Ninth IEEE International Symposium on Multimedia 2007

A Temporal UID Matrix Strategy for Indexing Video Databases

Ye-In Chang, Wei-Horng Yeh, Jiun-Rung Chen, and You-Cheng Chen
Dept. of Computer Science and Engineering, National Sun Yat-Sen University
Kaohsiung, Taiwan, R.O.C
Telephone: (886) 7–525–4350
Fax: (886) 7–525–4301

Abstract

In video databases, each video contains temporal and spatial relationships between content objects. One of the well-known video indexing strategies is the 3D C-string strategy. However, it cannot deal with the condition that an object appears and then disappears for more than one time. To solve this problem, in this paper, we propose an indexing strategy, called Temporal UID Matrix[1]*. Based on the original 13 spatial relationships proposed by 2D C-string and our three new spatial relationships, we can derive the temporal relationships from the sequence of spatial relationships. Therefore, in our proposed strategy, although we build only index for spatial relationships, and we still can answer the video queries, i.e., spatial, temporal, and spatio-temporal queries. From our simulation study, we show that our proposed strategy is more efficient for video searching than the 3D C-string strategy.*

1 Introduction

With the advances in computer technologies, digital video becomes popular, including digital library, interactive video analysis, multimedia publishing, and geographic information systems, *etc*. Much attention has been paid to the design of video database systems for discriminating the videos based on the information of spatial and temporal relationships. The difficulty in designing a content-based video database system is how to store and describe the spatio-temporal relationships between moving objects completely.

Many content-based video retrieval systems have been proposed in [1, 4, 5, 8, 11, 12]. The indexes by content can be classified into three categories: videos classifying methods [7], iconic indexing methods for videos [2, 6], and syntactic structuring methods [13]. The iconic indexing methods for videos concern issues of dividing videos into frames and supporting query based on spatial, temporal, and spatio-temporal relationships among the objects. For example, one of the queries may be "Please search videos that A jumps over B."

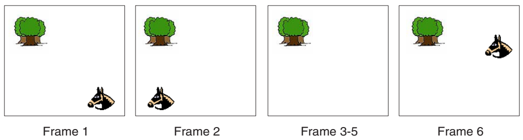
A video consists of a sequence of frames in general. Since the relationships between two symbol objects in a frame are semantically stronger than those in two different frames, it is convenient to adopt a frame as a basic index unit. Some of video indexing and retrieving strategies used semantic information about the meaning of the frames to describe an video, and therefore require intensive manual annotation [9]. While other methods extended those index strategies for images in the 2D space to index video data by taking the $time$-axis into consideration.

Lee and Hsu proposed the 2D C-string representation used in 2D space, and then they proposed a 2D C-tree representation [6] and employed it to describe the spatial information of video contents. Then, Chan *et al.* [2] transformed the 9DLT matrix into a linear string, 9DLT string, and defined a common component binary tree (CCBT) structure to solve the sequence matching problem between a query frame sequence and a video frame sequence. In [14], Liu and Chen extended the notion of the 2D string and proposed the 3D string to represent the relationships between the objects in a video. The knowledge structure of 3D string used the projections of video objects to represent spatial and temporal relations between them and a video object is represented by its central point and starting frame number.

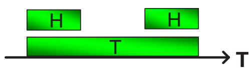
Lee *et al.* [10] extended the concepts of the 2D C^+-string to propose the 3D C-string to derive spatio-temporal relationships by keeping track of the motions and size changes associated with the video objects. It builds the $time$-string to index the temporal relationships between objects as the spatial indexes. There is an assumption in the 3D C-string strategy that an object is not allowed to reappear if it has dis-

[1]This research was supported in part by the National Science Council of Republic of China under Grant No. NSC-87-2213-E-110-014. The authors also like to thank "Aim for Top University Plan" project of NSYSU and Ministry of Education, Taiwan, for partially supporting the research.

0-7695-3058-3/07 $25.00
DOI 10.1109/ISM.2007.9



(a) six continuous frames



(b) relationships in the $time$-axis

Figure 1. Video V_1

appeared before. For example, Figure 1 shows that a horse is running from the right bottom to the left bottom, and then it disappears during the following three frames. Finally, it shows up in the right top in the last frame. Thus, there are two different temporal relationships between objects $horse$ and $tree$ in Video V_1. However, the 3D C-string strategy cannot index the video in this case. Another drawback of 3D C-string strategy is the complex representation of the strings themselves. It is time-consuming to derive the spatial relationships based on the x- and y-strings in the 3D C-string strategy.

To solve the above discussed problems, in this paper, first, we introduce three new spatial relationships. With the original 13 spatial relationships and the three new ones, we propose a video indexing strategy. Our proposed strategy records only spatial relationships between objects in a video, and can derive the temporal relationships from the sequence of spatial relationships. Finally, we compare the the performance of our proposed strategy with that of the 3D C-string strategy.

The rest of this paper is organized as follows. Section 2 briefly describes the meaning of the 13 spatial relationships and the UID matrix strategy [3], which is related to our proposed strategy. Section 3 presents our proposed video indexing strategy, the Temporal UID matrix strategy. Section 4 studies the performance of our proposed strategy. Finally, Section 5 gives the conclusions.

2 The Related Work

Table 1 describes the meaning of the spatial operators proposed by Lee and Hsu [6]. The illustration of the spatial operators and the inverse ones is shown in Figure 2. Let us denote the white rectangle by O_1 and the gray rectangle by O_2. If the spatial relationship between O_1 and O_2 is "$O_2 < O_1$" as shown in the second row and the first column in Figure 2, then we could use the inverse operator to represent the spatial relationship as "$O_1 <^* O_2$".

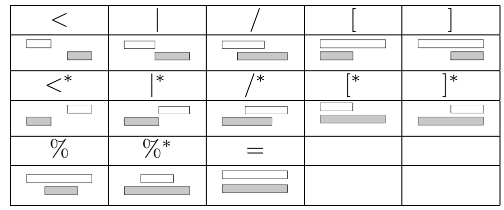


Figure 2. 13 types of spatial operators

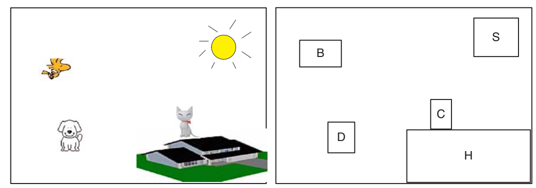


Figure 3. An image and its symbolic representation

Chang *et al.* [3] proposed an iconic indexing scheme called the Unique-ID-based Matrix (UID Matrix) for symbolic pictures in which each spatial relationship between any two objects is assigned to a unique number and is recorded in a matrix. The assignments are shown in Table 2.

Take Figure 3 as an example. According to the *uid* values, the corresponding *UID Matrix* T is shown as follows:

$$T = \begin{array}{c} \\ S \\ B \\ D \\ H \\ C \end{array} \begin{array}{c} \begin{array}{ccccc} S & B & D & H & C \end{array} \\ \left[\begin{array}{ccccc} 0 & 6 & 2 & 2 & 2 \\ 2 & 0 & 2 & 2 & \mathbf{2} \\ 2 & 5 & 0 & 6 & 5 \\ 13 & 1 & 1 & 0 & 3 \\ 2 & \mathbf{1} & 1 & 9 & 0 \end{array} \right] \end{array}$$

The lower left triangular area of the matrix records the spatial relationships among the *x*-axis, and the upper right triangular area of the matrix stores the relationships among the *y*-axis. For example, the *uid* numbers for the cat (C) and the bird (B) among the *x*- and *y*-axes are 1 and 2, respectively. Thus, we could derive that the bird is in the upper left of the cat.

3 Video Indexing Strategy

In this Section, we first present the definition of the three new spatial relationships. Next, we describe how to use these spatial relationships to derive the temporal relationships from a sequence of spatial relationships. Then, we

Table 1. Definitions of Lee *et al.*'s spatial operators

Notation	Condition	Meaning
$A < B$	end(A) $<$ begin(B)	A disjoins B
$A\|B$	end(A) $=$ begin(B)	A is edge to edge with B
A/B	begin(A) $<$ begin(B) $<$ end(A) $<$ end(B)	A is partly overlapping with B
$A]B$	begin(A) $<$ begin(B) end(A) $=$ end(B)	A contains B and they have the same end bound
$A[B$	begin(A) $=$ begin(B) end(A) $>$ end(B)	A contains B and they have the same begin bound
$A\%B$	begin(A) $<$ begin(B) end(A) $>$ end(B)	A contains B and they do not have the same bound
$A = B$	begin(A) $=$ begin(B) end(A) $=$ end(B)	A is at the same position as B

Table 2. Uids of 13 spatial operators

$operator$	$<$	$<^*$	$\|$	$\|^*$	$/$	$/^*$	$]$	$[$	$\%$	$=$	$]^*$	$[^*$	$\%^*$
uid	1	2	3	4	5	6	7	8	9	10	11	12	13

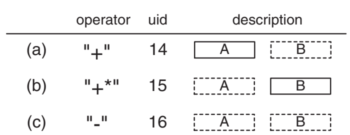


Figure 4. Definitions of new operators: (a) "$+$" (14, B disappears); (b) "$+^*$" (15, A disappears); (C) "$-$" (16, both objects disappear).

present our proposed *Temporal UID matrix strategy*. Finally, we describe the algorithms to do the similarity retrieval with our proposed strategy.

3.1 Three New Spatial Relationships

In our proposed strategy, we make use of a sequence of spatial relationships to derive the temporal relationship. Thus, we define three new spatial relationships to achieve this goal. There are 13 spatial relationships among x- and y-axes. In the $time$-axis, there are 13 temporal relationships as well. Thus, we use the same notation of the spatial relationships to represent the temporal relationships. For example, in the x-axis, $A < B$ represents that the projection of object A is before that of object B. In the $time$-axis, $A < B$ denotes that object A disappears before object B appears.

Those three new spatial operators are shown in Figure 4 in which the dotted rectangle means object disappearance. The first one is operator "$A + B$," which denotes the spatial relationship that object A appears but object B disappears in a frame. The second one is operator "$A +^* B$," which denotes the spatial relationship that object A disappears but object B appears in a frame. The last one is operator "$A - B$," which denotes the spatial relationship that object A and object B both disappear in a frame. For example, Figure 1-(a) shows that a horse is running from the lower-right corner to the lower-left corner. Then, the horse disappears and shows up in the upper-right corner in the next three frames. The spatial relationship between the horse and the tree in frames 3–5 can be represented by "$tree{+}horse$." We extend 13 spatial operators to 16 ones and assign each of those 16 spatial operators to a unique identifier as shown in Table 3.

There are 13 temporal operators in $time$-axis with the corresponding unique identifiers as shown in Table 4. Suppose the temporal relationship between objects A and B is denoted by $(Ar^t_{A,B}B)$. The corresponding *uid* value is $(Auid^t_{A,B}B)$. In the spatial index structure, we record the spatial information of objects that appear in the same shot. The spatial relationship between objects A and B in the x- (or y-) axis may change over time in the video. This change can be observed from a spatial relationship sequence. The sequence of spatial relationships is represented as a sequence of S_f, where S is the spatial relationship uid, and f is the frame number during which the spatial relationship keeps. Take Figure 1-(a) as an example, the spatial relationships which are represented by uid between the tree and the horse in the x-axis are "$tree1horse$" of frame 1,

Table 3. $Tuids$ **of 16 spatial operators**

$operator$	$<$	$<^*$	$\mid$	$\mid^*$	$/$	$/^*$	$]$	$[$	$\%$	$=$	$]^*$	$[^*$	$\%^*$	$+$	$+^*$	$-$
$Tuid$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Table 4. Rules of deriving temporal relationships

Part	Sequence	DTR	$Tuid$	ETR
a	$14, 16, 15$	$A < B$	1	No
a	$15, 16, 14$	$A <^* B$	2	No
a	$14, 15$	$A \mid B$	3	No
a	$15, 14$	$A \mid^* B$	4	No
a	$*$	$A = B$	10	No
b	$14, *$	$A]B$	7	$A = B$
b	$*, 14$	$A[B$	8	$A = B$
b	$15, *$	$A]^*B$	11	$A = B$
b	$*, 15$	$A[^*B$	12	$A = B$
c	$14, *, 15$	A/B	5	$A = B$, $A]B$, $A[^*B$
c	$15, *, 14$	$A/^*B$	6	$A = B$, $A]^*B$, $A[B$
c	$14, *, 14$	$A\%B$	9	$A = B$, $A]B$, $A[B$
c	$15, *, 15$	$A\%^*B$	13	$A = B$, $A]^*B$, $A[^*B$

DTR: Derived Temporal Relationship
ETR: Embedded Temporal Relationship

"$tree9horse$" of frame 2, "$tree14horse$" of frames 3–5, and "$tree1horse$" of frame 6. The corresponding spatial relationship sequence between the tree and the horse in the x-axis is represented as "$1_1, 9_1, 14_3, 1_1$."

Before deriving the temporal relationships, we have to preprocess the spatial relationship sequence to eliminate the redundant symbols in it. The preprocess part contains two steps: (1) $ignorance$, and (2) $accumulation$. First, we ignore the difference among those 13 spatial relationships, since each of these 13 spatial relationships implies the appearance of both objects. To achieve this goal, we replace each of the spatial relationships 1 to 13 with the symbol "$*$" Second, for the given sequence of spatial relationships "S_a, S_b, S_c," we will replace it with "S_{a+b+c}." For example, for the given sequence "$14_3, 14_7$," we replace it with "14_{10}," and for the given sequence "$*_3, *_2$," we replace it with "$*_5$." For example, for the spatial relationship sequence in the x-axis shown in Figure 1-(a), it is "$1_1, 9_1, 14_3, 1_1$." The corresponding preprocessing steps are like this: $(1_1, 9_1, 14_3, 1_1)$ $\rightarrow (*_1, *_1, 14_3, *_1) \rightarrow (*_2, 14_3, *_1)$.

Finally, after getting the resulting string, we can use an exact string matching algorithm to derive the 13 temporal relationships. The derived rules are shown in Table 4. The related explanatory charts for Table 4 are shown in Figures 5 and 6. In Table 4, basically, those sequences of uid's can be classified into three parts, a, b, and c. In part a, the derived temporal relationship does not contain any other temporal relationships. In part b, the derived temporal relationship implies temporal relationship "$A = B$." For example, in Figure 7-(a), the temporal relationship between objects A and B is equal to $A[B$. Moreover, there is one temporal relationships implied $A = B$. In part c, each of the derived temporal relationship implies the other three temporal relationships. For example, in Figure 7-(b), the temporal relationship between objects A and B is equal to $A\%B$. Moreover, there are three temporal relationships implied $A = B$, $A]B$, and $A[B$.

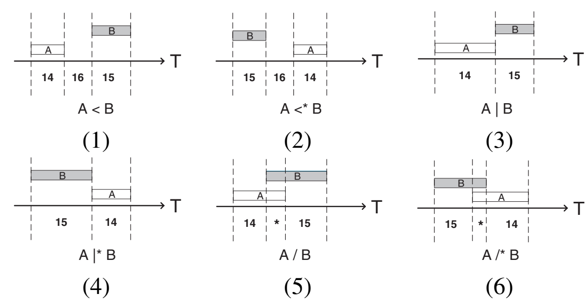


Figure 5. Derived temporal relationships (1–6)

3.2 The Temporal Unique-ID-Based Matrix (TUID Matrix)

Since there are two spatial relationship sequences (in x- and y-axes) between any pair of objects in one shot, we use a data structure similar to the UID matrix to store each spatial relationship sequence.

Suppose a video V contains m objects and let O = $\{o_1, o_2, ..., o_m\}$. An $m \times m$ *Temporal UID matrix* T of video V is defined as follows:

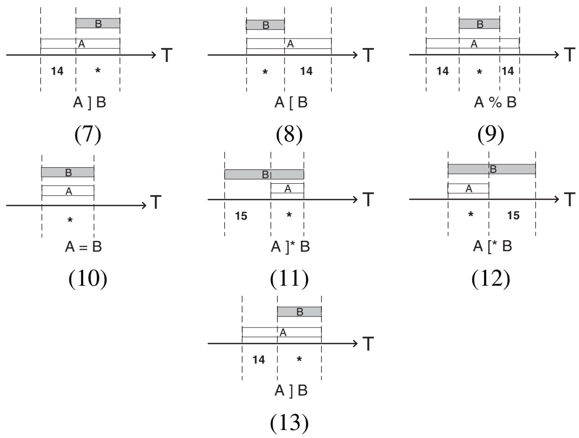


Figure 6. Derived temporal relationships (7–13)

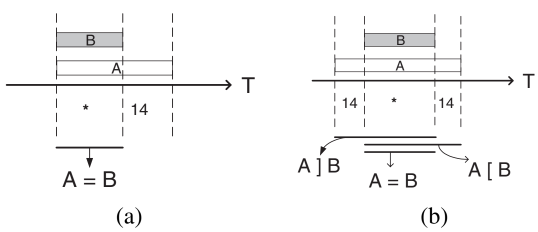


Figure 7. Embedded temporal relationships: (a) $(*, 14) \Rightarrow A[B \cup A = B$**; (b)** $(14, *, 14) \Rightarrow A\%B \cup (A = B, A]B, A[B)$**.**

$$
T = \begin{array}{c} \\ o_1 \\ o_2 \\ \vdots \\ o_{m-1} \\ o_m \end{array}
\begin{array}{c}
\begin{array}{ccccc} o_1 & o_2 & \cdots & o_{m-1} & o_m \end{array} \\
\left[\begin{array}{ccccc}
0 & s^y_{1,2} & \cdots & \cdots & s^y_{1,m} \\
st^x_{1,2} & 0 & \ddots & & \vdots \\
\vdots & \ddots & 0 & \ddots & \vdots \\
\vdots & & \ddots & 0 & s^y_{m-1,m} \\
st^x_{1,m} & \cdots & \cdots & st^x_{m-1,m} & 0
\end{array}\right]
\end{array}
$$

where the lower triangular matrix stores the spatial information along the x-axis, and the upper triangular matrix stores the spatial information along the y-axis. $st^x_{j,i}$ is the spatial sequence of numbers between objects o_i and o_j along the x-axis and $s^y_{i,j}$ is the spatial sequence between objects o_i and o_j along the y-axis. Assume that the number of changes of spatial relationships between o_i and o_j is n, the spatial sequence $st^x_{j,i}$ is $(uid1_{t1}, uid2_{t2}, \cdots, uidn_{tn})$, and the spatial sequence $s^y_{i,j}$ is $(uid1, uid2, \cdots, uidn)$, where $uid1_{t1}$ means that there is a spatial relationship $uid1$ between o_i and o_j during the time t_1. For example, the corresponding Temporal UID matrix T of Figure 1-(a) is shown as follows, where the tree and the horse are denoted by t and h, respectively.

$$
T = \begin{array}{c} \\ t \\ h \end{array}
\begin{array}{c}
\begin{array}{cc} t & h \end{array} \\
\left[\begin{array}{cc}
0 & 2, 2, 14, 6 \\
1_1, 9_1, 14_3, 1_1 & 0
\end{array}\right]
\end{array}
$$

The TUID-Matrix strategy can support three query types, query by spatial relationships, temporal relationships, and spatio-temporal relationships. For the query type of the spatial relationship, since we record all the spatial relationships between the objects, we can answer the spatial relationships directly. For example, in Figure 1-(a), a user would like to query the video that the spatial relationship in the x-axis between the tree and the horse is "$t1h$." We check each spatial relationship of the sequence in the x-axis between the tree and the horse. Then we find that frames 1 and 6 are of the user of interest.

For the query type of the temporal relationship, according to the way to derive temporal relationships from the sequence of spatial relationships, we can use an exact string matching algorithm to derive the 13 temporal relationships. For example, in Figure 1-(a), the user wants to query the video that the temporal relationship between the tree and the horse is "$t8h$." First, we apply the deriving rules shown in Table 4 to transform the query into a string "$*, 14$." Then, the index structure sequence $T[t, h]$ will be preprocessed into "$*_2, 14_3, *_1$." The string "$*, 14$" exactly matches the sequence "$*_2, 14_3, *_1$." So we can answer the temporal relationships.

For the query type of the spatio-temporal relationship, we take an example to illustrate the process. Figure 8 shows

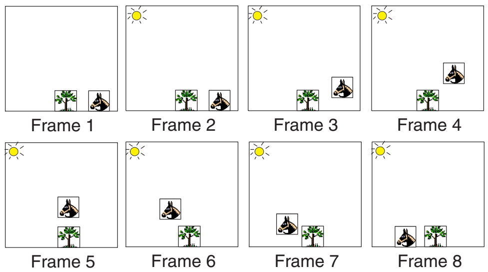


Figure 8. Video V_2

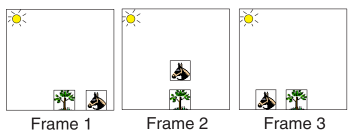


Figure 9. Query video V_3

video V_2 in which a horse is jumping over a tree, and the sun is in the sky.

The corresponding Temporal UID matrix T_{v2} is shown as follows:

$$T_{v2} = \begin{array}{c} \\ t \\ h \\ s \end{array} \begin{array}{c} \begin{array}{ccc} t & h & s \end{array} \\ \left[\begin{array}{ccc} 0 & 10,5,1,1,1,5,10 & 14,1 \\ 1_2,1_1,1_1,10_1,6_1,2_1,2_1 & 0 & 14,1 \\ 14_1,2_7 & 14_1,2_7 & 0 \end{array} \right] \end{array}$$

where t denotes object "tree," h denotes object "horse," and s denotes object "sun." There is a query video V_3 as shown in Figure 9. The related TUID matrix of V_3 is shown follows:

$$T_{v3} = \begin{array}{c} \\ t \\ h \\ s \end{array} \begin{array}{c} \begin{array}{ccc} t & h & s \end{array} \\ \left[\begin{array}{ccc} 0 & 10,1,10 & 1 \\ 1_1,10_1,2_1 & 0 & 1 \\ 2_2 & 2_2 & 0 \end{array} \right] \end{array}$$

By comparing these two TUID matrices, those objects shown in the T_{v3} are also in T_{v2}. Then, we find out the intervals of frames for each pair of objects that can match the change of spatial relationships as shown in T_{v3}. To find out the interval of frames for the pair of the objects t and h, we have to find out the subsequences of the spatial relationships in the x- and y-axes in T_{v2} based on the sequences of spatial relationships in the x- and y-axes in T_{v3}. That is, along x-axis, there are six subsequences of spatial relationships in T_{v2} that match the sequence, (1,10,2), in T_{v3}. Those subsequences are shown in the left-hand column of Table 5. The order of each spatial relationship in the subsequence forms an order sequence shown in the right-hand column of Table 5. Similarly, there are three subsequences

Table 5. Subsequences and orders in the x-axis

subsequence	order
$(1_2, 10_1, 2_1)$	$(1,4,6)$
$(1_2, 10_1, 2_1)$	$(1,4,7)$
$(1_1, 10_1, 2_1)$	$(2,4,6)$
$(1_1, 10_1, 2_1)$	$(2,4,7)$
$(1_1, 10_1, 2_1)$	$(3,4,6)$
$(1_1, 10_1, 2_1)$	$(3,4,7)$

Table 6. Subsequences and orders in the y-axis

subsequence	order
$(10,1,10)$	$(1,3,7)$
$(10,1,10)$	$(1,4,7)$
$(10,1,10)$	$(1,5,7)$

of spatial relationships in the y-axis in T_{v2} that match the sequence, *i.e.,* (10,1,10), in T_{v3}. Table 6 shows these subsequences and the corresponding order sequences.

Note that, there exists one order sequence, that is (1,4,7), both shown in Tables 5 and 6. This means that there exist subsequences both in the x- and y-axes according to the same order that match the corresponding sequence in the x- and y-axes in T_{v3}. Based on the subsequence, $(1_2, 10_1, 2_1)$, in the x-axis, we can obtain two intervals of frames for the pair of objects t and h that match the frame sequence as shown in T_{v3}. The intervals are calculated as follows. The begin bound of the frame interval is calculated by the first element in the subsequence. That is 1_2. According to the position of 1_2 in the sequence of spatial relationships, since 1_2 means that two continuous frames have the same spatial relationship, we have frames 1 and 2 to be the begin bound of the interval of frames. Similarly, the end bound of the interval of frames is calculated by the last element in the subsequence, *i.e.,* 2_1. According to the position of 2_1 in the sequence of spatial relationships, we have frame 8 $(= 2+1+1+1+1+1+1)$ to be the end bound of the interval of frames. Thus, the interval of frame 1 to frame 8 and the interval of frame 2 to frame 8 match the query video V_3 for the pair of objects h and t. We denote these two intervals by $[1,8]$ and $[2,8]$, respectively. Table 7 summaries the matched intervals of frames for each pair of objects. Note that, the interval $[2,8]$ appears in each pair of the objects. Finally, we conclude that those frames among frame 2 to frame 8 in V_2 match the query video V_3.

Table 7. Matched intervals of frames for each pair of objects

pair	interval
(t, h)	$[1, 8]$, $[2, 8]$
(t, s)	$[2, 2], [2, 3], \cdots, [2, 8], [3, 3], [3, 4], \cdots, [7, 8], [8, 8]$
(h, s)	$[2, 2], [2, 3], \cdots, [2, 8], [3, 3], [3, 4], \cdots, [7, 8], [8, 8]$

Table 8. Range of the four factors

parameter	videos	objects	frames	query objects
range	$100 \sim 400$	$20 \sim 35$	$1000 \sim 2500$	$4 \sim 10$

4 Performance Study

In this Section, we present the performance study of our proposed strategy ($TUID$) and compare it with Lee *et al.*'s 3D C-string strategy ($3DCS$) [10] by simulation.

4.1 The Simulation Model

In experiments, a preprocessing phase is needed to generate and the indexes of all videos in the database and query videos. But, in this paper, we concern about showing the efficiency of our proposed strategy, so we do not consider the index generation as the cost of the execution time. We perform 100 queries with the target videos generated as follows: (1) select one randomly from the videos in the database; (2) extract a sub-video from the selected video, (3) take a random value from an appropriate range for each window of a frame. The cost of the execution time of the experiment is measured by the average elapsed time of 100 video queries.

In our simulation, the experiments are made according to the synthetic video database. Basically, the number of objects which could appear in the video database is 50. There are four factors dominating the performance of the video retrieval algorithm: the number of objects in a query video, the number of videos in the database, the number of objects and frames in a video [10]. Table 8 shows the range of the four factors in the synthetic video database. Since we can specify the number of objects in query videos, it is set in the range between 4 and 10, and each video contains about 500 frames. There are 100 to 400 videos in the database. For each video, we assign 20 to 35 objects to it. The number of frames in a video is in the range between 1000 and 2500. In each experiment, we change one cost factor and set the other three factors in default values as shown in Table 9.

Table 9. Default values of each of four factors

parameter	videos	objects	frames	query objects
default value	200	30	1000	8

4.2 Simulation Results

We focus on the performance of answering queries of spatial relationships, temporal relationships, and spatio-temporal relationships. In Tables 10, 11, we show that our strategy needs much less query time than Lee *et al.*'s strategy.

Table 10 shows the simulation results based on changing the number of frames in a video. The execution time increases linearly as the number of frames in a video increases for the query types of spatial, temporal, and spatio-temporal. The reason is that the more frames in a video is, the longer the execution time is required.

In Table 11, the execution time increases linearly as the number of objects in a query video increases for both strategies in the query type of spatio-temporal. The reason is that the more the number of objects in a query video is, the more the number of combinations of objects in each video is needed to be checked. However, in the query types of spatial and temporal, we randomly choose two objects from the objects of a query video. And we calculate the spatial or temporal relationships between the two objects to do the query. Moreover, in the query type of objects, we focus on the motions of one object. Therefore, the cost factor, the number of objects in query video, has no effect on the execution time for the query types of spatial, temporal, and objects.

5 Conclusion

In this paper, we have proposed an indexing strategy, the TUID matrix strategy, for video retrieval based on the original 13 and three new spatial relationships. According to those 16 spatial relationships, we can derive the temporal relationships from the sequence of spatial relationships. In this way, we do not have to record temporal relationships by extra storage space. The TUID matrix strategy could efficiently query spatial, temporal, and spatio-temporal relationships. From our simulation results, we have shown that our proposed strategy needs less searching time than the 3D C-string strategy.

References

[1] Z. Aghbari, K. Kaneko, and A. Makinouchi. Content-Trajectory Approach for Searching Video Databases. *IEEE*

Table 10. Comparison of query-time (*ms*) in different number of frames

	1000	1500	2000	2500
TUID	5.7	5.9	6.2	6.3
3DCS	585.4	593.6	611.9	623.1

(a) spatial query

	1000	1500	2000	2500
TUID	6.1	6.3	6.6	6.8
3DCS	379.8	391.0	393.2	406.9

(b) temporal query

	1000	1500	2000	2500
TUID	69.0	72.3	78.6	81.3
3DCS	2450.1	2848.9	3282.5	3440.7

(c) spatio-temporal query

Table 11. Comparison of query-time (*ms*) in different number of objects in a query video

	4	6	8	10
TUID	4.9	5.2	5.3	5.8
3DCS	610.1	621.9	632.9	612.3

(a) spatial query

	4	6	8	10
TUID	5.7	5.8	5.8	6.6
3DCS	360.9	379.4	386.2	378.9

(b) temporal query

	4	6	8	10
TUID	52.2	71.3	82.8	94.5
3DCS	1731.1	2333.1	2496.5	2325.2

(c) spatio-temporal query

Trans. on Multimedia, 5(4):516–531, Dec. 2003.

[2] Y. K. Chan and C. C. Chang. Spatial Similarity Retrieval in Video Databases. *Journal of Visual Communication and Image Representation*, 12(2):107–122, June 2001.

[3] Y. I. Chang, H. Y. Ann, and W. H. Yeh. A Unique-Id-Based Matrix Strategy for Efficient Iconic Indexing of Symbolic pictures. *Pattern Recognition*, 33(8):1263–1276, Aug. 2000.

[4] C. Y. Chiu, S. P. Chao, M. Y. Wu, S. N. Yang, and H. C. Lin. Content-Based Retrieval for Human Motion Data. *Journal of Visual Communication and Image Representation*, 15(3):446–466, Sept. 2004.

[5] M. E. Donderler, E. Saykol, U. Arslan, O. Ulusoy, and U. Gudukbay. Bilvideo: Design and Implementation of a Video Database Management System. *Multimedia Tools and Application*, 27(1):79–104, Sept. 2005.

[6] F. J. Hsu, S. Y. Lee, and B. S. Lin. 2D C-Tree Spatial Representation for Iconic Images. *Journal of Visual Languages and Computing*, 10(2):147–164, April 1999.

[7] V. Kobla, D. DeMenthon, and D. Doermann. Identifying Sports Video Using Replaying, Text, and Camera Motion Features. *Proc. of the SPIE Conf. on Storage and Retrieval for Media Databases*, pages 332–342, 2000.

[8] O. Kucuktunc, U. Gudukbay, and O. Ulusoy. A Natural Language-Based Interface for Querying a Video Database. *IEEE Multimedia*, 14(1):83–89, Jan.–March 2007.

[9] T. C. T. Kuo and A. L. P. Chen. Content-Based Query Processing for Video Databases. *IEEE Tran. on Multimedia*, 2(1):1–13, 2000.

[10] A. J. T. Lee, H. P. Chiu, and P. Yu. 3D C-String: A New Spatio-Temporal Knowledge Representation for Video Database Systems. *Pattern Recognition*, 35(11):2521–2537, 2002.

[11] A. J. T. Lee, P. Yu, H. P. Chui, and H. H. Lin. Video Algebra for Spatio-Temporal Reasoning of Iconic Videos Represented in 3D C-string. *Journal of Information Science and Engineering*, 23(1):1–19, Jan. 2007.

[12] Z. Li, A. K. Katsaggelos, and B. Gandhi. Fast Video Shot Retrieval Based on Trace Geometry Matching. *IEE Proc.-Vision Image and Signal Processing*, 152(3):367–373, May 2005.

[13] C. H. Lin and A. L. P. Chen. Motion Event Derivation and Query Language for Video Databases. *Storage and Retrieval for Media Databases 2001, Proc. of SPIE*, pages 208–218, 2001.

[14] C. C. Liu and A. L. P. Chen. 3D-List: A Data Structure for Efficient Video Query Processing. *IEEE Trans. on Knowledge and Data Engineering*, 14(1):106–122, Feb. 2002.