



Available at

www.ElsevierComputerScience.com

POWERED BY SCIENCE @ DIRECT®

Computer Standards & Interfaces 26 (2004) 317–328

COMPUTER STANDARDS
& INTERFACES

www.elsevier.com/locate/csi

Video retrieval using successive modular operations on temporal similarity

Chi-Chun Lo^a, Shuenn-Jyi Wang^{b,*}, Lee-Wen Huang^a

^a*Institute of Information Management, National Chiao-Tung University, 1001 Tah Hsueh Road, Hsinchu 300, Taiwan*

^b*Department of Computer Science, Chung Cheng Institute of Technology, National Defense University, 190 Sanyuan 1st St., Dashi Jen, Taoyuan 335, Taiwan*

Received 9 November 2003; received in revised form 9 November 2003; accepted 10 November 2003

Abstract

A video retrieval system consists of two major subsystems for indexing and querying, respectively. In this paper, we propose a framework for retrieving video sequences using modular operations on temporal similarity. In the indexing process, the contents (objects) of a video sequence are used to build temporal triples representing temporal relationships between objects. For every temporal triple generated from different video sequences, a prime number is assigned to it by the proposed prime number assignment (PNA) algorithm. For each video sequence, a standing value is generated via its set of prime numbers. In the query process, successive modular operations (SMO) on the standing values are used to retrieve video sequences. Based on the PNA and the SMO, we built an experimental video retrieval system. From the simulation results, we notice that a query can be done in a linear time.

© 2003 Elsevier B.V. All rights reserved.

Keywords: Video retrieval; Video indexing; Temporal relationship; Similarity retrieval; Modular operation

1. Introduction

With the advances in computer technologies, such as the increasing speed of CPU, the capacity of the storage device and various compression methods, digital video is becoming more and more common in almost every aspect of our life, including education, entertainment, communications, etc. For the ever-increasing amount of video data, a systematic approach of retrieving video data is needed. A video retrieval system consists of two major subsystems for indexing

and querying, respectively. In the indexing process, video segmentation [1–3] is used to segment video sequence into shots where each shot represents a sequence of frames having the same contents. Once shots are identified, key frames are extracted from each shot for indexing [4,5]. By using the indices, the query process provides a means of retrieving video data.

Since the contents of video data are getting richer and richer, the traditional ways of indexing and querying, e.g., query by keywords or by their combinations, are no longer satisfactory. Therefore, the content-based video retrieval [6] has emerged as an important and challenging research area for recent years. In this approach, the contents (objects) of video data are used to construct indices. Both spatial and temporal relation-

* Corresponding author. Tel.: +886-33805249218; fax: +886-33894770.

E-mail address: sjwang@ccit.edu.tw (S.-J. Wang).

ships between objects are of a particular interest to researchers.

To query the image database, Chang et al. [7] used the 2D-string data structure to decompose an image P into a set of 2D-string rules that can preserve the spatial relationship among objects of the original image. Later, Chang and Lee [8] combined the nine Direction Low Triangular (9DLT) matrix and the hashing techniques to determine similarity matching. The 9DLT matrix is used to describe the spatial relationships among objects. A pair of objects, along with their spatial relationship, forms a triple. An image can be represented by its set of triples. Chang [9] developed a fast superimposed coding technique based upon the 9DLT matrix. In Chang and Liang [10], they proposed the modular operation, coupled with a perfect hashing function, to process the triples. The perfect hashing function provides a one-to-one mapping between the key space and the address space.














Allen's work [11] on temporal intervals laid the foundation for many researches concerned with time intervals [12–15]. In Hjelmsvold et al. [12], temporal interval-based models have been presented for time dependent multimedia such as video. Pradhan et al. [14] have mainly focused on the operations for finding new intervals but not on the relationships between given intervals. Several video database systems, such

as the OVID [16], the VideoSTAR [17], and the VIQS [18] concerned with the temporal intervals. The problem of these systems is that manually annotating temporal intervals is time-consuming and may cause inconsistent perceptions by different users.

In this paper, based on Chang's work [9,10] on similarity retrieval on the image database and Allen's work [11] on temporal intervals, we propose a framework for retrieving video sequences using modular operations on temporal similarity. In the indexing process, the contents (objects) of a video sequence are used to build the temporal triples representing temporal relationships between objects. For every temporal triples generated from different video sequences, a prime number is assigned to it by the proposed prime number assignment (PNA) algorithm. For each video sequence, a standing value is generated via its set of prime numbers. In the query process, successive modular operations (SMO) on the standing values are used to retrieve video sequences. Based on the PNA and the SMO, we built an experimental video retrieval system. From the simulation results, we notice that a query operation can be done in a linear time.

In Section 2, the definitions of the temporal relationship and the temporal similarity are presented. Section 3 discusses both the indexing process and the query process. Complexity analysis on the proposed

Table 1
Refined temporal relationships

| Relationship | Name | Illustration using Allen's definition | | |
|--------------|---------------|--|---|--|
| 1 | before | X before Y  | X meet Y  | |
| 2 | after | X after Y  | X meet-by Y  | |
| 3 | overlaps | X overlaps Y  | | |
| 4 | overlapped by | X overlapped by Y  | | |
| 5 | contains | X contains Y  | X started-by Y  | X finish-by Y  |
| 6 | during | X during Y  | X start-with Y  | X finish-with Y  |
| 7 | equal | X equal Y  | | |

mechanism is given. In section 4, we present computational results and analyses. Section 5 concludes this paper with possible future research directions.

2. Temporal relationship and similarity

2.1. Temporal relationship

Allen [11] defined 13 relationships to describe the temporal relationship between objects. In this paper, we refine the 13 relationships into seven, as shown in Table 1. This refinement is aimed to reduce user's confusion while querying the video database. Take the two relationships, before and meet, as an example. To a user doing a query, it is easier to remember that event X happened before event Y than to recall that events X and Y happened at same time. Therefore, we combine these two relationships into one single relationship, before. For the same reason, we also reduce after and meet-by into the after relationship; during, start-with and finish-with into the during relationship. As to the three relationships, contains, started-by and finished-by, they are combined into one relationship, contains. Note that, in this refinement, before and after are inverse to each other; so do overlaps and overlapped by, and contains and during.

2.2. Temporal similarity

Two types of similarity, similarity matching and similarity retrieval, are concerned. Chang et al. [7] defined three types of similarity matching for image data. Following their discussions, we also define three types of similar matching in terms of the temporal relationship, for video data. Video V' is claimed to be a Type-2 subvideo of video V if all of the objects of V' can be found in V ; furthermore, the temporal relationship and the time interval that all objects occur are preserved. Video V' is claimed to be a Type-1 subvideo of video V if all of the objects of V' can be found in V and the temporal relationship is preserved. However, the time interval that all objects occur may be different. Video V' is claimed to be a Type-0 subvideo of video V if, in addition to the conditions set for Type-1 subvideo, the length of the time interval may be different. In this paper, we only consider Type-0 subvideos; i.e., Type-2, 1 and 0 subvideos are all treated as the same.

As to similarity retrieval, it is determined in terms of containment; for example, the percentage of the objects in a video sequence contained in the retrieved subvideo.

3. Similarity retrieval using the temporal relationship

Similarity retrieval using the temporal relationship is accomplished by two processes, the indexing process and the query process.

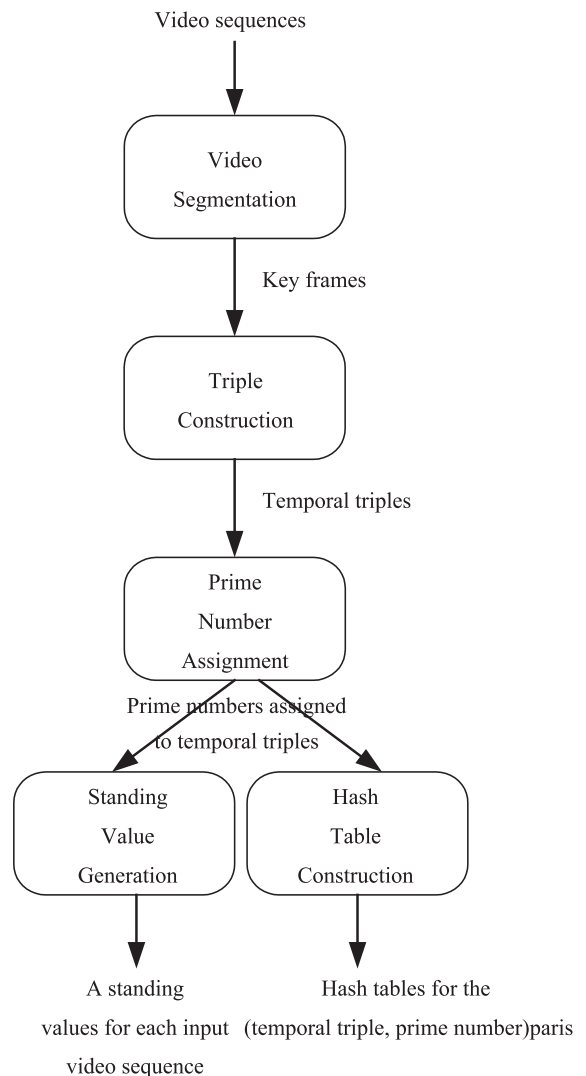


Fig. 1. The indexing process.

3.1. The indexing process

The indexing process, as shown in Fig. 1, has four steps. A step-by-step discussion of the indexing process is given as follow.

3.1.1. Video segmentation

For a given video sequence, shot changes are identified and then key frames are selected. Segmentation algorithms, as proposed in [1,3] can be used in this step.

3.1.2. Triple construction

For a given video sequence, the temporal triples are constructed according to the temporal relationships between objects appearing in the selected key frame. In most related researches [12–15], the temporal triples are manually generated. Here, we proposed a semi-automatic approach to construct the triple. In the manual phase, an interactive annotation interface (IAI) is designed for users to describe which objects appeared in which frames. Lots of graphic user interface design methods for image and video [21–23] can also be used to compose the IAI. Fig. 2 illustrates four annotated video sequences using IAI.

In the automatic phase, a triple construction (TC) algorithm is used to generate the temporal triples (O_i, O_j, r_{ij}) , where O_i and O_j are two objects, and r_{ij} represents the temporal relation between O_i and O_j ; r_{ij} is one of the seven relationships defined in Table 1.

The flowchart of the TC algorithm is depicted in Fig. 3. For an annotated video sequence, each object is represented by a bit string whose length is equal to the number of key frames selected. If a key frame contains the object, the corresponding bit is set to 1; otherwise 0. All bit strings are then sorted in the descending order. To construct a temporal triple, a bitwise and (&) operation between bit string B_i of object O_i and bit string B_j of object O_j is conducted. The bitwise ‘&’ operations are performed pair by pair by the descending order. By doing the sorting and the pair-wise bit operation, we can deduce the following assertions:

- (1) If $(B_i \text{ and } B_j) = B_i = B_j$, then O_i is equal to O_j ; the triple is represented as $(O_i, O_j, 7)$.
- (2) If $(B_i \text{ and } B_j) = 0$, then O_i is before O_j ; the triple is represented as $(O_i, O_j, 1)$.
- (3) If $(B_i \text{ and } B_j) = B_i \neq B_j$, then O_i contains O_j ; the triple is represented as $(O_i, O_j, 5)$.

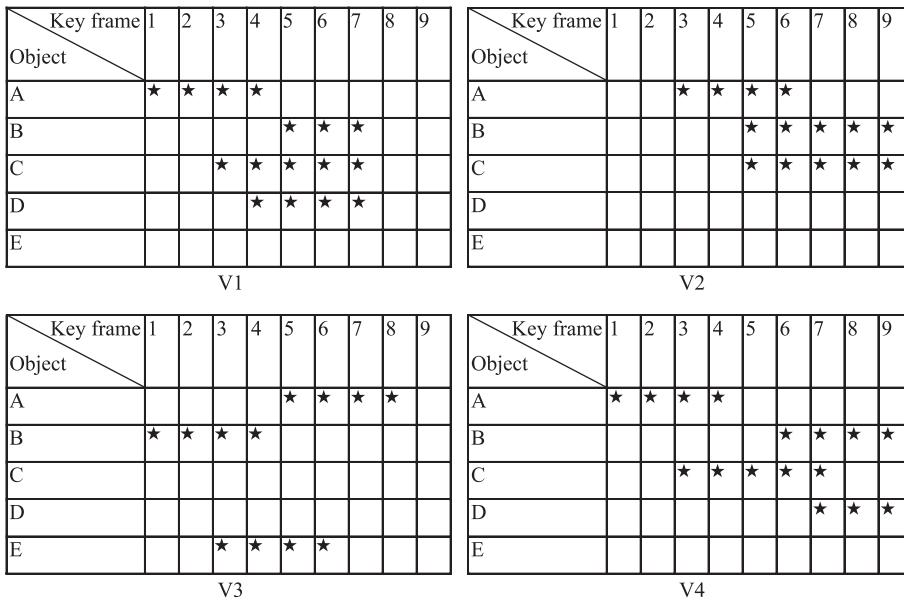


Fig. 2. Annotated video sequences.

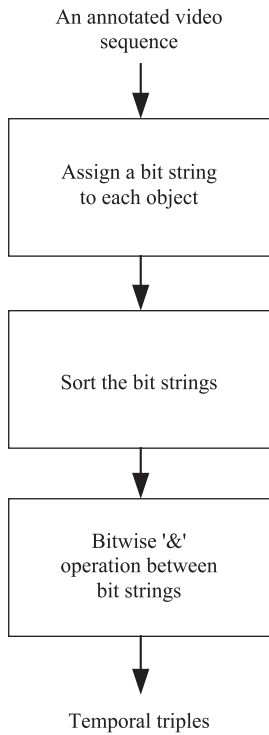


Fig. 3. The flowchart of the triple construction algorithm.

(4) Otherwise, O_i overlaps O_j ; the triple is represented as $(O_i, O_j, 3)$.

For the four annotated video sequences given in Fig. 2, Fig. 4 shows the corresponding temporal triples generated by using the TC algorithm.

3.1.3. Prime number assignment

For every temporal triple generated from the input video sequences, a prime number is assigned. To do this, we design a PNA algorithm. The basic design principle is to use as few prime numbers as possible. Some guidelines are as follows:

- (1) The larger the number of the temporal triples of a video sequence, the smaller the prime numbers assigned to the triples of the video sequence. Therefore, the assignment should begin with the video sequence that has the largest number of triples.
- (2) To a triple of a video sequence, the larger the frequency of occurrence, the smaller the prime number assigned.

- (3) Those triples which occur only once in the same video sequence are assigned the same prime number, since they can be regarded as one triple.
- (4) Same triples generated from different video sequences are assigned the same prime number.

The flowchart of the PNA algorithm is depicted in Fig. 5. The PNA algorithm begins with the adjustment of the temporal triples. For a given triple (O_i, O_j, r_{ij}) , if O_j is smaller than O_i in the alphabetical order, the triple is readjusted as (O_j, O_i, r_{ji}) , where r_{ji} is the inverse relationship of r_{ij} .

Second, for each video sequence V_i , E_i and G_i are computed, where E_i represents the number of triples whose frequency of occurrence is one and G_i the number of triples whose frequency of occurrence is more than one.

Third, video sequences are grouped; video sequences having the same number of triples are clustered into the same group. For example, V_1 and V_2 of Fig. 4 are in one group (group 1); V_2 and V_3 in the other group (group 2).

Forth, beginning with the group whose video sequences have the largest number of triples, the PNA algorithm assigns prime numbers to the triples whose frequency of occurrence is greater than one, followed by the triples whose frequency of occurrence is equal to one.

In order to determine which triple having more than one occurrence is selected first, a weight function for triple (O_i, O_j, r_{ij}) is defined as follows:

$$W(O_i, O_j, r_{ij}) = \text{freq}(O_i, O_j, r_{ij}) + \sum [G_k(O_i, O_j, r_{ij}) + 1]$$

where $\text{freq}(O_i, O_j, r_{ij})$ represents the number of occurrences of triple (O_i, O_j, r_{ij}) in all input video sequences, G_k the G value of video sequence V_k to

| V1 | V2 | V3 | V4 |
|---------|---------|---------|---------|
| (A,C,3) | (A,B,3) | (B,E,3) | (A,C,3) |
| (A,D,3) | (A,C,3) | (B,A,1) | (A,B,1) |
| (A,B,1) | (B,C,7) | (E,A,3) | (A,D,1) |
| (C,D,5) | | | (C,B,3) |
| (C,B,5) | | | (C,D,3) |
| (D,B,5) | | | (B,D,5) |

Fig. 4. Temporal triples of annotated video sequences of Fig. 2.

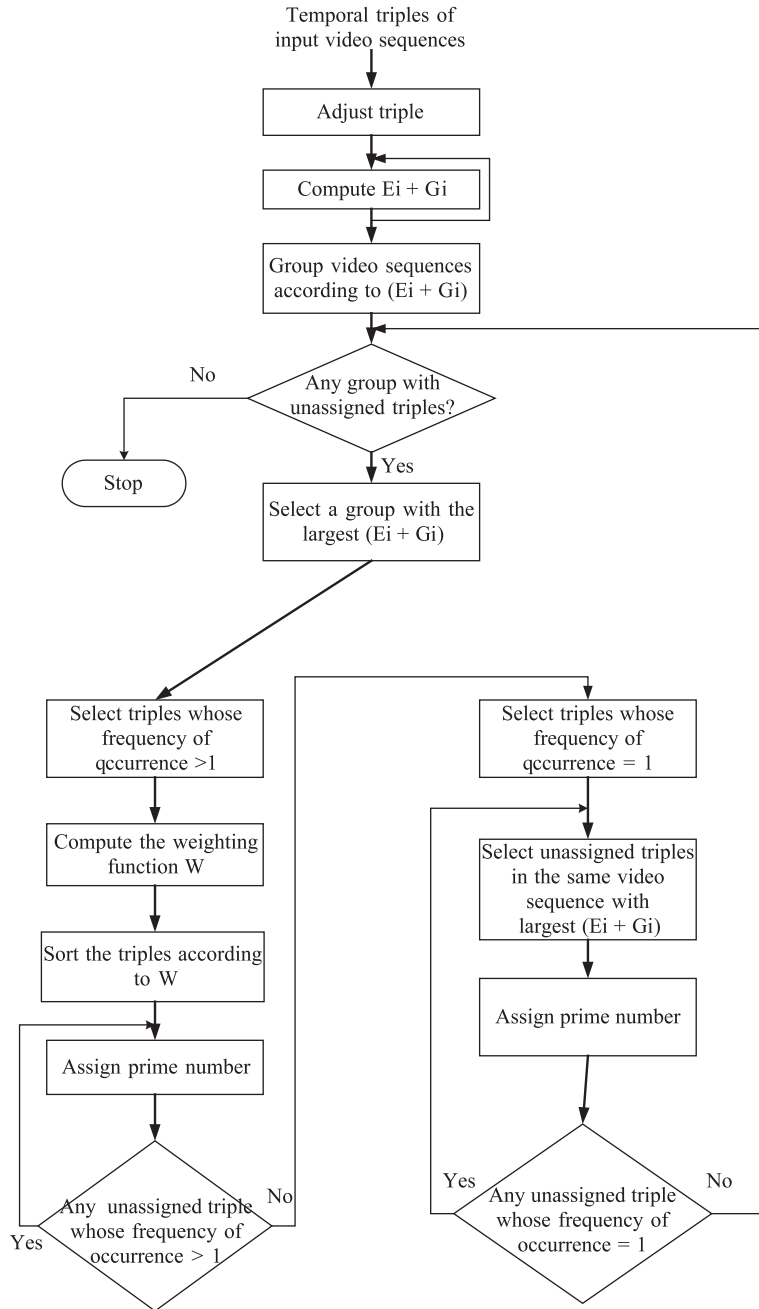


Fig. 5. The flowchart of the prime number assignment algorithm.

which triple (O_i, O_j, r_{ij}) belongs. The triple with the largest W value is selected first.

As to those triples occurring only once in the same video sequence, the same prime number is assigned.

Taking Fig. 4 as an example, group 1 is chosen as the first group and then V_1 is selected since it has the largest G value. Then, $W(A,C,3) = 3 + [(2 + 1) + (1 + 1) + (2 + 1)] = 11$ and $W(A,B,1) = 2 + [(2 + 1) + (2 + 1)] = 8$ are

| V1 | V2 | V3 | V4 |
|-----------|-------------|------------|-----------|
| (A,B,1):3 | (A,B,3): 11 | (A,B,2):13 | (A,B,1):3 |
| (A,C,3):2 | (A,C,3): 2 | (A,E,4):13 | (A,C,3):2 |
| (A,D,3):5 | (B,C,7): 11 | (B,E,3):13 | (A,D,1):7 |
| (B,C,6):5 | | | (B,C,4):7 |
| (B,D,6):5 | | | (B,D,5):7 |
| (C,D,5):5 | | | (C,D,3):7 |

Fig. 6. Prime numbers assigned to video sequences of Fig. 4.

computed. Since $W(A,C,3) > W(A,B,1)$, prime numbers 2 and 3 are assigned to $(A,C,3)$ and $(A,B,1)$, respectively. Fig. 6 presents the prime number assignment with respect to video sequences V_1, V_2, V_3 and V_4 of Fig. 4.

3.1.4. Hash table construction

In order to accelerate the speed of a query, a fast mapping mechanism between a temporal triple and its assigned prime number is needed. Here, we propose a two-stage mapping algorithm. In the first stage, the relationship between two objects is used as an index into the corresponding hashing table, i.e., one hash table per relationship. In the second stage, the perfect hashing algorithm proposed by Chang et al. [19,20] is used. This algorithm maps a set of n distinct pairs (O_i, O_j) 's to a set of m consecutive address = $v_1(O_i) + v_2(O_j)$, where O_i and O_j are two objects, and v_1 and v_2 are two value assignment functions. It has been shown that the search time of this algorithm is linear.

3.1.5. Standing values generation

Each video sequence is identified by a standing value (SV), which is equal to the product of the prime numbers assigned to the triples of the video sequence. Fig. 7 shows the standing values of V_1, V_2, V_3 and V_4 of Fig. 6.

However, because of the limitation on the computer register, the overflow of SV is an outstanding problem. To solve this problem, we use a linked list of integers instead of a single integer to represent the SV. The SV is expressed as follows:

$$SV = \prod_j sv_{ij}$$

where sv_{ij} represents the j th sub-standing value of video sequence i . The SV is calculated in the follow-

ing way: first, all prime numbers of video sequence i are sorted in the ascending order; then, starting from the lowest prime number, the sub-standing values are computed. A new sub-standing value is created only if overflow occurs.

3.2. The query process

Query is based on the standing values computed in the indexing process. As shown in Fig. 8, the query process consists of four steps: interactive annotation [21–23], triple translation, hash table look-up and successive modular operations. The algorithms used in triple translation and hash table look-up are the same as those outlined in Section 3.1. Therefore, we only elaborate interactive annotation and successive modular operations.

3.2.1. Interactive annotation

As to “query by example”, the annotation procedure is the same as the one detailed in Section 3.1. To a user using “query by combinations”, he/she knows the queried objects and the relationships between the objects. He/she only marks their occurrence in the key frames.

3.2.2. Successive modular operations

Successive modular operations provide the basis of similarity retrieval. In this paper, similarity retrieval is determined by the level of similarity (LoS) defined as follows:

$$\text{Level of Similarity} = \frac{\text{the number of temporal triples contained in the retrieved subvideo}}{\text{the total number of temporal triples contained in the queried video}}$$

Let $SV_i (= \prod_j sv_{ij})$ represent the standing value of video (or subvideos) sequence i being queried; $SV_q (= \prod_j sv_{qj})$ the standing value generated by query q .

| V1 | V2 | V3 | V4 |
|-----------|-------------|------------|-----------|
| (A,B,1):3 | (A,B,3): 11 | (A,B,2):13 | (A,B,1):3 |
| (A,C,3):2 | (A,C,3): 2 | (A,E,4):13 | (A,C,3):2 |
| (A,D,3):5 | (B,C,7): 11 | (B,E,3):13 | (A,D,1):7 |
| (B,C,6):5 | | | (B,C,4):7 |
| (B,D,6):5 | | | (B,D,5):7 |
| (C,D,5):5 | | | (C,D,3):7 |

$$SV: 2*3*5 = 30 \quad 2*11 = 22 \quad 13 \quad 2*3*7 = 42$$

Fig. 7. Standing values of V1, V2, V3 and V4 of Fig. 6.

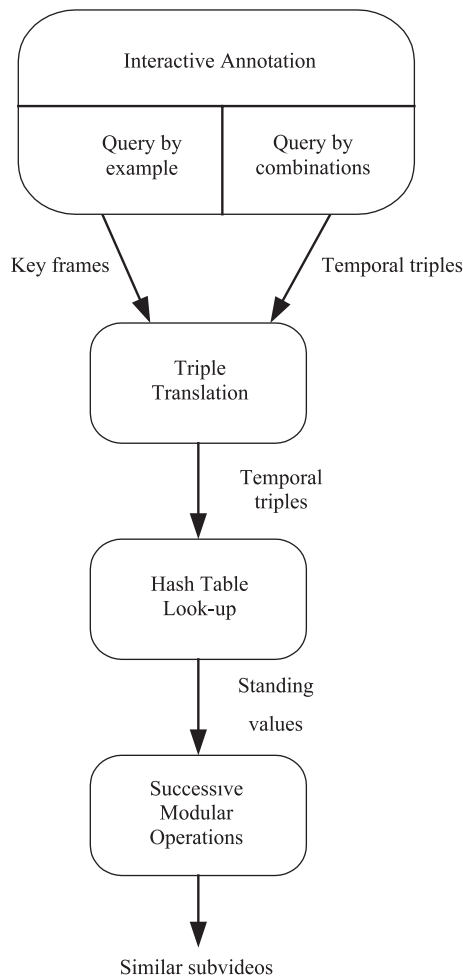


Fig. 8. The query process.

N successive modular operations between sv_{ij} and sv_{qi} , $j=1, 2, \dots, n$ are performed. If $sv_{ij} \bmod sv_{qi} = 0$ for $\forall j$, then V_q is a subvideo of V_i . The number of subvideos retrieved is determined by the LoS.

One noticeable problem of using the SMO is that a query might fail improperly. For example, for a particular j , the prime numbers contained in SV_{qj} might be scattered in two or more consecutive SV_{ij}' where $j=j, j+1, j+2, \dots$. However, since the PNA is designed to use as few prime numbers as possible and the sorting of the set of prime numbers of video sequence i is done before computing the SV_i , this phenomenon seldom occurs. Even if it did occur, we can separate the largest prime number (only if this prime number is reasonably large compared to the size

of the CPU register in use) from SV_{qi} and put it into SV_{qj+1} .

3.3. Complexity analysis

We assume that there are n video sequences in the database; for video sequence i ($i=1, 2, \dots, n$), it has m_i objects and the number of query items of query q is k . The maximum number of temporal triples of these n video sequences is P , which is equal to $\sum_{i=1}^n \frac{(m_i-1) \times m_i}{2}$. To find the prime number assigned to a temporal triple of query q by using the hash tables, the search time is close to a function of k with order $O(k)$. To compute the standing value of query q , k multiplications are needed. To perform similarity retrieval, successive modular operations takes time with order $o(a*n)$, where a represents the average number of sub-standing values. Therefore, the query time is close to a function of k , a and n with order $O(k+k+an)$. Since, in most situations, both k and a are much smaller than n , we claim that the query time is close to a function of n with order $O(n)$.

3.4. Discussions

The advantage of using prime number and the modular operation is the rapid response of a query (a query is done in a linear time).

One limitation of the proposed mechanism is that all prime numbers have to be recomputed every time a new video sequence is added to the database. However, this assignment can be done off-line.

Also, in order to further reduce the possibility of the overflow problem, it is recommended that the number of objects in the video database should be as small as possible. This implies that the proposed mechanism is more effective while applying it to a subcategory of a particular knowledge domain; for instance, the insets of the family of animals, the heart surgery of medical operations, etc.

4. Computational results and analyses

Based on the mechanism presented in Section 3, we have built an experimental video query system. An IBM PC with the Intel Pentium III processor and 256 MB RAM is used to develop the experimental system.



Fig. 9. The video sequences and key frames of Test I.

A graphic user interface is designed by using the MATLAB language. In the experiments, the following assumptions are assumed:

- All objects are continuous (an object is continuous if its occurrence can be found only in consecutive key frames). Therefore, if there are discrete objects in a video sequence, the video will be split into subvideos.
- All object names are predefined by the system.
- The standing value of a stored video sequence is greater than or equal to that of a query sequence.

4.1. Test cases

Two test cases, I and II, are designed to examine the accuracy and the efficiency of the experimental system, respectively.

Test I, as shown in Fig. 9, contains 10 video sequences extracted from the movie, “Harry Potter”. Twelve objects, “Harry Potter”, “Albus Dumbledore”, “Rubeus Hagrid”, “Professor Severus Black”, “Professor Quinell”, “Ron Weasley”, “Hermione

Granger”, “Voldemort”, “Minerva McGonagall”, “Nimbus 2000”, “Sorting Hat” and “Sorcerer’s Stone” are selected.

Test II includes seven randomly generated video databases, the seven databases contain 25, 50, 100, 200, 400, 800 and 1000 video sequences, respectively. The number of temporal triples of the seven database is 839, 1665, 3321, 6659, 13,326, 26,584 and 33,267, respectively. The number of objects in a video sequence is 9. Also, each query sequence has four temporal triples.

4.2. Results and analyses

4.2.1. Test I

Fig. 10 shows the results using “query by example”. In the example, we can find the temporal triple (Ron Weasley, Harry Potter, during). Fig. 11 presents the results using “query by combinations”. Two query items, Harry Potter contains Ron Weasley and Ron Weasley contains Hermione Granger, are specified.

By examining both results, we find that correct subvideos are extracted.



Fig. 10. Results using “query by example”.

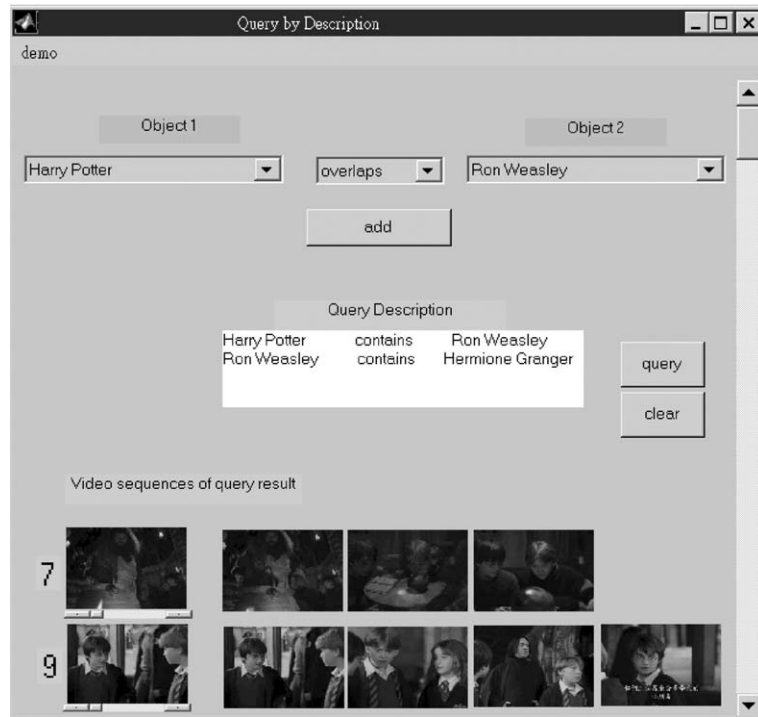


Fig. 11. Results using “query by combinations”.

4.2.2. Test II

Fig. 12 depicts the average response time for every 10 queries. The response time is linear and within few seconds, even for the largest video sequence.

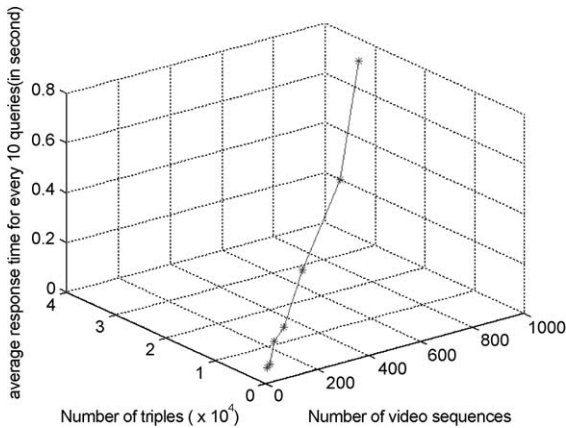


Fig. 12. The response time with respect to the number of video sequences and the number of triples.

5. Conclusions

5.1. Summary

In this paper, a novel approach of video retrieval is presented. The temporal relationships between objects and their assigned prime numbers form the basis of the indexing process. In the query process, successive modular operations allow a query to be finished in a linear time. In addition, the triple construction algorithm provides a semi-automatic method to generate temporal triples, as appose to most existing methods which are completely manual.

5.2. Future works

Here, we would like to mention the following areas of investigation which may merit further study.

- (1) The interactive annotation procedure could be fully automated by the assistance of a knowledge-based system.

- (2) In the indexing process, both spatial and temporal relationships could be jointly considered to create triples.

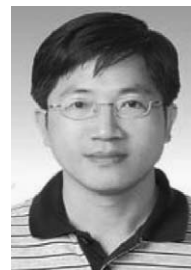
References

- [1] H.J. Zhang, A. Kankanhalli, S.W. Smoliar, Automatic partitioning of full-motion video, *ACM Multimedia Systems* 1 (1) (1993) 10–28.
- [2] U. Gargi, R. Kasturi, S. Strayer, Performance characterization of video-shot-change detection methods, *IEEE Transactions on Circuits and Systems for Video Technology* 10 (1) (2000) 1–13.
- [3] C.C. Lo, S.J. Wang, Video segmentation using a histogram-based fuzzy c-means clustering algorithm, *Computer Standards & Interfaces* 23 (2001) 429–438.
- [4] A.K. Jain, A. Vailaya, X. Wei, Query by video clip, *Multimedia Systems* 7 (1999) 369–384.
- [5] Y.J. Zhang, H.B. Lu, A hierarchical organization scheme for video data, *Pattern Recognition* 35 (2002) 2381–2387.
- [6] S.W. Smoliar, H.J. Zhang, Content-based video indexing and retrieval, *IEEE Multimedia*, (1994) 62–71.
- [7] S.K. Chang, C.W. Yan, D. Dimitrof, T. Arndt, An intelligent image database system, *IEEE Transactions on Software Engineering* 14 (1988) 681–688.
- [8] C.C. Chang, S.Y. Lee, Retrieval of similar pictures on pictorial databases, *Pattern Recognition* 24 (7) (1991) 675–680.
- [9] C.C. Chang, Spatial match retrieval of symbolic picture, *Journal of Information Science and Engineering* 7 (1991) 405–422.
- [10] C.C. Chang, J. Liang, Similarity retrieval on pictorial databases based upon module operation, *Proceeding of the Third International Symposium on Database Systems for Advanced Applications*, 1993, pp. 19–26.
- [11] J.F. Allen, Maintaining knowledge about temporal intervals, *Communications of the ACM* 26 (11) (1983) 832–843.
- [12] R. Hjelqvold, R. Midtstraum, O. Sandst, A temporal foundation of video databases, *Proc. Int'l Workshop Temporal Database*, 1995.
- [13] T.D.C. Little, A. Ghafoor, Interval-based conceptual models for time-dependent multimedia data, *Transactions on Knowledge and Data Engineering* 5 (4) (1993) 551–563.
- [14] S. Pradhan, K. Tajima, K. Tanaka, A query model to synthesize answer intervals from indexed video units, *IEEE Transactions on Knowledge and Data Engineering* 23 (5) (2001) 824–838.
- [15] M.E. Donderler, O. Ulusoy, U. Gudukbay, A rule-based video database system architecture, *Information Sciences* 143 (2002) 13–45.
- [16] E. Oomoto, K. Tanaka, OVID: design and implementation of a video-object database system, *Transactions on Knowledge and Data Engineering* 5 (4) (1993) 629–643.
- [17] R. Hjelqvold, R. Midtstraum, O. Sandst, Searching and browsing a shared video database, *Multimedia Database Systems. Design and Implementation Strategies Chapter 4*, Kluwer Academic Publishing, Boston, 1996.
- [18] E.J. Hwang, V.S. Subrahmanian, Querying video libraries, *Journal of Visual Communication and Image Representation* 7 (1) (1996) 44–60.
- [19] C.C. Chang, C.Y. Chen, J.K. Jan, On the design of a machine-independent perfect hashing scheme, *Computer Journal* 34 (5) (1991) 469–474.
- [20] C.C. Chang, C.C. Lee, A spatial match retrieval mechanism for symbolic pictures, *Journal of Systems and Software* 44 (1998) 73–83.
- [21] O. Herzog, A. Miene, Th. Hermes, P. Alshuth, Integrated information mining for texts, images, and videos, *Computer and Graphics* 22 (6) (1998) 675–685.
- [22] R. Smith, S. Huang, M. Ortega, M. Mehrotra, Relevance feedback: a power toll for interactive content-based image retrieval, *IEEE Transactions on Circuits and systems for Video Technology* 8 (1998) 644–655.
- [23] S. Santini, R. Jain, Integrated browsing and querying for image database, *IEEE Multimedia* 7 (3) (2000) 26–39.



Chi-Chun Lo was born in Taipei, Taiwan. He received a B.S. degree in Mathematics from the National Central University, Taiwan, in 1974, an M.S. degree in Computer Science from the Memphis State University, Memphis, TN, in 1978, and a Ph.D. degree in Computer Science from the Polytechnic University, Brooklyn, NY, in 1987. From 1981 to 1986, he was employed by the AT&T Bell Laboratories, Holmdel, NJ. From 1986 to 1990, he worked for the Bell

Communications Research, Piscataway, NJ. Since 1990, he has been with the Institute of Information Management, National Chiao-Tung University, Taiwan. At present, he is a full professor of the Institute. His major current research interests include network design algorithm, network management, network security, network architecture, and wireless communications.



Shuenn-Jyi Wang was born in Taoyuan, Taiwan, Republic of China, on September 30, 1963. He received the BS degree in Applied Mathematics, and MS degree in Electronic Engineering from the Chung Cheng Institute of Technology, in 1987 and 1993, respectively, and the Ph.D. degree in the Institute of Information Management from the National Chiao-Tung University, Hsinchu, Taiwan, in 2003. Currently, he is an assistant professor of the

Department of Computer Science, Chung Cheng Institute of Technology, National Defense University, Taoyuan, Taiwan. His major current research interests include digital image processing, and multimedia database system.