A LOW COMPLEXITY METHOD FOR DETECTION OF TEXT AREA IN NATURAL IMAGES

Katherine L. Bouman[†], *Golnaz Abdollahian*[‡], *Mireille Boutin*[‡], *and Edward J. Delp*[‡]

†Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109, USA

‡Video and Image Processing Laboratory (*VIPER*)
School of Electrical and Computer Engineering
Purdue University, West Lafayette, Indiana, USA

ABSTRACT

We propose a low complexity method for segmentation of text regions in natural images. This algorithm is designed for mobile applications (e.g. unmanned or hand-held devices) in which computational and energy resources are limited. No prior assumption is made regarding the text size, font, language, character set or the camera angle. However, the text is assumed to be located on a piecewise homogeneous background with a contrasting color. We have deployed our method on a Nokia N800 Internet tablet as part of a system for automatic detection and translation of outdoor signs. Our experiments show that the 0.3 megapixel images taken by the phone camera can be accurately segmented within the device in a fraction of a second.

Index Terms— Text Detection, Text Segmentation, Sign Detection, Mobile Devices

1. INTRODUCTION

The automatic detection of text within a natural image is an important problem in many applications. Once identified, the text can be analyzed, recognized and interpreted. Text in an image can be in different character sets, languages and fonts. Due to such a large diversity of text characteristics, reliable text features are difficult to find.

Several approaches for automatic detection and translation of text in images and videos have been proposed. Most of these methods aim to detect the characters based on general properties of character pixels. The distribution of edges, for example, is used in many text detection methods [1, 2, 3]. In these methods the edges are grouped together based on features such as size, color and aspect ratio. Texture is another commonly used feature for text segmentation [4, 5]. First, a texture analysis method such as Gabor filtering is used to extract the texture features. Then, a classifier (e.g., support vector machines [4, 5]) is used to classify the regions into text/non-text. The methods that are based on edge or texture features are not robust to skewness due to the camera angle. Therefore, an affine rectification step is often added [2].

Text segmentation methods are often designed for a specific application such as detecting text in videos [6], license plates [7] and signs [2, 8]. Our method focuses on the detection of text on signs. It relies on the properties of the background of the text region, as opposed to the text itself. Therefore, it is independent of language, character set and angle of the sign with respect to the camera.

Our text segmentation method is part of the "Rosetta Phone" [9] system, a handheld device (e.g., PDA or mobile telephone) we are developing that is capable of acquiring a picture of a sign, identifying the text within the image, and producing both an audible and a visual English interpretation of the text. Similar systems achieve this result by having the user select the text [10]. However, automatic identification of the text's location simplifies the system for the user.

We require a computationally simple and energy savvy approach to text segmentation, one that involves a small number of operations and preferably simple operations such as additions and subtractions. Assuming that the text background is (piecewise) homogeneous, our method quickly rules out large non-text regions. As a result, the computational resources can be used in further analysis of the remaining areas.

This work was sponsored by Next Wave Systems, LLC. Address all correspondence to M. Boutin, mboutin@ecn.purdue.edu

2. APPROACH

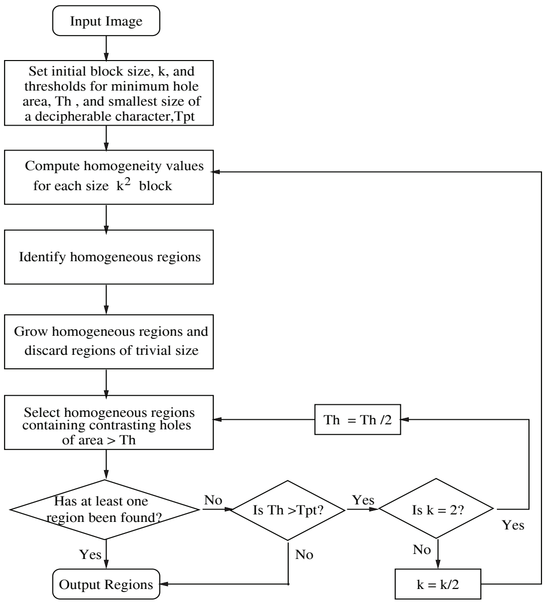

Input Image
Set initial block size, k, and thresholds for minimum hole area, Th , and smallest size of a decipherable character,Tpt
Compute homogeneity values for each size k² block
Identify homogeneous regions
Grow homogeneous regions and discard regions of trivial size
Select homogeneous regions containing contrasting holes of area > Th
Th = Th /2
Has at least one region been found?
No
Is Th >Tpt?
Yes
Is k = 2?
Yes
No
Yes
No
Output Regions
k = k/2

Fig. 1. Schematic representation of proposed method.

A schematic representation of our method is shown in Figure 1. We begin by dividing the image into blocks. Blocks whose luminance is homogeneous are then selected. Homogeneity is measured by using a set of binary filters for each block. The uniform areas are then grown using connected components. Grown regions without a large hole are subsequently discarded; those whose hole does not contain a color contrasting with the background are discarded as well. If no text region is found, the image is divided again into smaller blocks and the above steps are repeated. If no text region is found after the smallest block size is used, a fine search is performed. This final search is designed to find text in a document or any other text area containing small characters. We now describe each step of the procedure in detail.

Background Detection

We begin by locating areas with nearly uniform luminance and texture. To do this, the image is divided into $k \times k$ non-overlapping blocks, where k is a power of two (Fig. 2). The homogeneity of each block is calculated, and blocks that are classified as homogeneous are selected as potential backgrounds. The homogeneity of a block is defined as follows. For each $k \times k$ block, let I be the vector of k^2 luminance values for the block. We compute M homogeneity features for each block as

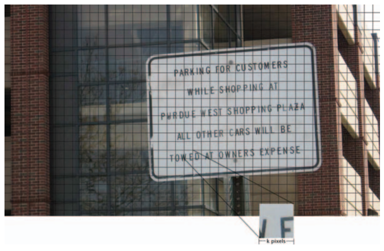


Fig. 2. Input image divided into $k \times k$ pixel blocks

$$\delta^{(m)} = \frac{2}{k^2} \sum_{i=1}^{k^2} I_i w_i^{(m)}$$

where $w^{(m)}$ for $0 \leq m < M$ is a weight vector with binary entries (i.e. each entry is ± 1) that sum to zero. We use $M = 3$ features corresponding to the three binary weight vectors shown in Fig. 3.

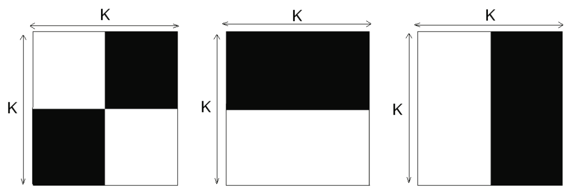


Fig. 3. Graphical representation of the $M = 3$ weight vectors used to quantify homogeneity of the blocks. White and black are used to represent +1 and -1 respectively.

After all $\delta^{(m)}$ values have been computed, we select blocks for which the L_∞ norm of the vector $\delta = [\delta_0, ..., \delta_{M-1}]$ is less than a threshold, $T_u = 1$ (chosen based on human perception of uniformity and training), and classify them as being homogeneous. Homogeneous blocks whose neighboring blocks are all nonhomogeneous are discarded from the list of potential background areas. Smaller block sizes have the advantage of identifying smaller homogeneous areas, while larger blocks are less susceptible to noise. Fig. 4 shows the homogeneous blocks identified for k equal to 64, 32, 16, 8, 4, and 2. Notice that homogenous blocks generally do not contain edges.

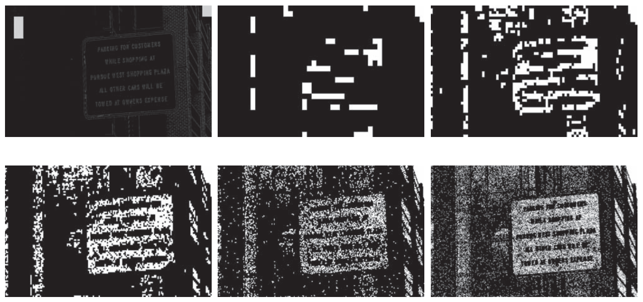


Fig. 4. Homogeneous block selection for the image in Fig. 2 using various block sizes, k= 64, 32, 16, 8, 4, 2 (left to right and top to bottom). Homogeneous regions are shown as white areas.

Background Segmentation

Once homogeneous blocks have been identified, we expand them using a region growing method in order to form connected regions corresponding to potential text backgrounds. Denote the set of 2-D lattice points making up the image as S, and the individual locations of pixels in the image as $s \in S$ where $s = (s_i, s_j)$. Two neighboring pixels are considered connected if the difference between their luminance values is less than a fixed threshold, $T_x = 5$. We considered the four-point neighborhood for each pixel.

To grow the blocks, we begin at any seed point s_0 within the detected uniform blocks and identify s_0's connected set. Once this has been done, we label all the pixels in the connected set, $C(s_0)$, as one region. We then select a new seed point in a uniform block which is not in $C(s_0)$ and determine its connected set. This process is repeated until every pixel contained in the image's uniform blocks maps to a connected set. Fig. 5 shows the potential background regions in an image after the uniform blocks have been grown for a block size of 64, 32, 16, 8, 4, and 2.

False Background Rejection

Once the potential background regions of an image are identified, each region is further tested for the presence of text based on two criteria: 1) it must contain at least one hole, and 2) the hole color must contrast against the background color.

Holes

In order to identify holes we employ connected components again. First, each potential background region is isolated from the rest of the image. In order to do so, we assign all pixels belonging to a given potential background region the value 1 (white), and all the remaining pixels the value 0

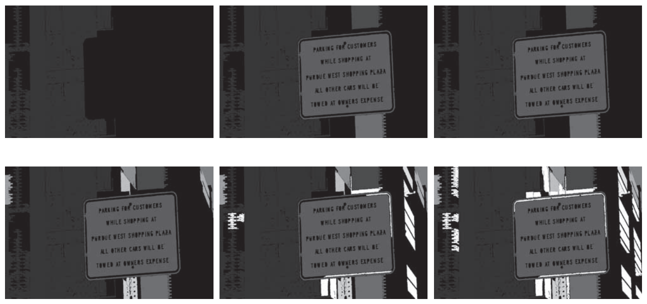


Fig. 5. Segmented potential backgrounds for various block sizes, k= 64, 32, 16, 8, 4, 2 (left to right and top to bottom). Non-black regions are considered homogeneous. Each shade of gray corresponds to one region.

(black). Then, a minimum rectangular bounding box is defined around the region that contains all the white pixels. Two rows and columns of black pixels are added to each side of the sub-image so that the white region is surrounded on all sides by black. Fig. 6 shows how an image containing different regions is broken into sub-images. The bounding box has two purposes: to decrease computation and to assist in the next step of the process.

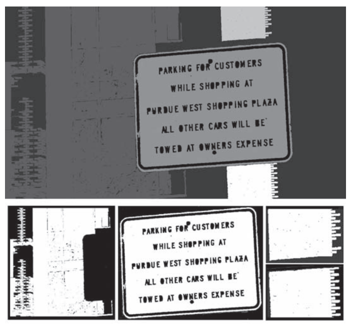


Fig. 6. Isolating the connected regions of an image: connected regions (top) and isolated regions of the image enclosed by exaggerated bounding boxes (bottom)

All of the connected sets are then extracted from the binary sub-image. Since each sub-image is always surrounded on all sides by black pixels, there will always be at least two connected sets: the bounding region and the potential background region. If a region's binary sub-image has more than two connected sets, the potential background region has at least one hole.

Noise or dirt may cause small holes to be enclosed by the potential background region. Therefore, a constraint is placed on the size of a connected set in order for a hole to be considered a text hole. Connected sets that have an area less than or equal to $T_h = \frac{|C(b)|}{60}$ pixels are discarded from the list of connected sets, where $|C(b)|$ is the size of a potential background region. Defining T_h as a function of region size allows small signs to contain smaller text holes, while requiring larger signs to contain larger text holes. Ideally, each character or figure within a sign is defined as a hole.

Color Contrast

In order to be seen easily, text or figures must contrast against the background. Given the background region, B, and a hole region, H,

$$|\bar{x}_B - \bar{x}_H| \geq T_c$$

must hold true, where for region R, $\bar{x}_R$ is the average luminance value of R's pixels, and T_c is a fixed threshold of 60.

Many characters contain holes. However, when determining the average luminance of the text, the luminance of hole pixels within characters or figures should not be included. If the luminance of a pixel contained in the text region is less than two standard deviations from the average luminance of the background region, it is considered to be part of a hole within a character. Figure 7 shows how a character's region is determined after discarding pixels similar to the character's background.

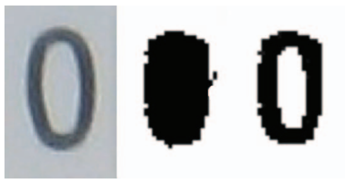


Fig. 7. (Left) Original image. (Middle) Text hole region. (Right) Text hole region after discarding pixels similar to surrounding background.

Text regions that do not meet the contrast criterion are removed from the potential text background area being examined. If there still remains more than 2 connected sets (at least one text hole) in an examined area, the area is considered to contain text or a figure.

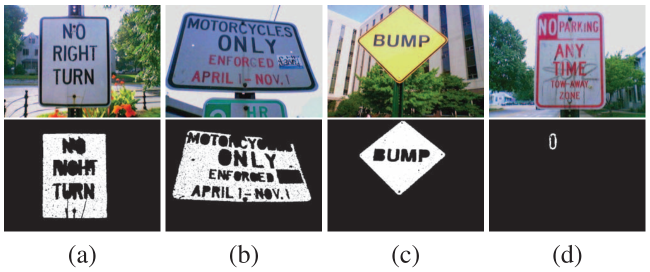
Multi-scale Search Strategy

If no text regions have been identified, the block size k is reduced by half and the process is repeated. If no region has been identified by the end of the procedure for the smallest possible block size, $k = 2$, then the threshold for the minimum size of a hole, T_h, is reduced by half until a region is found or T_h is smaller than the size of a decipherable letter, $T_{pt} = 6$ pixels.

3. TESTING RESULTS

We tested our method on a database of 265 0.3-megapixel images of signs and documents. These images, which contained a large variety of text and figures, were taken using the VGA camera on the Nokia N800. The accuracy of locating signs (239 images) and documents (26 images) within the images were determined separately. Images of road signs, plaques, and posters were classified as signs, while images of business cards and small handwritten documents were categorized as documents. Because the camera used did not have a flash, a majority of the images were taken outside.

Using our method, 97.1% of sign were identified in images. The 7 images of signs that were undetected contained either grainy images or dirty signs. Sixteen regions were misclassified as text background regions, yielding a false positive



NO RIGHT TURN
MOTORCYCLES ONLY ENFORCED APRIL 1 - NOV. 1
BUMP
NO PARKING ANY TIME TOW-AWAY ZONE
(a)
(b)
(c)
(d)

Fig. 8. Sample of the results of text area segmentation on 0.3-megapixel images signs. (a) A complex scene, (b) a dirty sign containing text of different colors and sizes and (c) a non-rectangular sign. White regions are identified as text background and the holes within the background correspond to the detected text areas. An undetected sign is shown in (d).

rate of 6.64% for signs. Document backgrounds were correctly identified in 96.2% of images. Documents had a false positive rate of 3.8%. A sample of our results can be seen in Fig. 8.

The Nokia N800 (CPU 330MHz, 128MB RAM) takes 0.766 seconds on average to identify a sign (standard deviation of 0.678 seconds). A PC (CPU 2.40GHz, 1GB RAM) takes an average of 0.031 seconds (standard deviation of 0.021 seconds) to correctly identify a sign in a 0.3-megapixel image. Images containing text regions that are not correctly identified by the algorithm take a longer time to process (on average 6.53 seconds on the N800, and 0.152 seconds on the PC).

To test the computational complexity of our algorithm, we tested it on a database of 144 images used in the ICDAR 2005 competition on locating text in camera captured scenes [11]. The average time that it took to run the algorithm was then compared to the average times of the competition's submitted algorithms. Our system proved to be very computationally inexpensive, running over six times faster than the fastest system submitted to the ICDAR 2005 competition. Table 1 contains the average time in seconds to process an image for each system on a 2.4 GHz processor.

The energy consumption of our method is also very low. We took several hundred images and processed them on the Nokia N800, and we have not seen any appreciable changes in the battery charge.

System	Time (seconds)
Our System	0.055
Alex Chen	0.35
Qiang Zhu	1.6
Jisoo Kim	2.2
Nobuo Ezaki	2.8
Hinnerk Becker	14.4

Table 1. Average Time to Locate Text in ICDAR images

4. CONCLUSIONS

We have presented a low-complexity method for locating text within natural images. Our proposed method searches for the text's background rather than the actual text. This allows for a large variation in the distribution of text features while requiring little computation. Our test show that our method is robust, segmenting 97.1% of the 256 0.3-megapixel images correctly (both signs and documents), and energy efficient.

5. REFERENCES

[1] Q. Yuan and C. Tan, "Text extraction from gray scale document images using edge information," *Sixth International Conference on Document Analysis and Recognition*, 2001, pp. 302–306.

[2] X. Chen, J. Yang, J. Zhang, and A. Waibel, "Automatic detection and recognition of signs from natural scenes," *IEEE Transactions On Image Processing*, vol. 13, pp. 87–99, 2004.

[3] N. Ezaki, M. Bulacu, and L. Schomaker, "Text detection from natural scene images: towards a system for visually impaired persons," *Proceeding of the 17th International Conference on Pattern Recognition*, vol. 2, 2004, pp. 683–686.

[4] K. Kim, K. Jung, and J. H. Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Transactions On Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1631–1638, 2003.

[5] K. Jung, J. Han, K. Kim, and S. Park, "Support vector machines for text location in news video images," *TENCON*, vol. 2, 2000, pp. 176–180.

[6] R. Lienhart and W. Effelsberg, "Automatic text segmentation and text recognition for video indexing," *Multimedia Systems*, vol. 8, pp. 69–81, 2000.

[7] Y.-T. Cui and Q. Huang, "Character extraction of license plates from video," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 502–507, 1997.

[8] J. Gao and J. Yang, "An adaptive algorithm for text detection from natural scenes," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2001.

[9] S. A. R. Jafri, A. Mikkilineni, M. Boutin, and E. Delp, "The rosetta phone: a hand-held device for automatic translation of signs in natural images," *SPIE*, vol. 6821, 2008.

[10] A. Eisenburg, "What's next; point, shoot and translate into english," *The New York Times*, 2002.

[11] S. Lucas, "ICDAR 2005 text locating competition results," *Proceedings of the 8th International Conference on Document Analysis and Recognition*, vol. 1, 2005, pp. 80–84.