Fast and Efficient Method for Fire Detection Using Image Processing

Turgay Celik

Conventional fire detection systems use physical sensors to detect fire. Chemical properties of particles in the air are acquired by sensors and are used by conventional fire detection systems to raise an alarm. However, this can also cause false alarms; for example, a person smoking in a room may trigger a typical fire alarm system. In order to manage false alarms of conventional fire detection systems, a computer vision-based fire detection algorithm is proposed in this paper. The proposed fire detection algorithm consists of two main parts: fire color modeling and motion detection. The algorithm can be used in parallel with conventional fire detection systems to reduce false alarms. It can also be deployed as a stand-alone system to detect fire by using video frames acquired through a video acquisition device. A novel fire color model is developed in CIE $L*a*b*$ color space to identify fire pixels. The proposed fire color model is tested with ten diverse video sequences including different types of fire. The experimental results are quite encouraging in terms of correctly classifying fire pixels according to color information only. The overall fire detection system's performance is tested over a benchmark fire video database, and its performance is compared with the state-of-the-art fire detection method.

Keywords: Fire detection, image processing, video processing, color modeling, motion detection, image segmentation.

I. Introduction

Fire detection systems are among the most important components in surveillance systems used to monitor buildings and the environment. As part of an early warning mechanism, it is preferable that the system has the capacity to report the earliest stage of a fire. Currently, almost all fire detection systems use built-in sensors that depend primarily on the reliability and the positional distribution of the sensors. It is essential that these sensors are distributed densely for a high-precision fire detection system. In a sensor-based fire detection system for an outdoor environment, coverage of large areas is impractical due to the necessity of a regular distribution of sensors in close proximity.

Due to rapid developments in digital camera technology and video processing techniques, there is a major trend to replace conventional fire detection methods with computer vision-based systems. In general, computer vision-based fire detection systems employ three major stages: fire pixel classification, moving object segmentation, and analysis of the candidate regions. This analysis is usually based on two figures: the shape of the region and the temporal changes of the region. The fire detection performance depends critically on the effectiveness of the fire pixel classifier which generates seed areas that the rest of the system will exercise. The fire pixel classifier is thus required to have a very high detection rate and preferably, a low false alarm rate. There exist few algorithms which directly deal with the fire pixel classification in the literature.

The fire pixel classification can be considered both in grayscale and color video sequences. Most of the work on fire pixel classification in color video sequences is rule-based. Chen and others [1] used raw R, G, and B color information and developed a set of rules to classify the fire pixels. Instead of

using the rule-based color model as in Chen and others, Toreyin and others [2] used a mixture of Gaussian models in RGB space which is obtained from a training set of fire pixels. In a recent paper, the authors employed Chen's fire pixel classification method along with motion information and Markov field modeling of the fire flicker process [3]. Celik and others [4] used background subtraction to segment changed foreground objects and three rules of RGB color components to detect fire pixels. The overall system can result in very high false alarm rates when intensity changes are considered, and it is very sensitive to the tuning parameters employed in background subtraction.

Celik and others [5] used normalized RGB (*rgb*) values for a generic color model for fire. The normalized RGB is proposed in order to alleviate the effects of changing illumination. The generic model is obtained using statistical analysis carried out in *r-g*, *r-b*, and *g-b* color planes. Due to the distribution of the sample fire pixels in each plane, three lines are used to specify a triangular region representing the region of interest for the fire pixels. Therefore, triangular regions in respective *r-g*, *r-b*, and *g-b* planes are used to classify a pixel. A pixel is declared to be a fire pixel if it falls into three of the triangular regions in *r-g*, *r-b*, and *g-b* planes. Krull and others [6] used low-cost CCD cameras to detect fires in the cargo bay of long range passenger aircraft. This method uses statistical features based on grayscale video frames, which include mean pixel intensity, standard deviation, and second-order moments as well as non-image features, such as humidity and temperature to detect fire in the cargo compartment. The system is commercially used in parallel with standard smoke detectors to reduce the number of false alarms caused by the smoke detectors, and it also provides visual inspection capability which helps the aircraft crew confirm the presence or absence of fire. However, the statistical image features are not considered to be used as part of a standalone fire detection system.

Marbach and others [7] used the *YUV* color model for the representation of video data, where time derivative of luminance component *Y* was used to declare the candidate fire pixels, and the chrominance components *U* and *V* were used to classify whether or not the candidate pixels were in the fire sector. In addition to luminance and chrominance, they also incorporated motion into their work. They report that their algorithm detects less than one false alarm per week; however, there is no mention of the number of tests conducted. Horng and others [8] used the *HSI* color model to roughly segment the fire-like regions for brighter and darker environments. The initial segmentation is followed by removing the lower intensity and the lower saturation pixels to eliminate spurious fire-like regions, such as smoke. A metric based on binary contour difference images is also introduced to measure the

degree of burning of fire into classes, such as 'no fire,' 'small,' 'medium,' and 'big' fires. They report a 96.94% detection rate including false positives and false negatives for their algorithms. However, there is no attempt to reduce the false positives and false negatives by changing their threshold values. Phillips and others [9] proposed a sophisticated method for recognizing fires in color video. They used both motion and color information. However, their methods require a look-up table generation at the beginning of system start-up. This adds the drawback of dependency on an operator to detect fire in video sequences. Moreover, the approach is too complicated to process in real-time.

A good color model for fire modeling and robust moving pixel segmentation are essential because of their critical role in computer vision-based fire detection systems. In this paper, we propose an algorithm that models the fire pixels using the CIE *L*a*b** color space. The motivation for using CIE *L*a*b** color space is because it is perceptually uniform color space, thus making it possible to represent color information of fire better than other color spaces. The moving pixels are detected by applying a background subtraction algorithm together with a frame differencing algorithm on the frame buffer filled with consecutive frames of input video to separate the moving pixels from non-moving pixels. The moving pixels which are also detected as a fire pixel are further analyzed in consecutive frames to raise a fire alarm.

This paper is organized as follows. Section II presents the essentials of color modeling for fire detection and introduces the different types of concepts to reduce the false alarm rate. The moving pixel detection algorithm is also presented in section II. Section III provides experimental results and comparisons with the state-of-the-art fire detection algorithm. The paper concludes in section IV.

II. Fire Detection

This section covers the details of the fire detection algorithm. Figure 1 shows the flow chart of the proposed algorithm for fire detection in a video. It is assumed that the image acquisition device produces its output in RGB format. The algorithm consists of three main stages: fire pixel detection using color information, detecting moving pixels, and analyzing dynamics of moving fire pixels in consecutive frames. In following, each part is described in detail.

1. RGB to CIE *L*a*b** Color Space Conversion

The first stage in our algorithm is the conversion from RGB to CIE *L*a*b** color space. Most of the existing CCTV video cameras provide output in RGB color space, but there are also
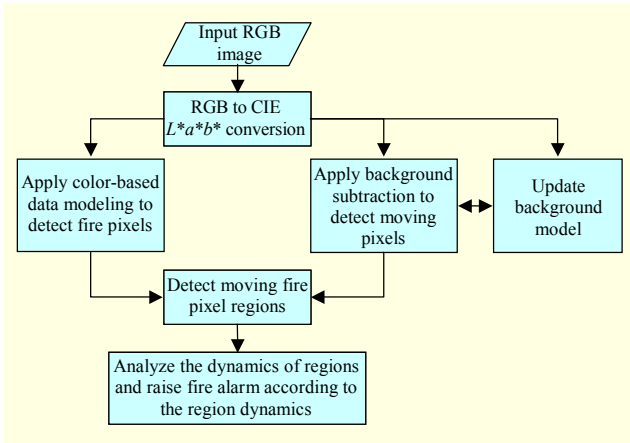
Fig. 1. Flow chart of proposed algorithm for fire detection in image sequences.

other color spaces used for data output representation. The conversion from any color space representation to CIE $L^*a^*b^*$ color space is straightforward [10]. Given RGB data, the conversion to CIE $L^*a^*b^*$ color space is formulated as follows [10]:

$$
\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix},
$$

$$
L^* = \begin{cases} 116 \times (Y/Y_n)^{1/3} - 16, & \text{if } (Y/Y_n) > 0.008856, \\ 903.3 \times (Y/Y_n), & \text{otherwise,} \end{cases}
$$

$$
a^* = 500 \times \left(f(X/X_n) - f(Y/Y_n) \right),
$$

$$
b^* = 200 \times \left(f(Y/Y_n) - f(Z/Z_n) \right),
$$

$$
f(t) = \begin{cases} t^{1/3}, & \text{if } t > 0.008856, \\ 7.787 \times t + 16/116, & \text{otherwise,} \end{cases} \tag{1}
$$

where X_n, Y_n, and Z_n are the tri-stimulus values of the reference color white. The data range of RGB color channels is between 0 and 255 for 8-bit data representation. Meanwhile, the data ranges of L^*, a^*, and b^* components are [0, 100], [–110, 110], and [–110, 110], respectively.

2. Color Modeling for Fire Detection

A fire in an image can be described by using its visual properties. These visual properties can be expressed using simple mathematical formulations. In Fig. 2, we show sample images which contain fire and their CIE $L^*a^*b^*$ color channels (L^*, a^*, b^*). Figure 2 gives some clues about the way CIE $L^*a^*b^*$ color channel values characterize fire pixels. Using such visual properties, we develop rules to detect fire using CIE $L^*a^*b^*$ color space.

The range of fire color can be defined as an interval of color values between red and yellow. Since the color of fire is generally close to red and has high illumination, we can use this property to define measures to detect the existence of fire in an image. For a given image in CIE $L^*a^*b^*$ color space, the following statistical measures for each color channel are defined as

$$
L^*_m = \frac{1}{N} \sum_x \sum_y L^*(x, y),
$$

$$
a^*_m = \frac{1}{N} \sum_x \sum_y a^*(x, y),
$$

$$
b^*_m = \frac{1}{N} \sum_x \sum_y b^*(x, y), \tag{2}
$$

where L^*_m, a^*_m, and b^*_m are a collection of average values of the L^*, a^*, and b^* color channels, respectively; N is the total number of pixels in the image; and (x, y) is spatial pixel location in an imaging grid. The numeric color responses L^*, a^*, and b^* are normalized to [0, 1]. It is assumed that the fire in an image has the brightest image region and is near to the color red. Thus, the following rules can be used to define a fire pixel:

$$
R1(x, y) = \begin{cases} 1, & \text{if } L^*(x, y) \ge L^*_m, \\ 0, & \text{otherwise,} \end{cases} \tag{3}
$$

$$
R2(x, y) = \begin{cases} 1, & \text{if } a^*(x, y) \ge a^*_m, \\ 0, & \text{otherwise,} \end{cases} \tag{4}
$$

$$
R3(x, y) = \begin{cases} 1, & \text{if } b^*(x, y) \ge b^*_m, \\ 0, & \text{otherwise,} \end{cases} \tag{5}
$$

$$
R4(x, y) = \begin{cases} 1, & \text{if } b^*(x, y) \ge a^*(x, y), \\ 0, & \text{otherwise,} \end{cases} \tag{6}
$$

where $R1$, $R2$, $R3$, and $R4$ are binary images which represent the existence of fire in a spatial pixel location (x, y) by 1 and the non-existence of fire by 0. $R1(x, y)$, $R2(x, y)$, and $R3(x, y)$ are calculated from global properties of the input image. $R4(x, y)$ represents the color information of fire; for example, fire has a reddish color. Figure 3 shows sample images from Fig. 2(a), and binary images created using (3)-(6). Figure 3(f) shows a combination of these binary images with the binary AND operator. Figure 3(g) displays the segmented fire image.

In order to find the correlation between L^*, a^*, and b^* values of fire pixels, the following strategy was applied. A set of 500 RGB images was collected from the Internet. Then, each image was manually segmented to identify all fire regions. Segmented fire regions are converted to L^*, a^*, and b^* color space. A histogram of fire pixels is created for each of the 3
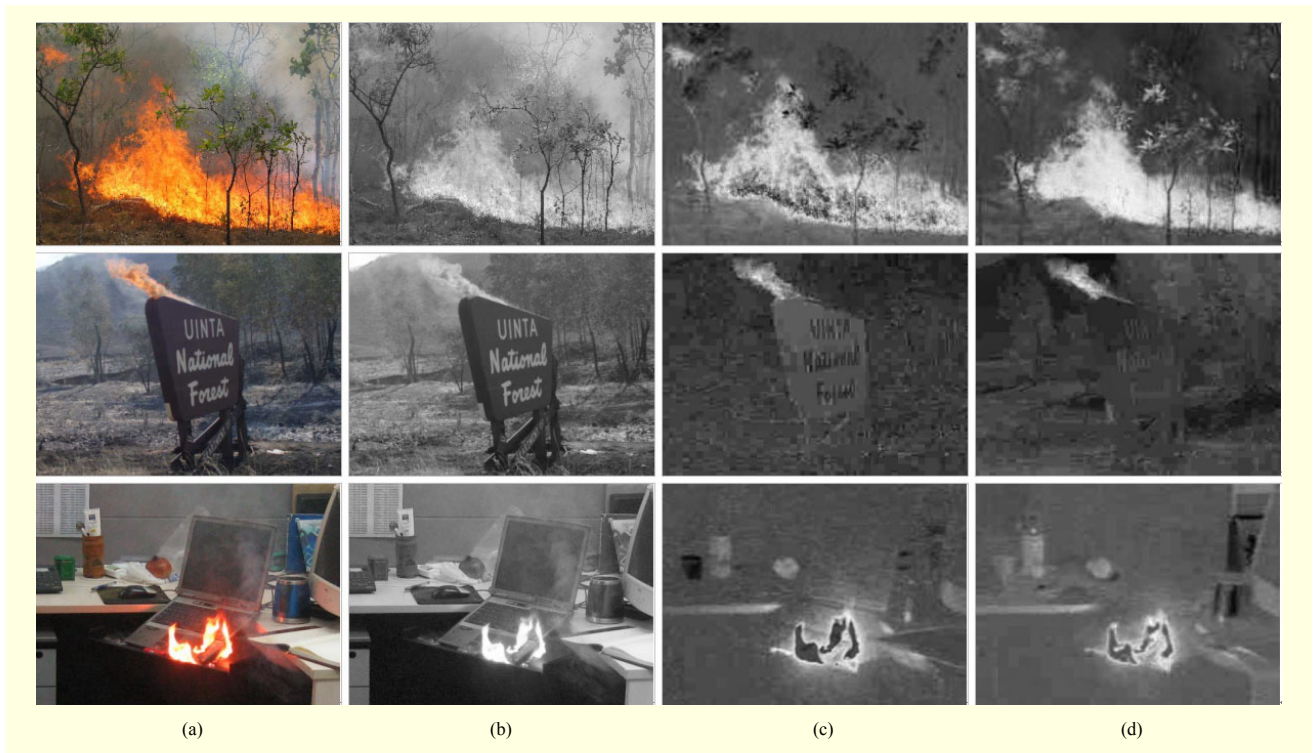
Fig. 2. Sample RGB images containing fire and their CIE $L^*a^*b^*$ color channels: (a) RGB image, (b) L^* color channel, (c) a^* color channel, and (d) b^* color channel. For visualization purposes, responses in different color channels are normalized into interval [0, 1].
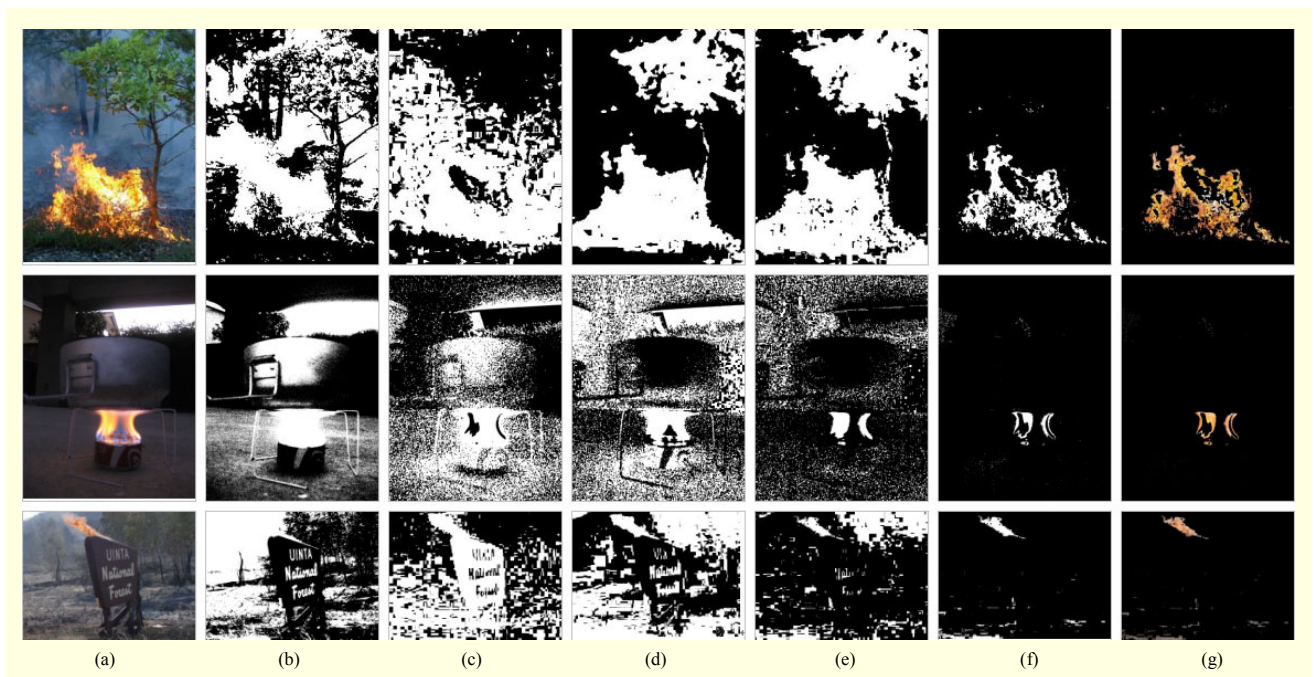

Fig. 3. Applying (3)-(6) to input images: (a) original RGB image, (b) binary image using (3), (c) binary image using (4), (d) binary image using (5), (e) binary image using (6), (f) combining results of (b)-(e) by binary AND operator, and (g) segmented fire region.

different color planes, that is, $(L^*\text{-}a^*)$, $(L^*\text{-}b^*)$, and $(a^*\text{-}b^*)$. Figure 4 shows the histograms of three different color planes

where L^*, a^*, and b^* channels are quantized into 24 levels, and 6,223,467 pixels are used to create each histogram. The
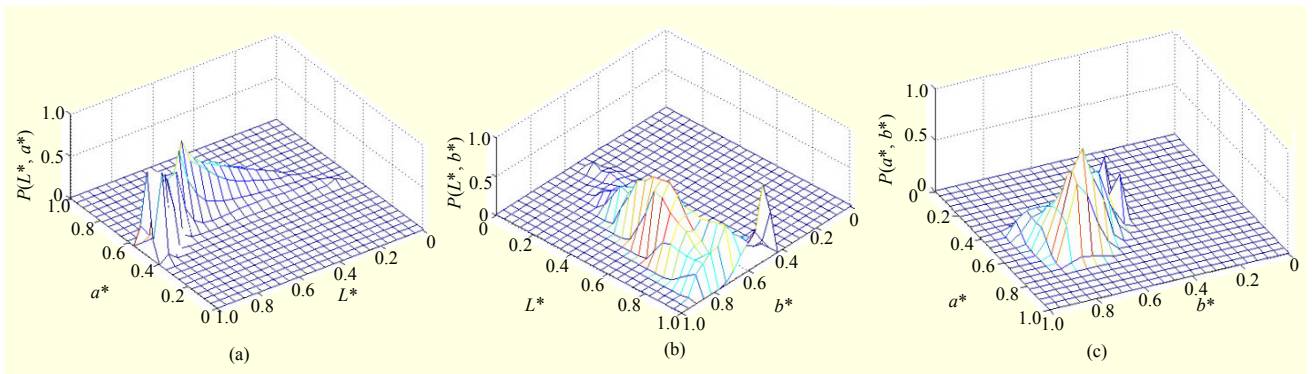
Fig. 4. Distributions of labeled fire pixels in fire images: (a) (L^*, a^*) color channels, (b) (L^*, b^*) color channels, and (c) (a^*, b^*) color channels.

number of quantization levels can be changed, but through experimentation, 24 levels were found to give satisfactory results. A look-up table is created for each pair of 24 quantized levels to keep track of the likelihood that any pair of L^*, a^*, and b^* belongs to a fire. It is clear from Figs. 4(a), 4(b), and 4(c) that a fire can be defined by the combination of three histograms. Given the L^*, a^*, and b^* color values at spatial location (x, y), the likelihood that L^*, a^*, and b^* belong to a fire $P(L^*, a^*, b^*)$ is defined as

$$P\left(L^*, a^*, b^*\right) = P\left(L^*, a^*\right) P\left(L^*, b^*\right) P\left(a^*, b^*\right), \quad (7)$$

where $P(L^*, a^*)$, $P(L^*, b^*)$, and $P(a^*, b^*)$ are the likelihoods that (L^*, a^*), (L^*, b^*), and (a^*, b^*) belong to a fire, respectively. The likelihood of being fire as defined by (7) can be used to detect a fire pixel by using simple thresholding:

$$R5(x, y) = \begin{cases} 1, & \text{if } P\left(L^*(x, y), a^*(x, y), b^*(x, y)\right) \geq \alpha, \\ 0, & \text{otherwise,} \end{cases} \quad (8)$$

where α is a threshold value. Figure 5 shows input RGB images, corresponding likelihood images $P(L^*, a^*, b^*)$ resulted from (7), and corresponding $R5$ images resulted from (8) for $\alpha = 0.005$. The pixel value $P(L^*(x, y), a^*(x, y), b^*(x, y))$ of likelihood image $P(L^*, a^*, b^*)$ is a measure in the range of [0, 1] for which a higher value of $P(L^*(x, y), a^*(x, y), b^*(x, y))$ means that there is a higher likelihood that the corresponding pixel belongs to a fire.

The optimum value of α can be estimated using receiver operating characteristic (ROC) analysis. The labeled image set is used in estimating the value of α along with the following evaluation criterion. For each value of α, the likelihood in (8) is calculated and binarized for each image in the dataset. Using the ground truth regions which were manually labeled as a fire in the training images, the number of correct detections and false detections are calculated for the whole image set. The correct detection is defined as any pixel detected as a fire pixel using (8) which is also manually labeled as a fire pixel in the
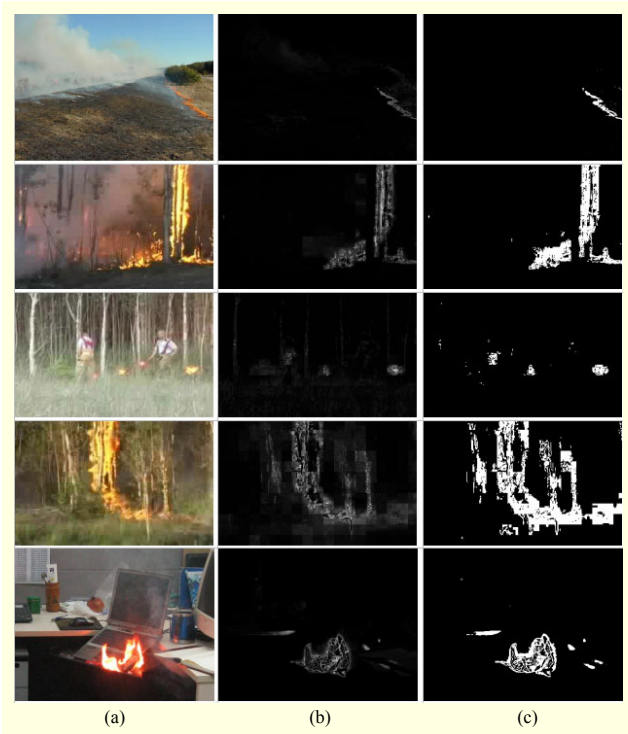

Fig. 5. Calculating $P(L^*, a^*, b^*)$ and thresholding it with $\alpha = 0.005$: (a) RGB input image which contains fire, (b) corresponding likelihood image $P(L^*, a^*, b^*)$ computed according to (7), and (c) thresholded $P(L^*, a^*, b^*)$ computed according to (8).

original image. Similarly, false detection is defined as any pixel detected as a fire pixel using (8) but is not in the manually labeled fire regions. For each value of α, the average rate of correct detection and false detection is evaluated on a training image set and used in the ROC curve. Figure 6 shows the ROC curve. Using the ROC curve, a threshold value for α can be easily selected for the fire detection algorithm with a pre-defined correct detection versus false detection rates.

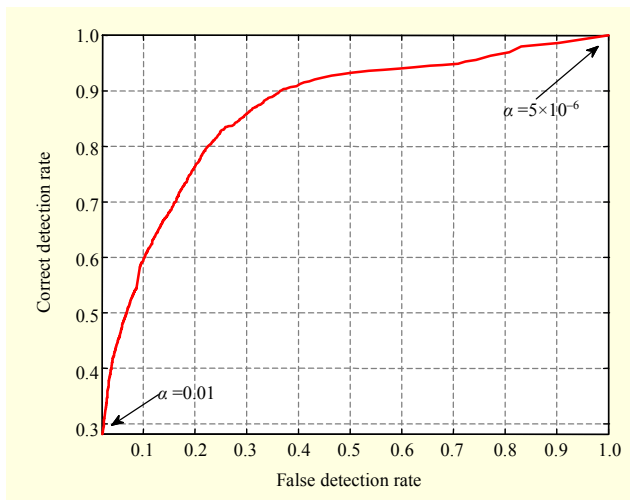Different values of α result in different system performances.

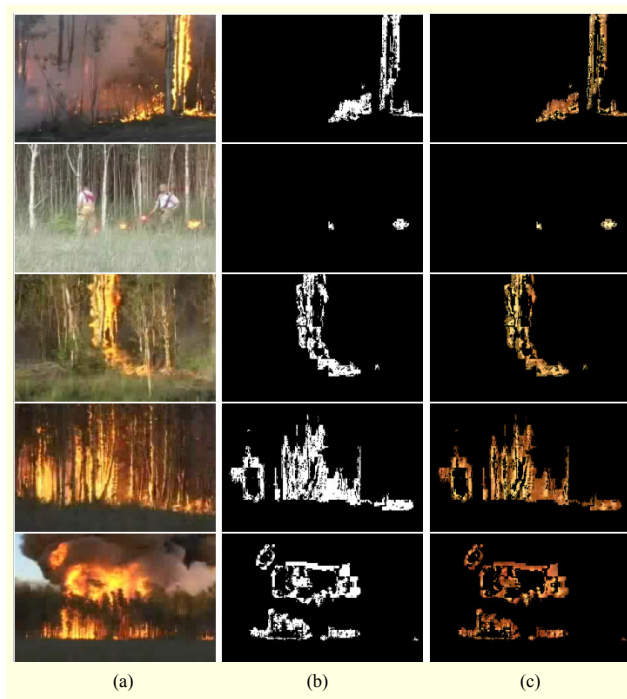Fig. 6. ROC curve for variable α ranging in [0, 0.01].

Fig. 7. Sample fire segmentations using (9): (a) original image, (b) fire map using (9), and (c) segmented fire image using original image and (9).

In our implementation, we chose α as the value which gives more than a 90% correct detection rate. The very first value of α which satisfies this condition is $\alpha = 0.00016$. However, it also produces a 36.5% false detection rate as shown in Fig. 6. The smaller value of α makes the algorithm produce a higher correct detection rate but also produces a higher false detection rate, and vice versa. The value of α can be changed at any time to adjust for higher correct detection or lower false detection rates.

Using (3)-(6) and (8), a final fire pixel detection equation can be defined as

$$F(x, y) = \begin{cases} 1, & \text{if } \sum_{i=1}^{5} R_i(x, y) = 5, \\ 0, & \text{otherwise,} \end{cases} \quad (9)$$

where $F(x, y)$ is the final decision on whether a pixel located at spatial location (x, y) results from fire or not. Equation (9) means that if inequalities defined in (3)-(6) and (8) give 1 as their output for spatial location (x, y), then there is a fire in that spatial location.

Figure 7 shows the performance of fire segmentation using (9) on sample RGB images. Figure 7(a) is the original image, Fig. 7(b) is the result of applying (9), and Fig. 7(c) is the segmented image using binary map in Fig. 7(b). It is clear that the proposed fire color model can adequately detect fire pixels under different conditions. For instance, the illumination shows high diversity in between input images (see Fig. 7(a)), and the proposed fire color model can still detect fire regions.

3. Moving Pixel Detection

In moving pixel detection, it is assumed that the video camera is stable, that is, the camera is still, and there is no movement in spatial location of the video camera. There are three main parts in moving pixel detection: frame/background subtraction, background registration, and moving pixel detection.

The first step is to compute the binary frame difference map by thresholding the difference between two consecutive input frames. At the same time, the binary background difference map is generated by comparing the current input frame with the background frame stored in the background buffer. The binary background difference map is used as primary information for moving pixel detection. In the second step, according to the frame difference map of past several frames, pixels which are not moving for a long time are considered as reliable background in the background registration. This step maintains an updated background buffer as well as a background registration map indicating whether the background information of a pixel is available or not. In the third step, the binary background difference map and the binary frame difference map are used together to create the binary moving pixel map. If the background registration map indicates that the background information of a pixel is available, the background difference map is used as the initial binary moving pixel map. Otherwise, the value in the binary frame difference map is copied to binary moving pixel map. The intensity channel L^* is used in moving pixel detection.

The frame difference between the current frame $L^*(x, y, t)$ at

time t and the previous frame $L^*(x, y, t-1)$ at time $t-1$ is computed and thresholded to create a binary frame difference map, FD, at time t, that is,

$$FD(x,y,t) = \begin{cases} 1, & \text{if } \left| L^*(x,y,t) - L^*(x,y,t-1) \right| \geq T_{FD}, \\ 0, & \text{otherwise,} \end{cases} \quad (10)$$

where T_{FD} is a threshold value.

Similar to the FD, the background difference is applied between the current frame and the background image to generate a binary background difference map, BD, that is,

$$BD(x,y,t) = \begin{cases} 1, & \text{if } \left| L^*(x,y,t) - BG(x,y,t-1) \right| \geq T_{BD}, \\ 0, & \text{otherwise,} \end{cases} \quad (11)$$

where $BG(x, y, t-1)$ is background image pixel value at spatial location (x, y) at time $t-1$, and T_{BD} is a threshold value.

The values of T_{FD} and T_{BD} should be selected carefully to generate adequate binary maps. The fixed-threshold setting provides a computationally inexpensive solution for the threshold selection problem. However, it generally suffers from changes in illumination as well as the fact that noise inherently exists in image acquisition systems. In order to overcome these drawbacks, a dynamic threshold selection method is developed.

Let DI be the difference image representing either interframe difference or background difference image. Using the difference image DI, the dynamic threshold value $\tau(DI)$ is computed as follows:

$$\tau(DI) = \begin{cases} \mu(DI) + \sigma(DI), & \text{if } \mu(DI) + \sigma(DI) \geq 10, \\ 10, & \text{otherwise,} \end{cases} \quad (12)$$

where $\mu(DI)$ and $\sigma(DI)$ are the mean and the standard deviation of the difference image pixels, respectively, that is,

$$\mu(DI) = \frac{\sum_x \sum_y DI(x,y)}{N},$$

$$\sigma(DI) = \sqrt{\frac{\sum_x \sum_y \left(DI(x,y) - \mu(DI) \right)^2}{N}}.$$

Equation (12) automatically determines a dynamic threshold value according to mean and standard deviation of pixel values in the difference image DI. In order to reduce false detections resulting from the lower values of dynamic threshold which happens in the case of no motion occurring in between consecutive video frames, a lower limit value of 10 is used. The lower limit value can be tuned according to the system dynamics or other external effects under consideration. For example, if the level of noise is high, then the lower limit value should be selected high enough to reduce the false alarm rate, and vice versa. Using (12), the threshold values T_{FD} and T_{BD}

are computed as $T_{FD} = \tau(F_{t,\,t-1})$ and $T_{BD} = \tau(B_{t,\,t-1})$, where $F_{t,t-1} = \left| L^*(x, y, t) - L^*(x, y, t-1) \right|$ is the interframe difference, and $B_{t,t-1} = \left| L^*(x, y, t) - BG(x, y, t-1) \right|$ is the background difference.

In order to construct reliable background information from the video sequence, a background registration step is applied using the binary frame difference map and the binary background difference map. Using the FD, pixels not moving for a long time are considered as reliable background pixels. For each pixel (x, y), a stationary index (SI) is kept to count how long it is detected as non-moving pixel, that is,

$$SI(x,y,t) = \begin{cases} SI(x,y,t-1)+1, & \text{if, } FD(x,y,t) = 0, \\ 0, & \text{otherwise.} \end{cases} \quad (13)$$

Using the SI, the background image BG is updated as follows:

$$BG(x,y,t) = \begin{cases} L^*(x,y,t), & \text{if } SI(x,y,t) = T_{SI}, \\ BG(x,y,t-1), & \text{otherwise,} \end{cases} \quad (14)$$

where T_{SI} is a threshold which identifies how long it takes for each pixel to be considered as a non-moving pixel. The value of threshold T_{SI} is equal to the number of frames that can be processed by the system per second. The initial values of SI and BG are all set to 0.

The FD and the BD are considered together to compute a resultant binary moving pixel map M according to

$$M(x,y) = FD(x,y,t) \oplus BD(x,y,t), \quad (15)$$

where $\oplus$ is binary OR operator.

4. Analyzing Fire Regions in Consecutive Frames

The pixels which are detected as both fire using (9) and moving using (15) are employed to detect a candidate fire pixel (CF) as

$$CF(x,y,t) = F(x,y) \otimes M(x,y), \quad (16)$$

where $\otimes$ is binary AND operator. The connected components of binary image CF are further analyzed in consecutive frames. The connected components of a size one pixel are labeled as noise and are not considered for further analysis. Let $O(t)$ and $NO(t)$ be the set of all pixels which compose one of the connected components of $CF(t)$ and the number of pixels of the connected component $O(t)$ at time t, respectively. The connected component $O(t)$ is tracked in time to make a further decision by considering the behavior of fire. A fire grows spatially at its early stage and shows flickering behavior. In order to quantify this behavior, $O(t)$ is observed in consecutive frames. The counter $CGO(t)$ is generated for $O(t)$ with

$$CGO(t) = \begin{cases} CGO(t-1)+1, & \text{if } NO(t) \geq NO(t-1), \\ CGO(t-1), & \text{otherwise} \end{cases} \quad (17)$$

to count the number of times that the connected component $O(t)$ at time t has higher number of pixels than that of the same connected component $O(t-1)$ at time $t-1$. The number of detected fire pixels at the initial stage of fire is less than the number of detected fire pixels at later stages when fire grows spatially. The counter $CGO(t)$ is designed by considering this behavior of fire. The counter increases its value at time t when the number of detected fire pixels is higher than that of detected pixels at time $t-1$. However, because of the flickering behavior, it is possible to detect less fire pixels at time t with respect to time $t-1$. In this case, (16) does not update its value. The initial value of the counter in (16) is set to 0 and updated in time.

The value of counter $CGO(t)$ is used to measure the temporal behavior of fire at early stages. That is, at the early stages of a fire, the fire should be spatially growing, hence the number of detected fire pixels should be increasing. Let FPS be the frames per second that our system can process. Using (17), we define a metric $D(t)$ to decide if the connected $O(t)$ is a fire region or not by

$$D(t) = \frac{CGO(t) - CGO(t - FPS + 1)}{FPS}. \quad (18)$$

The value of $D(t)$ is near to 1 when the fire is spatially growing, which happens at the early stages of fire. On the other hand, the value of $D(t)$ approaches 0 when the fire region is not having considerable spatial motion. Each connected component is tracked in consecutive frames, and the value of $D(t)$ is computed. It is empirically found that $D(t) \geq 0.4$ provides a go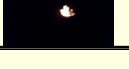od tradeoff between correct fire region identification and false alarm reduction, and a fire alarm is raised when $D(t) \geq 0.4$. Note that $D(t)$ starts to produce its values after FPS frames pass from the first time the corresponding connected component starts being processed. The metric $D(t)$ automatically eliminates the false alarms caused by fire-like colored moving but not spatially growing objects. For such objects, the value of $D(t)$ is much smaller than 0.4. The above procedure is applied to each connected component of CFs.

III. Tests and System Performance

1. Tests on Proposed Fire Color Model

First, the performance of the proposed fire color model is tested. The color model is tested on different video sequences for a variety of environmental conditions, for example, daytime,

Table 1. Experimental results of fire detection using color information.

Video	F_t	F_f	F_c	FP	FN	R_d(%)	Sample picture
1	432	432	432	0	0	100.00	
2	408	408	408	0	0	100.00	
3	345	345	345	0	0	100.00	
4	332	332	332	0	0	100.00	
5	407	407	407	0	0	100.00	
6	362	362	362	0	0	100.00	
7	2,237	1,358	2,219	11	7	99.20	
8	2,736	2,064	2,734	0	2	99.93	
9	2,294	1,403	2,287	3	4	99.69	
10	3,461	2,351	3,460	1	0	99.97	
Total	13,014	9,462	12,986	15	13	99.88	

nighttime, indoor, and outdoor. Equation (9) is applied to each frame of each video. The fire alarm is raised if the number of connected fire pixels detected is greater than five. The experimental results of the proposed fire detection method are shown in Table 1. F_t is the number of frames of a video sequence, and F_f is the number of frames containing fire in a video sequence. F_c is the number of frames (including fire and non-fire frames) that correctly classify fire pixels by the proposed algorithm. FP and FN refer to the number of frames that are classified as false positive and false negative, respectively. False positive means that the system recognizes fire in an image frame when there is no fire. Similarly, false negative means that the system does not detect fire in an image frame when there is indeed fire. The detection rate, R_d, of a video is defined as

$$R_d = \frac{F_c}{F_t}. \quad (19)$$

The average detection rate that can be achieved is more than 99.88% with the test sequences of samples shown in Table 1. The false negative detection rates are due to very small fire regions on the initial combustion in some of the video sequences. False positives are mainly caused from the reflection of fire onto the sides of the metal container. Figure 8

Fig. 8. Sample fire detection results produced by using only proposed fire-color model on different types of videos as shown in different rows: columns 1, 3, and 5 are RGB frames from input video sequences and columns 2, 4, and 6 are segmented fire regions according to the proposed fire color model.

shows sample fire detection results using only color information on different video sequences. It is clear from Table 1 and Fig. 8 that under different conditions, the proposed fire color model can detect fire adequately, which is very important for a robust fire detection system.

2. Tests on Proposed Fired Detection System

The performance of the proposed fire detection system was tested on video sequences which consist of diverse conditions. We implemented the proposed fire detection system on Intel Core 2 Duo 2.0 GHz CPU. The operating system was Windows Vista. The system performed minimum 30 FPS for input video sequence of frame size 320×240 pixels. The frame rate was changeable depending on the size of the fire. To validate the effectiveness of the proposed fire detection system, the detection performance of the proposed system was compared with that of the Toreyin algorithm [3].

The test was performed using 9 online video sequences [3] downloaded from http://signal.ee.bilkent.edu.tr/VisiFire. The properties of video sequences are given in Table 2. The frame rate of the video data varied from 15 Hz to 30 Hz, and the size of the input image was 320×240 pixels. The movie sequences Movie 1, Movie 2, Movie 3, and Movie 4 are outdoor fire videos, and Movie 8 includes a car accident in a tunnel without fire. We also considered fire-colored moving persons and a truck (Movie 5, Movie 6, Movie 7, and Movie 9) to check the performance of the proposed system in cases of false alarms caused by non-fire objects.

The fire detection test results on nine video sequences are

Table 2. Properties of video sequences.

Name	F_t	Description
Movie 1	510	Man in fire-colored shirt behind fire
Movie 2	368	Burning tree
Movie 3	213	Fire in garden
Movie 4	190	Fire in forest
Movie 5	120	Fire-colored moving truck
Movie 6	165	Three men walking on ground
Movie 7	412	Three men walking in hallway
Movie 8	393	Accident in tunnel
Movie 9	394	Dancing man in fire-colored shirt

shown in Table 3. Overall, the proposed method performs better than the method of Toreyin and others [3]. In particular, the Toreyin method gave several false alarms for Movie 3 and Movie 4 due to light reflection from a fire onto white smoke being confused with a burning fire. However, the proposed algorithm was able to remove these false positives using a better color model which separates luminance from the chrominance to make a better decision on fire color and a better moving region analysis in consecutive frames. In the case of Movie 1, which contains relatively small-sized fires, our system shows a relatively lower missing rate compared to the Toreyin method. The detection rate on four videos including the truck and fire-colored moving objects shows 100% on both methods.

Table 3. Performance comparisons of different fire detection methods on video sequences detailed in Table 2.

| Name	Method of [3]			Proposed method		
	FP (%)	FN (%)	R_d (%)	FP (%)	FN (%)	R_d (%)
Movie 1	0.0	66.0	34.0	6.0	40.8	53.2
Movie 2	4.9	7.6	87.5	1.0	19.8	79.2
Movie 3	44.1	4.7	51.2	0.5	41.2	58.3
Movie 4	10.0	16.3	73.7	1.1	1.4	97.5
Movie 5	0.0	0.0	100	0.0	0.0	100
Movie 6	0.0	0.0	100	0.0	0.0	100
Movie 7	0.0	0.0	100	0.0	0.0	100
Movie 8	0.0	0.0	100	0.0	0.0	100
Movie 9	0.0	0.0	100	1.5	0.2	98.3
Average	6.6	10.5	82.9	1.1	11.5	87.4

IV. Conclusion

In this paper, a new image-based real-time fire detection method was proposed which is based on computer vision techniques. The proposed method consists of three main stages: fire pixel detection using color, moving pixel detection, and analyzing fire-colored moving pixels in consecutive frames to raise an alarm. The proposed fire color model achieves a detection rate of 99.88% on the ten tested video sequences with diverse imaging conditions. Furthermore, the experiments on benchmark fire video databases show that the proposed method achieves comparable performance with respect to the state-of-the-art fire detection method.

The performance of the proposed fire detection system can be further improved by considering smoke at early stages of fire. However, detecting smoke is a challenging task and prone to high false detections caused from fog, different lighting conditions caused by nature, and other external optical effects. Such high false detections can be resolved by analyzing every smoke-like region. However, this yields a high computational load.

The motion information of fire is also considered to characterize fire regions. The proposed system assumes that the fire will grow gradually in a spatial domain. This might not be the case in some situations. For instance, the system might not be able to detect a fire caused by a sudden explosion. In order to alleviate such cases, the proposed system will be further improved to include different scenarios. Furthermore, texture and shape information of fire regions will also be investigated to improve the system's fire detection performance.

References

[1] T. Chen, P. Wu, and Y. Chiou, "An Early Fire-Detection Method Based on Image Processing," *Proc. IEEE Int. Image Process.*, 2004, pp. 1707-1710.

[2] B.U. Toreyin, Y. Dedeoglu, and A.E. Cetin, "Flame Detection in Video Using Hidden Markov Models," *Proc. IEEE Int. Conf. Image Process.*, 2005, pp. 1230-1233, 2005.

[3] B.U. Toreyin, Y. Dedeoglu, and A.E. Cetin, "Computer Vision Based Method for Real-Time Fire and Flame Detection," *Pattern Recognition Lett.*, vol. 27, no. 1, 2006, pp. 49-58.

[4] T. Celik et al., "Fire Detection Using Statistical Color Model in Video Sequences," *J. Visual Commun. Image Representation*, vol. 18, no. 2, Apr 2007, pp. 176-185.

[5] T. Celik, H. Demirel, and H. Ozkaramanli, "Automatic Fire Detection in Video Sequences," *Proc. European Signal Process. Conf.*, Florence, Italy, Sept. 2006.

[6] W. Krüll et al., "Design and Test Methods for a Video-Based Cargo Fire Verification System for Commercial Aircraft," *Fire Safety J.*, vol. 41, no. 4, 2006, pp. 290-300.

[7] G. Marbach, M. Loepfe, and T. Brupbacher, "An Image Processing Technique for Fire Detection in Video Images," *Fire Safety J.*, vol. 41, no. 4, 2006, pp. 285-289.

[8] W.-B. Horng, J.-W. Peng, and C.-Y. Chen, "A New Image-Based Real-Time Flame Detection Method Using Color Analysis," *Proc. IEEE Networking, Sensing Control*, 2005, pp. 100-105.

[9] W. Phillips III, M. Shah, and N. da Vitoria Lobo, "Flame Recognition in Video," *Proc. 5th Workshop Appl. Computer Vision*, 2000, pp. 224-229.

[10] D. Malacara, *Color Vision and Colorimetry*, SPIE Press, 2002.
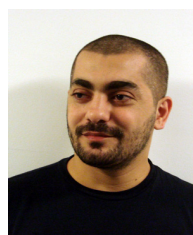
Turgay Celik received the PhD in electrical and electronic engineering from Eastern Mediterranean University, Gazimagusa, TRNC, TURKEY. He is currently a research fellow with the Department of Chemistry, National University of Singapore, Singapore, and Bioinformatics Institute, Agency for Science, Technology and Research (A*STAR), Singapore. He has produced extensive publications in various international journals and conferences. He has been acting as a reviewer for various international journals and conferences. His research interests are in the areas of biophysics, digital signal, image and video processing, pattern recognition, and artificial intelligence. These include fluorescent microscopy, digital image/video coding, wavelets and filter banks, image/video processing, content-based image indexing and retrieval, and scene analysis and recognition.