Available online at www.sciencedirect.com

SCIENCE DIRECT®

J. Vis. Commun. Image R. 16 (2005) 749–773

JOURNAL OF VISUAL Communication & IMAGE Representation

www.elsevier.com/locate/jvci

Similarity retrieval of videos by using 3D C-string knowledge representation

Anthony J.T. Lee*, Han-Pang Chiu, Ping Yu

Department of Information Management, National Taiwan University, No. 1, Section 4, Roosevelt Road, Taipei 106, Taiwan, ROC

Received 9 March 2004; accepted 15 November 2004
Available online 7 January 2005

Abstract

We have proposed a new spatio-temporal knowledge structure called 3D C-string to represent symbolic videos accompanying with the string generation and video reconstruction algorithms. In this paper, we extend the idea behind the similarity retrieval of images in 2D C^+-string to 3D C-string. Our extended approach consists of two phases. First, we infer the spatial relation sequence and temporal relations for each pair of objects in a video. Second, we use the inferred relations to define various types of similarity measures and propose the similarity retrieval algorithm. By providing various types of similarity between videos, our proposed similarity retrieval algorithm has discrimination power about different criteria. Finally, some experiments are performed to show the efficiency of the proposed approach.
© 2004 Elsevier Inc. All rights reserved.

Keywords: Video databases; Spatio-temporal inference; Similarity retrieval; 3D C-string

1. Introduction

With the advances in information technology, videos have been promoted as a valuable information resource. Because of its expressive power, videos are an

* Corresponding author. Fax: +866 2 23621327.
E-mail address: jtlee@ntu.edu.tw (A.J.T. Lee).

1047-3203/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2004.11.006

appropriate medium to show dynamic and compound concepts that cannot be easily explained or demonstrated using text or other media. Recently, there has been widespread interest in various kinds of database management systems for managing information from videos. The video retrieval problem is concerned with retrieving videos that are relevant to the users' requests from a large collection of videos, referred to as a video database.

Over the last decade, many image/video indexing and retrieval methods have been proposed, for example, OVID [32], KMED [8], QBIC [11], VideoQ [3], etc. Doulamis et al. [10] presented a fuzzy histogram of visual content for video summarization and content-based retrieval. Lei and Lin [23] developed a geometry-based indexing approach to retrieve videos by using the geometric structure of objects and their spatial relationships. Naphade and Huang [30] proposed a probabilistic framework mapping low-level features to high-level semantics. VideoText [17] provided the conceptual graph representation to capture the semantic associations among the concepts described in text annotations of videos. Guimar et al. [12] used topological and morphological tools to transform each video event into a different pattern on a 2D image, called visual rhythm, to segment a video. VideoQA [37] allowed users to use short natural language questions with implicit constraints on contents to retrieve short precise news video summaries.

Moreover, many methods using the temporal properties to index and retrieve videos have been proposed. Caspi and Irani [1] proposed a sequence-to-sequence alignment approach to establish correspondences between two different video sequences of the same dynamic scene and to integrate information across multiple video sequences. Liu and Kender [26] presented a heuristic method called sort-merge feature selection to segment and categorize videos in a coarse-fine manner. Ngo et al. [31] proposed a motion computation method based on a structure tensor formulation to encode visual patterns of spatio-temporal slices in a tensor histogram and to describe the motion trajectories of moving objects. Lo et al. [27] presented a framework for retrieving video sequences using successive modular operations on temporal similarity. Snoek and Worring [36] developed the TIME framework to classify the semantic events in multimodal video documents.

To retrieve desired videos from a video database, one of the most important methods for discriminating videos is the perception of the objects and the spatio-temporal relations that exist between the objects in a video. Much progress in the area of content-based video retrieval has been made in recent years, such as object searching based on localized color [29], moving object extraction and indexing [13,22,24,28,35], motion trail matching in MPEG [9], key-frame-based video browsing [39], and abstraction of high-level video units [38].

To represent the spatial relations between the objects in a symbolic image, many iconic indexing approaches have been proposed, such as, 2D string [4], 2D G-string [16], 2D C-string [20], 2D C^+-string [15], unique-ID-based matrix [5], GPN matrix [6], virtual image [33], BP matrix [7], and 2D Z-string [18]. In image similarity retrieval, Chang et al. [4] defined three types of 2D subsequences to perform subpicture matching on 2D strings and transformed the image retrieval problem into a problem of 2D subsequence matching. Lee et al. [21] proposed a similarity retrieval algorithm based on 2D string longest common subsequence (LCS). They showed that an LCS problem of 2D string can be transformed to the maximum complete subgraph problem [21]. Based on 13 spatial relations and their categories, Lee and Hsu [21] proposed three types of similarity between images represented by 2D C-string. In the knowledge structure of 2D C^+-string [15], they used the same clique finding algorithm to find the most similar pictures.

To represent the spatial and temporal relations between the objects in a symbolic video, many iconic indexing approaches, extended from the notion of 2D string to represent the spatial and temporal relations between the objects in a video, have been proposed, for example, 2D B-string [34], 2D C-tree [14], 9DLT strings [2], 3D-list [25], and 3D C-string [19]. In the 3D C-string, we used the projections of objects to represent spatial and temporal relations between the objects in a video, and to keep track of the motions and size changes of the objects in a video. We also developed the string generation and video reconstruction algorithms for the 3D C-string. The string generated by the string generation algorithm is unique for a given video and the video reconstructed from a given 3D C-string is unique too. This approach can provide us an easy and efficient way to retrieve, visualize, and manipulate video objects in video database systems.

The capability of similarity retrieval is important in video database management systems. Therefore, in this paper, we extend the idea behind the similarity retrieval of images in 2D C^+-string to 3D C-string. Our extended approach consists of two phases. First, we infer the spatial relation sequence and temporal relations for each pair of objects in a video. Second, we use the inferred relations and sequences to define various types of similarity measures and propose the similarity retrieval algorithm. By providing various types of similarity between videos, our proposed similarity retrieval algorithm has discrimination power about different criteria. Our proposed approach can be easily applied to an intelligent video database management system to infer spatial and temporal relations between the objects in a video and to retrieve the videos similar to a given query video from a video database.

The remainder of this paper is organized as follows. We first review the 3D C-string approach of representing symbolic videos in Section 2. In Section 3, we describe the spatial and temporal relation inference algorithms from which all relation sequences between objects can be easily derived. Then we discuss the similarity retrieval algorithm based on 35 types of similarity between videos in Section 4. In Section 5, the results on two series of performance experiments are presented. Finally, concluding remarks are made in Section 6.

2. 3D C-string

In the knowledge structure of 3D C-string [19], we use the projections of objects to represent the spatial and temporal relations between the objects in a video. The objects in a video are projected onto the *x*-, *y*-, and time-axes to form three strings representing the relations and relative positions of the projections in the *x*-, *y*-, and time-axes, respectively. These three strings are called *u*-, *v*-, and *t*-strings.

The projections of an object onto the x-, y-, and time-axes are called x-, y-, and time-projections, respectively. In comparison with the 2D C^+-string, the 3D C-string has one more dimension: time-dimension. So the 3D C-string is different from the 2D C^+-string that has only spatial relations between objects, it has spatial and temporal relations. Hence, it is required to keep track of the information about the motions and size changes of the objects in a video in 3D C-string.

In the knowledge structure of 3D C-string, there are 13 relations for two one-dimensional intervals. For the x (or y) dimension, there are 13 spatial relations and its corresponding spatial operators have been presented in 2D C-string [20] as listed in Table 1, where Begin(P) and End(P) are the begin-bound (beginning point) and end-bound (ending point) of the x- (or y-) projection of object P, respectively. The notations used in this paper follow those defined in the 2D C-string. In the time dimension, there are 13 temporal relations, too. So we use the same temporal operators as the spatial operators. For example, in the x (or y) dimension, $P < Q$ represents that the projection of object P is before that of object Q. In the time dimension, $P < Q$ denotes that object P disappears before object Q appears. According to the research of Lee and Hsu [20], all the 13 operators except "/" can precisely (no ambiguity) represent the relations between two objects. To avoid using the operator "/", we replace P/Q with $P]Q|Q$ in our cutting mechanism and string generation algorithm.

In the knowledge structure of 3D C-string, an object is approximated by a minimum bounding rectangle (MBR) whose sides are parallel to the x-axis and y-axis. For each object, we keep track of its initial location and size. That is, we keep track of the location and size of an object in its starting frame. After keeping track of the initial location and size of an object, we record the information about its motions and size changes in the 3D C-string.

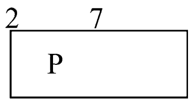
To represent the time intervals when the motion and size of an object are changed, we introduce one more temporal operator, $|_t$. For example, $P_3 \, |_t \, P_5$ denotes that in the first three frames, object P remains in the same state of the motion and size change. However, from the fourth frame to the eighth frame, the state of the motion and size change of object P is changed into another.

We also define some metric measures to the knowledge structure of 3D C-string, and the metric measures are shown as follows:

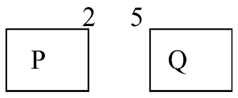
Table 1
The definitions of spatial operators in 2D C-string

Notations	Conditions
$P < Q$	$\text{End}(P) < \text{Begin}(Q)$
$P = Q$	$\text{Begin}(P) = \text{Begin}(Q)$, $\text{End}(P) = \text{End}(Q)$
$P \vert Q$	$\text{End}(P) = \text{Begin}(Q)$
$P\%Q$	$\text{Begin}(P) < \text{Begin}(Q)$, $\text{End}(P) > \text{End}(Q)$
$P[Q$	$\text{Begin}(P) = \text{Begin}(Q)$, $\text{End}(P) > \text{End}(Q)$
$P]Q$	$\text{Begin}(P) < \text{Begin}(Q)$, $\text{End}(P) = \text{End}(Q)$
P/Q	$\text{Begin}(P) < \text{Begin}(Q) < \text{End}(P) < \text{End}(Q)$

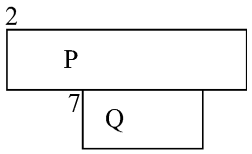
1. The size of an object: P_σ denotes that the size (length) of the x- (y- or time-) projection of object P is equal to σ, where $\sigma = \text{End}_x(P) - \text{Begin}_x(P)$ ($\sigma = \text{End}_y(P) - \text{Begin}_y(P)$, or $\sigma = \text{End}_{\text{time}}(P) - \text{Begin}_{\text{time}}(P)$), $\text{Begin}_x(P)$, and $\text{End}_x(P)$ are the x coordinates of the begin-bound and end-bound of the x-projection of object P, respectively. For example,



is represented by P_5.
2. The distance associated with operator <: $P <_\delta Q$ denotes that the distance between the x- (y- or time-) projection of object P and that of object Q is equal to δ, where $\delta = \text{Begin}_x(Q) - \text{End}_x(P)$ ($\delta = \text{Begin}_y(Q) - \text{End}_y(P)$, or $\delta = \text{Begin}_{\text{time}}(Q) - \text{End}_{\text{time}}(P)$). In the knowledge structure of 3D C-string, we only keep track of the initial location of an object. Hence, for $P <_\delta Q$, the starting frame of object P may be different from that of object Q. For example,



is represented by $P <_3 Q$.
3. The distance associated with operator %: $P\%_\delta Q$ denotes that the distance between the x- (y- or time-) projection of object P and that of object Q is equal to δ, where $\delta = \text{Begin}_x(Q) - \text{Begin}_x(P)$ ($\delta = \text{Begin}_y(Q) - \text{Begin}_y(P)$, or $\delta = \text{Begin}_{\text{time}}(Q) - \text{Begin}_{\text{time}}(P)$). For example,



is represented by $P\%_5 Q$.

4. The velocity and rate of size change associated with motion operators $\uparrow_{v,r}$ and $\downarrow_{v,r}$: operator $\uparrow_{v,r}$ denotes that the object moves along the positive direction of the x- (or y-) axis. Operator $\downarrow_{v,r}$ denotes that the object moves along the negative direction of the x- (or y-) axis. v is the velocity of the motion and r is the rate of size change of an object. For example, an u-string: $P \uparrow_{2,1}$ denotes that object P moves along the positive direction of the x-axis with the velocity = 2 and the rate of size change = 1. That is, the size of object P remains unchanged.

To see how 3D C-string works, let us consider the example as shown in Fig. 1. The 3D C-string of video V_a shown in Fig. 1 can be represented as follows:

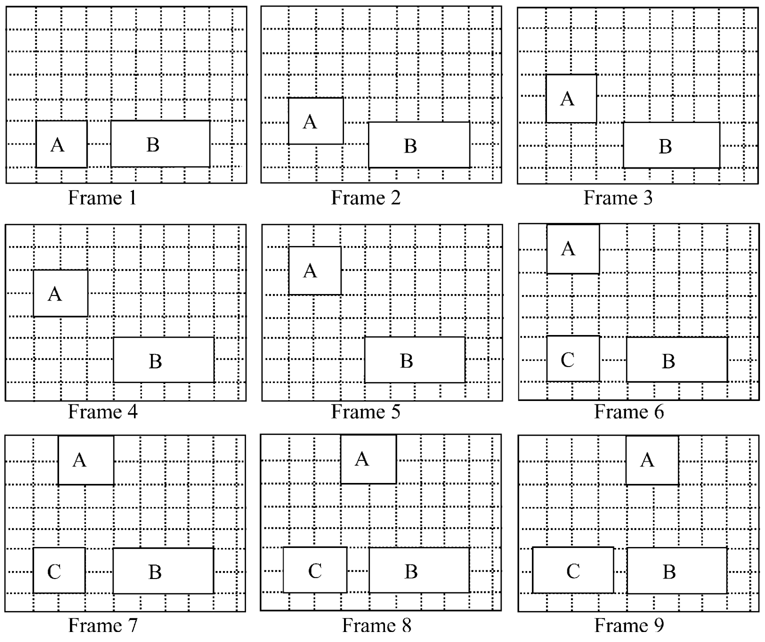


Fig. 1. Video V_a.

u-string: $((A_2 \uparrow_{0,1} \uparrow_{1,1} = C_2 \uparrow_{0,\,1} \uparrow_{0,1.225}) <_1 B_4)$
v-string: $(A_2 \uparrow_{1,\,1} \uparrow_{0,1} = B_2 = C_2 \uparrow_{0,\,1} \uparrow_{0,1})$
t-string: $((A_6|_t A_3 = B_9)]\ C_2|_t C_2)$

From frame 1 to frame 6, object A moves from bottom to top along the positive direction of the y-axis with the velocity of 1 unit/frame, but no motion along the x-axis. From frame 7 to frame 9, object A changes its motion and moves from left to right along the positive direction of the x-axis with the velocity of 1 unit/frame, but no motion along the y-axis. Object C is getting larger along the x-axis with the rate of size change of 1.225 units/frame, but no size change along the y-axis. Because objects A and C change their states, operator $|_t$ appears in the t-string of objects A and C. Therefore, the knowledge structure of 3D C-string provides an easy and efficient way to represent the spatio-temporal relations between the objects in a video.

3. Spatio-temporal relation inference

Like the method proposed by Huang and Jean [15], we can use the *spatial decision tree* as shown in Fig. 2 to determine the spatial relation between the x- (or y-) projection of object P and that of object Q. In Fig. 2, B_P denotes the begin-bound of the x- (or y-) projection of object P and S_P denotes the size of the x- (or y-) projection of object P. Similarly, we can construct the *temporal decision tree* to determine the temporal relation between the time-projection of object P and that of object Q.

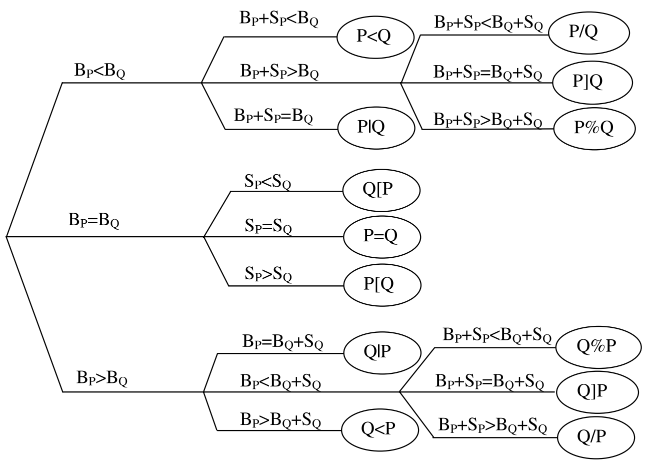


Fig. 2. Spatial decision tree.

The spatial relation between the x- (or y-) projection of object P and that of object Q may change over time in the video. These changing relations will form a spatial relation sequence. Therefore, we need to compute and record the new spatial relation whenever their spatial relation differs. Intuitively, we need to compute the spatial relation sequence between objects P and Q in their overlapped time interval. However, to precisely record the spatial relation changes, we also need to compute the spatial relation sequence between objects P and Q in the non-overlapped time interval. To do so, we compute the spatial relation sequence by comparing one object in every frame in the non-overlapped time interval with the other object at the nearest time point.

Let us consider that the temporal relation between objects P and Q is P/Q. First, we need to compute the spatial relation sequence between P and Q in their overlapped time interval, [$\text{Begin}_{\text{time}}(Q)$, $\text{End}_{\text{time}}(P)$]. Second, we need to compute the spatial relation sequence between P and Q in the interval [$\text{Begin}_{\text{time}}(P)$, $\text{Begin}_{\text{time}}(Q)$] by comparing Q at $\text{Begin}_{\text{time}}(Q)$ with P in every frame in [$\text{Begin}_{\text{time}}(P)$, $\text{Begin}_{\text{time}}(Q)$]. Third, we need to compute the spatial relation sequence between P and Q in the interval [$\text{End}_{\text{time}}(P)$, $\text{End}_{\text{time}}(Q)$] by comparing P at $\text{End}_{\text{time}}(P)$ with Q in every frame in [$\text{End}_{\text{time}}(P)$, $\text{End}_{\text{time}}(Q)$]. Then, merge those three spatial relation sequences together. The merged sequence is the spatial relation sequence between P and Q. This is the main idea of the *spatial relation inference algorithm*. The algorithm is described in detail in Fig. 3.

For example, let us show how to compute the spatial relation sequence of objects A and B as shown in Fig. 1. We apply the *spatial relation inference algorithm* to objects A and B. Their temporal relation is $A = B$. Let us consider the spatial relation changes in the x dimension. In frame 1, the spatial relation between objects A and B is $A < B$. In frame 9, the spatial relation between objects A and B is $B[A$. There exist some relation changes from frame 1 to frame 9 for objects A and B. We need to compute those relation changes and find that the spatial relations between objects A and B are $A < B$ in frames 1–6, $A|B$ in frame 7, A/B in frame 8, and $B[A$ in

```
Algorithm: spatial relation inference
Input: a video represented by the 3D C-string
Output: spatial relation sequences between each pair of objects in the video
1. Apply our 3D C-string video reconstruction algorithm [19] to the given u-string, v-string, and
   t-string. The algorithm returns the initial locations, sizes, and the motion lists of all objects in
   the u-string (or v-string). It also returns the starting frame numbers, duration, and change lists
   of all objects in the t-string.
2. For each pair of objects P and Q, perform steps 3~10.
3. Create an empty spatial relation sequence for objects P and Q.
4. Determine the temporal relation between objects P and Q by inputing their starting frame
   numbers and duration into the temporal decision tree. The temporal decision tree returns the
   temporal relation between them.
5. If the temporal relation between objects P and Q is equal to one of the following relations: P/Q,
   P]Q, P%Q, Q[P, P=Q, P[Q, Q%P, Q]P, or Q/P, record the starting and ending frame numbers
   of their overlapped time interval. Compute all the spatial relations between P and Q in their
   overlapped time interval, and append those spatial relations and their associated time intervals
   to the spatial relation sequence.
6. If the temporal relation between objects P and Q is equal to P|Q or P/Q,
   (a) compute the spatial relations between P and Q by comparing Q at $Begin_{time}(Q)$ with P in
       every frame in the interval [$Begin_{time}(P)$, $Begin_{time}(Q)$];
   (b) compute the spatial relations between P and Q by comparing P at $End_{time}(P)$ with Q in
       every frame in the interval [$End_{time}(P)$, $End_{time}(Q)$];
   (c) merge those spatial relations and associated intervals to their spatial relation sequence.
7. If the temporal relation between objects P and Q is equal to P%Q,
   (a) compute the spatial relations between P and Q by comparing Q at $Begin_{time}(Q)$ with P in
       every frame in the interval [$Begin_{time}(P)$, $Begin_{time}(Q)$];
   (b) compute the spatial relations between P and Q by comparing Q at $End_{time}(Q)$ with P in
       every frame in the interval [$End_{time}(Q)$, $End_{time}(P)$];
   (c) merge those spatial relations and associated intervals to their spatial relation sequence.
8. If the temporal relation between objects P and Q is equal to P]Q,
   (a) compute the spatial relations between P and Q by comparing Q at $Begin_{time}(Q)$ with P in
       every frame in the interval [$Begin_{time}(P)$, $Begin_{time}(Q)$];
   (b) merge those spatial relations and associated intervals to their spatial relation sequence.
9. If the temporal relation between objects P and Q is equal to P<Q,
   (a) compute the spatial relations between P and Q by comparing Q at $Begin_{time}(Q)$ with P in
       every frame in the interval [$Begin_{time}(P)$, $End_{time}(P)$];
   (b) compute the spatial relations between P and Q by comparing P at $End_{time}(P)$ with Q in
       every frame in the interval [$Begin_{time}(Q)$, $End_{time}(Q)$];
   (c) merge those spatial relations and associated intervals to their spatial relation sequence.
10. Similarly, if the temporal relation between objects P and Q is equal to Q|P, Q/P, Q%P, P[Q,
    Q]P, Q[P, or Q<P, their spatial relations can be computed likewise.
```

Fig. 3. Spatial relation inference algorithm.

frame 9. The spatial relation sequence of objects A and B can be represented as $A < B$:[1, 6], $A|B$:[7, 7], A/B:[8, 8], and $B[A$:[9, 9]. Similarly, we can find that the spatial relation sequence between objects A and C in the x dimension is $A = C$:[1, 6], C/A:[7, 8], and $C < A$:[9, 9].

Before discussing the temporal relation inference, we define a notation to represent a time interval set.

Definition 1. $[\theta_1 \sim \theta_2,\ \theta_3 \sim \theta_4]$ is a time interval set, which is defined as $\{[\tau_1, \tau_2] | \theta_1 \leqslant \tau_1 \leqslant \theta_2,\ \theta_3 \leqslant \tau_2 \leqslant \theta_4,\ \tau_1 \leqslant \tau_2\}$ where θ_1, θ_2, θ_3, and θ_4 are positive integers and $\theta_1 \leqslant \theta_2$, $\theta_3 \leqslant \theta_4$.

Next, let us consider how to compute the possible temporal relations between objects P and Q in every sub-interval of [min(Begin$_{\text{time}}(P)$, Begin$_{\text{time}}(Q)$), max (End$_{\text{time}}(P)$, End$_{\text{time}}(Q)$)], where function $\min(a,b)$ returns the smaller value of a and b, and function $\max(c,d)$ returns the larger value of c and d. To compute the possible temporal relations between P and Q, we need to find the time points at which the temporal relations between them can be changed. For example, if the temporal relation between P and Q is equal to $P]Q$, the temporal relations can be changed at Begin$_{\text{time}}(Q)$. So, we have the two possible temporal relations between them: (a) $P]Q$ holds for each interval in the interval set [Begin$_{\text{time}}(P) \sim$ Begin$_{\text{time}}(Q) - 1$, Begin$_{\text{time}}(Q) \sim$ End$_{\text{time}}(Q)$]; (b) $P = Q$ holds for each interval in the interval set [Begin$_{\text{time}}(Q) \sim$ End$_{\text{time}}(Q)$, Begin$_{\text{time}}(Q) \sim$ End$_{\text{time}}(Q)$]. The *temporal relation inference algorithm* is described in detail in Fig. 4.

For example, let us apply the *temporal relation inference algorithm* to objects A and C as shown in Fig. 1. Since the temporal relation between objects A and C is $A_9]C_4$, all the possible temporal relations between objects A and C are $A]C$:$[1 \sim 5,\ 6 \sim 9]$ and $A = C$:$[6 \sim 9,\ 6 \sim 9]$.

4. Similarity retrieval

In this section, we extend the idea behind the spatial similarity retrieval in 2D C$^+$-string to 3D C-string. We define 35 types of the similarity measures and propose the similarity retrieval algorithm.

Before defining the similarity measures, we introduce some notations used later.

Definition 2. $]\theta_1 \sim \theta_2,\ \theta_3 \sim \theta_4]$ is a time interval set and its begin-bound is extendible.$]\theta_1 \sim \theta_2,\ \theta_3 \sim \theta_4]$ is defined as $\{[\tau_1, \tau_2] | \min(1, \theta_1) \leqslant \tau_1 \leqslant \theta_2,\ \theta_3 \leqslant \tau_2 \leqslant \theta_4,\ \tau_1 \leqslant \tau_2\}$ where θ_1, θ_2, θ_3, θ_4 are positive integers and $\theta_1 \leqslant \theta_2$, $\theta_3 \leqslant \theta_4$. Similarly, $[\theta_1 \sim \theta_2,\ \theta_3 \sim \theta_4[$ is a time interval set and its end-bound is extendible. $[\theta_1 \sim \theta_2,\ \theta_3 \sim \theta_4[$ is defined as $\{[\tau_1, \tau_2] |\ \theta_1 \leqslant \tau_1 \leqslant \theta_2, \theta_3 \leqslant \tau_2 \leqslant \max(\theta_4, V_L),\ \tau_1 \leqslant \tau_2\}$ where V_L is the time length of video V. Likewise, $]\ \theta_1 \sim \theta_2, \theta_3 \sim \theta_4[$ is a time interval set and its begin-bound and end-bound are extendible. $]\theta_1 \sim \theta_2,\ \theta_3 \sim \theta_4[$ is defined as $\{[\tau_1, \tau_2] | \min(1, \theta_1) \leqslant \tau_1 \leqslant\ \theta_2, \theta_3 \leqslant \tau_2 \leqslant \max(\theta_4, V_L),\ \tau_1 \leqslant \tau_2\}$.

Definition 3. Given two time interval sets Φ and Ψ, the intersection of Φ and Ψ is defined as $\Phi \cap \Psi = \{[\tau_1, \tau_2] | [\tau_1, \tau_2] \in \Phi$ and $[\tau_1, \tau_2] \in \Psi\}$.

Definition 4. Given two spatial relation sequences $\alpha = a_1$:$[t_0, t_1]$, a_2:$[t_1 + 1, t_2], \ldots,$ a_n:$[t_{n-1} + 1, t_n]$ and $\alpha' = a'_1$:$[t'_0, t'_1], a'_2 : [t'_1 + 1, t'_2], \ldots, a'_m$:$[t'_{m-1} + 1, t'_m], n \geqslant m > 0$, where a_k:$[t_{k-1} + 1, t_k]$ means that the spatial relation in the interval $[t_{k-1} + 1, t_k]$ is a_k, if $a_{j_i} = a'_i$, for all $i = 1, 2, \ldots, m$, and $0 \leqslant j_1 \leqslant j_2 \leqslant \cdots \leqslant j_m \leqslant n$, we said that α' is a sub-sequence of α. Function subseq(α', α) means that α' is a sub-sequence of α. Function intervalSubseq(α', α) is defined as follows:

Algorithm: temporal relation inference
Input: the time-projections of objects P and Q
Output: all the possible temporal relations between objects P and Q (Each relation has a time interval set associated with it.)

1. If the temporal relation between objects P and Q is equal to $P{=}Q$, $P{=}Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{End}_{\text{time}}(P)$, $\text{Begin}_{\text{time}}(P)$~$\text{End}_{\text{time}}(P)$].
2. If the temporal relation between objects P and Q is equal to $P<Q$ or $P|Q$, the relation holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{End}_{\text{time}}(P)$, $\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(Q)$].
3. If the temporal relation between objects P and Q is equal to P/Q,
   (a) $P]Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{Begin}_{\text{time}}(Q)$-1, $\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(P)$];
   (b) $P{=}Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(P)$, $\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(P)$];
   (c) $Q[P$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(P)$, $\text{End}_{\text{time}}(P)$+1~$\text{End}_{\text{time}}(Q)$];
   (d) P/Q holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{Begin}_{\text{time}}(Q)$-1, $\text{End}_{\text{time}}(P)$+1~$\text{End}_{\text{time}}(Q)$].
4. If the temporal relation between objects P and Q is equal to $P]Q$,
   (a) $P]Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{Begin}_{\text{time}}(Q)$-1, $\text{Begin}_{\text{time}}(Q)$ ~$\text{End}_{\text{time}}(Q)$];
   (b) $P{=}Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(Q)$, $\text{Begin}_{\text{time}}(Q)$ ~$\text{End}_{\text{time}}(Q)$].
5. If the temporal relation between objects P and Q is equal to $P[Q$,
   (a) $P[Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{End}_{\text{time}}(Q)$, $\text{End}_{\text{time}}(Q)$+1 ~$\text{End}_{\text{time}}(P)$];
   (b) $P{=}Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{End}_{\text{time}}(Q)$, $\text{Begin}_{\text{time}}(P)$ ~$\text{End}_{\text{time}}(Q)$].
6. If the temporal relation between objects P and Q is equal to $P\%Q$,
   (a) $P]Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{Begin}_{\text{time}}(Q)$-1, $\text{Begin}_{\text{time}}(Q)$ ~$\text{End}_{\text{time}}(Q)$];
   (b) $P{=}Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(Q)$, $\text{Begin}_{\text{time}}(Q)$ ~$\text{End}_{\text{time}}(Q)$];
   (c) $P[Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(Q)$~$\text{End}_{\text{time}}(Q)$, $\text{End}_{\text{time}}(Q)$+1 ~$\text{End}_{\text{time}}(P)$].
   (d) $P\%Q$ holds for each interval in the interval set [$\text{Begin}_{\text{time}}(P)$~$\text{Begin}_{\text{time}}(Q)$-1, $\text{End}_{\text{time}}(Q)$+1 ~$\text{End}_{\text{time}}(P)$].
7. Similarly, if the temporal relation between objects P and Q is equal to $Q<P$, $Q|P$, Q/P, $Q]P$, $Q[P$, or $Q\%P$, their possible temporal relations can be computed likewise.

Fig. 4. Temporal relation inference algorithm.

$$\text{intervalSubseq}(\alpha', \alpha') = \begin{cases}]t_0 \sim t_1, t_{j_m-1}+1 \sim t_{j_m}], & j_1 = 1 \text{ and } m \neq n, \\ [t_{j_1-1}+1 \sim t_{j_1}, t_{n-1}+1 \sim t_n[, & j_m = n \text{ and } j_1 \neq 1, \\ ]t_0 \sim t_1, t_{n-1}+1 \sim t_{n-1}[, & j_1 = 1 \text{ and } m = n, \\ [t_{j_1-1}+1 \sim t_{j_1}, t_{j_m-1}+1 \sim t_{j_m}], & \text{otherwise.} \end{cases}$$

For each interval in the interval set intervalSubseq(α', α), α' is a sub-sequence of α in the interval.

Assume that a pair of objects (P, Q) in a video V' matches a pair of objects (P, Q) in another video V. We use three criteria to measure the similarity degree between both matched pair of objects. They are listed as follows:

(1) $\underline{s}$patial relation,
(2) $\underline{t}$emporal relation,
(3) $\underline{d}$uration (that an object lasts in the video).

For each criterion, there are different levels of similarity (defined later). The similarity between (P, Q) in video V' and (P, Q) in video V can be one of combinations of different levels of those criteria. We use the initial character to represent each criterion (the underlined character) and call the similarity the *type-std similarity*. (P, Q) is called a *type-std similar pair* between videos V' and V. Objects P and Q are called *matched objects*.

To define the levels of three criteria, for (P, Q) in video V, we use the following notations to define the similarity measures:

- SRS^{u}_{PQ} (or SRS^{v}_{PQ}): the spatial relation sequence between P and Q in the x (or y) dimension.
- CSS_{PQ}: the category sequence of the spatial relations between P and Q. The concept of the categories of spatial relations was proposed by Lee and Hsu [21]. They divided 169 spatial relations into 5 spatial categories, namely, *disjoin*, *join*, *overlap*, *contain*, and *belong*.

SRS^{u}_{PQ} (or SRS^{v}_{PQ}) can be computed by the *spatial relation inference algorithm*. After both SRS^{u}_{PQ} and SRS^{v}_{PQ} are obtained, we easily obtain CSS_{PQ}.

- $ST_P(a,b)$: the size of the time-projection of object P in the interval $[a,b]$. If the argument (a,b) is omitted, ST_P denotes the size of the time-projection of object P in video V.
- $TR_{PQ}(a,b)$: the temporal relation between the time-projection of object P and that of object Q in the interval $[a,b]$.
- $CTR_{PQ}(a,b)$: the category of $TR_{PQ}(a,b)$. The 13 temporal relations are also divided into five categories, namely, *disjoin*, *join*, *overlap*, *contain*, and *belong*.

If $TR_{PQ}(a,b)$ and $CTR_{PQ}(a,b)$ omit argument (a,b), TR_{PQ} and CTR_{PQ} denote the temporal relation, and the category of the temporal relation between the time-projections of P and Q in the interval $[\min(\text{Begin}_{\text{time}}(P), \text{Begin}_{\text{time}}(Q)), \max(\text{End}_{\text{time}}(P), \text{End}_{\text{time}}(Q))]$, respectively.

For (P, Q) in video V', we can similarly obtain the following information: $SRS^{u'}_{PQ}$ $(SRS^{v'}_{PQ})$, CSS'_{PQ}, ST'_P, TR'_{PQ}, and CTR'_{PQ}.

Since (P, Q) is a type-*std* similar pair between videos V' and V, there exists a sub-interval of $[\min(\text{Begin}_{\text{time}}(P),\text{Begin}_{\text{time}}(Q)), \max(\text{End}_{\text{time}}(P),\text{End}_{\text{time}}(Q))]$ in video V such that the type-*std* similarity holds for (P, Q) in the sub-interval. If the number of such sub-intervals is more than one, they will form an interval set. For each interval in the interval set, the type-*std* similarity holds for (P, Q). Let $[a, b]$ be an interval in the sub-interval set. The levels of each criterion are defined as follows:

"*s*" denotes the similarity degree of the spatial relations between P and Q, and there are four levels.

$s = 0$: none.
$s = 1$: $\text{subseq}(CSS'_{PQ},\ CSS_{PQ})$.
$s = 2$: $\text{subseq}(CSS'_{PQ},\ CSS_{PQ})$ and $(\text{subseq}(SRS^{u'}_{PQ},\ SRS^{u}_{PQ})$ or $\text{subseq}(SRS^{v'}_{PQ},\ SRS^{v}_{PQ}))$.
$s = 3$: $\text{subseq}(SRS^{u'}_{PQ},\ SRS^{u}_{PQ})$ and $\text{subseq}(SRS^{v'}_{PQ},\ SRS^{v}_{PQ})$.

"*t*" denotes the similarity degree of the temporal relations between P and Q, and there are three levels.

$t = 0$: none.
$t = 1$: $CTR'_{PQ} = CTR_{PQ}(a, b)$.
$t = 2$: $TR'_{PQ} = TR_{PQ}(a, b)$.

"*d*" denotes the similarity degree of the duration between the time-projections of P and Q, and there are three levels.

$d = 0$: none.
$d = 1$: $ST'_{P} = ST_{P}(a, b)$ and $ST'_{Q} = ST_{Q}(a, b)$.
$d = 2$: $ST'_{P}/ST_{P}(a, b) = ST'_{Q}/ST_{Q}(a, b)$.

Because type-000 similarity is meaningless, there are $4 \times 3 \times 3 - 1 = 35$ types of similarity in total. For the same criterion, the higher the level is, the stronger the similarity is. For example, type-300 is stronger than type-200 and type-320 is stronger than type-210. For example, the type-111 similarity between (P, Q) in videos V' and V can be expressed that "in video V, there exists a sub-interval $[a, b]$ in $[\min(\text{Begin}_{\text{time}}(P), \text{Begin}_{\text{time}}(Q)), \max(\text{End}_{\text{time}}(P), \text{End}_{\text{time}}(Q))]$ such that $\text{subseq}(CSS'_{PQ}, CSS_{PQ})$, $CTR'_{PQ} = CTR_{PQ}(a, b)$, $ST'_{P} = ST_{P}(a, b)$ and $ST'_{Q} = ST_{Q}(a, b)$ in the sub-interval." If the number of such sub-intervals is more than one, they will form an interval set. For each interval in the interval set, the type-111 similarity holds.

Next, let us consider how to compute the interval set. For each interval in the interval set, the type-*std* similarity holds for (P, Q) in videos V' and V. First, we compute the interval set, Φ, such that the type-*s*00 similarity holds for (P, Q) for each interval in Φ. Second, we compute the interval set, Ψ, such that the type-0*t*0 similarity holds for (P, Q) for each interval in Ψ. Third, let $\Theta = \Phi \cap \Psi$. For each interval in Θ, the type-*st*0 similarity holds for (P, Q). If the criterion of duration, d, is equal to 1, we shall check whether $ST'_{P} = ST_{P}(a, b)$ and $ST'_{Q} = ST_{Q}(a, b)$ for each interval $[a, b]$ in Θ or not. If not, remove the interval from Θ. Similarly, if the criterion of duration, d, is equal to 2, we shall check whether $ST'_{P}/ST_{P}(a, b) = ST'_{Q}/ST_{Q}(a, b)$ for each interval $[a, b]$ in Θ or not. If not, remove the interval from Θ. Finally, for each interval in Θ, the type-*std* similarity holds for (P, Q).

To compute Φ, we can check if the spatial relation sequence, β', between (P,Q) in video V' is a sub-sequence of the spatial relation sequence, β, between (P,Q) in video V. If yes, let $\Phi = \text{intervalSubseq}(\beta', \beta)$; otherwise, $\Phi = \phi$ (empty set). For each interval in Φ, the type-s00 similarity holds for (P,Q).

To compute Ψ, we can apply the *temporal relation inference algorithm* to inferring all possible temporal relations for (P,Q) in video V. Then, we can check if the temporal relation, τ, between (P,Q) in video V' is equal to one of the inferred temporal relations. If yes, let Ψ be the interval set associated with the inferred temporal relation which is equal to τ; otherwise, $\Psi = \phi$. For each interval in Ψ, the type-0t0 similarity holds for (P,Q).

Definition 5. Let $O = \{O_1, O_2, \ldots, O_r\}$, $r \geqslant 2$, be the largest set of matched objects in videos V_1 and V_2 where every pair of matched objects in O is a type-*std* similar pair. The video V_3 constructed from O is a type-*std* common subvideo of V_1 and V_2.

Definition 6. Let video V_3 be the type-*std* common subvideo of videos V_1 and V_2. The type-*std* similarity degree between V_1 and V_2 is defined as the number of objects in V_3, and we say that V_1 is type-*std* similar to V_2. If V_1 is type-*std* similar of V_2 with the largest similarity degree, V_1 is the type-*std* most similar video of video V_2.

To compute the similarity degree between two given videos, V' and V, we build an association graph in which the vertices are formed by the matched objects between two videos. For every type-*std* similar pair, the corresponding vertices in the association graph are connected by an edge. Let K_j be a clique with j vertices, $j \geqslant 2$. If every pair of vertices in K_j is type-*std* similar, we call K_j a type-*std* clique. The number of vertices of the largest type-*std* clique in the association graph is the similarity degree between V' and V. The *similarity retrieval algorithm* is described in detail in Fig. 5.

Let us consider the type-320 similarity. The video V_b shown in Fig. 6 is formed by the frames 5–8 from video V_a as shown in Fig. 1. The 3D C-string of video V_b can be represented as follows:

u-string: $((A_2 \uparrow_{0,1} \uparrow_{1,1} = C_2 \uparrow_{0,\ 1} \uparrow_{0,1.225}) <_1 B_4)$
v-string: $((B_2 = C_2 \uparrow_{0,\ 1} \uparrow_{0,1}) <_2 A_2 \uparrow_{1,1} \uparrow_{0,1})$
t-string: $((A_2|_t A_2 = B_4)]\ C_2|_t C_1)$

There are three objects A, B, and C in both videos V_a and V_b. First, we apply the *spatial relation inference algorithm* to analyzing the spatial relation sequences between each pair of objects in video V_b. The spatial relation sequences are shown as follows:

x-dimension: $A < B$:[1,2], $A|B$:[3,3], A/B:[4,4],
$A = C$:[1,2], C/A:[3,4],
$C < B$:[1,4];
y-dimension: $B < A$:[1,4],
$C < A$:[1,4],
$B = C$:[1,4];

Algorithm: similarity retrieval
Input: type-std (type of similarity), two videos V' and V
Output: the similarity degree between V' and V, the matched objects and interval set associated with V

```
1. Apply the spatial relation inference algorithm to computing the spatial relation sequence for
   each pair of objects in video $V'$.
2. Apply the spatial relation inference algorithm to computing the spatial relation sequence for
   each pair of objects in video $V$.
3. Apply the temporal relation inference algorithm to computing all the possible temporal
   relations for each pair of objects in video $V$.
4. Construct the association graph for videos $V'$ and $V$, where the matched objects between $V'$
   and $V$ form the set of vertices.
5. Find every type-std similar pair between videos $V'$ and $V$. For each type-std similar pair, the
   corresponding vertices in the association graph are connected by an edge which associates
   with an interval set. For each interval in the interval set, the type-std similarity holds.
6. For those edges found in step 5, they form a set of type-std $K_2$.
7. Let $j = 2$.
8. Repeat steps 9~10 until the largest type-std clique is found.
9. If every sub-clique $SK_j$ of a clique $K_{j+1}$ is a type-std $K_j$, construct a type-std clique $K_{j+1}$ with the
   interval set which is the intersection of the interval sets associated with those $SK_j$.
10. Let $j = j+1$.
11. Output the number of vertices, matched objects and interval set associated with the largest
    type-std clique, where the number of vertices is the similarity degree between $V'$ and $V$.
```

Fig. 5. Similarity retrieval algorithm.

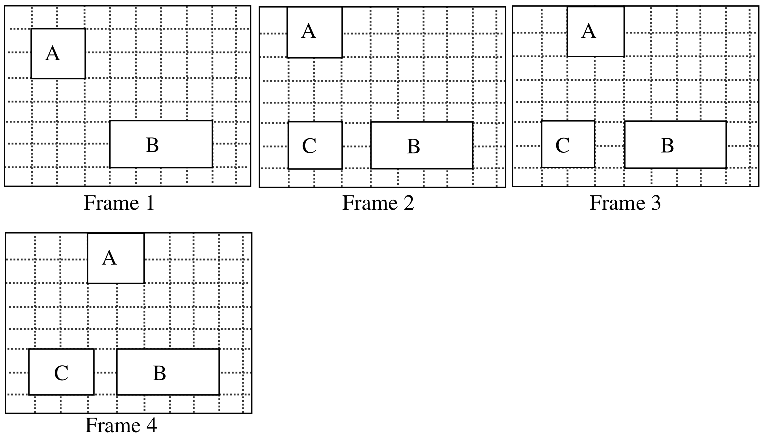


Fig. 6. Video V_b.

time-dimension: $A = B$,
$A]C$,
$B]C$.

Second, we apply the *spatial relation inference algorithm* to analyzing the spatial relation sequences between each pair of objects in video V_a. The spatial relation sequences are shown as follows where the spatial relation between objects A and C in frames 1–5 is determined by comparing object C in frame 6 with object A in every frame from frame 1 to frame 5. The spatial relation sequence between objects B and C is determined in a similar way.

x-dimension: $A < B$:[1,6], $A|B$:[7,7], A/B:[8,8], $B[A$:[9,9],
$A = C$:[1,6], C/A:[7,8], $C < A$:[9,9],
$C < B$:[1,9];
y-dimension: $A = B$:[1,1], B/A:[2,2], $B|A$:[3,3], $B < A$:[4,9],
$A = C$:[1,1], C/A:[2,2], $C|A$:[3,3], $C < A$:[4,9],
$B = C$:[1,9].

Third, we apply the *temporal relation inference algorithm* to computing all the possible temporal relations between each pair of objects in video V_a. The temporal relations are listed as follows:

time-dimension: $A = B$:[$1 \sim 9, 1 \sim 9$],
$A]C$:[$1 \sim 5, 6 \sim 9$], $A = C$:[$6 \sim 9, 6 \sim 9$],
$B]C$:[$1 \sim 5, 6 \sim 9$], $B = C$:[$6 \sim 9, 6 \sim 9$].

Since objects A, B, and C are the matched objects in both videos, we can construct the association graph with three vertices (A, B, and C). Let us first consider the type-320 similarity between the similar pair (A, B). For the spatial relation sequence between objects A and B in the x dimension, we find that $A < B$:[1,2], $A|B$:[3,3], A/B:[4,4] in V_b is a sub-sequence of $A < B$:[1,6], $A|B$:[7,7], A/B:[8,8], $B[A$:[9,9] in V_a and function intervalSubseq returns]$1 \sim 6, 8$]. For the spatial relation sequence between objects A and B in the y dimension, $B < A$:[1,4] in V_b is a sub-sequence of $A = B$:[1,1], B/A:[2,2], $B|A$:[3,3], $B < A$:[4,9] in V_a and function intervalSubseq returns [$4 \sim 9, 4 \sim 9$[. For the temporal relation in the time dimension, we find that the temporal relation between objects A and B in V_b is $A = B$ and $A = B$ holds in V_a for each interval of [$1 \sim 9, 1 \sim 9$]. Hence, we use an edge to connect vertices A and B in the association graph as shown in Fig. 7. The interval set associated with edge $\overline{AB}$ is the intersection of interval sets]$1 \sim 6, 8$], [$4 \sim 9, 4 \sim 9$[and [$1 \sim 9, 1 \sim 9$]. The intersection is equal to [$4 \sim 6, 8$]. When we consider the type-320 similarity between the similar pair (A, C), we can find that the interval set associated with edge $\overline{AC}$ is [$4 \sim 5, 7 \sim 8$]. Similarly, when we consider the type-320 similarity between the similar pair (B, C), we can find that the interval set associated with edge $\overline{BC}$ is [$1 \sim 5, 6 \sim 9$].

Now, we have a type-320 K_2 set which is $\{\overline{AB}$: [$4 \sim 6, 8$], $\overline{AC}$: [$4 \sim 5, 7 \sim 8$], $\overline{BC}$: [$1 \sim 5, 6 \sim 9$]$\}$. Since all the sub-cliques SK_2 of K_3 ABC are in the type-320 K_2 set, we can obtain a type-320 K_3. The interval set associated with the type-320 K_3 ABC is the intersection of the interval sets associated with $\overline{AB}$, $\overline{AC}$, and $\overline{BC}$. That is, the interval set associated with the type-320 K_3 ABC is equal to the intersection of [$4 \sim 6, 8$], [$4 \sim 5, 7 \sim 8$], and [$1 \sim 5, 6 \sim 9$]. The intersection is

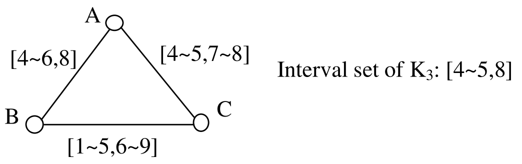


Fig. 7. The association graph based on the type-320 similarity.

equal to $[4 \sim 5, 8]$. Hence, the similarity degree between V_a and V_b is 3. Since the interval set $[4 \sim 5, 8]$ contains two intervals $[4, 8]$ and $[5, 8]$, we have two sub-videos which are type-320 similar to video V_b. One is formed by extracting frames $4 \sim 8$ from video V_a. The other is formed by extracting frames $5 \sim 8$ from video V_a. The result is shown in Fig. 7.

If we impose a stronger constraint to find the type-321 (or type-322) similarity between videos V_a and V_b, we shall find a type-321 (or type-322) K_3 associated with an interval $[5, 8]$. The sub-video formed by extracting frames $5 \sim 8$ from video V_a is exactly the same as video V_b.

5. Performance analysis

To compare the performance of our *similarity retrieval algorithm* with the *9DLT approach* [2], we perform two series of experiments. The first series of experiments is made on the synthesized video indexes. There are four cost factors dominating the performance of the *similarity retrieval algorithm*: the number of objects in a query video, the number of database videos, the average number of objects in a database video, and the average number of frames in a database video. We freely set the values of the four cost factors in the synthesized video database. Since the string matching algorithms proposed in the past are not suitable to apply on 3D C-string, we also use a naive algorithm for the comparison. The naive algorithm goes quite simple: to match the *u*-, *v*-, or *t*-strings of a database video with the *u*-, *v*-, or *t*-string of a query video, we attempt to compare every possible combination of objects. If a combination satisfies a certain type-*std* similarity, we append it to the similar set. After finding all the satisfying combinations, we use the ones with the largest similarity degree to be the query result.

The second series of experiments is made on 200 real videos. Each video is a video clip about one minute long. The objects in a video are specified by using the video index tool. All algorithms are implemented on a Pentium III-800 PC with Windows 2000.

5.1. Video index generation and query types

In these two series of experiments, a preprocessing phase is needed to generate and reconstruct all database videos or query videos. But in this paper we want to show the efficiency of our *similarity retrieval algorithm*, so we do not add the cost of video generation and reconstruction to the execution time.

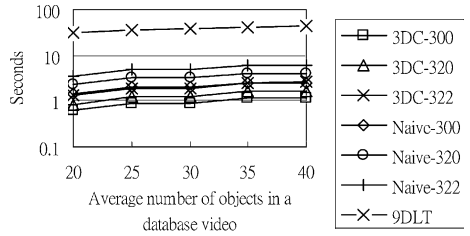
In query types, we choose three different types from 35 types to compare the cost of executing queries. They are: type-300 in which we retrieve the videos having the same spatial relation sequence; type-320 in which we retrieve the videos having the same temporal relation and spatial relation sequence; type-322 in which we retrieve the videos having the same temporal relation and spatial relation sequence, and the duration adjusted proportionally.

5.2. Synthesized videos

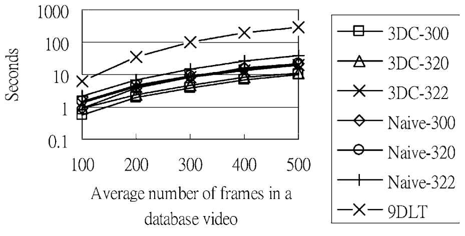
In this subsection, we show the efficiency of our *similarity retrieval algorithm* and compare it with the 9DLT and naive algorithms. The execution cost of every experiment is measured by the average elapsed time of 100 video queries. Since we can specify the number of objects in the query videos, the number of objects is set in the range between 4 and 12, and each video is 1 min long. We freely set the values to generate 1000 database videos. For each database video, we assign 20–50 objects to it. The number of frames in a database video is in the range from 1000 to 5000. Based on these synthesized video indices, we perform four experiments. In each experiment we change one cost factor and fix the other three cost factors. The values we choose for the fixed cost factors are 8 objects in a query video, 200 database videos, 25 objects, and 2000 frames in each database video. The experiment results are shown as follows.

Fig. 8A illustrates the execution time versus the number of objects in a query video. The execution time of the 3D C-string approaches grows slowly as the number of objects in a query increases. However, the execution time of the naïve approaches grows sharply as the number of objects in a query increases since they have to check most combinations of objects in each database video. Since the 9DLT approach needs to compare every query frame with the frames in its index structure of the database videos, it takes the most execution time among all of the approaches. All the 3D C-string approaches need less execution time than the naïve approaches do. The 3D C-string approach is 2–20 times faster than the 9DLT approach.

Fig. 8B illustrates the execution time versus the number of videos in the database. The execution time grows nearly linearly as the number of database videos increases for all the approaches in all three types of video queries. All the 3D C-string approaches need less execution time than the naive approaches do. The 3D C-string approach is 2–18 times faster than the 9DLT approach.

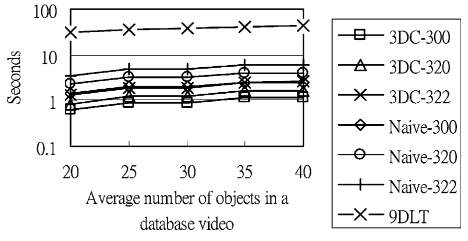


(A) Number of objects in a query.

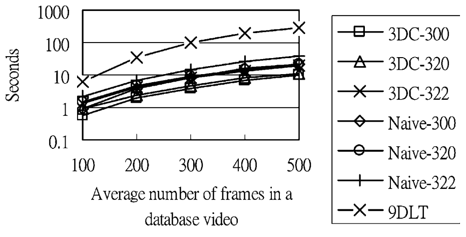


(B) Number of database videos.

Fig. 8. Execution time vs. the number of query objects and database videos.



(A) Number of objects in a database video.



(B) Number of frames in a database video.

Fig. 9. Execution time vs. the number of objects and frames in a database video.

Fig. 9A illustrates the execution time versus the number of objects of in a database video. The execution time grows as the number of objects in a database video increases for all the approaches in all three types of video queries. All the 3D C-string approaches need less execution time than the naive approaches do. The 3D C-string approach is 30–50 times faster than the 9DLT approach.

Fig. 9B illustrates the execution time versus the number of frames in a database video. The execution time grows nearly linearly as the number of frames in a database video increases for all the approaches in all three types of video queries. All the 3D C-string approaches need less execution time than the naive approaches do. The 3D C-string approach is 6–30 times faster than the 9DLT approach.

In summary, the execution time of both 3D C-string and naïve approaches increases nearly linearly as the number of database videos or the number of frames in a database video increases. The execution time of the 3D C-string approaches grows comparatively slowly as the number of objects in a query increases. For all cases, all the 3D C-string approaches need less execution time than the naïve approaches do. The 3D C-string approach is 2–50 times faster than the 9DLT approach. For the type-322 queries, most of the execution time is spent on computing and matching spatial relation sequences. Computing and matching temporal relations takes the second most time. Checking the criterion of duration takes the least time.

5.3. Real videos

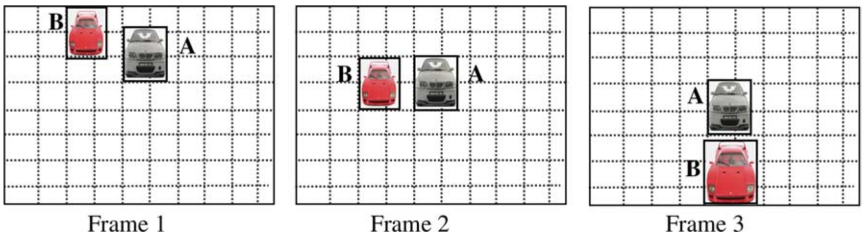
In this subsection, we show the efficiency of our *similarity retrieval algorithm* on the real videos. First, we demonstrate a video query example. Then, we present the performance analysis on the real videos. In the example video database, there are four sets of videos: 60 videos of traffic, 60 videos of campus activities, 40 videos of cartoons, and 40 videos of TV news. All videos are 1 min long. Typically, a video of 1 min long contains 1800 frames. To represent the movements of objects, at least a frame should be indexed for every 10 frames.

Next, an example of video query is presented in Fig. 10. The query video represented by a 3D C-string is shown in Fig. 10A. The query video represents an overtaking event in which car *B* overtakes car *A*. The corresponding video is shown in u-string: $(B_{14}\uparrow_{0,1}\downarrow_{18,1}<_{6}A_{16}\downarrow_{0,1})$

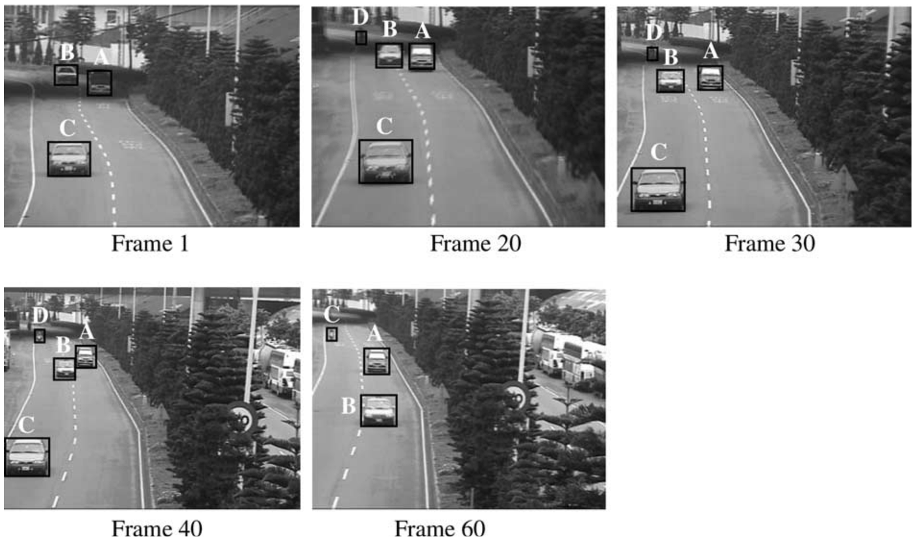
v-string: $(A_{22}\downarrow_{10,1}<_{8}B_{20}\downarrow_{20,1}\downarrow_{40,1})$

t-string: $(A_3{=}B_2|_tB_1)$

(A) A 3D C-string query.



(B) The corresponding video of the 3D C-string in (A).



(C) A matched database video.

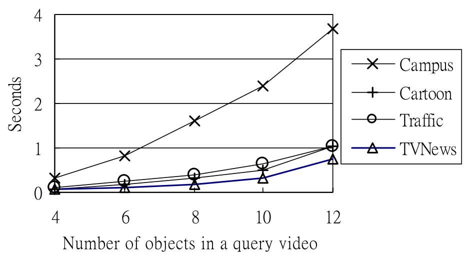
Fig. 10. Example of the type-320 similarity.

Fig. 10B. A database video, which is type-320 similar to the query video, is shown in Fig. 10C.

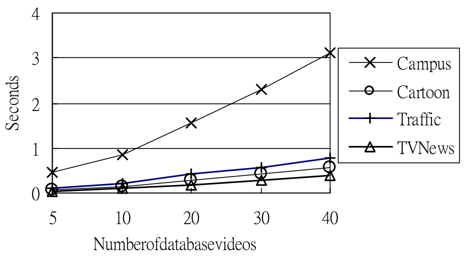
The second part of the experiment is performance analysis on real videos. In this experiment, we use four sets of videos. Generally speaking, different set of videos may contain the different number of objects. For example, the traffic videos contain more objects than the other three sets of videos. The average number of objects in four sets of videos is about 37. The average number of objects of each set of videos is listed in Table 2.

Table 2
Average number of objects in real videos

Videos	Campus	Cartoon	Traffic	TV news
Average number of objects	18	28	64	31



(A) Number of objects in a query.



(B) Number of database videos.

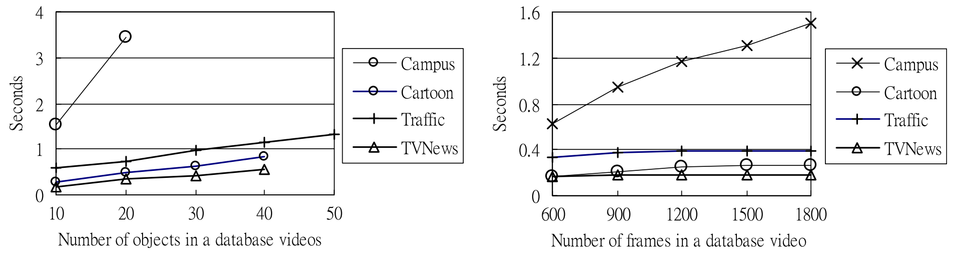
Fig. 11. Execution time vs. the number of query objects and database videos.

The execution cost of each experiment is measured by the average elapsed time of 20 video queries. Each video is around one minute long. We perform four experiments. In each experiment, we change one cost factor and fix the other three cost factors. The values we choose for the fixed cost factors are 8 objects in a query video, 40 database videos, and 1800 frames in each database video.

Fig. 11A illustrates the execution time versus the number of objects in a query video for the different sets of videos. The execution time increases as the number of objects in a query video increases. The execution time of querying the traffic, cartoon, and TV news videos is quite close. However, the execution time of querying the campus activities video is obviously greater than that of querying the other sets of videos. Because a video of campus activities contains a lot of motions and size changes and the corresponding 3D C-string contains a lot of motion operators, it is required much time to process the string.

Fig. 11B illustrates the execution time versus the number of database videos for the different sets of videos. The execution time grows nearly linearly as the number of database videos increases. The execution time of querying the traffic, cartoon, and TV news videos is quite close. However, the execution time of querying the campus activities video is obviously greater than that of querying the other sets of videos because a video of campus activities contains a lot of motion operators.

Fig. 12A illustrates the execution time versus the number of objects in a database video for the different sets of videos. The execution time grows as the number of objects in a database video increases. The execution time of querying campus activities videos does not grow when the number of objects in a database video is greater than 20. This is because there is no campus activities video containing more than 20 objects. Similarly, there is no cartoon or TV news video, which contains more than 40 objects. The execution time of querying cartoon and TV news videos does not grow



(A) Number of objects in a database video. (B) Number of frames in a database video.

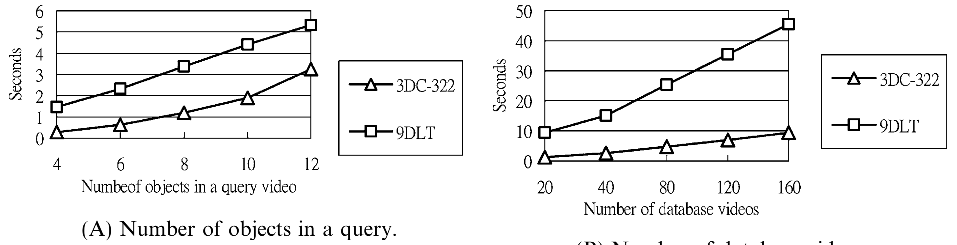
Fig. 12. Execution time vs. the number of objects and frames in a database video.

when the number of objects in a database video is greater than 40. The execution time of querying traffic videos keeps growing as the number of objects increases.

Fig. 12B illustrates the execution time versus the number of frames in a video for the different sets of videos. The execution time increases as the number of frames in a database video increases. The execution time of querying campus activities videos is greater than that of querying traffic videos, which is greater than that of querying cartoon videos. The execution time of querying cartoon videos is greater than that of querying TV news videos.

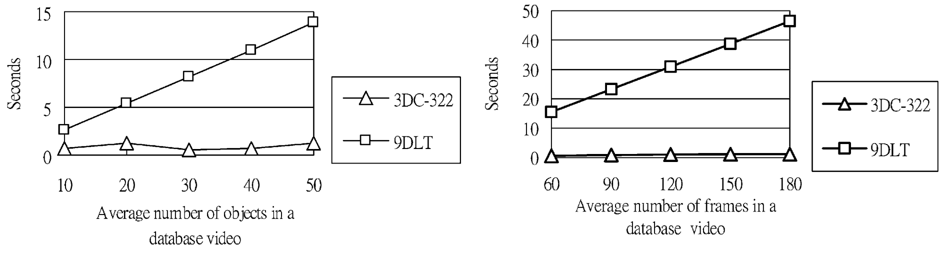
Fig. 13A illustrates the execution time versus the number of objects in a query video for the 9DLT and 3D C-string approaches. The execution time of the 3D C-string approach grows slowly as the number of objects in a query increases. However, the execution time of the 9DLT approach grows sharply as the number of objects in a query increases since the 9DLT approach needs to compare every query frame with the frames in its index structure of the database videos. Fig. 13B illustrates the execution time versus the number of videos in the database. The execution time grows nearly linearly as the number of database videos increases for both approaches. The 3D C-string approach needs less execution time than the 9DLT approach.

Fig. 14A illustrates the execution time versus the number of objects of in a database video for the 9DLT and 3D C-string approaches. The execution time grows as the number of objects in a database video increases for both approaches. The 3D C-string approach needs less execution time than 9DLT approach. Fig. 14B illustrates the



(A) Number of objects in a query. (B) Number of database videos.

Fig. 13. Execution time vs. the number of query objects and database videos.



(A) Number of objects in a database video. (B) Number of frames in a database video.

Fig. 14. Execution time vs. the number of objects and frames in a database video.

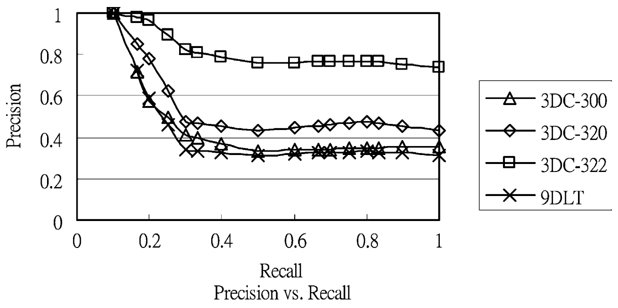


Fig. 15. Precision vs. recall.

execution time versus the number of frames in a database video. The execution time grows nearly linearly as the number of frames in a database video increases for both approaches. The 3D C-string approach needs less execution time than the 9DLT approach.

Fig. 15 illustrates the precision versus recall for the 3D C-string and 9DLT approaches. As expected, the result of type-322 query is better than that of type-320 query, which is better than that of type-300 query. The results of type-300 query and 9DLT-string are quite close, because both approaches only use the spatial relations to query the database.

In summary, the execution time of querying the traffic, cartoon, and TV news videos is quite close. However, the execution time of querying the campus activities video is obviously greater than that of querying the other sets of videos. The average number of objects in the campus activities videos is the smallest among them. However, an object in this set of videos changes its states quite often including motions or rates of size changes. Hence, the 3D C-strings generated by this set of videos contain a lot of motion operators. The execution time of querying the campus activities videos is higher than expected. In comparison with the 9DLT approach, the 3D C-string needs less execution time. Moreover, the 3D C-string can provide various types of similarity between videos and have discrimination power about different criteria.

6. Concluding remarks

We have proposed a new spatio-temporal knowledge structure called 3D C-string to represent symbolic videos accompanying with the string generation and video reconstruction algorithms. In this paper, we extend the idea behind the similarity retrieval of images in 2D C^+-string [15] to 3D C-string. Our extended approach consists of two phases. First, we infer the spatial relation sequence and temporal relations for each pair of objects in a video. Second, we use the inferred relations and sequences to define various types of similarity measures and propose the similarity retrieval algorithm. Three criteria are used to define similarity measures. The concept of processing spatial relation sequences and temporal relations can also be easily extended to process other criteria such as velocities, rates of size changes, distances, and so on. We also show that different types of similarity have different multi-granularity to meet users' need by examples. By providing various types of similarity between videos, our proposed similarity retrieval algorithm has discrimination power about different criteria. Our proposed approach can be easily applied to an intelligent video database management system to infer spatial and temporal relations between the objects in a video and to retrieve the videos similar to a query video from a video database.

A video contains rich visual and audio (or sound) information. In the 3D C-string representation and the similarity retrieval algorithm based on the 3D C-string, we focused on utilizing the visual information to process videos. How to integrate the audio information with the visual information to represent a video and perform similarity retrieval is worth further study.

Acknowledgments

The authors are grateful to the referees for valuable comments and suggestions. This research was supported in part by Center for Information and Electronics Technologies (CIET), National Taiwan University and the National Science Council of Republic of China under Grant No. NSC-91-2213-E-002-092.

References

[1] Y. Caspi, M. Irani, Spatio-temporal alignment of sequences, IEEE Trans. Pattern Anal. Mach. Intell. 24 (11) (2002) 1409–1424.

[2] Y.K. Chan, C.C. Chang, Spatial similarity retrieval in video databases, J. Visual Commun. Image Represent. 12 (2001) 107–122.

[3] S. Chang, W. Chen, H.J. Meng, H. Sundaram, D. Zhong, VideoQ: an automated content-based video search system using visual cues, in: Proceedings of the ACM International Conference on Multimedia Conference, Seattle, WA, 1997, pp. 313–324.

[4] S.K. Chang, Q.Y. Shi, C.W. Yan, Iconic indexing by 2D strings, IEEE Trans. Pattern Anal. Mach. Intell. 9 (1987) 413–429.

[5] Y.I. Chang, H.Y. Ann, W.H. Yeh, A unique-ID-based matrix strategy for efficient iconic indexing of symbolic pictures, Pattern Recogn. 33 (2000) 1263–1276.

[6] Y.I. Chang, B.Y. Yang, W.H. Yeh, A generalized prime-number-based matrix strategy for efficient iconic indexing of symbolic pictures, Pattern Recogn. Lett. 22 (2001) 657–666.

[7] Y.I. Chang, B.Y. Yang, W.H. Yeh, A bit-pattern-based matrix strategy for efficient iconic indexing of symbolic pictures, Pattern Recogn. Lett. 24 (2003) 537–545.

[8] W.W. Chu, A.F. Cardenas, R.K. Taira, KMeD: a knowledge-based multimedia medical distributed database system, Inf. Syst. 20 (2) (1995) 75–96.

[9] N. Dimitrova, F. Golshani, Rx for semantic video database retrieval, in: Proceedings of the ACM International Conference on Multimedia, San Francisco, CA, 1994, pp. 219–226.

[10] A.D. Doulamis, N.D. Doulamis, S.D. Kollias, A fuzzy video content representation for video summarization and content-based retrieval, Signal Process. 80 (2000) 1049–1067.

[11] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, Query by image and video content: the QBIC system, IEEE Comput. 28 (1995) 23–32.

[12] S.J.F. Guimar, M. Michel Couprie, A.D.A. Arauj, M.J. Leite, Video segmentation based on 2D image analysis, Pattern Recogn. Lett. 24 (2003) 947–957.

[13] R.H. Guting, M.H. Bohlen, M. Ervig, C.S. Jensen, N.A. Lorentzos, M. Schneider, M. Vazirgiannis, A foundation for representing and querying moving objects, ACM Trans. Database Syst. 25 (1) (2000) 1–42.

[14] F.J. Hsu, S.Y. Lee, B.S. Lin, Video data indexing by 2D C-trees, J. Visual Languages Comput. 9 (1998) 375–397.

[15] P.W. Huang, Y.R. Jean, Using 2D C^{+}-string as spatial knowledge representation for image database systems, Pattern Recogn. 27 (1994) 1249–1257.

[16] E. Jungert, Extended symbolic projections as a knowledge structure for spatial reasoning, in: Proceedings of the 4th BPRA Conference on Pattern Recognition, 1988, pp. 343–351.

[17] F. Kokkoras, H. Jiang, I. Vlahavas, A.K. Elmagarmid, E.N. Houstis, W.G. Aref, Smart VideoText: a video data model based on conceptual graphs, Multimedia Syst. 8 (4) (2002) 328–338.

[18] A.J.T. Lee, H.P. Chiu, 2D Z-string: a new spatial knowledge representation for image databases, Pattern Recogn. Lett. 24 (2003) 3015–3026.

[19] A.J.T. Lee, H.P. Chiu, P. Yu, 3D C-string: a new spatio-temporal knowledge structure for video database systems, Pattern Recogn. 35 (2002) 2521–2537.

[20] S.Y. Lee, F.J. Hsu, 2D C-string: a new spatial knowledge representation for image database system, Pattern Recogn. 23 (1990) 1077–1087.

[21] S.Y. Lee, F.J. Hsu, Spatial reasoning and similarity retrieval of images using 2D C-string knowledge representation, Pattern Recogn. 25 (1992) 305–318.

[22] S.Y. Lee, H.M. Kao, Video indexing-an approach based on moving object and track, Proc. IS&T/SPIE 1908 (1993) 25–36.

[23] Z. Lei, Y.T. Lin, 3D shape inferencing and modeling for Video Retrieval, J. Visual Commun. Image Represent. 11 (2000) 41–57.

[24] J.Z. Li, M.T. Ozsu, D. Szafron, Modeling of moving objects in a video database, in: Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Ottawa, Canada, 1997, pp. 336–343.

[25] C.C. Liu, A.L.P. Chen, 3D-list: a data structure for efficient video query processing, IEEE Trans. Knowledge Data Eng. 14 (2002) 106–122.

[26] Y. Liu, J.R. Kender, Fast video segment retrieval by Sort-Merge feature selection, boundary refinement, and lazy evaluation, Comput. Vision Image Understand. 92 (2003) 147–175.

[27] C.C. Lo, S.J. Wang, L.W. Huang, Video retrieval using successive modular operations on temporal similarity, Comput. Standards Interfaces 26 (2004) 317–328.

[28] M. Nabil, A.H. Ngu, J. Shepherd, Modeling and retrieval of moving objects, Multimedia Tools Appl. 13 (2001) 35–71.

[29] A. Nagasaka, Y. Tanaka, Automated video indexing and full-video search for object appearance. Visual Database Systems II, 1992.

[30] M.R. Naphade, T.S. Huang, A probabilistic framework for semantic video indexing, filtering, and retrieval, IEEE Trans. Multimedia 3 (1) (2001) 141–151.
[31] C.W. Ngo, T.C. Pong, H.J. Zhang, Motion analysis and segmentation through spatio-temporal slices processing, IEEE Trans. Image Process. 12 (3) (2003) 341–355.
[32] E. Oomoto, K. Tanaka, OVID: design and implementation of a video object database system, IEEE Trans. Knowledge Data Eng. 5 (1993) 629–643.
[33] G. Petraglia, M. Sebillo, M. Tucci, G. Tortora, Virtual images for similarity retrieval in image databases, IEEE Trans. Knowledge Data Eng. 13 (6) (2001) 951–967.
[34] K. Shearer, S. Venkatesh, D. Kieronska, Spatial indexing for video databases, J. Visual Commun. Image Represent. 7 (1997) 325–335.
[35] A.P. Sistla, O. Wolfson, S. Chamberlain, S. Dao, Modeling and querying moving objects, Proc. IEEE Int. Conf. Data Eng. 1 (1997) 422–432.
[36] C.G. Snoek, M. Worring, Multimedia event based video indexing using time intervals, IEEE Trans. Multimedia (2004) 1–10.
[37] H. Yang, L. Chaisorn, Y. Zhao, S. Neo, T. Chua, VideoQA: question answering on news video, Proc. ACM Intl. Conf. Multimedia (2003) 632–641.
[38] M.M. Yeung, B.L. Yeo, B. Liu, Extracting story units from long programs for video browsing and navigation, Proc. Intl. Conf. Multimedia Comput. Syst. (1996).
[39] D. Zhong, H.J. Zhang, S.F. Chang, Clustering methods for video browsing and annotation. Proc. Storage Retrieval Image Video Database IV, IS&T/SPIE's Electronic Imaging, 1996.