VIDEO SEQUENCE MATCHING

Rakesh Mohan

IBM T.J. Watson Research Center
PO Box 704
Yorktown Heights, NY 10598
rakesh@watson.ibm.com

ABSTRACT

We present a novel scheme to match a video clip against a large database of videos. Unlike previous schemes that match videos based on image similarity, this scheme matches videos based on similarity of temporal activity, i.e., it finds similar "actions." Furthermore, it provides precise temporal localization of the actions in the matched videos.

Video sequences are represented as a sequence of feature vectors called fingerprints. The fingerprint of the query video is matched against the fingerprints of videos in a database using sequential matching. The fingerprints are computed directly from compressed MPEG videos. The matching is much faster than real-time. We have used this scheme to find similar actions in sporting events, such as diving and baseball.

Keywords: ***video matching, video search, video databases.***

1. INTRODUCTION

Video is increasingly available in digital form. Digital video is now broadcast into millions of homes (DSS etc.), it is delivered over the Internet, and the High Definition TV (HDTV) format is digital. This has spurred work on applications dealing with digital video, such as databases, video browsing systems, video analysis, editing, and administration. Video matching is a key component of many of these applications.

Most video matching schemes first reduce videos to a small set of key-frames [7][8][9][12], and then use image matching schemes to match the key frames [1]. These schemes have some drawbacks. Firstly, as they depend on the detection of edits to segment video into shots [4][10][12], if an edit is missed, such schemes may fail. Secondly, it is not clear as to which image should be used as the key-frame for a shot [11]. Thirdly, and most importantly, such schemes largely ignore the "action" within a video. This drawback has been addressed to some extent by including some motion information (for example, from MPEG motion vectors) with the key frames [5].

Videos, unlike images, capture actions that occur over a period of time. Therefore, in querying video databases, one can pose queries directed at this temporal activity. For example, one may wish to locate all the dives in a diving competition or all the hits in a baseball game. Such queries can not be answered solely based on image similarity; we need the notion of "action similarity." In this work, we address the issue of matching videos on the basis of similar actions.

We encode the sequence of frames of a video clip as a sequence of feature vectors, which we call a fingerprint. The action that gets encoded in video as a sequence of differing images is also captured in the fingerprint as a sequence of features. The fingerprints are directly computed from compressed video and the video sequences are matched based on Euclidean distance between the fingerprints.

The paper is organized as follows. In Section 2 we present video sequence matching, temporal localization and fingerprints. In Section 3 we present some experiments. In Section 4 we present our conclusions and propose future directions.

2. VIDEO SEQUENCE MATCHING

An action is a pattern of activity occurring over a period of time. Thus, actions have a specific temporal sequence and temporal extent. When an action is filmed, it is captured as a sequence of images; each image slightly different. Matching videos on similar action then maps to matching a sequence of video frames to another sequence of video frames. We call this *video sequence matching*. A video sequence has two important temporal properties: (1) the specific ordering of the frames in time (2) the length of the sequence. While the length of the sequence is actually in time, it is convenient talk of it terms of number of frames.

We define the problem of video sequence matching as that of determining if a match for a given video sequence appears in another video, and if so at what location. Formally, given a query video sequence $X = x_1 x_2 \cdots x_n$. x_i = i^{th} frame of X, and a database video $Y = y_1 y_2 \cdots y_m$. y_i = i^{th} frame of Y, and $Y(i) = y_i y_{i+1} \cdots y_{i+n-1}$ a consecutive sub-sequence of frames in Y, we say that $X \approx Y(i)$ or X matches Y at i if the video sequence $Y(i)$ matches the video sequence X.

We call the determination of the exact location i in Y where X matches, as *temporal localization*. Just as in image matching, spatial localization allows us to place objects within an image, in video matching, temporal localization is important for locating actions precisely in time, even when the surrounding frames are visually similar (for example, when the shot in Y containing frame i, is much longer than X). Video matching schemes based only on key-frames cannot provide temporal localization.

A central thesis of our work is that for matching actions (i.e. for video sequence matching) we need a representation that emphasizes the temporal description of the video and de-emphasizes the visual description of each frame. In other words, we can use a representation for each frame that is so compact as to be essentially useless for matching images, but when a

representation for the sequence is built from it, it is sufficient to distinguish among a huge number of sequences. We call such a representation, a *fingerprint* of the video sequence.

2.1 Features

As we wish to match actions, the length of a video sequence, and consequently its representation, depends on the temporal extent of the action. In most video matching schemes, the lengths of the video segments in the database are predetermined: they are either chopped up into fixed length segments or at shot boundaries [1][5][11]. Since different actions have different temporal extent, the videos in a database can not be pre-segmented. For purposes of efficiency, it is therefore desirable, that the representation of a video sequence $Y(i) = y_i y_{i+1} \cdots y_{i+n}$, from a video Y, does not require re-computation if i or n change. This implies that each frame be represented independently of other frames and the representation of a video sequence be easily derived from the representation of its constituent frames.

This notion of using the full video for representation is similar in spirit to the encoding of the time varying TV signal in systems used by TV ratings agencies for monitoring commercials [3][6].

We define the fingerprint of a video sequence as a vector of some representation of its constituent frames. The fingerprints a and $b(i)$ on X and Y respectively are defined as

$$a = \left[f(x_1), f(x_2), \cdots, f(x_n) \right],$$

$$b(i) = \left[f(y_i), f(y_{i+1}), \cdots, f(y_{i+n-1}) \right], \quad 1 \le i \le m - n + 1$$

Different fingerprints can be defined based on different choices for f. One f that we have used is the *ordinal measure* [2] of a reduced intensity image of x. First, frame x is reduced, via averaging, to an $u \times v$ intensity image x'. Then, an ordinal measure of x' is computed as a vector $a = \left[a_1, a_2, \cdots a_{uv} \right]$ of the ranks of the intensity values of x' scanned in row-major order. We get $f(x) = a$ and $f(y) = b$, where $|a| = |b| = uv$.

In our experiments we have used $u = v = 3$. The size of the 3x3 fingerprint for each frame is 4bits*9 or 4.5 bytes. Thus the size of a 3x3 fingerprint for a video sequence of length n is $4.5n$ bytes. The rationale for using this feature runs as follows. The frames within a short sequence have similar global features since they depict the same scene. Thus position independent global visual features such as intensity or color histograms are not useful for capturing action. As objects move, the colors and intensity associated with them move also. Thus a position dependent representation is required. A reduced image would provide such a representation. However, the intensity and color values for all pixels in the 2x2 or 3x3 reduced image are all nearly equal to the average for that frame. By using ordinal values instead, we are able to differentiate between these close values. In the future, we plan to investigate features that use some motion information in order to emphasize the moving parts of the frames.

All our experiments were conducted on MPEG-1 encoded video. The MPEG-1 video was first reduced to all **D**-frames using a technique by Yeo [10]. The D-frames are the DC coefficients of the frames. The DC coefficients for the **I** frames are obtained directly, while those of the **B** and **P** frames are obtained by performing motion compensation only on the DC coefficients. Since this technique does not require complete decompression, it works faster than real-time on most Pentium based PC's.

The frame size of MPEG-1 is 352x240. The **D**-frames are a reduced 44x30 representation for the **Y** component and 22x15 for the **U** and **V** components. These **D**-frames are then used for computing the fingerprints as outlined above.

Time taken for computing the D-frames (Y,U and V together) is is 30 minutes for each hour of video, or twice real time. Time to compute the 3x3 fingerprint is 1 minute for 1 hour of video (all times on a Pentium II 200 Mhz).

2.2 Matching

A video database is prepared for video sequence matching by first computing a compact representation, such as the 3x3 ordinal measure described above, for each frame of each video in the database. This operation needs to be performed only once. The query for locating an action comes as a video sequence X, of length n, depicting the action. A fingerprint of X is computed. X is matched to a video Y, of length m, in the database by matching it against each sub-sequence of consecutive n frames, i.e. X is matched against $Y(1)$ then $Y(2)$ and so on until $Y(m-n+1)$. This is repeated for each video in the database.

The sequence X is matched to a sub-sequence $Y(i)$ by computing the distance between their fingerprints. This distance is defined as:

$$D(a, b(i)) = \left(\sum_{j=1, k=i}^{j=n, k=i+n-1} d(f(x_j), f(y_k)) \right) / n$$

For the ordinal measure based fingerprint, we use

$$d(f(x), f(y)) = d(a, b) = \sum_i \left(|a_i - b_i| \right)$$

The database videos are not presegmented as there is no simple analogue to detection of shot boundaries to detecting boundaries of action. Therefore, we have to match X against all possible successive sequences of n frames in Y. This gives us m-n+1 sequences to match. The best possible scheme would need to consider at least m/n sequences. As we slide X along Y, we get a sequence of m-n+1 distance values. Next we detect the local minimas in these distance values. We then we suppress any minimas that are less than n units apart in the sequence as they come from sub-sequences in Y that overlap by at least one frame.

In our implementation, we have found it better to detect maximas in $1/(D(a, b(i)))$. The resulting minimas (or maximas of the reciprocals) are then sorted to give an ordering to the matches. As in other image and video databases, the top choices are then returned to the user.

In the future, we plan to explore (1) changing the sampling of the frames (2) matching between dissimilar length sequences. We currently employ different sampling of frames to reflect the frame rate of the capture.

Video sequence matching exhibits tolerance to differences in time an action in a database video takes to that in the query video. In our experiments (Section 3) we have found that actions that are longer or shorter by even a factor of two are still identified as matches.

Time to match a 2.5 second sequence (75 frames) against one hour of video is 12 seconds.

2.3 Fingerprint Evaluation

Video sequence matching is geared towards videos with a high activity content: i.e. where the frames are changing rapidly (and therefore key-frame based matching is not appropriate). The selectivity and temporal localization capability of a fingerprint then depends on the variability between the features from adjacent frames. One measure for evaluating sequences for suitability for matching via video sequence matching, and for evaluating the features used to form the fingerprints, is the frame to frame feature variation in a fingerprint.

$$\text{Variability measure } V = \left(\sum_{i=1}^{t-n-1} d\left(f(x_i), f(x_{i+1})\right)\right) / n$$

Based on this measure V, we have been able to check if there is enough visible activity in a sequence that gets picked by a feature to be used in video sequence matching. For the fingerprint presented in the previous section, we get the following results. For talking heads, this measure lies below 0.3. For activities such as diving (see Figure 1) this measure is > 2.0. In videos of baseball games (see Figure 2), while there is a lot of activity when a ball is hit, the camera view is wide and the activity takes up a very small part of the frame. Here the measure lies between 0.5 to 1.0, indicating that the ordinal based fingerprints may not be suitable.

3. EXPERIMENTS

The primary application we considered was that of providing tools for summarizing videos of sporting events. Often the interesting and exciting parts of the game are shown again in replays. These replays can be at the same speed or in "slow-motion" which is usually at half-speed. Our technique for detecting replays is to take a video sequence, subsample it by 2 and match its fingerprint against the video sequence preceding it.

Also, in many sporting events, the interesting actions are repeated, for example, dives in a diving competition or hits in a baseball games. In our experiments, we searched for the best matches for a particular action (dives, hits) that we wished to detect.

One video we used was of a 1996 Olympic diving competition. The video is 23 minutes long. There are 8 dives in this video and there is a slow-motion replay of six of the dives. A 5-second sequence (150 frames) showing a dive in slow motion was used as the query. The sequence starts at the point where the diver takes off the board and ends at the point of entry in the water. It was down sampled by two in time: i.e. the fingerprint was computed at every other frame.

We used the video itself as the database. The query video sequence is matched against every 2.5 seconds long (75 frames) video sequence, or 36,309 sequences. We used a fingerprint based on the ordinal values of 3x3 reduced intensity image for each frame. Thus, the query and the database sequence fingerprints were each of size 3x3x75.

The top match is the original, real-time dive of which we used the slow motion. In the top 5 matches, all sequences are correctly identified dive sequences. In the next 5 matches, only one is not a diving action (match # 8), and one is of a dive but not precisely located temporally (match # 10). The top 6 matches are all real-time (original) dives, the remaining two correct matches are to slow-motion sequences.

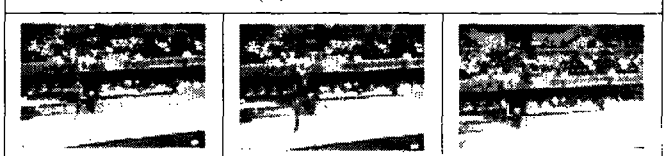
Some of the matched sequences are reverse angle shots and show different types of dives. Each of the matched diving sequences shows precise temporal localization; the matched sequences start exactly at the time of take of from the board, even though the actual video often shows the divers on the board and approaching for the dive for 2 to3 seconds. That is, if we look at the shots, they show the diver on the board for a time period comparable to the actual dive. This indicates that the video sequence matching algorithm is able to precisely locate the action within similar visual scenes.

(a) Query (slow motion replay)

(b) First match, original of replay above

(c) Second match

(d) Third match



(e) Seventh match, slow-motion, reverse angle

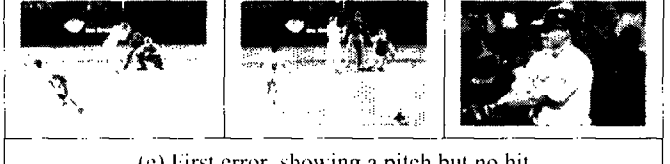
Figure 1. Diving sequence. Original video is 23 minutes, each is sequence 2.5 seconds

(a) Query sequence showing pitch + hit

(b) First match

(c) Second Match

(d) Third Match



(e) First error, showing a pitch but no hit

Figure 2. A Baseball match. Video is 24:35 minutes long. Each sequence is 3.3 seconds long.

From these top matches, the real-time dive sequences can be easily picked. This gives us a summary video short enough (< 30 sec) to be included in a web page for display over the Internet.

We ran another set of experiments with a video of a baseball game. Our goal was to identify the parts where a hit occurred. We selected a video sequence 100 frames or 3.3 seconds long, depicting a pitch and a subsequent hit, as the query. Fingerprints were 3x3 ordinal values of reduced **Y D**-frames. Each frame of the sequence was used for the fingerprint, so each fingerprint was of size 3x3x100. The baseball video, used as the database, was 24:35 minutes long.

A total of 44,131 sequences of 100 frames are matched. In the top 5 matches, 3 are video sequences correctly identified as pitches followed by hits. The 2 errors show the pitcher on the mound with the batter facing. In the next 5 matches only one more correct match was found. This may indicate that the 3x3 ordinal based fingerprint may not be the ideal feature when the action occupies only a small part of the frame.

The temporal localization is excellent. In each of the correctly identified sequences, the ball left the pitchers hand within 10 frames of each other.

4. CONCLUSIONS

We have presented a novel scheme for matching videos based on similarity of "actions." The computation of the fingerprint feature of a sequence, and the matching, are both fast. The scheme shows precise localization of actions in time. We believe that matching of actions will complement key-frame based visual matching in video databases.

There are certain shortcomings in the current work. The ordinal based fingerprints do not emphasize motion but capture the whole frame. The matching is not robust to change in viewpoint; a side view of a dive will not match a top-view of a dive in the current scheme. Future directions in research will attack these topics.

5. AKNOWLEDGEMENTS

The project is partially funded by a grant from NIST ATP. We would like to thank Boon-Lock Yeo for providing the software to compute DCT frames.

6. REFERENCES

[1] E. Ardizzone, et. al., "Content-based indexing of image and video databases by global and shape features." *Proc. Of the International Conference on Pattern Recognition*, 1996.

[2] D.N. Bhat and S.K.Nayar, "Ordinal measures for visual correspondence," to appear *IEEE-PAMI*.

[3] M.D. Ellis, et. al., (The Arbitron Company), U.S. Patent 5436653, "Method and system for recognition of broadcast," issued July 1995, (filed April 1992).

[4] A. Hampapur, R. Jain and T. Weymouth, "Production model based digital video segmentation," *Multimedia tools and applications*, vol. 1, pp. 9-46, March 1995.

[5] V. Kobla and D. Doerman, "Compressed domain video indexing techniques using DCT and motion vector information in MPEG video," *SPIE, Vol 3022*, Feb 1997.

[6] J.G. Lert Jr., D. Lu, (A.C. Nielsen Company), U.S. Patent 4677466, "Broadcast program identification method and apparatus," issued June 1987, (filed July 1985).

[7] B.C. O'Connor, "Selecting key frames of moving image documents: A digital environment for analysis and navigation." *Microcomputers for Information Management*, 8(2), pp. 119-133, 1991.

[8] L. Teodosio and W. Bender, "Salient video stills: content and context preserved," in *Proceeding ACM Multimedia 93*, Anaheim, CA, 1993, pp. 39-46.

[9] Y. Tonomura, and S. Abe, "Content oriented visual interface using video icons for visual database systems," in *Journal of Visual Languages and Computing*, vol. 1, 1990, pp183-198.

[10] B.L. Yeo and B. Liu, "A unified approach to temporal segmentation of Motion JPEG and MPEG compressed video," in *International Conference on Multimedia Computing and Systems*, pp. 81-88, May 1995.

[11] M. Yeung, "Analysis, modelling and representation of digital video," Ph.D. thesis, Princeton Univ., 1996.

[12] H. Zhang, A. Kankanhalli, and S.W. Smoliar, "Automatic partitioning of full-motion video," *Multimedia Systems*, vol. 1, pp. 10-28, July 1993.