Machine Vision and Applications (2012) 23:1103–1113
DOI 10.1007/s00138-011-0369-1

ORIGINAL PAPER

Covariance matrix-based fire and flame detection method in video

Yusuf Hakan Habiboğlu · Osman Günay · A. Enis Çetin

Received: 2 November 2010 / Revised: 18 August 2011 / Accepted: 25 August 2011 / Published online: 17 September 2011

Abstract This paper proposes a video-based fire detection system which uses color, spatial and temporal information. The system divides the video into spatio-temporal blocks and uses covariance-based features extracted from these blocks to detect fire. Feature vectors take advantage of both the spatial and the temporal characteristics of flame-colored regions. The extracted features are trained and tested using a support vector machine (SVM) classifier. The system does not use a background subtraction method to segment moving regions and can be used, to some extent, with non-stationary cameras. The computationally efficient method can process 320×240 video frames at around 20 frames per second in an ordinary PC with a dual core 2.2 GHz processor. In addition, it is shown to outperform a previous method in terms of detection performance.

Keywords Fire detection · Covariance descriptors · Support vector machines

1 Introduction

Automatic fire detection in images and videos is crucial for early detection of fire. Video-based systems can detect uncontrolled fires at an early stage before they turn into catastrophic events. In this article, a new video fire detection method using the region covariance matrix approach is proposed. The main feature of the covariance method is that it can model color, spatial and temporal information in video into a compact feature vector capable of modeling flames.

There are several methods in the literature developed for fire detection from images and videos [5,6,8,12–16,18,21–25,28]. Almost all of the current methods use motion and color information for flame detection. In [18], a Gaussian-smoothed histogram is used as a color model. A fire color probability value is estimated by using several values of a fixed pixel over time. Considering the temporal variation of fire and non-fire pixels with fire color probability, a heuristic analysis is carried out. In addition a solution for the reflection problem is proposed, and a region growing algorithm is implemented for flame region estimation. After moving pixels are segmented, chromatic features are used for determining fire and smoke regions in [5]. Dynamics analysis of flames is done. According to this analysis, dynamic features of fire and smoke regions are used to determine whether it is a real flame or a flame-alias.

In another work [25], moving and fire-colored pixels are found according to a background estimation algorithm and Gaussian Mixture Models. Quasi-periodic behavior in flame boundaries and color variations are detected using temporal and spatial wavelet analysis. Irregularity of the boundary of the fire-colored region is taken into consideration at the decision step.

In [24], background estimation algorithm and a color chrominance model are used to detect both moving and fire-colored pixels. Two Markov models are proposed to separate flame flicker motion from flame-colored ordinary object motion. The same Markov model is used to evaluate spatial color variance.

Y. H. Habiboğlu · O. Günay (✉) · A. E .Çetin
Department of Electrical and Electronics Engineering,
Bilkent University, Ankara 06800, Turkey
e-mail: osman@ee.bilkent.edu.tr

Y. H. Habiboğlu
e-mail: yhakan@ee.bilkent.edu.tr

A. E. Çetin
e-mail: cetin@bilkent.edu.tr

An adaptive flame detector is implemented and trained using the weighted majority based online training found in [22]. Outputs of Markov models representing the flame and flame colored ordinary moving objects and spatial wavelet analysis of boundaries of flames are used as weak classifiers for training.

In [13], the HSI color model-based segmentation algorithm is used to find fire-colored regions. Image differencing, which is taking the absolute value of the pixel-wise difference between two consecutive frames, and the same segmentation process are both used to separate fire-colored objects from flames. Further, a method for estimating the degree of fire flames is proposed.

As described above, most fire detection systems first find the flame-colored regions using background subtraction and flame color analysis. These regions are then analyzed spatially and temporally to detect the irregular and flickering characteristics of flames. In this work, we use a different approach by combining the color, spatial and temporal domain information in the feature vectors for each spatio-temporal block using region covariance descriptors [26,27]. The blocks are obtained by dividing the flame colored regions into 3D regions that overlap in time. Classification of the features is performed only at the temporal boundaries of blocks instead of at every frame. This reduces the computational complexity of the method so that the processing frame rate of the developed system is generally above 20 frames per second (fps) when 320×240 video frames are used.

To the best of our knowledge, none of the methods proposed in the literature, including our previous works [24,25], can handle video captured by moving cameras. In [24], the flame flicker process is modeled using wavelets and Markov models. But the camera should be stationary in this approach, in order to obtain reliable results in the Markov models, which require approximately two seconds of temporal video data. The proposed covariance matrix approach does not use the background subtraction method, which requires a stationary camera for moving flame regions detection, and therefore can be used with moving cameras. This is an important advantage of the proposed method because cameras may shake in the wind or closed-circuit television (CCTV) cameras can only slowly pan an area of interest for possible fire and flames.

The rest of the paper is organized as follows: In Sect. 2, the basic building blocks of the proposed scheme are described. In Sect. 2.1, the simple flame color model is presented. Video blocks satisfying the color requirements are further analyzed by the spatio-temporal covariance descriptors described in Sect. 2.3. The test procedure and experimental results are presented in Sects. 3 and 4, respectively.

2 Covariance matrix-based detection algorithm

2.1 Chromatic color model

Chen et al. [5] suggested a chromatic model to classify pixel colors. Its implementation is simple and its computational cost is low in terms of memory and processing power. They analyzed fire-colored pixels and realized that the hue of fire-colored pixels is in the range of 0° and 60°. The RGB domain equivalent of this condition is

Condition 1 $R \geq G > B$

Since fire is a light source its pixel values must be larger than some threshold. R_T is the threshold for the red channel:

Condition 2 $R > R_T$

The last rule is about saturation. S is the saturation value of a pixel and S_T is the saturation value of this pixel when R is R_T.

Condition 3 $S > (255 - R)S_T R_T$

We modified this model by excluding condition 3 to reduce the computational cost, therefore using this new model as the first step of our algorithm. Cameras have their own color balancing and light adjustment settings, therefore it is impossible to detect fire and flames using only the color information with a high detection rate. When a relatively low threshold for the red channel is selected (in this algorithm $R_T = 110$), it is possible to detect all flame colored regions in our dataset without any false rejections. We tried other complex color models including Gaussian Mixture Models (GMMs) as in [25], but we experimentally observed that this simple chromatic model is sufficient for our purposes, because regions satisfying this simple condition are further analyzed by the spatio-temporal covariance descriptors.

2.2 Covariance descriptors

Tuzel et al. [26,27] proposed covariance descriptors and applied this method to object detection and texture classification problems. In this subsection, we first review the covariance matrix method introduced in [27] and temporally extend it to videos in the next subsection.

For a given image region the covariance matrix can be estimated as follows:

$$\widehat{\Sigma} = \frac{1}{N-1} \sum_i \sum_j \left(\Phi_{i,j} - \overline{\Phi}\right)\left(\Phi_{i,j} - \overline{\Phi}\right)^{\mathrm{T}} \text{ where}$$

$$\overline{\Phi} = \frac{1}{N} \sum_i \sum_j \Phi_{i,j} \tag{1}$$

where N is the number of pixels, $\Phi_{i,j}$ is a property vector of the pixel $P_{i,j}$ at location (i, j). Further, it may include the following parameters.

$$X(i,j) = i, \tag{2}$$
$$Y(i,j) = j, \tag{3}$$
$$R(i,j) = \text{Red}(i,j), \tag{4}$$
$$Rx(i,j) = \left|\frac{\partial \text{Red}(i,j)}{\partial i}\right|, \tag{5}$$
$$Ry(i,j) = \left|\frac{\partial \text{Red}(i,j)}{\partial j}\right|, \tag{6}$$
$$Rxx(i,j) = \left|\frac{\partial^2 \text{Red}(i,j)}{\partial i^2}\right|, \tag{7}$$
$$Ryy(i,j) = \left|\frac{\partial^2 \text{Red}(i,j)}{\partial j^2}\right|, \tag{8}$$
$$G(i,j) = \text{Green}(i,j), \tag{9}$$
$$B(i,j) = \text{Blue}(i,j), \tag{10}$$
$$I(i,j) = \text{Intensity}(i,j), \tag{11}$$
$$Ix(i,j) = \left|\frac{\partial \text{Intensity}(i,j)}{\partial i}\right|, \tag{12}$$
$$Iy(i,j) = \left|\frac{\partial \text{Intensity}(i,j)}{\partial j}\right|, \tag{13}$$
$$Ixx(i,j) = \left|\frac{\partial^2 \text{Intensity}(i,j)}{\partial i^2}\right|, \tag{14}$$

and

$$Iyy(i,j) = \left|\frac{\partial^2 \text{Intensity}(i,j)}{\partial j^2}\right|, \tag{15}$$

The vector $\Phi_{i,j}$ can include locations of pixels and intensity values of pixels, or it can only include the red, green, and blue values of pixels and their first and second derivatives. Several pixel property parameters which are defined in Eqs. 2–15 can be included in the $\Phi_{i,j}$ vector. By using combinations of these properties, different covariance descriptors for a given image region can be defined.

During the implementation of the covariance method, the first derivative of the image is computed by filtering the image with [−1 0 1] filters and second derivative is found by filtering the image with [1 −2 1] filters, respectively.

The lower or upper triangular parts of the covariance matrix Σ form the feature vector of a given image region.

2.3 Covariance descriptors for videos

2.3.1 Definition of property sets

As explained in [27], assuming a property set of a region in an image has a wide-sense stationary multivariate normal distribution, covariance descriptors provide an excellent description of the given image region. Wide-sense stationarity is a reasonable assumption for a flame-colored image region because such regions do not contain strong edges in video. In [7], dynamic textures are characterized by autoregressive moving average (ARMA) models. For the second order stationary models, they provide a sub-optimal closed form solution and show that it can be used to characterize many dynamic textures, including flames and water. Motivated by these results, we experimentally showed that covariance descriptors can be used to model the spatio-temporal characteristics of fire regions in images. To model the temporal variation and the flicker in fire flames, we introduce temporally extended covariance descriptors in this article. Covariance features are successfully used for texture classification in [26,27]. Flame detection is a dynamic texture detection problem. Therefore, we included temporal derivatives to the feature vector to describe flame regions in video. In covariance approach several image statistics are computed inside a flame region to form the descriptor. We experimentally observed that these statistical parameters take similar values in flame regions which can be classified as dynamic textures [2].

Temporally extended covariance descriptors are designed to describe spatio-temporal video blocks. Let $I(i,j,n)$ be the intensity of the (i,j)th pixel of the nth image frame of a spatio-temporal block in video let Red, $Green$, $Blue$ represent the color values of the pixels of the block. The property parameters defined in Eqs. 17–23 are used to form a covariance matrix representing spatial information. In addition to spatial parameters we introduce temporal derivatives, $\partial_t I$ and $\partial_t^2 I$, which are the first and second derivatives of intensity with respect to time, respectively. By adding these two features to the previous property set, covariance descriptors can be used to define spatio-temporal blocks in video. The temporal derivatives can also be used when the camera is non-stationary under the assumption that a camera scans a room in a very slow manner.

$$R(i,j,n) = \text{Red}(i,j,n), \tag{16}$$
$$G(i,j,n) = \text{Green}(i,j,n), \tag{17}$$
$$B(i,j,n) = \text{Blue}(i,j,n), \tag{18}$$
$$I(i,j,n) = \text{Intensity}(i,j,n), \tag{19}$$
$$Ix(i,j,n) = \left|\frac{\partial \text{Intensity}(i,j,n)}{\partial i}\right|, \tag{20}$$
$$Iy(i,j,n) = \left|\frac{\partial \text{Intensity}(i,j,n)}{\partial j}\right|, \tag{21}$$
$$Ixx(i,j,n) = \left|\frac{\partial^2 \text{Intensity}(i,j,n)}{\partial i^2}\right|, \tag{22}$$
$$Iyy(i,j,n) = \left|\frac{\partial^2 \text{Intensity}(i,j,n)}{\partial j^2}\right|, \tag{23}$$
$$\partial_t I(i,j,n) = \left|\frac{\partial \text{Intensity}(i,j,n)}{\partial n}\right|, \tag{24}$$

and

$$\partial_t^2 I(i,j,n) = \left| \frac{\partial^2 \text{Intensity}(i,j,n)}{\partial n^2} \right|. \quad (25)$$

2.3.2 Computation of covariance values in spatio-temporal blocks

In this section, details of the covariance matrix computation in video is described. We first divide the video into blocks of size $16 \times 16 \times F_{\text{rate}}$ where F_{rate} is the frame rate of the video. Computing the covariance parameters for each block of the video would be computationally inefficient. We use the simple color model to eliminate the blocks that do not contain any fire colored pixels. Therefore, only pixels corresponding to the non-zero values of the following color mask are used in the selection of blocks, using the following function:

$$\Psi(i,j,n) = \begin{cases} 1 & \text{if } \text{Red}(i,j) > \text{Green}(i,j,n) \geq \text{Blue}(i,j,n) \\ & \text{and } \text{Red}(i,j,n) > R_T \\ 0 & \text{otherwise} \end{cases} \quad (26)$$

This is not a tight condition for fire colored pixels. Almost all flame-colored or red-toned regions satisfy Eq. 26. Furthermore, in order to reduce the effect of non-fire colored pixels, only property parameters of chromatically possible fire-pixels are used in the estimation of the covariance matrix, instead of using every pixel of a given block.

A total of 10 property parameters are used for each pixel satisfying the color condition. This requires $0.5 \times (10 \times 11) = 55$ covariance computations. To further reduce the computational cost we compute the covariance values of the pixel property vectors:

$$\Phi_{\text{color}}(i,j,n) = \left[\text{Red}(i,j,n) \ \text{Green}(i,j,n) \ \text{Blue}(i,j,n) \right]^{\text{T}} \quad (27)$$

and

$$\Phi_{\text{ST}}(i,j,n) = \begin{bmatrix} I(i,j,n) \\ Ix(i,j,n) \\ Iy(i,j,n) \\ Ixx(i,j,n) \\ Iyy(i,j,n) \\ \partial_t I(i,j,n) \\ \partial_t^2 I(i,j,n) \end{bmatrix} \quad (28)$$

in a separate manner. Therefore, the property vector $\Phi_{\text{color}}(i,j,n)$ produces $0.5 \times (3 \times 4) = 6$, while the property vector $\Phi_{\text{ST}}(i,j,n)$ produces $0.5 \times (7 \times 8) = 28$ covariance values. Therefore, instead of 55, only 34 covariance parameters are used in the training and testing of the SVM. The system is tested using both 34 and 55 covariance parameters to see if there is any difference between detection performance and computational cost. Experimental results are presented in Sect. 4.

We also assume that the size of the image frames in video is the is 320×240. If not the video is scaled to 320×240 in order to run the fire detection algorithm in real-time.

3 SVM classification

3.1 Dataset

There is no publicly available dataset for fire detection experiments. We acquired our own data from various online sources and recorded some test and training video clips. These are also the video clips used in the European Community-funded Firesense project, consisting of eight research institutes and universities [9]. In this article, seven positive and ten negative video clips are used for training. Negative video clips contain flame-colored regions as shown in Fig. 3. For positive videos (video clips containing fire), only the parts of the video clips that actually contain fire are used. We also used videos from the DynTex dynamic texture database [17] to test the robustness of the system against false alarms with videos that have flame-colored moving objects that are difficult to distinguish from actual fire regions.

3.2 Training and testing

For training and testing, $16 \times 16 \times F_{\text{rate}}$ blocks are extracted from various video clips. Using larger spatial block sizes (e.g., 32×32) decreases the resolution of the algorithm such that smaller fire regions cannot be detected. Using smaller blocks sizes increases the computational cost of the algorithm because the number of blocks to be tested by the SVM increases for the same fire region. The temporal dimension of the blocks is determined by the frame rate parameter F_{rate}, which is between 10 and 25 in our train and test videos. These blocks do not overlap in the spatial domain, but there is a 50% overlap in the time domain. This means that classification is not repeated after every frame of the video. We perform the classification at the temporal boundaries of the overlapping blocks and use the same classification result for the next $F_{\text{rate}}/2$ frames. A sketch for the spatio-temporal block extraction and classification is shown in Fig. 1. This is done to reduce the computational cost and get rid of the delay that would be introduced if we were to wait for the classification result of the next block.

For the covariance estimation formula defined in Eq. 1, it is necessary to know the mean of the data, which requires knowing all of the data, such that data must be accumulated at the end of each period. Fortunately, there is another covariance matrix estimation formula given in Eq. 29, the calculation of which can be started without waiting for the entire dataset.

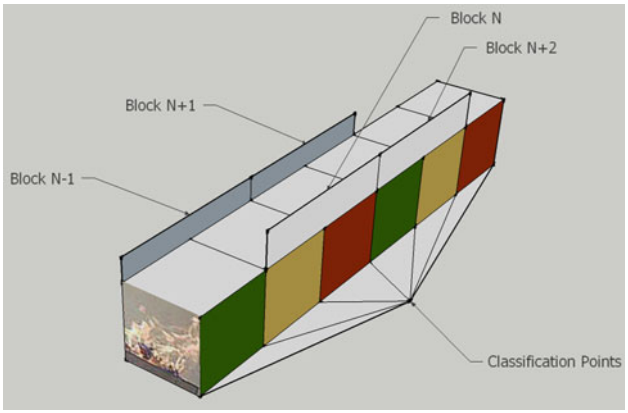


Fig. 1 An example for spatio-temporal block extraction and classification

$$\widehat{\Sigma}(a,b) = \frac{1}{N-1}\left(\sum_i \sum_j \Phi_{i,j}(a)\Phi_{i,j}(b) - C_{ab}\right)$$

where

$$C_{ab} = \frac{1}{N}\left(\sum_i \sum_j \Phi_{i,j}(a)\right)\left(\sum_i \sum_j \Phi_{i,j}(b)\right) \quad (29)$$

There is also a fast covariance computation method based on multiple integral images as described in [19]. Since we were able to obtain a real-time system with SVM and regular covariance computation we did not implement the integral images method. This method can be useful to obtain real-time system that works with higher resolution images.

After the spatio-temporal blocks are constructed, features are extracted and used to form a training set. The features are obtained by averaging the correlations between the 34-parameters for every $F_{\text{rate}}/2$ frames. A support vector machine with a radial basis function (RBF) kernel is trained for classification [4]. The SVM is first trained with seven positive and seven negative video clips given in Figs. 2 and 3. The SVM model obtained from this training is tested with the three remaining negative video clips. Sample frames from these clips are given in Fig. 4. As it is seen from the figure, these clips contain scenes that usually cause false alarms. So it is beneficial that these non-fire blocks that are classified as fire are added to the negative training set and that the SVM is retrained with the new dataset.

We have a total of 69,516 feature vectors in the training set. Of these 29,438 of those are from positive videos while 40,078 are from negative videos. In the training, we randomly select half of the elements of the negative set and one-tenth of the elements of the positive set. The SVM model obtained from these vectors is then tested using the whole dataset. We found that using the RBF kernel with gamma $=$ $1/(\text{number of; features})$ and $\text{cost}(c) = 18{,}000$ gives the best training results.

When the system is trained with 34 parameter feature vectors, we get the confusion matrix in Table 1 for the whole set, in which, the number of support vectors is 656. When the system is trained with 55 parameter feature vectors, we get the confusion matrix in Table 2 for the training set, where the number of support vectors is 1,309. When we remove the

Fig. 2 Sample image frames from the positive training video clips

Fig. 3 Sample image frames from the negative training video clips

Fig. 4 Sample image frames from the training video clips that are used to reduce false alarms

Table 1 Confusion matrix when 34-parameter feature vectors are used

		Predicted labels	
		Not fire (%)	Fire (%)
Actual Labels	Not fire	40066/100.0	12/0.0
	Fire	999/ 3.4	28439/96.6

temporal derivatives $\partial_t I(i, j, n)$ and $\partial_t^2 I(i, j, n)$ from the property set and obtain 21-parameter feature vectors, we get the confusion matrix in Table 3, with the number of support vectors being 2,063. The true detection rates are 96.6, 95.5 and 91.7% for the 34 parameter, 55 and 21 parameter cases, respectively. When 55 parameters are used the detection rate decreases and the number of support vectors increases for our training set. When 21-parameters are used we lose the temporal flickering characteristics of flame-colored blocks and the true detection rate decreases. Therefore, we will use the SVM model obtained using the 34 parameter case in the experiments.

In Table 4 SVM, k-nearest neighbor (k-NN) and boosted tree based classifiers are compared in terms of the True Detection and (TDR) and True Rejection Rate (TRR) percentages when 34-parameter feature vectors are used. OpenCV [1] implementations of the classifers are used. OpenCV implements Real [20], Gentle [10], and Logit [10] boosting. For all boosted classifiers, the number of weak tree learners is

Table 2 Confusion matrix when 55-parameter feature vectors are used

		Predicted labels	
		Not fire (%)	Fire (%)
Actual Labels	Not fire	40023/99.9	55/0.1
	Fire	1314/4.5	28124/95.5

Table 3 Confusion matrix when 21-parameter feature vectors are used

		Predicted labels	
		Not fire (%)	Fire (%)
Actual Labels	Not fire	40007/99.8	71/0.2
	Fire	2429/8.3	27009/91.7

Table 4 Comparison of different classifiers on the test set containing 69,516 image patches in terms of true detection (TDR)/true rejection (TRR) rate percentages and the total classification time

Classifier	# of patches/TRR (%)	# of patches/TDR (%)	Time (s)
SVM-RBF	40066/100	28439/96.6	2.297
SVM-Linear	39953/99.7	26758/90.9	4.218
Real Boost	40058/100.0	27987/95.1	3.031
Gentle Boost	40076/100.0	28737/97.6	6.281
Logit Boost	40048/99.9	27729/94.2	4.937
k-NN	40046/99.9	28661/97.4	69.83

200 and the max tree depth is 10. Two different SVM classifiers that use RBF and linear kernels are used. For the *k*-NN classifier, the number of nearest neighbors is five. We see that except for SVM-Linear all algorithms have comparable detection performance, but SVM-RBF has the least computational complexity which becomes important when the classifier is used in a real-time application. The most computationally costly algorithm is *k*-NN. It is possible to implement the *k*-NN algorithm in a parallel architecture to reduce the computational cost of the *k*-NN approach [11]. Since most classifiers have similar performance, we can also conclude that covariance features are the main factor contributing detection performance.

During the implementation, in each spatio-temporal block, the number of chromatic fire-pixels, $\sum_i \sum_j \sum_n \Psi(i, j, n)$, is found. If this number is higher than or equal to two-fifths of the number of elements in the block ($16 \times 16 \times F_{\text{rate}}$), then that block is classified as a flame-colored block. This thresholding is done because only fire-colored pixels according to the chromatic model described in Sect. 2.1 is used in covariance analysis. If the number of possible fire-pixels is enough, then classification is done by the SVM classifier using the augmented feature vector described in Sect. 2.3.

Fig. 5 An example post processing of patches in an image: Small blocks indicate regions, which are classified as flame regions

At the final step of our flame detection method, a confidence value is determined according to the number of positively classified video blocks and their positions. We use spatial 8-neighborhood to find the confidence level of the algorithm for the current frame. At the final step of our flame detection method, a confidence value is determined according to the number of positively classified video blocks and their positions. Assuming that the initial patch decisions are as shown in Fig. 5, these small blocks are initially classified as regions containing flames pixels. Final classification is done by sliding a 3×3 window around the patches classified as fire. If its center is fire and the sum of the decisions that are fire is greater than or equal to a predetermined confidence level in a given region, then that image is classified as a fire region. In Fig. 6, there are positive and negative test blocks for the maximum confidence level of 3. The result of Fig. 6f is negative because its center patch is not classified as a fire block.

4 Experimental results

The proposed system is compared with one of our previous fire detection methods [25]. Our previous system is one of the state of the art methods using in video based fire detection. It is the baseline system in the Firesense project from which we have distributed more than 100 copies of our dataset and software [3]. In the decision process, if the confidence level of any block of the frame is greater than or equal to three then that frame is marked as a fire-containing frame. The method described in [25] has a similar confidence level metric to determine the alarm level. Further, if the confidence level of any region of the frame that is classified as fire is greater than or equal to three, then that frame is marked as a fire-containing frame. Results are summarized in Table 5 and Table 6, in terms of the true detection and the false alarm ratios, respectively. The true detection rate is defined as:

$$\frac{\text{The number of correctly classified frames, which contain fire, in test video}}{\text{Number of frames which contain fire in test video}} \quad (30)$$

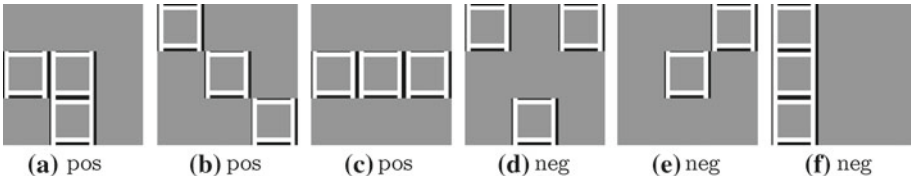


Fig. 6 Positive (pos) and negative (neg) test blocks for confidence level = 3

Table 5 Comparison of our new method with the previous method proposed in [25] in terms of true detection rates in video clips that contain fire

Video name	True detection rates	
	New method	Old method
posVideo1[a]	161/293 (54.9%)	0/293 (0.0%)
posVideo2[a]	413/510 (81.0%)	0/510 (0.0%)
posVideo3	310/381 (81.4%)	0/381 (0.0%)
posVideo4	1643/1655 (99.3%)	627/1655 (37.9%)
posVideo5	2394/2406 (99.5%)	2348/2406 (97.6%)
posVideo6	35/258 (13.6%)	0/258 (0.0%)
posVideo7	495/547 (90.5%)	404/547 (73.9%)
posVideo8[a]	501/513 (97.7%)	0/513 (0.0%)
posVideo9[a]	651/663 (98.2%)	64/663 (9.7%)
posVideo10	223/235 (94.9%)	181/235 (77.0%)
posVideo11	35/178 (19.7%)	23/178 (12.9%)
posVideo12	234/246 (95.1%)	139/246 (56.5%)
posVideo13	196/208 (94.2%)	164/208 (78.8%)

[a] Clips marked are recorded with moving cameras

Table 6 Comparison of our new method with the previous method proposed in [25] in terms of false alarm rates in video clips that do not contain fire

Video name	False alarm rates	
	New method	Old method
negVideo1[a]	160/4539 (3.5%)	260/4539 (5.7%)
negVideo2	0/155 (0.0%)	0/155 (0.0%)
negVideo3	0/160 (0.0%)	0/160 (0.0%)
negVideo4[a]	140/1931 (7.3%)	0/1931 (0.0%)
negVideo5[a]	10/439 (2.3%)	228/439 (51.9%)
negVideo6[a]	185/1142 (16.2%)	0/1142 (0.0%)
negVideo7[a]	0/541 (0.0%)	0/541 (0.0%)
negVideo8[a]	0/3761 (0.0%)	0/2943 (0.0%)
negVideo9[a]	5/645 (0.8%)	20/645 (3.1%)

[a] Clips marked are recorded with moving cameras

and the false alarm rate is defined as:

$$\frac{\text{The number of miss classified frames, which do not contain fire, in test video}}{\text{Number of frames which do not contain fire in test video}} \tag{31}$$

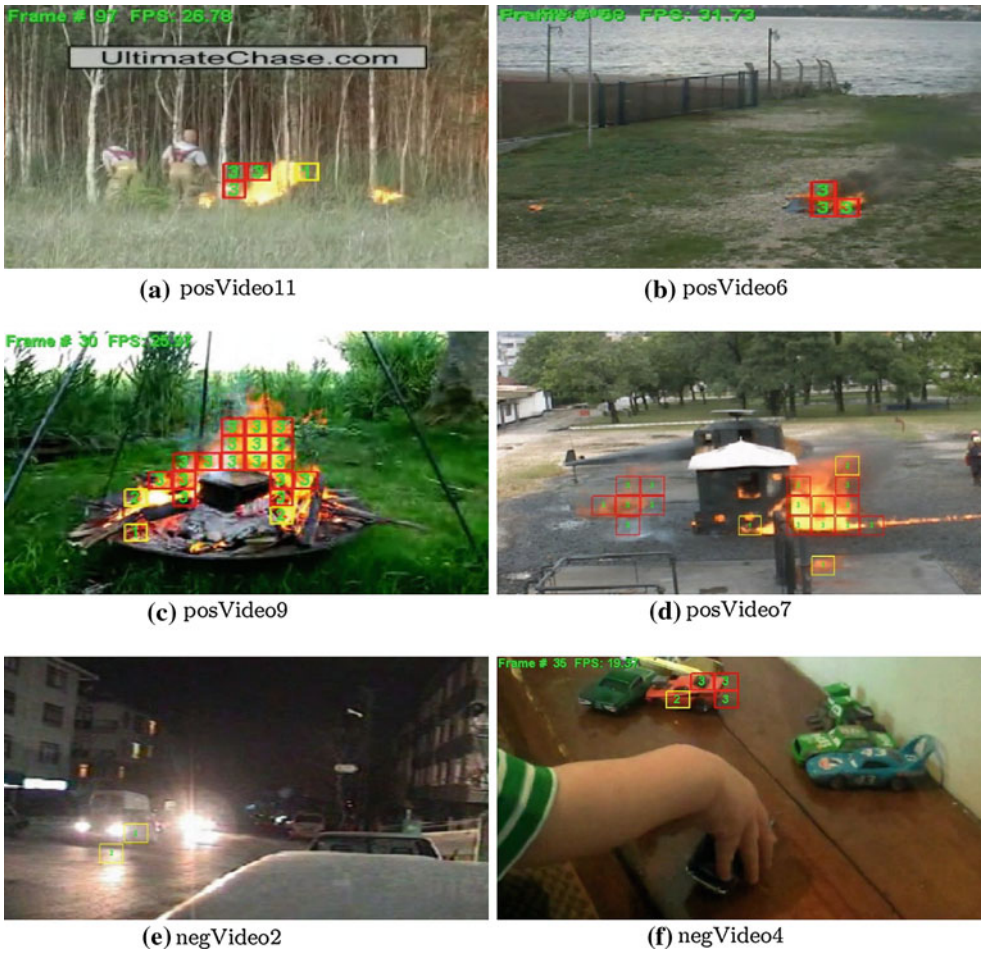
In the experiments, 22 video clips are used to test the proposed system. The first 13 videos do contain fire while the remaining nine videos contain neither fire nor flames but do contain flame-colored regions. In Table 5, the true detection rates of the two algorithms are presented for the 13 videos containing fire. In Table 6, the false alarm rates of the two algorithms are presented for the nine videos that do not contain fire. According to the results, our system is able to classify all video files containing fire, with a reasonable false alarm ratio in videos without fire. Although the true detection rate is low in some videos, we do not need to detect all fire frames correctly to issue an alarm. It is enough to detect fire in a short time without too many false alarms. The first detection time is less than two seconds in all the test video clips except for one video clip containing a small uncontrolled fire in which the fire is detected in 7 s.

As compared to the previous method, the new method has a higher true detection rate in all of the videos that contain actual fires. While the older method has a lower false alarm rate than the new method, it cannot handle videos containing camera motion. Some of the positive videos in the test set are recorded with hand-held moving cameras; since the old method assumes a stationary camera for background subtraction, it cannot correctly classify most of the actual fire regions. Our algorithm is designed to detect all fires with a reasonable amount of false alarms. In Fig. 7, there are sample frames after classification.

The proposed method is also tested with video clips from the DynTex dynamic texture database [17]. With this database, 142 video clips, most of which contain fire-colored moving regions are selected. The total number of frames in the clips is 75,327; 9,938 of them are incorrectly classified as fire-containing frames. Although 13.2% might seem as a high false alarm ratio for a fire detection system, most of the clips with false alarms contain dynamic textures that would not be present in a general surveillance environment as shown in Fig. 8. Most of the false alarms are issued in such cases as flashing red lights, red fish and moving red plants in water, and red flags in the wind. These have similar temporal characteristics to fire flames but are not very important to distinguish from flames, since they are not common in typical surveillance scenarios.

The proposed method is computationally efficient. The experiments are performed with a PC that has a Core 2 Duo 2.2 GHz processor Video clips are generally processed around 20 fps when image frames of size 320×240 are used.

The detection resolution of the algorithm is determined by the video block size. Since we require three neighboring



(a) posVideo11 (b) posVideo6

(c) posVideo9 (d) posVideo7

(e) negVideo2 (f) negVideo4

Fig. 7 Detection results from test videos

Fig. 8 Sample image frames from DynTex dynamic texture database

blocks to reach the highest confidence level the fire should occupy a region of size at least 32 × 32 in video.

5 Conclusions

In this paper, a real-time video fire detection system is developed based on the covariance texture representation method. The covariance method is ideally suited for flame detection for two reasons: flames exhibit random behaviour [25], and it is experimentally observed that the underlying random process can be considered as a wide-sense stationary process for a small video region. Thus, the second order statistical modeling using the covariance method provides a good solution to flame detection in video. The proposed method is computationally efficient and can process 320 × 240 frames at 20 fps in an ordinary computer. An important contribution of this article is the use of temporal covariance information in the decision process. Most fire detection methods use color, spatial and temporal information separately, but in this work we use temporally extended covariance matrices in order to use all the information together. The method works very well when the fire is clearly visible and in close range, such that the flicker and irregular nature of flames are easily observable. On the other hand, if the fire is small and far away from the camera or covered by dense smoke, in which the flickering behaviour of flames cannot be visible in video, the method might perform poorly, because the system is trained with video clips containing flickering flames.

Acknowledgments This work was supported in part by FIRESENSE (Fire Detection and Management through a Multi-Sensor Network for the Protection of Cultural Heritage Areas from the Risk of Fire and Extreme Weather Conditions, FP7-ENV-2009-1244088-FIRESENSE).

References

1. Bradski, G.: The OpenCV Library. Dr. Dobb's J. Softw. Tools (2000)
2. Çetin, A.E., Porikli, F.: Special issue on dynamic textures in video. Mach. Vis. Appl. **22**(5), 739–740 (2011)
3. Cetin, A.E.: Computer vision based fire detection software (2007). http://signal.ee.bilkent.edu.tr/VisiFire/index.html
4. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001). Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
5. Chen, T.H., Wu, P.H., Chiou, Y.C.: An early fire-detection method based on image processing. In: International Conference on Image Processing (ICIP), vol. 3, pp. 1707–1710 (2004)
6. Dedeoğlu, Y., Töreyin, B.U., Güdükbay, U., Çetin, A.E.: Real-time fire and flame detection in video. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP, pp. 669–672 (2005)
7. Doretto, G., Chiuso, A., Wu, Y.N., Soatto, S.: Dynamic textures. Int. J. Comput. Vis. **51**(2), 91–109 (2003)
8. Ferrari, R., Zhang, H., Kube, C.: Real-time detection of steam in video images. Pattern Recogn. **40**(3), 1148–1159 (2007)
9. FIRESENSE: Fire detection and management through a multi-sensor network for the protection of cultural heritage areas from the risk of fire and extreme weather conditions, fp7-env-2009-1-244088-firesense (2009). http://www.firesense.eu
10. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Ann. Stat. **28** (1998)
11. Garcia, V., Debreuve, E., Barlaud, M.: Fast *k*-nearest neighbor search using GPU. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1–6 (2008)
12. Gunay, O., Tasdemir, K., Töreyin, B.U., Çetin, A.E.: Fire detection in video using lms based active learning. Fire Technol. **46**, 551–577 (2010)
13. Hong, W., Peng, J., Chen, C.: A new image-based real-time flame detection method using color analysis. In: IEEE International Conference on Networking, Sensing and Control, pp. 100–105 (2005)
14. Phillips, W III., Shah M., da Vitoria Lobo, N.: Flame recognition in video. In: Fifth IEEE Workshop on application of Computer Vision, vol. 23, pp. 224–229 (2000)
15. Liu, C.B., Ahuja, N.: Vision based fire detection. In: 17th International Conference on Pattern Recognition (ICPR), vol. 4, pp. 134–137 (2004)
16. Lu, T.F., Peng, C.Y., Horng, W.B., Peng, J.W.: Flame feature model development and its application to flame detection. In: Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC), pp. 158–161 (2006)
17. Péteri, R., Fazekas, S., Huiskes, M.J.: DynTex : a Comprehensive Database of Dynamic Textures. Pattern Recogn. Lett. (2010). http://projects.cwi.nl/dyntex/
18. Phillips, W I., Shah, M., Da Vitoria Lobo, N.: Flame recognition in video. In: Fifth IEEE Workshop on Applications of Computer Vision, pp. 224–229 (2000)
19. Porikli, F., Tuzel, O.: Fast construction of covariance matrices for arbitrary size image. In: IEEE International Conference on Image Processing (ICIP), pp. 1581–1584 (2006)
20. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Mach. Learn. **37**, 297–336 (1999)
21. Shi, L., Cao, Z.: The flame detection and background interference filtering algorithm based on video images. In: Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering (CSIE), pp. 559–563 (2009)
22. Töreyin, B.U., Çetin, A.E.: Online detection of fire in video. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–5 (2007)
23. Töreyin, B.U., Cinbis, R.G., Dedeoğlu, Y., Çetin, A.E.: Fire detection in infrared video using wavelet analysis. Opt. Eng. **46**, 551–577 (2007)
24. Töreyin, B.U., Dedeoğlu, Y., Çetin, A.E.: Flame detection in video using hidden markov models. In: IEEE International Conference on Image Processing (ICIP), vol. 2, pp. 1230–1233 (2005)
25. Töreyin, B.U., Dedeoğlu, Y., Güdükbay, U., Çetin, A.E.: Computer vision based method for real-time fire and flame detection. Pattern Recogn. Lett. **27**(1), 49–58 (2006)
26. Tuna, H., Onaran, I., Çetin, A.E.: Image description using a multiplier-less operator. IEEE Signal Process. Lett. **16**(9), 751–753 (2009)
27. Tuzel, O., Porikli, F., Meer, P.: Region covariance: a fast descriptor for detection and classification. In: European Conference on Computer Vision (ECCV), pp. 589–600 (2006)
28. Yamagishi, H., Yamaguchi, J.: A contour fluctuation data processing method for fire flame detection using a color camera. In: 26th Annual Conference of the IEEE Industrial Electronics Society (IECON), vol. 2, pp. 824–829 (2000)

Author Biographies

Yusuf Hakan Habiboğlu received his B.Sc. degree in electrical and electronics engineering in 2008 from the Eskişehir Anadolu University. He received his M.S. degree in electrical and electronics engineering from the Bilkent University, Ankara, Turkey, in 2010. His research interests include computer vision, pattern recognition, feature extraction and image segmentation.

Osman Günay received his B.Sc. and M.S. degrees in electrical and electronics engineering from the Bilkent University, Ankara, Turkey. Since 2009, he has been a Ph.D. student at the Department of Electrical and Electronics Engineering in the Bilkent University, Ankara, Turkey. His research interests include computer vision, video segmentation and dynamic texture recognition.

A. Enis Çetin received his Ph.D. degree from the University of Pennsylvania in 1987. Between 1987 and 1989, he was an assistant professor of electrical engineering at the University of Toronto. He has been with the Bilkent University, Ankara, Turkey, since 1989. A. E. Cetin was an associate editor of the IEEE Trans. on Image Processing between 1999 and 2003. Currently, he is on the editorial boards of journals, Signal Processing and Journal of Advances in Signal Processing (EURASIP), and Journal of Machine Vision and Applications (IAPR), Springer. He is a fellow of IEEE. His research interests include signal and image processing, human–computer interaction using vision and speech and audiovisual multimedia databases.