2012 2nd International Conference on Management and Artificial Intelligence
IPEDR Vol.35 (2012) © (2012) IACSIT Press, Singapore

Fast Search of Sequences of Multiple Moving Objects

Zaher Al Aghbari [1] and Ayoub Al-Hamadi [2]

[1] Department of Computer Science, University of Sharjah, UAE
[2] Institute for Electronics, Signal Processing and Communications (IESK), Germany

Abstract. In this paper, we propose a technique to captures the features of a moving object at sampled instances. These features of every sampled instance are mapped from the high-dimensional feature space into a point in low-dimensional distance space. Thus, each moving object, is represented by a sequence of points that is considered the time-varying feature trajectory of the object in the low-dimensional space. Each of these sequences is represented by a minimum bounding box (MBB) to reduce further reduce the storage requirements of an index structure. The proposed technique indexes a sequence by the MBBs of its objects using a spatial access method (SAM), such as an R-tree to speed up the search. The proposed technique does not result in any false dismissal, but might contain a few false alarms, which are remove by a two-step refinement process.

Keywords: features of moving objects, retrieval of feature trajectories, dimensionality reduction, indexing trajectories.

1. Introduction

Recently, millions of video sequences have been produced by YouTube and Google Video and others and are made accessible to users. As a result, new applications have emerged that either use or produce such sequences such as surveillance systems, distance learning, medical imaging, news-on-demand and public visual information systems. These applications need to process the content of sequences in real time; therefore, a fast and effective indexing and search algorithms are essential. Content-based retrieval of sequences is a difficult task and thus existing systems, such as YouTube and Google Video, depend on keyword based search [1]. However, some research works in video indexing and retrieval, such as [4][5][13], have concentrated on extracting visual features (color, motion, shape, etc.) from video shots to represent the visual contents of these shots. Moreover, these systems represent the sequences by the extracted low level features ignoring the semantics of sequences. To bridge this semantic gap between low level features and user's semantics, some researchers use motion information analysis [2] or feature trajectories [3].

An approach to represent shots by the time-varying directions of objects and the time-varying spatial relationships between objects is proposed by the systems in [7], [8] and [1]. However, these systems only address an exact match algorithm for matching user queries with target shots. In time series databases, the works in [6] and [9] have discussed an ad hoc solution to the problem of high-dimensionality and large number of time sequences. The work in [10], proposed a method to efficiently locate subsequences within a collection of time series sequences, such that the subsequences match a given query pattern within a specified tolerance. Another method for exact ranked subsequence matching is proposed in [11]. In [12], an efficient method for retrieving subsequences under Dynamic Time Warping (DTW) is presented. In [15], subsequences were clustered to find association rules and to facilitate their efficient search.

In this paper, we propose a technique that extracts the features of individual objects in a sequence (e.g. video) at sampled instances. To reduce dimensionality, the extracted features of an object at each sampled instance are mapped from high-dimensional feature space into a point in low-dimensional distance space. Thus, every object will be represented by a point in low dimensional space at every instance it exist in. In a

sequence, say a video, an object is represented by a chain of points in low-dimensional space that is considered the time-varying feature trajectory of the object. Each trajectory is represented by a minimum bounding box (MBB). Then, the proposed indexes a sequence by the MBBs of its objects. These MBBs are inserted into a spatial access method (SAM), such as an R-tree, instead of the individual points; thus, greatly reducing storage requirements of the index and thus speeding up the search time.

The proposed technique supports a query in high-dimensional feature space of the form "Find objects similar to the query object Q" that is converted into a query in low-dimensional distance space of the form "Find points that are in the proximity of the query point q". That is a query becomes a range query or a K-nearest-neighbor query in the low-dimensional space.

The rest of this paper is organized as follows: In Section 2, we discuss how to represent and index a sequence by an object trajectory computation. In Section 3, we present a survey of related work. Searching for sequences is explained in Section 4. Section 5 presents and analyzes the conducted experiments. Finally, we conclude the paper in Section 6.

2. Representing and Indexing Moving Object

In this paper, we use video shots collected from the Internet as our sequences that contain multiple moving objects. We use the video shot structure as that of [3], where a shot structure is represented by an acyclic graph that consists of four layers: a shot layer (upper layer), event layer, keyframe layer, and object layer (lower layer). Where each shot constitutes one or more events, $E_1, E_2, ..., E_{Ne}$, and nn event, E_i, is a subsequence of consecutive frames that express a particular activity and contain a fixed number of semantically meaningful objects. Due to the close similarity between frames of an event E_i, only two keyframes (first and last frames of an event) are extracted to represent the event. Then, from each keyframe, a set of semantically meaningful objects ($O_1, O_2, ..., O_{No}$) are extracted. An object O_i is a semantically meaningful physical entity within a frame. Each object is represented by an RGB color histogram (maximum 10 colors).

A feature trajectory ζ_i of O_i as a sequence of its color histograms that are extracted at different instances (e.g. keyframes) in which O_i exists. Since each O_i is represented by one ζ_i, then each sequence S is represented by a number of object trajectories equal to number of objects in the sequence N_O.

$$S : \zeta_1, \zeta_2, \ldots, \zeta_{No} \quad (1)$$

The proposed technique uses a *topological mapping* [3] to map high-dimensional color histograms into points in low-dimensional distance space. The topological mapping method maps a color histogram into a 3-dimensional *distance* space and guarantees the correctness of results. Where, the 3 dimensions are the *red* (R_d), *green* (G_d) and *blue* (B_d) distance spaces (or topological spaces). Mapping the color histogram into the R_d G_d B_d distance space is achieved by, first, computing the distance between the color histogram C and a *virtual object* that is assumed to be at the origin of the topological axis (distance space) and then a point representing a color histogram is placed on the topological axis based on the computed distance. Indexing those mapped points in low-dimensional space using an R-tree will result in a much faster search as compared with sequential scanning method. To have a compact R-tree, we group the mapped points that belong to the same object trajectory by an *MBB*. Then, the *MBBs* are stored in the R-tree instead of individual points.

3. Searching for Sequences

To search for sequences such as videos, it is not practical for users to formulate exact match queries due to the difficulty to remember the exact features of objects. An obvious solution to this problem is similarity retrieval. To formulate a query that describes the trajectories of objects in a wanted sequence(s), we developed the user interface shown in Figure 1. With this interface, a user can specify the colors of objects at different time instances (or, keyframes) at which the these objects exist. First, a user selects an object from the 'Current Object' menu and a keyframe from the 'Current Frame' menu, then specifies the colors and their percentages of the selected object using the color palette and the text area, respectively. Then, a user clicks on the 'Include Specified Colors' button to accept the currently specified colors for current object at the

current frame. The 'Include Previous Colors' button allow a user to include the object`s colors specified at a previous frame into the current frame. That is, if the colors of an object are the same as in the previous frame, a user can copy them into the current frame without the need to re-specify them.

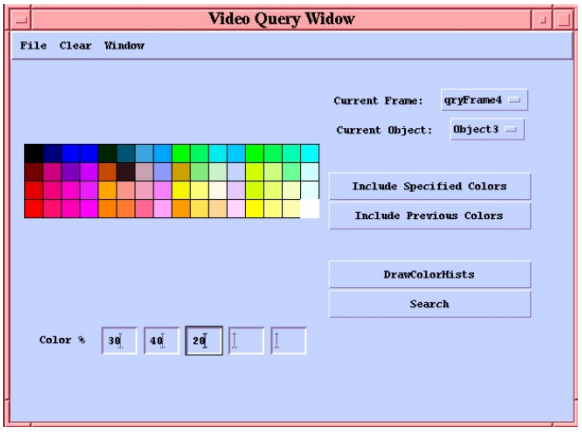

Fig. 1: A user interface for formulating queries investigating trajectories of objects in the wanted sequences.

The number of mapped points N_{qk} in the query object trajectory ζ^q_i is normally less than, or equal to, the number of mapped points N_k in the target object trajectory ζ_i. Users are not expected to have prior knowledge about N_k. Therefore exact match is not possible in this case, which requires $N_{qk} = N_k$, and thus the proposed supports similarity retrieval. That is, the distance between ζ^q_i and ζ_i should be based on how many color histograms in ζ^q_i have a match, within tolerance $\in$, with color histograms in ζ_i,.

When matching Q with the database of sequences, such as videos, each ζ^q_{MBBi} in Q is matched with *MBBs* in the R-tree separately; as a result, a cluster (a small subset of the MBBs in the R-tree) that is most similar to ζ^q_{MBBi} is retrieved. Hence, N_{qo} clusters are retrieved. These clusters are combined and a list is generated, where each item in the list contains a sequence identifier (SID) and a number of retrieved *MBBs* (N_b) that belong to this sequence. Where, $1 \leq N_b \leq N_{qo}$. The retrieved list of candidate sequences contains all the qualifying sequences [14], in addition to some false alarms. To remove these false alarms, we propose a two-step filtering process.

Filtering: First Step In this step, we make use of the number of retrieved *MBBs*, N_b, that belongs to each shot in the retrieved cluster to compute a *lower-bound* distance $D_{lb}(Q, S)$ between the query Q and a sequence S. We define $D_{lb}(Q, S)$ as follows:

$$D_{lb}(Q,S) = \frac{N_{qo} - N_b}{N_{qo}} \quad (2)$$

The $D_{lb}(Q, S)$ is a rough distance that underestimates the actual distance $D(Q, S)$ between Q and S. Based on the computed $D_{lb}(Q, S)$, sequences that have distances greater than, or equal to, σ from Q are removed because they are not considered similar. Where, σ is a variable that can be set by a user depending on the application and the desired strictness of similarity required by a search. In this paper, we set σ to 0.5.

Filtering: Second Step As a second step of filtering, we propose a *cTraj_Match* algorithm [14], which matches a query with the list of candidate sequences. Each sequence S_x in the list of candidate sequences is matched with the query Q by first Computing the distance $D(C^q_k, C_l)$ between each color histogram C^q_k in the queried object trajectory ζ^q_i and every color histogram C_l in the target object trajectory ζ_j. The algorithm only supports one-to-one correspondence, that is no one object trajectory in Q is corresponded to more than one object trajectory in S_x. The sum of all the distances between the corresponded object trajectories is considered the distance between the Q and S_x. Finally, all the sequences are sorted in an ascending order based on the computed distances. The top K shots in the sorted list, which are considered the most similar to Q, are returned to the user as a result of a search.

4. Experimental Results

The experiments shown below are performed on a collection of 490 video sequences categorized into sports (bike racing, car racing, skiing, soccer and football), movie, and animation. We have implemented the

system using Visual C++ and the query interface is developed using Java. We performed several experiments to evaluate the effectiveness and efficiency of the proposed method.

In information retrieval systems, the effectiveness of retrieval is normally measured by two metrics: *recall* and *precision*. Specifically, for our sequence collection, *recall measures the capacity of the proposed to retrieve all relevant sequences from the collection.*

$$recall = \frac{\rho}{\gamma} \tag{3}$$

On the other hand, *precision measures the retrieval accuracy; that is the ability of the proposed to reject false alarms.*

$$precision = \frac{\rho}{\delta} \tag{4}$$

To compute the recall and precision of the proposed technique, we issued 20 sample queries to search for 20 randomly selected sequences. These sample queries vary in the number of object trajectories, number of queryFrames in each color trajectory, number of colors used to describe each object's color histograms. Then, we compared the returned list of shots of each sample query with its ground truth. By varying the number of returned sequences from 1 to 10, a curve representing the average recall and precision is generated as shown in Figure 2. From Figure 2, we notice that precision is quite high when the number of returned sequences is small and then starts to slope down as the number of returned sequences increases. On the other hand, recall starts low and smoothly increases as the number of returned sequences increases. That is due to the fact that during the process of trying to retrieve all relevant sequences to a sample query some irrelevant sequences (false alarms) are also retrieved, reducing the precision.

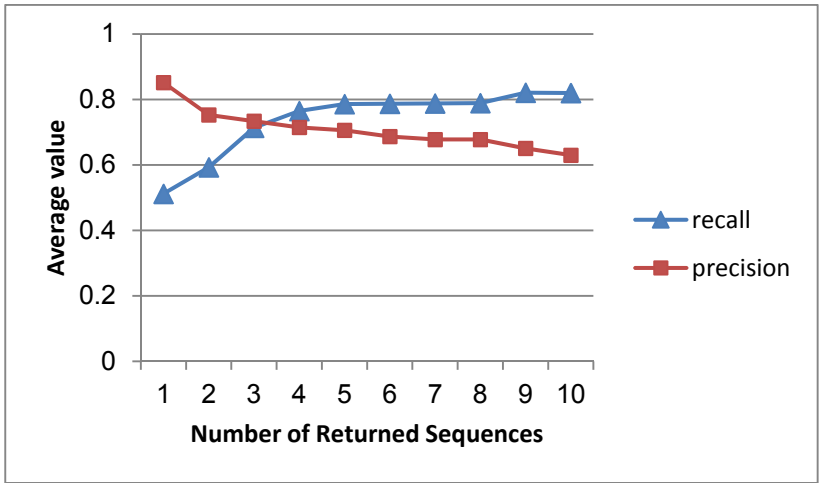

Fig. 2: The precision and recall curves averaged over 20 sample queries.

Table 1: Comparison between the naive-I and the proposed techniques in terms of storage requirements of the R-tree.

Description	***naïve***	***Proposed technique***
Number of inserted *MBBs*	4747	1545
Number of nodes	238	53
Size of R-tree in bytes	216938	63525

The proposed optimized storage requirements of the R-tree and that is the main reason behind the speed up of the search. Table 1 presents a summary of the optimization of storage requirements using the proposed technique as compared to the *naive* method, which indexes the instantaneous features of objects in the R-tree rather than their object trajectories.. Notice that by using the proposed technique, the 490 shots (the database used for experimentation) contain 1545 object trajectories, and by using the *naive* method, 4747 objects are extracted from those sequences. Thus, the size of the R-tree, which is used by *the proposed* is 63525 bytes, while the size of the R-tree used by the *naive* method is 216938 bytes. That is, the proposed reduces the size of the R-tree, as compared to the *naive* method, by more than 70%.

5. Conclusion

We proposed an effective and efficient technique for indexing and retrieving sequences by specifying the time-varying features of multiple moving objects that exist in the wanted sequence(s). In addition to supporting queries that investigate the time-varying color contents of sequences, the proposed optimizes the storage requirements of the R-tree and speeds up the response time. In the proposed technique, we used a mapping mechanism, called topological mapping, to map the object trajectories into points in a low-dimensional distance space and at the same time guarantee the correctness of result. The performed experiments show a substantial optimization in storage requirements of the R-tree and great speed up of response time when using the proposed technique as compared with the sequential scanning method or the *naive* method.

6. References

[1] M. Broilo, N. Piotto, G. Boato, N. Conci, F. De Natale. "Object Trajectory Analysis in Video Indexing and Retrieval Applications", Video Search and Mining, Studies in Computational Intelligence, Springer-Verlag, vol. 287, pp. 3-32, 2010.

[2] B.T. Morris, M.M. Trivedi. "A Survey of Vision-Based Trajectory Learning and Analysis for Surveillance", IEEE Trans. on Circuits and Systems for Video Tech. vol. 18, no. 8, pp. 1114-1127, 2008.

[3] Z. Aghbari, K. Kaneko, A. Makinochi. "Content-Trajectory Approach for Searching Video Databases", IEEE Trans. on Multimedia, vol. 5, no. 4, pp. 516-531, 2003.

[4] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D, Lee, D. Perkovic, D. Steele, P. Yanker. "Query by Image and Video Content: The QBIC System", IEEE, Sept. 1995.

[5] M.La.Cascia, E.Ardizzone. "JACOB: Just A Content-Based Query System for Video Databases", Proc. ICASSP, Atlanta, 1996.

[6] C.Faloutsos. "Searching Multimedia Databases By Content". Kluwer Academic Publishers, Boston, USA, 1996.

[7] J.Z.Li,M.T.Ozsu,D.Szafron. "Modeling of Moving Objects in a Video Database" IEEE int'l Conf. on Multimedia Computing and Systems. pp.336-343, June 1997.

[8] M.Nabil, A.H.Ngu, J.Shepherd. "Modelling Moving Objects in Multimedia Databases." Fifth Int'l Conference on Database Systems for Advanced Applications. Melbourne, Australia, April, 1997.

[9] C.Faloutsos and K.Lin. "A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets". ACM SIGMOD, May 1995.

[10] C.Faloutsos, M.Ranganathan and Y.Manolopoulos. "Fast Subsequence Matching in Time-Series Databses". ACM SIGMOD, May 1994.

[11] W.S. Han, J. Lee, Y.-S. Moon, and H. Jiang. "Ranked Subsequence Matching in Time-Series Databases". International Conference on Very Large Data Bases (VLDB), pp. 423–434, 2007.

[12] E. Keogh. "Exact Indexing of Dynamic Time Warping". International Conference on Very Large Data Bases, pp. 406–417, 2002.

[13] J.R.Smith, S.F.Chang. "VisualSEEK: A Fully Automated Content-Based Image Query System." ACM Multimedia Conference, Boston, pp.87-98, Nov. 1996.

[14] Z. Al Aghbari. "cTraj: Efficient Indexing and Searching of Sequences Containing Multiple Moving Objects", Springer International Journal on Intelligent Information Systems, DOI 10.1007/s10844-011-0180-5.

[15] R. Al-Mulla, Z. Al Aghbari. "Incremental Algorithm for Discovering Frequent Subsequences in Multiple Data Streams", IGI International Journal of Data Warehousing and Mining, Vol. 7, no. 4, pp. 1-20, 2011.