FIRE DETECTION IN H.264 COMPRESSED VIDEO

Murat Muhammet Savcı[⋆] *Yasin Yıldırım* [⋆] *Görkem Saygılı* [†] *Behçet Uğur Töreyin* [⋆]

[⋆] Informatics Institute, İstanbul Technical University, Maslak, İstanbul, Turkey
[†]Department of Biomedical Engineering, Ankara University, Ankara, Turkey.

ABSTRACT

In this paper, we propose a compressed domain fire detection algorithm using macroblock types and Markov Model in H.264 video. Compressed domain method does not require decoding to pixel domain, instead a syntax parser extracts syntax elements which are only available in compressed domain. Our method extracts only macroblock type and corresponding macroblock address information. Markov model with fire and non-fire models are evaluated using offline-trained data. Our experiments show that the algorithm is able to detect and identify fire event in compressed domain successfully, despite a small chunk of data is used in the process.

Index Terms— fire detection, compressed domain, macroblock type, H.264/AVC

1. INTRODUCTION

Today, the majority of computer vision applications in video analysis are carried out by detecting and extracting compact and robust features in pixel domain. With the recent improvements in imaging and camera sensor technologies in terms of video resolution, the amount of data in computation follows an increasing trend. This increases the computational burden of real-time video processing applications such as object detection, tracking and retrieval. On the other hand, computational overhead due to considerable amount of data in the pixel domain can be effectively reduced by transferring the computation to the compressed domain. Performing analysis in the compressed domain also eliminates the computation cost of decoding phase, which is necessary in conventional pixel level analysis. In this work, we propose a novel fire detection method in H.264 compressed domain by modelling transitions between selected groups of macroblock types and parameters using a Markov model.

With the increasing use of compressed data over the past decades, compressed video analysis has been applied on various vision tasks including moving object segmentation [1, 2, 3], person detection [4], object tracking [4, 5] and face detection [6]. In [1], a moving object detection method in wavelet compressed domain was presented by comparing the wavelet transform (WT) of adjacent frames without computing inverse WT's of the image frames.

Laumer et al. [3] proposed an algorithm to detect moving regions within video frames by extracting and parsing macroblock types, partition modes and quantization parameters (QP). Zhao et al. [7] proposed a moving object classification method in HEVC compressed domain based on a codebook learning of spatio-temporal HEVC syntax words. They first detected moving regions by performing motion vector (MV) interpolation for intra-coded prediction unit followed by removal of the MV outliers and grouping of non-zero MVs. After the moving object segmentation step, they used prediction modes, length of MVs and MV difference as HEVC syntax features to construct a bag of words learning model. Our method, however, exploits only temporal macroblock patterns of moving fire regions in H.264 domain, which has a less computational complexity. An extensive survey about the advances in fire detection algorithms was presented by Çetin et al. [8], in which both the pixel level and compressed domain algorithms as well as the uses of different types of sensors and hardware were explored.

A recent survey on compressed domain video analysis was done by Babu et al. [9] which included more extensive information about the advances through the past decade. In [10], hidden-Markov model based fire detection algorithm was proposed by means of observing rapid temporal variation over the boundaries of flame regions in a video. Apart from fire detection in compressed domain, Benazza et al. [11] proposed a smoke detection method in MJPEG and MPEG2 videos by exploiting the recursivity of the DCT coefficients with respect to block size. More recently, a hidden-Markov model based method for fire detection in wavelet compressed domain was proposed in [12]. In [13], an experimentally-defined reduced complexity deep Convolutional Neural Network (CNN) architectures were proposed for fire detection of a video or an image without using any temporal information.

In contrast, our work is focusing on H.264 compressed domain by exploiting the macroblock transition patterns of fire motion by means of a Markov model. In this work, our main contribution is a novel fire detection method for the H.264 compressed domain using only temporal macroblock patterns.

This work is in part funded by TÜBİTAK 114E426 and İTÜ BAP MGA-2017-40964.

The rest of the paper is organized as follows: Section 2 introduces the related syntax of the H.264/AVC standard. Section 3 describes the learning structure of the Markov model. In Section 4, we discuss the performance of the proposed method in the experiments. We draw our conclusions and elaborate on the future improvements in Section 5.

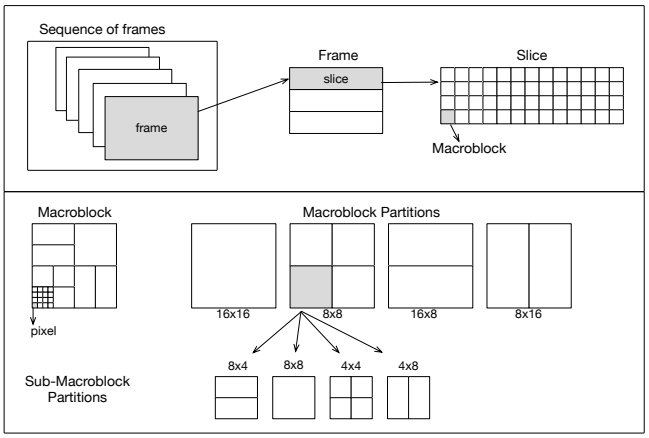

Fig. 1. Macroblock structures of the H.264/AVC standard.

2. SYNTAX OF H.264/AVC

This section gives a brief overview of H.264/AVC which provides only the relevant syntax elements of this work. H.264/AVC is a block-based video compression standard. In H.264/AVC, rather than encoding a picture as a whole, each frame is divided into square blocks and each of these blocks is encoded independently. These blocks are known as macroblocks, which consist of 16x16 pixels. Several numbers of macroblocks grouped together to shape a slice. Each frame has at least one slice. H.264 defines five slice types: I, P, B, SI and SP.

Every slice is coded as Intra Slice (I Slice), Predicted Slice (P Slice), or Bi-Directional Predicted Slice (B Slice). SI and SP slices also called switching slices can be used for transitions between H.264 video streams. Both are not very commonly used. A slice consists of several consecutive macroblocks. Each macroblock can be divided into smaller blocks of 4x4 pixels which are called sub-macroblocks. Fig. 1 gives a generic representation of the data elements in a video sequence.

3. METHOD

General scheme of the proposed method can be seen from the flowchart in the Fig.2. At first, we create training data from fire and non-fire motion videos. Macroblocks which contain partial fire information are trained offline to learn the temporal characteristics of fire, mainly at the fire borders. Specifically, the procedure is used to capture fire flicker process.

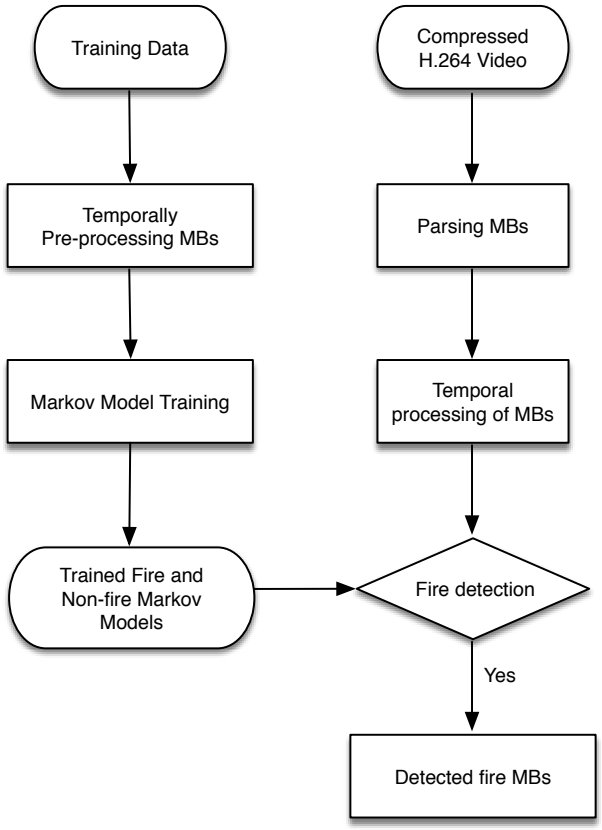

Fig. 2. Overall flow chart of the proposed method.

Furthermore, non-fire macroblocks are also trained in a similar fashion to separate real fire from non-fire moving objects. Thereafter, a seven states Markov model is used in an effort to calculate the transition probabilities between specific macroblock types to capture temporal characteristics of the fire and non-fire models. That concludes the method to create fire and non-fire Markov models.

Next, compressed H.264 videos are parsed and only the syntax elements which are related to this work are extracted. Extracting the syntax elements is achieved by integrating the JM software, which is the reference software for H.264/AVC video standard.[14]. As this work solely focuses on the macroblocks, only macroblock syntax elements and corresponding macroblock addresses are extracted. Then, every macroblock address and its type information are parsed frame by frame throughout the video. Hence, the temporal processing of macroblock type information is obtained for the next phase, which is the detection of fire.

In fire detection process, specific macroblock types are set to seven Markov states to determine whether a macroblock is in the fire or non-fire type. Table 1 shows all the states and their corresponding macroblock types that we proposed in this work. S1 and S2 states are classified for Intra-frame predicted macroblock types I_4x4,I_16x16 which are available in I and

Table 1. Markov Model States Table

State No	Macroblock Type
State 1	I_4x4
State 2	I_16x16
State 3	P_8x8
State 4	{P_8x16, P_16x8}
State 5	P_16x16
State 6	P_SKIP
State 7	Other MB Types

P slices. In this work, we only consider H.264 Baseline profile in which I and P slices are possible whereas B slices are not allowed. Intra-coded macroblocks are available in both I slices and P slices. If relevant information does not exist in previous frames, H.264 encoder uses I_4x4 and I_16x16 macroblocks. Thus, existence of I_4x4 and I_16x16 exhibits a fast motion in P slice.

S3, S4, S5 states are classified for macroblock types of P_8x8, {P_8x16, P_16x8}, and P_16x16, respectively. If there is a small change in the current frame compared to the previous frame, H.264 encoder uses 8x8, 8x16, and 16x8 macroblocks. This indicates the presence of a moving object in a particular area of the frame. In addition, P_16x16 macroblock also shows possible existence of a slow movement, so that H.264 encoder determines not to divide it into smaller sub-macroblocks.

S6 state is classified for P_SKIP category of a macroblock which indicates most likely no motion at all. Specifically, macroblock P_SKIP indicates that no additional data is available in that macroblock. H.264 Encoder decides if there is no difference between current and previous frames for a macroblock, it is set to a P_SKIP macroblock. S7 state is classified as the remaining macroblock types which are considered irrelevant to this work.

4. EXPERIMENTAL RESULTS

The experiments are performed on a Mac OSX 10.13.6 High Sierra computer with 3.40 GHz Intel Core i7-4770 CPU and 16 GB RAM. The performance of the proposed method is tested on several videos. Sample results of the fire detection yield from our method is given in Fig. 3. We classify a frame as fire frame if at least one mabroblock is detected as fire correctly.

Our algorithm can detect fire, especially fire flicker process which occurs at the borders of the fire. As a by product, some smoke elements is also detected, which exhibits similar behaviour like fire flicker process. Detection performance on different videos are presented in Table 2. In our algorithm, we used a window length parameter in an effort to evaluate the performance of the detection algorithm for different

Fig. 3. Sample fire detection result from the 'Video 3': MBs detected as fire are marked with red rectangles.

time frames. In fact, window length stores macroblock types that changes in every 8, 10 , 15, and 20 frames. Transition probabilities for every window length is calculated using offline trained data. If fire probability is higher than non-fire, then it is fire macroblock between that window length. Every macroblock is processed in this context to classify fire macroblocks. Table 2 shows 5 different videos which are evaluated using 4 different window length values.

In the first video, there is no fire information, thus our algorithm successfully skip all the frames for all window lengths. In second video, we have a scene that a man is walking and waves hitting to the beach. Our algorithm in this case, successfully skips non-fire information for window length 10, 15, and 20. However, when window length is 8, 248 frame falsely detected as fire macroblock due to wave motion behavior. Video 3, 4, and 5 contain only fire scenes. Our algorithm's performance on these videos varies over different window lengths which are depicted in Table 2.

We also perform quantitative analysis of the our algorithm on the macroblock level accuracy. The result of the macroblock level experiments is given in the Table 3. At first, for each frame k, we manually label ground truth of the set of macroblocks $S^{mb}[k]$ of the fire regions. Then, we compute *precision*, *recall* and *F-score* metrics for testing the performance of the proposed method as given in the following equations:

$$\mathcal{P} = \frac{S_{tp}^{mb}[k]}{S_{tp}^{mb}[k] + S_{fp}^{mb}[k]}, \tag{1}$$

$$\mathcal{R} = \frac{S_{tp}^{mb}[k]}{S_{tp}^{mb}[k] + S_{fn}^{mb}[k]}, \tag{2}$$

$$\mathcal{F} = \frac{2 \times \mathcal{P} \times \mathcal{R}}{\mathcal{P} + \mathcal{R}}, \tag{3}$$

where $\mathcal{P}$, $\mathcal{R}$ and $\mathcal{F}$ denote the precision, recall and *F-score*. Moreover, $S_{tp}^{mb}[k]$, $S_{fp}^{mb}[k]$ and $S_{fn}^{mb}[k]$ stand for the number of true positives (TP), false positives (FP) and false negatives (FN) in the set of macroblocks, respectively.

Table 2. Detection performance of the proposed method on different videos.

Video Sequences	Number of Frames	Number of Frames with Fire	Window Length	Number of Frames Detected as Fire	Description
Video 1	545	0	8	0	girl Walking Down
			10	0	
			15	0	
			20	0	
Video 2	840	0	8	248	man walking on a beach
			10	0	
			15	0	
			20	0	
Video 3	870	870	8	869	summer grass is burning
			10	777	
			15	499	
			20	293	
Video 4	465	465	8	465	wild brush and grass is burning
			10	465	
			15	465	
			20	397	
Video5	187	187	8	156	wood fire
			10	126	
			15	68	
			20	37	

Table 3. Macroblock level result of the proposed method.

Video name	TP	FP	TN	FN	Precision	Recall	F-score
Video 6	478	6	91449	59	0.99	0.89	0.94

Table 4. Comparison of our method with two other methods.

Method	TP	FP	TN	FN	Precision	Recall	F-score
FireNet [13]	698	841	544	824	0.45	0.46	0.46
InceptionV1-OnFire [13]	613	820	565	909	0.43	0.40	0.42
Ours	1368	248	1137	154	**0.85**	**0.90**	**0.87**

We compared our performance with two architectures in [13] using our test data. FireNet and InceptionV1-OnFire are both pixel domain, reduced complexity deep CNN architectures which are not using any temporal information. The results are presented in Table 4. Number of true positives, false positives, true negatives and false negatives are denoted as TP, FP, TN and FN, respectively. We used all of the five test videos in total of 2907 frames and evaluated the performances on the frame level. More precisely, our method detects a fire alarm based on if any macroblock is detected as fire on a frame. Our method performs significantly better than the two variations of the deep neural network architectures, FireNet and InceptionV1-OnFire, namely.

5. CONCLUSION

In this paper, we introduce a novel compressed domain fire detection algorithm using a Markov model. The temporal process extracts macroblock types, corresponding coordinates as well as macroblock addresses from H.264/AVC syntax elements. Macroblock type is temporally processed and fire and non-fire Markov models are developed. The proposed model detects especially fire flicker process using the model information. Simulation results show that even though we only use macroblock type from syntax elements, we managed to identify fire regions reliably within the H.264 videos without the necessity of a decoding phase. Furthermore, we measure our performance by comparing with FireNet and InceptionV1-OnFire which are pixel based, non-temporal, reduced complexity deep CNN architectures. Our compressed domain method shows considerably better performance than the other two pixel domain architectures. For the future work, the proposed system can be further improved by using the spatio-temporal information based on the DCT coefficients and MVs as a future work. We also plan to focus on the analysis of the additional syntax elements to achieve more accurate results in the future.

6. REFERENCES

[1] B Ugur Töreyin, A Enis Cetin, Anil Aksay, and M Bilgay Akhan, "Moving object detection in wavelet compressed video," *Signal Processing: Image Communication*, vol. 20, no. 3, pp. 255–264, 2005.

[2] Marcus Laumer, Peter Amon, Andreas Hutter, and Andre Kaup, "Compressed domain moving object detection by spatio-temporal analysis of h. 264/avc syntax el-

ements," in *Picture Coding Symposium (PCS), 2015*. IEEE, 2015, pp. 282–286.

[3] Marcus Laumer, Peter Amon, Andreas Hutter, and André Kaup, "Moving object detection in the h. 264/avc compressed domain," *APSIPA Transactions on Signal and Information Processing*, vol. 5, 2016.

[4] Philipp Wojaczek, Marcus Laumer, Peter Amon, Andreas Hutter, and André Kaup, "Hybrid person detection and tracking in h. 264/avc video streams.," in *VISAPP (1)*, 2015, pp. 478–485.

[5] Sayed Hossein Khatoonabadi and Ivan V Bajic, "Video object tracking in the compressed domain using spatio-temporal markov random fields," *IEEE transactions on image processing*, vol. 22, no. 1, pp. 300–313, 2013.

[6] Pedro Fonseca and Jan Nesvadba, "Face detection in the compressed domain.," in *ICIP*, 2004, vol. 3, pp. 24–27.

[7] Liang Zhao, Zhihai He, Wenming Cao, and Debin Zhao, "Real-time moving object segmentation and classification from hevc compressed surveillance video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 6, pp. 1346–1357, 2018.

[8] A Enis Çetin, Kosmas Dimitropoulos, Benedict Gouverneur, Nikos Grammalidis, Osman Günay, Y Hakan Habibolu, B Uğur Töreyin, and Steven Verstockt, "Video fire detection–review," *Digital Signal Processing*, vol. 23, no. 6, pp. 1827–1843, 2013.

[9] R. Venkatesh Babu, Manu Tom, and Paras Wadekar, "A survey on compressed domain video analysis techniques," *Multimedia Tools and Applications*, vol. 75, no. 2, pp. 1043–1078, Jan 2016.

[10] B Ugur Toreyin, Yigithan Dedeoglu, and A Enis Cetin, "Flame detection in video using hidden markov models," in *Image Processing, 2005. ICIP 2005. IEEE International Conference on*. IEEE, 2005, vol. 2, pp. II–1230.

[11] Amel Benazza-Benyahia, N Hamouda, Fethi Tlili, and Safa Ouerghi, "Early smoke detection in forest areas from dct based compressed video.," in *EUSIPCO*, 2012, pp. 2752–2756.

[12] Behçet Uğur Töreyin, "Smoke detection in compressed video," in *Applications of Digital Image Processing XLI*. International Society for Optics and Photonics, 2018, vol. 10752, p. 1075232.

[13] Andrew J Dunnings and Toby P Breckon, "Experimentally defined convolutional neural network architecture variants for non-temporal real-time fire detection," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 1558–1562.

[14] "H.264/AVC JM Reference Software," aug 2008.