VIRON: An Annotation-Based Video Information Retrieval System

Ki-Wook Kim Ki-Byoung Kim Hyoung-Joo Kim

SNU-OOPSLA-Lab.
Dept. of Computer Eng.
Seoul National Univ.
Seoul, Korea 151-742
{kwkim@candy.snu.ac.kr, kbkim@candy.snu.ac.kr, hjk@papaya.snu.ac.kr}

Abstract

To provide efficient search and retrieval of video data from large archives, we need to model video data appropriately. In this paper we propose a video data model and describe the design and implementation of the annotation-based video retrieval system VIRON (Video Information Retrieval On Notation) [1] *based on the proposed model. This model provides the mechanism of sharing and reusing annotations among users by introducing descriptor schema. In order to process query efficiently, an annotated video unit is mapped into an unified video annotation stream.*

VIRON is composed of three tools: CVU manager, which is used to manage and visualize conceptual video units; annotator to annotate video data by offering interactive video player; and video query tool to pose and process video queries.

1 Introduction

Recently, advances in computer hardware have made significant progress in the development of application systems supporting video data. Large scale of video archive is now available to users as various forms - video on demands, interactive television, personalized news, etc. Without an efficient and reasonable mechanism for retrieving video data, large archive of video data remains as merely unmanageable resources of data. Accordingly, the retrieval and representation of video data becomes one of the main research issues in video database.

As for the representation of the video data, there have been mainly two approaches: (1) content-based video retrieval [3, 4, 18], and (2) annotation-based video retrieval [6, 10, 14, 15, 16, 17]. Even though the state-of-the-art image processing technologies make it possible to automatically analyze scene breaks and pauses in audio [19, 20], posing and processing video queries are still technically difficult.

In this paper we consider the issue of how users could manipulate the video data more efficiently. We propose a video data model based on the video annotation. In our model, it is possible to represent video information as structured data, provided that computer-supported human annotation could support reasonable facilities for annotation and retrieval of video data.

2 Related works and our approach

2.1 Related works

Over the last decade, there has been noticeable research progress in the area of video databases.

EVA [15, 16] developed by Mackay et. al. is an annotation system for video data. Though EVA system enables users to make annotations on video data and to analyze them, it does not support efficient sharing and reusing of annotations among users.

Oomoto and Tanaka [17] proposed a video-based object oriented data model, OVID. They introduce the notion of video object in which they can identify the meaningful features, and compose those features. This model offers an efficient framework for organizing a lot of video descriptional data, but it does not address the separation of temporal data and descriptional data of annotation. The OVID system has a limitation on the point of the sharability and reusability of annotation among users.

[1] This research is supported by KOSEF under project no. 95-1022, "A study of set-top box for multimedia demand driven systems".

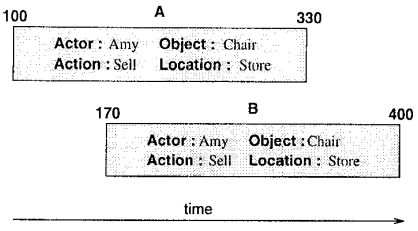

Figure 1: Example of annotation objects

Gibbs et. al.[7] modeled the stream-based temporal multimedia data using object oriented methods. One of their main subjects is concerned with generalized modeling of the time-based media, but they does not focus on the mechanism for handling video data annotation.

Hjelsvold and Midtstraum [9] presented a generic data model for capturing video information and structure. Although it provides a mean for indexing a video stream, the facility for controlling, sharing, and querying video data annotation are relatively weak.

Other works on video data annotation or video query processing related to this paper include Athena Muse [10] which introduced the notion of multi-dimensional information, HERMES [11] which described the query of video data, TGQL [8] which specified temporal relationships between video objects, and Media Streams [5] which proposed the mechanism for video data annotation using icons.

2.2 Terms and issues

2.2.1 Terminology

Before describing problems related to annotation, we need to clarify the terms - annotation, descriptional data, temporal data, descriptor, description, attribute, and value - as they are used in the context of this paper.

We use the term, video *annotation* (*object*), to express an object that is temporally linked to one or more video segments and associated with descriptional information (text, audio, or graphical object) to characterize chosen segments. In this paper, an annotation will be used to mean a text annotation. As for an annotation, its information can be divided into *descriptional data* which describe the content of video and *temporal data* which subsume the start and end frame number. The descriptional data are represented by the *descriptor* as a type of specification and the *description* as an instance of the type. Additionally, a descriptor could be composed of one or more *attributes* and a description be composed of corresponding *values*. In Figure 1, an annotation object A has descriptional data such as "`Actor: Amy`" where "`Actor`" is an attribute of descriptor and "`Amy`" is a corresponding value of description, and temporal data that include the range from frame number 100 to 330.

2.2.2 Issues in our approach

In our video data model, the descriptional data and temporal data of an annotation object are managed respectively. The proposed video data model has the following features.

1. **Division of descriptional data and temporal data**
   We divide the content of an annotation object into descriptional data and temporal data. Descriptor schema is introduced to manage descriptional data efficiently and effectively. Temporal data and the reference to descriptions on schema are mapped into video annotation stream to process the query.

2. **Descriptor schema for descriptional data**
   In order to represent descriptional data, we propose a *descriptor schema* based on OODB schema [12]. Because descriptor schema organizes the descriptor and description, users can create and manage a consistent annotation database effectively. Also, it is relatively easy to understand the descriptional data made by others.

3. **Video annotation stream for temporal data**
   The *video annotation stream* includes temporal data of annotation and the reference to the description. As new annotations are brought in a database, their temporal and descriptional data are progressively mapped into the unique video annotation stream properly. On the basis of temporal and referenced descriptional data, the video annotation stream provides a framework for efficient query processing. Also it functions as a basis of powerful queries such as semantic relationship query and implicit conjunctive query.

4. **Semantic relationship query**
   Additionally, the system allows users to pose queries which include fundamental query, relationship query, and conjunctive query. Especially, the semantic relationship query, newly proposed in this paper, connotes the query that inquires whether there exist video units which are related semantically with a common attribute value.

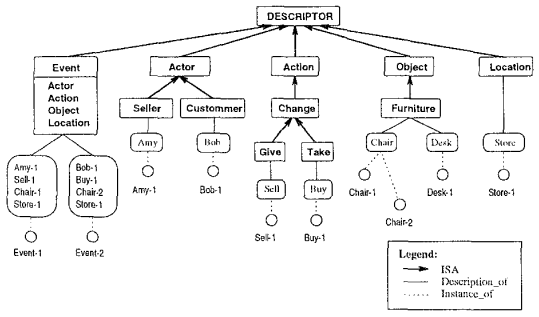

Figure 2: Example of the descriptor schema

3 Video data model

3.1 Overview

Users make an annotation on meaningful segments of video sequence. In a sense that an annotation object connotes the conceptual level of annotation, we call the annotation object made directly by users *conceptual video unit*. Users make a descriptional data for selected segments of video frame sequence, as in Figure 1. Accordingly, a CVU can be represented in terms of descriptional data and temporal data. We manage descriptional data and temporal data separately, in order to improve the reusability of descriptional data and to process query efficiently.

On the one hand, the descriptional data include the properties of CVU which could be specified with descriptor and description. The descriptor and description eventually embody a descriptional data structure, *descriptor schema*. On the other hand, the *video annotation stream* includes temporal data and the set of reference to descriptions on descriptor schema.

3.2 Descriptor schema

In this section, we will define descriptor schema incorporating the descriptional data extracted from CVUs, say, descriptors and corresponding descriptions.

Hereafter, $\mathcal{T}, \mathcal{D}, \mathcal{A}$, and $\mathcal{OID}$ denote the disjoint countably infinite set of the name of descriptor, description, attribute, and the object identifiers of description respectively.

Definition 3.1: A *descriptor schema* is an ordered tuple (T, ISA, $\top$, D, $Description\_of$).

- (T, ISA, $\top$) is a directed tree (with edges pointing from child to parent) rooted on $\top$, such that:
  1. T is a finite subset of $\mathcal{T}$ such that $T = T^{sh} \cup T^{unsh}$, where $T^{sh} \cap T^{unsh} = \emptyset$. T^{sh} denotes a set of *sharable* descriptor types; T^{unsh} denotes a set of *unsharable* descriptor types. Element of T consists of one or more *attributes* A which is a finite subset of $\mathcal{A}$.
  2. $ISA \subseteq T \times T$, and s is a *subtype* of t (t is a *supertype* of s), denoted $s \preceq t$ ($t \succeq s$), if there is a path from s to t in $(T, ISA, \top)$.
  3. $\top$ is a system-defined *meta descriptor type*, named *DESCRIPTOR*.
- D is a finite subset of $\mathcal{D}$ such that $D = \{d \mid d =< OID, V >\}$, where OID, of which element is denoted *OID*, is a finite subset of $\mathcal{OID}$ and V is a set of *attribute values* of description d, of which elements are atomic values, i.e. strings and numbers, or *OID* of descriptions.
- $Description\_of \subseteq T \times D$, where $Description\_of(t, d)$ denotes that d is a description of descriptor type t, denoted $d \prec t$, and d has attribute values corresponding to attributes of t.

Intuitively, an unsharable description can have *one or more OID* corresponding to the number of reference to this description by CVU, whereas a sharable description can have only one *OID* because it can be shared.

Example 3.1: In Figure 2, we assume that `Object` $\in T^{unsh}$ and `Location` $\in T^{sh}$.

`Furniture` $\preceq$ `Object`, so `Furniture` $\in T^{unsh}$. Since `Chair` $\prec$ `Furniture`, `Chair` is a description of an unsharable descriptor type. Therefore, represented as $<$ {"`Chair-1`", "`Chair-2`"}, {"`Chair`"} $>$, `Chair` has two *OID* after CVUs in Figure 1 are defined in annotation database.

On the contrary, `Store` $\prec$ `Location`, so `Store` is a description of a sharable descriptor type. Hence, represented as $<$ {"`Store-1`"}, { "`Store`"} $>$, `Store` can have one *OID*. □

3.3 Video annotation stream

Since descriptional data in a CVU have been extracted as in section 3.2, a CVU could be now represented as a set of *OID* of descriptions and temporal information. We call the CVU in this temporal state *video unit*. A video unit is then represented as a tuple (Val, I), where Val is a *value set* and I is an *interval* of the video unit. Val is a set of *OID* of descriptions in descriptor schema. Following the notation in [1], we use the term interval such that $I = [i, j)$ which

denotes $\{n \mid i \leq n < j$, where i : *start frame number* and j : *end frame number*$\}$.

Then, we could specify the temporal relationship between video units according to their intervals. First, video unit u_1 and u_2 are *partially ordered*, denoted $u_1 \sqsubseteq u_2$, if $i_1 < j_1 \leq i_2 < j_2$ for u_1's interval $[i_1, j_1)$ and u_2's interval $[i_2, j_2)$. Especially, if $i_2 = j_1$, then u_1 and u_2 are *adjacent*. Alternatively, if $u_1 \not\sqsubseteq u_2$ and $u_2 \not\sqsubseteq u_1$, then u_1 and u_2 are *overlapped*, denoted $u_1 \propto u_2$. Intuitively, $u_1 \sqsubseteq u_2$ means that u_1 precedes u_2, and $u_1 \propto u_2$ means that u_1 and u_2 have common segments of video frame sequence.

Then, we can define a unique video annotation stream in one video annotation database.

Definition 3.2: A *video annotation stream*, denoted STR, is a set of video units, where $\forall u_i, u_j \in STR$, $u_i \sqsubseteq u_j$, iff $i < j$.

Intuitively, video annotation stream is a set of partially ordered video units made up from all CVUs. All the references of descriptions and temporal data are mapped into video annotation stream.

3.3.1 Value equality and object identity

We will introduce two basic operations applicable in query condition. As stated previously, a description can have several *OID*, corresponding to the reference of the description. Hence, there could be two types of relationship between values of video units: the same attribute value of description but different *OID*; and the same attribute value of description and *OID*.

To reflect two types of relationship, we propose two kinds of equalities which are *different* from object equality and value equality defined in OODB [12] in the following sense. We classify equality into value equality and object identity. Denoted by "=", value equality implies that values of two video units to be compared have the same attribute value. Object identity, denoted "==", means that values of two video units have the same attribute value and *OID* also.

4 Query on video data

Here we will describe the types of queries and the mechanism for query processing using descriptor schema and video annotation stream. We categorize queries such as: fundamental query, relationship query, and conjunctive query. Due to space limitations in this paper, we only describe the overall description of our proposed query. More specified explanation about query are described in [13].

4.1 Fundamental query

Fundamental query is about the video units themselves satisfying the condition clause. According to predicates involving video units, we classify fundamental queries as follows. into fundamental description query, fundamental content query, and fundamental occurrence query.

Fundamental description query provides a way to retrieve descriptions in result video units. With fundamental content query, we can retrieve all video units containing given descriptions. Conditioned with a temporal span, fundamental occurrence query enable to find all video units in that span.

4.2 Relationship query

The query in this category involves the relationship between video units. We classify the query into temporal relationship query and semantic relationship query.

Temporal relationship query reflects the 13 primitive temporal relationships between two video units described in [2]. Semantic relationship query involves the semantic link connected between a reference video unit and the others. In order to specify the semantic relationship, we introduce the object identity into the relationship query. The object identity means the equality of object identifiers and is used to incorporate the identification of two video units into the query.

4.3 Conjunctive query

Users are able to pose the conjunctive query in that the queries described previously can be combined with connectives such as "AND", "OR". For example, "Find all the video units where Amy drives a car and Bob converses with Jane in the road".

5 Design and implementation of the VIRON system

5.1 Overview of the VIRON system

The overall architecture of VIRON is shown in Figure 3. In terms of functionalities, the system consists of two subsystems: annotation subsystem and query processing subsystem.

With the annotation subsystem, users can construct an annotation database for video data. The annotation subsystem consists of two modules: annotator and CVU manager. Users can interact with the

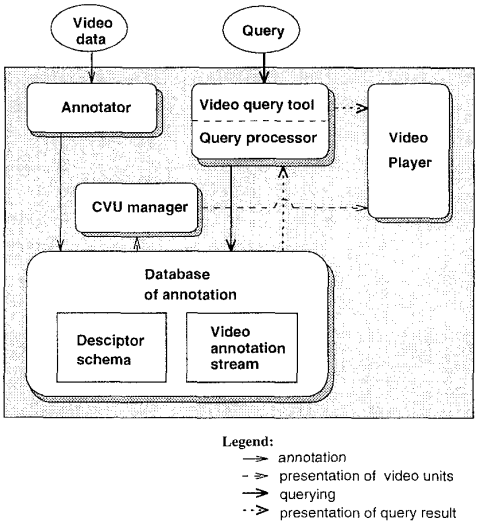

Figure 3: Overall architecture of the VIRON system

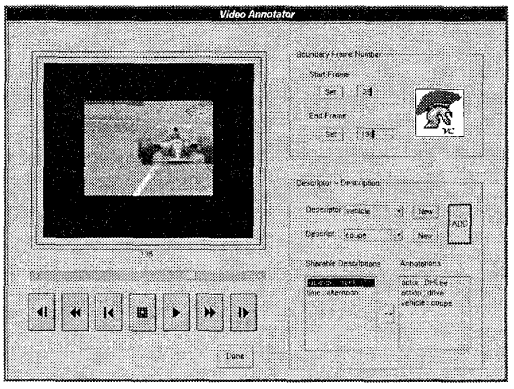

Figure 4: Annotator

system through the annotator to make an annotation on a specified video frame sequence. The CVU manager transforms video units in video annotation stream into a CVU and presents it with a video player. With the query processing subsystem, users can pose query and examine results of the query.

We adopt SOP [2]as a video annotation database.

5.2 Annotator

As shown in Figure 4, annotator enables users to annotate video data interactively. To increase user

[2]SOP (SNU OODBMS Platform) is an OODBMS developed by Seoul National University OOPSLA Lab.

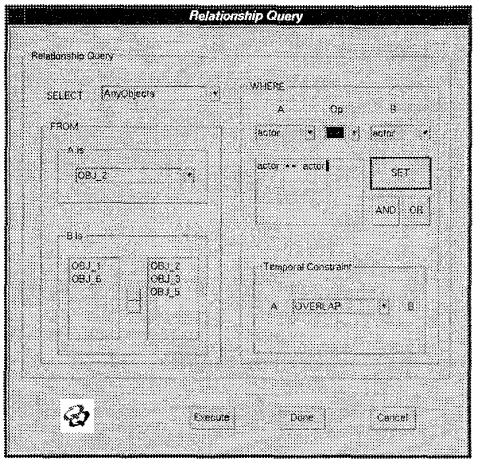

Figure 5: Relationship query

interactivity with the system, video player module is combined with the annotator. Thus a user can make an annotation on the right video sequence frame that he watches.

As a result of computer-supported selection or creation of descriptors and descriptions as well, a CVU is created at last and then mapped into video annotation stream.

5.3 CVU manager

Using CVU manager, users can view the whole configuration of CVUs, and move to annotator and Video query tool via CVU manager. The function CVU manager performs is CVU management and semantic link connection.

5.4 Video query tool

Video query tool enables users to pose a query. Via video query tool, users can pose several kinds of fundamental queries on annotated video data. After fundamental query results are formed, users can pose a relationship query.

Figure 5 shows a relationship query which means: "Find any video units among video units (`OBJ_2`, `OBJ_3`, and `OBJ_5`) in which the same actor appears as in a reference video unit `OBJ_2` (semantic relationship query), and which are temporally overlapped with `OBJ_2` (temporal relationship query)". Additionally, users can pose conjunctive query via set operation tool for query results. The result of set operation is represented as an ellipsis form in the window of query

tool.

6 Conclusion

We have proposed a new powerful model for annotation-based video retrieval. The proposed model is comprised of two main parts: descriptor schema and video annotation stream. Based on this model, descriptor schema enables users to share and reuse existing annotations. Due to the flexibility of descriptor schema, users can reform and reuse the annotation data structure. Because the video annotation stream consists of partially ordered video units which include reference to descriptor schema and temporal information, queries posed by users can be processed efficiently.

According to the proposed model, the annotation-based video retrieval system, named VIRON, has been implemented. The VIRON consists of two main subsystems: annotation subsystem and query processing subsystem. annotation subsystem is composed of annotator and CVU manager. Query processing subsystem includes video query tool to pose and process query.

VIRON supports various types of video queries supported by other systems and suggests a new useful kind of video query, called semantic relationship query.

References

[1] S. Adali, K. Candan, S. Chen, K. Erol, and V. Subrahmanian. "Advanced Video Information System: Data Structures and Query Processing". Technical report, Univ. of Maryland, 1995.

[2] J. Allen. "Maintaining knowledge about temporal intervals". *Communication of the ACM*, 26:832–843, Nov. 1983.

[3] I. A. R. Center. "Query by Image and Video Content: The QBIC System". *IEEE Computer*, 28(9):23–32, Sept. 1995.

[4] T. Chiueh. "Content-Based Image Indexing". *Proc. of the 20st VLDB Conf.: Santiago Chile*, pages 582–593, 1994.

[5] M. Davis. "Media Streams: An Iconic Visual Language for Video Representation.". *Readings in Human-Computer Interaction: Toward the Year 2000*, pages 854–866, 1995.

[6] M. Davis. "Media Streams: Representing Video for Retrieval and Repurposing.". *Ph.D. Thesis, Massachusetts Institute of Technology*, 1995.

[7] S. Gibbs, C. Breiteneder, and D. Tsichritzis. "Data Modelling of Time-Based Media". *Proc. 1994 ACM SIGMOD Conf. on Management of Data*, pages 91–102, 1994.

[8] S. Hibino and E. Rundensteiner. "Interactive Visualizations for Exploration and Spatio-Temporal Analysis of Video Data". *IJCAI'95 Workshop on Intelligent Multimedia Information Retrieval*, 1995.

[9] R. Hjelsvold and R. Midtstraum. "Modelling and Querying Video Data". *Proc. 1994 Int'l. Conf. on Very Large Databases*, pages 686–694, 1994.

[10] M. Hodges, R. Sasnett, and M. Ackermann. "A construction set for multimedia applications". *IEEE Software*, 6:37–43, Jan. 1989.

[11] E. Hwang and V. Subrahmanian. "Querying Video Libraries". Technical report, Univ. of Maryland, 1993.

[12] W. Kim. *Introduction to Object-Oriented Databases*. The MIT Press, 1990.

[13] K.W.Kim and H.J.Kim. "Design and Implementation of Video Annotation System". *submitted to Journal of Korea Information Science Society*, 1996.

[14] A. M. Lab. "The visual almanac". *Apple Computer*, 1989.

[15] W. Mackay. "EVA: An experimental video annotator for symbolic analysis of video data". *SIGCHI Bulletin*, 21:68–71, Oct. 1989.

[16] W. Mackay and G. Davenport. "Virtual video editing in interactive multimedia applications". *Communication of the ACM*, 32:802–810, July 1989.

[17] E. Oomoto and K. Tanaka. "OVID: Design and Implementation of a Video-Object Database System". *IEEE Trans. on Knowledge and Data Engineering*, 5(4):629–643, 1993.

[18] S. Smoliar and H. Zhang. "Content-Based Video Indexing and Retrieval". *IEEE Multimedia*, 1(2):62–72, 1994.

[19] Y. Tonomura and et. al. "VideoSpaceIcon: tools for anatomizing content". *Proc. of INTERCHI'93 conf. on human factors in computing systems*, pages 131–136, 1993.

[20] H. Zhang, A. Kankanhali, and S. Smoliar. "Automatic partitioning of full motion video". *Multimedia Systems*, pages 10–28, Jan. 1993.