binary region memory (also refer to Fig. 1). The region memory represents the current state of written content on the whiteboard and is robust to irrelevant changes in the camera image, like illumination or particular users standing in front of the whiteboard. Therefore the general assumption is that the camera image does not contain anything but the interior of the whiteboard. The camera and the whiteboard are fixed. In this scenario the system has to handle images that can consist of three different regions, namely:

- text (indicated by bright blocks in Fig. 3).
- background (indicated by dark blocks in Fig. 3).
- noise (indicated by blocks with grid pattern in Fig. 3).

As proposed by [16], segmentation is not done on pixel but on block level. The image is therefore divided into two layers of overlapping blocks. Each block is now segmented into one of the formerly mentioned categories on the basis of three features: gradients, gray level and changes between two consecutive images.

The first feature (ξ^k_{edge}) uses image gradients: A block breaks the edge threshold (Θ_{edge}) if it exceeds a number of gradients of a minimum magnitude. Gradient magnitudes are derived by convolving the block with horizontal and vertical Sobel masks. The minimum magnitude is an additional parameter that has to be given.

The second feature (ξ^k_{gray}) uses the average gray level: A block breaks the gray level threshold (Θ_{gray}) if its average gray level falls beyond the scaled overall image average gray level. That means that the block is darker than the overall image.

The third feature (ξ_{diff}^k) uses the change between two consecutive images: A block breaks the difference threshold (Θ_{diff}) if the sum-of-absolute-differences error metric computed between corresponding blocks exceeds the threshold value.

The categorization is done depending on whether the blocks meet the following criteria:

- Text:
$$\left(\xi_{\text{edge}}^k > \theta_{\text{edge}}\right) \wedge \left(\xi_{\text{gray}}^k > \theta_{\text{gray}}\right) \wedge \left(\xi_{\text{diff}}^k < \theta_{\text{diff}}\right) .$$
Text blocks contain many strong gradients and have a bright average gray level because the pen stroke is the only dark element on a bright background. Furthermore, there should be no movement since both camera and whiteboard are supposed to stand still.
- Noise:
$$\left(\xi_{\text{gray}}^k \leq \theta_{\text{gray}}\right) \vee \left(\xi_{\text{diff}}^k \geq \theta_{\text{diff}}\right) .$$
Noise blocks are mainly caused by users being captured by the camera. In contrast to the whiteboard their appearance is generally darker. Additionally, their main activity will be writing to the whiteboard, so we can assume that blocks containing users also contain movement.
- Background: If the block is considered neither text nor noise. The camera is supposed to capture only the interior of the whiteboard, thus a block can be considered background if it is not text or noise.

After categorizing all blocks the region memory can be updated.

Noise blocks are discarded because the whiteboard is potentially occluded at those locations. To be even more robust also blocks in a noise block's local neighborhood can be discarded. The occurrence of eventually appearing parts of the user's shape in the region memory can be minimized this way. The information contained in a falsely discarded block will simply be added to the region memory later.

Background blocks do not contain any written content, so the corresponding regions in the region memory can be erased.

Finally text blocks are binarized with the local Niblack method [10] and inserted into the region memory if their XOR errors with the region memory exceed a certain empirically selected threshold. This way the memory does not get updated for very small changes in the camera image but only if there is a modification to the written content. Those small changes are likely to be caused by illumination changes in the conference room. The different thresholds considered in the segmentation process were selected based on trial runs. Though the detection is stable, considerable change in lighting conditions requires a new threshold set. For further details please refer to [16].

The result as depicted in Fig. 2 consists of a binary representation of the whiteboard content and can be used for further processing.

Segmentation of the Whiteboard Image. A key issue to success is the accurate segmentation of the whiteboard content. We separate the whiteboard from the rest of the scene (see Section 3.2), but we do not have any prior information about the content itself. To recognize and reconstruct the mind map, we need to separate text elements from non-text items, namely in this scenario, lines, circles and arrows. The detection process is based on connected components (CC) extracted from the binarized image. Working with CC is suitable as the connected components are easy to extract and no specific prior knowledge is necessary.

Instead of using heuristics - rooting from the work of Fletcher and Kasturi [5], we propose a solution to classify CCs based on statistical learning. A descriptor of 12 components (i.e. contrast, edge density, homogeneity, number of foreground gray levels, foreground mean gray level, relative amount of gradient orientations, Sobel gradient orientation and magnitude, etc.) is extracted from each CC and a multi-layer perceptron is meant to classify the pixel patches into text, line circle and arrow. For more details, please refer to [13]. This text component detector is suitable not only for Roman script but also for Chinese, Arabic or Bangla, where even more complex characters shapes will occur.

3.3 Layout Analysis

The layout analysis of a document consists of identifying the baseline elements composing the document and their spatial relationship among each other. While for printed documents a certain type of regularities like font type, font size, line structures, etc. can be detected, in a handwritten mind map documents none of these is to be identified, hence the layout analysis is more challenging in such unconstrained handwritten document scenarios.

Layout Modeling. As described above, we separate first text items from non-text items. For further processing we will concentrate our effort to model only the different text patches. The lines, circles and arrows detected previously will serve to build the digital representation of the mind map into the so-called ”MindMap Manager”, discussed later in Section 3.6.

Our proposition is to adapt the model to the analyzed document considering the gravity centers of the text CCs (see Fig. 2a) and model the structure of the text patches (CCs) throughout these points. For each text component, the gravity center is calculated. At the left and right side of the bounding box a new center is calculated inheriting the height form the original gravity center. For larger components, exceeding the *average width*, estimated over all connected components from the document, at each slice of *average width*/4 of the CC, a new gravity center is computed w.r.t. the pixels counted in that window.

3.4 Word Detection

Once the modeling part is done, the different gravity centers will form ”dense” regions (see Fig. 2) corresponding to possible words. These agglomerations into different clusters need to be identified in order to separate the different words from each other. For this purpose the DBSCAN algorithm [2] has been considered. While other clustering methods rely mainly on some distance metrics, in this case the distance is combined with the density.

The gravity centers will be clustered not only by the distances (between the different text patches), but also by the density which is definitely higher around the different text components (see Fig. 2).

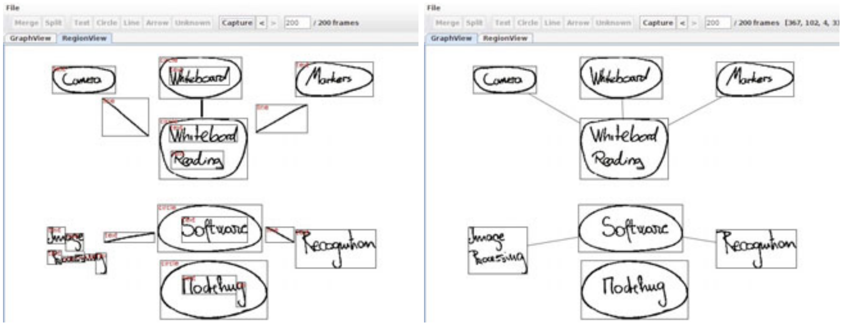

File
Merge Split Text Circle Line Arrow Unknown Capture < > 200 / 200 frames
GraphView RegionView
Camera
Whiteboard
Markers
Whiteboard Reading
Image Processing
Software
Recognition
Modelling
File
Merge Split Text Circle Line Arrow Unknown Capture < > 200 / 200 frames [367, 102, 4, 3
GraphView RegionView

Fig. 4. User interface of the *Mindmap Manager* showing the region view on the left and the graph view on the right. In the region view segmentation and recognition results can be corrected. The graph view shows the final graph representation of the mind map.

Let $D_n = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ be the coordinates of the gravity centers, where $x_i, y_i \in R^+$ and n denotes the number of gravity centers in the set.

Let us denote by β-neighborhood of a point $p_k \in D_n$ the $N_\beta(p_k) = \{(p_k) \in D_n | dist(p_k, p_l) < \beta, k \neq l\}$, where $dist()$ is the Euclidean distance.

Considering the β-neighborhood of a point, we define the notion of: p_k is density reachable from p_l if $p_k \in N_\beta(p_l)$ and $|N_\beta(p_l)| \geq P_{min}$, where P_{min} is the minimal number of points (gravity centers) which should be around point p_k.

The proposed clustering process is based on the β-neighborhood of a given point. We select as belonging to one cluster all the points which are density reachable considering a given number of k (number of neighbors). The expansion of each cluster is based on this idea allowing to get rid of the noisy points which density reachability indices are lower than for the others. For more details about the clusters expansion, please refer to work [2]. Finally, the original CCs' gravity centers are mapped to the different clusters established by DBSCAN (see Fig. 2).

3.5 Word Recognition

For the recognition, we use the same recognizer that in our previous work was successfully applied to the task of reading text paragraphs in high-resolution whiteboard images [11]. Handwritten words are modeled by semi-continuous character HMMs. A sliding window is applied on the normalized text snippets considering 8 pixel wide analysis window with 25% overlap. For each frame a set of nine geometric features and the approximation of their first derivatives are computed [17]. In total, 75 models considering upper and lower case letters, numerals and punctuation marks have been trained on the IAM database [11]. The HMM models are decoded with a time-synchronous Viterbi beam search algorithm [4].

3.6 The Mindmap Manager

The *Mindmap Manager* is the front end of the presented system. Segmentation, grouping and recognition results are consecutively — for each change in the region memory

— saved and made accessible to the user for post-editing and exporting. Fig. 4 shows its user interface. The region view (left side of Fig. 4) contains editing functionality for incorrectly segmented or classified components. Please note that those components refer to classified CCs that in case of text elements might have undergone further grouping. After selecting a component its category can be changed or the entire component can be split horizontally or vertically. If the categories of two different components are compatible, they can be merged.

The graph view (right side of Fig. 4) contains a graph representation of the mind map. Text and circle elements are treated as nodes and lines and arrows are treated as edges (compare region and graph view in Fig. 4). This way the user is able to rearrange the layout of the nodes by simply dragging and dropping them. Connected edges will follow. Besides the possibility to rearrange the mind map layout the graph view has no further editing capabilities and is rather intended to contain the final outcome of the system. A compatibility with existing digital mind map formats would allow to use other, already existing tools to further manipulate the documents.

The graph representation has to be created from the previously mentioned components. Because there is no prior knowledge of which nodes are to be connected by a line, an estimation is necessary. By transferring components classified as lines to Hough space [6], a parametric representation of the line can be estimated. Finally, two nodes being connected by the line component are determined trough intersection of the parametric line with neighboring components.

3.7 Interaction with the Whiteboard

The purpose of user interaction is to give a feedback of segmentation and recognition results while the mind map creation is still in progress. This way the user can react accordingly (e.g. writing things clearer) to improve the overall performance. The feedback is given by highlighting newly recognized elements on the whiteboard using the projector. The highlighting color indicates the classification result (text, line, circle, arrow). To use the camera and the projector in conjunction a camera-projector calibration has to be performed initially (also see Section 3.1).

Segmentation and recognition results can be retrieved whenever the region memory changes (see Section 3.2). Ideally those updates to the region memory occur only if there is a change to the written content on the whiteboard. In such cases changed CCs are determined by computing a difference (XOR) image in the region memory. From their bounding boxes highlighting events are now generated that additionally contain the category (text, line, circle, arrow) for highlighting in a specific color. The bounding boxes are given in camera image coordinates and have to be mapped to projection image coordinates for rendering. This mapping can be computed through the formerly estimated homography (see Section 3.1). After rendering, the user will see a colored rectangle around all recently changed parts of the mind map for a few seconds. In order to be more robust to false updates of the region memory (e.g. due to illumination changes), highlighting events will only be generated if the change in the region memory exceeds a certain threshold.

Finally we have to deal with the effect that projections to the whiteboard are also captured by the camera. This way projections can result in CCs being extracted that do not correspond to written content on the whiteboard. The idea is to filter the area in the camera image, where there will be a projection, from the region memory.

4 Experiments

In this section a brief description of the data and the results achieved by the described method will be presented.

4.1 Data Description

The dataset consist of 31 mind maps written by 11 different writers around the topics ”study”, ”party” and ”holiday”. 2 writers sketched only 2 mind maps. All writers were provided with different color markers. While the usage of some words was imposed, the writers were not restricted in creating their own mind maps around the previously mentioned topics. Once the mind map was ready a photo (2048x1536 resolution) of the whiteboard was taken. The data is annotated with respect to the nature of connected components (line, circle, text, arrow) and words [13].

4.2 Results

To evaluate thoroughly the method we need to evaluate the text detection solution, the subsequent modeling strategy and the text recognition. As the text detection method was originally proposed in [13], we just briefly describe the results, and we focus rather our evaluation on the layout analysis and the recognition. For more details on the results concerning the text detection, please refer to [13].

Text Detection: The neural network provides an average recognition score of 95.7% for the different CCs as being text, line, circle or arrow. However, while for text components the recognition scores are high (99.4%), for lines and arrows there are elevated confusion rates.

Layout Analysis: For the evaluation of the proposed method we use the method introduced in the context of the ICDAR 2005 Text Locating Competition [9]. That way we produce comparable and comprehensible evaluation results. The bounding boxes of the annotated ground truth T and the agglomerated text components E are compared – the larger the overlap of the bounding boxes, the higher the level of match. A match m_p between two rectangles r, r' is defined as the quotient of their intersection area and their union area:

$$m_p = \frac{A(\bigcap(r, r'))}{A(\bigcup(r, r'))}. \tag{1}$$

The evaluation scheme is based on *precision* and *recall*. Having a binary answer to whether there is a fitting ground-truth rectangle to an estimated one or not would not cope with partial matches. This is why the quality for a single match m_p in this case lies in the range of $[0; 1]$. In order to calculate these adapted versions of precision and recall the best match between a rectangle within the agglomerations and all rectangles

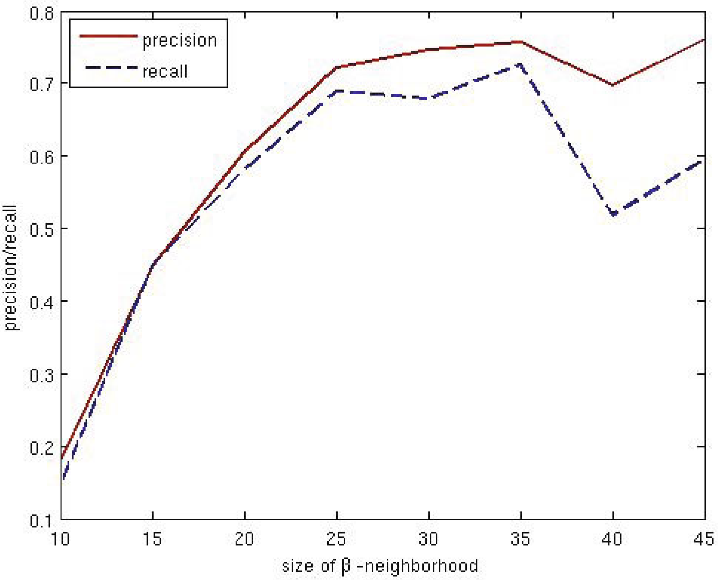


Fig. 5. Layout analysis results (Precision and recall) for the document shown in Fig. 7

within the set of annotations is taken into consideration – and vice versa. The best match $m(r, R)$ of a rectangle r within a set of other rectangles R is defined as:

$$m(r, R) = \max \{m_p(r, r')|r' \in R\}. \tag{2}$$

The *recall* then is the quotient of the sum of the best matches of the ground truth among the agglomerated areas and the number of all annotated bounding boxes within the ground truth.

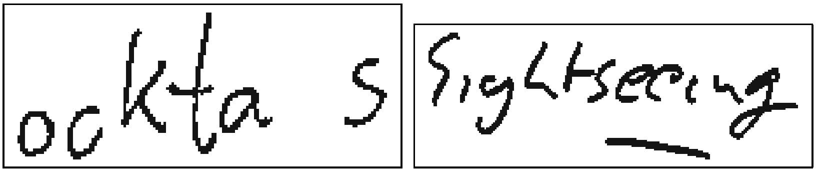
$$recall = \frac{\sum_{r_t \in T} m(r_t, E)}{|T|}. \tag{3}$$

The *precision* relates to the quotient of the sum of the best matches of the agglomerated areas among the annotated regions and the number of all agglomerated areas:

$$precision = \frac{\sum_{r_e \in E} m(r_e, T)}{|E|}. \tag{4}$$

We evaluated the output of the agglomeration (modeling) using both schemes described above. In Fig. 5 we display a typical result of the hierarchical clustering, stating in this case the maxima for precision and recall at 75% and 72%, respectively. The average recall value for the test documents is 67.09%. The main error source is due to the high number of non-text patches labeled as text not retrieved anymore in the ground truth.

While in some cases, the agglomeration is successful, in some other cases it fails because of some CCs were recognized as non-text (e.g. M in "Motto" or D in "Dance"



(a) missing letters (b) agglomeration with noise

Fig. 6. Typical error cases occurred in the agglomeration

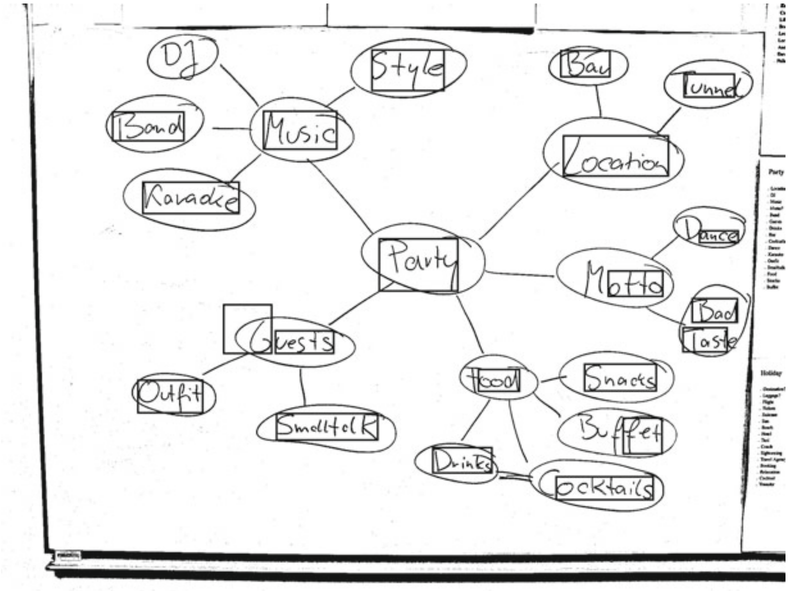


Fig. 7. Layout analysis results on an exemplary mind map

in Fig. 7) or due to some distances which lead to agglomeration or separation (see “Guests” in Fig. 7) of different text items. Overall, in 211 cases the agglomeration produced non-text word hypothesis, in 194 cases some parts of the word (mainly characters) were missing. Finally, in 353 cases complete words, or words preceded or followed by noises (mainly lines) were detected. Some typical agglomeration errors are depicted in Fig. 6. Also a common error is encountered, namely the first letters of the words are often are connected with the surrounding circles, hence the letter is analyzed with the circle and classified as graphical element (e.g. F in ”Food” or T and l in ”Tunnel” in Fig. 7). This is one major limitation of the connected component based methods. To separate these elements, more sophisticated methods like skeletonization and curves tracing solutions should be applied.

Word Recognition: The recognition of the word snippets by the HMM are reported only on those 353 words considered as successful (complete) for the grouping. The lexicon (164 entries) was generated from the transcripts including all tentatively written words irrespective of segmentation errors. To reject test fragments with erroneous ink elements a rejection module was used, defined as an arbitrary sequence of character models. The overall word recognition score in the different snippets is 40.5%. 83.3% of the snippets were recognized correctly (i.e one word snippets). The low scores can be explained by the fact that the recognizer is trained on completely different data, while the recognition is performed on low-resolution image snippets, with huge writing style variations and containing also additional noise components inherited from the grouping process.

5 Summary

In this paper we proposed a basic prototype reading system to automatically recognize mind maps written in an unconstrained manner on a whiteboard. Instead of considering expensive equipment, only common tools like e.g. whiteboard, markers, camera, and a projector were considered in the recognition scenario, usually available items in a conference room.

Instead of establishing some rules, the method adapts to the layout of each analyzed document. The modeling of the text components by their gravity centers followed by Density Based Spatial Clustering will provide the solution to merge the detected text patches (connected components) into words which serve as input for a handwriting recognizer. For this preliminary work, the recognition results, even though the recognizer was trained on completely different data, are not satisfying yet, but with some post-processing of the grouping more complete and accurate word agglomerations can be submitted to the recognizer. The software tool and the interactivity with the whiteboard provides a straightforward solution for a human-computer interaction in this challenging automatic whiteboard reading scenario.

Acknowledgment. This work has been supported by the German Research Foundation (DFG) within project **Fi799/3**.

References

1. openCV (Open Source Computer Vision) library, http://opencv.willowgarage.com/wiki/
2. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: International Conference on Knowledge Discovery and Data Mining, pp. 226–231 (1996)
3. Farrand, P., Hussain, F., Henessy, E.: The efficiency of the ”mind map” study technique. Journal of Medical Education 36(5), 426–431 (2003)
4. Fink, G.A.: Markov Models for Pattern Recognition, From Theory to Applications. Springer, Heidelberg (2008)
5. Fletcher, L., Kasturi, R.: A robust algorithm for text string separation from mixed text/graphics images. IEEE Trans. on Pattern Analysis and Machine Intelligence 10(6), 910–918 (1988)

6. Gonzalez, R.C., Woods, R.E.: Digital Image Processing, 2nd edn. Addison-Wesley Longman Publishing Co., Inc., Boston (2001)
7. Liwicki, M., Bunke, H.: Handwriting recognition of whiteboard notes. In: Conference of the International Graphonomics Society, pp. 118–122 (2005)
8. Liwicki, M., Bunke, H.: Handwriting recognition of whiteboard notes – studying the influence of training set size and type. International Journal of Pattern Recognition and Artificial Intelligence 21(1), 83–98 (2007)
9. Lucas, S.M.: Text locating competition results. In: International Conference on Document Analysis and Recognition, pp. 80–85 (2005)
10. Niblack, W.: An introduction to digital image processing. Strandberg Publishing Company, Birkeroed (1985)
11. Plötz, T., Thurau, C., Fink, G.A.: Camera-based whiteboard reading: New approaches to a challenging task. In: International Conference on Frontiers in Handwriting Recognition, pp. 385–390 (2008)
12. Plötz, T., Fink, G.A.: Markov models for offline handwriting recognition: a survey. International Journal on Document Analysis and Recognition 12(4), 269–298 (2009)
13. Vajda, S., Plötz, T., Fink, G.A.: Layout analysis for camera-based whiteboard notes. Journal of Universal Computer Science 15(18), 3307–3324 (2009)
14. Vajda, S., Roy, K., Pal, U., Chaudhuri, B.B., Belaid, A.: Automation of Indian postal documents written in Bangla and English. International Journal of Pattern Recognition and Artificial Intelligence 23(8), 1599–1632 (2009)
15. Weber, M., Eichenberger-Liwicki, M., Dengel, A.: A.scatch - a sketch-based retrieval for architectural floor plans. In: International Conference on Frontiers of Handwriting Recognition, pp. 289–294 (2010)
16. Wienecke, M., Fink, G.A., Sagerer, G.: Towards automatic video-based whiteboard reading. In: International Conference on Document Analysis and Recognition, Washington, DC, USA, pp. 87–91 (2003)
17. Wienecke, M., Fink, G.A., Sagerer, G.: Toward automatic video-based whiteboard reading. International Journal on Document Analysis and Recognition 7(2-3), 188–200 (2005)
18. Yoshida, D., Tsuruoka, S., Kawanaka, H., Shinogi, T.: Keywords recognition of handwritten character string on whiteboard using word dictionary for e-learning. In: Proceedings of the 2006 International Conference on Hybrid Information Technology, ICHIT 2006, vol. 01, pp. 140–145. IEEE Computer Society, Washington, DC, USA (2006)
19. Yoshida, D., Tsuruoka, S., Kawanaka, H., Shinogi, T.: Keywords recognition of handwritten character string on whiteboard using word dictionary for e-learning. In: International Conference on Hybrid Information Technology, pp. 140–145 (2006)

Border Noise Removal of Camera-Captured Document Images Using Page Frame Detection

Syed Saqib Bukhari[1], Faisal Shafait[2], and Thomas M. Breuel[1]

[1] Technical University of Kaiserslautern, Germany
{bukhari,tmb}@informatik.uni-kl.de
[2] German Research Center for Artificial Intelligence (DFKI),
Kaiserslautern, Germany
faisal.shafait@dfki.de

Abstract. Camera-captured document images usually contain two main types of marginal noise: textual noise (coming from neighboring pages) and non-textual noise (resulting from the page surrounding and/or binarization process). These types of marginal noise degrade the performance of the preprocessing (dewarping) of camera-captured document images and subsequent document digitization/recognition processes. Page frame detection is one of the newly investigated areas in document image processing, which is used to remove border noise and to identify the actual content area of document images. In this paper, we present a new technique for page frame detection of camera-captured document images. We use text and non-text contents information to find the page frame of document images. We evaluate our algorithm on the DFKI-I (CBDAR 2007 Dewarping Contest) dataset. Experimental results show the effectiveness of our method in comparison to other state-of-the-art page frame detection approaches.

Keywords: Border Noise Removal, Page Frame Detection, Camera-Captured Document Images.

1 Introduction

When a page of a book is photographed, the captured image usually contains undesired parts of text from the neighboring page. Besides, some regions of background (table surface etc.) also appear in the image. These undesired regions of the image are usually referred to as border noise [11]. These types of border noise are called textual noise and non-textual noise, respectively. When textual noise regions are fed to a character recognition engine, extra characters appear in the output of the OCR system along with the actual contents of the document. These extra characters in the OCR output result in inaccurate retrieval results, since the keywords given by the user might match some text from the textual noise instead of the actual document contents. Non-textual noise, on the

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 126–137, 2012.

other hand, makes further processing of document like text-line extraction or dewarping a difficult task.

The problem of border noise is also well-known in the domain of scanned document analysis. Many approaches have been reported in literature to deal with border noise of scanned images. Most of the these approaches (e.g. [8,1,7]) focus only on removal of non-textual noise. Cinque et al. [5] propose an algorithm for removing both textual and non-textual noise from grayscale images based on image statistics like horizontal/vertical difference vectors and row luminosities. The method presented in [10] detects border noise using different black/white filters. These approaches rely on certain assumption about scanned documents (like an axis-aligned pattern of noise or presence of thick black non-textual noise regions). However, these assumptions do not hold for camera-captured documents since the document can be captured from any perspective (hence page border is not axis-aligned any more). Besides, the captured document is binarized using a local thresholding method like [12] (hence no thick black regions appear in the binarized image).

Instead of identifying and removing noisy components themselves, some methods focus on identifying the actual content area or the page frame of the documents [13,14]. The page frame of a scanned document is defined as the smallest region (rectangle or polygonal) that encloses all the foreground elements of the document image. The method presented in [13] finds the page frame of structured documents (journal articles, books, magazines) by exploiting their text alignment property. The method by Fan et. al [6] estimates page frame using a rectangular active contour. This method is not directly applicable to page frame detection of camera-captured documents due to the presence of perspective distortions. Stamatopoulos et al. [14] proposed a method for splitting double-page scanned document images into two pages without noisy borders. Their method is based on vertical and horizontal white runs projections.

So far very few approaches are developed for camera-captured document images. Shafait et al. [13] applied their page frame detection approach to camera-captured document images. When applied to camera-captured document images, the method focuses on finding the left and right page border lines only using a geometric matching method. The method gives good results for camera-captured document images, but does not remove border noise on the upper and lower sides of the document images. Stamatopoulos et al. [15] proposed an algorithm for detecting borders of camera-captured document images based on projection profile. This method works well for a small degree of skew/curl in document images, but can not handle document images with a large degree of skew/curl, which is usually present in hand-held camera-captured documents.

In this paper we present a page frame detection method for camera-captured document images. The method starts with preprocessing which includes binarization and text and non-text segmentation steps. Then, text-lines are detected by applying the ridge based text-line finding method [3]. Finally, page frame is detected by using text-lines and text and non-text information. Our method can detect the upper and lower borders together with the left and right borders, and is robust to a large degree of skew/curl in camera-captured document images.

The rest of the paper is organized as follows. The proposed page frame detection method is described in Section 2. Experiments and results are discussed in Section 3. Section 4 presents our conclusions.

2 Page Frame Detection Method

The proposed page frame detection method consists of three main steps: i) preprocessing, ii) text-line detection, iii) page frame detection. Preprocessing (binarization and text and non-text segmentation) of camera-captured document images is discussed in Section 2.1. Text-line detection method is described in Section 2.2. Page frame detection method using text-line and text and non-text contents information is explained in Section 2.3.

2.1 Preprocessing

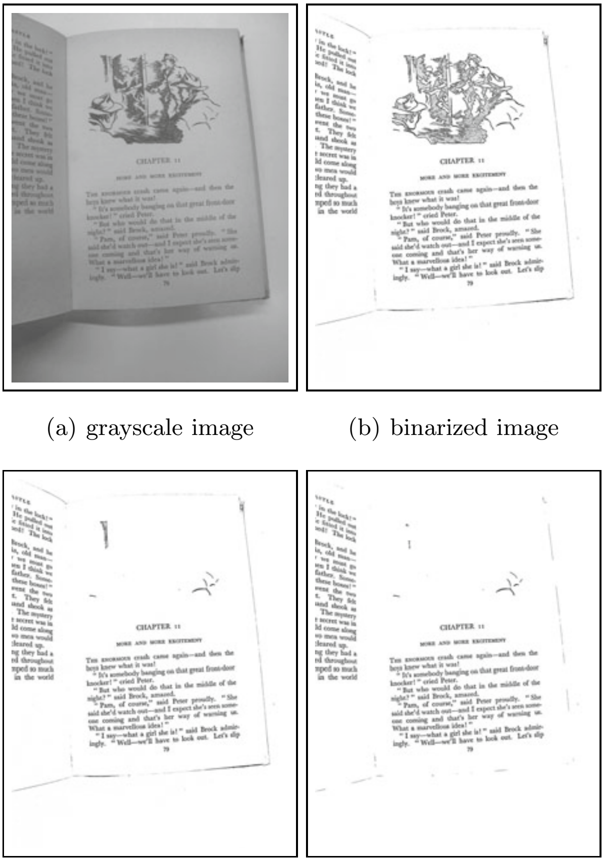
Our preprocessing approach mainly consists of binarization and text segmentation steps. An input grayscale camera-captured document image is first binarized using the adaptive thresholding technique mentioned in [16], which is described as follows: "for each pixel, the background intensity $B(p)$ is defined as the 0.8-quantile in a window shaped surrounding; the pixel is then classified as background if its intensity is above a constant fraction of $B(p)$". An example grayscale document and its correspoding binarized document images are shown in Figure 1(a) and Figure 1(b), respectively.

We presented a multiresolution morphology based text and non-text segmentation algorithm in [4], that can segment text from different types of non-text elements like halftones, drawing, graphics, etc. In this approach, the resolution of an input (binary) document image is reduced iteratively by applying *threshold reduction* strategy for removing text elements and keeping non-text elements. The reduced image, after appropriate expansion, is used as non-text mask image. The segmented text form the binarized document image (Figure 1(b)) is shown in Figure 1(c).

After text and non-text segmentation, a heuristic size based noise cleanup process is applied for removing comparatively large (marginal noise) and small (salt-and-pepper noise) elements as follows. A connected component is considered as a large noisy component if its height/width is greater than 10% of document height/width or greater than 7 standard deviation above mean height/width. Similarly, a connected component is removed as a small noisy component if its area is smaller than $\frac{1}{3}^{rd}$ of the mean area. The document image in Figure 1(c) after noise cleanup is shown in Figure 1(d).

2.2 Text-Line Detection

We introduced a ridge based text-line extraction method for warped camera-captured document images in [3]. Our ridge based text-line finding method consists of two standard and easy to understand image processing algorithms: (i)



(a) grayscale image

(b) binarized image

(c) segmented text parts

(d) after noise cleanup

Fig. 1. Preprocessing: (a) a sample grayscale camera-captured document image, (b) binarized document image, (c) segmented text parts of the binarized image, (d) cleaned image after noise removal

Gaussian filter bank smoothing and (ii) ridge detection. The ridge based text-line detection method is briefly described here for the completeness of this paper.

First, the ranges for Gaussian filter's parameters, i.e. σ_x, σ_y and θ, are defined empirically for generating a set of filters. Then, the set of filters is applied to each pixel and the maximum output response is selected for the smoothed image. Figure 2(a) shows the smoothed version of the document image as shown in Figure 1(d). After smoothing, text-lines are extracted by detecting ridges from the smoothed image.

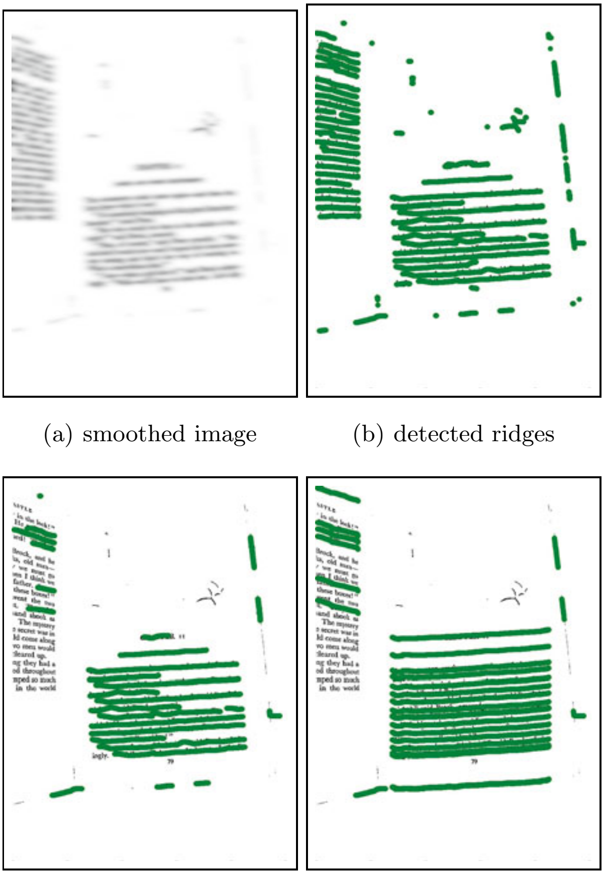
Most of the detected ridges, that are shown in Figure 2(c), are situated over text-lines. Some of them are also very small in size as compared to others, and some on them lie over marginal textual noise. A ridge is considered as a small-size ridge if its length is smaller than $\frac{1}{10}^{th}$ of document width. Textual-noise is usually present in the left and right corners of the document. Therefore, a ridge is considered as a ridge over textual-noise if its starting/ending point exists very close (within ±25 pixels) to the left/right corner of document image and its length is smaller than $\frac{1}{5}^{th}$ of document width. After filtering small-size ridges and ridges over textual-noise, the starting and ending points of the remaining ridges can be used for approximating left and right borders, respectively. The remaining ridges are shown in Figure 2(c). Most of these remaining ridges are present over the actual content area of the page.

Another major problem in using these remaining ridges/text-lines for left and right borders approximation is that, their starting and ending positions are not aligned with respect to each other. We presented a ridges alignment method in [2] for solving this problem, which is described here as follows. For each ridge, the neighboring top and bottom ridges are projected over it, and then are combined together to produce a new (aligned) ridge. The aligned ridges are shown in Figure 2(d). Some more results of ridges after alignment for document images in DFKI-I dataset [9] are shown in Figure 3.

2.3 Page Frame Detection

The left and right borders are calculated by applying a straight-line approximation algorithm over the starting and ending points of the ridges, respectively. For this purpose, we have chosen RANdom SAmple Consensus (RANSAC) method, which approximates slope and intercept parameters. The left and right borders are shown in Figure 4(a) in blue color. The initial estimation of upper and lower borders are done by selecting the top and bottom most ridges within the left and right borders, respectively. The upper and lower borders are also in Figure 4(b) in red color, where the lower border is correct, but upper border is incorrect with respect to the non-text content area of the page.

The initial page frame possesses only non-text elements which lie between text-lines and misses others, as shown in Figure 4(b). The page frame is improved by dragging the upper and/or lower borders according to the non-text elements such that: i) if the top most pixel of non-text elements is above the top most



(a) smoothed image

(b) detected ridges

(c) cleaned/filtered ridges

(d) aligned ridges

Fig. 2. Text-Line Detection: (a) the smoothed image is generated using Gaussian filter bank smoothing, (b) ridges are detected from the smoothed image; most of the ridges represent text-lines, (c) small ridges and ridges near corners are removed using heuristically applied rules, (d) ridges have been aligned (with respect to their staring and ending positions) by projecting neighboring ridges over each of them

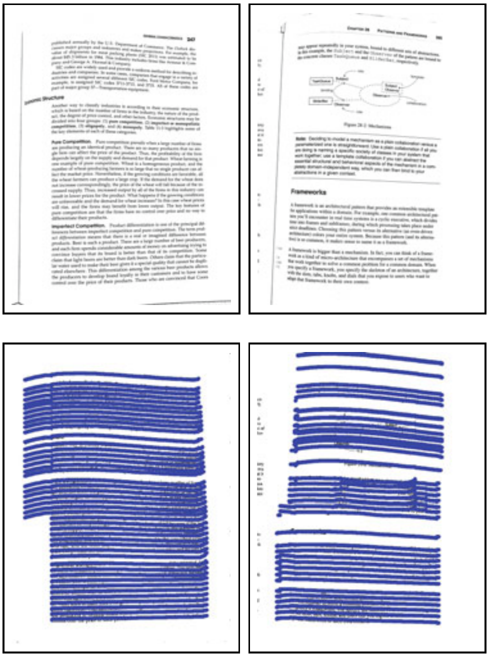
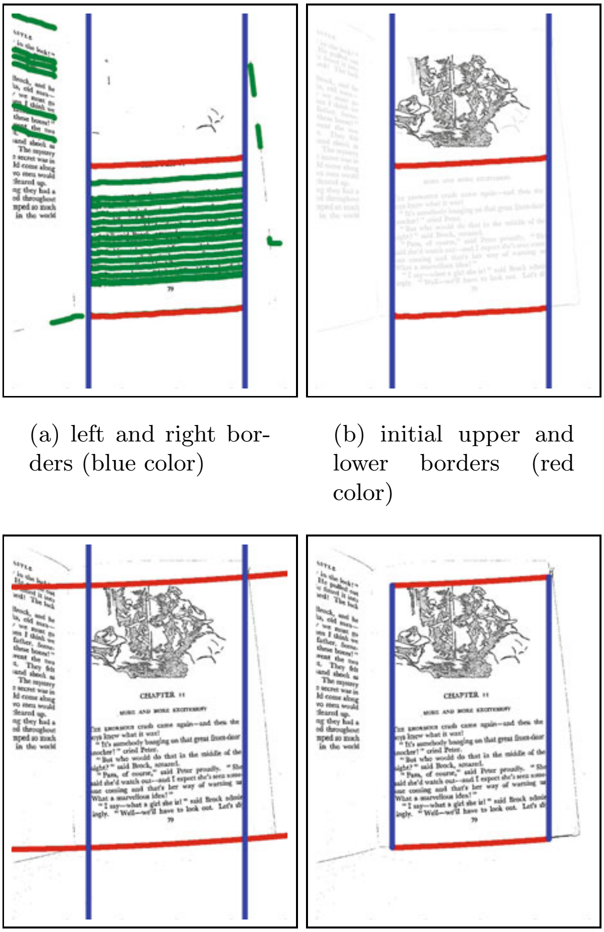


Fig. 3. Sample results of aligned ridges for documents in DFKI-I dataset

pixel of the upper border, the upper border is dragged up to the top most pixel of non-text element, ii) similarly, if the bottom most pixel of non-text elements is below the bottom most pixel of the lower border, the lower border is dragged up to the bottom most pixel of non-text element. Finally, the upper and lower borders are extended across the document width using polynomial fitting. The upper and lower borders after dragging and extending are shown in Figure 4(c). The final page frame is shown in Figure 4(d). Some of example results of the presented page frame detection method on DFKI-I dataset are also shown in Figure 5.

Generally, the starting points of some of the text-lines in a document image coincide with the document's left border line and similarly the ending points of some of the text-lines coincide with the right border line. The ridge alignment step helps in propagating this information to neighboring text-lines. Therefore, left and right borders estimation using starting and ending points of text-lines gives correct results In a special case where a document image contains only short or centered text-lines with non-text elements spanning throughout the page width, the left and right borders can not estimate the actual page contents area. In order to solve this problem, the left and/or right borders can also be dragged with respect to non-text elements, same as it is done in case of upper and/or lower border dragging.



(a) left and right borders (blue color)

(b) initial upper and lower borders (red color)

(c) dragged and extended upper and lower borders

(d) page frame

Fig. 4. Page Frame Detection: (a) left and right borders (blue colors) are detected using starting and ending points of ridges (green color), (b) the top most and the bottom most ridges inside vertical borders are selected as upper and lower borders; non-text parts (black color), that were deleted in preprocessing, are pasted back into the document image, (c) the upper and lower borders are dragged up to the top most pixel and bottom most pixel of the non-text elements, and finally both of them are extended upg to the page width, (d) page frame

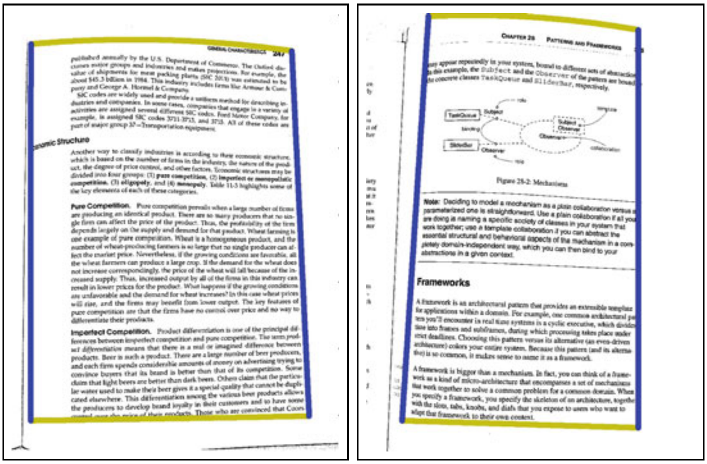


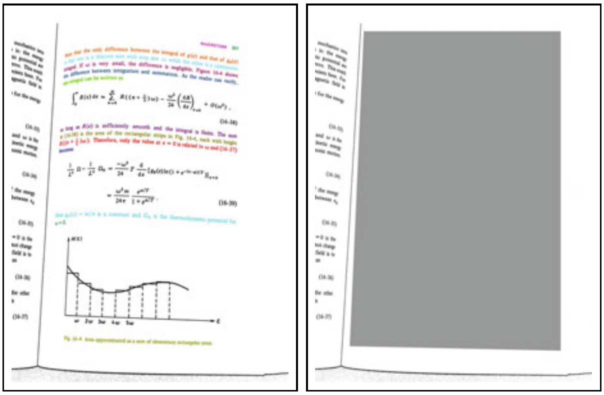
Fig. 5. Sample results of our page frame detection method for DFKI-I dataset

3 Experiments and Results

We have compared our page frame detection method with state-of-the-art methods [13,15] by evaluating them on publicly available DFKI-I (CBDAR 2007 dewarping contest) dataset [9]. We have conducted two different experiments for performance evaluation: i) text-line based evaluation, ii) pixel based evaluation.

DFKI-I dataset contains 102 grayscale and binarized document images of pages from several technical books captured using an off-the-shelf hand-held digital camera in a normal office environment. Document images in this dataset consist of warped text-lines with a high degree of curl, different directions of curl within an image, non-text (graphics, halftone, etc.) components, and a lot of textual and non-textual border noise. Together with ASCII-text ground-truth, this dataset also contains pixel based ground-truth for zones, text-lines, formulas, tables and figures. For text-line based performance evaluation method, text-line based ground-truth images are generated from the original ground-truth images. A text-lines based ground-truth image contains labeling only for text-lines and all the other foreground objects, like formulas, tables and figures, are marked as noise with black color. For pixel based performance evaluation method, ground-truth images are generated by masking the actual page contents only. An example image and its corresponding text-lines and pixel based ground-truth images are shown in Figure 6.

In document images, text-lines are the main source of information from optical character recognition (OCR) point of view. For each text-line in a text-line based ground-truth image, the pixel-correspondence (P) is defined as the ratio of the number of overlapping pixels between the ground-truth image and the correspoding cleaned image and total number of pixels of a particular text-line. Text-line based performance evaluation metrics using pixel-correspondence (P)



(a) labeled text-lines (b) masked page contents

Fig. 6. Text-line based and page contents based ground-truth images for an example image in DFKI-I (CBDAR 2007 dewarping contest) dataset [9]

Table 1. Text-Line Based Performance Evaluation of our page frame detection method and state-of-the-art methods [13,15] on DFKI-I dataset. The results of Shafait et. al method [13] have been copied from their paper. TI: totally-in text-lines; TO: totally-out text-lines; PI: partially-in text-lines. (Note: total number of document images = 102; total number of text-lines = 3097.)

Method	**TI**	**PI**	**TO**
Shafait et. al [13]	95.6%	2.3%	2.1%
Stamatopoulos et. al [15]	96.48%	0.71%	2.81%
our method	**98.10%**	**1.13%**	**0.78%**

were defined in [9]. Here, we use the same metrics. These metrics are defined as follows: i) TI: totally-in text-line ($P \geq 90\%$), ii) TO: totally-out text-line ($P = 0\%$), and iii) PI: partially-in text-line ($P < 90\%$). These metrics measures the percentage of totally-in, partially-in, and totally-out text-lines within page contents with respect to the page frame. The text-line based performance evaluation of the proposed page frame detection method, Shafait et. al [13], and Stamatopoulos et. al [15] page frame detection methods are shown in Table 1. The results show that our method outperforms other two methods mentioned above.

Tex-line based performance metrics only measure the performance of a page frame detection method for text within actual page content area. They report nothing about the performance of a page frame detection method for marginal noise as well as non-text elements within page content area. Furthermore,

Table 2. Pixel Based Performance Evaluation of our page frame detection method and state-of-the-art method [15] on DFKI-I dataset. 'Page Contents' represents the percentage of page contents inside detected page frame. 'Marginal Noise' represent the percentage pf noise outside detected page frame. (Note: total number of page contents pixels = 48188808 (88.52%); total number marginal noise pixels = 6247054 (11.48%)).

method	**Page Contents**	**Noise**
Stamatopoulos et. al [15]	99.11%	36.04%
our method	98.96%	74.81%

text-line based performance evaluation is not a useful measure for the case where the boundary of a complete document image, which contains both textual and non-textual noise, is marked as the page frame. In such a case, text-line based performance evaluation reports 100% totally-in text-lines with no partial-in or totally-out text-lines. Therefore, text-line based performance evaluation alone is not enough for comparing the performance of different page frame detection algorithms. In order to measure how well a page frame detection method works with respect to both marginal noise and actual page contents, a pixel based performance evaluation is used. Our pixel based performance evaluation method measures the pixel-correspondence (P) for both actual page contents and marginal noise between a ground-truth image and the correspoding cleaned image. Pixel correspondence for page content is defined as the ratio of the number of overlapping pixels between the page contents of ground-truth image and the correspoding cleaned image and total number of page contents pixels in ground-truth image. Likewise, the pixel correspondence is defined for marginal noise. The pixel based performance evaluation results of our proposed method and Stamatopoulos et. al [15] page frame detection method are shown in Table 2. It shows that both methods give good performance for actual page contents, but our method performs better for marginal noise cleanup.

4 Discussion

In this paper, we have presented a page frame detection method for warped camera-captured document images. Our method uses text-lines and non-text contents information for detecting page frame (left, right, upper, and lower borders). We have developed a ridge based text-line finding method [3] and a multiresolution based text/non-text segmentation method [4], which we have used here for detecting text-lines and non-text elements, respectively. For the performance evaluation of the presented method and its comparison with state-of-the-art methods, two different methodologies, text-line based and pixel based, have been used. For both performance evaluation methodologies, the presented method has achieved better results than Shafait et. al [13] and Stamatopoulos et. al [15] page frame detection methods, as shown in Table 1 and Table 2.

References

1. Ávila, B.T., Lins, R.D.: Efficient Removal of Noisy Borders from Monochromatic Documents. In: Campilho, A.C., Kamel, M.S. (eds.) ICIAR 2004,Part II. LNCS, vol. 3212, pp. 249–256. Springer, Heidelberg (2004)
2. Bukhari, S.S., Shafait, F., Breuel, T.M.: Dewarping of document images using coupled-snakes. In: Proceedings of Third International Workshop on Camera-Based Document Analysis and Recognition, Barcelona, Spain, pp. 34–41 (2009)
3. Bukhari, S.S., Shafait, F., Breuel, T.M.: Ridges Based Curled Textline Region Detection from Grayscale Camera-Captured Document Images. In: Jiang, X., Petkov, N. (eds.) CAIP 2009. LNCS, vol. 5702, pp. 173–180. Springer, Heidelberg (2009)
4. Bukhari, S.S., Shafait, F., Breuel, T.M.: Improved document image segmentation algorithm using multiresolution morphology. In: Proc. SPIE Document Recognition and Retrieval XVIII, San Jose, CA, USA (January 2011)
5. Cinque, L., Levialdi, S., Lombardi, L., Tanimoto, S.: Segmentation of page images having artifacts of photocopying and scanning. Pattern Recognition 35(5), 1167–1177 (2002)
6. Fan, H., Zhu, L., Tang, Y.: Skew detection in document images based on rectangular active contour. International Journal on Document Analysis and Recognition 13(4), 261–269 (2010)
7. Fan, K.C., Wang, Y.K., Lay, T.R.: Marginal noise removal of document images. Pattern Recognition 35(11), 2593–2611 (2002)
8. Le, D.X., Thoma, G.R., Wechsler, H.: Automated borders detection and adaptive segmentation for binary document images. In: 13th Int. Conf. on Pattern Recognition, Vienna, Austria, pp. 737–741 (August 1996)
9. Shafait, F., Breuel, T.M.: Document image dewarping contest. In: 2nd Int. Workshop on Camera-Based Document Analysis and Recognition, Curitiba, Brazil, pp. 181–188 (September 2007)
10. Shafait, F., Breuel, T.M.: A simple and effective approach for border noise removal from document images. In: 13th IEEE Int. Multi-topic Conference, Islamabad, Pakistan (December 2009)
11. Shafait, F., Breuel, T.M.: The effect of border noise on the performance of projection based page segmentation methods. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(4), 846–851 (2011)
12. Shafait, F., Keysers, D., Breuel, T.M.: Efficient implementation of local adaptive thresholding techniques using integral images. In: Proc. SPIE Document Recognition and Retrieval XV, San Jose, CA, USA, pp. 101–106 (January 2008)
13. Shafait, F., van Beusekom, J., Keysers, D., Breuel, T.: Document cleanup using page frame detection. International Journal on Document Analysis and Recognition 11, 81–96 (2008)
14. Stamatopoulos, N., Gatos, B., Georgiou, T.: Page frame detection for double page document images. In: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, Boston, MA, USA, pp. 401–408 (2010)
15. Stamatopoulos, N., Gatos, B., Kesidis, A.: Automatic borders detection of camera document images. In: Proceedings of Second International Workshop on Camera-Based Document Analysis and Recognition, Curitiba, Brazil, pp. 71–78 (2007)
16. Ulges, A., Lampert, C., Breuel, T.: Document image dewarping using robust estimation of curled text lines. In: Proc. Eighth Int. Conf. on Document Analysis and Recognition, pp. 1001–1005 (August 2005)

An Image Based Performance Evaluation Method for Page Dewarping Algorithms Using SIFT Features

Syed Saqib Bukhari[1], Faisal Shafait[2], and Thomas M. Breuel[1]

[1] Technical University of Kaiserslautern, Germany
{bukhari,tmb}@informatik.uni-kl.de
[2] German Research Center for Artificial Intelligence (DFKI),
Kaiserslautern, Germany
faisal.shafait@dfki.de

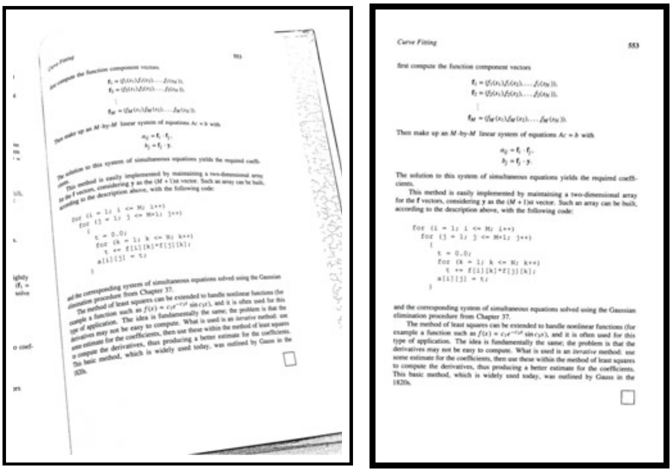
Abstract. Dewarping of camera-captured document images is one the important preprocessing steps before feeding them to a document analysis system. Over the last few years, many approaches have been proposed for document image dewarping. Usually optical character recognition (OCR) based and/or feature based approaches are used for the evaluation of dewarping algorithms. OCR based evaluation is a good measure for the performance of a dewarping method on text regions, but it does not measure how well the dewarping algorithm works on the non-text regions like mathematical equations, graphics, or tables. Feature based evaluation methods, on the other hand, do not have this problem, however, they have following limitations: i) a lot of manual assistance is required for ground-truth generation, and ii) evaluation metrics are not sufficient to get meaningful information about dewarping quality. In this paper, we present an image based methodology for the performance evaluation of dewarping algorithms using SIFT features. For ground-truths, our method only requires scanned images of pages which have been captured by a camera. This paper introduces a vectorial performance evaluation score which gives comprehensive information for determining the performance of different dewarping methods. We have tested our performance evaluation methodology on the participating methods of CBDAR 2007 document image dewarping contest and illustrated the correctness of our method.

Keywords: Performance Evaluation, Dewarping, Camera-Captured Document Images, SIFT.

1 Introduction

The goal of page dewarping is to flatten a camera-captured document such that it becomes readable by current OCR systems. Page dewarping has triggered a lot of interest in the scientific community over the last few years and many

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 138–149, 2012.



Curve Fitting 553

first compute the function component vectors

$$\mathbf{f}_1 = (f_1(x_1), f_1(x_2), \ldots, f_1(x_N)),$$
$$\mathbf{f}_2 = (f_2(x_1), f_2(x_2), \ldots, f_2(x_N)),$$
$$\vdots$$
$$\mathbf{f}_M = (f_M(x_1), f_M(x_2), \ldots, f_M(x_N)).$$

Then make up an M-by-M linear system of equations $Ac = b$ with

$$a_{ij} = \mathbf{f}_i \cdot \mathbf{f}_j,$$
$$b_j = \mathbf{f}_j \cdot \mathbf{y}.$$

The solution to this system of simultaneous equations yields the required coefficients.

This method is easily implemented by maintaining a two-dimensional array for the **f** vectors, considering **y** as the $(M + 1)$st vector. Such an array can be built, according to the description above, with the following code:

```
for (i = 1; i <= M; i++)
  for (j = 1; j <= M+1; j++)
    {
      t = 0.0;
      for (k = 1; k <= N; k++)
        t += f[i][k]*f[j][k];
      a[i][j] = t;
    }
```

and the corresponding system of simultaneous equations solved using the Gaussian elimination procedure from Chapter 37.

The method of least squares can be extended to handle nonlinear functions (for example a function such as $f(x) = c_1 e^{-c_2 x} \sin c_3 x$), and it is often used for this type of application. The idea is fundamentally the same; the problem is that the derivatives may not be easy to compute. What is used is an *iterative* method: use some estimate for the coefficients, then use these within the method of least squares to compute the derivatives, thus producing a better estimate for the coefficients. This basic method, which is widely used today, was outlined by Gauss in the 1820s.

□

(a) camera-captured document

(b) scanned document (ground-truth dewarped image)

Fig. 1. A sample camera-captured document and its correspoding scanned image from DFKI-I dataset. The scanned images in DFKI-I dataset are used here as ground-truth dewarped images.

approaches have been proposed. These dewarping approaches can be broadly divide into two main categories: i) 3-D document shape reconstruction [4,1,13] and ii) 2-D image processing (monocular dewarping) [5,15,14,8,7,3].

Despite a large number of dewarping techniques, performance evaluation of page dewarping methods is still an unsolved problem. Most of the time it has been done on the basis of visual quality of dewarped images [7,2], but it is a subjective evaluation and gives no quantitative measure. In order to objectively compare dewarping methods, OCR based [11,3] and feature based [12] performance evaluation methods have been proposed. OCR based performance evaluation is an indirect method which can only measure the performance of a dewarping method on text regions. Nowadays commercial OCR software can handle degradations in documents to some extend, therefore, OCR based evaluation can not measure how well text elements have been dewarped with respect to their shapes. On the other hand, feature based performance evaluation do not have these problems and can measure the performance of a dewarping method for both text and non-text regions. However, existing feature based performance evaluation methods have following limitations: i) a cumbersome manual marking is required for generating ground-truth data, and ii) a single performance evaluation metric is used which may not be sufficient to compare the performance of different dewarping methods.

In this paper, we propose an image based performance evaluation methodology for dewarping methods to overcome the limitations of the existing feature based performance evaluation methods. We use scanned images of pages, that were captured by camera, as ground-truth dewarped images. In this way, no manual efforts are required for generating ground-truth data for a publicly available dataset that contains scanned documents (like DFKI-I [11]), or a very less manual efforts are required for creating a new dataset. For measuring the performance, instead of a single performance evaluation metric, we present a vectorial score that is particularly useful in analyzing the behavior of different page dewarping algorithms. On the basis of SIFT features matching between a dewarped image and its correspoding ground-truth dewarped image, we calculate the percentage and the mean error of matching features.

The rest of the paper is organized as follows. We describe the proposed image based performance evaluation in Sections 2. Experiments and results are discussed in Section 3. Section 4 presents our conclusions.

2 Image Based Performance Evaluation

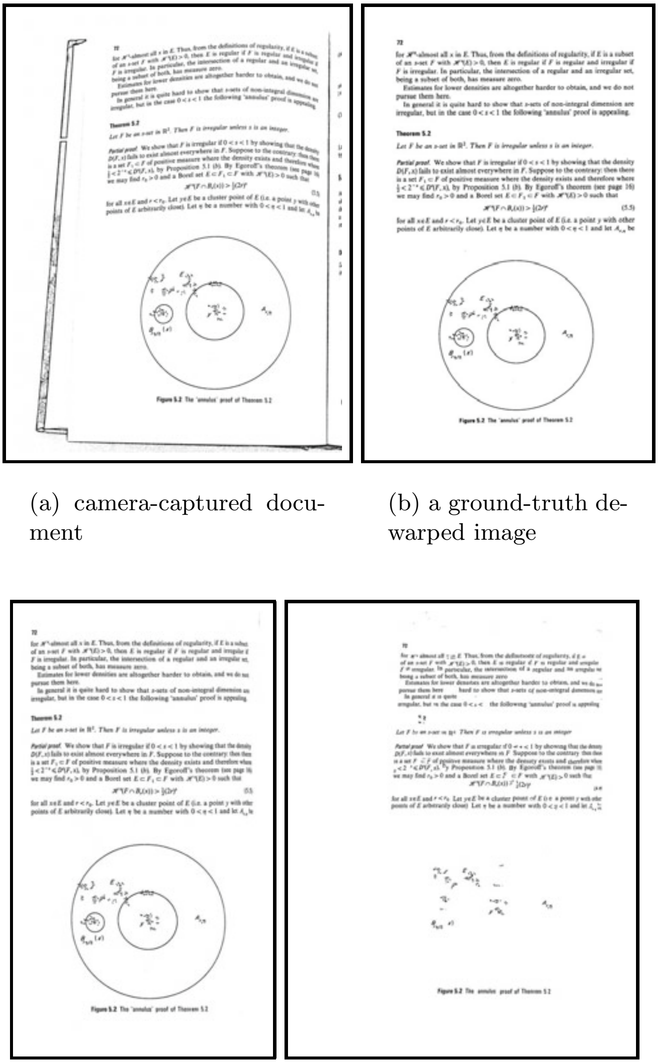
The proposed performance evaluation metrics are described here in detail along with the requirement of ground-truth dewarped images. This section is organized as follows. In Section 2.1, we discuss about the ground-truth dewarped images. The performance evaluation metrics using SIFT based matches are explained in Section 2.2.

2.1 Ground-Truth Dewarped Images

The presented image based performance evaluation method requires ground-truth dewarped images. So far, DFKI-I [11] is the only publicly available dataset of camera-captured document images. We prepared this dataset to compare different page dewarping approaches in a Document Image Dewarping Contest that was held at CBDAR 2007 [11]. The following types of ground-truth were provided with the dataset: i) ground-truth ASCII text in plain text format, ii) ground-truth page segments (text-lines and zones and their types) in color coded form, iii) scanned images of pages which have been captured by a camera. A sample camera-captured document and its correspoding scanned image from the dataset are shown in Figure 1. The scanned document images in this dataset, as shown in Figure 1, are flat and straight. Therefore, they can be used as ground-truth dewarped images. For the purpose of performance evaluation, scanning of pages together with capturing them through camera requires very less manual effort as compared to marking images manually [12] or to generate ASCII text ground-truth [11].

2.2 Performance Evaluation Methodology

To compare the quality of a dewarped document against a ground-truth dewarped document, image based features are calculated using SIFT [9]. For an



72

for $\mathcal{H}^s$-almost all x in E. Thus, from the definitions of regularity, if E is a subset of an s-set F with $\mathcal{H}^s(E) > 0$, then E is regular if F is regular and irregular if F is irregular. In particular, the intersection of a regular and an irregular set, being a subset of both, has measure zero.

Estimates for lower densities are altogether harder to obtain, and we do not pursue them here.

In general it is quite hard to show that s-sets of non-integral dimension are irregular, but in the case $0 < s < 1$ the following 'annulus' proof is appealing.

Theorem 5.2

Let F be an s-set in $\mathbb{R}^1$. Then F is irregular unless s is an integer.

Partial proof. We show that F is irregular if $0 < s < 1$ by showing that the density $D(F, x)$ fails to exist almost everywhere in F. Suppose to the contrary: then there is a set $F_1 \subset F$ of positive measure where the density exists and therefore where $\frac{1}{2} < 2^{-s} \leqslant D^s(F, x)$, by Proposition 5.1 (b). By Egoroff's theorem (see page 16) we may find $r_0 > 0$ and a Borel set $E \subset F_1 \subset F$ with $\mathcal{H}^s(E) > 0$ such that

$$\mathcal{H}^s(F \cap B_r(x)) > \tfrac{1}{2}(2r)^s \qquad (5.5)$$

for all $x \in E$ and $r < r_0$. Let $y \in E$ be a cluster point of E (i.e. a point y with other points of E arbitrarily close). Let η be a number with $0 < \eta < 1$ and let $A_{r,\eta}$ be

Figure 5.2 The 'annulus' proof of Theorem 5.2

(a) camera-captured document

(b) a ground-truth dewarped image

(c) a good dewarped output

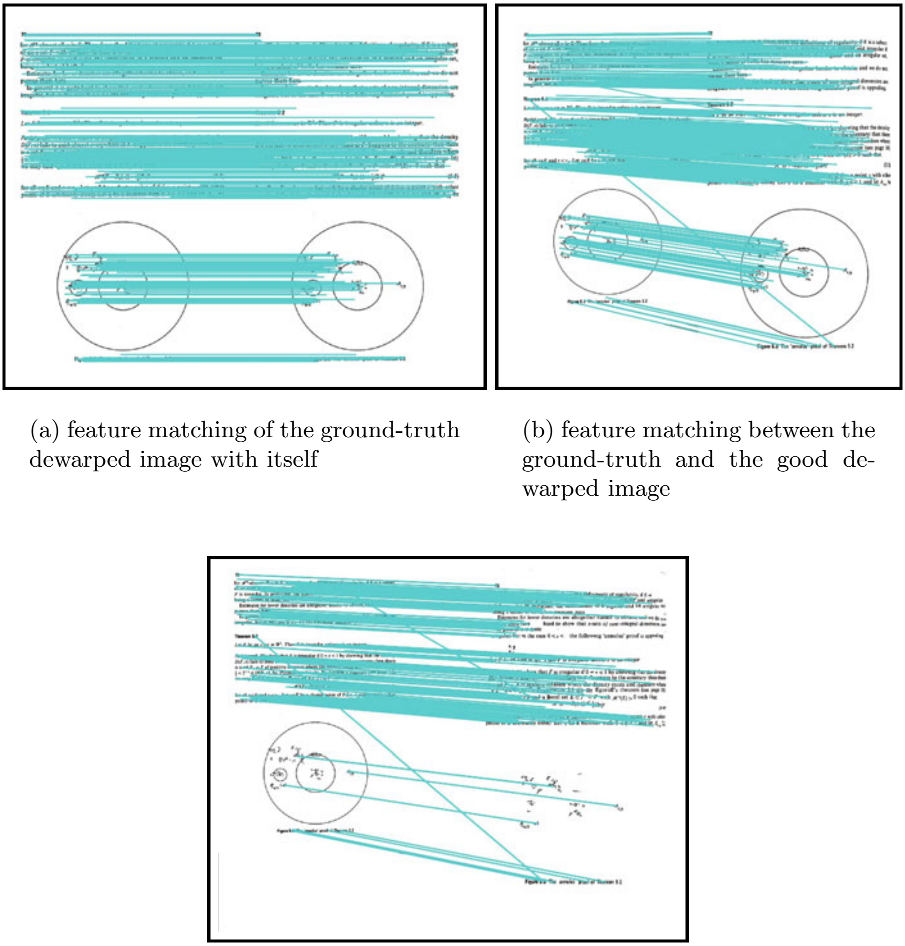
(d) a relatively bad dewarped output

Fig. 2. A sample camera-captured document image and its corresponding ground-truth dewarped image and a good and a bad dewarped images

image, SIFT estimates key features and returns their correspoding locations and descriptors. Matching between the features of two different images is done by calculating cosine inverse of the dot product of their normalized descriptors. The bad matches are removed by applying a thresholding criteria such as, a match is considered bad if the angle ratio between first and second nearest neighbors is greater than a predefined threshold. In our case, we set this threshold equal to 0.6. We have also noticed that there are some wrong SIFT based matches between two similar document images at high image resolutions, but not at low image resolutions. Therefore, we downscale document images by the factor of 4 before SIFT based comparision.

A sample camera-captured, warped document image and its corresponding ground-truth dewarped image are shown in Figure 2 and Figure 2, respectively. For the camera-captured image (Figure 2), two different, a good one and a bad one, dewarped images are also shown in Figure 2 and Figure 2, respectively. Here, it can be noticed that the good dewarped image visually looks similar to the ground-truth image and contains both text and non-text elements, except slight non-linearity in text-lines and different aspect ratio. The bad dewarped image, on the other hand, missed most of the non-text elements and some of the text elements along with irregularity/non-linearity in text-lines. The SIFT based matching between: i) the ground-truth image with itself is shown in Figure 2.2, ii) the ground-truth image and the good dewarped image is shown in Figure 2.2, and iii) the ground-truth image and the bad dewarped is shown in Figure 2.2. The ground-truth image matches perfectly with itself as shown in the Figure 2.2. Most of the matches in Figure 2.2 and Figure 2.2 are correct with respect to the corresponding descriptors and their locations, and some of them are only correct with respect to the corresponding descriptors, but not with the correspoding locations. In order to remove these types of wrong matches, a filtering criteria is used, according to which, all those matches that have distances greater than $T\%$ of document diagonal are removed. The value of T can be set in-between 0% to 100%, where $T = 0\%$ means that the matched descriptors should be at the perfectly same locations otherwise discarded, and $T = 100\%$ means that the locations of matched descriptors can be far apart. Both of these extreme values are not suitable for our case. The reasonable value can be set in-between 10% to 30%. It is also important to note that, the number of matches between the ground-truth image and the good dewarped image are more than the number of matches between the the ground-truth image and the bad dewarped image. Therefore, the number of matches and other related metrics can be used for the performance evaluation of page dewarping methods, which are discussed below.

Consider that we are given two dewarped images, the dewarped image I, and the ground-truth dewarped image G. Let, L_I and D_I represent the locations and normalized descriptors of SIFT features for the dewarped image I, and L_g and D_g represent SIFT features for the ground-truth dewarped image G. If the dewarped image I agrees perfectly with the ground-truth dewarped image G, there will be a perfect matching between their correspoding SIFT features as



(a) feature matching of the ground-truth dewarped image with itself

(b) feature matching between the ground-truth and the good dewarped image

(c) feature matching between the ground-truth and the bad dewarped image

Fig. 3. The matching between SIFT features of: a) the ground-truth image (Figure 2) with itself, b) the ground-truth image and the good dewarped image (Figure 2), c) the ground-truth image and the bad dewarped image (Figure 2)

shown in Figure 2.2. If there are differences between the two dewarped images, then there will not be a perfect matching as shown in Figure 2.2 and Figure 2.2.

Here, we define two different performance measures to evaluate different aspects of the behavior of a page dewarping algorithm using SIFT based feature matching. These measures are defined as follows:

1. **Matching Percentage M_p:** let total number of matches between G and I is represented by N, and total number of features in G is represented by N_G. The matching percentage (M_p) is defined as:

$$M_p = \frac{N}{N_G} \tag{1}$$

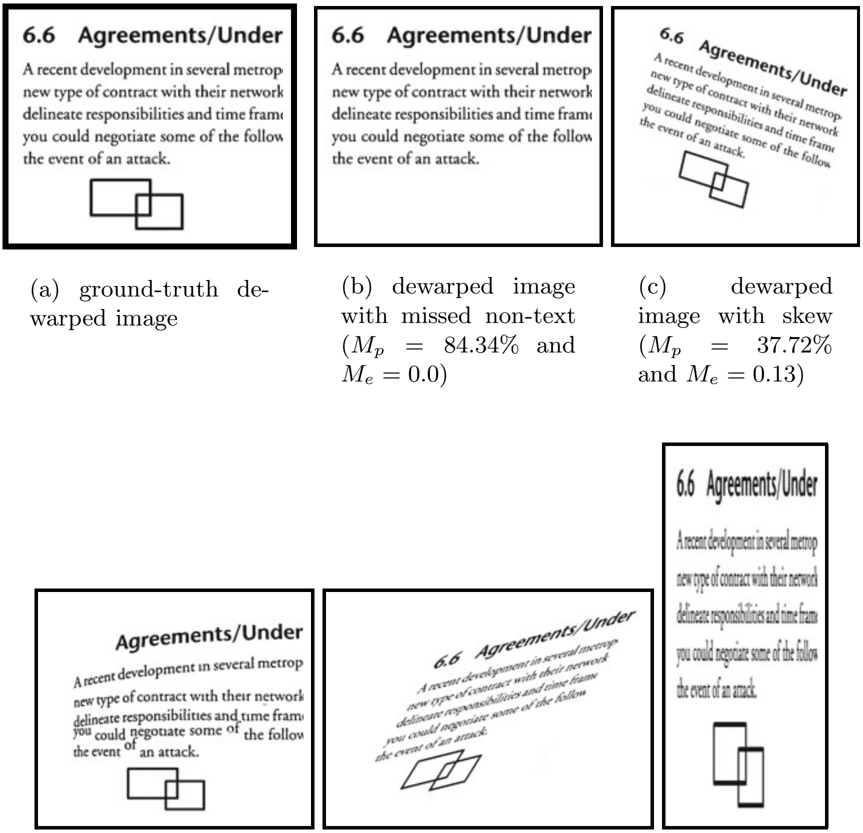
2. **Matching Error M_e:** for a pair of matched descriptors p, let $D_G(p)$ represents a descriptor in G, and $D_I(p)$ represent a corresponding matched descriptor in I. The mean error of all matching pairs is calculated as follows:

$$M_e = \frac{\sum_{p=1}^{N} \arccos(D_G(p) \cdot D_I(p))}{N} \tag{2}$$

We can analyze the effectiveness and correctness of the presented metrics by comparing a ground-truth dewarped image with a good and a bad dewarped images, such an example is shown in the Figure 2. For the good dewarped image (Figure 2), the values of these metrics are as follows: $M_p = 44.57\%$ and $M_e = 0.15$. Similarly, these values for the bad dewarped image (Figure 2) are as follows: $M_p = 11.73\%$ and $M_e = 0.19$. As shown in the Figure 2, the qualities of the good and the bad dewarped images are consistent with their correspoding values of matching percentage (M_p) and matching error (M_e).

The proposed metrics are also effective in terms of indicating typical errors produced by dewarping methods such as i) missed non-text parts as shown in Figure 2.2 where $M_p = 84.34\%$ and $E_m = 0.0$, ii) global skew as shown in Figure 2.2 where $M_p = 37.72\%$ and $M_e = 0.13$, iii) warped, missed, and irregular text as shown in Figure 2.2 where $M_p = 14.59\%$ and $M_e = 0.19$, iv) perspective distortion as shown in Figure 2.2 where $M_p = 0\%$, and v) incorrect aspect ratio as shown in Figure 2.2 where $M_p = 0\%$. The main purpose of dewarping is to transform warped, non-planar documents into planar images so that traditional scanner based OCR softwares can also process them equally like scanned documents. These results are consistent with the visual (planar) quality of dewarped images as well as with respect to OCR accuracy.

In order to analyze some additional visual quality aspects of a dewarping method that do not directly influence OCR accuracy, we can estimate standard deviation of matching locations between a ground-truth image and its corresponding dewarped image. For example, the standard deviations of plain, skewed and irregular document images as shown in Figure 4 with respect to the ground-truth image are equal to 0, 8, and 4.65, respectively. It is important to note that, the skewed image (Figure 2.2) has bigger standard deviation as compared to the


6.6 Agreements/Under
A recent development in several metrop
new type of contract with their network
delineate responsibilities and time fram
you could negotiate some of the follow
the event of an attack.

(a) ground-truth dewarped image

(b) dewarped image with missed non-text ($M_p = 84.34\%$ and $M_e = 0.0$)

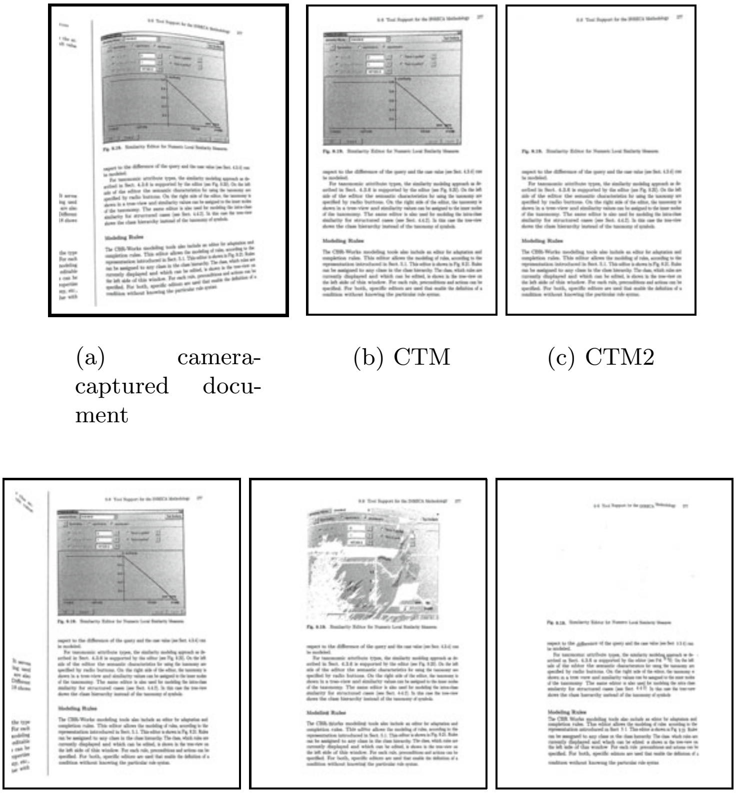
(c) dewarped image with skew ($M_p = 37.72\%$ and $M_e = 0.13$)

(d) dewarped image with warped, missed and irregular text ($M_p = 14.59\%$ and $M_e = 0.19$)

(e) dewarped image with perspective distortions ($M_p = 0\%$)

(f) dewarped image with incorrect aspect ratio ($M_p = 0\%$)

Fig. 4. Behavior of the proposed performance evaluation metrics (matching percentage (M_p) and matching error (M_e) in the presence of typical errors produced by dewarping methods. A dewarped image with warped text, perspectively distorted text, and/or incorrect aspect ratio can be considered as the much more erroneous than missed non-text or global skew with respect to OCR performance.



(a) camera-captured document

(b) CTM

(c) CTM2

(d) SKEL

(e) SNAKE

(f) SEG

Fig. 5. Example results of different methods for a sample camera-captured document of DFKI-I dataset: b) CTM [6], c) CTM2 [6], d) SKEL [10], e) SNAKE [3], f) SEG [7]

irregular text (Figure 2.2), but the skewed image may produce less number of OCR errors than the irregular text, mainly because a skew correction step is a part of standard OCR pipeline.

3 Experiment and Results

As a first step towards comparative evaluation of page dewarping techniques, a page dewarping contest using DFKI-I camera-captured documents dataset was organized along with CBDAR 2007 [11]. Three groups participated in the contest. These three method are referred as CTM [6], SKEL [10], and SEG [7]. The CTM method also used their programs to remove graphics and images from the processed pages. The results thus produced are referred to as CTM2. For the description of the participating methods please refer to [11]. We have also proposed an active contour (snake) based dewarping method in [3], referred to

Table 1. Comparative OCR based error rate (edit distance) of different dewarping methods on DFKI-I dataset

Algorithm	Edit Distance
CTM2 [6]	1.758
SNAKE [3]	1.917
CTM [6]	2.113
SKEL [10]	2.162
SEG [7]	4.088

as SNAKE, and compared its performance with those of contest participants. For a sample camera-captured document image of DFKI-I dataset, the dewarped images of all these methods are shown in Figure 5.

These different methods have been compared with each other through OCR based edit distance by using ASCII text ground-truth in [11,3]. The OCR based performance evaluation results, that are copied from [3], are shown in Table 1. The CTM2 method performs the best on DFKI-I dataset, and its results are better than CTM, i.e. after post-processing to remove graphics and images. This is because the ground-truth ASCII text contains text coming only from the textual parts of the documents, so the text that is present in graphics or images is ignored. Hence, the dewarped documents that contain text inside graphics regions get higher edit distances. On the basis of OCR based performance evaluation, CTM, SKEL and SNAKE have similar performance, and SEG has relatively inferior performance.

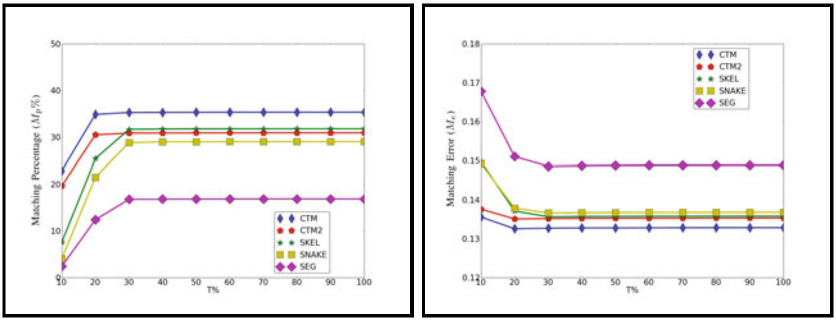
From the methods descriptions, we have determined that both CTM and SKEL handle non-text elements together with text elements, but SEG and SNAKE methods mainly perform dewarping for text elements and do not handle non-text elements. One of such example for DFKI-I dataset can be seen in Figure 5.

In this paper, we compare these dewarping methods using the presented performance evaluation metrics (matching percentage (M_p), matching error (M_e)) on DFKI-I dataset. The feature based performance evaluation results of the dewarping methods for different values of T (10% to 100%) are shown in Figure 6. For an optimal value of T (i.e. $T = 20\%$), feature based performance evaluation results are shown in Table 2. CTM method has achieved the best matching percentage (M_p) among all other methods. The matching percentage and matching error of CTM are better than the CTM2, which is also perfectly consistent with the definition of CTM2 (i.e. removed graphics and images). CTM method has also achieved the lowest matching error (M_e) as compared to other methods. On the other hand, SEG has comparatively achieved the lowest matching percentage and highest matching error in comparison to other methods. It is very interesting to note that these feature-based performance evaluation results are

Table 2. Comparative feature based performance evaluation results of different dewarping methods on DFKI-I dataset using proposed vectorial performance evaluation metrics (matching percentage (M_p) and matching error (M_e))

Algorithm	M_p%	M_e
CTM [6]	34.90%	0.13
CTM2 [6]	30.51%	0.14
SKEL [10]	25.45%	0.14
SNAKE [3]	21.52%	0.14
SEG [7]	12.44%	0.15

also closely consistent with the OCR based results. However, feature based results give more details about the quality of dewarped images with respect to both text and non-text elements.



(a) Matching Percentage (M_p%)

(b) Matching Error (M_e)

Fig. 6. Comparative performance evaluation of different methods for DFKI-I dataset by using the presented feature-based performance evaluation metrics (matching percentage (M_p) and matching error (M_e)) for different values of T

4 Conclusion

In this paper, we have proposed an image based performance evaluation methodology for dewarping algorithms using SIFT features. Unlike OCR based performance evaluation techniques [11,3], a feature based technique indicates how well a dewarping method performs on both text and non-text elements in warped images. Unlike previous feature based performance evaluation techniques [12], our proposed featured based technique does not require manual labeling for generating ground-truth images, and calculate vectorial performance evaluation metrics (matching percentage (M_p) and matching error (M_e)), instead of single score. We have also demonstrated that the feature based performance evaluation results are consistent with the OCR base results.

References

1. Brown, M.S., Seales, W.B.: Image restoration of arbitrarily warped documents. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(10), 1295–1306 (2004)
2. Brown, M.S., Tsoi, Y.C.: Geometric and shading correction for images of printed materials using boundary. IEEE Transactions on Image Processing 15(6), 1544–1554 (2006)
3. Bukhari, S.S., Shafait, F., Breuel, T.M.: Dewarping of document images using coupled-snakes. In: Proceedings of First International Workshop on Camera-Based Document Analysis and Recognition, Barcelona, Spain, pp. 34–41 (2009)
4. Cao, H., Ding, X., Liu, C.: Rectifying the bound document image captured by the camera: a model based approach. In: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), Edinburgh, Scotland, pp. 71–75 (2003)
5. Clark, P., Mirmehdi, M.: Rectifying perspective views of text in 3d scenes using vanishing points. Pattern Recognition 36(11), 2673–2686 (2003)
6. Fu, B., Wu, M., Li, R., Li, W., Xu, Z.: A model-based book dewarping method using text line detection. In: 2nd Int. Workshop on Camera-Based Document Analysis and Recognition, Curitiba, Brazil (September 2007)
7. Gatos, B., Pratikakis, I., Ntirogiannis, K.: Segmentation based recovery of arbitrarily warped document images. In: Proceedings 9th International Conference on Document Analysis and Recognition, Curitiba, Barazil, pp. 989–993 (2007)
8. Liang, J., DeMenthon, D., Doermann, D.: Flattening curved documents in images. In: Proceedings 18th International Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, pp. 338–345 (2005)
9. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60, 91–110 (2004)
10. Masalovitch, A., Mestetskiy, L.: Usage of continuous skeletal image representation for document images de-warping. In: 2nd Int. Workshop on Camera-Based Document Analysis and Recognition, Curitiba, Brazil (September 2007)
11. Shafait, F., Breuel, T.M.: Document image dewarping contest. In: Proceedings 2nd International Workshop on Camera Based Document Analysis and Recognition, Curitiba, Brazil, pp. 181–188 (2007)
12. Stamatopoulos, N., Gatos, B., Pratikakis, I.: A methodology for document image dewarping techniques performance evaluation. In: Proceedings of the 2009 10th International Conference on Document Analysis and Recognition, Barcelona, Spain, pp. 956–960 (2009)
13. Tan, C.L., Zhang, L., Zhang, Z., Xia, T.: Restoring warped document images through 3d shape modeling. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(2), 195–208 (2006)
14. Ulges, A., Lampert, C.H., Breuel, T.M.: Document image dewarping using robust estimation of curled text lines. In: Proceedings 8th International Conference on Document Analysis and Recognition, Seoul, Korea, pp. 1001–1005 (2005)
15. Zhang, Z., Tan, C.L.: Correcting document image warping based on regression of curved text lines. In: Proceedings 7th International Conference on Document Analysis and Recognition, Edinburgh, Scotland, pp. 589–593 (2003)

NEOCR: A Configurable Dataset for Natural Image Text Recognition

Robert Nagy, Anders Dicker, and Klaus Meyer-Wegener

University of Erlangen-Nürnberg
Chair for Computer Science 6 (Data Management)
Martensstr. 3., Erlangen, Germany
{robert.nagy,anders.dicker,klaus.meyer-wegener}@cs.fau.de

Abstract. Recently growing attention has been paid to recognizing text in natural images. Natural image text OCR is far more complex than OCR in scanned documents. Text in real world environments appears in arbitrary colors, font sizes and font types, often affected by perspective distortion, lighting effects, textures or occlusion. Currently there are no datasets publicly available which cover all aspects of natural image OCR. We propose a comprehensive well-annotated configurable dataset for optical character recognition in natural images for the evaluation and comparison of approaches tackling with natural image text OCR. Based on the rich annotations of the proposed NEOCR dataset new and more precise evaluations are now possible, which give more detailed information on where improvements are most required in natural image text OCR.

1 Introduction

Optical character recognition (OCR) for machine-printed documents and handwriting has a long history in computer science. For clean documents, current state-of-the-art methods achieve over 99% character recognition rates [23].

With the prevalence of digital cameras and mobile phones, an ever-growing amount of digital images are created. Many of these natural images contain text. The recognition of text in natural images opens a field of widespread applications, such as:

- help for visually impaired or blind [16] (e.g., reading text not transcribed in braille),
- mobile applications (e.g., translating photographed text for tourists and foreigners [5, 11, 13, 22]),
- object classification (e.g., multimodal fusion of text and visual information [26]),
- image annotation (e.g., for web search [17]),
- vision-based navigation and driving assistant systems [25].

Recently growing attention has been paid to recognizing text in real world images, also referred to as natural image text OCR [22] or scene text recognition [23].

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 150–163, 2012.

Table 1. Typical characteristics of OCR on scanned documents and natural image text recognition

Criteria	Scanned documents	Natural image text
background	homogeneous, usually white or light paper	any color, even dark or textured
blurredness	sharp (depending on scanner)	possibly motion blur, blur because of depth of field
camera position	fixed, document lies on scanner's glass plate	variable, geometric and perspective distortions almost always present
character arrangement	clear horizontal lines	horizontal and vertical lines, rounded, wavy
colors	mostly black text on white background	high variability of colors, also light text on dark background (e.g. illuminated text) or only minor differences between tones
contrast	good (black/dark text on white/light background)	depends on colors, shadows, lighting, illumination, texture
font size	limited number of font sizes	high diversity in font sizes
font type (diversity in document)	usually 1-2 (limited) types of fonts	high diversity of fonts
font type (in general)	machine-print, handwriting	machine-print, handwriting, special (e.g. textured such as light bulbs)
noise	limited / negligible	shadows, lighting, texture, flash light, reflections, objects in the image
number of lines	usually several lines of text	often only one single line or word
occlusion	none	both horizontally, vertically or arbitrary possible
rotation (line arrangement)	horizontally aligned text lines or rotated by ±90 degrees	arbitrary rotations
surface	text "attached" to plain paper	text freestanding (detached) or attached to objects with arbitrary nonplanar surfaces, high variability of distortions

Natural images are far more complex in contrast to machine-printed documents. Problems arise not only from background variations and surrounding objects in the image, but from the depicted text, too, which usually takes on a great variety of appearances. Complementary to the survey of [16], which compared the capturing devices, we summarize main characteristics of scanned document OCR and scene text recognition in table 1.

For the evaluation and comparison of techniques developed specifically for natural image OCR, a publicly available well-annotated dataset is required. All current datasets (see section 3) annotate only the words and bounding boxes in images. Also most text appears in horizontal arrangement, while in natural scenes humans are often confronted with text arranged vertically or circularly (text following a curved, wavy or circular line). Currently there is no well-annotated dataset publicly available that covers all aspects distinguishing scene text recognition from scanned document OCR.

We propose the NEOCR (Natural Environment OCR) dataset consisting of real world images extensively enriched with additional metadata. Based on this metadata several subdatasets can be created to identify and overcome weaknesses of OCR approaches on natural images. Main benefits of the proposed dataset compared to other related datasets are:

- annotation of *all* text visible in images,
- additional distortion quadrangles for a more precise ground truth representation of text regions,
- rich metadata for simple configuration of subdatasets with special characteristics for more detailed identification of shortcomings in OCR approaches.

The paper is organized as follows: In the next section we describe the construction of the new dataset and the annotation metadata in detail. In section 3 a short overview of currently available datasets for OCR in natural images is given and their characteristics are compared to the new NEOCR dataset. We describe new evaluation possibilities due to the rich annotation of the dataset and its future evolution in section 4.

2 Dataset

A comprehensive dataset with rich annotation for OCR in natural images is introduced. The images cover a broad range of characteristics that distinguish real world scenes from scanned documents. Example images from the dataset are shown in figure 1.

The dataset contains a total of 659 images with 5238 bounding boxes (text occurences, hereinafter referred to as “textfields”). Images were captured by the authors and members of the lab using various digital cameras with diverse camera settings to achieve a natural variation of image characteristics. Afterwards images containing text were hand-selected with particular attention to achieving a high diversity in depicted text regions. This first release of the NEOCR dataset covers

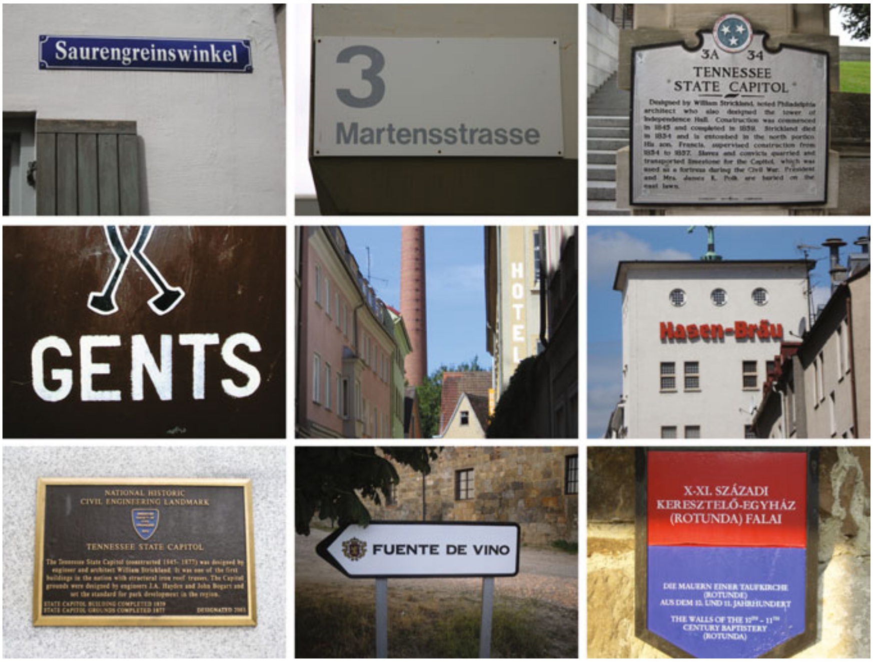
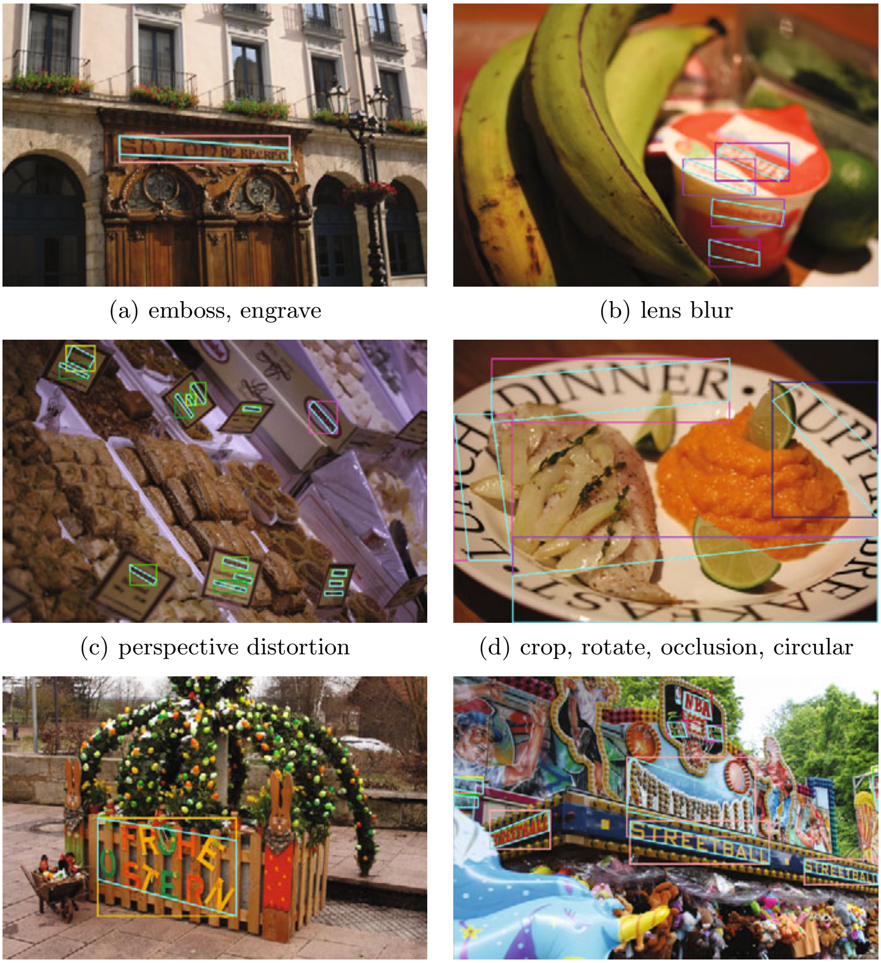

Saurengreinswinkel
3
Martensstrasse
3A 34
TENNESSEE STATE CAPITOL
GENTS
HOTEL
Hasen-Bräu
NATIONAL HISTORIC CIVIL ENGINEERING LANDMARK
TENNESSEE STATE CAPITOL
FUENTE DE VINO
X-XI. SZÁZADI KERESZTELŐ-EGYHÁZ (ROTUNDA) FALAI
DIE MAUERN EINER TAUFKIRCHE (ROTUNDE) AUS DEM 10. UND 11. JAHRHUNDERT
THE WALLS OF THE 10th - 11th CENTURY BAPTISTERY (ROTUNDA)

Fig. 1. Example images from the NEOCR dataset. Note that the dataset also includes images with text in different languages, text with vertical character arrangement, light text on dark and dark text on light background, occlusion, normal and poor contrast

the following dimensions each by at least 100 textfields. Figure 2 shows examples from the NEOCR dataset for typical problems in natural image OCR.

Based on the rich annotation of optical, geometrical and typographical characteristics of bounding boxes, the NEOCR dataset can be tailored into specific datasets to test new approaches for specialized scenarios. Additionally to bounding boxes, distortion quadrangles were added for a more accurate ground truth annotation of text regions and automatic derivation of rotation, scaling, translation and shearing values. These distortion quadrangles also enable a more precise representation of slanted text areas which are close to each other.

For image annotation, the web-based tool of [21] for the LabelMe dataset [6] was used. Due to the simple browser interface of LabelMe the NEOCR dataset can be extended continuously. Annotations are provided in XML for each image separately describing global image features, bounding boxes of text and its special characteristics. The XML-schema of LabelMe has been adapted and extended by tags for additional metadata. The annotation metadata is discussed in more detail in the following sections.



(a) emboss, engrave

(b) lens blur

(c) perspective distortion

(d) crop, rotate, occlusion, circular

(e) textured background

(f) textured text

Fig. 2. Example images from the NEOCR dataset depicting typical characteristics of natural image text recognition

2.1 Global Image Metadata

General image metadata contains the filename, folder, source information and image properties. For each whole image its width, height, depth, brightness and contrast are annotated. Brightness values are obtained by extracting the luma channel (Y-channel) of the images and computing the mean value. The standard deviation of the luma channel is annotated as the contrast value. Both brightness and contrast values are obtained automatically using ImageMagick [4].

2.2 Textfield Metadata

All words and coherent text passages appearing in the images of the NEOCR dataset are marked by bounding boxes. Coherent text passages are one or more lines of text in the same font size and type, color, texture and background (as they usually appear on memorial plaques or signs). All bounding boxes are rectangular and parallel to the axes. Additionally annotated distortion quadrangles inside the bounding boxes give a more accurate representation of text regions. The metadata is enriched by optical, geometrical and typographical characteristics.

Optical Characteristics. Optical characteristics contain information about the blurredness, brightness, contrast, inversion (dark text on light or light text on dark background), noise and texture of a bounding box.

Texture. Texture is very hard to measure automatically, because texture differences can form the text and text itself can be texture, too. Following three categories have been defined:

- low: single color text with single color background,
- mid: multi-colored text or multi-colored background,
- high: multi-colored text and multi-colored background, or text without a continuous surface (e.g., luminous advertising built from light bulbs).

Brightness and contrast. Brightness and contrast values for bounding boxes are obtained the same way as for the whole image (see section 2.1). As an attribute of the contrast characteristic we additionally annotate whether the dark text is represented on light background or vice versa (inverted).

Resolution. In contrast to 1000dpi and more in high resolution scanners, images taken by digital cameras achieve resolutions only up to 300dpi. The lower the focal length, the bigger the area captured by the lens. Depending on the pixel density and the size of the camera sensor small text can get unrecognizable. As a measure we define text resolution as the number of pixels in the bounding box divided by the number of characters.

Noise. Image noise can originate from the noise sensitivity of camera sensors or from image compression artifacts (e.g., in JPEG images). Usually, the higher the ISO values or the higher the compression rates, the bigger the noise in images. Because noise and texture are difficult to distinguish, we classify the bounding boxes into low, mid and high noise judged by eye.

Blurredness. Image blur can be divided into lens and motion blur. Lens blur can result from depth of field effects when using large aperture depending on the focal length and focus point. Similar blurring effects can also result from image compression. Motion blur can originate either from moving objects in the scene or camera shakes by the photographer. [15] gives an overview on different approaches for measuring image blur. As a measure for blurredness we annotated kurtosis to the bounding boxes. First edges are detected using a Laplacian-of-Gaussian filter. Afterwards the edge image is Fourier transformed and the steepness (kurtosis) of the spectral analysis is computed. The higher the kurtosis, the more blurred the image region.

Geometrical Characteristics. Character arrangement, distortion, occlusion and rotation are subsumed under geometrical characteristics.

Distortion. Because the camera sensor plane is almost never parallel to the photographed text's plane, text in natural images usually appears perspectively distorted. Several methods can be applied to represent distortion. In our annotations we used 8 floating point values as described in [24]. The 8 values can be represented as a matrix, where s_x and s_y describe scaling, r_x and r_y rotation, t_x and t_y translation, and p_x and p_y shearing:

$$\begin{pmatrix} s_x & r_y & t_x \\ r_x & s_y & t_y \\ p_x & p_y & 1 \end{pmatrix} \tag{1}$$

The equations in [24] are defined for unit length bounding boxes. We adapted the equations for arbitrary sized bounding boxes. Based on the matrix and the original coordinates of the bounding box, the coordinates of the distorted quadrangle can be computed using the following two equations:

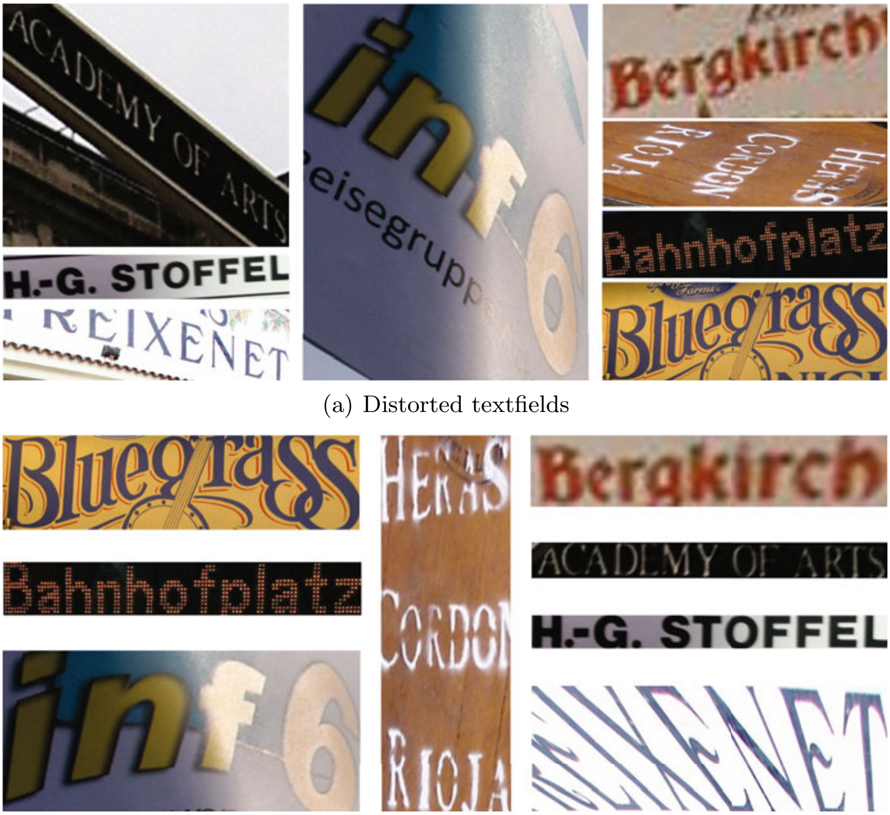
$$x' = \frac{s_x x + r_y y + t_x}{p_x x + p_y y + 1} \tag{2}$$

$$y' = \frac{r_x x + s_y y + t_y}{p_x x + p_y y + 1} \tag{3}$$

Figure 3(a) shows example bounding boxes from the NEOCR dataset with perspective distortion. In figure 3(b) the according straightened textfields are depicted based on the annotated distortion quadrangles. Problems with straightening distorted textfields arise for images with low resolution, strings not completely contained in their bounding boxes and texts with circular character arrangement. Overall, the resulting straightened textfields are largely satisfying.

Rotation. Because of arbitrary camera directions and free positioning in the real world, text can appear diversely rotated in natural images. The rotation values are given in degrees as the offset measured from the horizontal axis given by the image itself. The text rotation angle is computed automatically based on the distortion parameters.

Arrangement. In natural images characters of a text can be arranged vertically, too (e.g., some hotel signs). Also some text can follow curved baselines. In the annotations we distinguish between horizontally, vertically and circularly arranged text. Single characters were classified as horizontally arranged.



(a) Distorted textfields

(b) Straightened textfields

Fig. 3. Examples of textfields with perspective distortion and their straightened versions. Note that while bounding boxes often overlap and include therefore characters from other words, our annotated distortion quadrangles are more exact. Additionally, the quadrangles enable evaluations comparing the performance of text recognition on distorted and straight text.

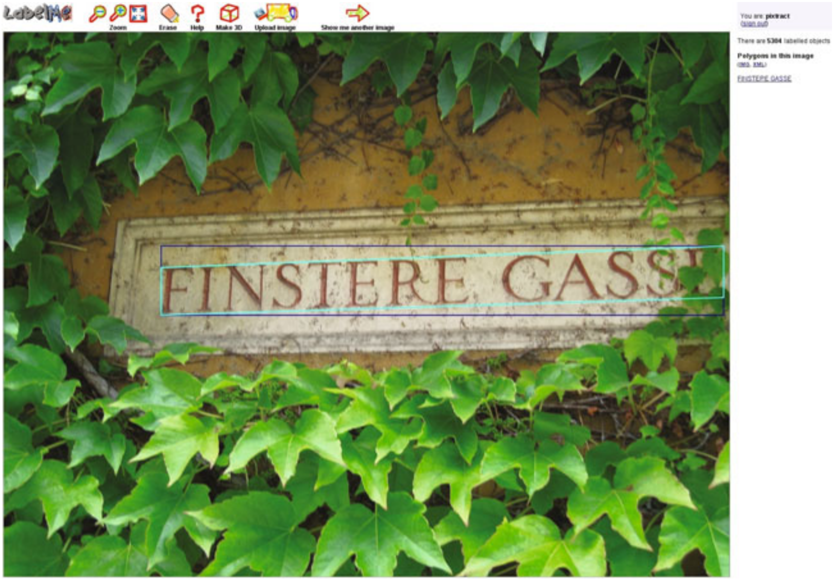
Occlusion. Depending on the chosen image detail by the photographer or objects present in the image, text can appear occluded in natural images. Because missing characters (vertical cover) and horizontal occlusion need to be treated separately, we distinguish between both in our annotations. Also the amount of cover is annotated as percentage value.

Typographical Characteristics. Typographical characteristics contain information about font type and language.

Typefaces. Typefaces of bounding boxes are classified into categories print, handwriting and special. The annotated text is case-sensitive, the font size can be derived from the resolution and bounding box size information. Font thickness is not annotated.

Language. Languages can be a very important information when using vocabularies for correcting errors in recognized text. Because the images were taken in several countries, 15 different languages are present in the NEOCR dataset, though the visible text is limited to latin characters. In some cases, text cannot be clearly assigned to any language. For these special cases we introduced categories for numbers, abbreviations and business names.

Difficulty. The attribute “difficult” was already included in the XML schema of the LabelMe annotation tool, where it is used for marking hardly recognizable objects. In the NEOCR dataset bounding boxes marked as difficult are texts which are illegible without knowing their context due to extreme distortion, occlusion or noise. Overall 190 of the 5238 bounding boxes are tagged as difficult in the dataset, which can be omitted for training and testing (similarly to the PASCAL Challenges [10]).


You are: **pixtract**
(sign out)

There are **5304** labelled objects

Polygons in this image
(IMG, XML)

FINSTERE GASSE

Fig. 4. Example image from the NEOCR dataset. The annotated metadata is shown in table 2.

2.3 Summary

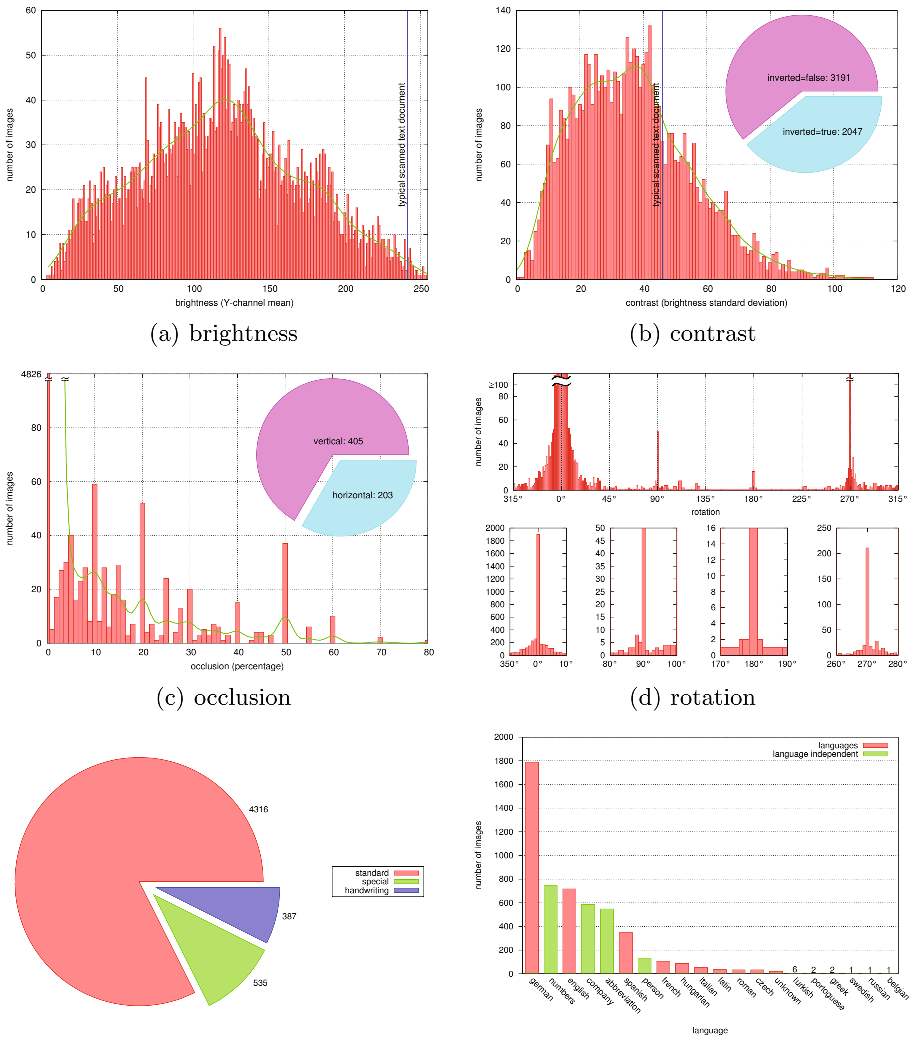
Figure 4 shows a screenshot of the adapted LabelMe annotation tool with an example image. The corresponding annotation for the example image and the range of values for each metadata dimension are listed in table 2.

Further details for the annotations, the XML-schema and the dataset itself can be found in the technical report [20] and on the NEOCR dataset website [8]. Some OCR algorithms rely on training data. For these approaches a disjoint split of the images in training and testing data is provided on the NEOCR dataset website. Both training and testing datasets contain approximately the same number of textfields for each metadata dimension.

Table 2. Range of values for each metadata dimension and annotations for the example image depicted in figure 4

Category	Datatype	Values range	Example value
texture	string	low, mid, high	mid
brightness	float	[0;255]	164.493
contrast	float	[0;123]	36.6992
inversion	boolean	true, false	false
resolution	float	[1;1000000]	49810
noise	string	low, mid, high	low
blurredness	float	[1;100000]	231.787
distortion	8 float values	sx: [-1;5], sy: [-1;1.5], rx: [-15;22], ry: [-23;4], tx: [0;1505], ty: [0;1419], px: [-0.03;0.07], py: [-0.02;0.02]	sx: 0.92, sy:0.67, rx: -0.04, ry: 0, tx: 0, ty: 92, px:-3.28-05, py: 0
rotation	float	[0;360]	2.00934289847729
character arrangement	string	horizontal, vertical, circular	horizontal
occlusion	integer	[0;100]	5
occlusion direction	string	horizontal, vertical	vertical
typeface	string	standard, special, handwriting	standard
language	string	german, english, spanish, hungarian, italian, latin, french, belgian, russian, turkish, greek, swedish, czech, portoguese, numbers, roman date, abbreviation, company, person, unknown	german
difficult	boolean	true, false	false

Figure 5 shows statistics on selected dimensions for the NEOCR dataset. The graphs prove the high diversity of the images in the dataset. The accurate and rich annotation allows more detailed inspection and comparison of approaches for natural image text OCR.



(a) brightness

(b) contrast

(c) occlusion

(d) rotation

(e) font

(f) language

Fig. 5. Brightness, contrast, rotation, occlusion, font and language statistics proving the diversity of the proposed NEOCR dataset. Graphs 5(a) and 5(b) also show the usual value of a scanned text document taken from a computer science book. The number of images refers to the number of textfields marked by bounding boxes.

3 Related Work

Unfortunately, publicly available OCR datasets for scene text recognition are very scarce. The ICDAR 2003 Robust Reading dataset [3, 18, 19] is the most widely used in the community. The dataset contains 258 training and 251 test images annotated with a total of 2263 bounding boxes and text transcriptions. Bounding boxes are all parallel to the axes of the image, which is insufficient for marking text in natural scene images with their high variations of shapes and orientations. Although the images in the dataset show a considerable diversity in font types, the pictures are mostly focused on the depicted text and the dataset contains largely indoor scenes depicting book covers or closeups of device names. The dataset does not contain any vertically or circularly arranged text at all. The high diversity of natural images, such as shadows, light changes, illumination, character arrangement is not covered in the dataset.

The Chars74K dataset introduced by [1, 12] focuses on the recognition of Latin and Kannada characters in natural images. The dataset contains 1922 images mostly depicting sign boards, hoardings and advertisements from a frontal viewpoint. About 900 images have been annotated with bounding boxes for characters and words, of which only 312 images contain latin word annotations. Unfortunately, images with occlusion, low resolution or noise have been excluded and not all words visible in the images have been annotated.

[22] proposed the Street View Text dataset [9], which is based on images harvested from Google Street View [2]. The dataset contains 350 outdoor images depicting mostly business signs. A total of 904 rectangular textfields are annotated. Unfortunately, bounding boxes are parallel to the axes, which is insufficient for marking text variations in natural scenes. Another deficit is that not all words depicted in the image have been annotated.

In [14] a new stroke width based method was introduced for text recognition in natural scenes. The algorithm was evaluated using the ICDAR 2003 dataset and additionally on a newly proposed dataset (MS Text DB [7]). The 307 annotated images cover the characteristics of natural images more comprehensively than the ICDAR dataset. Unfortunately, not all text visible in the images has been annotated and the bounding boxes are parallel to the axes.

Additionally, there also exist some special datasets of license plates, book covers or digits. Still sorely missed is a well-annotated dataset covering the aspects of natural images comprehensively, which could be applied for comparing different approaches and identifying gaps in natural image OCR.

Ground truth annotations in the related datasets presented above are limited to bounding box coordinates and text transcriptions. Therefore, our comparison of current datasets in table 3 is limited to statistics on the number of annotated images, the number of annotated textfields (bounding boxes) and the average number of characters per textfield. The Chars74K dataset is a special case, because it contains word annotations and redundantly its characters are also annotated. For this reason, only annotated words with a length larger than 1 and consisting of latin characters or digits only were included in the statistics in table 3.

Compared to other datasets dedicated to natural image OCR the NEOCR dataset contains more annotated bounding boxes. Because not only words, but also phrases have been annotated in the NEOCR dataset, the average text length per bounding box is higher. None of the related datasets has added metadata

Table 3. Comparison of natural image text recognition datasets

Dataset	#images	#boxes	avg. #char/box
ICDAR 2003	509	2263	6.15
Chars74K	312	2112	6.47
MS Text DB	307	1729	10.76
Street View Text	350	904	6.83
NEOCR	**659**	**5238**	**17.62**

information to the annotated bounding boxes. NEOCR surpasses all other natural image OCR datasets with its rich additional metadata, that enables more detailed evaluations and more specific conclusions on weaknesses of OCR approaches.

4 Conclusion

In this paper the NEOCR dataset has been presented for natural image text recognition. Besides the bounding box annotations, the dataset is enriched with additional metadata like rotation, occlusion or inversion. For a more accurate ground truth representation distortion quadrangles have been annotated, too. Due to the rich annotation several subdatasets can be derived from the NEOCR dataset for testing new approaches in different situations. By the use of the dataset, differences among OCR approaches can be emphasized on a more detailed level and deficits can be identified more accurately. Scenarios like comparing the effect of vocabularies (due to the language metadata), the effect of distortion or rotation, character arrangement, contrast or the individual combination of these are now possible by using the NEOCR dataset. In future we plan to increase the number of annotated images by opening access to our adapted version of the LabelMe annotation tool.

References

[1] Chars74K Dataset, http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

[2] Google Street View, http://maps.google.com

[3] ICDAR Robust Reading Dataset, http://algoval.essex.ac.uk/icdar/Datasets.html

[4] ImageMagick, http://www.imagemagick.org

[5] knfbReader, http://www.knfbreader.com

[6] LabelMe Dataset, http://labelme.csail.mit.edu/

[7] Microsoft Text Detection Database, http://research.microsoft.com/en-us/um/people/eyalofek/text_detection_database.zip

[8] NEOCR Dataset, http://www6.cs.fau.de/research/projects/pixtract/neocr

[9] Street View Text Dataset, http://vision.ucsd.edu/~kai/svt/

[10] The PASCAL Visual Object Classes Challenge, http://pascallin.ecs.soton.ac.uk/challenges/VOC/

[11] Word Lens, http://questvisual.com/

[12] de Campos, T.E., Babu, M.R., Varma, M.: Character Recognition in Natural Images. In: International Conference on Computer Vision Theory and Applications (2009)

[13] Chang, L.Z., ZhiYing, S.Z.: Robust Pre-processing Techniques for OCR Applications on Mobile Devices. In: ACM International Conference on Mobile Technology, Application and Systems (2009)

[14] Epshtein, B., Ofek, E., Wexler, Y.: Detecting Text in Natural Scenes with Stroke Width Transform. In: IEEE International Conference on Computer Vision and Pattern Recognition, pp. 2963–2970 (2010)

[15] Ferzli, R., Karam, L.J.: A No-Reference Objective Image Sharpness Metric Based on the Notion of Just Noticeable Blur (JNB). IEEE Transactions on Image Processing 18(4), 717–728 (2009)

[16] Liang, J., Doermann, D., Li, H.: Camera-based Analysis of Text and Documents: A Survey. International Journal on Document Analysis and Recognition 7, 84–104 (2005)

[17] Lopresti, D., Zhou, J.: Locating and Recognizing Text in WWW Images. Information Retrieval 2(2-3), 177–206 (2000)

[18] Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R., Ashida, K., Nagai, H., Okamoto, M., Yamamoto, H., Miyao, H.M., Zhu, J., Ou, W., Wolf, C., Jolion, J.M., Todoran, L., Worring, M., Lin, X.: ICDAR 2003 Robust Reading Competitions: Entries, Results, and Future Directions. International Journal on Document Analysis and Recognition 7(2-3), 105–122 (2005)

[19] Lucas, S., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: ICDAR 2003 Robust Reading Competitions. In: IEEE International Conference on Document Analysis and Recognition, pp. 682–687 (2003)

[20] Nagy, R., Dicker, A., Meyer-Wegener, K.: Definition and Evaluation of the NEOCR Dataset for Natural-Image Text Recognition. Tech. Rep. CS-2011-07, University of Erlangen, Dept. of Computer Science (2011)

[21] Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: LabelMe: A Database and Web-Based Tool for Image Annotation. International Journal of Computer Vision 77, 157–173 (2008)

[22] Wang, K., Belongie, S.: Word Spotting in the Wild. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part I. LNCS, vol. 6311, pp. 591–604. Springer, Heidelberg (2010)

[23] Weinman, J.J., Learned-Miller, E., Hanson, A.R.: Scene Text Recognition Using Similarity and a Lexicon with Sparse Belief Propagation. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(10), 1733–1746 (2009)

[24] Wolberg, G.: Digital Image Warping. IEEE Computer Society Press, Los Alamitos (1994)

[25] Wu, W., Chen, X., Yang, J.: Incremental Detection of Text on Road Signs from Video with Application to a Driving Assistant System. In: ACM International Conference on Multimedia, pp. 852–859. ACM, New York (2004)

[26] Zhu, Q., Yeh, M.C., Cheng, K.T.: Multimodal Fusion using Learned Text Concepts for Image Categorization. In: ACM International Conference on Multimedia, pp. 211–220. ACM, New York (2006)

The IUPR Dataset of Camera-Captured Document Images

Syed Saqib Bukhari[1], Faisal Shafait[2], and Thomas M. Breuel[1]

[1] Technical University of Kaiserslautern, Germany
bukhari@informatik.uni-kl.de, tmb@informatik.uni-kl.de
[2] German Research Center for Artificial Intelligence (DFKI),
Kaiserslautern, Germany
faisal.shafait@dfki.de

Abstract. Major challenges in camera-base document analysis are dealing with uneven shadows, high degree of curl and perspective distortions. In CBDAR 2007, we introduced the first dataset (DFKI-I) of camera-captured document images in conjunction with a page dewarping contest. One of the main limitations of this dataset is that it contains images only from technical books with simple layouts and moderate curl/skew. Moreover, it does not contain information about camera's specifications and settings, imaging environment, and document contents. This kind of information would be more helpful for understanding the results of the experimental evaluation of camera-based document image processing (binarization, page segmentation, dewarping, etc.). In this paper, we introduce a new dataset (the IUPR dataset) of camera-captured document images. As compared to the previous dataset, the new dataset contains images from different varieties of technical and non-technical books with more challenging problems, like different types of layouts, large variety of curl, wide range of perspective distortions, and high to low resolutions. Additionally, the document images in the new dataset are provided with detailed information about thickness of books, imaging environment and camera's viewing angle and its internal settings. The new dataset will help research community to develop robust camera-captured document processing algorithms in order to solve the challenging problems in the dataset and to compare different methods on a common ground.

Keywords: Dataset, Camera-Captured Document Processing, Performance Evaluation.

1 Introduction

Ground-truth datasets are crucial for objectively measuring the performance of algorithms in many fields of computer science. The availability of such datasets for use in research and development lays the basis for comparative evaluation of algorithms. However, collecting a real-world dataset and preparing its ground-truth is not a trivial task. Therefore, a good practice in research is to focus on developing algorithms that solve the problem at hand and use existing public

M. Iwamura and F. Shafait (Eds.): CBDAR 2011, LNCS 7139, pp. 164–171, 2012.

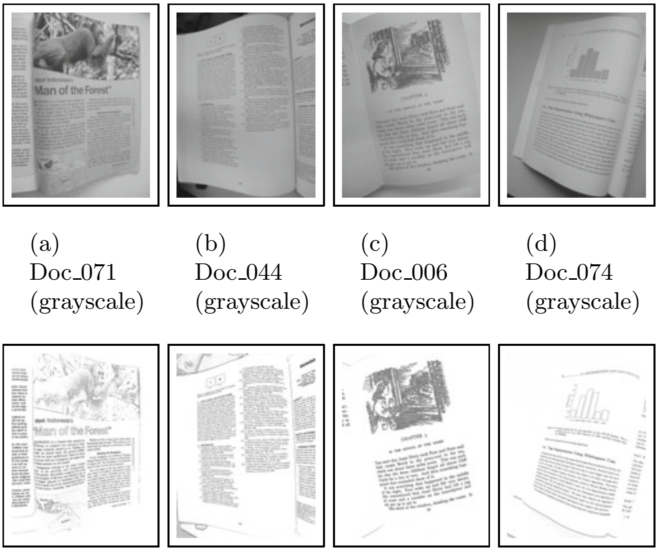
datasets for evaluating the performance of the developed algorithms. In doing so, one not only saves the effort needed to create a representative dataset and its ground-truth, but also the results obtained can be directly compared to those of other algorithms on the same dataset. For instance, in the machine learning community, evaluating new classification algorithms on datasets from the UCI repository [1] has become a de facto standard.

In document analysis and recognition, collecting real-world datasets and sharing them with the community has received quite a lot of attention. As a result, several representative datasets are available for different tasks. Examples of such dataset include the MNIST dataset [2] for handwritten character recognition, the UNLV ISRI dataset [3] for optical character recognition, the UW-I/II/III datasets [10] for document layout analysis, the MARG dataset [9] for logical labeling, the UvA color documents dataset [20] for handling colored magazine pages, the IAM database [12] for off-line handwritten text-line and word segmentation, IFN/ENIT dataset [14] for Arabic handwritten word recognition, and last but not least the Google 1000 books dataset [22] for optical character recognition of old books.

While such rich datasets provide solid grounds for experimentation, all of these datasets focus on scanned documents. With the advent of digital cameras, the traditional way of capturing documents is changing from flat-bed scans to camera captures [19,4]. Recognition of documents captured with hand-held cameras poses many additional technical challenges like perspective distortion, non-planar surfaces, uneven lighting, low resolution, and wide-angle-lens distortions [11]. These challenges have opened new directions of research like binarization and noise removal from camera-captured documents, page segmentation (zone segmentation, curled text-line extraction) and document image dewarping.

We have developed the first camera-captured document image dataset (DFKI-I) [17] for benchmark. Researchers have used this dataset for benchmarking binarization [5,13], noise cleanup using page frame detection [16,8], text-line extraction [7], and dewarping methods [17,6]. All the document images in the DFKI-I dataset belong to simple technical books with single column layout and contain small skew/curl angles. Therefore, there is no variety of the images in the DFKI-I dataset. Additionally, the dataset is not provided with the details of imaging environment, camera (viewing angle, internal settings, resolution, etc.) and document contents, even though such type of information would be more helpful for understanding the experimental evaluation results of camera-based document image processing tasks.

To fill these gaps, we developed a new dataset of camera-captured documents. The new dataset contains documents from a large variety of technical and non-technical books and bound pages, and the details related to imaging environment, camera settings, and document contents are also provided with each document image. Like DFKI-I dataset, we prepared ground-truth information for text-lines, text-zone, and zone-type, dewarped images (scanned documents), and ASCII text for all documents in the new dataset. We refer our new dataset as the IUPR dataset. This paper describes the IUPR dataset in detail and presents



(a) Doc_071 (grayscale)

(b) Doc_044 (grayscale)

(c) Doc_006 (grayscale)

(d) Doc_074 (grayscale)

(e) Doc_071 (binarized)

(f) Doc_044 (binarized)

(g) Doc_006 (binarized)

(h) Doc_074 (binarized)

Fig. 1. Samples of grayscale documents and their binarized images from the IUPR dataset

it as a foundation of comparative performance evaluation for different tasks in the camera-captured document analysis domain.

The rest of this paper is organized as follows. We present the IUPR camera-captured document images dataset in Section 2. The process of generating the ground-truth is illustrated in Section 3. Section 4 represents our conclusion.

2 The IUPR Dataset

The dataset consists of 100 grayscale document images of pages that were captured by using a hand-held camera. The captured documents were binarized using a local adaptive thresholding technique described in [21]. Some sample grayscale documents and their binarized images in the dataset are shown in Figure 1. The details about imaging environment and camera settings that were used for capturing images, and the contents of the dataset are described here.

Imaging Environment:

Documents were captured by placing books on flat table. All documents were captured during daylight in an office-room having a normal white-light on ceiling.

Table 1. Some of the information about camera setting and document content (that are provided with each document in the dataset) for the sample documents in Figure 1

Document ID	Camear Setting		Document Content	
	Viewing Angle	Mega Pixel	Book Type	book Thickness
Doc_071	Left	15	Magazine	1.5 cm
Doc_044	Right	15	Conference Proceedings	2.5 cm
Doc_006	Top-right	9	Old Story Book	2.0 cm
Doc_074	Bottom-Right	15	Bound Pages (Technical)	1.0 cm

Camera Setting:

A cannon PowerShot G10 camera was used for capturing document images. Images were captured by setting the camera to the "macro" mode and without any digital zoom and flash. Documents were captured with a variety of resolutions (5, 9, or 15 Mega Pixel) and different viewing angles of camera (like left, top-right, bottom-left etc.) for adding a verity of perspective distortions in the dataset. The viewing angle can be roughly estimated with respect to the document's center point. Camera settings that were used to capture the sample documents in Figure 1 are shown in Table 1.

Document Content:

Documents have been selected from several different technical books, magazines, old story books, bound pages, etc. These documents belong to a large variety of layouts, some of them can be seen in Figure 1. For the sample document images as shown in Figure 1, the thickness of their correspoding books are mentioned in Table 1. In general, geometric distortion in a document image depends upon book's thickness and its position (folded/unfolded).

Some statistics about the documents in the IUPR dataset are as follow. Out of 100 documents, 75 documents consist of single-column layout and 25 documents contain multi-column layout. 51 documents consist of complete page border (like Figure 1(b)) and remaining 49 documents consists of incomplete page border (like Figure 1(c)). 85 documents were captured from unfolded books (like Figure 1(a)) and remaining 15 documents were captured from folded books (like Figure 1(d)).

The following information is provided with each document image in the dataset.

- name, publisher, and thickness of book

- book type (proceedings, magazine, story, bound pages etc.)
- page number, contents detail (text, graphics, etc.) and number of columns
- folded/unfolded book
- complete/incomplete page border
- camera viewing angle
- camera resolution

3 Ground-Truth

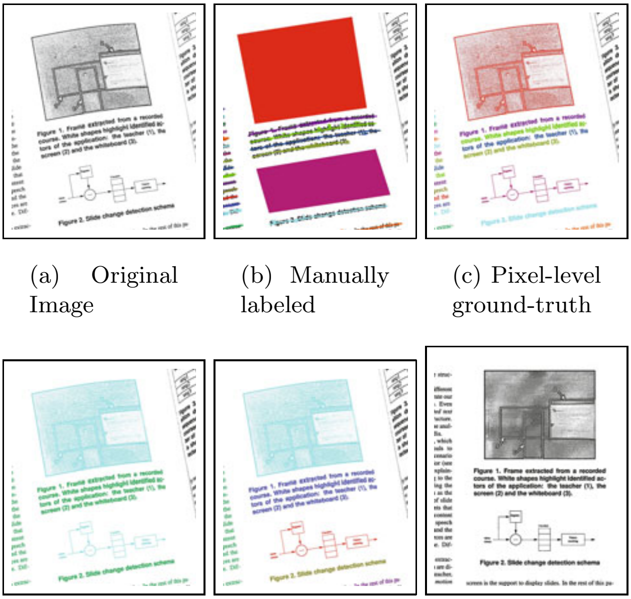
The dataset is provided with different types of ground-truth information as follows:

1. ground-truth text-lines in color coded form (Figure 2(c))
2. ground-truth text-zones in color coded form (Figure 2(e))
3. content type (half-tone/figure, equation, table, text, marginal noise) ground-truth information (Figure 2(d))
4. reading order of text-lines and text-zones
5. ground-truth ASCII text in plain text format
6. ground-truth dewarped (scanned) document images (Figure 2(f))

Generating pixel-level ground-truth can become quite cumbersome since a document image typically contains over one million foreground pixels. Therefore, we have developed semi-automatic technique [18] for preparing pixel-level ground-truth. For each text-lines/figure-captions/formulas, a line is drawn manually with a unique color, and for each table/figure/graphics, a bounding polygon is drawn with a unique color. The manual labeling for a sample image (Figure 2(a)) is shown in Figure 2(b). Manual color labeling is done in such a way that R, G, and B channel contains information about content type, zone number and text-line number in reading order, respectively, where color channel R is set to '1' for mathematical equations, '2' for tables, '3' for figure/graphics, and '4' for text-lines. The R, G, and B color channels of background and marginal noise pixels are all set equal to 255 and 0, respectively.

From manual labeling, the pixel-level ground-truth image of a document is generated as follows. First, connected components are extracted from the document image (Figure 2(a)), and then each connected component is assigned the color of the manually labeled line/polygon that touches with the connected component. The pixel-level ground-truth image is shown in Figure 2(c). In this figure, each text-line as well as non-text element can be uniquely identified. By using the information provided in color channel R and G, content type and zone level ground-truths can also be generated, respectively, which are shown in Figure 2(d) and Figure 2(e), respectively.

All documents that were captured with a camera were also scanned with a flat-bed scanner. These scanned documents are flat and straight as shown in Figure 2(f). Therefore, they can be used as ground-truth dewarped images for image based performance evaluation of dewarping methods [8]. Additionally, ASCII text ground-truth of scanned documents is intended for use as the



(a) Original Image

(b) Manually labeled

(c) Pixel-level ground-truth

(d) Content type level ground-truth

(e) Zone level ground-truth

(f) Dewarped (scanned) image ground-truth

Fig. 2. An example image to demonstrate the process of generating different types of ground-truth that are provided with the IUPR dataset

overall performance measure of a dewarping system by using OCR on the dewarped document. A commercial OCR system was then used to generate the text ground-truth from the scanned documents. The OCR system was used in an interactive mode such that it presented to the operator all characters for which the recognition confidence was not high. We also replaced all mathematical and other non-ASCII symbols with a '#' sybmol as was done in the datasets used in UNLV annual tests of OCR accuracy [15].

The dataset can be downloaded from the following website: **www.sites.google.com/a/iupr.com/bukhari/**. It is not split into training and test sets, because some algorithms need larger training sets as compared to others. It is expected that when other researchers use this dataset, they will split it into test and training sets as per requirements.

4 Conclusion

This paper presented a new camera-captured documents dataset–the IUPR dataset. Unlike the previous DFKI-I dataset [17], the new dataset consists of images from different technical and non-technical books with a diversity of layouts as well as a large variety of perspective and/or geometric distortions. Therefore, the new dataset is much more challenging as compared to the previous DFKI-I dataset. According to the ground-truth information that is provided with the dataset, the new dataset can be used for the performance evaluation and benchmarking of camera-captured document image processing approaches, like binarization, page (text-line/zone) segmentation, zone classification, dewarping, etc. Detailed information about the imaging environment, camera settings, and document contents is also provided with each image in the dataset, which can help in analyzing the performance evaluation results. This dataset makes a good base for comparative evaluation of camera-captured document analysis algorithms.

References

1. http://archive.ics.uci.edu/ml/datasets.html
2. http://yann.lecun.com/exdb/mnist/
3. http://www.isri.unlv.edu/ISRI/OCRtk
4. Breuel, T.: The future of document imaging in the era of electronic documents. In: Int. Workshop on Document Analysis, Kolkata, India (March 2005)
5. Bukhari, S.S., Shafait, F., Breuel, T.M.: Adaptive binarization of unconstrained hand-held camera-captured document images. Journal of Universal Computer Science (J.UCS) 15(18), 3343–3363 (2009)
6. Bukhari, S.S., Shafait, F., Breuel, T.M.: Dewarping of document images using coupled-snakes. In: Proceedings of Third International Workshop on Camera-Based Document Analysis and Recognition, Barcelona, Spain, pp. 34–41 (2009)
7. Bukhari, S.S., Shafait, F., Breuel, T.M.: Performance evaluation of curled textlines segmentation algorithms on CBDAR 2007 dewarping contest dataset. In: Proceedings 17th International Conference on Image Processing, Hong Kong, China (2010)
8. Bukhari, S.S., Shafait, F., Breuel, T.M.: Border noise removal of camera-captured document images using page frame detection. In: Proceedings of Fourth International Workshop on Camera-Based Document Analysis and Recognition, Beijing, China (2011)
9. Ford, G., Thoma, G.R.: Ground truth data for document image analysis. In: Symposium on Document Image Understanding and Technology, Greenbelt, MD, USA, pp. 199–205 (April 2003)
10. Guyon, I., Haralick, R.M., Hull, J.J., Phillips, I.T.: Data sets for OCR and document image understanding research. In: Bunke, H., Wang, P. (eds.) Handbook of Character Recognition and Document Image Analysis, pp. 779–799. World Scientific, Singapore (1997)
11. Liang, J., Doermann, D., Li, H.: Camera-based analysis of text and documents: a survey. Int. Jour. of Document Analysis and Recognition 7(2-3), 84–104 (2005)
12. Marti, U., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. Int. Jour. on Document Analysis and Recognition 5(1), 39–46 (2002)
13. Oliveira, D.M., Lins, R.D.: A new method for shading removal and binarization of documents acquired with portable digital cameras. In: Proceedings of Third International Workshop on Camera-Based Document Analysis and Recognition, Barcelona, Spain, pp. 3–10 (2009)

14. Pechwitz, M., Maddouri, S.S., Maergner, V., Ellouze, N., Amiri, H.: IFN/ENIT-database of handwritten Arabic words. In: 7th Colloque Int. Francophone sur l'Ecrit et le Document, Hammamet, Tunis (October 2002)
15. Rice, S.V., Jenkins, F.R., Nartker, T.A.: The fourth annual test of OCR accuracy. Tech. rep., Information Science Research Institute, University of Nevada, Las Vegas (1995)
16. Shafait, F., van Beusekom, J., Keysers, D., Breuel, T.M.: Document cleanup using page frame detection. Int. Jour. on Document Analysis and Recognition 11(2), 81–96 (2008)
17. Shafait, F., Breuel, T.M.: Document image dewarping contest. In: 2nd Int. Workshop on Camera-Based Document Analysis and Recognition, Curitiba, Brazil, pp. 181–188 (September 2007)
18. Shafait, F., Keysers, D., Breuel, T.M.: Performance evaluation and benchmarking of six page segmentation algorithms. IEEE Trans. on Pattern Analysis and Machine Intelligence 30(6), 941–954 (2008)
19. Taylor, M.J., Zappala, A., Newman, W.M., Dance, C.R.: Documents through cameras. Image and Vision Computing 17(11), 831–844 (1999)
20. Todoran, L., Worring, M., Smeulders, M.: The UvA color document dataset. Int. Jour. on Document Analysis and Recognition 7(4), 228–240 (2005)
21. Ulges, A., Lampert, C., Breuel, T.: Document image dewarping using robust estimation of curled text lines. In: Proc. Eighth Int. Conf. on Document Analysis and Recognition, pp. 1001–1005 (August 2005)
22. Vincent, L.: Google book search: Document understanding on a massive scale. In: 9th Int. Conf. on Document Analysis and Recognition, Curitiba, Brazil, pp. 819–823 (September 2007)

Author Index