Spatio-Temporal Indexing for Large Multimedia Applications

Yannis Theodoridis
Computer Science Division
Dept. of Elec. and Comp. Engineering
National Tech. University of Athens
Zographou, Athens, 157 73 GREECE
e-mail: theodor@cs.ntua.gr

Michael Vazirgiannis
Computer Science Division
Dept. of Elec. And Comp. Engineering
National Tech. University of Athens
Zographou, Athens, 157 73 GREECE
e-mail: mvazirg@cs.ntua.gr

Timos Sellis
Computer Science Division
Dept. of Elec. and Comp. Engineering
National Tech. University of Athens
Zographou, Athens, 157 73 GREECE
e-mail: timos@cs.ntua.gr

Abstract

Multimedia applications usually involve a large number of multimedia objects (texts, images, sounds etc.). Spatial and temporal relationships among these objects should be efficiently supported and retrieved within a multimedia authoring tool. In this paper we present several spatial, temporal and spatio-temporal relationships of interest and propose efficient indexing schemes, based on multi-dimensional (spatial) data structures, for large multimedia applications that involve thousands of objects. Evaluation models of the proposed schemes are also presented as well as hints for the selection of the most appropriate one, according to the multimedia author's requirements.

1. Introduction

A Multimedia Application (MAP) involves a variety of individual multimedia objects presented according to the MAP scenario. The multimedia objects that participate in the MAP, called *actors*, are transformed either spatially or temporally in order to be presented according to author's requirements. Moreover, the author has to define the spatial and temporal ordering of actors within the application context and define the relationships among them. Finally, the way that the user will interact with the application as well as the way that the application will treat application or system events have to be defined.

It is evident that authoring complex MAPs that involve a large number of actors may be a very complicated task, keeping in mind the large set of possible events that may encounter in the application context and the number of actors as well as the various potential combinations of these parameters. Typically we would expect the number of actors and their relationships in the context of an MAP to be 10^4 as regards order of magnitude. Taking in account the vast number of possible events and their combinations as regards interaction, the amount of the entities that have to be managed by the MAP authors is considerable.

Thus the need for an indexing scheme that will support MAP authors to manage the large number of actors and spatio-temporal relationships among them is required. Current authoring tools do not provide such facilities. Sample requirements would include:

- which actors appear in the application at a specific time instance,
- what is the spatial layout (screen layout) at a specific time instance during the application,
- what is the temporal layout of the application, in terms of temporal intervals,
- what is the spatio-temporal relationship among a set of actors in the application (i.e. "does actor A spatially overlap with actor B in the application?" or "which actors temporally overlap with actor A?")

In this paper we propose indexing schemes for large multimedia applications in order to assist authors manage the large number of actors, have spatial and temporal layouts of parts or the entire application, answer queries regarding spatio-temporal relationships among actors. As regards multimedia application modelling we are based on previous research efforts [13, 14, 15].

The proposed indexing schemes are based on the R-tree index [7], which is widely used for indexing of spatial data in several applications, such as Geographic Information Systems (GIS), CAD and VLSI design, etc. We adapt R-trees in order to index either spatial, temporal or spatio-temporal occurrences of actors and relationships between them. Moreover, we evaluate the proposed schemes against the case of serial storage of actors and their spatio-temporal coordinates, giving also hints to multimedia database designers in order to select the most efficient scheme according to the requirements of MAP authors.

In the literature there is no previous work, according to our knowledge, on indexing spatio-temporal characteristics of actors (in the rest of the paper we will use the terms *actor* and *object* interchangeably). Research has mainly focused on *content-based* image indexing, i.e., fast retrieval of objects using their content characteristics (color, texture, shape). For example, in [5] a system, called *QBIC*, that couples several features from machine vision with fast indexing methods from the database area is proposed in order to support color, shape and texture matching queries. Nearest-neighbor queries (based on image content) are addressed in [3]. In general, indexing of actors' contents is an active research area while indexing of actors' extends in the spatio-temporal coordinate system sets a new direction.

2. Multimedia Application Modelling

MAP scenarios are classified into one of the following two categories: *pre-orchestrated* (pre-defined spatial and temporal ordering of actors) or *interactive* (the application flow is affected by events related to user, application or the system). In both cases the requirement for complete representation of the spatio-temporal ordering and relationships among actors arises. A model for scenarios should cover both pre-orchestrated and event-based cases. The general form of a scenario tuple (scenario fundamental unit) [14] should represent: the triggering event (simple or complex), stopping event, set of actions to be executed, constraints to be fulfilled for the scenario to start its execution.

In this paper we focus on the pre-orchestrated case, since event-based (or interactive) scenario cases include non-deterministic temporal and/or spatial occurrences of actors. Though this is a matter of our future research.

2.1. Spatio-Temporal Operators

As we have mentioned above, a crucial parameter for MAP development is the specification of the spatial and temporal presentation specifications as well as relationships among the participating actors. We support a set of operators that represent each possible spatio-temporal relationship between actors.

In the rest of the paper we are going to use the following spatial, temporal and spatio-temporal relationships, as typical queries in order to illustrate the proposed indexing schemes through these sample operators:

- spatial operators:
  - *overlap*(*p*,*q*): returns the list of objects *p* that spatially overlap object *q*.
  - *above*(*p*,*q*): returns the list of objects *p* that spatially lie above object *q*.
- temporal operators:
  - *during*(*p*,*q*): returns the list of objects *p* that are temporally included in the temporal interval that corresponds to the execution of object *q*.
  - *before*(*p*,*q*): returns the list of objects *p* that are temporally executed before object *q*.
- spatio-temporal operators:
  - *overlap_during*(*p*,*q*): returns the list of objects *p* that spatially overlap object *q* while its execution.

Similarly the following operators are defined: *overlap_before*(*p*,*q*), *above_during*(*p*,*q*), *above_before*(*p*,*q*)

We classify the above operators in two categories: *inclusive* (those that indicate spatial and temporal overlapping or inclusion among actors) and *exclusive* (those that indicate spatial or temporal disjointness among actors) ones. Three operators are classified as inclusive (*overlap*, *during*, *overlap_during*), while the rest ones are classified as exclusive. A general assumption is that queries that involve inclusive operators are highly selective (i.e., the result is a limited set of objects) while queries that involve exclusive operators indicate low selectivity (i.e., the result is a large set of objects). We will also define in the next subsection a sample MAP to use it as an example for illustration reasons as regards the indexing schemes that will be proposed in Section 3.

2.2. An Example Multimedia Application

In this subsection we describe an indicative MAP in terms of spatio-temporal relationships as defined above. The spatial layout of the application appears in Figure 1a while the temporal one appears in Figure 1b. The high level scenario of the application is the following:

"The application starts with video clip A (located at point 10,50 relatively to the application origin Θ). At the same time a narration E starts. 10 sec after the start of video A, image B appears to the right side (18 pts) and below the upper side of A (12 pts). The image B will be displayed for 7 sec. Just after the image B disappears, video A stops while a text window C appears 7 pts below (left aligned) the position of image B. Text window C will remain for 11 sec on the screen. 3 sec after C appears, a small image D appears inside C 8 pts above the bottom side of C, aligned to the right side. D will remain for 4 sec on the screen. Meanwhile, at the 10th sec of the application, a bitmap logo F appears at the bottom-left corner of the application window. F disappears after 3 sec. The application ends when narration E ends."

Typical queries submitted by the author of an application would be the following:

query 1:"which actors are executed during the presentation of actor D?" (only temporal relationship is involved)

query 2:"which actors spatially lie above actor D in the application window?" (only spatial relationship is involved)

query 3:"which actors spatially overlap with actor D during its presentation?" (spatio-temporal relationship is involved)

query 4:"what is the spatial layout of the screen on the 22nd sec of the application?"

query 5:"what is the temporal layout between the 10th and the 20th sec of the application?"

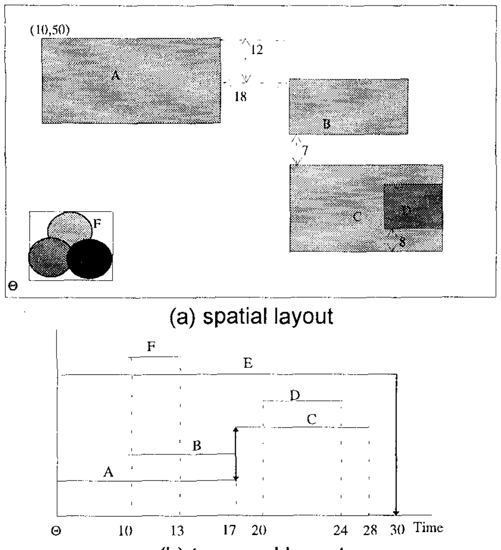


Figure 1: Spatial and temporal layout of a MAP

In the next section we propose efficient indexing mechanisms to support such queries in a large MAP.

3. Spatio-temporal indexing

MAPs usually involve a large amount of non-traditional objects, such as images, video, sound, and text. The quick retrieval of a qualifying set, among the huge amount of data, that satisfies a query based on spatio-temporal relationships is necessary for the efficient construction of a MAP. Spatial and temporal features of objects are identified by six coordinates: the projections on x- (points x_1, x_2), y- (points y_1, y_2), and t- (points t_1, t_2) axes. A serial scheme, maintaining the objects characteristics as a set of seven values (id, x_1, x_2, y_1, y_2, t_1, t_2) is not an efficient solution.

Queries involving spatio-temporal operators, such as the ones presented in Section 2, should usually require the retrieval of all pages of the serial scheme in order to be answered since no ordering is provided by such a scheme. As a conclusion, efficient indexing mechanisms that can support a wide range of spatio-temporal operators need to accompany MAP tools. In the next subsections we propose two indexing schemes and their retrieval procedures.

3.1. A simple spatial and temporal indexing scheme

A simple indexing scheme that could be able to handle spatial and temporal characteristics of actors consists of two indexes:

- a *spatial (two-dimensional) index* for the spatial characteristics (size and x-, y- coordinates) of the objects, and
- a *temporal index* for the temporal characteristics (duration and start/stop time) of the objects.

In the literature concerning the area of spatial databases, several data structures have been proposed for the manipulation of spatial data. Among others, R-trees [7] and their variants [1, 11] seem to be the most efficient ones. On the other hand, the manipulation of temporal information can be supported either by one-dimensional versions of the above data structures (since all of them have been designed for *n*-dimensional space in general) or by specialized temporal data structures.

For uniformity reasons we select a single multi-dimensional data structure (R-tree) to play the role of the spatial (2D R-tree) and temporal (1D R-tree) index. The above indexing scheme is illustrated in Figure 2.

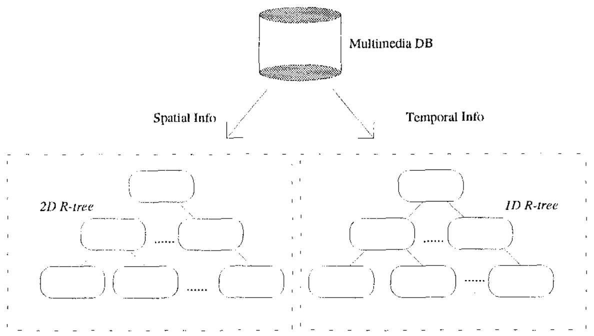


Figure 2: A simple indexing scheme of actors

The above indexing scheme clearly improves the retrieval of spatio-temporal operators compared to serial retrieval. Logarithmic times of response using hierarchical tree indexes are always better than linear ones when no indexes are present, especially when thousands or millions of objects are involved. Even for complex operators where both indexes need to be accessed (e.g. for the *overlap_during* operator) the cost of the two indexes' response times is expected to be lower than the cost of the serial retrieval.

A weak point of the above scheme already has been mentioned. The retrieval of objects according to their spatio-temporal relationships (e.g. the *overlap_during* one) with others, demands access to both indexes and, in a second phase, the computation of the intersection set between the two answer sets. Access to both indexes is usually costly and, in many cases, most elements of the two answer sets are not found in the intersection set. In other words, most of the disk accesses to each index separately are useless. To reduce this problem, several techniques for spatial join, between two R-trees for example, have been proposed [2] and could be applied to our indexing scheme.

However, this technique is not applicable when two completely different indexes are used for spatial and temporal information. A more efficient solution is the merging of the two indexes (the spatial and the temporal one) in a unified mechanism. This scheme is proposed in the next subsection.

3.2. A unified spatio-temporal indexing scheme

In this subsection we propose a unified spatio-temporal indexing scheme that eliminates the inefficiencies of the previous one and further improves the performance of a MAP tool. The proposed indexing scheme consists of only one index: a *spatial (three-dimensional) index* for the complete spatio-temporal information (location in space and time coordinates) of the objects. If we choose the R-tree to be the indexing mechanism then the unified scheme is illustrated in Figure 3.

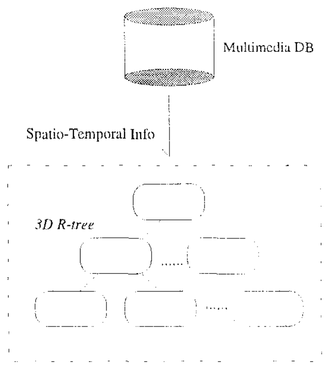


Figure 3: A unified indexing scheme of actors

The main advantages of the proposed scheme, when compared to the previous one, are the following:

- The indexing mechanism is based on a unified framework. Only one spatial data structure (e.g. R-tree) needs to be implemented and maintained.
- Spatio-temporal operators are efficiently supported. Using the appropriate definitions, spatio-temporal operators are implemented as three-dimensional queries and retrieved using the three-dimensional index. So the need for (time consuming) spatial joins is eliminated.

The superiority of the second indexing scheme against the first one and, obviously, the serial scheme on the retrieval of spatio-temporal operators will be shown in Section 4 where an analytical model that predicts the performance of each scheme will be presented

3.3. Retrieval of spatio-temporal operators

The majority of multi-dimensional data structures have been designed as extensions of the classic alphanumeric index, the B-tree [4]. They usually divide the plane into appropriate sub-regions and store these sub-regions in hierarchical tree structures. Multi-dimensional objects are represented in the tree structure by an approximation instead of their actual scheme, for simplicity and efficiency reasons, with the *Minimum Bounding Rectangle* (MBR) approximation, i.e., the smallest rectangle with sides parallel to the axes that totally encloses the object, being the most popular one.

When the MBRs of two objects are disjoint then we can conclude that the objects that they represent are also disjoint. If the MBRs however share common points, no conclusion can be drawn about the topological relation between the objects. For this reason, spatial queries involve the following two-step strategy [8]:

- *Filter step*: The tree structure is used to rapidly eliminate objects that could not possibly satisfy the query. The result of this step is a set of candidates which includes all the results and possibly some false hits.
- *Refinement step*: Each candidate is examined (by using computational geometry techniques). False hits are detected and eliminated.

The R-tree is one of the most efficient hierarchical multi-dimensional data structures. It is a height-balanced tree, which consists of intermediate and leaf nodes. The MBRs of the actual data objects are assumed to be stored in the leaf nodes of the tree. Intermediate nodes are built by grouping rectangles at the lower level. An intermediate node is associated with some rectangle which encloses all rectangles that correspond to lower level nodes. In order to retrieve objects that belong to the answer set of a spatio-temporal operator, with respect to a reference object, we have to specify the MBRs that could enclose such objects and then to search the nodes that contain these MBRs.

As an example, Figure 4 shows how the MBRs of Figure 1 are grouped and stored in the 3D R-tree of our unified scheme. We assume a branching factor of 4, i.e., each intermediate node contains at most four entries. At the lower level, MBRs of objects are grouped into two leaf nodes R1 and R2, which in turn compose the root of the index. If we consider query 3 of subsection 2.2, it corresponds to the *overlap_during* operator with D being the reference object *q*. In order to answer this query, only R2 is selected for propagation. Among the entries of R2, objects C and (obviously) D are the ones that constitute the qualified answer set.

Note that only the right subtree of the R-tree index of Figure 4a was propagated in order to answer the query. The rate of the accessed nodes heavily depends on the size of the reference object *q* and, of course, the kind of the operator (more selective operators result to smaller number of accessed nodes).

A special type of queries, which is very popular during MAP authoring, consists of *spatial* and *temporal layout* retrieval. In other words, queries of the type "Find the objects and their position in screen at the T_0 second" (*spatial layout*) or "Find the objects that appear in the application during the (T_1,T_2) temporal segment and their temporal duration" (*temporal layout*) need to be supported by the underlying scheme. As we will present next, both types of queries are efficiently supported by the unified scheme, since they correspond to the *overlap_during* operator and an appropriate reference object q: a rectangle q_1 that intersects t-axis at point T_0, or a cube q_2 that overlaps t-axis at the (T_1,T_2) segment, respectively.

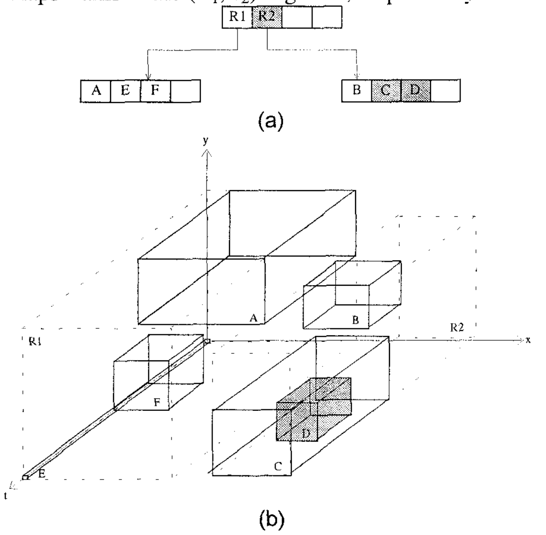


Figure 4: Retrieval of the *overlap_during* operator using 3D R-trees

The reference objects q_1 and q_2 are illustrated in Figure 5a. In a second step, the objects that compose the answer set are filtered in main memory in order to design their positions on the screen (spatial layout) or the intersection of their t- projections to the given temporal segment (temporal layout).

Queries 4 and 5 of subsection 2.2 belong to the "layout" type queries and could be processed as described above. In particular, query 4 could be answered by exploiting the reference object q_1 at the specific time instance $T_0 = 22$sec. The result would be a list of objects (the identifiers of the actors, their spatial and temporal coordinates) that are displayed at that temporal instance on the screen. This result may be visualised as a screen snapshot with the objects that are included in the answer set drawn in that (Figure 5b). As regards query 5, it could be answered using as reference object a cube q_2 having dimensions $(X_{max}-0)\cdot(Y_{max}-0)\cdot(T_2-T_1)$ where $X_{max}\cdot Y_{max}$ is the dimension of the screen and (T_2-T_1) is the requested temporal interval; $T_1 = 10$ and $T_2 = 20$ in our example. The result would be a list of objects (the identifiers of the actors, their spatial and temporal coordinates) that are included or overlapped with the cube q_2. This result can be visualised towards a temporal layout by drawing the temporal line segments of the retrieved objects that lie within the requested temporal interval (T_2-T_1) (Figure 5c).

On the other hand, the simple indexing scheme (consisting of two index structures) is not able to give straightforward answers to the above layout queries, since information stored in both indexes needs to be retrieved.

In the next section both schemes will be analytically evaluated and compared to each other. Their comparison will result to general conclusions on the advantages and disadvantages of each solution.

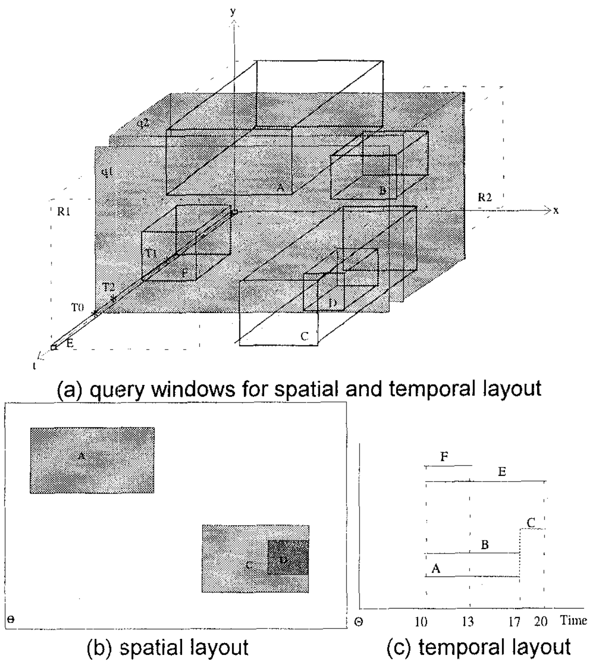


Figure 5: Spatial and temporal layout retrieval using 3D R-trees

4. Evaluation of the indexing schemes

In this section we use an analytical model [12] that estimates the performance of R-trees on the retrieval of n-dimensional queries in order to evaluate the proposed indexing schemes. The analytical formula is applicable to both schemes, if we keep in mind that the simple one consists of one 2D R-tree and one 1D R-tree while the unified one consists of one 3D tree. Using this model we estimate the performance of both schemes and compare their efficiency using several spatio-temporal operators.

4.1 Analysis of R-tree performance

Most of the work in the literature has dealt with the expected performance of R-trees for processing *overlap* queries, i.e., the retrieval of data objects p that share common area with a query window q [6, 13]. More particularly, let N be the total number of data objects indexed in a R-tree, D the density of the data objects in the global space and f the average capacity of each R-tree node. If we assume that q_i ($i = 1, \ldots, n$) is the projection of a query window q on each axis then the expected retrieval cost (i.e., number of disk accesses) of an overlap query using R-trees is [12]:

$$C(q) = 1 + \sum_{j=1}^{1+\left\lceil \log_f \frac{N}{f} \right\rceil} \left\{ \frac{N}{f^j} \cdot \prod_{i=1}^{n} \left(\left(D_j \cdot \frac{f^j}{N} \right)^{1/n} + q_i \right) \right\} \quad (1)$$

where the average density of the R-tree nodes D_j at each level j of the tree structure is given by:

$$D_j = \left\{ 1 + \frac{D_{j-1}^{1/n} - 1}{f^{1/n}} \right\}^n \quad (2)$$

In other words, D_j can be computed recursively using D_0 which denotes the density D of the data MBRs. Qualitatively, this means that we can estimate the retrieval cost of an overlap query based on information about the dataset (amount and density) and the query window only.

Since Eq. 1 expresses the expected performance of R-trees on *overlap* queries using a query window q, in order to estimate the retrieval cost of a spatio-temporal operator $R(p,q)$ we need the following transformation: $R(p,q) \Rightarrow$ *overlap*(p,Q). In other words, the retrieval of a spatio-temporal operator using R-trees is equivalent (in terms of cost) to the retrieval of an *overlap* query using an appropriate query window Q. The necessary transformation Q for each operator R should take into consideration the corresponding constraint of the R-tree nodes because only that is important when estimating the retrieval cost [9]. For each spatio-temporal operator that we consider in this paper, the appropriate query window $Q = (Q_{x1}, Q_{x2}, Q_{y1}, Q_{y2}, Q_{t1}, Q_{t2})$ for the unified scheme (or $Q = (Q_{x1}, Q_{x2}, Q_{y1}, Q_{y2})$, $Q = (Q_{t1}, Q_{t2})$ alternatively for the simple scheme) is defined in Table 1, as a function of the original query window $q = (q_{x1}, q_{x2}, q_{y1}, q_{y2}, q_{t1}, q_{t2})$.

Using information from Table 1 and Eq. 1 we can estimate the expected cost for the query window Q, which equals to the expected cost $C(R)$ for the retrieval of a spatio-temporal operator R. Intuitively, we assume that the unified scheme should be the most efficient one when both spatial and temporal information are included in the query while in the rest cases the simple scheme seems to be preferable.

Operator	Simple scheme	Unified scheme
overlap	$(q_{x1}, q_{x2}, q_{y1}, q_{y2})$	$(q_{x1}, q_{x2}, q_{y1}, q_{y2}, 0, 1)$
above	$(0, 1, q_{y2}, 1)$	$(0, 1, q_{y2}, 1, 0, 1)$
during	(q_{t1}, q_{t2})	$(0, 1, 0, 1, q_{t1}, q_{t2})$
before	$(0, q_{t1})$	$(0, 1, 0, 1, 0, q_{t1})$
overlap_during	-	$(q_{x1}, q_{x2}, q_{y1}, q_{y2}, q_{t1}, q_{t2})$
overlap_before	-	$(q_{x1}, q_{x2}, q_{y1}, q_{y2}, 0, q_{t1})$
above_during	-	$(0, 1, q_{y2}, 1, q_{t1}, q_{t2})$
above_before	-	$(0, 1, q_{y2}, 1, 0, q_{t1})$

Table 1: Query windows *Q* for spatio-temporal operators

4.2 Analytical comparison

In order to compare the efficiency of each scheme on the retrieval of spatio-temporal operators we assumed a multimedia application including 10,000 actors of the following distribution: portions of 75%, 15%, 5% and 5% characterised by (a) small projections on the three x-, y-, t-axes, e.g. text or video that cover a small space on the screen and last a short time interval, (b) zero projection on the two x-, y- axes and a small projection on the third t-axis, e.g. sounds that cover zero space on the screen and last a short time interval, (c) small projections on the two x-, y- axes and a large projection on the third t- axis, e.g. heading titles or logos that cover a small space on the screen and last a long time interval, and (d) large projections on the two x-, y- axes and a small projection on the third t- axis, e.g. full text or background patterns that cover a large space on the screen and last a short time interval, respectively.

The above distribution characterises, in general terms, a typical MAP and will be used as the sample for the comparison of the two indexing schemes. Different distributions of actors are also supported in a similar way by adapting their density D. The sizes of the reference objects q varied from 0% up to 50% of the global space per axis while the corresponding query windows Q for each combination of indexing scheme and operator were computed according to Table 1. Table 2 summarizes the comparative results for the operators discussed in the paper. For uniformity reasons we set the cost of serial retrieval to be 100% (assuming a size of 4 bytes X 7 numbers per object, the serial scheme demands 278 pages of 1024 bytes each in order to store 10,000 objects) and expressed the costs of the two schemes per operator as portions of that value.

Operator	Simple scheme	Unified scheme
overlap	5% - 10%	5% - 15%
above	45% - 50%	80% - 95%
during	2% - 10%	25% - 45%
before	25% - 35%	80% - 95%
overlap_during	5% - 20%	1% - 5%
overlap_before	35% - 40%	3% - 10%
above_during	55% - 60%	15% - 25%
above_before	70% - 85%	50% - 65%

Table 2: Comparison of the two schemes (with respect to serial cost)

Several conclusions arise from the analytical comparison results presented in Table 2:

- The intuitive conclusion that the simple scheme would outperform the unified one when dealing with operators that keep only temporal or spatial information while the opposite would be the case for spatio-temporal operators is really true. The first four operators are efficiently supported by the simple scheme while the cost of the unified scheme is usually two or three times higher. The reverse situation appears for the last four operators.
- Both schemes are much more efficient than the serial retrieval scheme. For the most selective operators (*overlap*, *during*, *overlap_during*) the improvement is at a level of one or even two orders of magnitude, compared to the serial cost. For the least selective operators (*above*, *before*, *above_before*) the cost of the most efficient scheme ranges between 25% and 50% of the serial cost.

The above conclusions are, more or less, expected. However, in real cases, a mixture of temporal, spatial and spatio-temporal operators needs to be supported. The question of the most efficient scheme for such mixed requirements arises. To propose guidelines for answering this question we present in Figure 6 the average cost of each indexing scheme when (a) all eight operators are involved, (b) only the most selective (*inclusive*) operators are involved, and (c) only the least selective (*exclusive*) operators are involved. For each case, we tune the rate of spatial or temporal operators against spatio-temporal ones to vary from 1:9 up to 9:1 and find the *threshold* point, i.e., the rate that indicates the change of the most preferable scheme.

The conclusions from Figure 6 are really interesting:

- First of all, if we do not distinguish between selective and non-selective operators, the threshold point appears at the rate 4.5:5.5. In other words, if the spatio-temporal queries compose more than 45% of the total, then the unified scheme is the best solution. The fact that the threshold is below the middle point (i.e., 50%) indicates that the unified scheme is more robust than the simple one; the extra cost because of the third axis is usually lower than the maintenance cost of two indexes.

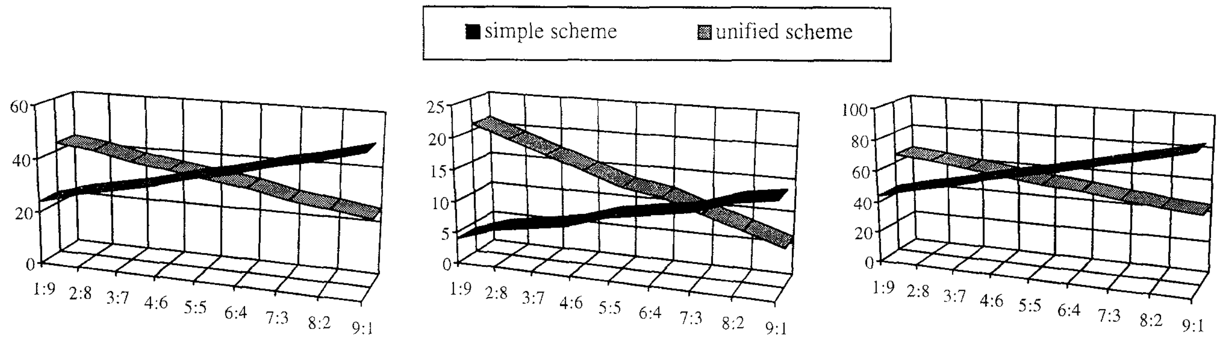
If we distinguish between high- and low- selective operators, then the thresholds shift right (high-selective operators) or left (low-selective operators). In other words, when dealing with selective operators, the simple scheme is sometimes preferable even if the majority (up to 65%) of the queries involve spatio-temporal information. A possible explanation is that 1D and 2D R-trees are extremely fast when dealing with high-selective temporal and spatial operators, respectively, while the extra one or two axes involved in the 3D R-tree significantly raise its cost. The conclusion is reversed for low-selective operators.

The above conclusions make a clear evaluation of the two proposed indexing schemes when various parameters (type of operators, rate of spatial and temporal operators against spatio-temporal ones, selectivity of operators) are involved. It is a choice of the multimedia database designer to select the most preferable solution, with respect to the requirements of the MAP author.

5. Conclusion

In this paper we proposed a mechanism for management of actors in large multimedia applications. This mechanism is based on indexing spatial and temporal presentation features of the actors during the application. The two indexing schemes proposed are based on the R-tree structure; the first scheme includes one 1D and one 2D R-tree that index temporal and spatial characteristics of actors, respectively, while the second scheme includes one 3D R-tree that indexes the spatio-temporal characteristics of actors, considering time to be the third axis of the coordinate system. We evaluated the two schemes against the "serial" one and presented guidelines that help one to select the most appropriate solution, with respect to the specific requirements of the application.

Authoring complex MAPs that involve a large number of actors is a complicated task, keeping in mind the large set of possible events that may encounter in the application context and the number of actors as well as the various potential combinations of these parameters. Thus the need for a scheme that will support the author to manage the large number of actors and spatio-temporal relationships among them is required. Current authoring tools do not provide such facilities. The mechanism we proposed provides an actor indexing scheme to authors who can query, before application execution, the application scenario for spatial and/or temporal relationships among actors (i.e., "does actor A spatially overlap with actor B in the application?" or "which actors temporally overlap with



(a) all operators involved (b) only the most selective (*inclusive*) operators involved (c) only the least selective (*exclusive*) operators involved

Figure 6: Retrieval cost of the two indexing schemes (% of the serial cost)

actor A?"). Moreover, authors may request spatio-temporal layouts of the application at specific spatial and/or temporal instances.

Our approach supports pre-orchestrated scenarios. Interactive scenarios are not supported because of the non-deterministic spatial and temporal occurrences of the actors, a fact that makes indexing a very complicated task. However, the investigation of this problem belongs to our targets for future work.

A second direction concerns *playout management* based on the indexing scheme: the model we proposed could be as well used during the execution phase of the scenario. In this case the appropriate media would be quickly located on the basis of the scenario.

References

[1] N. Beckmann, H.-P. Kriegel, R. Schneider, B. Seeger, "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles", *Proceedings of ACM SIGMOD International Conference on Management of Data*, May 1990.

[2] T. Brinkhoff, H.-P. Kriegel, B. Seeger, "Efficient Processing of Spatial Joins using R-trees", *Proceedings of ACM SIGMOD International Conference on Management of Data*, May 1993.

[3] T. Chiueh, "Content-Based Image Indexing", *Proceedings of the 20th International Conference on Very Large Databases* (VLDB), September 1994.

[4] D. Comer, "The Ubiquitous B-tree", *ACM Computing Surveys*, 11(2):121-137, June 1979.

[5] C. Faloutsos, W. Equitz, M. Flickner, W. Niblack, D. Petkovic, R. Barber, "Efficient and Effective Querying by Image Content", *Journal of Intelligent Information Systems*, 3:1-28, 1994.

[6] C. Faloutsos, I. Kamel, "Beyond Uniformity and Independence: Analysis of R-trees Using the Concept of Fractal Dimension", *Proceedings of the 13th ACM Symposium on Principles of Database Systems* (PODS), May 1994.

[7] A. Guttman, "R-trees: A Dynamic Index Structure for Spatial Searching", *Proceedings of ACM SIGMOD International Conference on Management of Data,* June 1984.

[8] J. Orenstein, "Spatial Query Processing in an Object-Oriented Database System", *Proceedings of ACM SIGMOD International Conference on Management of Data*, May 1986.

[9] D. Papadias, Y. Theodoridis, T. Sellis, M.J. Egenhofer, "Topological Relations in the World of Minimum Bounding Rectangles: a Study with R-trees", *Proceedings of ACM SIGMOD International Conference on Management of Data*, May 1995.

[10] D. Papadias, Y. Theodoridis, "Spatial Relations, Minimum Bounding Rectangles, and Spatial Data Structures", *International Journal of Geographic Information Systems* (to appear).

[11] T. Sellis, N. Roussopoulos, C. Faloutsos, "The R$^+$-tree: A Dynamic Index for Multidimensional Objects", *Proceedings of the 13th International Conference on Very Large Databases* (VLDB), September 1987.

[12] Y. Theodoridis, T. Sellis, "A Model for the Prediction of R-tree Performance", *Proceedings of the 15th ACM Symposium on Principles of Database Systems* (PODS), June 1996.

[13] M. Vazirgiannis, C. Mourlas, "An Object Oriented Model for Interactive Multimedia Applications", *The Computer Journal*, 36(1), 1993.

[14] M. Vazirgiannis, M. Hatzopoulos, "Integrated Multimedia Object and Application Modelling Based on Events and Scenarios", *Proceedings of the 1st IEEE International Workshop for MM-DBMSs*, August 1995.

[15] M. Vazirgiannis, Y. Theodoridis, T. Sellis, "Spatio-Temporal Composition in Multimedia Applications", *Proceedings of the International Workshop on Multimedia Software Development* (MMSD), March 1996.