Orthogonally modeling video structuration and annotation: exploiting the concept of granularity

M. Dumas, R. Lozano*, M.-C. Fauvet, H. Martin and P.-C. Scholl

LSR-IMAG, University of Grenoble
B.P. 72, 38402 St-Martin d'He-'res, France
Marlon.Dumas@imag.fr

Abstract

In this paper, we exploit the concept of granularity to design a video metadata model that addresses both *logical structuration* and *content annotation* in an orthogonal way. We then show that thanks to this orthogonality, the proposed model properly captures the interactions between these two aspects, in the sense that annotations may be independently attached to any level of video structuration. In other words, an annotation attached to a scene is not treated in the same way as an annotation attached to every frame of that scene. We also investigate what are the implications of this orthogonality regarding querying and composition.

Introduction

For a long time, videos were managed by specific environments due to their huge size and demanding hardware requirements. Improvements in data compression, continuously increasing network transfer rates and processing capacity, and advances in operating systems, now allow conventional computing systems to handle videos, thereby opening application areas such as video-on-demand, video conferencing and home video editing.

The specificities of these applications have been addressed from several perspectives by a large body of research works, prototypes and commercial tools (Elmagarmid et al., 1997). Many of these efforts have focused on performance issues related to storage, real-time transfer in distributed environments, quality of service, synchronization, etc. However, as the needs of these applications evolve, other issues such as content-based retrieval, summary extraction, interactive editing and browsing, etc. become increasingly important. In particular, advances in automatic indexing techniques, together with video metadata standardization efforts such as MPEG-7, have accentuated the need for high-level representation models for video annotations.

In this setting, we advocate that database technology is likely to play an increasingly important role in video management, provided that it offers high-level concepts and interfaces for storing, retrieving and editing videos and their associated metadata. Our work is intended to be a contribution in this direction.

*Supported by SFERE–CONACYT, Mexico.

Video modeling in the context of a database entail numerous issues which may be addressed from a temporal representation viewpoint. In this paper we focus on two of them, namely *structuration* (also known as *segmentation*) and *annotation* (also known as *indexing*). One of the main originalities of our approach, is that it allows annotations to be independently attached to any level of video structuration. This feature considerably increases the expressive power of the resulting model. Indeed, in most previous proposals (Oomoto and Tanaka, 1993; Gandhi and Robertson, 1994; Adali et al., 1996; Li et al., 1997), annotations may only be attached to the video frames and not to its shots or its scenes. This withholds the user from expressing facts which are true of a given scene, without being true of each frame in this scene. For instance, stating that in a given scene two characters talk to each other, is something that is true at the granularity of the scene, without being true of each frame in that scene. Similarly, given a tennis match video, structured into two levels respectively modeling the games and the sets of the match, stating that the score of a set is 6–4 does not entail anything about the score of a particular game in that set.

We also investigate the issues of metadata querying and composition in the context of our model. To address the first of these issues, we propose a set of algebraic operators on video annotations. Some of these operators allow the user to compare annotations attached to different levels of structuration. To address video composition on the other hand, we extend the semantics of some previously proposed composition operators, so as to propagate the structuration and the annotations of the argument videos into the resulting ones. Unlike previous proposals, the videos obtained through these composition operators therefore inherit some metadata from their sources.

For presentation purposes, we adopt throughout the paper the terminology and notations of the ODMG object database standard (Cattell et al., 2000). However, since the concepts and operators that we introduce are encapsulated into abstract datatypes, the proposed approach may be accommodated into other data models.

The rest of the paper is structured as follows. First, we describe our approach to model video structuration and annotation. Next, we discuss the implications of our modeling choices regarding querying and composition. Lastly, we

compare our proposal with respect to previous ones and provide some concluding remarks.

Video structuration

Video structuration is the process of grouping the frames of a video according to their semantical correlations. For instance, a video is usually structured into shots, scenes, and video sequences.

The structure of a video is analogue to the table of contents of a text document. As such, it can be used as a guide for browsing, as the entry point for building video abstracts, or as a mean to retrieve the context of a particular piece of video.

There is a strong similarity between the structuration of a video and the structuration of time in calendars. Accordingly, we propose a video structuration model based on the concept of *granularity*. This concept has been extensively studied in several contexts such as temporal reasoning in artificial intelligence (Hobbs, 1985), and temporal databases (Wang et al., 1995). The definition of granularity that we adopt is a generalization of the one developed in this latter work.

It is important to note that in this work, the concept of granularity is merely used to model a hierarchical structuration, and not as an absolute temporal metric system. Indeed, when time is structured into "regular" granularities such as minutes and hours, these granularities may then be used to express distances between time points.

The basic concept of the video data model that we consider is that of *time-line*. At an abstract level, a time-line TL is a pair (D, $<_{\mathsf{TL}}$), made up of a finite set of *time-marks* D, and a binary relation $<_{\mathsf{TL}}$ over this set defining a total linear order.

Time-lines model independent time-coordinate systems, with respect to which data and meta-data may be temporally anchored. Each video has its own time-line, which captures the temporal relations between its elementary components (i.e. its frames).

A *granularity* over a time-line TL is defined as a partition of TL into convex sets: each of these convex sets is then seen as an atomic *granule* and every time-mark of the time-line is approximated by the granule which contains it. A granularity is intrinsically attached to a unique time-line. The function that maps a granularity into a time-line is subsequently denoted TimeLine. Thus, the expression TimeLine(G) denotes the time-line that granularity G partitions.

A hierarchical structure is defined over the set of granularities of a time-line through the definition of the *finer-than* relation. A granularity G1 is *finer-than* another granularity G2 (G1 $\prec$ G2) if TL(G1) = TL(G2), and if a mapping can be established between each granule of G1 and a set of consecutive granules of G2

Given two granularities G1 and G2 such that G1 $\prec$ G2, two mapping functions are defined: one for *expanding* a granule of G2 into a set of granules of G1 (noted $\epsilon_{\mathrm{G2,G1}}$), and the other for *approximating* a granule of G1 by a granule of G2 (noted $\alpha_{\mathrm{G1,G2}}$).

- $\forall g_2 \in G2 \;\; \epsilon_{\mathrm{G2,G1}}(g_2) = \{g_1 \in G1 \mid g_1 \subseteq g_2\}$
- $\forall g_1 \in G1 \quad \alpha_{\mathrm{G1,G2}}(g_1) =$ the unique $g_2 \in G2$ such that $g_1 \subseteq g_2$

Any level of structuration of a video is naturally modeled by a granularity over its time-line. Indeed, the set of shots of a video may be seen as a set of disjoint intervals of time-marks covering the whole time-line of the video. Notice that it is also possible to view the set of frames as a "degenerated" partitioning of a video time-line, in which each partition is composed of a single time-mark. The advantage of this modeling is that the notion of frame is seen as a structuration level of a video in the same way as shots or scenes. This allows to attach annotations to shots or scenes in the same way as they are attached to frames, as detailed in the next section.

The following ODMG interfaces show how the concept of granularity can be used to model a classical structuration schema, in which videos are decomposed into sequences, scenes and shots.

```
interface TimeLine; /* not detailed here */
interface Granularity {
  TimeLine TimeLine();
  bool finerThan(Granularity other);
}
interface Video {
  attribute TimeLine VTL;
  attribute Granularity sequences;
  attribute Granularity scenes;
  attribute Granularity shots;
  attribute Granularity frames;
  /* Other attributes are described in the next section. */
}
```

For a given object V whose class implements the above Video interface, the following constraints apply:

- V.frames = MinGranularity(V.VTL), where MinGranularity denotes the function retrieving the finest granularity of a time-line (i.e. the granularity whose granules are all singletons). Such granularity is usually called the *chronon* of a time-line in the temporal database literature (Tansel et al., 1993).
- TimeLine(V.sequences) = TimeLine(V.scenes) = TimeLine(V.shots) = V.VTL
- V.shots $\prec$ V.scenes $\prec$ V.sequences

The above Video interface may be specialized through inheritance, so as to handle situations where other structuration levels are needed. For instance, in the context of a video about a tennis match, structuration levels such as point, game and set may be introduced.

Notice that two structuration levels within a single video are not necessarily comparable with respect to the "finer than" relation. For instance, consider a video about a tennis match possessing two structuration levels, namely point and shot. If a granule of the shot granularity overlaps two granules of the point granularity (without containing them completely), then neither point $\prec$ shot, nor shot $\prec$ point hold.

Video annotation

Given the current state of image processing technology, it is not reasonable in the general case, to dynamically extract

semantical information from a video during query evaluation over a video database. Instead, in order to formulate content-based queries over videos, their semantical content must be previously described as *annotations*.

Annotations are generally stored separately from the "raw" video data. This approach is quite natural, since annotations are normally only needed during video querying[1], while access to raw video data is only required during video playing. In addition, this approach allows to share the "raw" video data among several "virtual" videos, without necessarily sharing all the corresponding annotations (Lozano and Martin, 1998).

Our approach to video annotation is based upon the concepts of *instant* and *sequence*.

An *instant* is as an approximation of a connected region of a time-line by a granule. It is represented by a pair made up of a natural number (the *position* of the denoted granule) and a granularity. There is a subtle difference between an instant and a granule: a granule is simply a set of time-marks, while an instant is a reference to a granule, not the set of time-marks on itself. In particular, stating that a fact is true at an instant (representing for instants a particular scene) does not means that this information holds at each time-mark contained in the granule referenced by this instant.

A sequence is abstractly defined as a mapping from a set of instants defined over a single granularity, to a set of objects sharing a common structure. An example of a sequence is:

s1 → ⟨ main_character: "Linda", main_action: cook ⟩
s2 → ⟨ main_character: "Linda", main_action: cook ⟩
s3 → ⟨ main_character: "Linda", main_action: cook ⟩
s6 → ⟨ main_character: "Joe", main_action: eat ⟩
s7 → ⟨ main_character: "Joe", main_action: eat ⟩
s8 → ⟨ main_character: "Joe", main_action: eat ⟩

Where { s1, s2, s3, s6, s7, s8 } denote instants at the same granularity (e.g. a set of scenes of a video), and ⟨ att_name: value, att_name: value ⟩ denotes an object.

The above concepts are embedded into two ADTs respectively named Instant and Sequence. This latter is parameterized by a type modeling the set of possible values taken by this sequence at a given instant. In the sequel, we will use the notation Sequence<T> to denote the instantiation of the type Sequence with type T.

Since sequences are modeled as instances of an ADT, the user does not need to worry about the internal representation of a sequence in order to reason about them. For instance, the above sequence could be internally represented in the following way :

[s1..s3] → ⟨ main_character: "Linda", main_action: cook ⟩
[s6..s8] → ⟨ main_character: "Joe", main_action: eat ⟩

Where [s1..s3] and [s6..s8] denote intervals.

To achieve orthogonality between video structuration and annotation, we attach to each video, as many sequences as there are levels of structuration defined over it. Each of these sequences has its own granularity, and it groups all the annotations attached to a given level of structuration, as depicted in figure 1.

The following ODMG interface (which completes the one given previously) summarizes the above discussion.

```
interface Video {
  /* attributes described in the previous section go here */
  attribute Sequence<Object> sequencesAnnotations;
  attribute Sequence<Object> scenesAnnotations;
  attribute Sequence<Object> shotsAnnotations;
  attribute Sequence<Object> framesAnnotations;
}
```

For a given object V whose class implements the above interface, the following constraints apply:

- Granularity(V.sequencesAnnotations) = V.sequences, where Granularity(SQ) stands for the granularity of the domain of sequence SQ.
- Granularity(V.scenesAnnotations) = V.scenes
- Granularity(V.shotsAnnotations) = V.shots
- Granularity(V.framesAnnotations) = V.frames

As stated before, the above interface is not intended to be used "as is". Instead, the user may specialize it to account for particular kinds of videos. For example, consider a database managing a collection of movies identified by their title. The frames of each movie are annotated with the names of the actors appearing on them, while the scenes are annotated with the location where they take place. The following specification captures this situation, by introducing a class Movie which implements the interface Video. Notice that the attribute framesAnnotations "inherited" from this interface, is specialized to account for the structure of the annotations managed in this context. A similar remark applies to scenesAnnotations.

```
class Movie : Video (extent TheMovies, key title) {
  attribute string title;
  attribute sequence<Location> scenesAnnotations;
  attribute sequence<Set<string>> framesAnnotations;
}
```

Querying multi-level annotated videos

A set of algebraic operators is defined on sequences. These operators are divided into five groups: pointwise mapping, join, restriction, partitioning and splitting. The first three correspond to the classical projection, join and selection operators of the relational algebra, while the latter two are proper to sequences. In fact, partitioning and splitting are tightly connected to two important characteristics of sequences: granularity and order. In the sequel, we focus on the restriction and the partitioning operators. For details on the other operators, the reader may refer to (Dumas et al., 1999).

There are two restriction operators on sequences. The first one (noted during) restricts the domain of a sequence to the instants lying in a given set of instants. The second one (noted when) restricts a sequence to those instants at which its value satisfies a given predicate. Such predicate is given as a boolean function whose parameter denotes a value of the sequence's range. Concretely, the expression SQ as

[1] Close captions are an exception, since they are annotations that must be displayed during video playing.

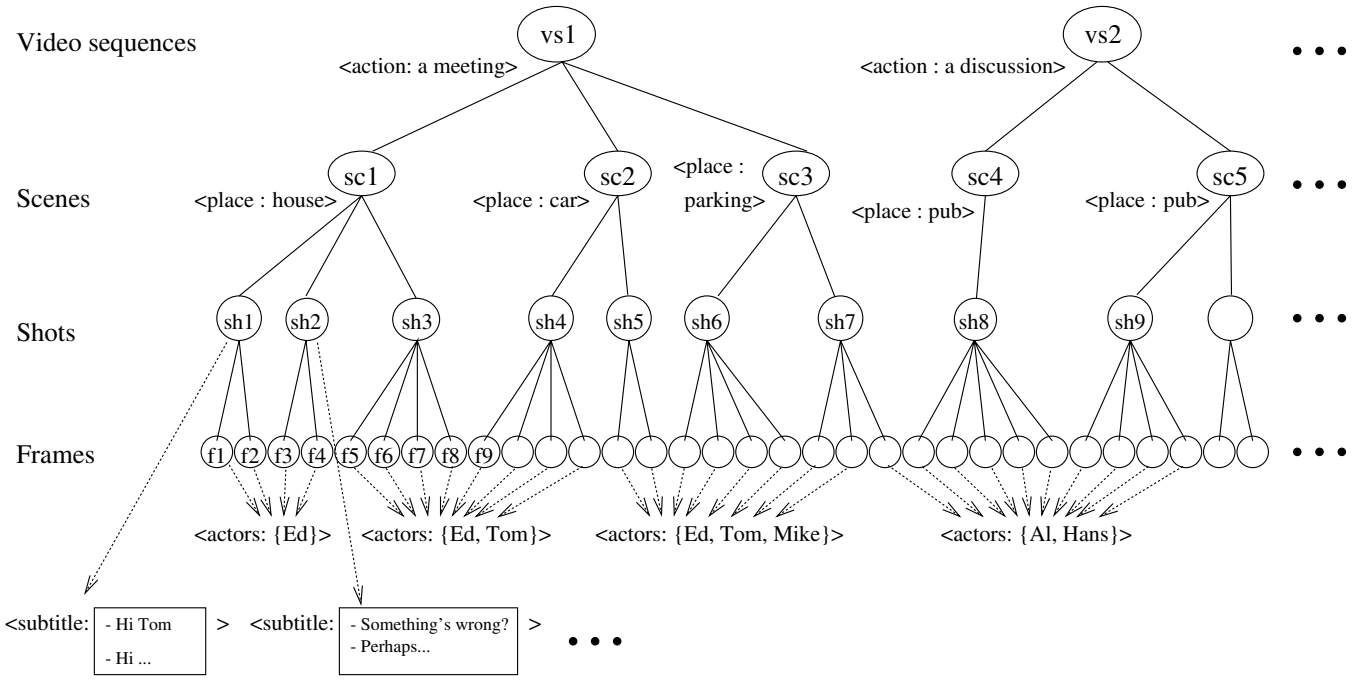

Figure 1: Sequences of annotations attached to a video. Video sequences (the coarsest units of structuration) are annotated with action descriptors, scenes with places, shots with close captions, and frames with the character names.

x when P(x) denotes a sequence obtained by restricting sequence SQ to those instants where its value satifies predicate P (variable x stands for the value of the sequence at a given instant).

The following two queries illustrate the above restriction operators.

Q1 : **Restriction (based on the domain)**
Retrieve the annotations attached to the frames appearing within the first 20 seconds of the movie entitled "Hopes" (assuming a constant presentation rate of 30 frames/second)

```
select F.framesAnnotations during
        [0 @ F.frames..(20 * 30) @ F.frames]
from TheMovies as F where F.title = "Hopes"
/* N @ G denotes the instant of granularity G whose position is N. Expression [I1..I2] denotes the closed interval of instants comprised between I1 and I2. */
```

Q2 : **Restriction (based on the range)**
In which movies is John present for more than 15 minutes (assuming the same presentation rate as above).

```
select F from TheMovies as F
where duration(F.framesAnnotations
                as anActorSet when "John" in anActorSet)
      >= (15 * 60 * 30)
      /* Operator "duration" retrieves the cardinality of the domain of a sequence. */
```

Notice that in both of these queries, we assumed a constant presentation rate when converting a number of seconds into a number of frames. However, virtual videos may involve several raw videos possibly recorded (and therefore presented) under different frame rates. In these situations, the conversion function between "seconds" and "frames" becomes far more complex. To our knowledge, this problem has not been addressed by any of the existing video data models. Indeed, the models which offer conversion functions between metric temporal values (e.g. durations expressed in terms of seconds) and frame numbers, assume a constant presentation rate (see for instance (Dionisio and Ca'rdenas, 1998)). We believe that this is an interesting perspective to our work.

The partitioning operator, namely partitioned by, accounts for granularity change (*zoom-out*). More specifically, SQ partitioned by G2, SQ being at granularity G1 (G1 $\prec$ G2), makes a partition of SQ according to granularity G2. The result is a sequence, at granularity G2, of sequences at granularity G1, such that the value of the main sequence at any instant I (at granularity G2) is the restriction of SQ to the interval expand(I, G1). This is illustrated in figure 2.

Since in our model, annotations may be independently attached to each level of video structuration, it is necessary to provide a mechanism for switching between these levels. The partitioned by operator provides this mechanism in our OQL extension.

Q3 : **Granularity change: sequence partitioning**
Retrieve the scenes of the movie entitled "Freedom", in which John is in at least half of the frames of the scene.

```
select F.framesAnnotations partitioned by F.scenes
      as aScene when (duration(aScene as anActorSet when
                                "John" in anActorSet)
                      >= duration(aScene) / 2)
```

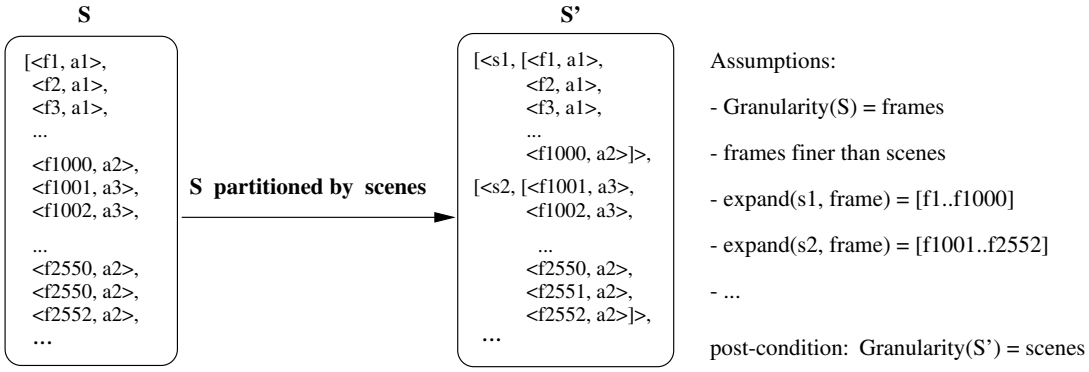

Figure 2: Partitioning operator: Instants at granularity frames (resp. scenes) are named f1, f2, etc. (resp. s1, s2, etc.)

from TheMovies as F where F.title = "Freedom"

In combination with the temporal join operator on sequences (not described here), the above partitioning operator may be used to express queries which simultaneously involve annotations at the granularity of the scene and at the granularity of the frame, as in : *"Retrieve those scenes in film Freedom, which take place in Paris, and in which John is present in at least half of the frames of the scene."*.

Composing multi-level annotated videos

Most existing video data models (Hjelsvold et al., 1995; Weiss et al., 1995) provide operators for generating videos by extracting and composing fragments of existing ones. In (Lozano and Martin, 1998) for instance, we studied five such operators : extraction, concatenation, union, intersection and difference.

However, none of these works considers the issue of propagating the metadata contained in the original videos when generating new ones through the above composition operators. As a result, the videos obtained through composition have no structuration and some of the annotations attached to the original videos are not inherited by the generated ones.

In (Dumas et al., 1999), we investigate this latter issue and extend the semantics of the five composition operators proposed in (Lozano and Martin, 1998), so as to preserve the structuration and annotations of the argument videos during composition. Figures 3 and 4 sketch the way in which this "metadata propagation" is carried out in the case of the extraction and the concatenation operators. The other three operators (union, intersection and difference) may be expressed in terms of these two.

Extraction.

The extraction operator (noted Extract), takes as parameter a video and a set of instants denoting video frames, and generates a new video containing exactly these frames.

The derivation of the structuration of the resulting video is essentially carried out through an operator Restrict(G, IS), which builds a new granularity, by restricting granularity G to the granules referenced within the set of instants IS. For instance :

Restrict({ {f1, f2, f3}, {f4, f5, f6}, {f7, f8, f9} },

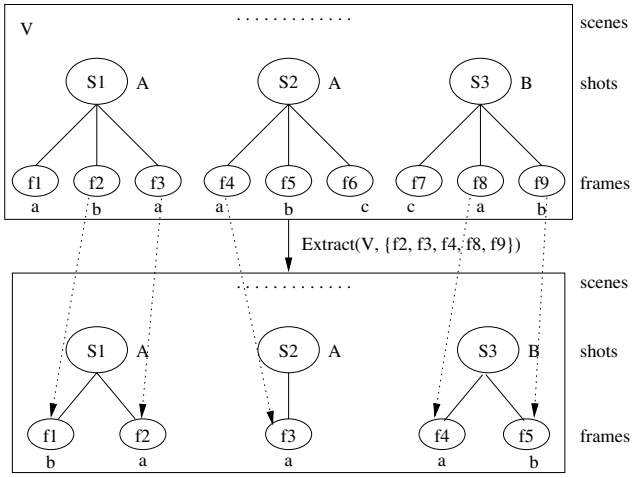

Figure 3: Propagation of the structuration and annotations during video extraction. A, B and C stand for annotations attached to the shots, while a, b, and c are annotations attached to the frames.

{ f1, f3, f7}) = { {f1', f2'}, {f3'} }

Which means that the resulting granularity has two granules : the first one composed of two time-marks (namely f1' and f2'), which correspond to frames f1 and f3 in the original video, and the second granule contains a single time-mark (namely f3'), which corresponds to frame f7 in the original video.

The derivation of the annotations, on the other hand, is performed in two steps. First, each of the sequences of annotations of the argument video is restricted to the set of frames given as parameter to Extract. This first step is carried out through the operator during on sequences.

Then, the resulting sequences of annotations are transformed into equivalent sequences over the granularities generated by the Restrict operator. This second step is carried out using an operator called Compact. Intuitively, this operator maps a sequence with a non-convex domain, into one with a convex domain. For example :

Compact({⟨3, v1⟩, ⟨5, v2⟩, ⟨6, v2⟩, ⟨9, v1⟩}) =

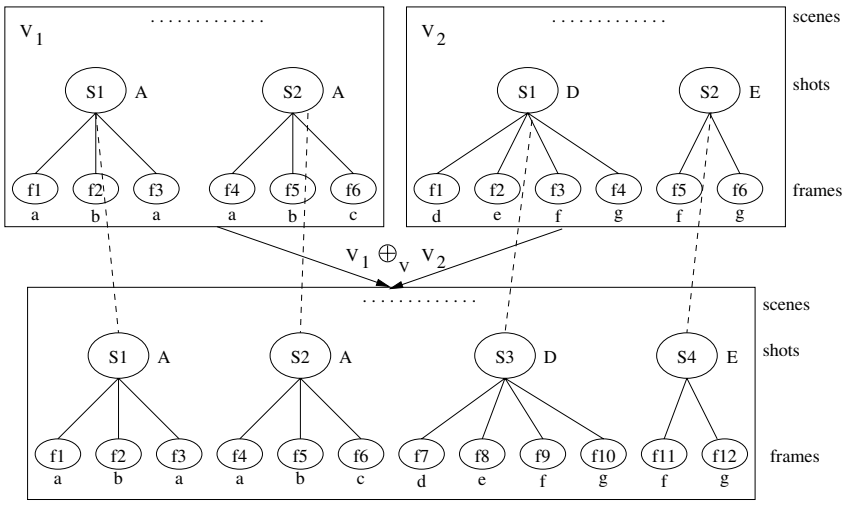

Figure 4: Propagation of the structuration and annotations during video concatenation

$\{\langle 1, v1\rangle, \langle 2, v2\rangle, \langle 3, v2\rangle, \langle 4, v1\rangle\}$

Notice that the extraction operator is defined in such a way that a shot is kept in the resulting video if at least one of the frames within this shot is kept (and idem for the other structuration levels). In some situations, this may lead to some semantical problems. For instance, if the annotation attached to a scene is "Linda walks", and that all the frames in this scene where Linda walks are taken off during the extraction process, then the resulting video has a scene whose annotation (i.e. "Linda walks"), is inconsistent with respect to its contents.

We believe that it is up to the user to cope with these situations. In other words, the user must make sure that the annotations derived by the Extract operator are consistent with the video contents, and correct them if needed.

Concatenation.

The concatenation operator (noted $\oplus_V$), takes as parameter two videos and simply concatenates them. The structuration and the annotations of the resulting video are obtained by simply concatenating the granularities and sequences of the two argument videos. During this process the frames, shots and scenes of the second argument are renumbered. For instance, in the concatenation depicted in figure 4, the shot number 1 of video V_2 becomes the shot number 3 in video $V_1 \oplus_V V_2$.

Related works

There is a wealth of works dealing with semi-structured formats for representing semantical video contents (i.e. annotations). The results of these works are currently being exploited by ongoing standardization efforts, such as MPEG-7 (Nack and Lindsay, 1999). In this approach, annotations are represented as "segment descriptors", i.e. hierarchically structured entities attached to a particular segment of a document, or possibly to an entire document. MPEG-7 is intended to be flexible, in the sense that user-defined annotation types are allowed in addition to the ones provided by the standard.

Our proposal is complementary to the above one, since it is not intended to define a low-level format for video storage, but rather a high-level data model for browsing, editing, composing and querying videos using the functionalities of an object DBMS.

The idea of using DBMS functionalities to store, browse, and query video contents is not novel: it has been applied in many prototype systems and data model proposals (Elmagarmid et al., 1997). The innovation of our approach, lies on the orthogonality with which the different aspects of video modeling are tackled. Indeed, in our proposal annotations may be independently attached to each level of the video structuration, whereas in most of the existing video data models, e.g. OVID (Oomoto and Tanaka, 1993), AVIS (Adali et al., 1996) and CVOT (Li et al., 1997), annotations may only be attached to the frames[2].

This approach withholds the user from expressing facts which are true of a given scene, without being true of each frame in that scene. For instance, suppose that the user wishes to encode into a set of annotations, the actions that take place in a movie. A straightforward labeling of the frames as proposed in the above models, forces the user to attach an action to each frame. However, an action, such as two characters having a discussion, is not an instantaneous event whose truth can be associated to a particular "image". It is rather the outcome of the concatenation of atomic events such as two characters looking at each other, intermittently talking about a common subject, with some kind of pitch and voice intensity, etc. In other words, it is a fact whose truth can only be attached to a video segment with some semantic unit, i.e. a scene.

(Hjelsvold et al., 1995) is perhaps one of the closest works to ours. This paper describes a framework for modeling

[2]In fact, an annotation in these proposals can be attached to an interval of frames, but the underlying semantics is the same as if the annotation was attached to each frame in that interval.

videos based on the concept of *frame stream*. As in AVIS and CVOT, annotations are attached to intervals of frames, but the underlying semantics is the same as if the annotations were directly attached to the frames. Temporal queries are formulated by using comparison operators on the lower and upper bounds of these intervals, and not through high-level algebraic operators as in our proposal. Several composition operators are studied, but the authors neglect the issue of propagating the annotations of the primitive videos when generating new ones through composition. This latter remark also applies to other similar works such as (Weiss et al., 1995).

In some respect, the work reported here is close to that of (Seshadri et al., 1996), which proposes an ADT-driven model for sequences and an associated query language. However, this latter work focuses on discretely-varying data such as financial time-series, and the authors do not discuss how it may be extended to deal with stepwisely varying data such as video annotations.

Conclusion

We have presented a model for video metadata which properly captures the orthogonality between logical structuration and annotation within a video, and we have investigated the impacts of this orthogonality regarding video querying and composition.

The concept of granularity plays a key role in our proposal. On the one hand, it is used to model the logical structuration of a video. On the other hand, it provides a foundation for expressing queries involving videos annotated at multiple levels of structuration.

The above model has been implemented as a prototype on top of the object-oriented DBMS O_2. In its current stage, this prototype is composed of a library of classes implementing the abstract datatypes discussed throughout the paper, and of four user interfaces : a schema definition and a query language preprocessor, a video editing user interface and a video player. (Dumas et al., 1999) provides details on this implementation.

The work developed in this paper illustrates how temporal granularity may be applied to design simple yet powerful video metadata models. We are convinced that other similar experiences may be carried out using the concept of spatial granularity. Indeed, spatial granularities could be used to address several issues that arise when modeling the relationships within and among the the images composing a video.

References

Adali, S., Candan, K. S., Chen, S., Erol, K., and Subrahmanian, V. (1996). The Advanced Video Information System: data structures and query processing. *Multimedia Systems*, 4:172 – 186.

Cattell et al., R., editor (2000). *The Object Database Standard: ODMG 3.0*. Morgan Kaufmann.

Clifford, J. and Tuzhilin, A., editors (1995). *Proc. of the workshop on Recent Advances in Temporal Databases*, Zurich, Switzerland. Springer Verlag.

Dionisio, J. and Ca'rdenas, A. (1998). A unified data model for representing multimedia, time-line and simulation data. *IEEE Transactions on Knowledge and Data Engineering*, 10(5).

Dumas, M., Lozano, R., Fauvet, M.-C., Martin, H., and Scholl, P.-C. (1999). A sequence-based object-oriented model for video databases. Research Report RR 1024-I-LSR 12, LSR-IMAG, Grenoble (France).

Elmagarmid, A., Jiang, H., Abdelsalam, A., Joshi, A., and Ahmed, M. (1997). *Video Database Systems: Issues, Products and Applications*. Kluwer Academic Publisher.

Gandhi, M. and Robertson, E. (1994). A data model for audio-video data. In *Proc. of the 6th Int. Conference on Management of Data (CISMOD)*, Bangalore (India). McGraw-Hill.

Hjelsvold, R., Midtstraum, R., and Sandsta, O. (1995). A temporal foundation of video databases. In (Clifford and Tuzhilin, 1995).

Hobbs, J. R. (1985). Granularity. In *Proc. of the 9th International Joint Conference on Artificial Intelligence (IJCAI)*.

Li, J., Goralwalla, I., Ozsu, M., and Szafron, D. (1997). Modeling video temporal relationships in an object database management system. In *Proc. of IS&T/SPIE Int. Symposium on Electronic Imaging: Multimedia Computing and Networking*, pages 80 – 91, San Jose, CA (USA).

Lozano, R. and Martin, H. (1998). Querying virtual videos with path and temporal expressions. In *proc. of the ACM Symposium on Applied Computing*, Atlanta, Georgia (USA).

Nack, F. and Lindsay, A. T. (1999). Everything You Wanted to Know About MPEG-7. *IEEE MultiMedia Magazine*, 6(3–4):65–77.

Oomoto, E. and Tanaka, K. (1993). OVID : Design and implementation of a video-object database system. *IEEE Transactions on Knowledge and Data Engineering*, 5(4).

Seshadri, P., Livny, M., and Ramakrishnan, R. (1996). The design and implementation of a sequence database system. In *Proc. of the Int. Conference on Very Large Databases (VLDB)*, pages 99 – 110, Bombay, India.

Tansel, A. U., Clifford, J., Gadia, S., Jajodia, S., Segev, A., and Snodggrass, R., editors (1993). *Temporal Databases*. The Benjamins/Cummings Publishing Company.

Wang, X. S., Jajodia, S., and Subrahmanian, V. (1995). Temporal modules : an approach toward federated temporal databases. *Information Systems*, 82.

Weiss, R., Duda, A., and Gifford, D. (1995). Composition and search with a video algebra. *IEEE Multimedia*, 2(1):12 – 25.