Motion Vector Based Features for Content Based Video Copy Detection

Kasım Taşdemir, A. Enis Çetin
Department of Electrical and Electronics Engineering, Bilkent University
{*tasdemir,cetin*}*@ee.bilkent.edu.tr*

Abstract

In this article, we propose a motion vector based feature set for Content Based Copy Detection (CBCD) of video clips. Motion vectors of image frames are one of the signatures of a given video. However, they are not descriptive enough when consecutive image frames are used because most vectors are too small. To overcome this problem we calculate motion vectors in a lower frame rate than the actual frame rate of the video. As a result we obtain longer vectors which form a robust parameter set representing a given video. Experimental results are presented.

1 Introduction

In Content Based Copy Detection (CBCD) algorithms, average color, color histogram, SIFT parameters, and motion are used as feature parameters [2, 7, 8, 4, 3, 6]. When a movie is recorded from a movie theater by a hand-held camera, then its color map, fps, size and position change and edges get soften. Color based algorithms will have difficulties detecting the camera recorded copy of an original movie because the information it depends on is significantly disturbed. However, motion in a copied video remains similar to the original video.

Motion information was considered as a weak parameter by other researchers [3]. This is true when motion vectors are extracted from consecutive frames in a video with a high capture rate. In a typical 25 Hz captured video most motion vectors are small or close to zero and they may not contain any significant information. In this article, we calculate motion vectors in a lower frame rate than the actual frame rate of the video. As a result we obtain longer vectors which form a robust parameter set representing a given video.

The goal of this article is not to develop a complete CBCD method solely based on motion vectors but to show that motion vectors are descriptive feature parameters of a given video clip. It should be pointed out that a complete CBCD system should be capable of fusing the information coming from motion, color and SIFT feature sets in an intelligent manner to reach a decision.

2 Motion Vector (MV) Extraction and Feature Sets

In general, motion vectors are extracted using consecutive frames in many video analysis and coding methods [3]. In order to capture the temporal behavior more efficiently we use every t^{th} and $(t+n)^{th}$, $n > 1$ frames instead of the traditional approach of using t^{th} and $(t+1)^{th}$ frames. In our approach, we use every t^{th} and $(t+n)^{th}$ frame for motion vector extraction. For example, human movements change slowly in a 25 fps video. If two consecutive frames are used in motion vector extraction step, resulting motion vectors will have small values because of the high capture rate of the video. Furthermore, some of the image-blocks (or macro blocks) inside the moving object may be incorrectly assumed as stationary or moving in an incorrect direction by the motion estimation algorithm because similar image blocks may exist inside the moving object. By computing the MVs using every n-th frame ($n > 1$) it is possible to get more descriptive motion vectors.

We define the mean of the magnitudes of motion vectors (MMMV) of macro blocks of a given frame as follows:

$$MMMV(k) = \frac{1}{N} \sum_{i=0}^{N-1} r(k,i) \qquad (1)$$

where $r(k,i)$ is the motion vector magnitude of the macro block in position i of k^{th} frame, and N is the number of macro blocks in an image frame of the video. We also define the mean of the phase angles of motion vectors (MPMV) of macro blocks of a given frame

1051-4651/10 $26.00 © 2010 IEEE
DOI 10.1109/ICPR.2010.767

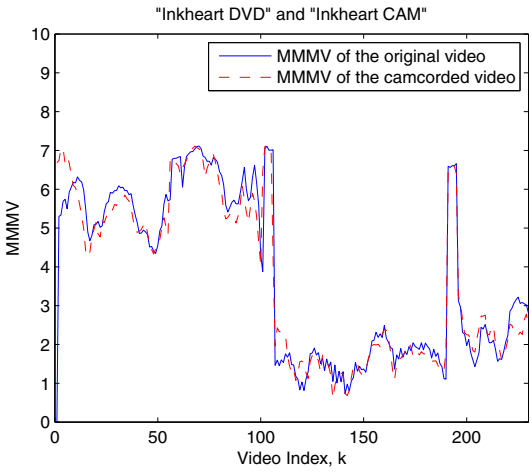

Figure 1. Similarity of the MMMV plots of "Inkheart DVD" and "Inkheart CAM", (with n=5).

(MPMV) as follows:

$$MPMV(k) = \frac{1}{N}\sum_{i=0}^{N-1}\theta(k,i), \qquad (2)$$

where $\theta(k,i)$ is the motion vector angle of the macro block in position i of the k^{th} frame of the video, and N is the number of macro blocks. The angle θ is in radians and $\theta \in (-\pi, \pi)$. So, the range of $MPMV$ is also in the same region: $MPMV(.) \in (-\pi, \pi)$.

We use discrete MMMV(k) and MPMV(k) functions as the feature sets representing a given video. Example MMMV plot is shown in Fig. 1. Storage requirement is low as both functions require a single real number for each frame k of the video. It is possible to divide the image frames into subimages and extract MMMV and MPMV values for each subimage but we experimentally observe that a single value for a given frame may be for most typical video clips.

3 Video Copy Detection Using MMMV and MPMV

This section investigates the similarity of MMMV-MPMV data of original movies and their hand-held camera versions. Test videos have different size and fps. Videos with CAM extensions are copies obtained using a hand-held camera. In Sec. 4 we present extensive comparisons using a video database used in [4, 1]. Although the original and hand-held camera recorded videos have different fps and size, they have similar MMMV plots as shown in Fig. 1. Original movie and its hand-held camera recorded version from a movie theater show significant similarities. The MVs are computed with a frame difference of n=5. In order to obtain a value that gives information about how much two movies resemble each other, the absolute different is calculated as distance, D. If frame sizes of the videos are different, motion vectors of videos will be also different. The video with a larger frame size will have larger motion vectors. In order to solve this problem we first normalize the MMMV and MPMV of the videos before making a comparison as follows:

$$\overline{MMMV} = \frac{MMMV - \mu_{MMMV}}{\sigma_{MMMV}} \qquad (3)$$

where μ_{MMMV} is the mean and σ_{MMMV} is the standart deviation of the MMMV array, respectively.

The sum of absolute values of difference of normalized MMMV values of two videos o-original and c-copy are calculated as the distance $D(o,c)$ as follows:

$$D(o,c) = \frac{1}{N}\sum_{t}\min_{|d|\le W}|\overline{MMMV}_o(t) - \overline{MMMV}_c(t+d)| \quad (4)$$

where W is the search window width. Experimentally, we time align the videos and select W as 2 frames because the fps of most commercial videos are between 20 and 30. In Eq. 4, N is the number of frames in the movie $MMMV_o$. If the original and the candidate video have different fps, then their frame indices corresponding to the same time instance should be calculated first. So, instead of comparing corresponding frame indexes, the aim is to compare image frames corresponding to the same time instant.

3.1 Most Active MBs In The Video Frames

Some MVs do not represent an actual motion, because in a moving object the vectors inside the object may point out arbitrary directions instead of the actual direction of the object. This is due to the fact that in an object pixel values of the neighboring macro blocks are almost the same. Therefore, we assume that the most meaningful information is in fast moving regions. Thus, we developed a method that takes the most active regions into account in a given frame instead of using all motion information as in Section 3. We applied the same algorithms in Equations 1, 2 except that we used most active α-percent of MVs where $\alpha \in (0, 100)$. $MMMV_s$ and $MPMV_s$ methods use first α-percent most moving of MVs and they are defined as

$$MMMV_{max}(k) = \frac{1}{\lceil N\frac{\alpha}{100}\rceil}\sum_{i=0}^{\lceil N\frac{\alpha}{100}-1\rceil} r_s(k,i) \qquad (5)$$

where $r_s(k,.)$ is the array of first α-percent of highest MV magnitudes of the frame k, N is the number of macro blocks and

$$MPMV_{max}(k) = \frac{1}{\lceil N\frac{\alpha}{100}\rceil} \sum_{i=0}^{\lceil N\frac{\alpha}{100}-1\rceil} \theta_s(k,i) \quad (6)$$

where $\theta_s(k,.)$ is the array of first α-percent of the highest MV angles of the k-th frame of the video.

4 Experimental Results

A video database is available in [1]. Original videos in this database are compared with the transformed versions of the same videos. There are 47 original videos taken from [1]. Duration of the videos are 30 seconds. Each video has eight different transforms such as changing contrast, zooming in and out, cropping, logo insertion, resizing to letter-box format and adding Gaussian noise. As a result there are a total of $47 \times 9 = 423$ videos in the database. For each parameter set 1457 comparisons are performed.

Original videos are compared with test videos in the database and its 8 transformations. For each test, the list of distance between the compared videos are calculated using Eq. 4 for different parameters or feature set types.

The performance of each test is plotted using its receiver operating characteristics (ROC) curve. The ROC curve is a plot of false positive rate F_{pr} and false negative rate F_{nr}. Let F_p and F_n the number of false positives and false negatives. False positive and negative rates are defined as

$$F_{pr}(\tau) = \frac{F_p}{N_p}, F_{nr}(\tau) = \frac{F_n}{N_n} \quad (7)$$

where N_p and N_n are the number of maximum possible false positive and false negative detections. Threshold is τ and its value is varied from 0 to its maximum value with an increment of 1%.

Effect of varying the frame skipping parameter n in motion vector extraction step is shown in Fig. 2. We can obtain more descriptive features of videos based on motion vectors if we use every 5^{th} frame instead of the current and the next frame in motion estimation step. As it is shown in Fig. 2 there is a dramatic increase in detection ratio with increasing n to 5.

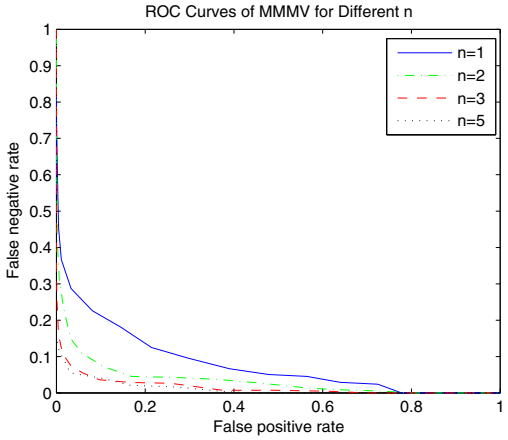

Figure 2. Effect of varying n on $MMMV$ plots.

We test the effect of using upper $\alpha\%$ of magnitudes of motion vectors. As it is seen in Fig. 3 increasing α increases the detection rate of the tests. The area under the ROC curve in Fig. 2 is 0.0115 when $n = 5$, and the area under the ROC curve in Fig. 3 is 0.0091 when $\alpha = 50$. Therefore, the use of upper 50% of the MVs does not significantly effect the accuracy. Instead of using all MVs, upper 50% of the MVs can be used in the MMMV algorithm. In other words, only large motion vectors can be used in practice.

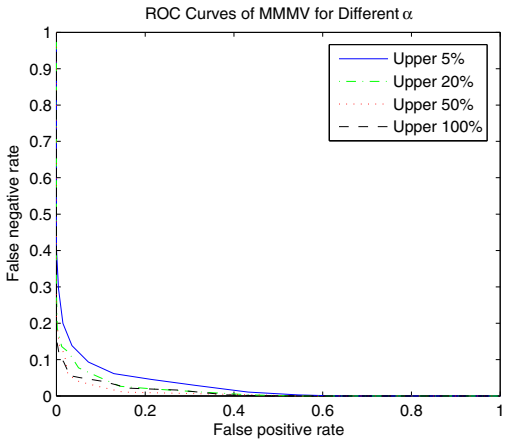

Figure 3. Effects of using different α for MMMV, $n = 5$.

As shown in Table 1 using upper 25% for n=1 and 50% for n=5 is closer to the ideal case. Best results are obtained when n=5 and α= 50% of Mvs. The area under

Table 1. The area under the ROC curves of MMMV for different α and n.

α	15%	25%	50%	100%
n=1	0.0611	**0.0577**	0.0599	0.0807
n=5	0.0205	0.0138	**0.0091**	0.0115

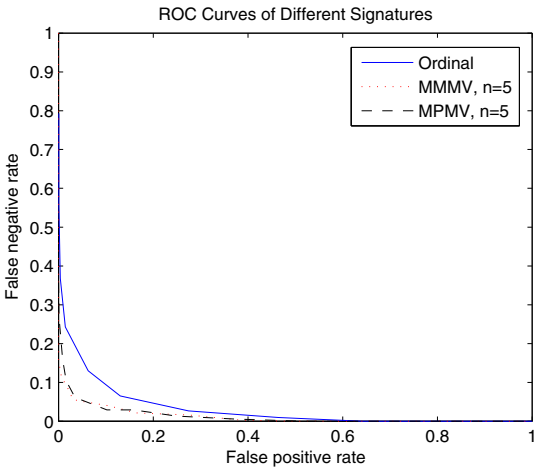

Figure 4. The ROC curve of the *ordinal* signature and the ROC curves of proposed methods when the frame difference parameter $n = 5$..

the ROC curve of MPMV is 0.0128 when $n = 5$, which is comparable to the ROC curve of MMMV.

In [3] it is stated that *ordinal signature* introduced in [5] outperforms the *Motion Signature*. This is true when the motion vectors are extracted using the current and the next frame ($n = 1$). On the other hand, if motion vectors of videos are extracted using every 5^{th} frame, motion vector based MMMV plot is closer to the ideal case than the ROC curve of ordinal signature as shown in Fig 4. The area under the ROC curve of *ordinal signature* is 0.0311 which is higher than the area under the ROC curves of both MMMV and the MPMV signatures.

As pointed above the ROC curves of the proposed MMMV and MPMV methods are very close to each other as shown in Fig. 4. It is experimentally shown that the MMMV and the MPMV are good descriptive features for videos. In this database the best results are obtained with alpha=50% and the frame difference parameter $n = 5$.

5 Conclusions

In this article, it is experimentally shown that motion vectors are unique signatures of videos. Motion vectors can be used in similar video detection or CBCD algorithms. In order to obtain reliable signature vectors for all videos motion vectors of the current and the next n^{th} frame ($n > 1$) are used in motion vector estimation algorithms. Resulting motion vectors provide a reliable representation for all types of videos. The proposed motion-based feature parameters are resistant to illumination and color changes in video.

Another important comparison criteria of the CBCD algorithms in terms of the practical results is the size of the feature set in a database. The MMMV and the MPMV information do not occupy much space in the database. They both occupy one byte (one feature) per frame in the database.

References

[1] Alexis Joly. Internet archive movie database, 2009. [Online; accessed 10-August-2009; `http://www.worldscinet.com/ijprai/ijprai.shtml`].

[2] A. Hampapur and R. Bolle. Comparison of distance measures for video copy detection. In *Multimedia and Expo, 2001. ICME 2001. IEEE International Conference on*, pages 737–740, Aug. 2001.

[3] A. Hampapur, K. Hyun, and R. M. Bolle. Comparison of sequence matching techniques for video copy detection. In M. M. Yeung, C.-S. Li, and R. W. Lienhart, editors, *Storage and Retrieval for Media Databases 2002*, volume 4676, pages 194–201. SPIE, 2001.

[4] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In *CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 371–378, New York, NY, USA, 2007. ACM.

[5] R. Mohan. Video sequence matching. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 6, pages 3697–3700 vol.6, May 1998.

[6] M. R. Naphade, M. M. Yeung, and B.-L. Yeo. Novel scheme for fast and efficent video sequence matching using compact signatures. volume 3972, pages 564–572. SPIE, 1999.

[7] L. Teodosio and W. Bender. Salient video stills: content and context preserved. In *MULTIMEDIA '93: Proceedings of the first ACM international conference on Multimedia*, pages 39–46, New York, NY, USA, 1993. ACM.

[8] Y. Tonomura and S. Abe. Content oriented visual interface using video icons for visual database systems. In *Visual Languages, 1989., IEEE Workshop on*, pages 68–73, Oct 1989.