An Unsupervised Approach to Color Video Thresholding

Eliza Yingzi Du and Chein-I Chang
Remote Sensing Signal and Image Processing Laboratory
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
1000 Hilltop Circle, Baltimore, MD 21250
Paul D. Thouin
Department of Defense, Fort Meade, MD 20755

Abstract

Thresholding color video images is challenging because of the low spatial resolution and the complex backgrounds. This paper investigates the issue of thresholding these images by reducing the number of colors in order to improve automated text detection and recognition. An unsupervised thresholding approach is presented which reduces the background complexity while retaining the important text character pixels. The experiments show that our proposed thresholding approach performs significantly better than simple image histogram-based methods which generally do not produce satisfactory results.

I. INTRODUCTION

Information retrieval from video images has become an increasingly important research area in recent years. The rapid growth of digitized video collections is due to the widespread use of digital cameras and video recorders combined with inexpensive disk storage technology. Textual information contained in video images can provide one of the most useful keys for successful information indexing and retrieval. Keyword searches for scene text of interest within video images can provide additional capabilities to the search engines. In video images, text characters generally have much lower resolution and dimmer intensity than binary document characters. In addition, video text characters may also have various colors, sizes, styles, and orientations within the same image. Furthermore, the video background is generally much more complex than that of document images. A combination of this complex background and a large variety of low-quality characters cause thresholding algorithms that are designed for document image processing to perform poorly on video images. In this paper, we investigate this issue and develop an RGB-based approach that thresholds a video image in the color domain and fuses the obtained threshold values into a set of multiple threshold values via a clustering process. These values are then used to segment text characters from the video image background.

There are many color segmentation techniques. For color image thresholding, many approaches transfer a color image to its gray scale intensity image via the HSI (hue-saturation-intensity) model and then apply various gray scale thresholding methods. A problem with this approach is that during the HSI conversion the color information is lost and is not used for thresholding. In many cases the thresholding results are not satisfactory. For example, if only the intensity information is used, pure red and pure blue will be transferred to the same gray scale value. Fig. 1(a) shows an original color image that has pure blue background and pure red text characters. Fig. 1(b) shows an HSI-converted gray scale intensity image with only one gray level where the text characters disappeared. Fig. 1(c) is the desired thresholded result, which segmented the text characters from the background.

(a) color image (b) HSI conversion (c) desired result

Figure 1. An example of color image thresholding

To resolve this problem, an RGB-based thresholding approach was developed that can be used for video and other low-resolution color images. It is basically a multi-thresholding technique that uses an unsupervised within-class/between-class clustering process to fuse threshold values obtained from each of the RGB color domains in order to derive a set of multiple thresholds that can segment text effectively while retaining original colors in the image background. The following sections describe this approach and discuss some preliminary experimental results.

II. UNSUPERVISED COLOR THRESHOLDING

Color images are composed of 3-D information commonly represented by red, green, and blue. The first step in the proposal thresholding technique is to apply the grayscale thresholding method to the red, green and blue dimensions respectively. A number of different grayscale thresholding methods could be used, including Otsu's method [1], an entropy based method, or a 2-D thresholding method. We selected two of these techniques for our evaluations.

Suppose that the threshold values obtained for each of the red, green, and blue colors are t_r, t_g, t_b respectively. Let each pixel in a video image be denoted by

$p_{i,j} = (r_{i,j}, g_{i,j}, b_{i,j})$ with (i,j) being the spatial coordinate of the pixel. The pixel can be thresholded as $\tilde{p}_{i,j} = (\tilde{r}_{i,j}, \tilde{g}_{i,j}, \tilde{b}_{i,j})$ with

$$\tilde{r}_{i,j} = \begin{cases} 1 & r_{i,j} > t_r \\ 0 & r_{i,j} \le t_r \end{cases}$$
$$\tilde{g}_{i,j} = \begin{cases} 1 & g_{i,j} > t_g \\ 0 & g_{i,j} \le t_g \end{cases}$$
$$\tilde{b}_{i,j} = \begin{cases} 1 & b_{i,j} > t_b \\ 0 & b_{i,j} \le t_b \end{cases} \quad (1)$$

As a result, each (r,g,b)-color pixel can be encoded by a 3-bit codeword (c_1,c_2,c_3) where c_i for $1 \le i \le 3$ is a binary value taking either 0 or 1. Then we cluster all image pixels into 8 classes $\{C_k\}_{k=1}^{8}$ according to their associated codewords where each codeword represents a clustered class. For example, all pixels encoded by (1,0,0) will be clustered into one class.

Next, the mean of the k-th clustered class C_k, denoted by $\mu_k = (r_k, g_k, b_k)$, is calculated by

$$r_k = \sum_{r_{i,j} \in C_k} r_{i,j} / N_{r,k}$$
$$g_k = \sum_{g_{i,j} \in C_k} g_{i,j} / N_{g,k}$$
$$b_k = \sum_{b_{i,j} \in C_k} b_{i,j} / N_{b,k} \quad (2)$$

The definitions of within-class and between-class in Otsu's method allow us to calculate the within-class distance $\sigma_k = (\sigma_{r,k}, \sigma_{g,k}, \sigma_{b,k})$ for each class of $\{C_k\}_{k=1}^{8}$ as follows

$$\sigma_k = \frac{1}{N_k} \sqrt{\sum_{i,j \in C_k} \left\{ (r_{i,j} - r_k)^2 + (g_{i,j} - g_k)^2 + (b_{i,j} - b_k)^2 \right\}}$$
(3)

where N_k is the number of pixels in class C_k. The between-class distance $\sigma_{kj} = (r_{kj}, g_{kj}, b_{kj})$ between two classes C_k and C_j with $k \ne j$ can be calculated in a similar manner.

Now we use the within-class distances σ_k, σ_j and the between-class distance σ_{kj} obtained above as criteria to reshuffle pixels to form a new set of clusters. If two classes C_k and C_j for $k \ne j$ with either $\sigma_k \ge \sigma_{kj}$ or $\sigma_j \ge \sigma_{kj}$, these two classes be merged to one class. This is because the distance between two classes, σ_{kj} must be greater than their individual within class distances, σ_k and σ_j. This reclustering process is repeated until all the between-class distances are greater than their corresponding within-class distances, in which case no classes will be reshuffled.

This idea is very similar to that of the ISODATA classifier [2] except for three subtle differences. One is that the criteria used in our process are not nearest neighboring rule as used in the ISODATA. A second difference is that if class A is merged with class B, and then class B is also merged with a third class C, then these three classes A, B and C must be merged into one class. As a result, the number of classes is therefore reduced. This yields a third difference, which is that there is no need of predetermining the number of classes to be clustered as required by the ISODATA.

Summary of Color Thresholding Algorithm

1. Use a gray-level thresholding technique, such as Otsu's method, to threshold the image in three color domains individually. Let t_r, t_g, t_b denote the resulting threshold values.
2. Use Eq. (1) to encode all video pixels into a 3-bit binary codewords and all pixels with the same codeword will be clustered into a single class.
3. Use Eq. (2) to calculate the mean for each class.
4. Use Eq. (3) to calculate the within-class distance and the between-class distance for each class.
5. For any two classes C_k and C_j with $k \ne j$, compare the within-class distances σ_k and σ_j against the between-class distance σ_{kj} to see if $\sigma_k \ge \sigma_{kj}$ or $\sigma_j \ge \sigma_{kj}$. If not, terminate the reclustering process. Otherwise, continue.
6. Merge the classes C_k and C_j into one class and go to Step 3. It should be noted that if the classes C_k and C_j are merged with a third class C_l, these three classes C_k, C_j, C_l must be merged into one class.

III. EXPERIMENTAL RESULTS

In this section, we illustrate several images that have been thresholded with this technique. Overall, this approach appears to do a good job separating text from background in complex color images. It is clearly better than the simple global thresholding approaches which are frequently used.

The initial step of thresholding the three color domains was implemented using two different methods – Otsu's method and a joint relative entropy approach. Two reasons for selection of Otsu's method are: (1) the proposed clustering process is based on a within-class/between-class criterion which is also the criterion used in Otsu's method; and (2) Otsu's method is a widely used thresholding technique, which has been shown to be very effective for gray level images. The selection of the joint relative entropy (JRE) is based on the recent results reported in [3] where the JRE was effective in color thresholding when a single threshold value was used.

In order to demonstrate the performance of this proposed thresholding, two techniques are evaluated. The first uses Otsu's method and is referred to as RGB/Otsu and the second technique uses joint relative entropy and is referenced as RGB/JRE. Both RGB-based techniques were evaluated and compared with their counterparts using only gray-level thresholding techniques.

Several images were thresholded shown in Figs. 2-4 that demonstrated that proposed RGB/Otsu and RGB/JRE thresholding techniques performed better than their gray-level counterparts. Overall, it was difficult to determine which method between the RGB/Otsu and the RGB/JRE performed better. Fig. 2 shows that the RGB/JRE performed better in the sense that the former produced a result similar to that produced by the latter with few number of colors. Fig. 3 shows opposite results where the RGB/Otsu performed better than RGB/JRE in the sense of detecting clear text characters "DENISE OLIVER". Fig. 4 provides an example where no significant preference can be made between RGB/Otsu and RGB/JRE.

IV. CONCLUSIONS

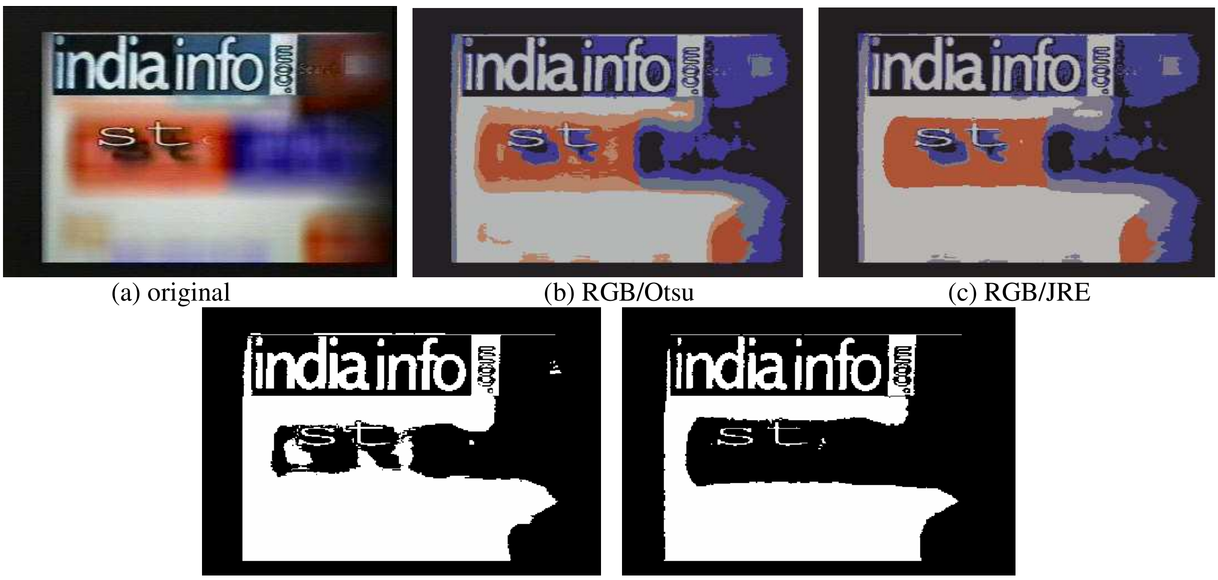
In this paper, an RGB thresholding approach is developed for text detection in color images. Several contributions have been made. One is that it extends single gray-level techniques to multi-gray level thresholding for color images. Another is that the designed clustering process in the RGB thresholding is a new approach that uses a within-class and between-class criterion to fuse different threshold values obtained from the RGB color domain. A third contribution is that the clustering process can be implemented in conjunction with any gray-level thresholding technique to adapt to various applications. Several images were thresholded which demonstrated that the proposed color segmentation technique performed better than existing grayscale-only methods. The RGB/Otsu method was able to detect less clean text but had more background visible. The RGB/JRE method created clearer text but was less able to successfully remove much of the background.

REFERENCES

[1] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. System Man Cybernetics*, Vol. SMC-9, No. 1, pp. 62-66, 1979.

[2] R.O. Duda, R.O. and P.E. Hart, *Pattern Classification and Scene Analysis*, John & Wiley Interscience, N.Y., 1973.

[3] Y. Du, C.-I Chang and P. Thouin, "Thresholding Video Images for Text Detection", *IEEE Conference on Pattern Recognition (ICPR)* 2002, Vol. III, pp.919-922.



(a) original (b) RGB/Otsu (c) RGB/JRE

(d) Otsu's method (e) JRE method

Figure 2. RGB/JRE does better than RGB/Otsu

(a) original

(b) RGB/Otsu

(c) RGB/JRE

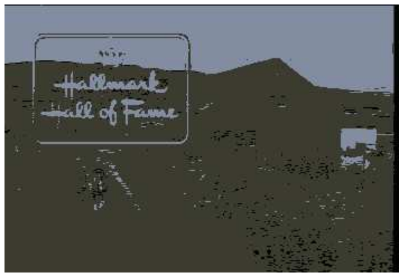
(d) Otsu's method

(e) JRE method

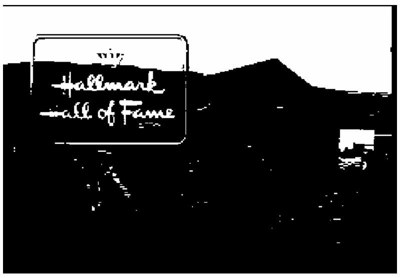
Figure 3. RGB/Otsu does better than RGB/JRE

(a) original

(b) RGB/Otsu



(c) RGB/JRE



(d) Otsu's method

(e) JRE method

Figure 4. No preference can be made between RGB/Otsu and RGB/JRE