Assessing Multimedia Similarity: A Framework for Structure and Motion

Vasilis Delis
Computer Engineering & Informatics Department, Patras University and Computer Technology Institute
Kolokotroni 3, Patras, Greece 26221
+30 61 273496
delis@cti.gr

Dimitris Papadias
Department of Computer Science, Hong Kong University of Science & Technology
Clearwater Bay, Hong Kong
+852 23586971
dimitris@cs.ust.hk

Nikos Mamoulis
Department of Computer Science, Hong Kong University of Science & Technology
Clearwater Bay, Hong Kong
+852 23586971
mamoulis@cs.ust.hk

ABSTRACT

In this paper we address the issue of *structural multimedia similarity*, which is based on the relations between the individual objects that comprise a multimedia document. We propose a binary string encoding for 1D relations which permits the automatic derivation of similarity measures. We then extend it to various resolution levels and many dimensions and show that reasoning on spatiotemporal structure is significantly facilitated in the new framework, by applying it to multimedia presentation and motion similarity.

Keywords

Multimedia Similarity, Similarity Queries, Spatiotemporal Relations

1. INTRODUCTION

The general theme of this work is the development of a formal framework for expressing queries on multimedia data. The motivation comes from two common recent observations: i) the increasing availability of multimedia documents in the WWW ii) the emerging need for efficient navigation in such document repositories. We consider a "multimedia document" (or simply document) as a collection of any form of data (text, images, sound, video, etc.) which a user may be interested in querying upon. The salient features of such documents identified so far are *content* (semantic information), *interactivity* (control flow, based on user interaction) and *structure* (which refers to spatiotemporal relations among the document's objects). Related research conducted in the information retrieval field so far mainly focuses on content-based retrieval, covering several data forms, like text [17], images and video [7] [12] [18] [19].

In the Proceedings of the 6th ACM International Multimedia Conference (ACM-SIGMM), Bristol, U.K., 12- 16 September 1998.

However, the advent of the WWW and its universal adoption pose new demands for retrieval queries [13]. First, the amounts of data are huge, thus the output of a typical content based query is likely to be still prohibitively large to be handled by the user. Second, as cognitive studies have shown [6], often human remembering favours structural information as opposed to exact content. Both above factors advocate to the development of new query and retrieval mechanisms based on spatiotemporal behaviour, in addition to content.

To clarify the above, assume a database consisting of tourist multimedia documents and a user who wishes to: "find all documents which contain a picture of an island accompanied by a textual description on its *left*, immediately *followed* by a video that *contains* both windows". Such a query combines both spatial and temporal aspects, although each of them may individually constitute a distinct query. Also, users might want to retrieve objects belonging to video frames or images of moving scenes, based on *motion*, which, in the general case, is a complex interplay of spatial and temporal properties: "find a video's portion (set of frames) in which a car moving to the *right* crashes into a house wall".

The processing of such queries should be sufficiently flexible to allow partial matches, as the difference between objects that satisfy the query and the ones that don't, may be quantifiable and gradual. This can be attributed to two main factors: i) users may want to specify varying degrees of satisfaction in order to retrieve configurations *similar* to the queried ones and ii) spatiotemporal objects or relations are sometimes fuzzy by nature (e.g. "northeast" may not conform to universally accepted semantics).

Our work proposes a framework for similarity retrieval of multimedia documents. We first introduce a new expressive encoding for relations and subsequently show how structural relation-based similarity can be effectively accommodated in this framework. The rest of the paper is organized as follows: Section 2 discusses a new encoding of 1D relations which facilitates similarity reasoning and its extensions to multiple resolution levels and dimensions. Section 3 applies the framework to spatiotemporal queries involving the retrieval of

multimedia documents that conform to some input structure. In Section 4 we discuss motion queries including examples. Section 5 concludes with a brief discussion and an outline of future work.

2. SPATIOTEMPORAL SIMILARITY

Several types of relations, such as, *temporal* [1], *topological* [5] (e.g., inside, overlap), *directional* [16] (e.g., left, northeast) and *qualitative distances* [10], have been defined and used in a wide range of applications. Our goal is to provide a unified and adjustable framework which permits the definition of any type of spatio-temporal relation and the automatic generation of similarity measures. We will initially confine our discussion to one dimension and consider a (reference) interval [a,b]. We identify nine potential regions of interest:

1.(-∞,a-δ) 2.[a-δ,a-δ] 3.(a-δ,a) 4.[a,a] 5.(a,b) 6.[b,b] 7.(b, b+δ) 8.[b+δ, b+δ] 9.(b+δ,+∞)

For each of the above regions we associate a binary variable, *r, s, t, u, v, w, x, y, z,* respectively (Figure 1a). Given a primary interval [c,d], the value of every variable indicates the result of the intersection between [c,d] and the variable's associated region ("0" corresponds to an empty intersection while "1" corresponds to a non-empty one). Thus, we can define 1D relations to be 9-tuples ($R_{rstuvwxyz}$: $r, s, t, u, v, w, x, y, z \in \{0,1\}$). For instance, $R_{011000000}$ can be interpreted as *left-near* while $R_{000011110}$ as *overlap-right*.

We call such a consecutive partitioning of space a *resolution scheme*. The rationale behind the above choice of regions is the need to express in a single framework all feasible topological and directional relations in 1D, as well as the simplest type of distance relations, *near* (within a distance of δ) and *far* (otherwise). However, there are several possible resolution schemes and a particular choice is affected by the users' expectations or the application's requirements, as every scheme can refine or generalize a particular relation class. In applications where more refined distance relations are required, multiple interval extensions of δ or possibly different lengths may be used. On the other hand, if distance relations are not necessary we may apply a simpler scheme such as (-∞,a) [a,a] (a,b) [b,b] (b,+∞) (see Fig. 1b). According to this resolution scheme, 13 distinct relations are possible, which coincide with the ones described by Allen in his seminal work on temporal intervals [1].

In the example of Figure 1c, the 1D regions of interest are (-∞,a), [a,b] and (b,+∞), respectively. The corresponding relations are of the form R_{tuv}, $t,u,v \in \{0,1\}$. This scheme allows for the definition of only 6 primitive relations since information content concerning the endpoints of [a,b] is reduced: R_{100} (*before*), R_{010} (*during*), R_{001} (*after*), R_{110} (*before-overlap*), R_{011} (*after-overlap*), R_{111} (*includes*). Figure 1c illustrates four distinct configurations that correspond to R_{010} and cannot be distinguished at this resolution. On the other hand, *overlap* relations can be refined to *strong-overlap* or *weak-overlap* by splitting (a,b) to two intervals using the mid-point (Figure 1d).

In general, by increasing (decreasing) the number of bits, we capture more (less) detail in the description of relations. For every resolution scheme, a binary string represents a valid relation if i) it contains a list of (at least one) consecutive "1"s, ii) in the case of a single "1", this should not correspond to a point variable (e.g. *s,u,w* or *y* in Figure 1a) otherwise [c,d] collapses to a point and iii) the intervals of interest form a consecutive partition of (-∞,+∞).

If b is the number of bits used by the resolution scheme, the number of feasible relations in 1D is $b(b+1)/2 - k$, where k is the number of point variables, i.e. intervals of the form [a,a]. If we fix the starting point at some bit then we can assign the ending point at the same or some subsequent bit. There are b choices if we fix the first point to the leftmost bit, $b-1$ if we fix it to the second from the left, and so on. The total number is $b(b+1)/2$ from which we subtract the k single-point intervals. For $b=9$, $k=4$ (resolution scheme of Figure 1a) we get 41 relations (see Figure 2a), while for $b=5$, $k=2$, there exist 13 (Allen's) relations. For the rest of the paper, we will call the feasible relations at a particular scheme, *primitive* relations.

The next step is to provide a mechanism for representing similarity among relations independently of the resolution scheme. Freksa [8] defined the concept of *conceptual neighborhood* as a cognitively plausible way to measure similarity among Allen's interval relations. A neighborhood is represented as a graph whose nodes denote relations that are linked through an edge, if they can be directly transformed to each other by continuous interval deformations. Depending on the allowed deformation (e.g., movement, enlargement), several graphs may be obtained. Figure 2a represents the neighborhood graph for the distance-enhanced resolution scheme, assuming that a minimal deformation is a movement of a single interval endpoint. Starting from

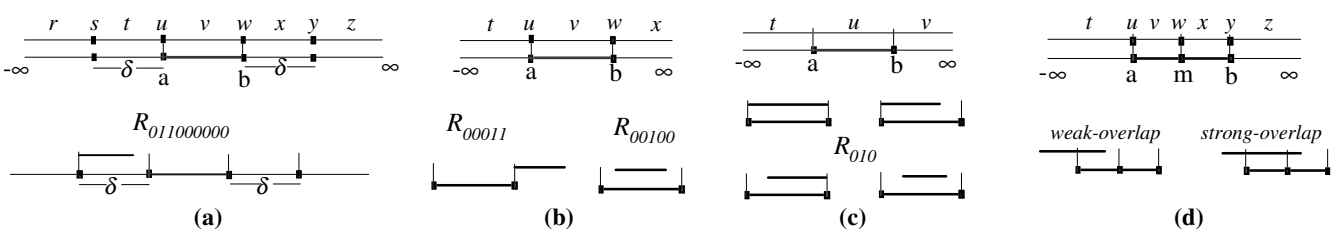

Figure 1 Binary encoding of relations

relation $R_{100000000}$ and extending the upper interval to the right, we derive relation $R_{110000000}$. With a similar extension we can get the transition from $R_{110000000}$ to $R_{111000000}$ and so on. $R_{100000000}$ and $R_{111000000}$ are called 1st degree neighbors of $R_{110000000}$. The distance d between two relations is equal to the length of the shortest path connecting them in the neighborhood graph.

Unlike other methods that utilise conceptual neighbourhoods by manually constructing distance tables (e.g., [8] [14]), the binary encoding allows the automatic derivation of relation distance at any resolution scheme, through the following pseudo code:

```
distance(relation R1, relation R2) {
R = R1 OR R2;  //bitwise OR
   d = 0;
   for i:= leftmost_1(R) to rightmost_1(R) {
      if R1[i]=0 then d++;
      if R2[i]=0 then d++;
    } //end-for
   return(d);
}
```

Intuitively, the above routine compares the corresponding binary strings and counts the minimal number of "1"s that have to be added in order to make the notations two identical binary strings with consecutive "1's. For example $d(R_{000110000}, R_{010000000}) = 5$ and $d(R_{011000000}, R_{111100000}) = 2$ (the underlined 0s are the ones counted during the calculation of distance).

The proposed encoding and distance calculation can be extended accordingly to multi-dimensional spaces. The relation between two objects in N dimensions corresponds to the combination of N 1D relations. Thus, an ND relation can be naturally defined as a N-tuple of 1D *projections*, e.g. $R_{110000000\text{-}111000000} = (R_{110000000}, R_{111000000})$. For example, 3D relations can be used to describe relationships between volumes or spatiotemporal objects (2D snapshots over time). Figure 2b illustrates a few indicative 3D spatiotemporal relations (the order of the constituent 1D projections being *x-y-time*) between five rectangles and a reference (gray) one. The *x-y* relation components are expressed in the distance-enhanced scheme while Allen's scheme is used for the temporal dimension.

Assuming block world metrics, the distance between two ND relations is the sum of the pair-wise distances between the constituent 1D projections, i.e. $d(R_{i1\text{-}i2\text{-}\ldots\text{-}iN}, R_{j1\text{-}j2\text{-}\ldots\text{-}jN}) = d(R_{i1}, R_{j1}) + d(R_{i2}, R_{j2}) + \ldots + d(R_{iN}, R_{jN})$. Other metrics (such as Euclidean [14]) can also be applied.

The advantages of the proposed framework are i) the expressiveness of the encoding in the sense that given the new notation, the corresponding spatiotemporal configuration can be easily inferred, and vice versa, ii) efficient automatic calculation of relation distance (similarity), and iii) the uniform representation of several types of relations in various resolution levels.

3. STRUCTURAL QUERIES

The multidimensional extension of 1D relations define *projection-based* primitive relations, as each 1D relation between ND objects corresponds to the relation between their corresponding 1D projections. Projection-based definitions of relations and similarity measures are particularly suitable for structural similarity retrieval in multimedia applications because:

- often in practice, multimedia documents consist of rectangular objects (e.g., window objects). Projection-based relations provide an accurate and effective means for spatio-temporal representation of collections of such objects [16].
- even for non-rectilinear objects, usually spatial/multimedia databases utilise minimum bounding rectangles (MBRs) for efficient indexing. MBRs provide a fast *filter step* which excludes the objects that could not possibly satisfy a given query. The actual representations of the remaining objects are then passed through a (computationally expensive) *refinement step* which identifies the actual output [3].
- multimedia queries do not always have exact matches and crisp results. Rather, the output documents should

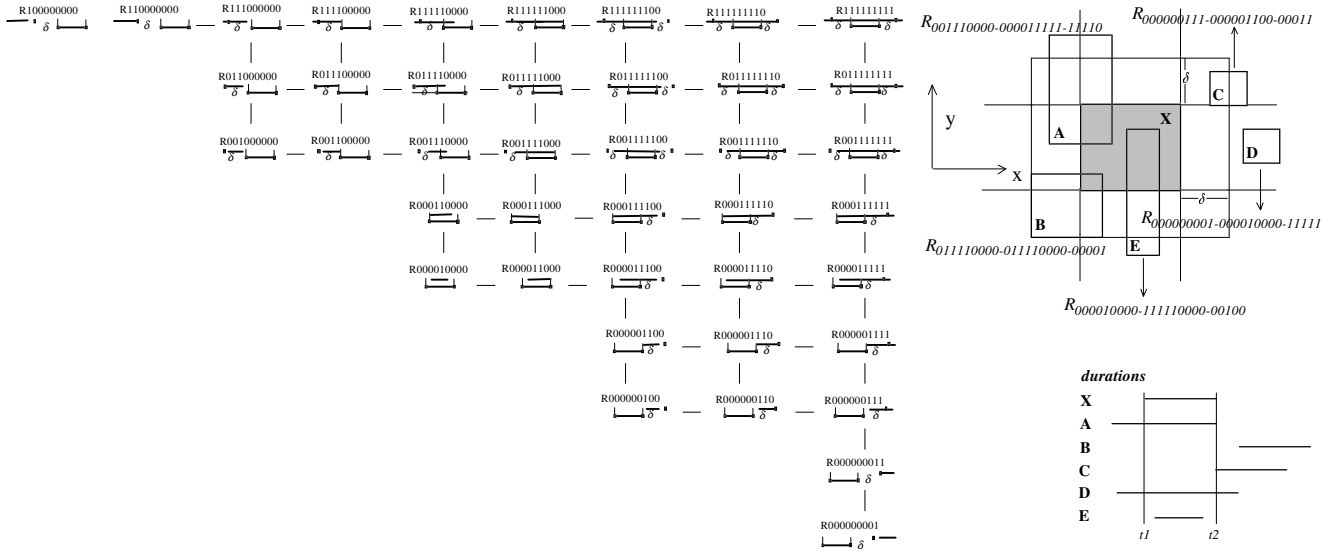

Figure 2 Conceptual neighborhood including distances and a multi-dimensional example

have an associated "score" to indicate the similarity between their retrieved spatiotemporal relations and the target relations of the query document.

Multiresolution and multidimensional conceptual neighborhoods can be tuned to provide flexible processing mechanisms for structural queries. For the sake of simplicity and readability of relation notation, in the following examples we use Allen's resolution scheme. Processing of queries at any other, sufficiently refined scheme, is completely analogous.

Consider the simple but indicative spatiotemporal query already mentioned in the introduction: "find all documents which contain a picture *I* of an island accompanied by a textual description *T adjacently* on its *left*, *followed* by a video *V* that *contains* both windows". In general, such queries may involve specific instances (e.g. Mauritius) or classes (island). This query essentially defines a spatiotemporal scene (see Fig. 3a), described by the following predicates (assuming an *x-y-time* order):

$R_{11000\text{-}U\text{-}01110}(T,I)$
$R_{11111\text{-}11111\text{-}00011}(V,I) \vee R_{11111\text{-}11111\text{-}00001}(V,I)$
$R_{11111\text{-}11111\text{-}00011}(V,T) \vee R_{11111\text{-}11111\text{-}00001}(V,T)$

In the above, R_U denotes the universal projection relation, i.e. the disjunction of all possible projection relations at the corresponding resolution scheme, and is used for defining *incomplete* (unspecified) constraints between query variables. Also, query relations might be *indefinite* (the constraint may be a disjunction of primitive relations). For example, in the above query the relation *followed* can be interpreted as $R_{x\text{-}y\text{-}00011} \vee R_{x\text{-}y\text{-}00001}$.

During query processing, stored images are sequentially examined and different instantiations of pairs of objects are assessed for matching each of the above predicates. In Figure 3 the gray objects portray two *complete instantiations* (all query variables have been instantiated); one with a low similarity (3c) and one that matches the query closely but not exactly (3b). In general, the "goodness" of a solution is calculated by various metrics [11], using the distances between the input constraints and the actual (instantiation) relations. For instance, the total distance (dissimilarity) of a solution can be defined as the sum of all pair-wise distances (the instantiation $R_{01110\text{-}10000\text{-}00001}(T_2,I)$, $R_{10000\text{-}11111\text{-}00001}(V_2,I)$ and $R_{10000\text{-}00011\text{-}00110}(V_2,T_2)$ in Fig. 3c has a total distance D = 7+4+9).

Formally, a structural query can be described as a *soft constraint satisfaction problem* (CSP) which consists of:

- A set of variables. In the example of Figure 3, there exist three variables *I*, *T*, and *V*.
- For each variable a finite domain of potential values. For instance the domain of *T* is the set of textual descriptions.
- For each pair of variables a binary spatio-temporal constraint (e.g., the constraint $C_{T\text{-}I}$ is : $R_{11000\text{-}U\text{-}01110}$). Each constraint is *soft* in the sense that it can be violated to a maximum distance specified by the user.

Let *n* be the query size (number of variables) and *N* be the domain size (assuming a common domain): in the worst case (exhaustive search), all *n*-permutations of *N* objects have to be searched in order to find solutions (i.e., *N!/(N-n)!*). In real DBMSs where N>>n, this number is $O(N^n)$, meaning that the retrieval of structural queries can be exponential to the query size. Query processing becomes more expensive if inexact matches are to be retrieved.

In order to deal with the problem effectively, we apply one of the most efficient constraint satisfaction algorithms, namely *forward checking with dynamic variable reordering* [2] combined with R*-trees [3], the most efficient variation of R-trees [9]. The main mechanism of forward checking is the following: every time a variable is instantiated, its constraints restrict the potential values for the set of variables yet to be instantiated (by pruning values inconsistent with the current instantiation).

We use the R*-tree to facilitate this pruning process. The relations between an uninstantiated variable and the instantiated ones can be transformed to a window in space; only the values that lie inside this window can be assigned to the variable. Thus, the domains of uninstantiated variables are formed after applying a window query to the R*-tree, which is a much cheaper operation than a brute force search in the whole domain space. A thorough algorithmic description and experimental evaluation can be found in [15].

4. MOTION QUERIES

Although motion is conceived as a continuous phenomenon by humans, computer motion is essentially a collection of discrete phenomena which we will refer to as *frames*. Assuming an ordered set of frames, either

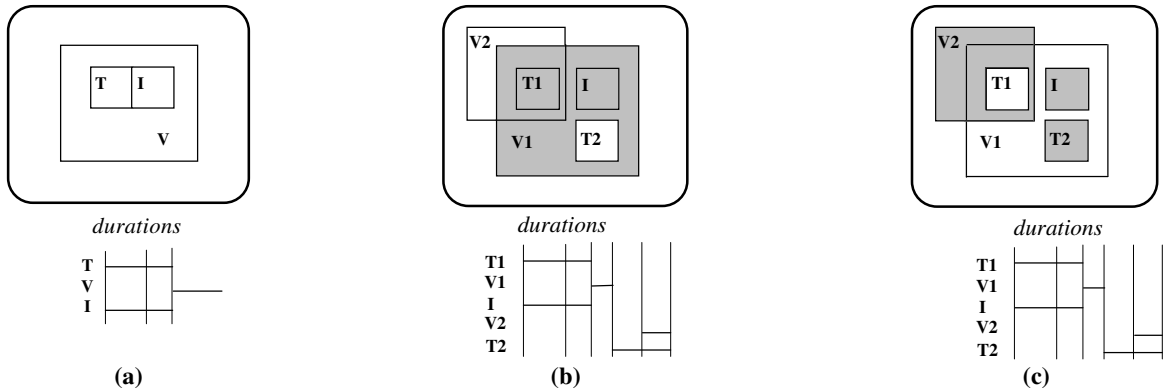

Figure 3 Example spatiotemporal similarity

captured from a video clip or representing any ordered collection of images of moving objects (satellite imagery, etc.), several questions may be of importance to a user:

1. Find the first frame where an object starts moving.
2. Find the video clip's portion (set of frames) where a set of objects move from some initial positions to some destinations.
3. Find the video clip's portion where an object performs a specific movement with respect to a reference object.
4. Describe the movement of an object as a set of relation variances.
5. Compare the velocity of different objects.
6. Given a video clip, find a frame with a specific spatial arrangement (which is reduced to a 2D structural query).

The core of any motion query processor must consist of a mechanism that compares consecutive frames and decides whether they are similar enough to be regarded as an "elementary" motion. Of course the similarity between different movement patterns relies on several factors:

- the resolution scheme; e.g. a small object's movement along a large reference object will not be considered as motion unless a sufficiently refined resolution is adopted to distinguish among several *overlap* relations
- the sampling rate of frames, as it controls the perceived motion's smoothness
- the user's expectations and the application requirements

There exist various interesting ways to elaborate on each of the above query types and identify motion patterns. For instance, assume that we are interested in identifying "smooth motion" as opposed to arbitrary movement. This could be based on an assessment of variances among relations, either between successive positions of the moving object or between the moving object and a fixed reference.

Figure 4 illustrates nine sample movement frames. Let R_i be the relation between A and B in frame i. Then a motion constraint can be defined as: $d(R_i, R_{i-1}) \leq \tau$, meaning that in order for an arbitrary movement to constitute motion the distance between the relations of A and B in two successive frames must be less or equal than a certain threshold (for the next example τ=2). An obvious implicit constraint is that A is not allowed to have the same position in any two successive frames. The degree of smoothness can be indicated by several possible measures, one of which is:

$$S = \left(\frac{\sum_{i=1}^{f-1} d(R_i, R_{i-1})}{f-1} \right)$$ where f is the number of frames

As long as the above constraints hold, the smaller the value of S, the smoother the movement in the corresponding set of frames. For the example in Figure 4, the value of S is (2+2+0+2+0+0+2+2)/8 = 1.25, which could be less for a more dense sampling of frames.

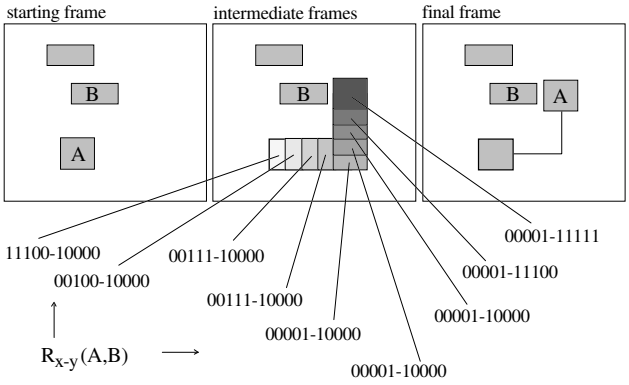

Figure 4 Assessing movement by fixed reference

In order to experimentally quantify the performance of motion similarity retrieval, we constructed several sets of frames, where each frame is a file of 5000 objects organized in an R-tree (the same set of objects in all frames). The global extent per axis was set 1000, while, between two successive frames, each object was allowed to move at a maximum distance of 5 on each axis with probability 0.1. Essentially, the majority of objects in two successive frames are in the same position where a few are in neighboring ones (a situation similar to satellite imagery).

The motion query was: "find all sets of objects that moved from a specified initial position (query window) to some final one". In order to answer this query we have to retrieve all objects that fall inside the initial query window, check whether any of these objects fall in the final window (in some subsequent frame), and assess whether their movement in intermediate frames constitutes motion. Figure 5 illustrates the CPU-time of the query (using a SUN UltraSparc2 200MHz with 256 MB of RAM) as a function of the number of frames (10, 20,…,100) and the ratio query_window_area/ average_object_area (50,100,…, 500).

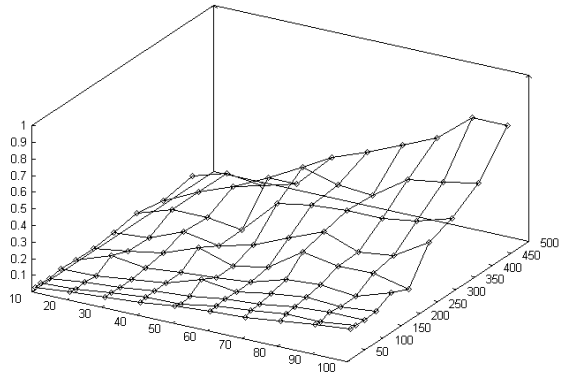

Figure 5 Experimental evaluation of a motion query

The processing of the above query is in general more efficient than structural queries since it is polynomial: it only involves retrievals using some fixed query windows which can be performed efficiently using spatial indexing. More complicated motion patterns can be analysed, e.g. periodical vs. non-periodical movement, which may be identified with the use of string matching algorithms.

5. CONCLUSIONS

This work addresses the issue of multimedia similarity queries, which form a special type of retrieval based on spatiotemporal structure. Multimedia documents (which are arbitrary collections of text, sound, images and video objects) are assessed according to the spatial and temporal relations that exist among their constituent objects. The result of such an assessment verifies whether the spatiotemporal behaviour of a document matches totally or partially that of an input query.

We reduce this general similarity problem to elementary 1D relation similarity which has been investigated in various ways, most of which relied on the notion of conceptual neighborhood. We propose a formal yet practical framework for encoding 1D relations in a way that allows efficient similarity calculation. We subsequently extend the model in a uniform way to arbitrary dimensions and multiple resolution levels, thus covering many potential applications. The novelty of the framework is its uniform treatment of all kinds of relations, in any number of dimensions and resolution.

Structural similarity is relatively neglected in the literature. [14] and [4] apply conceptual neighborhoods for configuration similarity problems in GIS. Unlike the proposed method, these techniques are restrictive in that they do not uniformly treat all types of spatial relations, while they assume fixed domains and queries, i.e. the permitted relations (direction and topological) are defined in advance and can't be tuned to different resolution levels.

Future continuation of this work is possible in both theoretical and practical directions. For example, the algebraic properties of different sets of relations that are feasible at different resolution levels could be studied and motivate the framework's extension for hierarchical relation similarity problems. From a practical point of view, a very fruitful research direction would be the coupling of our techniques with appropriate query languages [19] for spatiotemporal domains, and with multimedia indexing techniques in order to improve access to large multimedia documents.

Acknowledgements

This work was done during a first author's visit to the Hong Kong University of Science and Technology supported by DAG96/97.EG36 Hong Kong RGC. He also acknowledges a PhD studentship from Bodosakis Foundation.

References

[1] Allen, J. “Maintaining Knowledge About Temporal Intervals”. CACM, 26(11), 1983.

[2] Bacchus, F., van Run. “Dynamic Variable Ordering in CSPs”. 1st International Conference on Principles and Practice of Constraint Programming, 1995.

[3] Beckmann, N., Kriegel, H.P., Schneider, R., Seeger, B. “The R*-tree: An Efficient and Robust Access Method for Points and Rectangles”. ACM SIGMOD, 1990.

[4] Bruns,T., Egenhofer, M.“Similarity of Spatial Scenes”. 7th Symposium on Spatial Data Handling, 1996.

[5] Egenhofer, M.J., Herring, J., “Categorizing Binary Topological Relationships Between Regions, Lines and Points in Geographic Databases”, *Technical Report*, University of Maine, Orono, ME, 1991.

[6] Egenhofer, M., Mark, D. “Naïve Geography”. COSIT'95, LNCS 988, 1995.

[7] Faloutsos, C., Equitz, W., Flickner, M., Niblack, W., Petkovic, D., Barber, R. “Efficient and Effective Querying by Image Content”. Journal of Intelligent Information Systems, vol. 3, pp. 1-28, 1994.

[8] Freksa, C. “Temporal Reasoning based on Semi Intervals”. Artificial Intelligence, 54, 199-227, 1992.

[9] Guttman, A. "R-trees: a Dynamic Index Structure for Spatial Searching". ACM SIGMOD, 1984.

[10] Hernandez, D., Clementini, E., Di Felice, P. “Qualitative Distances”. COSIT'95, LNCS 988, 1995.

[11] Jain, R., Murthy, J., Tran, P., Chatterjee S. "Similarity Measures for Image Databases". FUZZ-IEEE, 1995.

[12] Maybury, M.T. (ed.) "Intelligent Multimedia Information Retrieval". AAAI Press/MIT Press, 1998.

[13] Megalou, E., Hadzilacos, Th., Mamoulis, N. “Conceptual Title Abstractions: Modeling and Querying Very Large Interactive Multimedia Repositories”. 3rd Multimedia Modelling, 1996.

[14] Nabil, M., Ngu, A., Shepherd, J. “Picture Similarity Retrieval Using 2D Projection Interval Representation”. IEEE TKDE, 8(4), 1996.

[15] Papadias, D., Mamoulis, N., Delis, V. “Algorithms For Querying by Spatial Structure”. VLDB, 1998.

[16] Papadias, D., Sellis, T. “Qualitative Representation of Spatial Knowledge in 2D Space”. VLDB Journal, 3(4), pp. 479-516, 1994.

[17] Salton, G., Allan, J., Buckley, C. “Automatic Structuring and Retrieval of Large Text Files”. Communications of the ACM, 37(2): 97-108, 1994.

[18] Seidl,T., Kriegel, H. “Efficient and User Adaptable Similarity Search in Large Multimedia Databases”. VLDB, 1997.

[19] Smith, J. R., Chang, S-F. “Searching for Images and Videos on the WWW”. Technical Report CU/CTR 459-96-25, Columbia University, 1996.