University of Alberta

Modeling of Moving Objects in a Video Objectbase Management System

by

John Z. Li, M. Tamer Özsu, Duane Szafron
Laboratory for Database Systems Research
Department of Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2H1
{zhong,ozsu,duane}@cs.ualberta.ca

Technical Report TR 96–12
April 1996

DEPARTMENT OF COMPUTING SCIENCE
The University of Alberta
Edmonton, Alberta, Canada

Modeling of Moving Objects in a Video Objectbase Management System *

April 1996

Abstract

Modeling moving objects has become a topic of increasing interest in the area of video databases. Two key aspects of such modeling are object spatial and temporal relationships. In this paper we introduce an innovative way to represent the trajectory of single moving object and the relative spatio-temporal relations between multiple moving objects. The representation supports a rich set of spatial topological and directional relations. It also supports both quantitative and qualitative user queries about moving objects. Algorithms for matching trajectories and spatio-temporal relations of moving objects are designed to facilitate query processing. These algorithms can handle both exact and similarity matches. We also discuss the integration of our moving object model, based on a video model, in an object-oriented system. Some query examples are provided to further validate the expressiveness of our model.

Keywords: multimedia, temporal, spatial, moving object, database, video.

*This research is supported by a grant from the Canadian Institute for Telecommunications Research (CITR) under the Network of Centre of Excellence (NCE) program of the Government of Canada.

Contents

List of Figures

List of Tables

1 Introduction

In the last few years, there has been significant research in modeling video systems [LG91, OT93, LG93, WDG94, SW94, GBT94, DDI+95, LGÖS96, TK96]. While there has been some research on spatial issues, the major focus has been on temporal aspect. There are only a few that have explored both spatial and temporal relationships. The most striking difference between still images and videos stems from movements and variations. Retrieving moving objects, which requires both spatial and temporal knowledge of objects, is part of content-based querying. Typical applications are automated surveillance systems, industrial monitoring, road traffic monitoring, video databases etc. Modeling moving objects has received some research attention recently [DG94, IB95, SAG95, ABL95] and it is certainly in its infancy. Most research in this area focuses on tracking the movement of single object, i.e., the *trajectory* of an object over a period of time, which is certainly very important. For example, object trajectories are needed for many useful video annotation tasks such as describing city street intersections, sporting events, pedestrian mall traffic, cell movements from quantitative fluorescence microscopy, groups of animals and meteorological objects [IB95]. However, another important aspect of moving objects is their relative spatial relationships.

To the best of our knowledge, no research has studied the relationships between moving objects in video databases. We believe that such support is essential for a video database because queries exploit these relationships. For example, a coach may have particular interest in the relative movement of his players during a game so that they can achieve the best cooperation. Alternatively, we may be interested in finding objects whose movements match user drawings in the case that it is difficult to verbally describe complex movement relations.

We use the Common Video Object Tree (CVOT) model [LGÖS96] to build an abstract model of a multimedia information system. The CVOT model is integrated into a powerful temporal object model to provide concrete objectbase management system (OBMS)[1] support for video data. The

[1] We prefer the terms "objectbase" and "objectbase management system" over the more popular terms "object-oriented database" and "object-oriented database management system", since the objects that are managed include

system that we use in this work is TIGUKAT[2] [ÖPS+95], which is an experimental system under development at the University of Alberta. Actually, any OBMS providing object-oriented support can be used. The major contributions of this paper are as follows. A new way of qualitatively representing the trajectory of a moving object and the relative spatio-temporal relations between moving objects is introduced. Such a representation is based on Allen's temporal interval algebra [All83], and it supports a rich set of spatial topological and directional relations. Algorithms for matching trajectories and spatio-temporal relations of moving objects are designed to facilitate query processing. The algorithms can handle both exact and similarity matches. A novel approach to integrating this moving object model with a video model in an OBMS is presented. The resulting system supports a broad range of user queries, especially for systems with graphical user interfaces.

The rest of the paper is organized as follows. Section 2 reviews the related work in object spatial representations and modeling of spatio-temporal semantics. Section 3 introduces our representation of object spatial properties and relationships. The new model, capturing the trajectories of moving objects and relative spatial relationships between moving objects, is presented and matching algorithms are discussed. Section 4 describes a video model which captures common objects in videos. An integration of the moving object model into an OBMS is also presented. Section 5 shows the expressiveness of our spatio-temporal representation by query examples. Section 6 summarizes our work and points out possible future work.

2 Related Work

Egenhofer [EF91] has specified eight fundamental topological relations that can hold between two planar regions. These relations are computed using four intersections over the *boundary* and *interior* of pointsets between two regions embedded in a two-dimensional space. These four intersections

code as well as data.

[2]TIGUKAT (tee-goo-kat) is a term in the language of Canadian Inuit people meaning "objects." The Canadian Inuits (Eskimos) are native to Canada with an ancestry originating in the Arctic regions.

result in eight topological relations: *disjoint, contains, inside, meet, equal, covers, covered_by*, and *overlap*. In a later paper [EAT92], Egenhofer studies gradual changes of these topological relations over time within the context of geographical information systems (GISs). It has been recognized that a qualitative change occurs if the deformation of an object affects its topological relation with respect to another object. A computational model is presented to describe the changes. Most importantly it reveals the internal relationships between topological relations which are useful in describing the closeness of topological relations. Our work is, in a sense, an extension of [EAT92] by considering directional relations as well as topological relations and time.

The Video Semantic Directed Graph (VSDG) model is a graph-based conceptual video model [DDI+95]. One feature of the VSDG model is an unbiased representation that provides a reference framework for constructing a semantically heterogeneous user's view of video data. The spatio-temporal semantics is captured by *conceptual spatial objects* and *conceptual temporal objects*. This model is able to capture some actions, such as walking and basketball slum-dunks. Although this model uses Allen's temporal interval algebra to model spatial relations among objects, its definitions of spatial relations are both incomplete and unsound.

Dimitrova and Golshani [DG94] describe a method of computing the trajectories of objects. Their objective was to discover motion using a dual hierarchy consisting of spatial and temporal parts for *video sequence* representation. Video sequences are identified by objects present in the scene and their respective motion. Motion vectors extracted during the motion compensation phase of video encoding is used as a coarse level optical flow. The motion information extraction is then used in the intermediate level by motion tracing and in the high level by associating an object and a set of trajectories with recognizable activities and events. They focus on trajectories of objects at a high level and do not study relations between moving objects.

Intille and Bobick propose an interesting model that uses a *closed-world assumption* to track object motions [IB95]. A closed-world is a region of space and time in which the specific context is adequate to determine all possible objects present in that region. Besides using closed-worlds to circumscribe the knowledge relevant to tracking, they also exploit them to reduce complexity. Two types of entities exist in a closed-world, *objects* and *image regions*. An important feature of this model is that the knowledge of the domain dictates how objects can interact and is independent of how the scene is captured for vision analysis. After an image region within a video frame is selected, each pixel within this region is assigned to one of the objects within its closed-world. Context-specific features are used to construct templates for tracking each moving object in the closed-world. Then, objects are tracked to the next frame using the templates. They describe a prototype for tracking football players. However, a major drawback of this model is its lacking of generality and its heavy dependence on domain knowledge.

A unified model for spatial and temporal information is proposed in [Wor94]. A *bitemporal relation*, including both event time and database time, is applied to objects in the system. This model is designated for GISs. There are also some research on the moving objects because of camera motions [ABL95, SAG95], such as *booming, tilting, panning, zooming* etc. We do not consider such kind movement in this paper. More work on motion detection can be found in [BH94, GD95].

3 Spatial Properties of Salient Objects

A *salient object* is an interesting physical object in a video frame. Each frame usually has many salient objects, e.g. persons, houses, cars, etc. In this section, we first describe the spatial representation of salient objects and briefly introduce Allen's temporal interval algebra. We then provide complete definitions of spatial directional and topological relations. Based on these definitions, we introduce the moving object model and matching algorithms. We use the term objects to refer to salient objects whenever this will not cause confusion.

3.1 Spatial Representations

It is a common strategy in spatial access methods to store object approximations and use these approximations to index the data space in order to efficiently retrieve the potential objects that satisfy the result of a query [PTSE95]. Depending on the application domain, there are several options in choosing object approximations. Minimum Bounding Rectangles (MBRs) have been used extensively to approximate objects because they need only two points for their representation. While MBRs demonstrate some disadvantages when approximating non-convex or diagonal objects, they are the most commonly used approximations in spatial applications. Hence, we use MBRs to represent objects in our system. We also assume there is always a finite set (possibly empty) of salient objects for a given video.

Definition 1 The *bounding box* of a salient object A_i is defined by its MBR (X_i, Y_i) and a depth D_i where $X_i = [x_{s_i}, x_{f_i}], Y_i = [y_{s_i}, y_{f_i}]$. x_{s_i} and x_{f_i} are A_i's projection on the X axis with $x_{s_i} \leq x_{f_i}$ and similarly for y_{s_i} and y_{f_i}. The two intervals are represented by A_{ix} and A_{iy} respectively. D_i is the depth of A_i in a three dimensional (3D) space. The *spatial property* of a salient object A_i is defined by a quadruple (X_i, Y_i, D_i, C_i) where $X_i = A_{ix}, Y_i = A_{iy}$ and C_i is the centroid of A_i. The centroid is represented by a three dimensional point (x_i, y_i, z_i). This can be naturally extended by considering time dimension: $(X_i^t, Y_i^t, D_i^t, C_i^t)$ captures the spatial property of a salient object A_i at time t.

Basically, the spatial property of an object is described by its bounding box and a representative point, called the centroid or mass point. In video modeling we must also consider the time dimension, as the spatial property of an object may change over time. For example, suppose the spatial property of A_i is $(X_i^{t_1}, Y_i^{t_1}, D_i^{t_1}, C_i^{t_1})$ at time t_1 and it becomes $(X_i^{t_2}, Y_i^{t_2}, D_i^{t_2}, C_i^{t_2})$ at time t_2. The displacement of A_i over time interval $I = [t_s, t_f]$ is $\mathit{DISP}(A_i, I) \equiv \sqrt{(x_i^{t_s} - x_i^{t_f})^2 + (y_i^{t_s} - y_i^{t_f})^2 + (z_i^{t_s} - z_i^{t_f})^2}$ which is the movement of the centroid of A_i. Also the Euclidean distance between two objects A_i and A_j at time t_k is $\mathit{DIST}(A_i, A_j, t_k) \equiv \sqrt{(x_i^{t_k} - x_j^{t_k})^2 + (y_i^{t_k} - y_j^{t_k})^2 + (z_i^{t_k} - z_j^{t_k})^2}$ which is also characterized by the centroid of A_i and A_j. Our goal is to support both quantitative and qualitative spatial retrieval.

3.2 Spatial Relationships

Spatial qualitative relations between objects are very important in multimedia objectbases because they implicitly support *fuzzy queries* which are captured by similarity matching or qualitative reasoning. Allen [All83] gives a temporal interval algebra (Table 1) for representing and reasoning about temporal relations between events represented as intervals. The temporal interval algebra essentially consists of the topological relations in one dimensional space, enhanced by the distinction of the order of the space. The order is used to capture the directional aspects in addition to the topological relations. Since the starting points of an interval are scale values, we can define an *ordering* directly over a list of intervals.

Definition 2 Let $I = \langle [t_{s_1}, t_{f_1}], [t_{s_2}, t_{f_2}], \ldots, [t_{s_n}, t_{f_n}] \rangle$ be a finite list of intervals. I is *ordered* if and only if $t_{s_i} \leq t_{s_{i+1}}$ $(\forall i \;\; 1 \leq i \leq n-1)$.

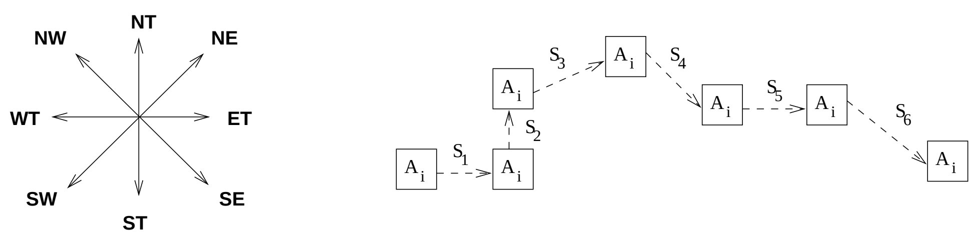
We consider 12 directional relations in our model and classify them into the following three categories: *strict directional relations* (north, south, west, and east), *mixed directional relations* (northeast, southeast, northwest, and southwest), and *positional relations* (above, below, left, and right). The definitions of these relations in terms of Allen's temporal algebra are given in Table 2. The symbols $\wedge$ and $\vee$ are the standard logical *AND* and *OR* operators, respectively. A short notation $\{\}$ is used to substitute the $\vee$ operator over interval relations. For example $A_{ix}\ \{\mathtt{b}, \mathtt{m}, \mathtt{o}\}\ A_{jx}$ is equivalent to $A_{ix}\, \mathtt{b}\, A_{jx} \vee A_{ix}\, \mathtt{m}\, A_{jx} \vee A_{ix}\, \mathtt{o}\, A_{jx}$. Detailed description of these definitions can be found in [LÖS96]. To simplify our description, we consider only 2D space. In 3D space, the depth of an object has to be considered and the extension is straightforward.

3.3 Modeling Moving Objects

A *moving object* is a salient object which moves its position over time. We assume moving objects are rigid or consisting of rigid parts connected together and these rigid parts are never disintegrated. For any moving object we consider eight possible directions shown in Figure 1(a).

Relation	Symbol	Inverse	Meaning
B before C	`b`	`bi`	BBB CCC
B meets C	`m`	`mi`	BBBCCC
B overlaps C	`o`	`oi`	BBB CCC
B during C	`d`	`di`	BBB CCCCC
B starts C	`s`	`si`	BBB CCCCC
B finishes C	`f`	`fi`	BBB CCCCC
B equal C	`e`	`e`	BBB CCC

Table 1: 13 Temporal Interval Relations



(a) Eight Moving Directions (b) Trajectory of Object A_i

Figure 1:

Definition 3 Let A_i be a moving object and the *motion* of A_i over time interval I_i is (S_i, d_i, I_i) where $S_i = \mathit{DISP}(A_i, I_i)$ is the displacement of A_i and d_i is a direction whose domain is the union of strict and mixed directional relations. For a given ordered list of time intervals $\langle I_1, I_2, \ldots, I_n \rangle$, the *trajectory* of A_i can be described by a list of motions

$$\langle (S_1, d_1, I_1), (S_2, d_2, I_2), \ldots, (S_n, d_n, I_n) \rangle.$$

Example 1 Figure 1(b) shows a trajectory of object A_i. The trajectory consists of a sequence of motions which can be expressed by $\langle (S_1, \texttt{ET}, I_1), (S_2, \texttt{NT}, I_2), (S_3, \texttt{NE}, I_3), (S_4, \texttt{SE}, I_4), (S_5, \texttt{ET}, I_5), (S_6, \texttt{SE}, I_6) \rangle$

Relation	Meaning	Definition
$A_i\,\mathtt{ST}\,A_j$	South	$A_{ix}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{e}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{b},\mathtt{m}\}\,A_{jy}$
$A_i\,\mathtt{NT}\,A_j$	North	$A_{ix}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{e}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{bi},\mathtt{mi}\}\,A_{jy}$
$A_i\,\mathtt{WT}\,A_j$	West	$A_{ix}\,\{\mathtt{b},\mathtt{m}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{e}\}\,A_{jy}$
$A_i\,\mathtt{ET}\,A_j$	East	$A_{ix}\,\{\mathtt{bi},\mathtt{mi}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{e}\}\,A_{jy}$
$A_i\,\mathtt{NW}\,A_j$	Northwest	$(A_{ix}\,\{\mathtt{b},\mathtt{m}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{bi},\mathtt{mi},\mathtt{oi}\}\,A_{jy}) \vee (A_{ix}\,\{\mathtt{o}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{bi},\mathtt{mi}\}\,A_{jy})$
$A_i\,\mathtt{NE}\,A_j$	Northeast	$(A_{ix}\,\{\mathtt{bi},\mathtt{mi}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{bi},\mathtt{mi},\mathtt{oi}\}\,A_{jy}) \vee (A_{ix}\,\{\mathtt{oi}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{bi},\mathtt{mi}\}\,A_{jy})$
$A_i\,\mathtt{SW}\,A_j$	Southwest	$(A_{ix}\,\{\mathtt{b},\mathtt{m}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{b},\mathtt{m},\mathtt{o}\}\,A_{jy}) \vee (A_{ix}\,\{\mathtt{o}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{b},\mathtt{m}\}\,A_{jy})$
$A_i\,\mathtt{SE}\,A_j$	Southeast	$(A_{ix}\,\{\mathtt{b},\mathtt{m}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{b},\mathtt{m},\mathtt{o}\}\,A_{jy}) \vee (A_{ix}\,\{\mathtt{oi}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{b},\mathtt{m}\}\,A_{jy})$
$A_i\,\mathtt{LT}\,A_j$	Left	$A_{ix}\,\{\mathtt{b},\mathtt{m}\}\,A_{jx}$
$A_i\,\mathtt{RT}\,A_j$	Right	$A_{ix}\,\{\mathtt{bi},\mathtt{mi}\}\,A_{jx}$
$A_i\,\mathtt{BL}\,A_j$	Below	$A_{iy}\,\{\mathtt{b},\mathtt{m}\}\,A_{jy}$
$A_i\,\mathtt{AB}\,A_j$	Above	$A_{iy}\,\{\mathtt{bi},\mathtt{mi}\}\,A_{jy}$
$A_i\,\mathtt{EQ}\,A_j$	Equal	$A_{ix}\,\{\mathtt{e}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{e}\}\,A_{jy}$
$A_i\,\mathtt{IN}\,A_j$	Inside	$A_{ix}\,\{\mathtt{d}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{d}\}\,A_{jy}$
$A_i\,\mathtt{CT}\,A_j$	Contain	$A_{ix}\,\{\mathtt{di}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{di}\}\,A_{jy}$
$A_i\,\mathtt{CV}\,A_j$	Cover	$(A_{ix}\,\{\mathtt{di}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{fi},\mathtt{si},\mathtt{e}\}\,A_{jy}) \vee (A_{ix}\,\{\mathtt{e}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{di},\mathtt{fi},\mathtt{si}\}\,A_{jy}) \vee$ $(A_{ix}\,\{\mathtt{fi},\mathtt{si}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{di},\mathtt{fi},\mathtt{si},\mathtt{e}\}\,A_{jy})$
$A_i\,\mathtt{CB}\,A_j$	Covered By	$(A_{ix}\,\{\mathtt{d}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{f},\mathtt{s},\mathtt{e}\}\,A_{jy}) \vee (A_{ix}\,\{\mathtt{e}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{d},\mathtt{f},\mathtt{s}\}\,A_{jy}) \vee$ $(A_{ix}\,\{\mathtt{f},\mathtt{s}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{d},\mathtt{f},\mathtt{s},\mathtt{e}\}\,A_{jy})$
$A_i\,\mathtt{OL}\,A_j$	Overlap	$A_{ix}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{o},\mathtt{oi},\mathtt{e}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{o},\mathtt{oi},\mathtt{e}\}\,A_{jy}$
$A_i\,\mathtt{TC}\,A_j$	Touch	$(A_{ix}\,\{\mathtt{m},\mathtt{mi}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{o},\mathtt{oi},\mathtt{m},\mathtt{mi},\mathtt{e}\}\,A_{jy}) \vee$ $(A_{ix}\,\{\mathtt{d},\mathtt{di},\mathtt{s},\mathtt{si},\mathtt{f},\mathtt{fi},\mathtt{o},\mathtt{oi},\mathtt{m},\mathtt{mi},\mathtt{e}\}\,A_{jx} \wedge A_{iy}\,\{\mathtt{m},\mathtt{mi}\}\,A_{jy})$
$A_i\,\mathtt{DJ}\,A_j$	Disjoint	$A_{ix}\,\{\mathtt{b},\mathtt{bi}\}\,A_{jx} \vee A_{iy}\,\{\mathtt{b},\mathtt{bi}\}\,A_{jy}$

Table 2: Directional and Topological Relation Definitions

Definition 4 Let A_i and A_j be two moving objects. Their *moving spatio-temporal relationship* (abbreviated as *mst-relation*) during time interval I_k is A_i (α, β, I_k) A_j, such that $A_i \alpha A_j$ and $A_i \beta A_j$ at time interval I_k where α is any topological relations and β is either one of the 12 directional relations or NULL which means no directional relation. For a given ordered list of time intervals $\langle I_1, I_2, \ldots, I_n \rangle$, all the mst-relations of A_i and A_j are defined by an *mst-list*:

$$\langle (\alpha_1, \beta_1, I_1), (\alpha_2, \beta_2, I_2), \ldots, (\alpha_n, \beta_n, I_n) \rangle.$$

From the definition, we can see that the spatial relationships over a time period between moving objects are captured by both their topological and directional relations. NULL is necessary because two objects may have no any directional relation, such as when A_i is *inside* of A_j. Following example further explains the concept of an mst-relation and an mst-list. It also indicates that the mst-relations of two objects is neither unique nor symmetric.

Example 2 Figure 2 shows moving objects approaching together, overlapping, and then going away from each other, which might represent two friends meeting on a street, shaking hands, hugging, and then leaving each other. During time interval I_1, the mst-relation between A_i and A_j is $(\texttt{DJ}, \texttt{WT}, I_1)$. This relation changes to A_i $(\texttt{TC}, \texttt{WT}, I_3)$ A_j at interval I_3 and becomes A_i $(\texttt{TC}, \texttt{ET}, I_5)$ A_j at interval I_5. The mst-relation between A_j and A_i at interval I_3 is $(\texttt{TC}, \texttt{ET}, I_3)$ which is different from A_i $(\texttt{TC}, \texttt{WT}, I_3)$ A_j. Hence, generally $A_i \theta A_j \neq A_j \theta A_i$ where θ is an mst-relation. Such an asymmetric property is caused by the directional relations. However, from the inverse property of all the directional relations, we can always derive the mst-relation between A_j and A_i from the mst-relation between A_i and A_j. The mst-list of A_i and A_j over ordered time interval $\langle I_1, I_2, \ldots, I_6 \rangle$ is $\langle (\texttt{DJ}, \texttt{WT}, I_1), (\texttt{DJ}, \texttt{WT}, I_2), (\texttt{TC}, \texttt{WT}, I_3), (\texttt{OL}, \texttt{NULL}, I_4), (\texttt{TC}, \texttt{ET}, I_5), (\texttt{DJ}, \texttt{ET}, I_6) \rangle$. Since, $A_i \, \texttt{WT} \, A_j$ can deduce $A_i \, \texttt{LT} \, A_j$ and $A_i \, \texttt{ET} \, A_j$ can deduce $A_i \, \texttt{RT} \, A_j$ according to the definitions of spatial directional relations, we can have another mst-list for A_i and A_j:
$\langle (\texttt{DJ}, \texttt{LT}, I_1), (\texttt{DJ}, \texttt{LT}, I_2), (\texttt{TC}, \texttt{LT}, I_3), (\texttt{OL}, \texttt{NULL}, I_4), (\texttt{TC}, \texttt{RT}, I_5), (\texttt{DJ}, \texttt{RT}, I_6) \rangle$.

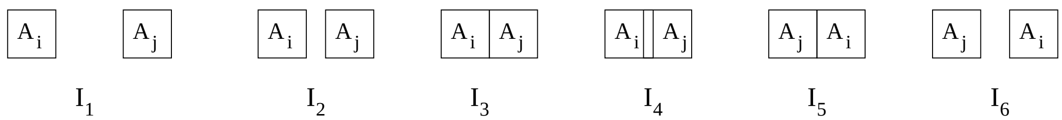
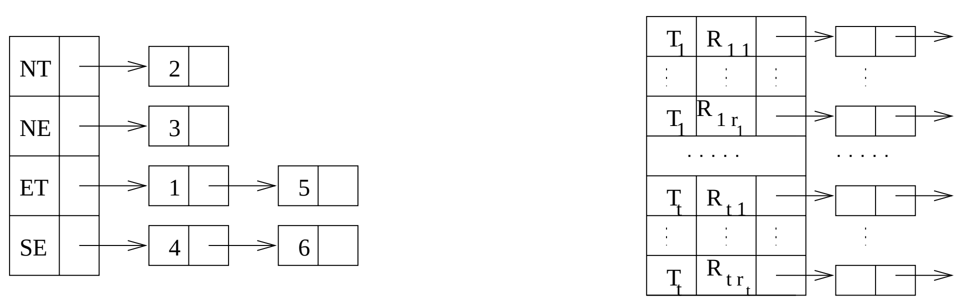


Figure 2: Two Moving Objects

3.4 Matching Moving Objects

To find matching trajectories and mst-lists is useful in handing user queries. In this subsection we introduce two algorithms to accomplish such a task. When a video OBMS being queried, a user may ask "*Is there any object whose trajectory matches trajectory of object* A?". A's trajectory, denoted by $\langle M_1, M_2, \ldots, M_m \rangle$, can be given through a graphical user interface. Therefore, we need a systematic way to find any matching object, which satisfies the above condition, within the system. The problem can be restated slightly differently: Does the trajectory of A match the trajectory $\langle N_1, N_2, \ldots, N_n \rangle$ of object B? To facilitate the retrieval of a particular motion in a trajectory, a set of linked lists are used to represent the trajectory of B. Each list starts corresponds to a directional relation and each entry consists of an integer and a pointer to the next element. The integer represents the relative order of a particular displacement in the trajectory. For example, the first entry in Figure 3(a) indicates that the object is displaced in `NT` direction as the second move in its trajectory. Figure 3(a) shows a linked list representation for the trajectory of Figure 1(b).



(a) Ai's Trajectory Link List

(b) Link List for an MST-set

Figure 3: Data Structures for Trajectory and MST-list

Similarly we can have a linked list representation for an mst-list of two moving objects. The only change we need is to accommodate directional relations. The data structure of the linked list for mst-lists is shown in Figure 3(b). The first column T_t is a topological relation with $1 \leq t \leq 8$ and the second column R_{tr_i} is a directional relation or `NULL` with $1 \leq r_i \leq 13$ $(1 \leq i \leq t)$.

We are now in a position to introduce our algorithms for matching objects' trajectories. Figure 4 describes an algorithm to test whether object A's trajectory matches object B's trajectory. The structure *linklist* is exactly the same structure as in Figure 3(a). Statement (5) indicates that if the number of motions of A is bigger than the number of motions of B, then A does not match B. Although in this case B may match A, we consider this to be a different case. Statements (6) and (7) look for B's first motion which is exactly the same as A's first motion. If there is no such match, FALSE is returned. Otherwise, the proper head pointer of B's linked list is located and from there a sequential comparison is conducted within the trajectories of A and B.

A similar algorithm for mst-lists is described in Figure 5. The only change is the data structure of the linked list. Here, we have to consider the topological relations. The data values are captured by TOPID (topological) and DIRID (directional) in the algorithm. This change results in some related changes in mst-list comparisons.

These two algorithms are exact match algorithms. It is well-known that exact match queries *normally* generate few results in multimedia systems. Therefore, fuzzy (similarity) queries must be supported. A query is *fuzzy* if the properties of objects being queried are not precisely defined (like *big region*) or the comparison operators in the query cannot provide exact matches. Again let us consider object trajectories first. In the case of object displacement, the similarity of two displacements can be measured by a predefined *tolerance*. However, in the case of directional relations a new measurement metric has to be introduced to describe the similarity. Our approach to this problem is to manually assign a *distance value* between any two directional relations as shown in Table 3. The way these values are assigned is completely determined by their *closeness* to each other. For example, *northwest* and *northeast* should have the same closeness value to direction *north*. They should be *closer* to the *north* than *west*, *east*, and *south* are. The smallest value of a distance is zero which is an exact match and the the biggest value of a distance is four which is the

```
TrajectoryMatch(A, B)
INPUT: A = {M_1, M_2, ..., M_m}: object A's trajectory
       B = {N_1, N_2, ..., N_n}: object B's trajectory
OUTPUT: TRUE /* A and B's trajectories match */
        FALSE /* A and B's trajectories do not match */
    int i, k = 1;
    struct linklist { int ID; struct linklist *next; }
        DIR[8]; /* The linked list of B's trajectory */
    struct linklist x; /* temporal variable */
    if (m > n) return FALSE;
    while (DIR[k] ≠ empty AND M_1 ≠ DIR[k].ID) k++;
    if (DIR[k] == empty) return FALSE;
    x = DIR[k].next;
    while (x ≠ empty) {
        i = 1;
        while (i ≤ m AND (i + x.ID)< n) {
            if (M_i ≠ x.ID) break;
                i++;
        } /* end of inner while */
        if (i > m) return TRUE;
        x = x.next;
    } /* end of outer while */
    return FALSE;
```

Figure 4: Trajectory Match Algorithm

```
MstMatch(A, B)
INPUT: A = {M_1, M_2, ..., M_m}: object A's trajectory
       B = {N_1, N_2, ..., N_n}: object B's trajectory
OUTPUT: TRUE /* A and B's trajectories match */
        FALSE /* A and B's trajectories do not match */
    int i, k = 1;
    struct linklist { int TOPID, DIRID; struct linklist *next; }
        TOPDIR[8 * 13]; /* The linked list of B's trajectory */
    struct linklist x; /* temporal variable */
    if (m > n) return FALSE;
    while (TOPDIR[k] ≠ empty ∧ (M_1 ≠ TOPDIR[k].TOPID ∨ M_1 ≠ TOPDIR[k].DIRID))
        k++;
    if (TOPDIR[k] == empty) return FALSE;
    x = TOPDIR[k].next;
    while (x ≠ empty) {
        i = 1;
        while (i ≤ m ∧ (i + x.TOPID) < n) {
            if (M_i ≠ x.TOPID ∨ M_i ≠ x.DIRID) break;
                i++;
        } /* end of inner while */
        if (i > m) return TRUE;
        x = x.next;
    } /* end of outer while */
    return FALSE;
```

Figure 5: MST-Relation Match Algorithm

opposite direction. For example, the distance of north (`NT`) versus south (`ST`) is 4.

	NT	NW	NE	WT	SW	ET	SE	ST
NT	0	1	1	2	3	2	3	4
NW	1	0	2	1	2	3	4	3
NE	1	2	0	3	4	1	2	3
WT	2	1	3	0	1	4	3	2
SW	3	2	4	1	0	3	2	1
ET	2	3	1	4	3	0	1	2
SE	3	4	2	3	2	1	0	1
ST	4	3	3	2	1	2	1	0

Table 3: Distances of Moving Directions

Definition 5 Let $\{M_1, M_2, \ldots, M_m\}$ $(m \geq 1)$ be the trajectory of A, $\{N_1, N_2, \ldots, N_n\}$ be the trajectory of B, and $m \leq n$. $minDiff(A, B)$ is the smallest distance between A and B calculated as follows:

$$minDiff(A, B) \;=\; MIN\ \{\sum_{i=1}^{m} distance(M_i, N_{i+j})\} \;\; (\forall j\ 0 \leq j \leq n-i).$$

The biggest difference between A and B happens only if A's moving direction is always opposite to B's moving directions in all the comparisons. Such a case can be quantified by $maxDiff(A, B) = 4*m$ since the maximum number of comparing motions is m. A normalized similarity function for the trajectory of A and B is defined as

$$TrajSim(A, B) = \frac{maxDiff(A, B) - minDiff(A, B)}{maxDiff(A, B)}.$$

Function $TrajSim(A, B)$ defines a similarity degree of the trajectories of A and B and its domain is $[0, 1]$. For example, in the case $minDiff(A, B) = 0$, we have $TrajSim(A, B) = 1$ which indicates an exact match. In the case $minDiff(A, B) = maxDiff(A, B)$, we have $TrajSim(A, B) = 0$ which indicates that A always moves in the opposite direction as B does. So it shows the similarity between A and B's trajectories is minimum.

The similarity function of two mst-lists can be in a similar manner. The directional relations must be extended to include positional relations and `NULL`, as presented in Table 4. The distance values of positional relations are assigned in the same manner as we did for other directional relations. The distance values between `NULL` and directional relations are assigned as an average distance among others, which is 2. The problem is complicated in assigning distance values for topological relations. A distance scheme of topological relations is presented in [EAT92] which we adopt as Table 5. Hence, we can compute the similarity function for mst-lists using a function analogous to $TrajSim$.

	NT	NW	NE	WT	SW	ET	SE	ST	LT	RT	AB	BL	NULL
NT	0	1	1	2	3	2	3	4	3	3	1	4	2
NW	1	0	2	1	2	3	4	3	2	4	2	4	2
NE	1	2	0	3	4	1	2	3	4	2	2	4	2
WT	2	1	3	0	1	4	3	2	1	4	3	3	2
SW	3	2	4	1	0	3	2	1	2	4	4	2	2
ET	2	3	1	4	3	0	1	2	4	1	3	3	2
SE	3	4	2	3	2	1	0	1	4	2	4	2	2
ST	4	3	3	2	1	2	1	0	3	3	4	1	2
LT	3	2	4	1	2	4	4	3	0	4	2	2	2
RT	3	4	2	4	4	1	2	3	4	0	2	2	2
AB	1	2	2	3	4	3	4	4	2	2	0	4	2
BL	4	4	4	3	2	3	2	1	2	2	4	0	2
NULL	2	2	2	2	2	2	2	2	2	2	2	2	0

Table 4: Distances of Directional Relations

Definition 6 Let $\{M_1, M_2, \ldots, M_m\}$ $(m \geq 1)$ be the mst-list of A, $\{N_1, N_2, \ldots, N_n\}$ be the mst-list of B, and $m \leq n$. $minDiff(A, B)$ is the smallest distance between A and B:

$$minDiff(A, B) \;=\; MIN\ \{\sum_{i=1}^{m} distance(M_i, N_{i+j})\} \quad (\forall j\ 0 \leq j \leq n - i)$$

where $distance(M_i, N_{i+j}) = distance(\alpha(M_i, N_{i+j})) + distance(\beta(M_i, N_{i+j}))$ where α and β extract the topological relation and the directional relation of an mst-relation respectively. $maxDiff(A, B) =$

	DJ	TC	EQ	IN	CB	CT	CV	OL
DJ	0	1	6	4	5	4	5	4
TC	1	0	5	5	4	5	4	3
EQ	6	5	0	4	3	4	3	6
IN	4	5	4	0	1	6	7	4
CB	5	4	3	1	0	7	6	3
CT	4	5	4	6	7	0	1	4
CV	5	4	3	7	6	1	0	3
OL	4	3	6	4	3	4	3	0

Table 5: Distances of Topological Relations (Table 1 in [EAT92])

$4*m+7*m$ since the maximum distance value of a topological relations is 7. An mst-list similarity function is defined as

$$MstSim(A,B) = \frac{maxDiff(A,B) - minDiff(A,B)}{maxDiff(A,B)}.$$

We do not consider the time intervals during the match because it is less important than others. If there is some interest in knowing the time length of a motion, the extension to the algorithms is straightforward.

4 Video Modeling

Video modeling is the process of translating raw video data into an efficient internal representation which helps to capture video semantics. The procedural process of extracting video semantics from a video is called *video segmentation*. There are two approaches to video segmentation in an object-oriented context: *stream-based* and *structured*. In a stream-based approach, a clip is considered as a sequence of *frames* displayed at a specified rate. In a structured approach, a clip is considered as a sequence of scenes. Each approach has its own advantages and disadvantages as described in [Gha96]. Very little work has been done on the structured approach because of its technical difficulties, However, the stream-based approach has received considerable research attention because of its technical feasibility. We concentrate here on stream-based approaches. In this section we briefly introduce the Common Video Object Tree (CVOT) model (a video model) and its integration into a temporal OBMS. Then, we show how our moving object model is integrated into the resulting OBMS.

4.1 The Common Video Object Tree Model

There are several different ways to segment a video into clips, two of which are *fixed time intervals* and *shots*. A *fixed time interval* segmentation approach divides a video into equal length clips using a predefined time interval (e.g. 2 seconds) while a *shot* is a set of continuous frames captured by a single camera action [HJW95]. Two common problems with existing models are restrictive video segmentation and poor user query support. The CVOT model [LGÖS96] is primarily designed to deal with these two problems. In this model, there is no restriction on how videos are segmented. Without loss of generality, we assume that any given video stream has a finite number of clips and any clip has a finite number of frames, as shown in Figure 6. One unique feature of the CVOT model is that a clip overlap is allowed. This can provide considerable benefit in modeling *events* which will be discussed in Section 4.3. Generally, a smooth transition of one event to another event, *event fading*, requires having some scene or activity overlap between the end of the previous event and the start of the next event. Such a transition phase is usually reflected in a few frames as shown in Figure 6.

The main purpose of the CVOT model is to find all the common objects among clips and to group clips according to these objects. A tree structure is used to represent such a clip group. The *time interval* of a clip is defined according to the clip's starting frame and ending frame.

Example 3 Figure 7 shows a video in which John and Mary walk toward their house. Later, Mary rides a horse on a ranch with her colt and dog. Let us assume that the salient objects are $SO = \{\mathsf{john}, \mathsf{mary}, \mathsf{house}, \mathsf{tree}, \mathsf{horse}, \mathsf{colt}, \mathsf{dog}\}$. If the video is segmented as in Figure 7, then we

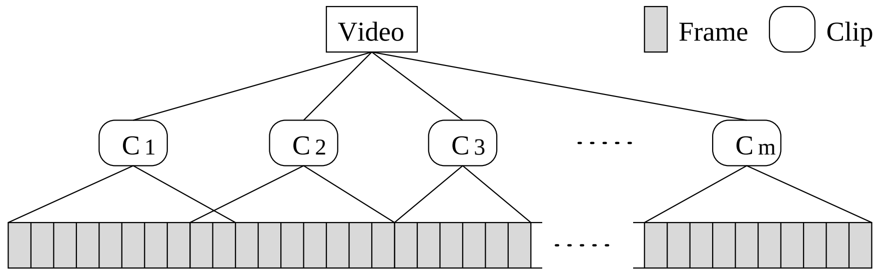


Figure 6: Stream-based Video Clips and Frames

have five clips $C = \{C_1, C_2, C_3, C_4, C_5\}$ with john, mary, house, and tree in C_1, john, house, and tree in C_2, mary, horse, colt, and dog in C_3, mary, horse, and colt in C_4, and mary, horse, colt, and dog in C_5.

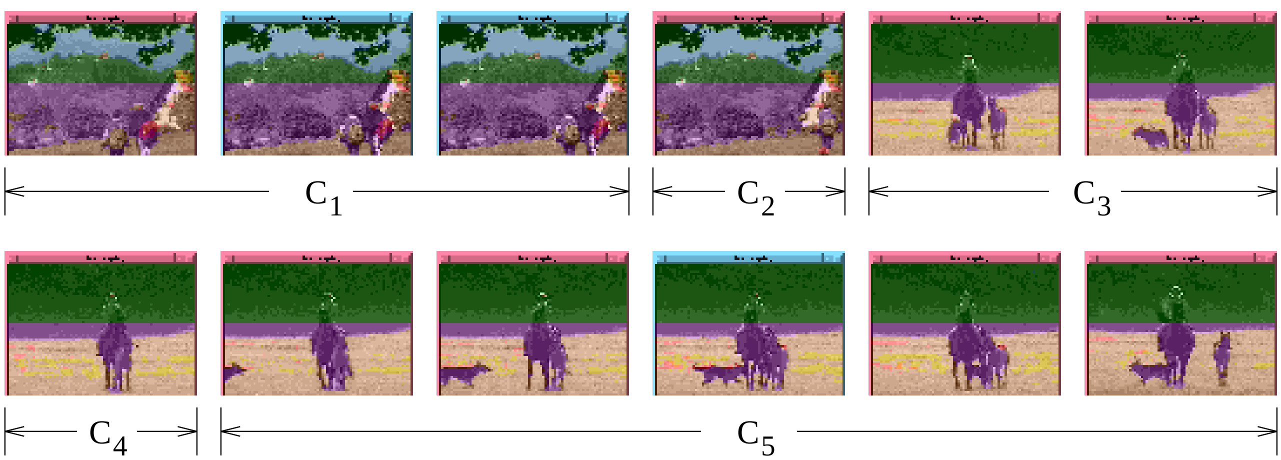


Figure 7: Salient Objects and Clips

Figure 8 shows a CVOT instance for Figure 7. In Figure 8, node C_1 has time interval $[1,3]$ and a set of salient objects $\{\mathsf{john}, \mathsf{mary}, \mathsf{house}, \mathsf{tree}\}$; node C_2 has time interval $[4,4]$ and a set of salient objects $\{\mathsf{john}, \mathsf{house}, \mathsf{tree}\}$; node C_3 has time interval $[5,6]$ and a set of salient objects $\{\mathsf{mary}, \mathsf{horse}, \mathsf{colt}, \mathsf{dog}\}$; node C_4 has time interval $[7,7]$ and a set of salient objects $\{\mathsf{mary}, \mathsf{horse}, \mathsf{colt}\}$; node C_5 has time interval $[8,12]$ and a set of salient objects $\{\mathsf{mary}, \mathsf{horse}, \mathsf{colt}, \mathsf{dog}\}$. There are 3 common objects between C_1 and C_2 and this number is reduced to 0 if C_3 is added. Therefore, C_1

and C_2 have a parent node N_1 with a time interval $[1, 4]$ and a salient object set $\{\mathsf{john}, \mathsf{house}, \mathsf{tree}\}$. There are 3 common objects between C_3 and C_4 and this number is not reduced if C_5 is added. Therefore, C_3, C_4, and C_5 have a parent node N_2 with time interval $[5, 12]$ and a set of salient objects $\{\mathsf{mary}, \mathsf{horse}, \mathsf{colt}\}$. As there is no common object between N_1 and N_2, the *Root* node has time interval $[1, 12]$ with an empty salient object set. The CVOT model directly supports queries of the type "*Find all the clips in which a salient object appears*" and "*How long does a particular salient object occur in a video*".

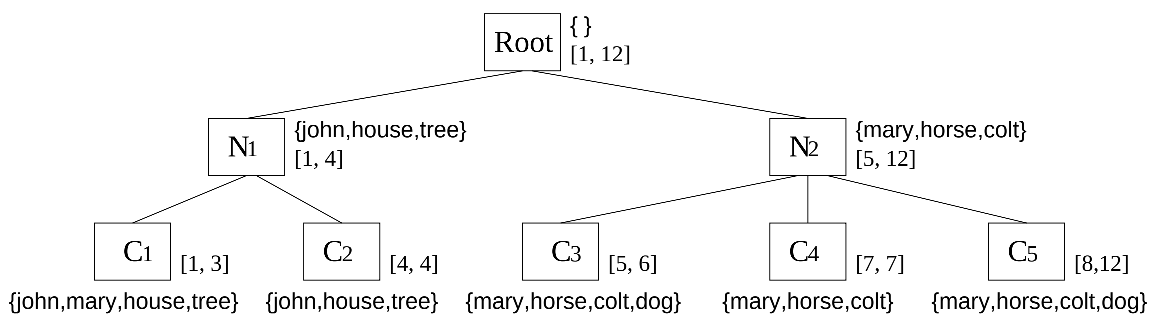


Figure 8: A Common Video Object Tree Built from Figure 3

4.2 The OBMS Support

CVOT is an abstract model. To have proper database management support for continuous media, this model needs to be integrated into a data model. We work within the framework of a uniform, behavioral object model such as the one supported by the TIGUKAT system [ÖPS+95]. The important characteristics of the model, from the perspective of this paper, are its *behaviorality* and and its *uniformity*. The model is *behavioral* in the sense that all access and manipulation of objects is based on the application of behaviors to objects. The model is *uniform* in that every component of information, including its semantics, is modeled as a *first-class object* with well-defined behavior. The typical object-oriented features, such as strong object identity, abstract types, strong typing, complex objects, full encapsulation, multiple inheritance, and parametric types are also supported.

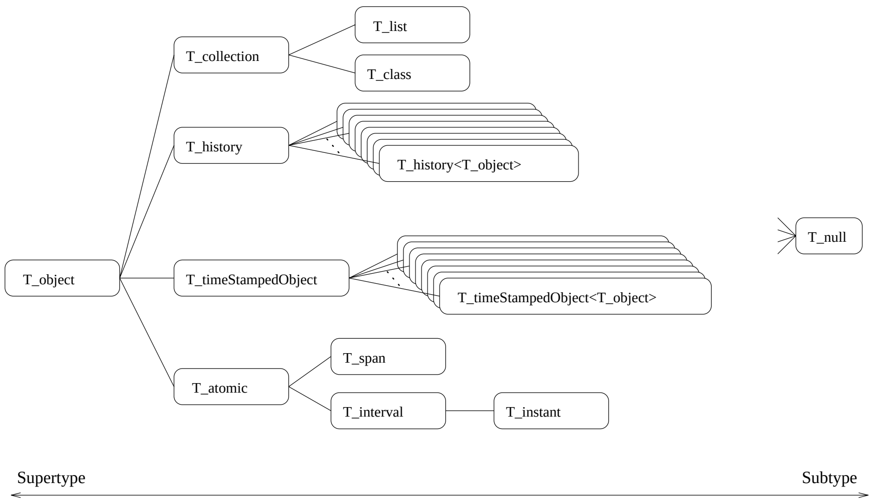


Figure 9: The Basic Time Type Hierarchy

The primitive objects of the model include: *atomic entities* (reals, integers, strings, etc.); *types* for defining common features of objects; *behaviors* for specifying the semantics of operations that may be performed on objects; *functions* for specifying implementations of behaviors over types; *classes* for automatic classification of objects based on type; and *collections* for supporting general heterogeneous groupings of objects. In this paper, a reference prefixed by "`T_`" refers to a type, "*C_*" to a class, "*B_*" to a behavior, and "`T_X`< `T_Y` >" to the type `T_X` parameterized by the type `T_Y`. For example, `T_person` refers to a type, *C_person* to its class, *B_age* to one of its behaviors and `T_collection`< `T_person` > to the type of collections of persons. A reference such as David, without a prefix, denotes some other application specific reference. Consequently, the model separates the definition of object characteristics (a *type*) from the mechanism for maintaining instances of a particular type (a *class*). The primitive type system is a complete lattice with the `T_object` type as the root of the lattice and the `T_null` type as the base.

Temporality has been added to this model [GLÖS96] as type and behavior extensions of the type system discussed above. Figure 9 gives part of the time type hierarchy that includes the temporal ontology and temporal history features of the temporal model. Unary operators which return the lower bound, upper bound, and length of the time interval are defined. The model supports a rich set of ordering operations among intervals, e.g., *before*, *overlaps*, *during*, etc. (see Table 1) as well as set-theoretic operations, viz. *union*, *intersection* and *difference*[3]. A time duration can be added or subtracted from a time interval to return another time interval. A time interval can be expanded or shrunk by a specified time duration.

A *time instant* (*moment*, *chronon*, etc.) is a specific anchored moment in time. A time instant can be compared with a time interval to check if it falls before, within, or after the time interval. A *time span* is an unanchored relative duration of time; it is basically an atomic cardinal quantity, independent of any time instant or time interval. One requirement of a temporal model is an ability to adequately represent and manage histories of objects and real-world events. Our model represents the temporal histories of objects whose type, is T_X as objects of the T_history<T_X> type as shown in Figure 9. A temporal history consists of objects and their associated timestamps (time intervals or time instants). A *timestamped object* knows its timestamp and its associated object (value) at (during) the timestamp. A temporal history is made up of such objects. Table 6 gives the behaviors defined on histories and timestamped objects. Behavior *B_history* defined on T_history<T_X> returns the set (collection) of all timestamped objects that comprise the history. Another behavior defined on history objects, *B_insert*, timestamps and inserts an object into the history. The *B_validObjects* behavior allows the user to get the objects in the history that were valid at (during) the given time.

T_history<T_X>	*B_history*:	T_collection<T_timeStampedObject<T_X>>
	B_insert:	T_X,T_interval → T_boolean
	B_validObjects:	T_interval → T_collection<T_timeStampedObject<T_X>>
T_timeStampedObject<T_X>	*B_value*:	T_X
	B_timeStamp:	T_interval

Table 6: Behaviors on Histories and Time-stamped Objects

[3]Note that the union of two disjoint intervals is not an interval. Similarly, for the difference operation, if the second interval is contained in the first, the result is not an interval. In the temporal model, these cases are handled by returning an object of the *null* type (T_null).

Each timestamped object is an instance of the `T_timeStampedObject<T_X>` type. This type represents objects and their corresponding timestamps. Behaviors *B_value* and *B_timeStamp*, defined on `T_timeStampedObject`, return the value and the timestamp of a timestamped object, respectively.

4.3 Integrating Moving Objects

Figure 10 shows our video type system. `T_discrete` defines all discrete value types and all the subtypes of `T_discrete` here are enumerated types. The types that are in a grey shade will be discussed and detailed description of the rest can be found in [LGÖS96]. For the completeness of the discussion we list all the behaviors of `T_video`, `T_clip`, and `T_frame` in Table 7 without giving any explanation.

`T_video`	*B_clips*:	`T_history<T_clip>`
	B_cvotTree:	`T_tree`
	B_search:	`T_salientObject, T_tree → T_tree`
	B_length:	`T_span`
	B_publisher:	`T_collection<T_company>`
	B_producer:	`T_collection<T_person>`
	B_date:	`T_instant`
	B_play:	`T_boolean`
`T_clip`	*B_frames*:	`T_history< T_frame >`
	B_salientObjects:	`T_collection<T_history<T_salientObject>>`
	B_events:	`T_collection<T_history<T_event>>`
`T_frame`	*B_location*:	`T_instant`
	B_format:	`T_videoFormat`
	B_content:	`T_image`

Table 7: Behavior Signatures of Videos, Clips, and Frames

The semantics or contents of a video are usually expressed by its *features* which include video attributes and the relationships between these attributes. Typical video features are salient objects and *events*. We focus on salient objects, more specifically moving salient objects. Since objects can appear multiple times in a clip or a video, we model the history of an object as a timestamped object of type `T_history< T_salientObject >`. The behavior *B_salientObjects* of `T_clip` returns all the objects within a clip. Using histories to model objects enables us to uniformly capture the

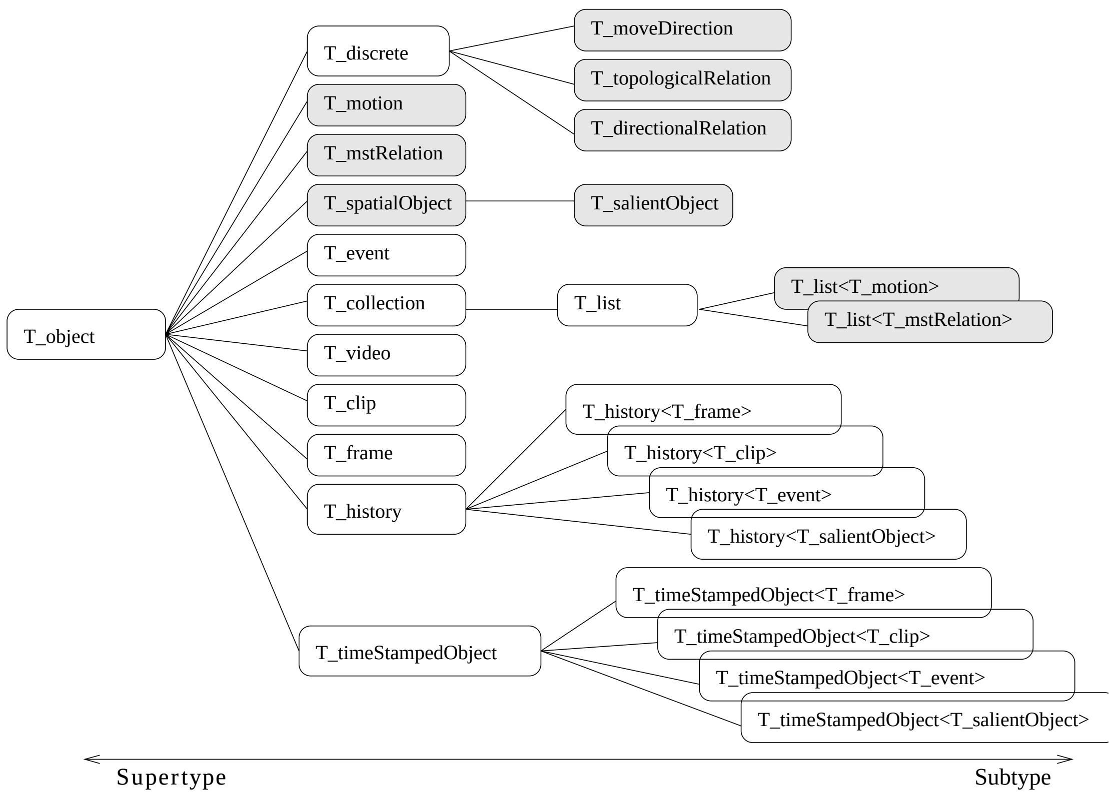


Figure 10: The Video Type System

temporal semantics of video data because a video is modeled as a history of clips and a clip is modeled as a history of frames.

Any object occupying some space is an instance of `T_spatialObject`. Table 8 also shows the behavior signatures of spatial objects. The behaviors *B_xinterval* and *B_yinterval* of type `T_spatialObject` define an object's 2D intervals and are computed from the projections of the object's MBR over x and y axes. We suppose the depth (*B_depth*) of the object is a real number. The behavior *B_centroid* returns the centroid of the object while the behavior *B_area* returns the region occupied by the object. The distance between objects at a certain time and the displacement of an object over time intervals are captured by *B_distance* and *B_displacement*.

In type `T_salientObject`, a subtype of `T_spatialObject`, the behavior *B_inClips* returns all the clips in which the object appears. *B_trajectory* of `T_salientObject` returns type `T_trajectory`

T_spatialObject	*B_xinterval*:	T_interval
	B_yinterval:	T_interval
	B_depth:	T_real
	B_centroid:	T_point
	B_area:	T_real
	B_displacement:	T_interval, T_interval $\rightarrow$ T_real
	B_distance:	T_spatialObject, T_interval $\rightarrow$ T_real
	B_south:	T_spatialObject $\rightarrow$ T_boolean
	B_north:	T_spatialObject $\rightarrow$ T_boolean
	B_west:	T_spatialObject $\rightarrow$ T_boolean
	B_east:	T_spatialObject $\rightarrow$ T_boolean
	B_northwest:	T_spatialObject $\rightarrow$ T_boolean
	B_northeast:	T_spatialObject $\rightarrow$ T_boolean
	B_southwest:	T_spatialObject $\rightarrow$ T_boolean
	B_southeast:	T_spatialObject $\rightarrow$ T_boolean
	B_left:	T_spatialObject $\rightarrow$ T_boolean
	B_right:	T_spatialObject $\rightarrow$ T_boolean
	B_below:	T_spatialObject $\rightarrow$ T_boolean
	B_above:	T_spatialObject $\rightarrow$ T_boolean
	B_equal:	T_spatialObject $\rightarrow$ T_boolean
	B_inside:	T_spatialObject $\rightarrow$ T_boolean
	B_contain:	T_spatialObject $\rightarrow$ T_boolean
	B_overlap:	T_spatialObject $\rightarrow$ T_boolean
	B_cover:	T_spatialObject $\rightarrow$ T_boolean
	B_coveredBy:	T_spatialObject $\rightarrow$ T_boolean
	B_touch:	T_spatialObject $\rightarrow$ T_boolean
	B_disjoint:	T_spatialObject $\rightarrow$ T_boolean
T_salientObject	*B_inClips*:	T_video $\rightarrow$ T_history$<$ T_clip $>$
	B_trajectory:	T_trajectory
	B_mstSet:	T_salientObject $\rightarrow$ T_mstSet
T_trajectory	*B_exactMatch*:	T_trajectory $\rightarrow$ T_boolean
	B_simMatch:	T_trajectory $\rightarrow$ T_real
	B_subtrajectory:	T_interval $\rightarrow$ T_trajectory
T_mstSet	*B_exactMatch*:	T_mstSet $\rightarrow$ T_boolean
	B_simMatch:	T_mstSet $\rightarrow$ T_real
	B_submstSet:	T_interval $\rightarrow$ T_mstSet
T_motion	*B_displacement*:	T_interval $\rightarrow$ T_real
	B_moveDirection:	T_moveDirection
T_mstRelation	*B_topology*:	T_topologicalRelation
	B_direction:	T_directionalRelation
	B_interval:	T_interval

Table 8: Primitive Behavior Signatures of Spatial and Salient Objects

which is a list of moving object's motions (`T_list` < `T_motion` >). A *list* is an ordered collection. Similarly, the behavior *B_mstSet* returns type `T_mstSet` which is a list of two moving objects' mst-relations (`T_list<T_mstRelation>`). *B_exactMatch* is the exact match algorithm (for either trajectories or mst-lists) described in Figure 4 and Figure 5. *B_simMatch* of `T_trajectory` returns the similarity degree of two trajectories, which is captured by the similarity function $trajSim(A, B)$. The returned value is a real number between 0 and 1. The behavior *B_simMatch* of `T_mstSets` is the similarity function $mstSim(A, B)$ for two moving objects' mst-list. *B_subtrajectory* and *B_submstSet* returns part of a trajectory and part of an mst-list, respectively, for a given time interval. `T_motion` describes one motion of a moving object while `T_mstRelation` describes one mst-relation of two moving objects. `T_moveDirection`, `T_topologicalRelation`, and `T_directionalRelation` are enumerate types and they represent the eight moving directions, the eight topological relations, and the twelve directional relations plus `NULL` respectively.

Example 4 Let mary and dog be two timestamped salient objects. Their spatial relations at time t (or frame t) can be decided by first binding mary and dog to a common time interval. That is, we assume t is a time interval `t` (whose starting time and ending times are t) and both `t`.*B_during*(mary.*B_timeStamp*) and `t`.*B_during*(dog.*B_timeStamp*) are true. Then we compare the spatial intervals of mary and dog according to the definitions given in Table 2 to check what topological and directional relations exist. These spatial intervals of mary can be extracted by mary.*B_value*.*B_xinterval* and mary.*B_value*.*B_yinterval*. Similarly, we have dog.*B_value*.*B_xinterval* and dog.*B_value*.*B_yinterval* for the spatial intervals of dog. The trajectory of dog is expressed by dog.*B_trajectory*. The mst-list of dog and mary is captured by dog.*B_mstSet*(mary).

5 Query Examples

In this subsection we present some examples to show the expressiveness of our model from the spatial properties point of view. We first introduce object calculus [Pet94]. The alphabet of the calculus consists of object constants (a, b, c, d), object variables (o, p, q, u, v, x, y, z), monadic predicates (C, P, Q), dyadic predicates $(=, \in, \notin)$, an n-ary predicate $(Eval)$, a function symbol (β) called *behavior specification* (Bspec), and logical connectives $(\exists, \forall, \wedge, \vee, \neg)$. The "evaluation" of a Bspec is accomplished by predicate $Eval$. A *term* is an object constant, an object variable or a Bspec. An *atomic formula* or *atom* has an equivalent Bspec representation. From atoms, *well-formed formulas* (WFFs) are built to construct the declarative calculus expressions of the language. WFFs are defined recursively from atoms in the usual way using the connectives $\wedge, \vee, \neg$ and the quantifiers $\exists$ and $\forall$.

A query is an object calculus expression of the form $\{t_1, \ldots, t_n | \phi(o_1, \ldots, o_n)\}$ where $t_1, \ldots, t_n$ are the terms over the multiple variables $o_1, \ldots, o_n$. ϕ is a WFF. Indexed object variables are of the form $o[\beta]$ where β is a set of behaviors defined on the type variable o. The semantics of this construct is to project over the behaviors in β for o, meaning that after the operation, only the behaviors given in β will be applicable to o. We assume that all the queries are posted to a particular video instance myVideo and salient objects and events are timestamped objects as discussed in Section 4.

Query 1 Is the salient object a in the clip c?

$$\{\mathsf{q} \mid \exists \mathsf{x} \exists \mathsf{y}(C\_history(x) \wedge C\_timeStampedObject(y) \wedge \mathsf{x} \in \mathsf{c}.B\_salientObjects \wedge$$
$$\mathsf{y} \in \mathsf{x}.B\_history \wedge \mathsf{q} = \mathsf{a}.B\_equal(\mathsf{y}.B\_value)\}$$

This query checks clip c through *B_salientObjects* which returns a collection of all the histories of salient objects in c. If any object in this collection is equal to (*B_equal*) a, then a boolean value true will be returned. Otherwise, value false is returned. For convenience, predicate $IN(\mathsf{o}, \mathsf{c})$ is used to denote that object o is in clip c.

Query 2 In clip c find all the objects which have a similar trajectory as shown in Figure 1(b) denoted by myTraj.

$$\{\mathsf{q} \mid \forall \mathsf{x} \forall \mathsf{y}(C\_real(r) \wedge \mathsf{x} \in \mathsf{c}.B\_salientObjects \wedge \mathsf{y} \in \mathsf{x}.B\_history \wedge$$
$$\mathsf{q} = \mathsf{y}.B\_value \wedge IN(\mathsf{p}, objc) \wedge \mathsf{q}.B\_trajectory.B\_simMatch(\mathsf{myTraj}).B\_greaterThan(\mathsf{r}))\}$$

For each object q in clip c, the trajectory of q is checked against mytraj. Such a comparison is determined by the similarity matching function between this two trajectories. r is a predefined (or user-provided) threshold valued between $[0, 1]$ for qualifying a match.

Query 3 Find a clip in which object a is at left of object b and later the two exchange their positions.

$$\{\mathsf{c} \mid \exists \mathsf{x} \exists \mathsf{x}_2 \exists \mathsf{x}_3 \exists \mathsf{y} \exists \mathsf{y}_2 \exists \mathsf{y}_3 (\mathit{C\_history}(x) \wedge \mathit{C\_history}(y) \wedge \mathsf{x}, \mathsf{y} \in \mathsf{c}.\mathit{B\_salientObjects} \wedge$$
$$\mathsf{x}_2, \mathsf{x}_3 \in \mathsf{x}.\mathit{B\_history} \wedge \mathsf{y}_2, \mathsf{y}_3 \in \mathsf{y}.\mathit{B\_history} \wedge \mathsf{x}_2.\mathit{B\_value} = \mathsf{a} \wedge \mathsf{y}_2.\mathit{B\_value} = \mathsf{b} \wedge$$
$$\mathsf{x}_2.\mathit{B\_timeStamp}.\mathit{B\_equal}(\mathsf{y}_2.\mathit{B\_timeStamp}) \wedge \mathsf{x}_2.\mathit{B\_value}.\mathit{B\_left}(\mathsf{y}_2.\mathit{B\_value}) \wedge$$
$$\mathsf{x}_3.\mathit{B\_value} = \mathsf{a} \wedge \mathsf{y}_3.\mathit{B\_value} = \mathsf{b} \wedge \mathsf{x}_3.\mathit{B\_timeStamp}.\mathit{B\_equal}(\mathsf{y}_3.\mathit{B\_timeStamp}) \wedge$$
$$\mathsf{y}_3.\mathit{B\_value}.\mathit{B\_left}(\mathsf{x}_3.\mathit{B\_value}) \wedge \mathsf{x}_3.\mathit{B\_timeStamp}.\mathit{B\_after}(\mathsf{x}_2.\mathit{B\_timeStamp}))\}.$$

Suppose clip c is the one we are looking for. Then there must be two objects, denoted by x_2 and y_2 respectively, in c's salient object set so that x_2 is a and y_2 is b. Similarly, two other objects, denoted by x_3 and y_3 respectively, must exist in c's salient object set so that x_3 is a and y_3 is b. The difference between x_2 and x_3 is only in their time stamps. Here we require that x_3 appears later than x_2 ($\mathsf{x}_3.\mathit{B\_timeStamp}.\mathit{B\_after}(\mathsf{x}_2.\mathit{B\_timeStamp})$). Therefore, if x_2 is at the left of y_2 at time $\mathsf{x}_2.\mathit{B\_timeStamp}$ and y_3 is at the left of x_3 at time $\mathsf{x}_3.\mathit{B\_timeStamp}$, we are sure that a and b have exchanged their directional positions. Such a query might be expressed by Figure 2 in Example 2 within a graphical user interface. Let $\mathsf{myMstSet}$ represent this mst-list, we could simplify the query as

$$\{\mathsf{c} \mid \mathsf{a}.\mathit{B\_mstSet}(objb).\mathit{B\_exactMatch}(\mathsf{myMstSet}) \wedge \mathit{IN}(\mathsf{a}, \mathsf{c}) \wedge \mathit{IN}(\mathsf{b}, \mathsf{c})\}$$

or

$$\{\mathsf{c} \mid \mathit{C\_real}(r) \wedge \mathsf{a}.\mathit{B\_mstSet}(objb).\mathit{B\_simMatch}(\mathsf{myMstSet}).\mathit{B\_greaterThan}(\mathsf{r}) \wedge \mathit{IN}(\mathsf{a}, \mathsf{c}) \wedge \mathit{IN}(\mathsf{b}, \mathsf{c})\}$$

if we want to use the similarity function and r is a threshold value less than 1.

Query 4 Find a video clip in which a dog approaches Mary from the left.

$$\{\mathsf{c} \mid \exists \mathsf{x} \exists \mathsf{x}_2 \exists \mathsf{x}_3 \exists \mathsf{y} \exists \mathsf{y}_2 \exists \mathsf{y}_3 (\mathit{C\_history}(\mathsf{x}) \wedge \mathit{C\_history}(\mathsf{y}) \wedge \mathit{C\_real}(\mathsf{h}_1) \wedge \mathit{C\_real}(\mathsf{h}_2) \wedge$$
$$\mathsf{x}, \mathsf{y} \in \mathsf{c}.\mathit{B\_salientObjects} \wedge \mathsf{x}_2, \mathsf{x}_3 \in \mathsf{x}.\mathit{B\_history} \wedge \mathsf{y}_2, \mathsf{y}_3 \in \mathsf{y}.\mathit{B\_history} \wedge$$
$$\mathsf{x}_2.\mathit{B\_value} = \mathsf{dog} \wedge \mathsf{y}_2.\mathit{B\_value} = \mathsf{mary} \wedge \mathsf{x}_2.\mathit{B\_timeStamp}.\mathit{B\_equal}(\mathsf{y}_2.\mathit{B\_timeStamp}) \wedge$$

$\mathsf{x}_2.B\_value.B\_left(\mathsf{y}_2.B\_value) \wedge \mathsf{x}_3.B\_value = \mathsf{a} \wedge \mathsf{y}_3.B\_value = \mathsf{b} \wedge$

$\mathsf{x}_3.B\_timeStamp.B\_equal(\mathsf{y}_3.B\_timeStamp) \wedge \mathsf{x}_3.B\_value.B\_left(\mathsf{y}_3.B\_value) \wedge$

$\mathsf{x}_3.B\_timeStamp.B\_after(\mathsf{x}_2.B\_timeStamp) \wedge$

$\mathsf{x}_2.B\_value.B\_displacement(\mathsf{x}_2.B\_timeStamp, \mathsf{x}_3.B\_timeStamp).B\_greaterThan(\mathsf{h}_1) \wedge$

$\mathsf{y}_2.B\_value.B\_displacement(\mathsf{x}_2.B\_timeStamp, \mathsf{x}_3.B\_timeStamp).B\_lessThan(\mathsf{h}_2))\}$

where dog and mary are two instances of `T_salientObject`. As with Query 6 we suppose clip c is what we are looking for and two salient objects, denoted by x_2 and x_3, are introduced to represent dog and to reflect different time stamps. The same strategy is used for the object mary. Then, we compute the dog's displacement over the time period and enforce this displacement to be greater than some predefined value h_1 to insure enough movement achieved. Furthermore, the displacement of mary is also computed and is required to be less than a predefined value h_2. This particular requirement of mary is to guarantee that it is the dog approaching Mary from the left, instead of Mary approaching the dog from the right.

This query can also be expressed in an mst-list described in Figure 11 and we denote such an mst-list as dogMaryMstSet. Then, the query is

$\{\mathsf{c} \mid C\_real(r) \wedge \mathsf{dog}.B\_mstSet(objmary).B\_simMatch(\mathsf{dogMaryMstSet}).B\_greaterThan(\mathsf{r}) \wedge$

$IN(\mathsf{dog}, \mathsf{c}) \wedge IN(\mathsf{mary}, \mathsf{c})\}$

However, this mst-list does not distinguish whether it is dog approaching mary from the left or it is mary approaching dog from the right. Further constraint must be put into this expression. One way to solve this problem is to make sure that mary's displacement changes very little over the time interval as we did before.

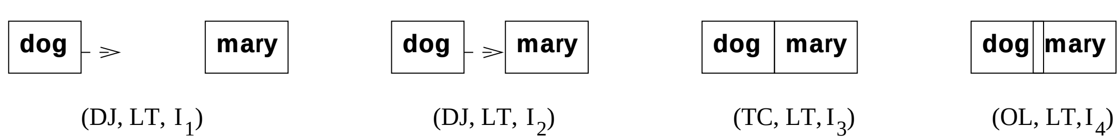


Figure 11: dog Approaches mary from Left

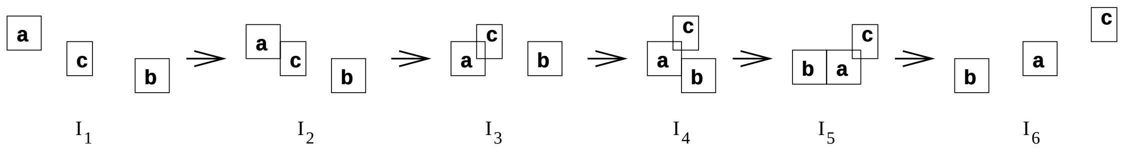


Figure 12: A Scene of Objects a, b, and c

Query 5 Find any clip which matches a scene described in Figure 12.

$$\{\mathsf{q} \mid C\_clip(q) \wedge \mathsf{a}.B\_mstSet(objb).B\_exactMatch(\mathsf{abMstSet}) \wedge \mathsf{a}.B\_mstSet(objc).B\_exactMatch(\mathsf{acMstSet}) \wedge \mathsf{b}.B\_mstSet(objc).B\_exactMatch(\mathsf{bcMstSet}) \wedge IN(\mathsf{a}, \mathsf{q}) \wedge IN(\mathsf{b}, \mathsf{q}) \wedge IN(\mathsf{c}, \mathsf{q})\}.$$

Here, abMstSet, acMstSet, and bcMstSet are the mst-lists between a and b, a and b, and b and c. Furthermore, abMstSet $= \{(\mathtt{DJ}, \mathtt{NW}, I_1), (\mathtt{DJ}, \mathtt{NW}, I_2), (\mathtt{DJ}, \mathtt{LT}, I_3), (\mathtt{TC}, \mathtt{NW}, I_4), (\mathtt{TC}, \mathtt{ET}, I_5), (\mathtt{DJ}, \mathtt{NE}, I_6)\}$, acMstSet $= \{(\mathtt{DJ}, \mathtt{NW}, I_1), (\mathtt{TC}, \mathtt{NW}, I_2), (\mathtt{OL}, \mathtt{NULL}, I_3), (\mathtt{OL}, \mathtt{NULL}, I_4), (\mathtt{OL}, \mathtt{NULL}, I_5), (\mathtt{DJ}, \mathtt{SW}, I_6)\}$, and

bcMstSet $= \{(\mathtt{DJ}, \mathtt{SE}, I_1), (\mathtt{DJ}, \mathtt{SE}, I_2), (\mathtt{DJ}, \mathtt{NE}, I_3), (\mathtt{DJ}, \mathtt{SE}, I_4), (\mathtt{DJ}, \mathtt{SW}, I_5), (\mathtt{DJ}, \mathtt{SW}, I_6)\}$. The scene of Figure 12 can be interpreted into a part of a basketball game: at time I_1, players a and b are trying to catch the ball c, but a is faster so he touches the ball first and grabs it; since b does not get the ball, he has to try to block a's advance at time I_3; then players a and b are collide with each other, but a is still holding the ball c; at time I_5 a manages to have passed b and shoots the ball finally. It is a very difficult query if it is expressed verbally. We see that the query has been greatly simplified using the concept of moving spatial-temporal sets.

6 Conclusions

The most striking difference between images and videos stems from movements and variations which involve both spatial and temporal knowledge of objects. Moving objects are very important feature of a multimedia OBMS. In this paper we concentrate on modeling video moving objects. In particular we present a way of representing the trajectory of a moving object and we are the first to propose a model for the relative spatio-temporal relations between moving objects. The proposed representation supports a rich set of spatial topological and directional relations and it captures not only quantitative properties of objects, but also qualitative properties of objects. Algorithms for matching trajectories and spatio-temporal relations of moving objects are designed to facilitate query processing. These algorithms can handle both exact and similarity matches. A novel approach to integrating such a moving object model into the CVOT model in an OBMS is presented and the expressiveness of such an integrated system is demonstrated by means of example queries within the context of the TIGUKAT system. We strongly believe that such a system, incorporated with a graphical user interfaces, can result in a powerful video retrieval system.

There are two major directions for our future work. One is to build a prototype based on this model and to gain insightful experience of it. Another one is to build a video query language based on the CVOT model. The spatial, temporal, and spatio-temporal queries can be translated into the query calculus and then the query algebra. Until that time it is possible to optimize these queries using object query optimization techniques [MDZ93, ÖB95].

References

[ABL95] G. Ahanger, D. Benson, and T. D. C. Little. Video query formulation. In *Proceedings of Storage and Retrieval for Images and Video Databases II, IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, San Jose, CA, February 1995.

[All83] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of ACM*, 26(11):832—843, 1983.

[BH94] H. Buxton and R. Howarth. Behavioural descriptions from image sequences. In *Proceedings of Workshop on Integration of Natural and Vision Processing Language*, August 1994.

[DDI+95] Y. F. Day, S. Dagtas, M. Iino, A Khokhar, and A. Ghafoor. Object-oriented conceptual modeling of video data. In *Proceedings of the 11th International Conference on Data Engineering*, pages 401—408, Taipei, Taiwan, 1995.

[DG94] N. Dimitrova and F. Golshani. Rx for semantic video database retrieval. In *Proceedings of the 2nd ACM International Conference on Multimedia*, pages 219—226, San Francisco, CA, October 1994.

[EAT92] M. Egenhofer and K. K. Al-Taha. Reasoning about gradual changes of topological relationships. In *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, pages 196—219. Springer Verlag, Septempber 1992.

[EF91] M. Egenhofer and R. Franzosa. Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(2):161—174, 1991.

[GBT94] S. Gibbs, C. Breiteneder, and D. Tscichritzis. Data modeling of time-based media. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 91—102, Minneapolis, Minnesota, May 1994.

[GD95] D. M. Gavrila and L. S. Davis. 3-D model-based tracking of human upper body movement: a multi-view approach. In *Proceedings of IEEE International Symposium on Computer Vision*, pages 253—258, November 1995.

[Gha96] S. Ghandeharizadeh. Stream-based versus structured video objects: Issues, solutions, and challenges. In V. S. Subrahmanian and S. Jajodia, editors, *Multimedia Database Systems: Issues and Research Directions*, pages 215—236. Springer Verlag, 1996.

[GLÖS96] I. A. Goralwalla, Y. Leontiev, M. T. Özsu, and D. Szafron. Modeling time: Back to basics. Technical Report TR-96-03, Department of Computing Science, University of Alberta, February 1996.

[HJW95] A. Hampapur, R. Jain, and T. E. Weymouth. Production model based digital video segmentation. *Multimedia Tools and Applications*, 1(1):9—46, March 1995.

[IB95] S. S. Intille and A. F. Bobick. Visual tracking using closed-worlds. In *Proceedings of International Conference on Computer Vision*, Cambridge, MA, June 1995.

[LG91] T. C. C. Little and A. Ghafoor. Spatio-temporal composition of distributed multimedia objects for value added networks. *Computer*, 24(10):42—50, 1991.

[LG93] T. C. C. Little and A. Ghafoor. Interval-based conceptual models for time-dependent multimedia data. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):551—563, August 1993.

[LGÖS96] J. Z. Li, I. Goralwalla, M. T. Özsu, and D. Szafron. Video modeling and its integration in a temporal object model. Technical Report TR-96-02, Department of Computing Science, University of Alberta, January 1996.

[LÖS96] J. Z. Li, M. T. Özsu, and D. Szafron. Integrating video spatial relationships into an object model. Technical Report TR-96-06, Department of Computing Science, University of Alberta, March 1996.

[MDZ93] G. Mitchell, U. Dayal, and S. B. Zdonik. Control of an extensible query optimizer: A planning-based approach. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 517—528, Dublin, Ireland, August 1993.

[ÖB95] M.T. Özsu and J. Blakeley. Query optimization and processing in object-oriented database systems. In W. Kim, editor, *Modern Database Systems*, pages 146—174. Addison-Wesley, 1995.

[ÖPS+95] M. T. Özsu, R. J. Peters, D. Szafron, B. Irani, A. Lipka, and A. Munoz. TIGUKAT: A uniform behavioral objectbase management system. *The VLDB Journal*, 4:100—147, 1995.

[OT93] E. Oomoto and K. Tanaka. OVID: Design and implementation of a video-object database system. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):629—643, August 1993.

[Pet94] R. Peters. TIGUKAT: A uniform behavioral objectbase management system. PhD thesis, Department of Computing Science, University of Alberta. Available as Technical Report TR-94-06, 1994.

[PTSE95] D. Papadias, Y. Theodoridis, T. Sellis, and M. J. Egenhofer. Topological relations in the world of minimum bounding rectangles: A study with R-trees. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 92—103, San Jose, CA, May 1995.

[SAG95] H. S. Sawhney, S. Ayer, and M. Gorkani. Model-based 2D&3D dominant motion estimation for mosaicing and video representation. In *Proceedings of International Conference on Computer Vision*, Cambridge, MA, June 1995.

[SW94] G. A. Schloss and M. J. Wynblatt. Building temporal structures in a layered multimedia data model. In *Proceedings of ACM Multimedia'94*, pages 271—278, San Francisco, CA, 1994.

[TK96] V. J. Tsotras and A. Kumar. Temporal database bibliography update. *ACM SIGMOD Records*, 25(1), March 1996.

[WDG94] R. Weiss, A. Duda, and D. K. Gifford. Content-based access to algebraic video. In *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pages 140—151, 1994.

[Wor94] M. F. Worboys. A unified model for spatial and temporal information. *The Computer Journal*, 37(1):26—34, 1994.