http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 9 (4): 758-765, 2010
ISSN 1812-5638

A Vector Quantization Based Automatic Fire Detection System

[1,2]Yu-Chun Wen, [1]Fa-Xin Yu, [1]Xiao-Lin Zhou and [1]Zhe-Ming Lu
[1]School of Aeronautics and Astronautics, Zhejiang University, Hangzhou, People's Republic of China
[2]Zhejiang California International NanoSystems Institute, Zhejiang University,
Hangzhou, People's Republic of China

Abstract: This study presents a novel fire detection method based on vector quantization. Before online fire detection, we generate a fire codebook and a non-fire codebook by the LBG algorithm based on the training set that are selected from 10 video clips under different scenes and conditions. For encoding convenience, we merged the two codebooks into one codebook and sorted the codewords in the ascending order of their mean values for the future Equal-average Equal-variance Equal-norm Nearest Neighbor Search (EEENNS) based fast encoding process. In the online fire detection process, the video to be detected was first segmented into successive frames and we performed the VQ (Vector Quantization) encoding process to find fire-colored frames and recorded the grade of each fire-colored area. Then, the moving pixel detection process was performed on each fire-colored frame to find candidate fire frames. Finally, we verified whether a fire occurs or not and graded the fire by analyzing the change in the number of blocks belonging to each grade between consecutive frames. Experimental results demonstrated the effectiveness of the proposed scheme and a 93.3% detection rate was obtained with 25 test video clips.

Key words: Fire detection, vector quantization, moving pixel detection, EEENNS algrithm, fire gradation

INTRODUCTION

A fire accident brings a lot of damage to our properties and even takes away many people's lives. In the event of a fire, there are certain features suitable for fire detection. The most common features are heat and smoke, depending on whether the fire is smoldering or flaming and on the type of fuel. In addition to these features, there are a number of other traits such as gases and light. All kinds of fire detection systems have been applied based on various sensors. The use of smoke detectors has increased steadily since the mid-1970s, however, there is a need for smoke detectors to discriminate between fires and nuisance sources. Recently, it has verified that a system consisting of multiple gas and Taguchi sensors combined with thermocouples is capable of detecting and discriminating between flaming and smoldering fires (Hagen and Milke, 2000). A more recent research has been focused on detecting a smoldering fire which is mainly caused by a cigarette in a general house based on tin oxide gas sensors (Sawada *et al*., 2008).

Besides sensor-based fire detection, vision-based fire detection is also a useful way to sound the fire alarm automatically. This study focuses on fire detection based on video processing, which can be viewed as light sensors. There are many significant static and dynamic characters in the fire video. Static characters include color, shape, spectral texture, edge irregularity and so on. Moving characters include the difference between two consecutive frames, the change in the number of fire-colored pixels, the movement of the fire-colored area, flame flickering and so on. To detect a fire in the video, many schemes must obtain the fire area in each frame first, where the limitation in the RGB color space or some other color spaces such as YCbCr, HIS and I1I2I3 (Yu *et al*., 2008). Alternatively, the Gaussian distribution based method can be used to obtain the general fire area (Ko *et al*., 2009). After detecting candidate fire areas, the next step is to remove unlikely pixels or unwanted areas. It should be noted that the range of fire intensity is similar to many nature phenomena such as the static and moving light from the sun or cars or some other bright moving objects. To eliminate the unwanted light, the static light is eliminated by comparing the fire image with the background image and the moving light is removed by applying the and operation to the first and the last frames and then the noises are removed by the erosion and dilation operations (Han and Lee, 2009). A modified hybrid background estimation method is used to detect the moving area and non-fire pixels are removed based on temporal luminance variation (Ko and Nam, 2006). Recently, some researchers have been dedicated to the study on the flame flickering and the influence of the

Corresponding Author: Zhe-Ming Lu, Teaching Building 5, Yuquan Campus, Zhejiang University, 38 ZheDa Road, Hangzhou 310027, People's Republic of China Tel/Fax:+86-571-87951589

surrounding on the main flame pulsation frequency, which permits a flame to be distinguished from a potential source of false alarms (Thuillard, 2002). Quasi-periodic behavior in flame boundaries is detected by performing temporal wavelet transform, while color variations in flame regions are detected by computing the spatial wavelet transform of moving fire-colored regions (Toreyin *et al*., 2006). In addition, some researchers focus on the fire that occurs in some special place such as automatic fire detection in road traffic tunnels (Aralt and Nilsen, 2009) and early fire detection for vessels (Wang *et al*., 2009).

Most existing schemes can not achieve low false alarm rates and high detection speed at the same time and the fire area is not graded. To improve these performances, we propose a novel fire detection method based on Vector Quantization (VQ). The VQ is an efficient signal compression and pattern clustering technique. In this study, we use VQ to generate a fire codebook with three-grade codewords and a non-fire codebook. Based on these codewords, we can identify if the input pattern is a fire pattern and then grade the fire pattern.

VECTOR QUANTIZATION

Because, VQ is the key technique used in this paper, we first give a brief introduction in this section. Vector Quantization (VQ) is a popular and efficient technique for signal compression and pattern clustering (Gray, 1984). The k-dimensional N-level vector quantizer is defined as a mapping from the k-dimensional Euclidean space R^k into a certain finite set $C = \{c_1, c_2, ..., c_N\}$, where $c_i = (c_{i1}, c_{i2}, ..., c_{ik})$ is a codeword in the codebook C and N is the codebook size. The quantizer is completely described by the codebook C together with the partitioned set consisting of subspaces of the k-dimensional Euclidean space R^k, $S = \{S_1, S_2, ..., S_N\}$ and the mapping function $Q(\bullet)$: $Q(x) = c_i$, if $x \in S_i$. The partitioned sets S_i $(1 \le i \le N)$ satisfy $S_1 \cup S_2 .. \cup S_N = R^k$ and $S \cap S_j = \Phi$, if $i \ne j$. In the encoding phase, each input vector $x = (x_1, x_2, ..., x_k)$ finds its best matching codeword c_i in the codebook C and the index i is assigned to x, satisfying:

$$i = \text{argmin}_p \sum_{l=1}^{k} (x_l - c_{pl})^2$$

Only, the index i is sent over the channel to the receiver. In the decoding process, for each index i, the decoder merely performs a simple table look-up operation to obtain c_i and then uses c_i to reconstruct the input vector x.

The two key techniques in the basic VQ system are codebook design and codeword search techniques. The first issue in the VQ system is to encode each input vector with distortion as small as possible. Obviously, the encoding performance is determined by the representativeness of the codebook. In general, VQ generates a representative codebook from a number of training vectors using the well-known iterative clustering algorithm (Linde *et al*., 1980) that is often referred to as the Generalized Lloyd Algorithm (GLA).

The second issue in the VQ system is to encode each input vector as fast as possible. As we know, if the squared Euclidean distance:

$$d(x, c_i) = \sum_{l=1}^{k} (x_l - c_{il})^2$$

is used to describe the distortion between x and c_i, then the Full Search (FS) of the nearest codeword for each input vector requires kN multiplications, (2k-1)N additions and N-1 comparisons. For the VQ system with large codebook size and high dimension, the computation load is very high during the codeword search. To release the computation burden, a lot of fast codeword search algorithms have been presented to speed up the searching process while maintaining the reconstructed performance the same as that of the full search. Among them, the method based on three characteristic values, i.e., the mean, variance and norm values of a vector, is the most representative one, which is called Equal-average Equal-variance Equal-norm Nearest Neighbor Search (EEENNS) algorithm (Lu and Sun, 2003).

PROPOSED AUTOMATIC FIRE DETECTION SYSTEM BASED ON VQ

The flow chart of the proposed fire detection scheme can be shown in Fig. 1. VQ is the central step in our scheme, thus we should generate a representative codebook from a training set before online fire detection. Here we generated a fire codebook with 100 codewords by the LBG algorithm based on the training set with 50000 27-dimensional vectors (each vector is a 3×3 block with 3 color channels) that are selected from 300 frames in 10 fire video clips under different scenes and conditions. Similarly, a non-fire codebook with 200 codewords was generated from non-fire frames. The fire codebook was classified into three grades denoting different degrees of severity. For encoding convenience, we merged the two codebooks into one codebook and sorted the codewords in the ascending order of their mean values for the future EEENNS fast encoding process (Lu and Sun, 2003). The online fire detection process can be described in brief as follows: First, the video to be detected was segmented into successive frames and each frame was segmented

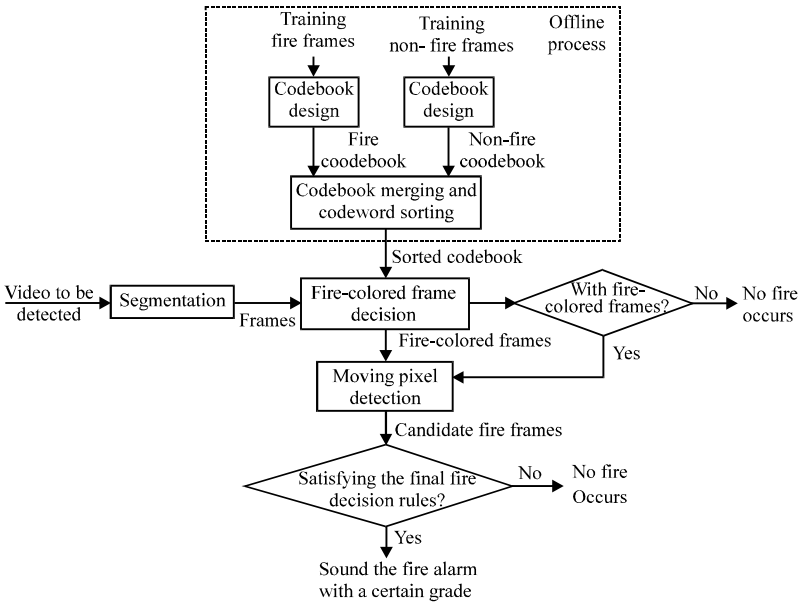

Fig. 1: The flow chart of the proposed fire detection scheme

into 3×3 non-overlapping blocks. For each block in each frame, we performed the VQ encoding process with the above sorted codebook to determine whether the input block is a fire-colored block or not and recorded the grade of the fire-colored block. If the number of fire-colored blocks in a frame is larger than a certain threshold, we call this frame a fire-colored frame. If there are no fire-colored frames, then the algorithm is terminated without fire alarm. Otherwise, the moving pixel detection process is performed on each fire-colored frame to determine whether the current fire-colored frame is a candidate fire frame or not according to the number of moving pixels in this frame. Finally, we verified whether a fire occurs or not and graded the fire by analyzing the change in the number of blocks belonging to each grade between consecutive frames.

Codebook design: For fire area detection, we need a uniform codebook for various scenes and conditions. Because, this codebook is designed offline, the time required is out of our consideration. In many pattern recognition systems, a rule for classification is necessary to detect the target area, such as the vision-based road detection (Yanqing *et al*., 2010) and the K-Means clustering to improve the accuracy of decision tree response classification (Ali *et al*., 2009), each representing a type. Thus, before codebook generation, we also require a rule to choose the fire area in video clips. Based on many experiments on various video clips, we draw the following simple rule to determine if a pixel $p = (p_R, p_G, p_B)$ is a fire-colored pixel:

$$\begin{cases} p \text{ is a fire-colored pixel} & p_R > 130 \text{ and } p_G > p_B \\ p \text{ is not a fire-colored pixel} & \text{otherwise} \end{cases} \quad (1)$$

From Eq. 1, we can see that the rule is loose so that we can get fire areas to cover all possible fire conditions although some non-fire areas may also be selected. In fact, non-fire areas can be excluded by subsequent steps.

Based on the above rule, we can construct two training sets to generate the fire and non-fire codebooks, $C_F = \{c_{F1}, c_{F2}, ..., c_{F100}\}$ and $C_{NF} = \{c_{NF1}, c_{NF2}, ..., c_{NF200}\}$, based on the LBG algorithm, respectively. The first set consists of 50000 27-dimensional vectors, each vector being a 3×3 fire-colored block with 3 color channels. Similarly, the second set consists of 50000 3×3 non-fire blocks. In order to use the EEENNS algorithm in the VQ encoding process, we merge the two codebooks into one codebook $C = \{c_1, c_2, ..., c_{300}\}$ and sort the codewords in the ascending order of their mean values.

In addition, to grade the severity degree of a fire, we require classifying the codewords in the fire codebook C_F into three grades. We draw the classification rule based on following statistical analysis. First, we get 15 fire video clips and generate their RGB histograms individually according to the rule as shown in Eq. 1. To obtain the general RGB distribution feature, we choose 10 fire video clips with both fire and non-fire areas according to Eq. 1 and the result are shown in Fig. 2a-c. To attain the RGB

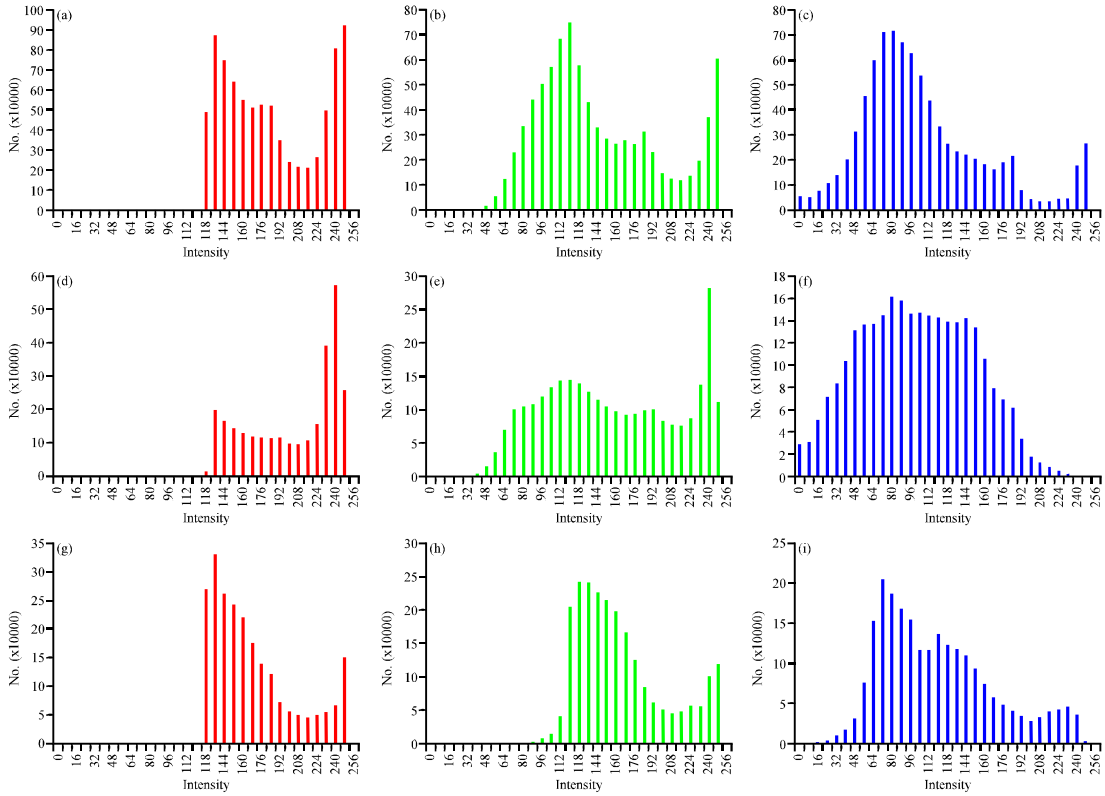

Fig. 2: The RGB distributions for different conditions. (a, b and c) are the RGB histograms for 300 frames from both fire and non-fire video clips. (d, e and f) are the RGB histograms for 100 fire frames only from fire video clips. (g, h and i) are the RGB histograms of 100 frames only with non-fire areas. (a, d, g) Red, (b, e, h) Green and (c, f, i) Blue

distribution character of the fire area, we only choose fire areas in five fire video clips and the histograms are shown in Fig. 2d-f. For comparison, we choose five video clips without fire areas but some are falsely detected as fire areas by Eq. 1 and the histograms for these areas are depicted in Fig. 2g-i. As we can see, there is a dramatic difference in red color distribution among above three cases. In the second case as shown in Fig. 2d, the red components are mainly with values greater than 230. On the contrary, they are mainly with values lower than 200 in the third case. Therefore, the value 230 can be taken as a dividing point for the red channel. In contrast, there is no significant difference among blue histograms, all of which have a peak in the value of 80 although, the smoothness is different to a certain extent. For the green channel as shown in Fig. 2b, d and h, there are some difference among their distributions. Based on Fig. 2h, we select the value 100 to be a dividing point for the green channel and choose the value 88 as another dividing point to classify the pixel with small green component value to another grade. Consequently, we classify the severity degree of each fire codeword c_{Fj} whose mean value is $p = (p_R, p_G, p_B)$ by the following rule:

$$\begin{cases} c_{Fj} \in \text{Grade1} & p_R > 230, p_G > 100 \\ c_{Fj} \in \text{Grade 2} & 130 < p_R < 230, 88 < p_G < 100 \\ c_{Fj} \in \text{Grade 3} & p_R > 130, p_G < 88, p_G > p_B \end{cases} \quad (2)$$

VQ encoding based fire-colored frame decision: Once, we have the sorted codebook with graded codewords in hand, all 3×3 non-overlapping blocks in each frame of the video to be detected can be classified by searching the best-match codeword for each input block whose grade is just set as that of the best-match codeword. Here, the 9 pixels in each block are classified into the same grade. Then, for each frame, the number of pixels belonging to each grade is counted and recorded for future fire pixel verification.

As mentioned in last section, the VQ encoding time should be reduced as much as possible to make the fire detection real-time, thus a fast nearest neighbor search algorithm should be adopted. To obtain the best-match codeword for each input vector as quickly as possible, we performed the EEENNS algorithm (Lu and Sun, 2003) here. The main idea is as follows: For an input k-dimensional vector x, its variance and norm values are given as:

$$v_x = \sqrt{\sum_{i=1}^{k}(x_i - m_x)^2}$$

and

$$\|x\| = \sqrt{\sum_{i=1}^{k} x_n^2},$$

respectively, where m_x is the mean value of the components in x. It can be shown that the distance calculation is necessary only for those codewords with variances ranging from $v_{min}=v_x-\sqrt{d_{min}}$ to $v_{max}=v_x+\sqrt{d_{min}}$ and with norms ranging from $norm_{min}=\|x\|-\sqrt{d_{min}}$ to $norm_{max}=\|x\|+\sqrt{d_{min}}$. The elimination process of the EENNS algorithm consists of three steps. In the first step, if $m_{cj}\geq m_{max}$ or $m_{cj}\leq m_{min}$, then the codeword c_j can be rejected. Otherwise, in the second step, if $v_{cj}\geq v_{max}$ or $v_{cj}\leq v_{min}$, then the codeword c_j can also be rejected. In the third step, if $norm_{cj}\geq norm_{max}$ or $norm_{cj}\leq nomr_{min}$, then the codeword c_j can be rejected too. To perform the EEENNS algorithm, N mean values, N variances and N norms of all codewords should be computed off-line and stored. In this study, N = 300.

Now, we turn to the decision of fire-colored frames. Assume the total number of pixels in each frame is M, the current frame to be processed is F_n (n = 1, 2, 3, ...) and the number of pixels belonging to Grade i in this frame is F_n (i = 1, 2, 3), the decision rule can be expressed as follows:

$$\begin{cases} F_n \text{ is a fire-colored frame} & M_i^{(n)} > \tau_1 M (i = 1 \text{ or } 2 \text{ or } 3) \\ F_n \text{ is not a fire-colored frame} & \text{otherwise} \end{cases} \quad (3)$$

where, τ_1 is a small threshold. In this study, we set $\tau_1 = 0.001$.

Moving pixel detection: This step is applied to check whether the pixels in the fire-colored frames are moving or not. If they are not moving we don't consider them as the candidate fire pixels since some other static objects, such as the sun and the road light, may also have the same color features as the fire. The aim of this step is to identify the general moving area but not the accurate area, because we only need to know whether the fire-colored area obtained in the first step is a real flickering area or not. In this paper, we detect the moving pixels by the following rule (Ko *et al.*, 2009):

$$y_{n+1}(k, l) = \begin{cases} \alpha y_n(k, l) + (1-\alpha) x_n(k, l) & |y_n(k, l) - x_n(k, l)| \leq \tau \text{ (non-moving)} \\ x_n(k, l) & \text{otherwise (moving)} \end{cases} \quad (4)$$

where, $x_n(k, l)$ is the luminance value of the pixel located at (k, l) in the frame F_n (n = 1, 2, 3, ...) and y_{n+1} (k, l) is the estimated value for the pixel located at (k, l) in the next frame F_{n+1}, α is the weighting factor of the estimated value, τ is the threshold (we set τ = 7 in this study). Then, we binary each fire-colored frame F_n as follows:

$$x_n(k,l) = \begin{cases} 0 & |y_n(k, l) - x_n(k, l)| \leq \tau \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

where, x_n (k, l) is the same threshold as in Eq. 5. Obviously, the pixel with the value 1 is a moving pixel. In the experiments, we find that the binary frame obtained by Eq. 5 is not as robust as we want, thus we should smooth it. Here, we adopt the following method. First, we divide the frame into 4×4 blocks, each having 16 pixels. For each block, if there are more than 8 pixels with the value 1, then we set all values in this block to 1; Otherwise, we set them to 0. Figure 3 gives a concrete example, where Fig. 3a is

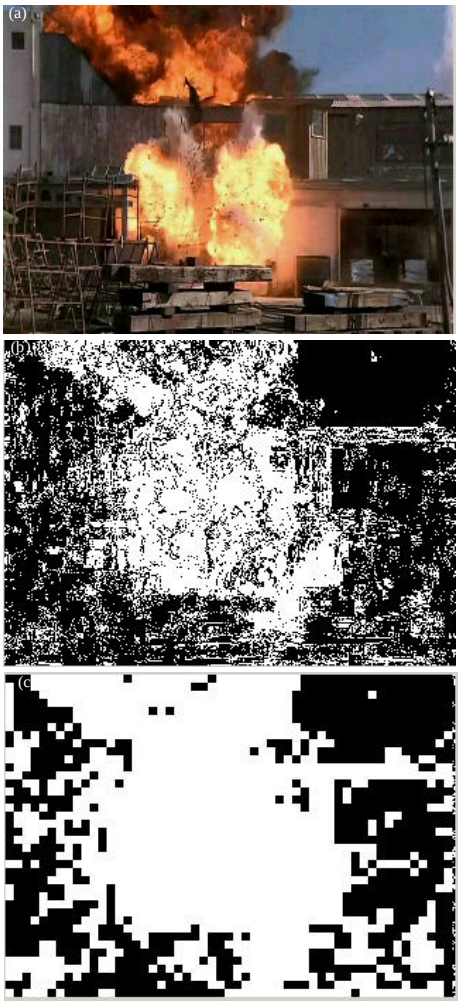

Fig. 3: An concrete example to show the smoothing effect. (a) is the original image, (b) is the result obtained by Eq. 5 and (c) is the smoothed result of b

the origin image, Fig. 3b is the result obtained by Eq. 5 and Fig. 3c is the smoothed result of Fig. 3b. Obviously, after moving pixel detection, some of the fire-colored pixels are eliminated and the number of graded pixels will be reduced.

Now, we turn to the decision of candidate fire frames. For the current fire-colored frame F_n, assume the number of moving pixels belonging to Grade i in this frame is $M_i^{(n)\_mov}$ (i = 1, 2, 3), the decision rule can be expressed as follows:

$$\begin{cases} F_n \text{ is a candidate fire frame} & M_i^{(n)\_mov} \geq \tau_2 M (i = 1 \text{ or } 2 \text{ or } 3) \\ F_n \text{ is not a candidate fire frame} & \text{otherwise} \end{cases} \quad (6)$$

where, $\tau_2 \geq \tau_1$ is a small threshold. In this study, we set $\tau_2 = \tau_1 = 0.001$. Here, if $M_i^{(n)\_mov} < \tau_2 M$ is satisfied for any Grade i, we set all the moving pixels with Grade i as non-candidate fire pixels. If we use $M_i^{(n)\_con}$ to represent the number of candidate fire pixels belonging to Grade i in the n-th frame, then we have:

$$M_i^{(n)\_can} = \begin{cases} 0 & M_i^{(n)\_mov} < \tau_2 M \\ M_i^{(n)\_mov} & M_i^{(n)\_mov} \geq \tau_2 M \end{cases} \quad (7)$$

Fire pixel verification: After above two phases, we have extracted candidate fire areas with different grades. Now we should verify them to determine whether a fire occurs in the video and give the fire grade. As we know, a fire usually spreads at the starting of occurrence, thus the number of fire pixels will also increase. When a fire has occurred for some time, the spreading speed will slow down and the number of fire pixels will not increase all the time but decrease at some time. Furthermore, it will not stay steadily at a certain number but change all the time. After performing numerous experiments, we find that if the number of frames whose candidate fire pixels belonging to any grade increase is greater than one third of the total number of the candidate frames, the fire will definitely occur. Otherwise, no fire occurs in the video. In other words, if the number of candidate fire pixels with Grade 1 increases as the fire spreads, we set the severity of the fire to be Grade 1. Otherwise, if the number of candidate fire pixels with Grade 2 increases, then the severity of the fire is set to be Grade 2. Otherwise, if the number of candidate fire pixels with Grade 3 increases, then the severity of the fire is set to be Grade 3. Otherwise, we consider there is no fire in the video. The process can be expressed by following Eq. 8-10:

$$f_i^{(n)} = \begin{cases} 1 & M_i^{(n+1)\_can} > M_i^{(n)\_can} \\ 0 & \text{otherwise} \end{cases} \quad (i = 1, 2, 3) \quad (8)$$

$$S_i = \sum_{n=1}^{K-1} f_i^{(n)}, \quad (i = 1, 2, 3) \quad (9)$$

$$\begin{cases} \text{Grade 1 fire occurs} & S_1 > \frac{1}{3}K \\ \text{Grade 2 fire occurs} & S_2 > \frac{1}{3}K, \quad S_1 \leq \frac{1}{3}K \\ \text{Grade 3 fire occurs} & S_3 > \frac{1}{3}K, \quad S_1, S_2 \leq \frac{1}{3}K \\ \text{No fire occurs} & S_1, S_2, S_3 \leq \frac{1}{3}K \end{cases} \quad (10)$$

where, i = 1, 2, 3 represent the three severity grades of a fire, $f_i^{(n)}$ is the binary flag denoting whether the number of candidate fire pixels with Grade i increases at the n-th frame or not and S_i denotes the number of candidate fire frames with increasing candidate fire pixels belonging to Grade i. In this study, we consider 15 consecutive frames each time, thus K = 15. In Eq. 10, K/3 is selected based on our experience.

EXPERIMENTAL RESULTS AND DISCUSSION

In general, there are various fire detection systems, but there isn't a uniform fire video clips database to compare the result for different systems. In our verification process, we adopted 19 video clips including fire video clips and non-fire video clips from the web-page http://signal.ee.bilkent.edu.tr/VisiFire/ and six fire video clips in our database. Then we cut down each video clips only leaving the first 30 frames for the equity of each video. Next, we will give a concrete example to illustrate the proposed algorithm and compare our result with Ko's scheme (Ko *et al*., 2009) with the same database. At last, we will discuss the computational complexity of the proposed approach.

To evaluate the effectiveness of the proposed scheme, we first gave a concrete example to show the intermediate results for a test fire video clip whose typical frame was shown in Fig. 4a. First, we found the fire-colored pixels in the frame based on VQ encoding as shown in Fig. 4b. Second, we detected the moving pixels to get the candidate fire pixels as shown in Fig. 4c and we synthesized Fig. 4b and Fig. 4c to produce the accurate fire area as depicted in Fig. 4d. Third, we verified the fire pixels to alarm a fire with severity degrees as shown in Fig. 4e. We can see that the number of pixels with Grade 1 and the number of pixels with Grade 2 both increase sharply during the fire spreading, while the number of pixels with Grade 3 increases slowly, so we set the severity degree to be Grade 1 for the example fire video clip.

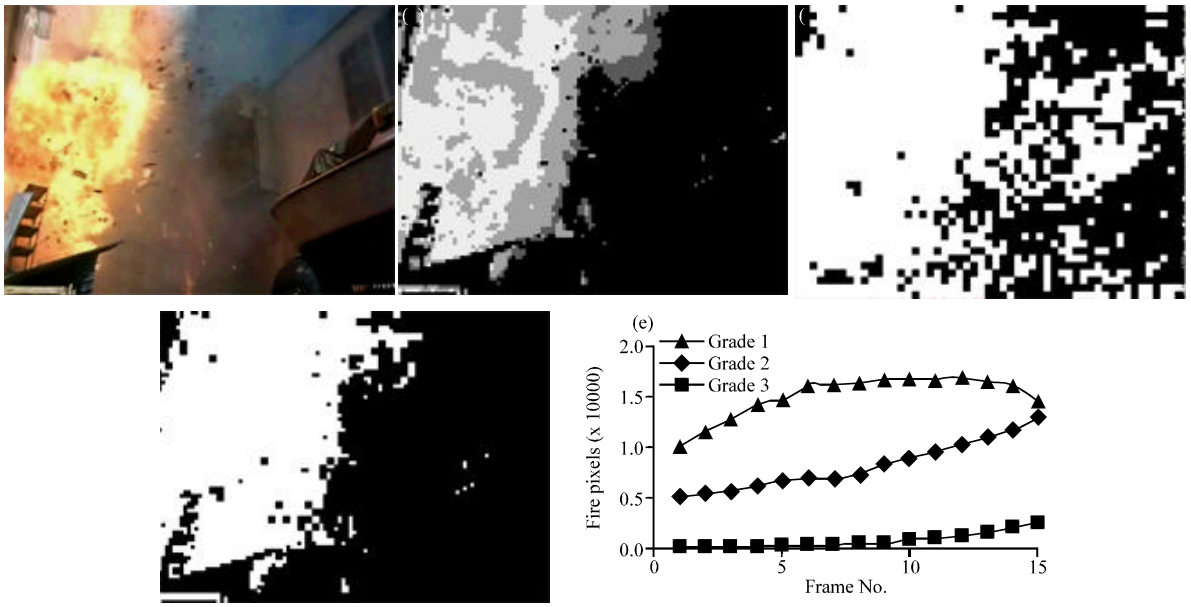

Fig. 4: The experimental results for a concrete example. (a) the original frame; (b) the fire-colored pixels in the frame; (c) the moving pixels in the frame; (d) the and result of (b) and (c) and (e) the number of fire pixels belonging to each grade

Table 1: The experimental results based on 25 test video clips

Grade	No. of fire video clips	No. of detected fire video clips	No. of fire frames	No. of detected fire frames	Detection rate (%)
1	10	11	300	315	100
2	4	2	120	75	62.5
3	1	1	30	30	100
Without fire	10	11	300	330	100
Total	25	25	750	750	93.3

Next, we selected 25 video clips, each having 30 frames, to prove the efficiency of the proposed method and the results are shown in Table 1. As shown in the Table 1, 15 video clips with fire and 10 video clips without fire are input into the system one by one and all of the videos with fires grow sharply with time are detected as fires of Grade 1 or Grade 2. All of the non-fire video clips are detected without a fire and a video clip with a stable fire is rejected to be a fire. From these results, we can see that our algorithm can exactly detect all the video clips with a growing fire. For the non-growing fire video clip, our algorithm regards it as a non-fire video clip, because a fire don't grow with time, such as cooking fires and stove fires, will not bring damage to our life.

Because in Ko's algorithm, the result doesn't include grade information, so we give a comparison table that only considers the fire and non-fire video clips without the gradation information. The comparison result is shown in Table 2, where the detection rate (R_d) represents the ratio of detected fire frames to overall fire frames and it includes true positive and false positive rates. The true

Table 2: The comparison results between our algorithm and Ko's algrithm

Comparison	Detection rate	True positive	False positive	Missing rate
	----------(%)----------			
Proposed scheme	93.3	100.0	0	6.7
Ko's scheme	86.5	86.1	0.4	13.2

positive rate (R_{tp}) means the rate of detecting a real fire as a fire and the false positive rate (R_{fp}) means the rate of recognizing a non-fire as a fire and the missing rate (R_m) represents the rate of recognizing a real fire as a non-fire. Let, N be the total number of frames, N_f and N_{nf} be the numbers of frames with fire and without fire, respectively and N_d be the number of frames detected with fire. The relationship among the above parameters can be shown as follows:

$$R_d = \frac{N_d}{N_f},\ R_{tp} = \frac{N_d \cap N_f}{N_f},\ R_{fp} = \frac{N_d \cap N_{nf}}{N_f},\ R_m = \frac{N_f - N_d \cap N_f}{N_f} \quad (11)$$

Table 2 shows our algorithm can get a higher detection rate of 93.3% compared to 86.5% and a lower missing rate of 6.7% compared to 13.2%. The true positive rate of our proposed algorithm is 100% that is higher than that of Ko's algorithm, while the false positive is 0 in contrast to 0.4% in Ko's algorithm.

Here, all experiments are performed on an Intel(R) core(TM)2 Duo PC with 3.16 GHz CPU and 4 GB memory. In the proposed algorithm, the computational time is short enough to process video with the capture rate up to

Table 3: The computational time in each phase for processing 100 consecutive frames (The size of each frame: 320×240)

Phase	Fire-colored frame decision	Moving pixel detection	Fire pixel verification	Total
Time (m sec)	1360	295	45	1700

58 frames sec^{-1} as shown in Table 3. From the table we can see that, given 100 consecutive frames of size 320×240, it only need 17 ms to process a frame, that is, it can process 58 frames in a second. The proposed algorithm can be used for real time application in general conditions, but it should be improved for HD video.

CONCLUSIONS

This study proposes a novel fire detection method based on vector quantization. Conventional fire detection algorithms are usually only suitable for the fires occur in some fixed conditions; however, our scheme considers all kinds of conditions and generates a combined codebook to detect the fire areas with different severity grades. As an efficient pattern clustering and classification technique, VQ encoding was introduced in the fire-colored frame decision stage, which was performed not pixel-by-pixel but block-by-block and the fast EEENNS algorithm was performed to speed up the VQ encoding process. Moving pixel detection was performed to reduce the possible turbulence from the sun, the moving car lights or other non-fire lights. In addition, the fact that the number of pixels belonging to each grade would increase when a fire occurs was used as an extra rule to identify a fire. Experimental results proved that the proposed method is efficient to detect all kinds of fire areas and it is robust against moving objects. Future work will focus on the implementation of the fire alarm system in Hardware and improve the detection speed.

REFERENCES

Ali, S.A., N. Sulaiman, A. Mustapha and N. Mustapha, 2009. K-means clustering to improve the accuracy of decision tree response classification. Inform. Technol. J., 8: 1256-1262.

Aralt, T.T. and A.R. Nilsen, 2009. Automatic fire detection in road traffic tunnels. Tunnelling Underground Space Technol., 24: 75-83.

Gray, R.M., 1984. Vector quantization. IEEE ASSP Magazine, 1: 4-29.

Hagen, B.C. and J.A. Milke, 2000. The use of gaseous fire signatures as a mean to detect fires. Fire Safety J., 34: 55-67.

Han, D. and B. Lee, 2009. Flame and smoke detection method for early real-time detection of a tunnel fire. Fire Safety J., 44: 951-961.

Ko, B.C. and J.Y. Nam, 2006. Object-of-interest image segmentation based on human attention and semantic region clustering. J. Optical Soc. Am. A, 23: 2462-2470.

Ko, B.C., K.H. Cheong and J.Y. Nam, 2009. Fire detection based on vision sensor and support vector machines. Fire Safety J., 44: 322-329.

Linde, Y., A. Buzo and R.M. Gray, 1980. An algorithm for vector quantizer design. IEEE Trans. Commun., 28: 84-95.

Lu, Z.M. and S.H. Sun, 2003. Equal-average equal-variance equal-norm nearest neighbor search algorithm for vector quantization. IEICE Trans. Inf. Syst., E86-D: 600-663.

Sawada, A., T. Higashino, T. Oyabu, Y. Takei, H. Nanto and K. Toko, 2008. Gas sensor characteristics for smoldering fire caused by a cigarette smoke. Sen. Actuators B Chem., 130: 88-93.

Thuillard, M., 2002. A new flame detector using the latest research on flames and fuzzy-wavelet algorithms. Fire Safety J., 37: 371-380.

Toreyin, B.U., Y. Dedeoglu, U. Gueduekbay and A.E. Cetin, 2006. Computer vision based method for real-time fire and flame detection. Pattern Recognition Lett., 27: 49-58.

Wang, S.J., D.L. Jeng and M.T. Tsai, 2009. Early fire detection method in video for vessels. J. Syst. Software, 82: 656-667.

Yanqing, W., C. Deyun, S. Chaoxia and W. Peidong, 2010. Vision-based road detection by monte carlo method. Inform. Technol. J., 9: 481-487.

Yu, F.X., J.Y. Su, Z.M. Lu, P.H. Huang and J.S. Pan, 2008. Multi-feature based fire detection in video. Int. J. Innovative Comput. Inform. Control, 4: 1987-1987.