Retrieval of Multimedia Documents by Pictorial Content:
A Prototype System

Panos Constantopoulos, John Drakopoulos
and Yannis Yeorgaroudakis

November 1989

FORTH/ICS/TR/1989/022
Technical Report Series
Institute of Computer Science, FORTH†
P.O.Box 1385, Heraklion, Crete, 711-10 Greece
Tel.: +30.81.229302, Fax: +30.81.229342, Telex: 262389 CCI GR
E-mail – UUCP: mcvax!ariadne!panos

† "Foundation of Research and Technology – Hellas," formerly "Research Center of Crete" (RCC).
RETRIEVAL OF MULTIMEDIA DOCUMENTS BY PICTORIAL
CONTENT: A PROTOTYPE SYSTEM

P. Constantinopoulos, J. Drakopoulos, Y. Yeorgaroudakis
Institute of Computer Science
Foundation of Research and Technology-Hellas
P.O. Box 1385
GR-71110 Heraklion, Crete, Greece
UUCP:.:.mcvax!ariadne!panos

ABSTRACT

A prototype system for the retrieval of multimedia documents by pictorial content is presented. It is designed to be part of an integrated multimedia document retrieval tool. Document images are accessed directly, using image and object attributes and the relative positions of objects within images, as well as indirectly, through associated document components. The prototype system currently operates on graphical images from various domains and is extensible to address bitmap images too.

1. Introduction

A multimedia document is a structured collection of attribute, text, image and voice data [Gibb87, Tsic83]. It is expected that in the not too distant future huge amounts of information, largely multimedia, which currently exist in the form of letters, journals, books, periodicals, designs, hospital patient records, product catalogues, etc., will be stored in high capacity storage devices such as optical and magnetic disks and handled by computer [Chri86]. This handling includes the generation, storage, retrieval, correlation and distribution of information. A considerable body of research on the subject as well as systems implementing the above functions already exist [Fein82, Tsic83, Fors84, ISO84, IEEE85, Thom85, Chri86, Cons86, Gibb87].

From the point of view of document retrieval, the various components of a multimedia document may be characterized as active or passive, depending on whether they can or cannot be used for accessing the document. To the best of our knowledge, only the attribute and text components are active in current multimedia document filing systems. In general, multimedia documents may be addressed by attributes or by content. The attribute (formatted) data are addressed using indexing techniques. On the other hand, the unformatted data comprise text, images (graphical and bitmap) and voice recordings. The textual content can be addressed using either indexing or other methods, such as signatures [Falo85]. The image and voice content is currently being addressed only indirectly, through associated text, such as captions, or image/voice attributes (e.g. subject, speaker,
etc.) [Chi86, Bada87]. Recently an approach to the problem of retrieving images from data bases of multimedia documents containing images as components, based on image analysis techniques and application domain knowledge has been proposed [Con87].

Image analysis techniques have been employed for the description of images in various fields, such as robot vision, industrial automation, character recognition, reconnaissance, etc. In general, a derived description of a given image is compared to a model of an image class in order to determine the content of that image [Bal82, Rose82]. Most of the available image description methods are based on the knowledge of some specific application domain and are not particularly well suited for efficient access to image data bases. This is mainly due to issues of knowledge representation, uncertainly and computational complexity [Tso84].

Standalone image data base systems (IDBS) do exist [Chan81, Tanu84], yet there is no single view of pictorial data that is adequate for all IDB applications [IDBS88]. An IDBS must provide support for various image representations, access to special purpose image processing functions (e.g., for feature extraction), a query language for content-based retrieval, and efficient access mechanisms. Approaches to fulfilling these requirements include applications and extensions of relational and hierarchical data base systems interfaced with image understanding and manipulation systems, and pictorial data abstractions imported from computer graphics [Chan81]. Extensions or generalisations of traditional data models have been proposed to accomodate the fact that for pictorial data the ratio of instances per class is small. A recent trend is to provide extensibility rather than a particular set of extensions to a DBMS, by building object-oriented data base systems, such as PROBE [Oren88], suitable for a range of image data base applications.

As pointed out in [Chan78, Chan81], two polar design problems in pictorial information systems are the storage, retrieval and processing of a large number of pictures on one hand and of very large, complex pictures on the other, whereas a major consideration is whether the system will be used mainly for retrieving or for processing pictures. Moreover, the importance of flexible, intelligent user interfaces for IDBS has been substantially advocated [Chan79, Chan81, Cons88a] and, in particular, the use of flexible query formulation and browsing has been proposed as a simple way of resolving uncertainly in the image retrieval process.

In a multimedia document, a picture is only one of many components: attributes, text, voice recordings and other pictures. In some multimedia document systems, such as MULTOS [Cons86], MUSE [Gibb87] and MINOS [Chi86], the multimedia document is the unit of information, containing its components as parts which belong exclusively to that document and are only accessed through it. Alternatively, components may be shared by various documents and enjoy relative autonomy: multimedia documents are composed from parts through a set of relations. This approach is adopted in object-oriented multimedia data bases, for instance see [Woe86]. In either case, when the target of the retrieval process is the document as a whole, this can be based on a number of clues: the content and structure of components, the structure of the document as a whole and semantic associations between components. This enables the realisation of
powerful, flexible retrieval strategies in which various access methods are combined in a complementary capacity.

In this paper we present the design and pilot implementation of the function of retrieving multimedia documents by pictorial content, which has been developed at the Institute of Computer Science, FORTH, in the context of a general retrieval tool for a multimedia document filing system.

The retrieval tool is part of the client subsystem of the MULTOS multimedia document filing system. MULTOS is project no. 28 of the ESPRIT programme, which aims at developing a multimedia document filing system based on an open, distributed, client-server architecture, supporting retrieval on the basis of the structure and content of documents, and using both magnetic and optical storage media. The entire client subsystem, including, of course, the retrieval tool, follows a completely new, object-oriented design [Cons88c]. Retrieval methods inherited from previous versions of MULTOS include retrieval by document attributes using indexing techniques, and retrieval by textual content using signatures (see [Cons86], [Falo85]). Furthermore, the retrieval tool provides the possibility of integrating an access method based on pictorial content with the above mentioned ones under a uniform user interface.

The experimental system discussed in this paper can provide the missing function of retrieval by pictorial content. The system operates on symbolic pictures containing objects from a fixed vocabulary pertinent to some application domain. These are either the output of some graphical package, or of processing and interpreting bitmap images. The objects contained in a picture can have adjoined attributes describing characteristic features, and so can the picture as a whole. Moreover, the order relations of the objects along two axes are recorded in a two-dimensional string [Chan87]. Pictures can be accessed by attribute matching and 2-d substring matching. Through a query-by-pictorial-example interface, multimedia documents are thus retrieved on the basis of the semantic content as well as the structure of the contained pictures. Various degrees of matching accuracy are allowed. Browsing and flexible query reformulation are used to resolve query uncertainty.

In the next section we give a brief overview of the retrieval tool. In section 3 we present the adopted approach to retrieval by pictorial content. In section 4 we describe the pilot implementation and, finally, we close with an overall discussion and further work.

2. A multimedia document retrieval tool

The MULTOS client subsystem comprises a number of tools each of which performs a set of operations related to a function offered to the user, e.g., document preparation, document retrieval, document browsing, document type definition, etc. [Cons88c]. Document retrieval is performed by the Retrieval Tool. This tool is made up of other tools: Query Editor, Query History, Query Execution, Document Browser and Structure
Browser. Mutually recursive calls of these component tools are allowed within an instantiation of the Retrieval Tool.

The functional design of the Retrieval Tool [Cons88b] provides for maximum flexibility in query formulation, resolution of user uncertainty by dynamic query reformulation and exploitation of associations between various document components. For retrieval purposes, two partially overlapping sets of documents which cover all documents in store are defined: a search space and an auxiliary space, pertaining to a retrieval operation, the content of which is listed in Table 1. Retrieval is made from the search space, while the auxiliary space contains documents which may be helpful in query formulation.

<table>
<thead>
<tr>
<th>SEARCH SPACE</th>
<th>AUXILIARY SPACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>• archival store</td>
<td>• query catalogue</td>
</tr>
<tr>
<td>(types, collections)</td>
<td></td>
</tr>
<tr>
<td>• temporary collections</td>
<td>• temporary collections</td>
</tr>
<tr>
<td>(derived from query answer sets)</td>
<td></td>
</tr>
<tr>
<td>• permanent private documents</td>
<td>• permanent private documents</td>
</tr>
<tr>
<td></td>
<td>• temporary documents</td>
</tr>
</tbody>
</table>

Table 1: Document spaces pertaining to a retrieval operation.

Some organisation is usually imposed on the document spaces. In MULTOS, for instance, archived documents are classified according to a hierarchy of document types and also belong to collections of documents of possibly different types. This organisation is exploited to restrict the search space which can further be restricted on some combination of query answer sets.

Queries may address, directly or indirectly, one or more document components. Indirect addressing involves references from associated components, e.g., an image caption is a text component referring to an image component and so is an in-text reference to an image.

Figure 1 shows the layout of the query editor interface as it appears for queries on pictorial content. The top part of the interface is for displaying restrictions on the search space imposed by the user. The bottom part is reserved for displaying queries in the syntax of the query language and can also be used by the sophisticated user for making queries in that language directly. The middle, larger part is the query editing area where support and guidance for query formulation is provided by a different interface depending on the document component (attribute, text, image) the query is based on.
Components (and the respective interfaces) are activated for querying through a pop-up menu. Any sequence and multiplicity of component activations are allowed. Thus, even very complex queries involving all document components can be formulated in simple, guided steps.

The main window appearing in the image query interface is the image specification canvas. It enables the specification of an image in a variety of ways, as explained in the next section. Queries specified through the graphical query editing interface are in expanded conjunctive normal form. Other auxiliary windows, such as application-specific object vocabularies, text specification window, scanner user interface and object attribute specification window are available on demand.

Document browsing is provided within the retrieval tool as an integral part of the retrieval process. A query answer set may contain irrelevant documents due to user uncertainty in query specification, inexact matching techniques (e.g., text signatures), or both. Browsing is the natural final stage of document selection. Moreover, browsing can be useful in the query formulation stage: inspecting the important features of documents accessed by a previous query, or some sample document, can help the user decide what exactly to look for. Two levels of browsing are provided: inter-document browsing, i.e. through a set of documents, only presenting relevant features; and Intra-document browsing, i.e. through a single document at any level of detail.


Once the search space is restricted on certain document collections and types, the query editor enables the specification of image query terms using image structure, object and image attributes, and associated text.

Image document components and, correspondingly, image query terms can be either graphical or bitmap images. Graphical images are produced by a graphical package and consist of graphical primitives supplied by the package, or ad hoc primitives related to an application domain. Bitmap images can be input through a scanner or loaded from some existing file. Graphical images already are in symbolic form, whereas bitmap images have to be interpreted into symbolic ones. It is on symbolic images that query processing is performed.

A symbolic image consists, in general, of objects, relations among objects and descriptions of object and image properties. Each application domain has, in general, a corresponding alphabet of basic objects and production rules whereby compound objects can be composed from the basic ones. All these objects together constitute a vocabulary specific to the application domain. We shall call image class the set of images pertaining to an application domain and image class vocabulary the corresponding vocabulary of objects. In the case of graphical images, the basic objects can fixed arrangements of even more elementary objects, such as the primitives of a graphical editor, or the elementary objects themselves. The alphabet of the graphical editor itself can be considered a
generic vocabulary.

The interpretation of bitmap images into symbolic images is the task of highly application-specific image analysis and understanding packages. Our multimedia document filing system assumes no particular, narrow application domain. Rather, it is intended for general office filing and is required to be amenable to specialisation through a flexible, modular design. Therefore, no image understanding module is included, though it can. On the other hand, manual classification of bitmap images is supported by the system.

The properties of objects and whole images are described by object and image attributes respectively. The relations among objects considered here are their relative positions in space without further interpretation attached to them.

So, a symbolic image is here taken to be a triple

\[ P = (S, R, a_I) \]

where:

- \( S \subseteq V \times A_0 \) the set of objects in the image,
- \( V \) the vocabulary of objects and
- \( A_0 = A_{0_1} \times A_{0_2} \times \cdots \times A_{0_n} \) the object attributes;
- \( R : S \times S \rightarrow R^P \) a mapping expressing the spatial relationships between objects,
- \( R^P = \{:, N, NE, E, SE, S, SW, W, NW\} \)
  - (E.g., (a,b,SE) denotes that b lies southeast of a and (a,b,:); that they are in the same position); and
- \( a_I \in A_I = A_{I_1} \times A_{I_2} \times \cdots \times A_{I_m} \) the global image attributes.

A fixed vocabulary is assumed and the domain production rules are supposed to be unavailable outside the graphical image editor or the image understanding package. Also, the above mapping \( R \) only represents relative position of objects in ordinal terms. Ricner image descriptions, supporting deeper understanding and interpretation do exist (see, e.g., [Rose82, Eshe86, IDBS88]). Here we aim at developing a relatively simple scheme which, as part of the multimedia document retrieval tool, will effectively address as wide a range of image classes as possible.

Encoded as explained further below, symbolic images are stored as image signatures in a sequential image signature file with pointers to the documents containing the corresponding images. Query processing involves scanning the image signature file to match the signatures of the image query terms (duly generated) with those contained in the file. The image signature file is updated by appending new signatures. This access structure bears an analogy to the text signature file for document retrieval by textual content [Falbo85], also used in our multimedia document retrieval tool.

An image signature consists of an attributed two-dimensional string and an image attribute vector. The former encodes the objects, their relative positions and their attributes, and is an extension of a data structure called two-dimensional (2-d) string to
include object attributes. The latter holds the global image attributes. In the rest of the paper, attention is drawn only on the attributed 2-d string component of the image signature. Query conditions on global attributes can simply be processed before conditions on attributed 2-d strings thus improving the performance of query processing.

A 2-d string is a pair of strings representing the order relations of the projections of the objects contained in a symbolic image along two axes. Partial picture matching is obtained by two-dimensional substring matching at various levels of matching accuracy. The 2-d string representation of symbolic images is defined as follows.

Given a symbolic image, the symbols contained in it are projected on two axes. Symbols can be considered as point objects. Their projections on the two axes are points on two lines with well-defined orderings respectively. These order relationships along each axis can be represented by a string of the form \( x_1 y_1 x_2 y_2 \cdots y_{(n-1)} x_n \) where \( x_1, x_2, \ldots, x_n \) are symbols in the image class vocabulary and \( y_1, y_2, \ldots, y_{(n-1)} \) are "\( \lt \)" or nothing. A sequence \( x_i x_{(i+1)} \) denotes coincidence, whereas \( x_i < x_{(i+1)} \) denotes precedence along an axis. The order relationships along both axes are represented by a pair of strings (hence the name 2-d string) of the form:

\[
(x_1 y_1 x_2 y_2 \cdots y_{(n-1)} x_n, x_p(1)z_1 x_p(2)z_2 \cdots z_{(n-1)} x_p(n)),
\]

where \( p(1), p(2), \ldots, p(n) \) is a permutation of \( 1, 2, \ldots, n \) and \( z_1, z_2, \ldots, z_{(n-1)} \) denote order like the \( y \)'s.

Under certain conditions, a picture can be unambiguously reconstructed from its 2-d string [Chan87].

The extraction of the 2-d string of an image is easiest when the objects in the image have mutually disjoint minimum enclosing rectangles. Then the objects can simply be represented by the corresponding symbols located at the centres of the respective minimum enclosing rectangles. Otherwise, objects can be segmented into so-called ortho-relational objects with respect to some point-of-view object, reducing the problem to the previous case [Chan88].

Three levels of matching accuracy corresponding to three different sets of conditions which may obtain in 2-d substring matching are distinguished. Let \( Q \) be the query image and \( P \) a document image. The levels of matching, in order of decreasing accuracy are:

**Level 2**: There exists a subpicture \( P' \) of \( P \) which contains all the objects in \( Q \) and only those, in the same order along both axes (i.e., in the same relative positions) and there are no objects in \( P-P' \) the projections of which fall strictly between projections of objects in \( P' \).

**Level 1**: Differs from level 2 in that \( P-P' \) can contain objects projecting strictly between objects in \( P' \).
**Level 0**: Differs from level 1 in that the order relations between certain objects in \( Q \) may be indifferent, at least along one axis. Order indifference along an axis between two objects is expressed by placing these objects in \( Q \) so that they project on the same position on that axis.

Clearly, when matching is achieved at a certain level it is achieved at the lower levels too. Matching levels are characterised analytically using the rank function of a symbol in a string [Chan87]. The *rank* of the symbol \( \sigma \) in the string \( u \) is denoted by \( r_u(\sigma) \) and is defined as:

\[
r_u(\sigma) = 1 + \text{number of '<' preceding } \sigma \text{ in } u
\]

A string \( u \) is *contained* in a string \( v \), if \( u \) is a subsequence of a permutation of \( v \). A string \( u \) is a *type-i 1-D subsequence* of string \( v \) \((i=0,1,2)\) iff:

1. \( u \) is contained in \( v \), and
2. if \( a_1w_1b_1 \) is a substring of \( u \), \( a_1 \) matches \( a_2 \) in \( v \) and \( b_1 \) matches \( b_2 \) in \( v \), then

   (type-0) \( : \quad r_v(b_2) - r_v(a_2) \geq r_u(b_1) - r_u(a_1) \)
   \[\text{or } r_u(b_1) - r_u(a_1) = 0\]

   (type-1) \( : \quad r_v(b_2) - r_v(a_2) \geq r_u(b_1) - r_u(a_1) > 0\)
   \[\text{or } r_u(b_1) - r_u(a_1) = 0\]

   (type-2) \( : \quad r_v(b_2) - r_v(a_2) = r_u(b_1) - r_u(a_1) \)

Finally, a 2-D string \((u_1, v_1)\) is a *type-i 2-D subsequence* of a 2-D string \((u_2, v_2)\) \((i=0,1,2)\) iff:

1. \( u_1 \) is a type-i 1-D subsequence of \( u_2 \), and
2. \( v_1 \) is a type-i 1-D subsequence of \( v_2 \)

Then picture \( P \) matches the query picture \( Q \) at level \( i \) if the 2-d string of \( Q \) is a type-i 2-d subsequence of the 2-d string of \( P \).

Adding attributes to the objects represented in 2-d strings gives rise to attributed 2-d strings. Picture matching on the basis of attributed 2-d strings involves the following:

Assume that image \( Q \) (query) is matched against image \( P \). Let \( q = (q_v, q_{A_0}) \) be an object in \( Q_S \) and \( p = (p_v, p_{A_0}) \) an object in \( P_S \), where subscripts correspondingly denote:\n
\( V \) the member of the vocabulary, \( A_0 \) the vector of attributes and \( S \) the set of objects in the image. We say that \( p \) in \( P_S \) is *compatible* with \( q \) in \( Q_S \) iff \( p_v = q_v \) and \( p_{A_0} \) is compatible with \( q_{A_0} \). Now the latter holds iff for each attribute \( x \) in \( q_{A_0} \) the following are true:
(1) name(x) = ANYNAME and value(x) = ANYVALUE and there exists at least one attribute y in p_{A_0} or

(2) name(x) = ANYNAME and value(x) ≠ ANYVALUE and there exists an attribute y in p_{A_0} such that value(y) = value(x) or

(3) name(x) ≠ ANYNAME and value(x) = ANYVALUE and there exists an attribute y in p_{A_0} such that name(y) = name(x) or

(4) name(x) ≠ ANYNAME and value(x) ≠ ANYVALUE and there exists an attribute y in p_{A_0} such that name(y) = name(x) and value(y) = value(x).

Also, the correspondence between attributes x in q_{A_0} and y in p_{A_0} must be 1-1.

The special values ANYNAME and ANYVALUE are introduced in order to provide for cases where agreement in attribute name or value is not actually required. For instance, one may be looking for a picture containing a house but cannot recall its colour. If "colour" is an attribute of "house" then its value can be overlooked by setting value(colour) = ANYVALUE.

Now define

\[ G_q = \{ p \in P_S \mid p \text{ is compatible with } q \} \] for each q in Q_{S_s}.

If \( G_q \neq \emptyset \) for every \( q \) in \( Q_{S_s} \) then 2-d substring matching is performed to determine distinct representatives of the sets \( G_q \) which have similar spatial relations with the respective \( q \). Similarity of spatial relations holds when subpicture matching at level 0, 1 or 2, defined above, is achieved.

Thus, attributed 2-d strings enable a combination of image access by object attributes with access by relative positions, giving priority to attribute conditions so as to restrict the search space over which 2-d substring matching is carried out.

4. A pilot implementation

A pilot system for the storage and retrieval of attributed symbolic pictures derived from graphical images has been developed using the graphical editor Fig and the C programming language on a SUN-3 workstation running UNIX 4.3 bsd. The Fig editor is used to create the graphical images stored in the image database as well as to edit images which are submitted as queries in a query-by-example mode. Fig offers a set of graphical primitives which figures can be drawn with on a grid of variable resolution. The graphical objects and their locations are recorded in a Fig output file which the 2-d string of the picture is derived from. The resolution of the grid is chosen according to the desired accuracy of location. The various images are stored as Fig files while their 2-d strings are
used to compile an index. Compound objects can be declared as such. Indeed, this is used
to define the vocabulary of an image class, which consists of all the objects encountered
in an application domain. This vocabulary is offered to the user in both image editing and
query editing modes, along with the original Fig graphical primitives. Fig has been
appropriately modified to enable the assignment of attributes to objects, so as to support
the attributed 2-d string image access method discussed above. The implementation of
2-D substring matching is based on a variant of the algorithm of [Chan87] which was
developed.

The system, currently supporting pictorial queries by name and by example, is
schematically depicted in figure 2. Queries by example address both semantic content
and structure.

We now give, by way of example, instances from two usage sessions of the pilot
system. In the first, no reference to a specific application domain is made. In the second,
the domain of interest is architectural design.

Example 1: As no specific domain is referred to, there is no particular image class
related to this session and the query image is composed from the primitives supplied by
the Fig graphical editor. The query (A) is shown in Figure 3a and the reply in Figure 3b.
The "object" in the query picture (sail) is actually a composite one, consisting of the
straight and curved lines (four in total) that were used to draw it. The match is at level 1
because in the picture of the sailship, the projection of the mast on the horizontal axis
falls between the projections of the vertical parts of the sail.

Example 2: We now restrict our attention to architectural designs, in particular
single room floor plans. The search is limited in that image subclass and the query interface
is adapted accordingly to offer a set of image class primitives representing the members
of the image subclass vocabulary, in a window appearing in the lower right.

Figure 4a shows a kitchen floor plan (K1) in the upper right (main) window as a
sample element of the image subclass and primitives used in kitchen design in the lower
right window. A kitchen with a different layout (K2) is shown in Figure 4b. Here, how-
ever, certain characteristics of the refrigerator have been recorded as attributes: it is of
blue colour, medium size and made in Greece. Note that display and modification of
attributes is made through a uniform interface which only appears on request using the
mouse.

Consider now queries B, C and D shown in Figures 5a, 5b and 5c respectively.
They all ask for a floor plan where a stove appears to the north-east of a refrigerator. In
addition, in query B the refrigerator is required to be of blue colour and made in Japan,
while in query C only blue colour is required. No match is found to query B because no
refrigerator in the stored kitchen plans is made in Japan. The floor plan K2 matches
query C because attribute compatibility holds and the refrigerator and stove are in the
same relative positions. Finally, both floor plans K1 and K2 match query D since no
attributes are specified in it.
Note that all the above matches are at level 1 because the minimum enclosing rectangles of the matching subpictures also contain objects not specified in the queries. Note also that the overall layouts in Figures 4a and 4b are different. Finally note that the primitives of the Fig graphical editor are available in addition to the image class primitives. If a declared member of the image class vocabulary is drawn using the editor primitives in a query picture, it will be processed as a set of simple graphical objects, forcing the matching to be performed at a level where objects are decomposed into their graphical components, thereby incurring inefficiency and possible mismatches. It is therefore preferable to use image class primitives when available.

5. Discussion

We have described a method for multimedia document retrieval by pictorial content in the context of a multimedia document filing system and a prototype implementation of the image retrieval function. It operates on attributed symbolic images which may be derived either from the output of a graphical editor, or from an application-specific image analysis and understanding system. In its present form, the prototype system employs the Fig graphical editor for generating pictures and editing queries-by-example, properly modified to accommodate object attributes.

To the authors’ best knowledge, the system most akin to the prototype described here is the prototype image data base system of Chang et al. [Chan88]. That system is a standalone IDBS which supports pictorial queries by name, keys and 2-d strings, and has a simple graphical user interface with a predefined set of icons for query-by-example. The system described here supports, in addition, object attributes and a corresponding enhancement of the 2-d string access method, uses a specially enhanced version of a general-purpose graphical editor, supports user-definable image classes corresponding to various application domains, and is designed to be part of an integrated multimedia document retrieval tool. Complex browsing functions are supplied by the browser of the retrieval tool.

The use of object attributes in combination with 2-d strings enriches the semantic content of the image representation and improves the specificity and effectiveness of queries. Moreover, efficiency improves by processing conditions on attributes before those on spatial relations. In fact, the cost of one 2-d substring matching operation by the algorithm of [Chan87] is \( O(M) + O(N^2 \cdot L^3) \) in time and \( O(N^2 M^2 L) \) in memory, where \( M = |P_S|, N = |Q_S|, L = \max\{ |G_q| / q \text{ in } Q_S \} \). It is clear that, in general, query conditions on object attributes reduce \( L \).

It was recently shown that the 2-d substring matching algorithm of [Chan87] allows, in rare occasions, false drops, i.e. situations in which a match is indicated without this being actually true [Drak89]. This reminds of a similar effect in text retrieval by text signatures [Falco85]. In [Drak89] an exact, yet slower algorithm for 2-d substring matching is proposed, as well as an inexact one which is faster than that of [Chan87]. Both the exact matching algorithm and the fast inexact matching algorithm have been
implemented in our prototype system.

As already mentioned, our prototype system currently operates on graphical images and, more generally, on attributed symbolic images. Automatic image understanding and classification depends heavily on the application and no such facility is provided at present. Not all applications, though, involving bitmap images do require a fully automatic facility. In certain environments, an interactive classification system where part or all of the analysis is carried out by a human operator can be useful. We plan to implement such a module as part of the multimedia document filing system.

The integration of the present system with the multimedia document retrieval tool called for some extra implementation effort. The implementation environment of the client subsystem of the MULTOS multimedia filing system is the Andrew toolkit, an object-oriented programming environment on top of the X windows, compatible with code in the C language. Andrew includes a graphical editor called Zip, which, however, was unavailable at the time that the pilot system was being developed. Figure 1 actually presents the query interface as it was built using Andrew and Zip. After finishing the integration, we plan to work on the evaluation of the multimedia document retrieval tool with respect to its access methods and the man-machine interaction it supports.

Acknowledgement

This work has been supported in part by the ESPRIT project MULTOS (no.28) of the European Communities. Partners in project MULTOS are: Battelle(FRG), Epsilon(GR), ERIA(S), ICS-FORTH(GR), IEI-CNR(I), Olivetti-DOR(I) and Triumph-Adler(FRG).
References


Figure 1. MULTOS query editor: interface for image queries.
Figure 2. Structure of the pilot system.
Figure 3a. Image query A: a sail, no attributes, unspecified image class.
Figure 3b. Reply to image query A.
Figure 4a. Kitchen floor plan K1, no attributes, interior architectural design image class.
Figure 4b. Kitchen floor plan K2, attributes, interior architectural design image class.
Figure 5a. Image query B, attributes, interior architectural design image class, no match.
Figure 5b. Image query C, attributes, interior architectural design image class, match with K2.
Figure 5c. Image query D, no attributes, interior architectural design image class, match with K1 and K2.