A Low Complexity Sign Detection and Text Localization Method for Mobile Applications

Katherine L. Bouman, *Student Member, IEEE*, Golnaz Abdollahian, *Member, IEEE*, Mireille Boutin, *Member, IEEE*, and Edward J. Delp, *Fellow, IEEE*

***Abstract*—We propose a low complexity method for sign detection and text localization in natural images. This method is designed for mobile applications (e.g., unmanned or handheld devices) in which computational and energy resources are limited. No prior assumption is made regarding the text size, font, language, or character set. However, the text is assumed to be located on a homogeneous background using a contrasting color. We have deployed our method on a Nokia N800 cellular phone as part of a system for automatic detection and translation of outdoor signs. This handheld device is equipped with a 0.3-megapixel camera capable of acquiring images of outdoor signs that typically contain enough details for the sign to be readable by a human viewer. Our experiments show that the text of these images can be accurately localized within the device in a fraction of a second.**

***Index Terms*—Mobile devices, sign detection, text detection, text localization, text segmentation.**

I. Introduction

THE automatic localization of text within a natural image is an important problem in many applications. Once identified, the text can be analyzed, recognized, and interpreted. However, many objects in natural images, such as tree branches or electrical wires, are easily confused for text by existing optical character recognition (OCR) algorithms. For this reason, applying OCR on an unprocessed natural image is computationally expensive and may produce erroneous results. Hence, robust and efficient methods are needed to identify the text-containing regions within natural images before performing OCR.

Several approaches for automatic detection and localization of text in images and videos have been proposed [1]–[6]. These algorithms mainly focus on the features of the text itself, such as edges [7], [8], corners [8], strokes [9], color [10], [11], and texture distribution [12], [13]. Some algorithms also make use of Gabor filters [12], [13], wavelets [14], and support vector machines [12], [13]. However, text in an image can be written in different character sets, languages, and fonts. It can also change size and style within the same region. Due to the large diversity of text characteristics, determining features that can be used to reliably detect text is difficult. Thus, existing methods based on text features tend to be trained to identify text with a limited character set and, typically, a restricted number of fonts.

Text localization methods are often designed for a specific application such as detecting text in videos [11], license plates [4], or signs [6], [15]. Our method focuses on the detection and localization of text on signs. Sign text localization can be useful for many applications including tourism, navigation for the blind, robot guidance, and intelligent transportation systems. In order to be easily seen from the road, most signs contain text printed on a homogenous background. We take advantage of this common feature of signs in searching for text by first finding the text's background region. Since our proposed algorithm relies solely on the properties of a sign's background, it is not dependent on language, character set, skew, or tilt of the sign.

Our sign detection and text localization method is part of the "Rosetta phone" [16] system, a handheld device (e.g., PDA or mobile telephone) we are developing that is capable of acquiring a picture of a sign, detecting the sign, localizing the text within the sign, 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 [17]. However, automatic identification of the text's location simplifies the system for the user.

Since the Rosetta phone is a low-power mobile device, the detection and localization algorithm must be implementable with a small number of relatively simple operations. An advantage of our approach is that it uses a multiscale search technique to quickly rule out large regions of the image that are not homogenous, thereby dramatically reducing computation. Once the sign is detected, our algorithm efficiently locates the text within it by identifying text or figures that contrast against the sign's background.

This paper is organized as follows. Section II presents a survey of what has been done in the area of text detection. Our method of sign detection and text localization is described in detail in Section III. In Section IV, we present the experimental results and the complexity analysis of our method. Finally, our concluding remarks are given in Section V.

II. Previous Work

Many text detection and localization approaches use features related to text character properties. For example, edge detection is a common first step in many such algorithms [15], [18]–[24]. Once detected, the edge regions are often grouped together based on features such as size, color, and aspect ratio [19].

Manuscript received March 31, 2010; revised October 19, 2010 and February 15, 2011; accepted April 14, 2011. Date of publication May 12, 2011; date of current version September 16, 2011. This work was supported by Next Wave Systems, LLC. Part of this work was presented at ICASSP 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Z. Jane Wang.

K. L. Bouman is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI USA (e-mail: klbouman@umich.edu).

G. Abdollahian, M. Boutin, and E. J. Delp are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: gabdollla@ecn.purdue.edu; mboutin@ecn.purdue.edu; ace@ecn.purdue.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2011.2154317

1520-9210/$26.00 © 2011 IEEE

Texture is another widely used feature for text detection and localization [12], [13]. First a texture analysis method such as Gabor filtering is used to extract the texture features. Then, a classifier is used to identify a region as a text or not-text based on these features. Support vector machines are used in [12], [13] for this purpose. In [20], authors use LoG-Gabor filters to obtain the stroke map and apply Harris corner detection to find the seed points for the connected component analysis.

Some approaches assume that the text is written in the horizontal or vertical direction [7], [8]. For example, the method in [8] localizes text on road signs in video based on the assumption that text on road signs occur in a vertical plane with respect to the camera motion and the optical axis of the camera. However, this constraint is not practical due to the possible tilt of the camera. Therefore, an affine rectification step is usually added to improve the results [15].

Text that appears on video frames is an important cue for content analysis and indexing of the video. Thus, several approaches have been proposed for the detection and tracking of text in videos [11], [20], [25], [26]. Some of these methods are used to detect the artificial text added to the video [11], [26], which is based on the fact that this type of text has a homogeneous color as opposed to the complex color distribution of the scene. Other approaches aim at detecting any type of text appearing in the video scene, which captures a more general scenario [20], [25]. Most of the latter approaches can also be applied for text detection in natural images. For example, the method in [25] finds candidate text regions in an image or video frame using Fourier-Laplacian filtering followed by k-means clustering. Then, the skeleton of each connected component is used to segment the candidate text strings. Finally, straightness and edge density criteria is applied to discard the false positives. As we show in Section IV-A3, our method runs much more efficiently compared to [25] when used on still images.

Another method that we compare our approach to is the multiscale-COS/CCC segmentation method recently proposed in [27]. This multiscale-COS/CCC segmentation method is designed to detect and segment text regions in an image for the purpose of document compression. In this method, the authors detect the text by applying two segmentation steps in a multiscale fashion: cost optimized segmentation and connected component classification.

A number of methods focus on identifying text so that the text can be extracted and interpreted. Chen and Yuile [28] proposed a method for text detection in images of city scenes to aid the blind. In this approach, the AdaBoost machine learning algorithm was employed to train a strong classifier based on a number of initial weak classifiers. After the Adaboost learning process, a cascade of four strong classifiers using 79 features is obtained. As we show in Section IV-A3, our method runs 6.4 times faster compared to this method.

Chen *et al.* [15] also proposed a method to extract text in signs for further processing. This system is designed to automatically detect signs in natural images and translate foreign text into English. This system combines multiscale edge detection, adaptive searching, color modeling, and affine rectification in a hierarchical framework to detect the signs. Due to the similarity of the application of this system to ours, we will give a more detailed complexity comparison between the two systems in Section IV-A4.

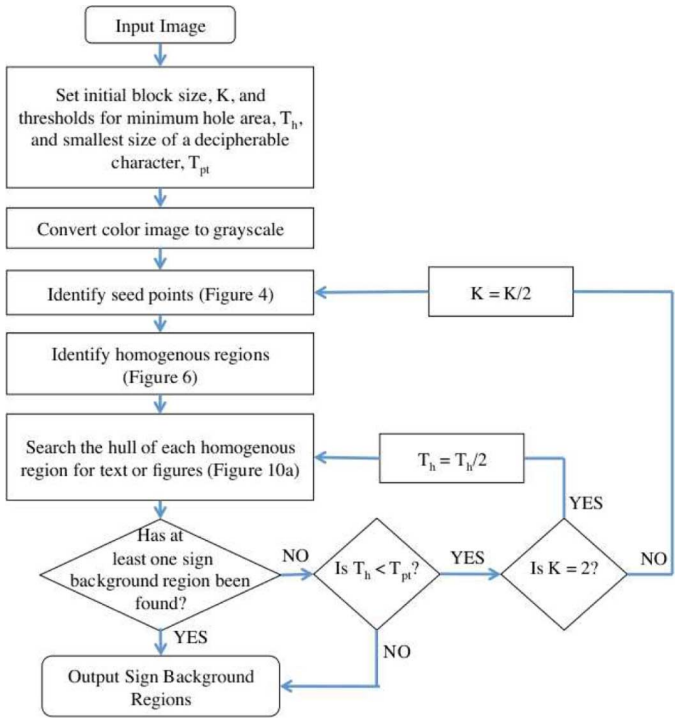

Fig. 1. Schematic representation of our method for finding text regions within natural images.

III. Approach

A schematic representation of our proposed method of sign detection and text localization in natural images is shown in Fig. 1.

In order to identify text containing regions, we first isolate areas within the image where luminance is homogeneous. We do this by calculating the homogeneity of equal sized blocks within the image and then identifying the homogenous blocks. The pixels in these homogenous blocks are the seed points used by a region growing algorithm to identify homogenous regions. Once regions containing homogenous blocks have been identified, we search the hull of each homogenous region for holes that potentially contain text. In order for these holes to be labeled as text, they must be a minimum size and have an average intensity that contrasts against the corresponding detected background region.

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 relaxed search is performed. In the relaxed search, the threshold for the minimum size of a character is reduced and the hull of all previous homogenous regions are once again searched for text. This final search is designed to find text areas that contain small characters. We now describe each step of the procedure in more detail.

A. Seed Point Detection

In order to identify signs, we begin by locating seed points in areas with homogeneous luminance. In order do this, the image is first divided into a grid of size $K \times K$ non-overlapping blocks

Fig. 2. Input image divided into a grid of $K \times K$-pixel blocks. Each pixel block is tested for homogeneity to determine if it is part of the background of a sign.

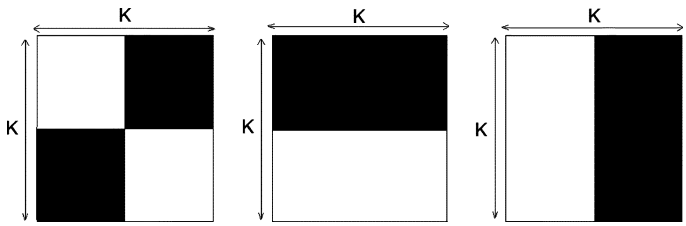

Fig. 3. Graphical representation of the $M = 3$ weight matrices used to quantify homogeneity for each $K \times K$ block in the image. The colors white and black are used to represent $+1$ and -1, respectively, in each $K \times K$ region.

where K is a power of two (see Fig. 2). The homogeneity of each block is calculated, and blocks which meet a given threshold are labeled as homogenous. The set of pixels from these homogenous blocks are then used as seed points. We determine the homogeneity of a block as follows.

For each $K \times K$ block, let I be the vector of dimension K^2 containing the luminance values for the block. We compute M homogeneity features for each block given by

$$\delta^{(m)} = \left| \frac{2}{K^2} \sum_{i=1}^{k^2} I_i w_i^{(m)} \right| \tag{1}$$

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. With this scaling, $\delta^{(m)}$ falls into the same range as the pixels. Since the maximum intensity difference in an individual pixel is 255, the average difference in pixel intensity, $\delta^{(m)}$, is then a value in the range of 0 to 255. We use $M = 3$ features corresponding to the three binary weight vectors shown in Fig. 3. Notice that each of the three features quantify the variation of pixels values within a block, with a smaller value of $\delta^{(m)}$ signifying a more homogenous texture.

After all $\delta^{(m)}$ values have been computed, we classify a block as homogenous if the L_∞ norm of the vector $\boldsymbol{\delta} = (\delta_0, \ldots, \delta_{M-1})^T$ is less than a chosen threshold, T_u, and at least one of its four neighboring blocks also meets the same condition. Seed points are exactly the set of all pixels in the homogenous blocks. A schematic representation of our proposed method of homogenous block selection is shown in Fig. 4.

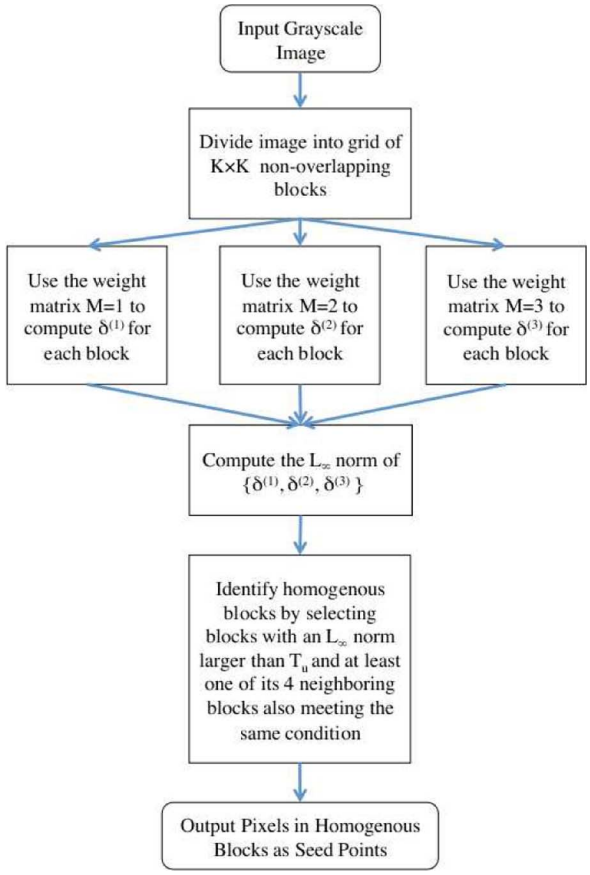

Fig. 4. Schematic representation of our method for finding homogenous blocks within the image.

Smaller block sizes have the advantage of identifying smaller homogeneous regions, while larger blocks are less susceptible to noise. Fig. 5 shows the homogeneous blocks identified for K equal to 32, 16, and 8. Homogeneous blocks in the image are illustrated as white areas. Notice that homogeneous blocks are generally not located in areas where edges reside.

It is possible that the weight matrices used to quantify homogeneity would identify a textured block, such as a chess board, as being homogeneous. In this case, these seed pixels will generally be eliminated in the next stage of the algorithm.

B. Homogenous Region Detection

Once seed points have been identified, we expand them in order to form homogeneous regions. In order to do this, a region growing method is used that produces connected pixel sets.

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 $s = (s_x, s_y)$. The neighborhood, $\psi(s)$, of a lattice point, s, is the set of pixels surrounding s. Here, we use a four point neighborhood defined as follows:

$$\begin{aligned} \psi(s) &= \psi(s_x, s_y) \\ &= \{(s_x - 1, s_y), (s_x + 1, s_y), \\ &\quad (s_x, s_y - 1), (s_x, s_y + 1)\}. \end{aligned} \tag{2}$$

A free boundary is also used so that pixels along the edge of an image have less than four neighbors each. Two neighboring

Fig. 5. Homogeneous block selection for the image in Fig. 2 using various block sizes, $k = 32$, 16, 8 (left to right). Homogeneous regions are shown as white areas. As K decreases, more homogeneous blocks are detected.

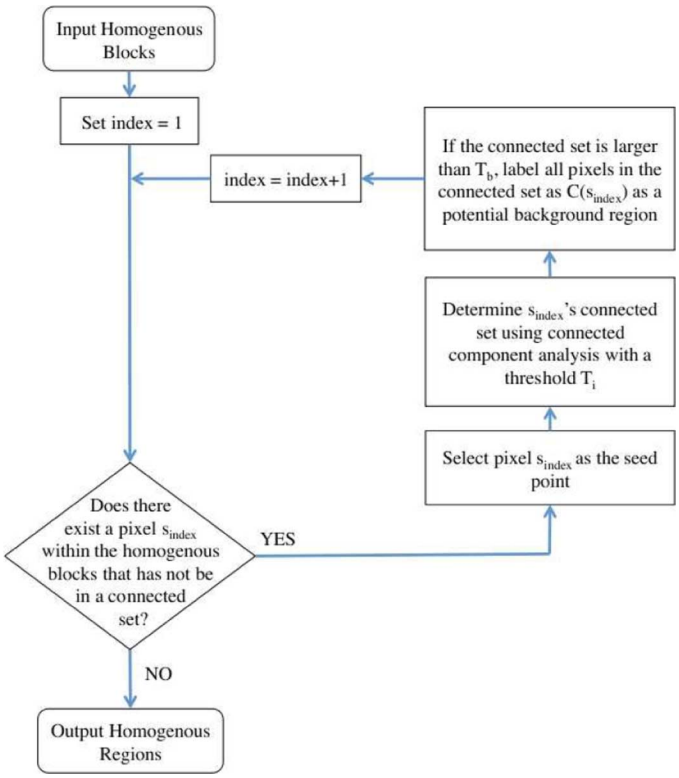

Fig. 6. Schematic representation of homogenous block growing algorithm.

pixels are considered connected if the difference between their luminance values is less than a fixed threshold, T_i. Let $c(s) \subset \psi(s)$ be the subset of neighbors which are connected to s:

$$c(s) = \{r \in \psi(s) || I(s) - I(r)| < T_i\}. \tag{3}$$

Two pixels in S, s and m, are connected if there is a path of connected neighbors from s to m.

Growing homogenous blocks into homogenous regions is performed as follows. By beginning at any seed point s_0 within the detected homogenous blocks, s_0's connected set is determined. 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 homogenous 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 homogenous blocks maps to a connected set. A schematic illustrating our region growing method is shown in Fig. 6.

All connected sets which contain more than a minimum number of pixels, T_b, are classified as homogenous regions. T_b is a function of the size of the image, $|S|$, and the block size, K : $T_b = |S| \times K/\alpha$ pixels. The value of α was obtained based on our observation of the minimum size of a readable sign in images taken by cameras with different resolutions. Since the threshold is a linear function of K, as K decreases, so does T_b. Fig. 7 illustrates the effect of this thresholding procedure. Note that, although for $K = 32$ and 16 the sky in the upper right-hand corner of the image is identified as containing a homogeneous block, the connected set is not large enough to include it as a region until $K = 8$. Fig. 7 shows the homogenous regions in an image after the homogenous blocks have been grown for a block size of 32, 16, and 8.

C. Sign Background Region Detection

Once the homogenous regions of an image are identified, each region is examined to determine if it is a sign background region. Two conditions must be met in order for the region to be identified as a sign background region: 1) the homogenous region must contain at least one hole of sufficient size, and 2) the hole's intensity must contrast against the intensity of its background.

1) Holes: In order to identify holes, we employ connected components analysis. First, each potential background region is isolated from the rest of the image in its own sub-image. In order to do so, we assign all pixels belonging to a given homogenous region the value 1 (white), and all the remaining pixels the value 0 (black). Then, a minimum rectangular bounding box is defined around the region that contains all the white pixels. The area inside the bounding box is defined as the sub-image. A row of black pixels is added to the top and bottom of the sub-image, and a column of black pixels is added to the left and right. This padding is added so that the white region is surrounded on all sides by black. Fig. 8 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.

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 homogenous region. If a region's binary sub-image has more than two connected sets, the homogenous region has at least one hole.

Noise or dirt may cause small holes to be enclosed by the homogenous 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 $T_h = \beta \times |C(b)|$ pixels are discarded from the list of connected sets, where $|C(b)|$ is the size of the homogenous region. The

Fig. 7. Segmented potential sign backgrounds found at various block sizes, $K = 32$, 16, 8 (left to right). Non-black regions are considered homogeneous. Each shade of gray corresponds to one region.

Fig. 8. Isolating regions of an image. (Top) Regions of an image. (Bottom) Isolated regions of the image enclosed by exaggerated bounding regions. Potential sign background regions are white and background regions are black.

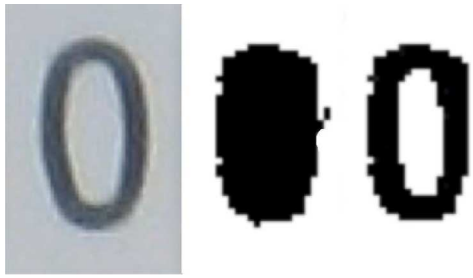

Fig. 9. (Left) Original image. (Middle) Text hole region after connected component analysis. (Right) Text hole region after discarding pixels similar to surrounding background.

bounding region and the homogenous region are always defined and never discarded from the list of connected sets. 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.

Remaining connected sets that are neither the bounding region nor the homogenous region are considered text holes. Therefore, if a region's binary sub-image has more than two connected sets remaining, the homogenous region encloses at least one text hole.

2) Intensity Contrast: In order to be seen easily, text must contrast against the background on which it is printed. In order for a hole region, H, on a background region, B, to be considered a text hole

$$|\bar{I}_B - \bar{I}_H| \geq T_c \tag{4}$$

must hold true, where $\bar{I}_B$ and $\bar{I}_H$ are the average luminance value of the pixels in B and H, respectively, and T_c is a fixed threshold.

Many characters contain holes. However, when determining the average luminance of a text hole, the luminance of holes within characters should not be calculated into the average luminance of the text hole. Therefore, if the luminance of a pixel contained in the text hole is less than γ standard deviations from the average luminance of the homogenous region, $\bar{I}_H$, it is considered to be part of the homogenous region and discarded from the text hole's set of pixels. Fig. 9 shows how a character's region is more accurately determined after discarding pixels similar to the character's background. If the text hole still meets the size constraint, and $|\bar{I}_B - \bar{I}_H| \geq T_c$ holds true, the text hole is labeled as containing text.

Any hole that does not meet this criterion is removed from the total number of text holes for the given homogenous region being examined. If there still remains at least one text hole, the region contains text, and the homogenous region is classified as a sign background region. A schematic representation of how each homogenous region is examined to determine if it contains text can be seen in Fig. 10.

The image of these text holes is then passed as an output of the algorithm. Fig. 11 shows the text holes outputted by our proposed algorithm for the image shown in Fig. 2. This output could then be passed to a binarization algorithm for finer text-background classification before applying OCR.

D. Multiscale Search Strategy

If no regions have been identified that contain holes meeting the given criteria, the block size K is reduced by half and the process is repeated until a region is identified. Repeating the process at a smaller block size typically results in new homogenous regions and may also identify homogenous regions found at previous steps. Only the newly discovered homogenous regions need to be tested to determine if they are background sign regions. This reduces the computation of the algorithm. If no region has been identified by the end of the procedure for the

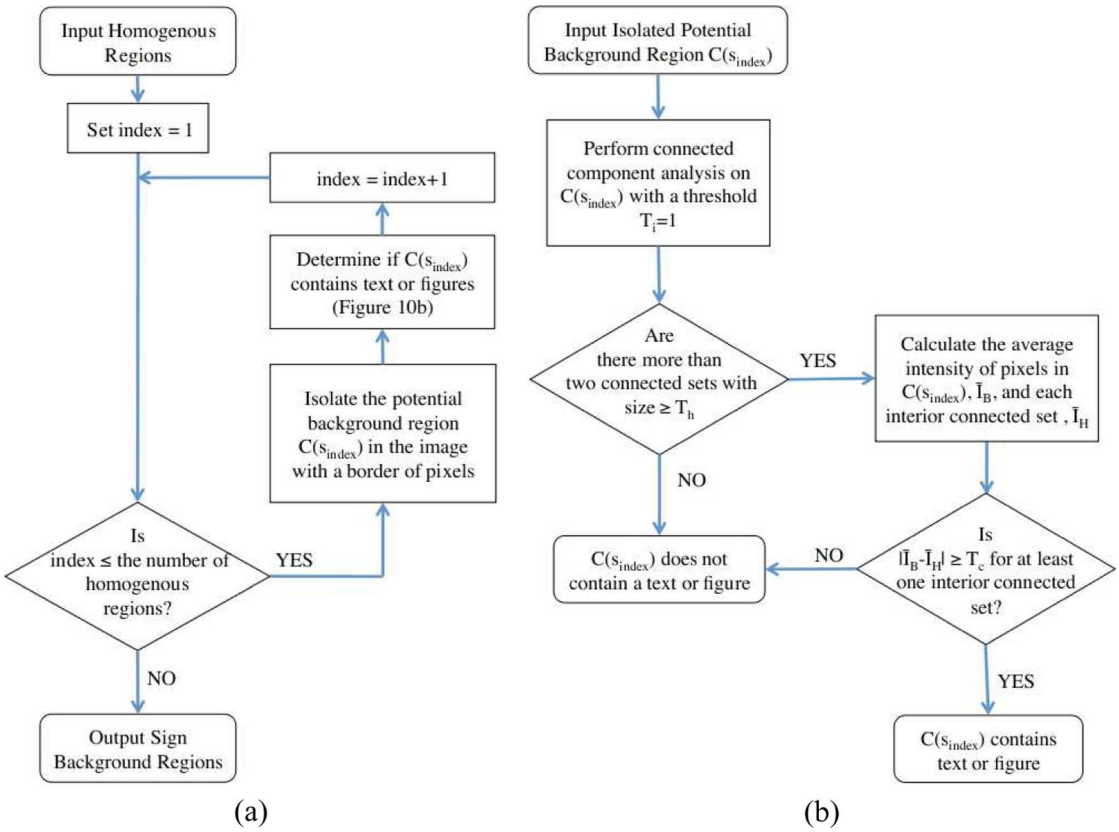

Fig. 10. Schematic representation of how each homogenous region is examined and classified as a sign background region or not a sign background region.

Fig. 11. Final output of text holes from our proposed algorithm. The white regions are identified as text characters.

smallest possible block size, $K = 2$, then the homogenous region with the largest text hole is defined as a background sign region if the size of the text hole is greater than the size of a decipherable character, T_{pt}.

IV. Results

In this section, we compare the results of our proposed sign detection and text localization method with the results of the multiscale-COS/CCC segmentation approach proposed in [27]. We chose to compare to the multiscale-COS/CCC algorithm because of its similarity to our proposed algorithm. Indeed, although the multiscale-COS/CCC algorithm was designed to be used on scanned documents, it does not assume any features of text specific to documents. Therefore, the algorithm is capable of locating the text found on signs. Both our proposed algorithm and the multiscale-COS/CCC segmentation method are able to identify text written in different fonts, colors, and languages as well as text that is tilted or skewed in the plane of the image. Additionally, both algorithms do not differentiate between alphanumeric text and simple line graphics. Therefore, in our analysis of the performance of each method, both simple line graphics and alphanumeric characters are classified as text characters. The authors of the multiscale-COS/CCC segmentation algorithm have made their source code available for testing online.

To measure the text localization accuracy of each algorithm, we used a set of 0.3-megapixel images along with corresponding ground truth segmentations. First, 241 images of road signs, flyers, and posters were taken using a VGA camera on a Nokia N800, a handheld mobile device.[1] These images were then separated into 81 training images and 160 testing images. Two binary ground truth images were manually created for each image. These images were used to objectively measure the number of false positives and false negatives found in each output image. In the first ground truth image, GT1, each character in the targeted sign region was manually segmented from the rest of the image. The most prominent sign region in the image was manually chosen as the targeted sign region. In GT1, each character is a single connected component region separated from any other character's connected component region. In the second ground

[1]This dataset is available at http://cobweb.ecn.purdue.edu/~ace/kbsigns/.

(a) (b) (c)

Fig. 12. Image (a) and its corresponding ground truth images, (b) GT1, and (c) GT2.

truth image, GT2, each sign region was manually segmented from the rest of the image. In GT2 all regions are identified, not just the targeted sign region. Fig. 12 shows an example of a test image and its two corresponding ground truth images.

1) Training: The threshold values, T_u, T_i, T_c, and weighting coefficients, β and γ, were found by minimizing the weighted error between results of training images and corresponding ground truth segmentations, GT1 and GT2. The weighted error criteria that we minimized is given by

$$\epsilon = 1 - \frac{10}{27}A_{100\%} - \frac{9}{27}A_{90\%} - \frac{8}{27}A_{80\%} + \frac{1}{100}\hat{N}_{FP}. \quad (5)$$

$A_{p\%} \in [0,1]$ is the fractional percentage of signs that have been correctly identified given at least $p\%$ of the characters in the targeted region are identified. The error criteria in (5) is defined in such a way that by minimizing the epsilon, the number of images with a large percentage of correctly identified characters ($A_{100\%}$, $A_{90\%}$, and $A_{80\%}$) is maximized. We chose to define ϵ in this way because in our application, the Rosetta phone system, it is important that all the characters of a sign are located. Locating only some of the characters may result in a false interpretation. Thus, the percentage of images with all characters correctly identified ($A_{100\%}$) is most highly weighted. Then the images with $A_{90\%}$ and $A_{80\%}$ of correctly identified characters have the second and third highest weights. The values of the weight are chosen empirically and are normalized to sum to 1.

$A_{p\%}$ is computed in the following manner. The text in ground truth images GT1 was manually segmented to produce N_{gt1-c} connected components. Each of these connected components corresponds to a character in the targeted sign region. A character is correctly identified if at least 80% of the pixels in its connected component region in GT1 were identified by the text localization algorithm. If the total number of correctly detected components is N_{TP}, then we define the fractional percentage of signs that have been correctly identified as

$$A_{p\%} = \frac{1}{N_I}\sum_{i=1}^{N_I} C_i \quad (6)$$

where N_I is the number of images used for training and

$$C_i = \begin{cases} 1 : \frac{N_{TP}}{N_{gt1-c}} \times 100 \geq p \\ 0 : \frac{N_{TP}}{N_{gt1-c}} \times 100 < p \end{cases} \quad (7)$$

for the ith image. $\hat{N}_{FP}$ is the average number of false detections in each image. $\hat{N}_{FP}$ is calculated by first counting the number of connected component regions identified by the text localization algorithm that are larger than the smallest size of a decipherable letter, T_{pt}. Then, each connected component region is classified as being contained within a sign region or being a false detection. A connected component is considered a false detection if less than 80% of the pixels are identified in the corresponding GT2 segmented image. The number of false detections in each image is counted and assigned to N_{FP}. $\hat{N}_{FP}$ is found by computing the average value of N_{FP} for the images used in training. For our application, missed detections are generally more serious than false detections, so we more heavily weight missed detections in our equation of ϵ.

TABLE I
SELECTED VALUES OF THRESHOLDS AND COEFFICIENTS BASED UPON TRAINING ON 81 0.3-MEGAPIXEL IMAGES

Threshold	Value
T_u	1
T_i	6
T_c	60
Coefficient	Value
β	.007
γ	3

We ran our algorithm on our 81 0.3-megapixel training images of signs, flyers, and posters with 1200 different combinations of thresholds and weighting coefficients. By calculating ϵ for each combination of thresholds, we were able to pick the threshold with the best performance on our set of training images. Our selected threshold and coefficient values can be seen in Table I. Plots showing the effect of varying values of individual thresholds around the optimum threshold combination are shown in Fig. 13. Note that T_u and T_i must take on a discrete value in the range of $[1, 255]$ while T_c, β, and γ are selected from a continuous range of values.

2) Experimental Results: We tested the performance of our algorithm against the results of the multiscale-COS/CCC segmentation approach proposed in [27] on a database of 160 0.3-megapixel images of signs, flyers, and posters. To measure the text localization accuracy of each algorithm, we compared the results of each method with the corresponding ground truth images, GT1 and GT2. A sample of results from the two algorithms can be seen in Fig. 14.

To evaluate the text localization accuracy, we first measured the percentage of signs that were correctly identified for both methods, $A_{p\%}$, for $p = 10$, 20, 30, 40, 50, 60, 70, 80, 90, and 100. $A_{p\%}$ is the fractional percentage of signs that have been correctly identified given at least $p\%$ of the characters in the

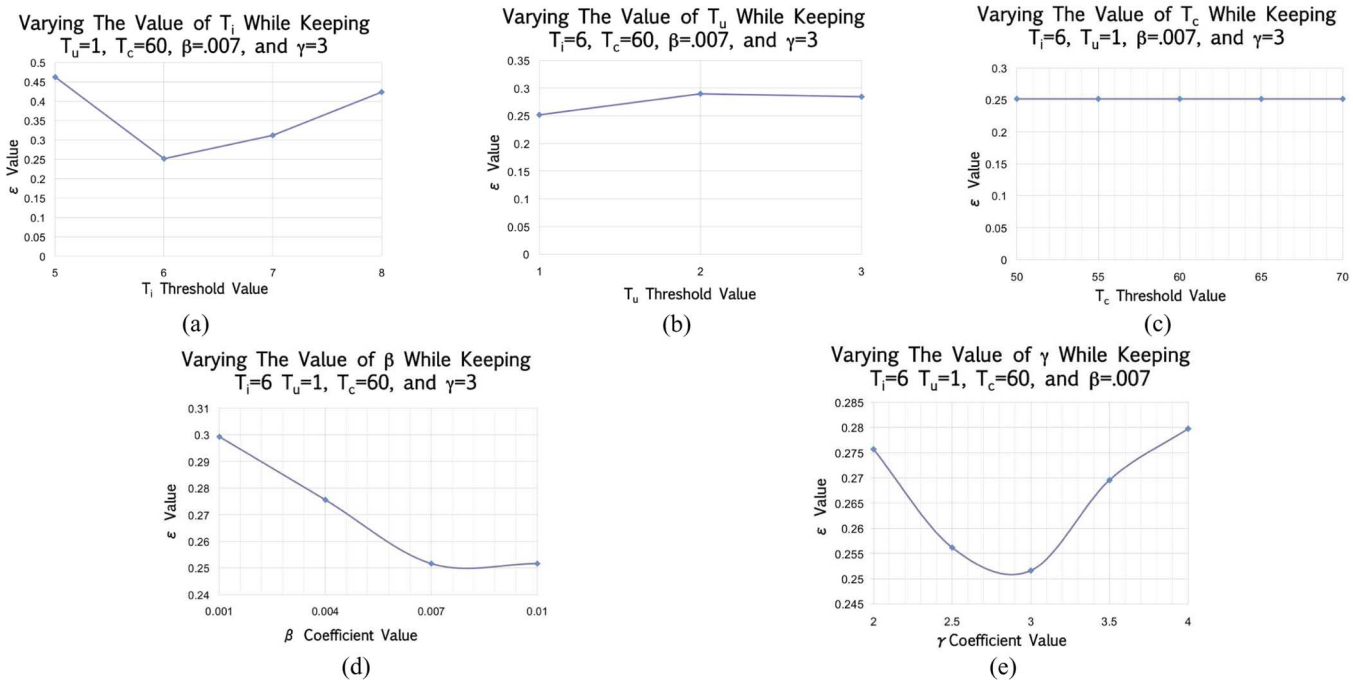

Fig. 13. Effect of varying values of individual thresholds or coefficients around the optimum threshold combination ($T_u = 1, T_i = 6, T_c = 60, \beta = .007$, and $\gamma = 3$) on the value of ϵ.

TABLE II
TRUE POSITIVE AND FALSE NEGATIVE COMPARISON OF OUR PROPOSED METHOD VERSUS THE MULTISCALE-COS/CCC SEGMENTATION ALGORITHM

Method	$\hat{N}_{TP}$	$\hat{N}_{FN}$
Our Method	29.09	4.69
Multiscale-COS/CCC segmentation	29.52	4.26

TABLE III
FALSE DETECTION COMPARISON OF OUR PROPOSED METHOD VERSUS THE MULTISCALE-COS/CCC SEGMENTATION ALGORITHM

Method	$\hat{N}_{FP}$	$\hat{X}_{FP}$
Our Method	6.46	53.32
Multiscale-COS/CCC segmentation	29.01	76.63

targeted sign region were identified. A more detailed explanation of how to compute $A_{p\%}$ is given in the training subsection above. The results of this evaluation can be seen in Fig. 15.

This plot shows that the multiscale-COS/CCC segmentation algorithm locates a higher percentage of signs than our proposed method does when no more than 80% of the characters in the targeted sign region must be identified in order for the sign to be located. However, our proposed algorithm outperforms the multiscale-COS/CCC segmentation algorithm when at least 90% or 100% of the characters must be located in the targeted sign region. Therefore, although the mutliscale-COS/CCC segmentation algorithm has a higher probability of locating some of the characters within the targeted sign region, our algorithm is more likely to be able to identify the entire message. Locating all characters within the targeted region is very important if the text is then to be interpreted, as is being done in the Rosetta phone system. If characters or entire words of the message are lost in the text localization phase, it can be very difficult to recover the original message. The average number of correctly detected characters, N_{TP}, and the average number of mis-detections, N_{FN}, in each image for both systems are shown in Table II.

We also compared our proposed algorithm with the multiscale-COS/CCC segmentation algorithm by measuring the average number of false detections, $\hat{N}_{FP}$. The value of $\hat{N}_{FP}$ is computed as described in the training subsection above. Additionally, the average number of pixels in each false detection ($\hat{X}_{FP}$) is computed by summing the number of pixels in all of the false detections and then dividing by the number of false detections, N_{FP}. More specifically

$$\hat{X}_{FP} = \frac{1}{N_{FP}} \sum_{i=1}^{N_{FP}} X_i \tag{8}$$

where X_i is the number of pixels in the ith false detection. The values of $\hat{N}_{FP}$ and $\hat{X}_{FP}$ for both our proposed algorithm and the multiscale-COS/CCC segmentation algorithm can be seen in Table III. As shown in the table, our proposed algorithm identified fewer false positives on average than the multiscale-COS/CCC segmentation algorithm for our database of 160 testing images. Additionally, the average size of a false positive in our algorithm is smaller than the average size of a false positive in the multiscale-COS/CCC segmentation algorithm. By identifying fewer false positives at the text localization phase, less computation is needed later on to filter out false positives before or during OCR.

The precision and recall of the two methods are shown in Table IV. The standard equations of precision and recall were used in order to compute these values. As shown in the table, our method and the multiscale-COS/CCC have a comparable recall value. However, our method produces a higher precision

Fig. 14. Sample of experimental results of text localization run on images of signs. Image (a) is the original image. Image (b) is the corresponding GT1. Image (c) is the resultant output of our proposed text localization algorithm. Image (d) is the resultant output of the multiscale-COS/CCC segmentation algorithm. White regions are identified as text characters. The varying characteristics of signs in this sample of images shows that our algorithm works on 1) simple line graphics, 2) skewed signs, 3) signs of different shapes, 4) signs with darker backgrounds and lighter text, 5) signs with text of different sizes, and 6) signs written in different languages.

TABLE IV
PRECISION AND RECALL COMPARISON OF OUR PROPOSED METHOD VERSUS THE MULTISCALE-COS/CCC SEGMENTATION ALGORITHM

Method	Precision	Recall
Our Method	0.818	0.861
Multiscale-COS/CCC segmentation	0.504	0.874

value than the multiscale-COS/CCC method. This once again shows that although both methods identify a similar percentage of text characters in the images, our method returns less false positives than the multiscale-COS/CCC method.

3) Time Comparison: Since our segmentation approach was developed to run on a handheld device, it was designed to be simple and computationally inexpensive. To evaluate the computational complexity of our proposed method, we compared the average computation time of our method and the multiscale-COS/CCC segmentation algorithm. The computation time of sign detection and text localization was measured for both algorithms on a database of 241 0.3-megapixel images of signs, flyers, and posters on a PC (CPU 2.80 GHZ, 1 GB RAM). The results of the time comparison can be seen in Table V. As can be seen in the table, our proposed method processes an image in much less time than the multiscale-COS/CCC segmentation algorithm.

We also measured the average computation time of our method on a handheld mobile device. Our text detection method processes a 0.3-megapixel image of a sign and returns the correct output on the Nokia N800 device (CPU 330 MHz, 128 MB RAM) in approximately 0.766 s (standard deviation of 0.678 s). Images containing text regions that are not correctly identified by the algorithm take a longer time to process. Processing these images takes an average of 6.53 s on the N800.

To additionally test the computational complexity, we ran our method on a database of 144 images used in the ICDAR 2005 competition on locating text in camera captured scenes. The average time that it took to run the algorithm was then compared to the average times of the competition's submitted algorithms presented in [29]. Each of the times reported is the average time it took an algorithm to run on an image from the ICDAR 2005 dataset using a 2.4-GHz PC running either Windows XP or Linux. Although we are unable to run our algorithm on the "exact" same hardware, we used a 2.4-GHz dual core processor running Linux and single threaded code for measuring the execution time of our algorithm on the same dataset. Our

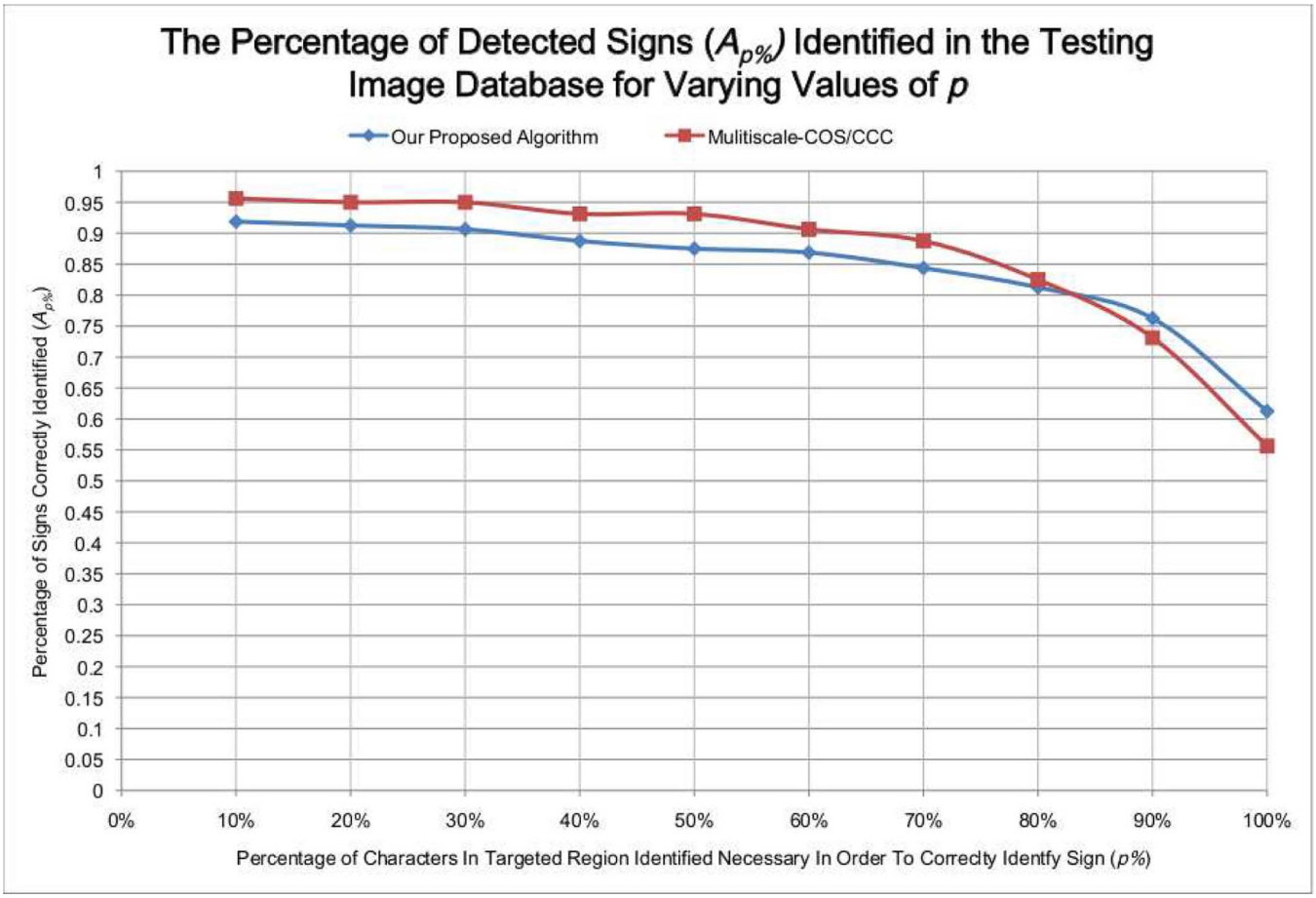

Fig. 15. Value of $A_{p\%}$ for varying values of p for our proposed algorithm and the multiscale-COS/CCC segmentation algorithm.

TABLE V
TIME COMPARISON OF OUR PROPOSED METHOD VERSUS THE MULTISCALE-COS/CCC SEGMENTATION ALGORITHM

Method	Average Real Time (s)	Average User Time (s)	Average System Time (s)
Our Method	0.1187	0.0274	0.0727
Multiscale-COS/CCC segmentation	1.1649	0.6047	0.1343

TABLE VI
AVERAGE TIME (IN SECONDS) TO LOCATE TEXT IN THE ICDAR 2005 DATABASE IMAGES ON A 2.4-GHZ PROCESSOR DEVICE

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

system proved to be very computationally inexpensive, running over six times faster than the fastest system submitted to the ICDAR 2005 competition. Table VI contains the average time in seconds to process an image for each system on a 2.4-GHz processor. Descriptions of the five algorithms submitted to the ICDAR 2005 Competition can be found at [28]–[32].

The run time of our proposed approach was also compared to the recent work by Shivakumara *et al.* [25]. This paper tested their method on their own 960 images as well as 251 images from ICDAR 2003 and 45 images from Microsoft Research data set. Before processing, they resized all images to 256 by 256 pixels to save computational costs. The processing time of their algorithm for 256 by 256 images on Core 2 Duo 2.0-GHz machine is reported to be 7.8 s for horizontal text and 10.3 s for non-horizontal text, whereas our proposed method can process images of size 480 by 640 on a PC with a CPU of 2.8 GHz in 0.1187 s regardless of the orientation of the text line. We believe that even given the differences in the processor speeds and image sizes, this comparison shows that our method is still considerably faster than the work reported in [25].

4) Complexity Comparison: To put the cost of our method in perspective, we compare it to the system proposed in [15], which we refer to as System CYZW. This system is also designed to detect signs in natural images. Below we compare the complexity of our approach against System CYZW's algorithm. A schematic representation of System CYZW's proposed method of sign detection in natural images is shown in Fig. 16.

Both our system and System CYZW begin by applying a set of filters that identify homogenous regions or edges, respectively. In our system, a set of three binary filters is applied to

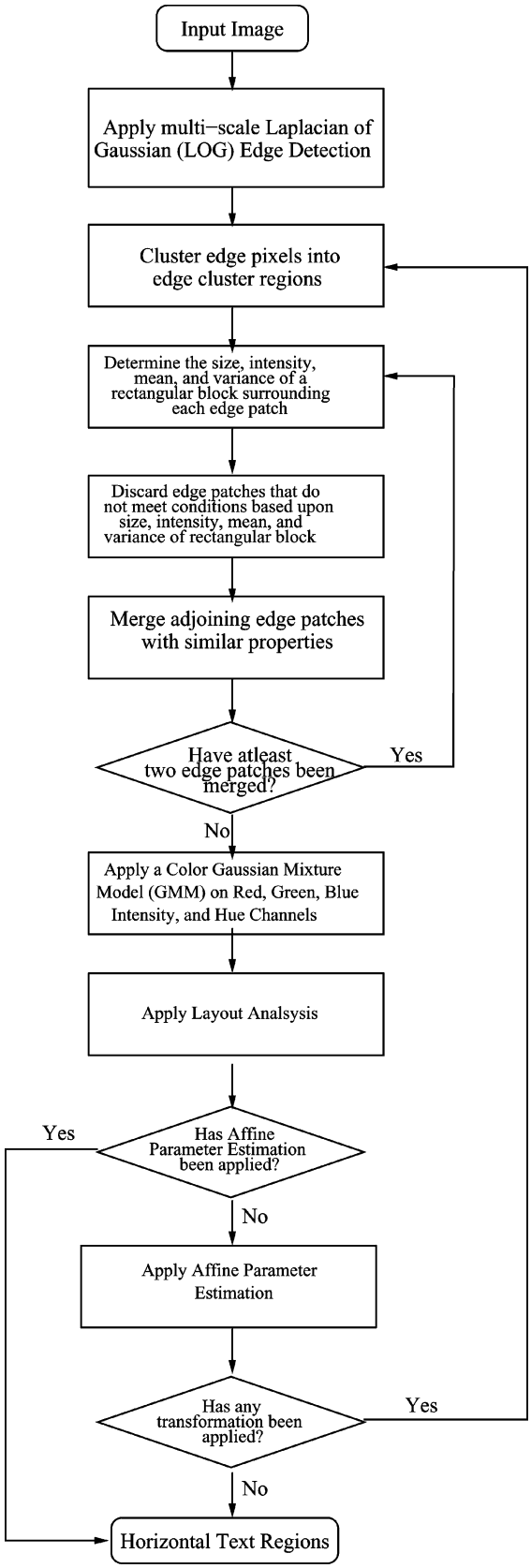

Fig. 16. Schematic representation of System CYZW's method for finding text regions within natural images.

non-overlapping blocks in the image. These three binary filters enable us to identify the homogenous regions in an image and ignore areas containing edges. System CYZW relies on a multi-scale Laplacian of Gaussian (LOG) filter in order to identify edges within the image. Both algorithms threshold the resultant filtered image to identify uniform or edge regions. Binary filtering used in our system is a low complexity process that involves only the summation and subtraction of pixel values, performed on non-overlapping blocks. In contrast, System CYZW's LOG edge detector obtains the edge set by passing over every pixel in the image and computing the values by performing float multiplication and summation. Although a LOG edge detector identifies the edges in an image in a precise manner, binary filters provide a simple and low-complexity alternative with similar results.

The next step in both our method and System CYZW's method is to discard areas in the image that were identified in the previous step as being uniform or edge-containing, respectively. In our method, homogenous blocks that are not adjacent to at least one other homogenous blocks are removed from the list of homogenous blocks. This places a size constraint on the homogenous region being identified. In System CYZW's method, the set of all edges in the image are clustered into edge patches. Once this is complete, some edge patches are excluded from further consideration based upon criteria applied to the size, intensity, mean, and variance of a rectangle surrounding each edge patch. After each method has identified its candidate areas of the image, these blocks are grown and merged to form entire regions within the image. In our method, a region growing method is used to expand the homogenous blocks into homogenous regions. The region growing method compares adjacent pixels and connects them if the difference between their intensity values is below a given threshold. Applying CC to the entire image requires at most two passes of the image. In each pass, every pixel is compared to its top and left neighbors. Since only homogenous blocks are expanded, usually less than two full passes over the image are necessary to fully grow the homogenous blocks. System CYZW's method grows edge patches by merging a patch with adjoining edges that have similar properties. Once multiple edge patches have been merged, the edge patches are evaluated again to determine if each edge patch should continue to the next step or be discarded. This recursive procedure is repeated until no more updates can be made to the edge patches.

Color is an important feature for verifying the text localization results and identifying the outliers. Regions that contain text and its background pixels tend to have a distinctive binomial distribution in one of the color subspaces. Both methods test this property for the candidate regions. In our method, the intensity of the background region and the holes within them are averaged separately, and the difference between the means is compared against a threshold. This simple step allows us to determine whether a candidate region is certified as a text area. In System CYZW's method, the color distribution of the candidate regions are modeled in five subspaces of Red, Green, Blue, Hue, and Intensity, by Gaussian mixture models (GMMs). The EM algorithm is used to estimate the parameters of the GMMs. The subspace with larger difference between the means and with smaller variances is selected for the future steps in the system. This color modeling process increases the complexity

of the system. Although color modeling in different subspaces to find the one with higher confidence is useful in applications where the text is engraved in a material or where there is a highlight in the text [15], the additional complexity is rarely necessary in applications such as sign text localization. In our proposed method, we chose to work with only the grayscale image during processing. This was done for two reasons. First, using the grayscale image over a color image reduces the complexity of methods during processing. Second, many handheld mobile devices contain low-quality cameras that often generate color noise in their images. Converting the image to grayscale helps to reduce the effect of color noise while processing the image. Although it is possible that the text of a grayscale image would not contrast with its background, this is rarely a problem in the case of text on signs. In fact, road signs are made to contrast in intensity in order to be easily seen by colorblind individuals [33].

At this point, our system has identified the sign background regions in the image. If there are not detected sign background regions, the process is repeated with a smaller block size. System CYZW's method moves to next steps which are the layout analysis and affine rectification. These steps are required in the text localization process since the method relies on the direction of the edges in the text. Thus, the skewness, caused by non-vertical view of the camera with respect to the text, may result in missing some of the text regions. The layout analysis in this system aims to align the characters and group the ones that are in the same context. The neighboring candidate regions are clustered together based on their color attributes. A Hough transform is used to find the line that fits the center of the cluster regions in a layout. For each layout region, that contains multiple characters, the affine parameters are obtained separately. Two pairs of parallel lines in the sign are required for this purpose. These lines can be either the boundary lines of the sign, if the sign is rectangular, or the lines of the characters in the layout. The assumption of existence of parallel lines in the text does not hold for every language (e.g., Arabic). Once the affine parameters are obtained, the text regions are warped using these parameters. A B-spline interpolation is used to fill out the transformed regions. After affine rectification for each candidate region, the text detection process is repeated to refine the results.

V. Conclusions

We have presented a method for locating text within natural images. The algorithm relies on a fundamental feature of text: text is usually surrounded by a contrasting, uniform background. Our proposed method of text segmentation 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. The algorithm has proved to have a high performance while also being computationally inexpensive.

References

[1] A. K. Jaina and B. Yu, "Automatic text location in images and video frames," *Pattern Recognit.*, vol. 31, pp. 2055–2076, 1998.

[2] Q. Yuan and C. Tan, "Text extraction from gray scale document images using edge information," in *Proc. 6th Int. Conf. Document Analysis and Recognition*, 2001, pp. 302–306.

[3] S. A. R. Jafri, M. Boutin, and E. J. Delp, "Automatic text area segmentation in natural images," in *Proc. ICIP*, 2008, pp. 3196–3199.

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

[5] J. Yang, J. Gao, Y. Zhang, X. Chen, and A. Waibel, "An automatic sign recognition and translation system," in *Proc. 2001 Workshop Perceptive User Interfaces (PUI '01)*, 2001, pp. 1–8.

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

[7] B. Sin, S. Kim, and B. Cho, "Locating characters in scene images using frequency features," in *Proc. IEEE Int. Conf. Pattern Recognition*, 2002, vol. 3, pp. 489–492.

[8] W. Wu, X. Chen, and J. Yang, "Detection of text on road signs from video," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 4, pp. 378–390, Dec. 2005.

[9] W. H. P. Shivakumara and C. Tan, "An efficient edge based technique for text detection in video frames," in *Proc. 8th IAPR Workshop Document Analysis Systems*, Sep. 2008, pp. 307–314.

[10] L. Agnihotri and N. Dimitrova, "Text detection for video analysis," in *Proc. IEEE Workshop Content-Based Access of Image and Video Libraries*, 1999, pp. 109–113.

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

[12] 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 Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1631–1638, Dec. 2003.

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

[14] T. I. Y. Liu and S. Goto, "A robust algorithm for text detection in color images," in *Proc. 8th Int. Conf. Document Analysis and Recognition (ICDAR05)*, Sep. 2005.

[15] X. Chen, J. Yang, J. Zhang, and A. Waibel, "Automatic detection and recognition of signs from natural scenes," *IEEE Trans. Image Process.*, vol. 13, no. 1, pp. 87–99, Jan. 2004.

[16] 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," in *Proc. SPIE*, 2008, vol. 6821.

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

[18] J. Wu, S. Qu, Q. Zhuo, and W. Wang, "Automatic text detection in complex color image," in *Proc. Int. Conf. Machine Learning and Cybernetics*, Nov. 2002, vol. 3, pp. 1167–1171.

[19] W. Jirattitichareon and T. H. Chalidabhongse, "Automatic detection and segmentation of text in low quality Thai sign images," in *Proc. IEEE Asia Pacific Conf. Circuits and Systems*, 2006, pp. 1000–1003.

[20] X. Huang and H. Ma, "Automatic detection and localization of natural scene text in video," in *Proc. 2010 20th Int. Conf. Pattern Recognition (ICPR)*, 2010, pp. 3216–3219.

[21] J. Gllavata, R. Ewerth, and B. Freisleben, "A robust algorithm for text detection in images," in *Proc. 3rd Int. Symp. Image and Signal Processing and Analysis*, Sep. 2003, vol. 2, pp. 611–616.

[22] P. Dubey, "Edge based text detection for multi-purpose application," in *Proc. 8th Int. Conf. Signal Processing*, 2006, vol. 4.

[23] Y. Liu, S. Goto, and T. Ikenaga, "A robust algorithm for text detection in color images," in *Proc. 8th Int. Conf. Document Analysis and Recognition*, 2005, vol. 1, pp. 399–403.

[24] N. Ezaki, M. Bulacu, and L. Schomaker, "Text detection from natural scene images: Towards a system for visually impaired persons," in *Proc. 17th Int. Conf. Pattern Recognition*, 2004, vol. 2, pp. 683–686.

[25] P. Shivakumara, T. Q. Phan, and C. L. Tan, "A Laplacian approach to multi-oriented text detection in video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 412–419, Feb. 2011.

[26] J. Yu and Y. Wang, "Apply SOM to video artificial text area detection," in *Proc. Int. Conf. Internet Computing in Science and Engineering*, 2009, pp. 137–141.

[27] E. Haneda and C. Bouman, "Multiscale segmentation for MRC document compression using a Markov random field model," in *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing (ICASSP)*, Mar. 2010, pp. 1042–1045.

[28] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," *Proc. Computer Vision and Pattern Recognition (CVPR) Conf.*, vol. 2, pp. 366–373, 2004.

[29] S. Lucas, "ICDAR 2005 text locating competition results," in *Proc. 8th Int. Conf. Document Analysis and Recognition*, 2005, vol. 1, pp. 80–84.
[30] C.-T. Wu, "Embedded-text detection and its application to anti-spam filtering," M.Sc. thesis, Univ. California, Santa Barbara, 2005.
[31] J. Kim, S. Park, and S. Kim, "Text locating from natural scene images using image intensity," in *Proc. Int. Conf. Document Analysis and Recognition*, 2005, vol. 0, pp. 655–659.
[32] N. Ezaki, "Text detection from natural scene images: Towards a system for visually impaired persons," in *Proc. Int. Conf. Pattern Recognition*, 2004, pp. 683–686.
[33] M. Neiva, A New Look at Colour Through Greyscale Eyes. [Online]. Available: http://www.icograda.org/feature/current/articles1681.htm.

Katherine L. Bouman (S'09) received the B.S.E. degree in electrical engineering (*summa cum laude*) from the University of Michigan, Ann Arbor, in 2011. She is currently pursuing the Ph.D. degree in the area of systems, communication, control, and signal processing at the Massachusetts Institute of Technology, Cambridge, as an Irwin Mark Jacobs and Joan Klein Jacobs Presidential fellow.

She is the recipient of several awards including a Barry M. Goldwater Scholarship and an NSF Graduate Fellowship. Her research interests include computer vision and machine learning with an emphasis on pattern recognition.

Golnaz Abdollahian (M'10) received the B.S. degree in electrical engineering from Sharif University of Technology, Tehran, Iran, in 2004 and the Ph.D. degree in the area of communications, networking, signal and image processing from Purdue University, West Lafayette, IN, in 2009.

She is currently a postdoc in the Department of Electrical and Computer Engineering, and Neural Research Institute at the University of California, Santa Barbara. Her research interests include multimedia content analysis, computer vision, and pattern recognition.

Mireille Boutin (M'05) received the Ph.D. degree in mathematics from the University of Minnesota, Minneapolis, under the direction of P. J. Olver.

She is an Assistant Professor in the School of Electrical and Computer Engineering at Purdue University, West Lafayette, IN. Her current research interests include lightweight image processing for portable device applications and computational mathematics.

Edward J. Delp (S'70–M'79–SM'86–F'97) was born in Cincinnati, OH. He received the B.S.E.E. (*cum laude*) and M.S. degrees from the University of Cincinnati, Cincinnati, OH, and the Ph.D. degree from Purdue University, West Lafayette, IN.

In May 2002, he received an Honorary Doctor of Technology from Tampere University of Technology, Tampere, Finland. From 1980 to 1984, he was with the Department of Electrical and Computer Engineering, The University of Michigan, Ann Arbor. Since August 1984, he has been with the School of Electrical and Computer Engineering and the School of Biomedical Engineering, Purdue University. He is currently the Charles William Harrison Distinguished Professor of Electrical and Computer Engineering and Professor of Biomedical Engineering. His research interests include image and video compression, multimedia security, medical imaging, multimedia systems, communication, and information theory.

Dr. Delp is a Fellow of the SPIE, a Fellow of the Society for Imaging Science and Technology (IS&T), and a Fellow of the American Institute of Medical and Biological Engineering. In 2004, he received the Technical Achievement Award from the IEEE Signal Processing Society (SPS) for his work in image and video compression and multimedia security. In 2008, he received the Society Award from the SPS. This is the highest award given by SPS and it cited his work in multimedia security and image and video compression. In 2009, he received the Purdue College of Engineering Faculty Excellence Award for Research.