Effective Similarity Search Methods for Large Video Data Streams*

Seok-Lyong Lee[1], Seok-Ju Chun[2], and Ju-Hong Lee[3]

[1]School of Industrial and Information Engineering, Hankuk University of FS, Korea
sllee@hufs.ac.kr
[2]Dept. of Information and Comm. Engineering, KAIST, Korea
chunsj@islab.kaist.ac.kr
[3]School of Computer Science and Engineering, Inha University, Incheon, Korea
juhong@inha.ac.kr

Abstract. In this paper, we investigate the similarity search methods for large video data sets that are the collection of video clips. A video clip, a sequence of video frames describing a particular event, is represented by a sequence in a multidimensional data space. Each video clip is partitioned into *video segments* considering temporal relationship among frames, and then similar segments of the clip are grouped into *video clusters*. Based on these video segments and clusters, we define similarity functions and present two similarity search methods: the *HR* (hyper-rectangle)-*search* and the *RP* (representative point)-*search*. Experiments on synthetic sequences as well as real video clips show the effectiveness of our proposed methods.

1 Introduction

A video database may contain a number of video clips that can be represented by multidimensional data sequences (MDS's). In our earlier work [7], we have formally defined an MDS S with K points in the n-dimensional space as a sequence of its component vectors, $S = \langle S[1], S[2], \ldots, S[K] \rangle$, where each vector $S[j]$ $(1 \le j \le K)$ is composed of n scalar entries, that is, $S[j] = (S^1[j], S^2[j], \ldots, S^n[j])$. A video clip consists of multiple frames in the temporal order, each of which can be represented by a multidimensional vector in a feature space such as RGB or YCbCr color space. Thus, a video clip is modeled as a sequence of points in a multidimensional space such that each frame of the sequence constitutes a multidimensional point, whose components are feature values of the frame. A sequence is partitioned into video segments (or video shots) and then similar segments are grouped into a video cluster.

A video segment or cluster is represented by a hyper-rectangle (HR) that bounds all points in the segment or cluster and is an extension of a minimum bounding rectangle (MBR). The similarity search problem in this paper is described as follows:

Given: A set of video clusters, a query video, and a threshold ε

Goal: To find the sets of video segments that are similar to a query within ε

* This work was supported by Hankuk University of Foreign Studies Research Fund of 2003

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2660, pp. 1030–1039, 2003.

A query is given as a video clip which may consist of a single video frame, a single video segment with multiple frames, or multiple video segments. In the first two cases, a single answer set that contains relevant video segments is obtained, while multiple answer sets are obtained for the third case. It is natural to provide one answer set for each query video segment since video segments may be quite different each other even though they belong to a single video clip.

Similarity search methods for one-dimensional sequences have been extensively studied in the domain of time-series analysis [1, 2, 6, 9, 11]. All these methods however address the similarity search for one-dimensional time-series data in which each element of the time series is a real number. Thus they do not handle an MDS. In this paper, we have extended the traditional similarity search method for one-dimensional time series data to support multidimensional data sequences.

On the other hand, various methods [3, 4, 5, 12] have been proposed for the video search, most of which are a key-frame-based method. These methods extract a set of feature attributes like color, texture, shape, or layout from key frames. The retrieval is performed by matching the feature attributes of the query with those of the key frames in a database. However, they suffer from two problems. They miss relevant video segments since the retrieval mechanisms are based on the selected frames, leading not to preserve the correctness, and they are not very effective in representing video segments since their key frames are usually selected to be start and/or end frames. In reality, if a video segment is a part of cartoons or animation films, the contents of frames change drastically within the segment and thus start/end frames do not represent the segment well. To represent the contents of a video segment more exactly, various sophisticated methods have been studied, such as PanoramaExcerpts [10] which presents the entire visible contents of a segment extended with the camera pan or tilt by automatically detecting camera movements. It however requires considerable overhead.

In this paper, we present two methods. First, the *HR-search* is to find relevant segments in which the retrieval is processed on the basis of hyper-rectangles and the correctness is preserved. Any qualified segment with respect to a given query is not missed. Preserving the correctness is an important motivation of the *HR*-search. We believe that guaranteeing the correctness is one of the important features in the similarity search, especially for application domains that are correctness-sensitive, such as an application detecting criminal activities from stored CCTV films.

Our next method is based on the representative point (RP) of a video segment or cluster, called *RP-search*. The representative point is not chosen to be a start or an end point. We present a simple and effective method to select the representative point, considering geometric characteristics of a hyper-rectangle. The *RP*-search however does not guarantee the correctness. This method is designed to achieve a high precision rate while sacrificing the correctness slightly. It can be applied to the applications in which the precision is more critical than the recall.

2 Preliminaries

In this section, we provide basic definitions and mention the segmentation and clustering technique briefly to describe our proposed methods. A hyper-rectangle *HR*

with k points, P_j ($1 \le j \le k$), in the n-dimensional space, is represented by two endpoints, L(low point) and H(high point), of its major diagonal, and the number of points in the rectangle as follows: $HR = \langle L, H, k \rangle$, where $L = \{(L^1, L^2, \ldots, L^n) \mid L^i = min_{1 \le j \le k} (P_j^i)\}$, and $H = \{(H^1, H^2, \ldots, H^n) \mid H^i = max_{1 \le j \le k} (P_j^i)\}$ for $i = 1, 2, \ldots, n$. Using a hyper-rectangle we define a video segment and a video cluster formally as follows:

Definition 1. A video segment VS that contains k points in the temporal order, P_j for $j = 1, 2, \ldots, k$, is defined as follows: $VS = \langle sid, offset, HR \rangle$, where sid is a segment-id, $offset$ is an offset of VS from a start point of a sequence, and $HR = \langle L, H, k \rangle$ such that $L = \{(L^1, L^2, \ldots, L^n) \mid L^i = min_{1 \le j \le k} (P_j^i)\}$ and $H = \{(H^1, H^2, \ldots, H^n) \mid H^i = max_{1 \le j \le k} (P_j^i)\}$ for $i = 1, 2, \ldots, n$.

Definition 2. A video cluster VC with r video segments, VS_j for $j = 1, 2, \ldots, r$, is defined as follows: $VC = \langle cid, slist, HR \rangle$, where cid is a cluster-id, $slist$ is a list of sid's, and $HR = \langle L, H, k \rangle$ such that $L = \{(L^1, L^2, \ldots, L^n) \mid L^i = min_{1 \le j \le r} (VS_j.HR.L^i)\}$, $H = \{(H^1, H^2, \ldots, H^n) \mid H^i = max_{1 \le j \le r} (VS_j.HR.H^i)\}$ for $i = 1, 2, \ldots, n$, and $k = \Sigma_{1 \le j \le r} (VS_j.HR.k)$.

The segmentation is the repeating process of merging a point of a sequence into a segment if predefined conditions are satisfied. Consider a point of a sequence to be merged to a segment in the n-dimensional space. Then, the segmentation is done in such a way that if the merging of the point into the segment satisfies the conditions then it is merged into the current segment, otherwise a new segment is started from the point. We have defined two categories of conditions: the geometric bounding condition that considers the volume and edge of a segment and the semantic bounding condition that considers the distance between consecutive points of the sequence. Then similar segments in a sequence are grouped into a cluster considering those conditions. Due to the space limitation, we do not describe the segmentation and clustering algorithm in detail. Interested readers are referred to [8] for the details such as the conditions, algorithms and experimental results. Next, we introduce the metric to measure the distance between video segments. It is defined for same-length segments and then extended to different-length segments.

Definition 3. The distance $D_{VS}(VS_1, VS_2)$ between two video segments VS_1 and VS_2 of equal lengths, each of which has k points, is defined as the mean distance of the two, where the mean distance is defined as follows:

$$D_{VS}(VS_1, VS_2) = D_{mean}(VS_1, VS_2) = \frac{1}{k} \cdot \sum_{i=0}^{k-1} d(VS_1[i], VS_2[i]) \quad (1)$$

where $d(*,*)$ is the Euclidean distance between two points. For different-length segments that cannot be compared directly, point by point, the shorter segment is compared to the other by sliding it from the beginning to the end. The shortest of all distances of compared pairs is adopted as the distance between two segments. The following definition describes it more formally.

Definition 4. Let us consider the distance $D_{VS}(VS_1, VS_2)$ between two video segments

VS_1 and VS_2 of different lengths, each of which has p and q points respectively. That is, $p = VS_1.HR.k$ and $q = VS_2.HR.k$. Without loss of generality we assume $p \le q$. Then $D_{VS}(VS_1, VS_2)$ is defined as the minimum of mean distances of compared pairs, where the minimum mean distance is defined as follows:

$$D_{VS}(VS_1, VS_2) = \min_{1 \le j \le q-p+1} D_{mean}(VS_1[1:p], VS_2[j:j+p-1]) \quad (2)$$

3 Similarity Search Methods

3.1 HR-Search Method

To measure the distance between hyper-rectangles of video segments or video clusters in the n-dimensional space, we define the distance D_{HR} as the following, depending on their relative placement.

Definition 5. The distance D_{HR} between two hyper-rectangles, HR_1 and HR_2, in the n-dimensional space, is defined as the minimum Euclidean distance between two hyper-rectangles. That is:

$$D_{HR}(HR_1, HR_2) = \sqrt{\sum_{i=1}^{n} x_i^2}, \text{ where } x_i = \begin{cases} |HR_1.H_i - HR_2.L_i| & \text{if } HR_1.H_i < HR_2.L_i \\ |HR_1.L_i - HR_2.H_i| & \text{if } HR_2.H_i < HR_1.L_i \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

Observation 6. The distance D_{HR} is shorter than the distance between any pair of points, one in a hyper-rectangle HR_1 and the other in a hyper-rectangle HR_2. That is:

$$D_{HR}(HR_1, HR_2) \le \min_{P_1 \in HR_1, P_2 \in HR_2} d(P_1, P_2)$$

where P_1 and P_2 are the points that are contained in HR_1 and HR_2, respectively.

Observation 7. When a hyper-rectangle HR_1 is contained in the other hyper-rectangle HR_2 ($HR_2 \supseteq HR_1$), for an arbitrary hyper-rectangle HR_3, $D_{HR}(HR_3, HR_2) \le D_{HR}(HR_3, HR_1)$.

Based on Observation 6 and 7, we derive the following two lemmas, showing the lower bounding relationships with respect to D_{HR} and D_{VS}.

Lemma 8 (*Lower Bounding Distance*: $D_{HR}(VS_q, VS_t) \le D_{VS}(VS_q, VS_t)$).
The distance $D_{HR}(VS_q, VS_t)$ between two hyper-rectangles of a query segment VS_q and a video segment VS_t in a database is the lower bound of the distance $D_{VS}(VS_q, VS_t)$ of the two segments.

$$D_{HR}(VS_q, VS_t) \le D_{VS}(VS_q, VS_t \quad (4)$$

Proof: Let VS_q, VS_t have k, l points, and P_q, P_t be the arbitrary points that are contained in VS_q, VS_t, respectively. Without loss of generality, we assume $k \le l$, since we can switch VS_q and VS_t if $k > l$. Then:

$$D_{VS}(VS_q, VS_t) = \min_{1 \le j \le l-k+1} D_{mean}(VS_q[1:k], VS_t[j:j+k-1]) \ge \min_{P_q \in VS_q.HR, P_t \in VS_t.HR} d(P_q, P_t)$$

By Observation 6, the following holds:

$$D_{HR}(VS_q, VS_t) \le \min_{P_q \in VS_q, P_t \in VS_t} d(P_q, P_t)$$

Therefore, we conclude: $D_{HR}(VS_q, VS_t) \le D_{VS}(VS_q, VS_t)$.

Lemma 9 (*Lower Bounding Distance*: $D_{HR}(VS_q, VC) \le min_{VSt \in VC} D_{HR}(VS_q, VS_t)$). Let a video cluster *VC* contain one or more video segments. Then, the distance $D_{HR}(VS_q, VC)$ between two hyper-rectangles of a query segment VS_q and a video cluster *VC* in a database is the lower bound of the distance $D_{HR}(VS_q, VS_t)$ between two hyper-rectangles of VS_q and any video segment VS_t in *VC* ($VS_t \in VC$). That is:

$$D_{HR}(VS_q, VC) \le min_{VSt \in VC} D_{HR}(VS_q, VS_t)) \quad (5)$$

Proof: By the definition of a video cluster in Definition 2, a hyper-rectangle of a video cluster tightly bounds all hyper-rectangles of video segments. Since $VS_t.HR \subseteq VC.HR$ for all $VS_t \in VC$, by Observation 7, the distance $D_{HR}(VS_q, VC)$ is equal to or shorter than the distance $D_{HR}(VS_q, VS_t)$ for all $VS_t \in VC$.

Using Lemma 8 and 9, we can use the distances $D_{HR}(VS_q, VS_t)$ and $D_{HR}(VS_q, VC)$ to prune irrelevant video segments from a database without *'false dismissals,'* since they provide the lower bound with respect to the distance $D_{VS}(VS_q, VS_t)$. Given a query segment VS_q, let us consider a query, "*Find a set of video segments* VS_t *such that* $D_{VS}(VS_q, VS_t) \le \varepsilon$". By Equation 4 and 5, the following holds:

$$D_{HR}(VS_q, VC) \le D_{HR}(VS_q, VS_t) \le D_{VS}(VS_q, VS_t) \le \varepsilon, \text{ for all } VS_t \in VC \quad (6)$$

We first prune irrelevant video clusters and collect candidate video clusters using Equation 5. Then, each video segment in candidate video clusters is evaluated with respect to Equation 4 in order to prune irrelevant video segments and collect candidate video segments. Those candidate video segments are finally evaluated in the refinement process using the distance $D_{VS}(VS_q, VS_t)$ to get the final answer set of video segments. Before a query is processed, we do some pre-processing to extract feature vectors from video frames of each video clip, and then to map the video clip to an MDS. Each MDS is segmented and clustered during the video segmentation and clustering process. The similarity search is performed to select similar video segments from a database. A brief description of the *HR*-search algorithm is as follows.

Algorithm HR-search

```
/* A query video is segmented to i(≥1) video segments(VS_i).*/
/* For VS_i, a single answer set VS_i_ANS is reported.          */
Input: a set VC of video clusters, a query video clip Q,
       and a threshold ε
Output: answer sets {VS_i_ANS} of video segments
Step 0: /* Initialization */
        VS_i_ANS ← φ,     /* an answer set for VS_i          */
        VS_Q ← φ          /* a set of query video segments */
Step 1: /* Query segment generation from a query video  */
        Map a query clip to a multidimensional sequence
        Segment the sequence to one or more video segments
        VS_Q ← {query video segments generated}
Step 2: /* Filtering by D_HR */
        for each query video segment VS_i in VS_Q
```

```
        for each video cluster VC_j in the set VC
          if D_HR(VS_i, VC_j) ≤ ε then
            for each video segment VS_k in the cluster VC_j
              if D_HR(VS_i, VS_k) ≤ ε then
                VS_i_ANS ← VS_i_ANS ∪ {VS_k}
Step 3: /* Refinement by D_VS: sequential scan of video segments*/
        for each VS_m in the set VS_i_ANS
          if D_VS(VS_i, VS_m) > ε then
            VS_i_ANS ← VS_i_ANS − {VS_m}
Step 4: return set {VS_i_ANS}
```

3.2 RP-Search Method

We propose a simple and effective method to choose a representative frame from each video segment and video cluster using geometric characteristics of a hyper-rectangle. The following two definitions describe how to select the representative point (i.e. frame) for the video segment and the video cluster, respectively.

Definition 10. Consider a video segment VS with k points P_j ($1 \le j \le k$). A representative point RP_{VS} of VS is defined as follows: $RP_{VS} = (Pt \mid Pt \in \wp, d(Pt, P_c) = min_{1 \le j \le k}\, d(P_j, P_c))$, where $\wp$ is a set of points in VS and P_c is the center of $VS.HR$ given by $P_c = \left(\frac{VS.HR.L^1 + VS.HR.H^1}{2}, ..., \frac{VS.HR.L^n + VS.HR.H^n}{2} \right)$.

Definition 11. Consider a video cluster VC with r video segments, VS_j for ($1 \le j \le r$). A representative point RP_{VC} of VC is defined as follows: $RP_{VC} = (Pt \mid Pt \in \wp, d(Pt, P_c) = min_{1 \le j \le r}\, d(RP_{VS,j}, P_c))$, where $\wp$ is a set of representative points of video segments in VC, $RP_{VS,j}$ is RP_{VS} of the jth video segment in VC, and P_c is the center of $VC.HR$ given by $P_c = \left(\frac{VC.HR.L^1 + VC.HR.H^1}{2}, ..., \frac{VC.HR.L^n + VC.HR.H^n}{2} \right)$.

The nearest point from the center of a hyper-rectangle may represent well the characteristics of all points in the video segment or cluster even though it is not always the case. The basic idea is that if this point is used for the similarity search processing instead of the start or end frame that is selected as a key frame, the search effectiveness would be improved since it summarizes the geometric characteristics of a hyper-rectangle better. Our experiment shows that the similarity search method based on this representative point performs better than the traditional key-frame based search. By applying a representative point to Definition 1 and 2, we can extend the definitions of a video segment and a video cluster to $VS = \langle sid, offset, HR, RP_{VS} \rangle$ and $VC = \langle cid, slist, HR, RP_{VC} \rangle$, respectively.

The distance between representative points of video segments or clusters, D_{RP}, used in the *RP*-search method, is obtained simply, since D_{RP} is basically the Euclidean distance between two points in the multidimensional data space. When V_x and V_y are arbitrary video segments or video clusters in the space, D_{RP} is defined as follows:

$$D_{RP}(V_x, V_y) = d(V_x.RP, V_y.RP) \quad (7)$$

The *RP*-search is based on the representative points of video segments and clusters. Different from the *HR*-search, only a representative point of each video segment is indexed for similarity search processing. We do not describe this method in detail since the retrieval procedure is almost the same as traditional methods such as [3, 5] that adopt the key frame based search. The only difference is that in our method the search is based on the representative point which is the nearest point from the center of a hyper-rectangle, while traditional methods use a start point and/or an end point of a video segment. A query clip is also parsed and segmented to one or more video segments. A representative point is selected for each query segment on which the query processing is based.

4 Experimental Evaluation

In order to measure the effectiveness of the proposed methods, we have conducted experiments on synthetic data sets as well as real videos, since real-life data sets are generally restricted in size and in setting various parameters while synthetically generated data provide better flexibility. Our methods are mainly designed for and evaluated on a video, but they are not restricted to be used on the video only. Other application domains whose data formats can be represented by sequences such as the time-series and signal processing can also benefit. That is one of the reasons we have also evaluated our methods on synthetic data sets. The evaluation system is implemented in Microsoft VC++ under Windows Server environment.

All experimental data sets are for convenience 3-dimensional, but our methods do not restrict the dimensionality. We use two data sets, synthetic data sequences that are obtained using a Fractal function and real video data that are captured from a collection of TV news, dramas, and documentary films, as depicted in Table 1.

Table 1. Test data used

	Synthetic data	Video data
# of segments	12,700	5,632
# of queries	50	30
Threshold (ε)	0.01-0.30	0.01-0.30

The number of query sequences is 50 for synthetic data and 30 for real video data, respectively. Multiple query results are collected and averaged to produce final results. The threshold range was chosen from 0.01 to 0.30. We believe this range provides the proper selectivity, since our search space is normalized to have the range [0, 1] for each dimension and the threshold 0.30 corresponds to the search sphere with a diameter 0.60. The evaluation is mainly made with respect to the precision and the recall that are well known in the similarity search applications. Let a set of retrieved segments by a query be *Set*(*ret*), and a set of relevant segments be *Set*(*rel*). When the number of elements in the set *S* is denoted by |*S*|, the precision and the recall are defined as follows:

$$precision = \frac{|Set(ret) \cap Set(rel)|}{|Set(ret)|}, \quad recall = \frac{|Set(ret) \cap Set(rel)|}{|Set(rel)|} \tag{8}$$

In order to measure the effectiveness of our proposed methods, it is necessary to set the ground truth that provides the basis to evaluate the retrieval quality. In general, a video contains a huge amount of information that consists of a great number of frames. It is not easy for the human to decide how similar a pair of video segments is. In our experiment, we set the ground truth as the set of retrieved segments by the sequential scanning method using a measure D_{VS}.

Precision: In Fig. 1, we present the comparison of the precision among the *SP*-search that selects a start frame as a representative, the *SE*-search that selects both a start and an end frame, the *RP*-search, and the *HR*-search for synthetic sequences and real videos. As we can see in the graphs, the *RP*-search shows the highest precision, 0.60-0.84 for synthetic sequences and 0.53-0.86 for real videos. It is superior to other methods along the whole range of threshold values. From the fact, we observe that the representative point by our method provides a good representation for a segment or a cluster. The *SP*-search is the next.

On the other hand, the *SE*- and the *HR*-search show relatively low precision. In these methods, two points are involved for the retrieval, start/end points for the *SE*-search and low/high points for the *HR*-search, respectively. It is general that more segments are retrieved when two points are used for the search, leading more irrelevant segments to be retrieved. That is the reason why those two methods show low precision. The precision of the *HR*-search is in the range of 0.17-0.70 for synthetic sequences and 0.26-0.81 for real videos. For thresholds higher than 0.05, it is above 0.38 for synthetic sequences and above 0.48 for real videos. We believe that this precision is quite usable in the practical environment. In addition, real videos show higher precision than synthetic sequences. It is because real-life data is usually well-segmented at the shot boundary, showing more agglomerative characteristics than synthetic data.

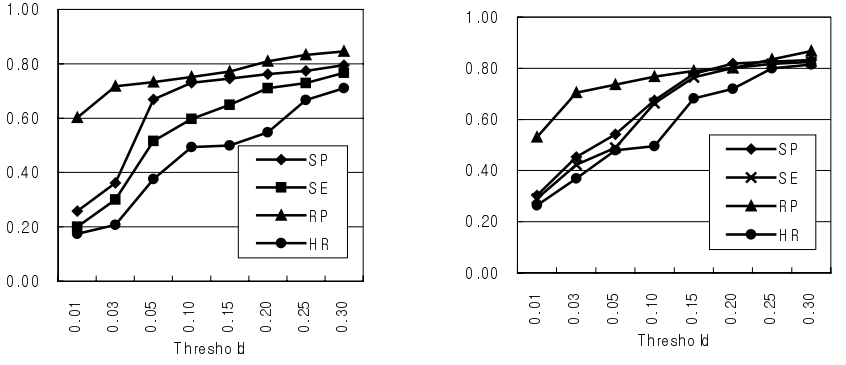

Fig. 1. Precision for synthetic sequences and real videos

Recall: Fig. 2 shows the recall of each method for synthetic sequences and real videos, respectively. In both graphs, the *HR*-search shows the recall of 1.0. It means that this method does not allow any false dismissal. The *SE*-search is the next, showing 0.16-0.77 for synthetic sequences and 0.55-0.85 for real videos. Different from the case of precision, the methods in which two points are involved for the search, such as the *SE*- and the *HR*-search, show higher recall than other methods that use a single point for the search. It is known that there is a trade-off between the precision and the recall. When the precision becomes high, the recall becomes low and vice versa. The *RP*-search shows relatively good recall. The range is 0.28-0.73 for synthetic sequences and 0.50-0.82 for real videos. In the case of real videos, the recall of the *RP*-search is very close to the *SE*-search, sometimes becoming superior for some thresholds. But the *SP*-search shows poor recall for both synthetic and real videos, preventing it to be used for recall-sensitive applications.

From experiments on the precision and recall, we conclude that the *RP*-search is the most preferable with respect to the precision while the *HR*-search is the best with respect to the recall.

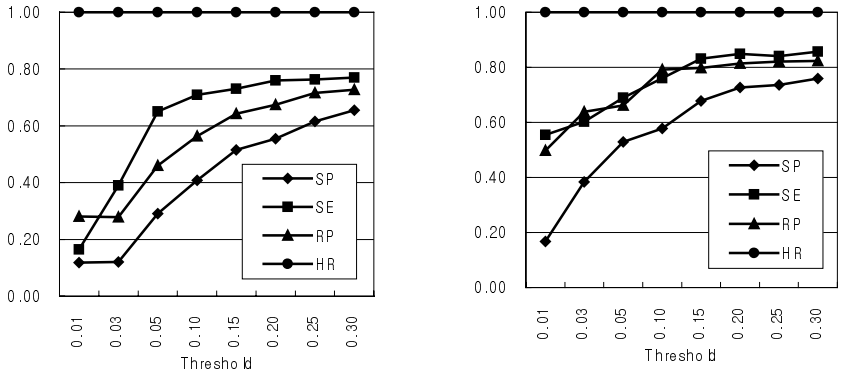

Fig. 2. Recall for synthetic sequences and real videos

5 Conclusions

The similarity retrieval of sequence data sets is one of the great potential areas in database applications since it can be extensively applied to various application domains, such as time-series data, digital and analog signals. In this paper, we have extended the traditional similarity search method for one-dimensional time series data to support multidimensional data sequences, and presented two similarity search methods. One is a correctness-preserving method called '*HR*-search' in which the query is processed based on the hyper rectangles of video segments and clusters, and the other is a method called the '*RP*-search' based on the representative points of video segments and clusters. To evaluate the similarity, we have defined the distance measures, D_{HR} between two hyper-rectangles and D_{RP} between two representative points, of video segments or clusters. The *HR*-search guarantees '*no false dismissal*'

while maintaining reasonable level of the precision, and the *RP*-search provides high precision compared to other traditional methods while maintaining a fairly good recall rate. Thus this method can be used for applications in which the precision is more important than the recall.

The methods proposed in this paper can be widely used for various application domains in which the data format is represented by multidimensional sequences. One of potential applications which is emphasized in this paper is the similarity query on large video data streams, but we believe other application areas can also benefit. As the future work, we plan to study on applying the proposed methods to other application domains considering their own characteristics, such as the voice signal matching and the region-based image search.

References

1. R. Agrawal, C. Faloutsos, and A. Swami: Efficient Similarity Search in Sequence Databases. Proceedings of Foundations of Data Organizations and Algorithms (1993)
2. C. Faloutsos, M. Ranganathan, and Y. Manolopoulos: Fast Subsequence Matching in Time-Series Databases. Proc. of ACM SIGMOD (1994) 419–429
3. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker: Query by Image and Video Content: The QBIC System. IEEE Computer, Vol. 28, No. 9 (1995) 23–32
4. A. Hampapur, A. Gupta, B. Horowitz, C. F. Shu, C. Fuller, J. Bach, M. Gorkani, and R. Jain: Virage Video Engine, Proc. of SPIE: Storage and Retrieval for Image and Video Databases V. (1997) 188–197
5. A. K. Jain, A. Vailaya, and X. Wei, Query by Video Clip, Multimedia Systems Vol.7 (1999) 369–384
6. E.J. Keogh, K. Chakrabarti, S. Mehrotra, and M.J. Pazzani: Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. Proc. of ACM SIGMOD (2001) 151–162
7. S. L. Lee, S. J. Chun, D. H. Kim, J. H. Lee, and C. W. Chung: Similarity Search for Multidimensional Data Sequences. Proc. of IEEE Int'l Conference on Data Engineering (2000) 599–608.
8. S. L. Lee and C. W. Chung: Hyper-Rectangle Based Segmentation and Clustering of Large Video Data Sets. Information Science, Vol. 141, No. 1–2 (2002) 139–168
9. D. Rafiei: On Similarity Queries for Time Series Data. Proc. of Int'l Conference on Data Engineering (1999) 410–417
10. Y. Taniguchi, A. Akutsu, and Y. Tonomura, PanoramaExcerpts: Extracting and Packing Panoramas for Video Browsing, Proceedings of ACM Multimedia (1997) 427–436
11. B. K. Yi and C. Faloutsos: Fast Time Sequence Indexing for Arbitrary Lp Norms. Proc. of Int'l Conference on VLDB (2000) 385–394
12. H. J. Zhang, J. Wu, D. Zhong, and S. W. Smoliar, An Integrated System for Content-Based Video Retrieval and Browsing, Pattern Recognition Vol. 30 (1997) 643–653