Medical Image Databases: Image Indexing by Content

Stelios C. Orphanoudakis ††, *Petros Kofakis* †, *Yannis Kavaklis* †, *and Euripides Petrakis* †

† Institute of Computer Science, FORTH
711 10 Heraklion, Crete, Greece
and
‡ Departments of Diagnostic Imaging and Electrical Engineering
Yale University, New Haven, CT 06510

Abstract

In this paper we present initial work on the design and implementation of an image database system for medical images. The objective of this project is the development of an environment that will permit the quick and intelligent access of images based on their content. An image database is a complex system consisting of an image analysis module, a database module, and a user interface and display module. These modules must be unified and integrated. This paper focuses primarily on the justification of the design decisions, an illustration of the major features of such a system, and a requirements definition for its implementation. An object-oriented approach is described which meets these requirements.

1 INTRODUCTION

The different components of an Image Database (IDB) system (i.e. image descriptors, raw image data, image processing procedures that are used to extract the image descriptors, display procedures, the DBMS, the storage and retrieval rules, etc.) are usually loosely coupled. That is, data and the procedures that operate on the data are quite distinct. An older version of our system [1] used this "loosely coupled" paradigm (Fig. 1). Raw images were processed by an image processing module and image descriptors were obtained. These image descriptors were handled by a RDBMS and used for the indexing of images. Queries were formulated using SQL and an expert system, operating on top of the RDBMS, was envisioned which would assist in the definition of domain specific retrieval strategies (matching).

In this paper, an object-oriented approach is described which enables the integration of the various components of an IDB system. Such a system is currently under development in our laboratory. In this system, data, the image analysis module, the database, the user interface, and image display modules are tightly coupled and are considered as a single module. The object-oriented approach is a very attractive one for the design and implementation of an IDB. It provides mechanisms for capturing the semantic content of an image in a highly modular and hierarchical manner at various levels of abstraction It also provides a framework for representing both image content and operations in a uniform way, while the property of inheritance can be used to reduce the amount of stored data. A number of multimedia filing systems have been developed which integrate graphical representation with a DBMS to broaden the bandwidth of information transfer between the computer and the user [2]. A method of storing and manipulating images, based on the object-oriented approach, is discussed in detail in [3].

The primary objective of the work presented in this paper is the development of a system that will permit the quick and intelligent access of images, based on their content, out of a large set of :

- reference images for diagnostic and educational purposes (data bank).
- images archived in a PACS image database.

A second objective is to increase the efficiency of diagnostic image interpretations. Thus, if a doctor encounters a difficult case and needs some supporting evidence to make his diagnosis, the proposed system would allow him to consult similar images associated with a confirmed diagnosis.

The basic criteria for the choice of an object-oriented approach to the design of an IDB system were : 1) The use of an environment and a representation/data model that would enable the integration of the different components of the system, 2) the efficient manipulation of the database schema (classes), 3) enabling new and complex data types that can represent the composition of and the complex relationships among image segments, 4) the need for an extensible and flexible system and 5) the need to facilitate rapid prototyping. The final goal is the design and implementation of a highly interactive system.

2 OBJECT-ORIENTED APPROACH

Object-oriented (OO) systems have received considerable attention in the past few years. Such systems provide powerful concepts for applications development and programming, such as modularity, encapsulation, inheritance, overriding, protocols and polymorphism [4,5]. Conventional database systems are effective in handling files containing fixed-length fields and provide a range of facilities such as interfaces for both end users and applications programmers and data management facilities to enforce security, reliability and integrity [6]. However, such systems have a number of shortcomings, principally due to inadequate support for semantic and functional properties of real-world objects. Thus, current data models are unable to support the semantic richness of the conceptual model of the real-world. Furthermore, DBMSs cannot store and handle rules, except for limited integrity constraints, and are unable to fully support the functional properties of data. The Object-Oriented database (OODB) model has several advantages over the existing DBMS-based models [7]. Specifically, it combines the advantages of:

Knowledge Based Systems: Complex and dynamic structures of objects in the virtual memory, abstraction and inheritance mechanisms, procedural objects, inferencing and deduction capabilities, constraint mechanisms.

Database Systems: Permanent data in the long-term storage, recovery from crash, resiliency of data, concurrent access, consistency checking, query capabilities.

Furthermore, the concepts used in an object-oriented environment can drastically accelerate the prototyping of a complex system [8].

In contrast, conventional database systems emphasize data independence by separating the world into two independent parts, namely the data and the procedures operating on the data. Thus, we have to split our application into persistent, uninterpreted data and programs that share and manipulate such data. Object-oriented systems emphasize object independence, by encapsulation of individual objects, and a design composed entirely of objects, only some of which are persistent.

3 IMAGE REPRESENTATION

3.1 Image analysis

To retrieve images stored in an image database based on image content, image analysis techniques must first be used to generate appropriate representations of these images. Image analysis can be performed off-line or interactively under the supervision of an expert. In our current work, the performance of individual image analysis algorithms is not our primary concern. Instead, we are interested in providing the user with an environment which supports direct manipulation of image data for interactive, flexible, and globally robust image analysis as needed for image storage and retrieval by content. In many other highly interactive systems that are based upon principles known as direct manipulation and graphical control, it is desirable to manipulate data directly, rather than issuing commands to the system, by exploiting the interactive communications facilities afforded by the display system. Anything displayed on the screen (pictures) is active. Pointing at a specific location of a picture should activate a specific set of actions (processing, display etc.). Low-level image processing can be run in a model-driven fashion. We know a priori, from the header of an image, which class the image belongs to at a high-level of abstraction. Information contained in the description of this class can be used to set default parameters, rules, or procedures so that preliminary image processing is fairly robust for this class of images. Once some easy to detect constructs of image content have been identified, we can iterate using information contained in the description of classes at a lower level of abstraction and selectively execute some of the image processing procedures (automatically or interactively) with parameters set at a more discriminating level.

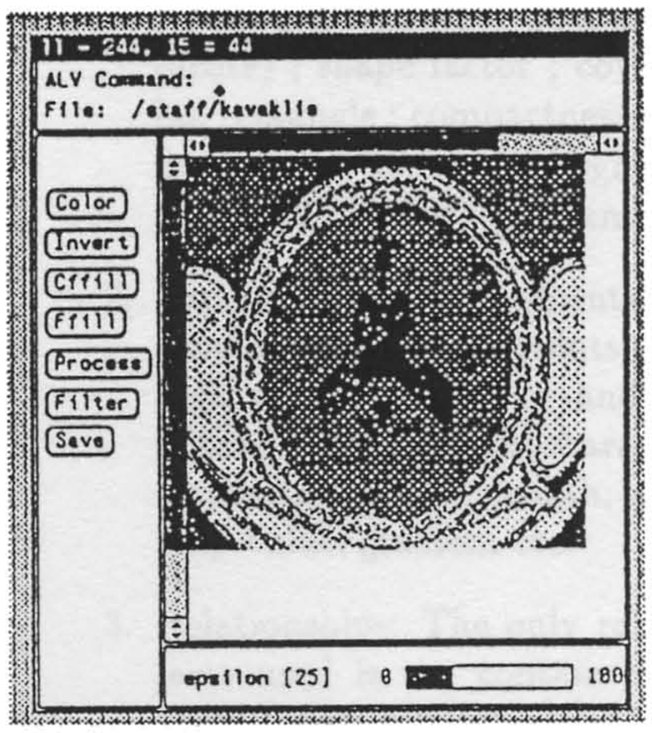

Figure 1: Image processing window

The need for an interactive image analysis environment arises from the fact that image analysis methods are not always robust. Sometimes it is difficult to automatically detect regions of interest or adequately segment parts of the image that represent soft tissues. Thus, we can automatically extract the "easy" or reference contours and add manually other interesting or ambiguous contours. Abnormal stuctures can then be defined interactively. To edit the results of automatic

image segmentation or to define other regions of interest, the user must be provided with a variety of tools which are easy to use. In our particular implementation, an integration of many different image processing libraries was achieved in a way which is transparent to the user. These are accessed through an image processing window which is shown in Fig.1. The images to be processed are selected with the help of a browser.

3.2 Image content description

Upon completion of image analysis, an image description is available based on a collection of attributes and features at different levels. It is not trivial to determine the optimum representation based on which a given object can be compared with other objects in the database. Picture descriptions are generally given in terms of properties of objects contained within the picture and relationships among such objects. A picture description may also include properties or attributes of the picture as a whole. In order to describe image content for storage and retrieval purposes, we do not need as detailed a description as is normally needed for diagnostic image interpretation. A description that can adequately represent the important structures contained within the picture may be sufficient.

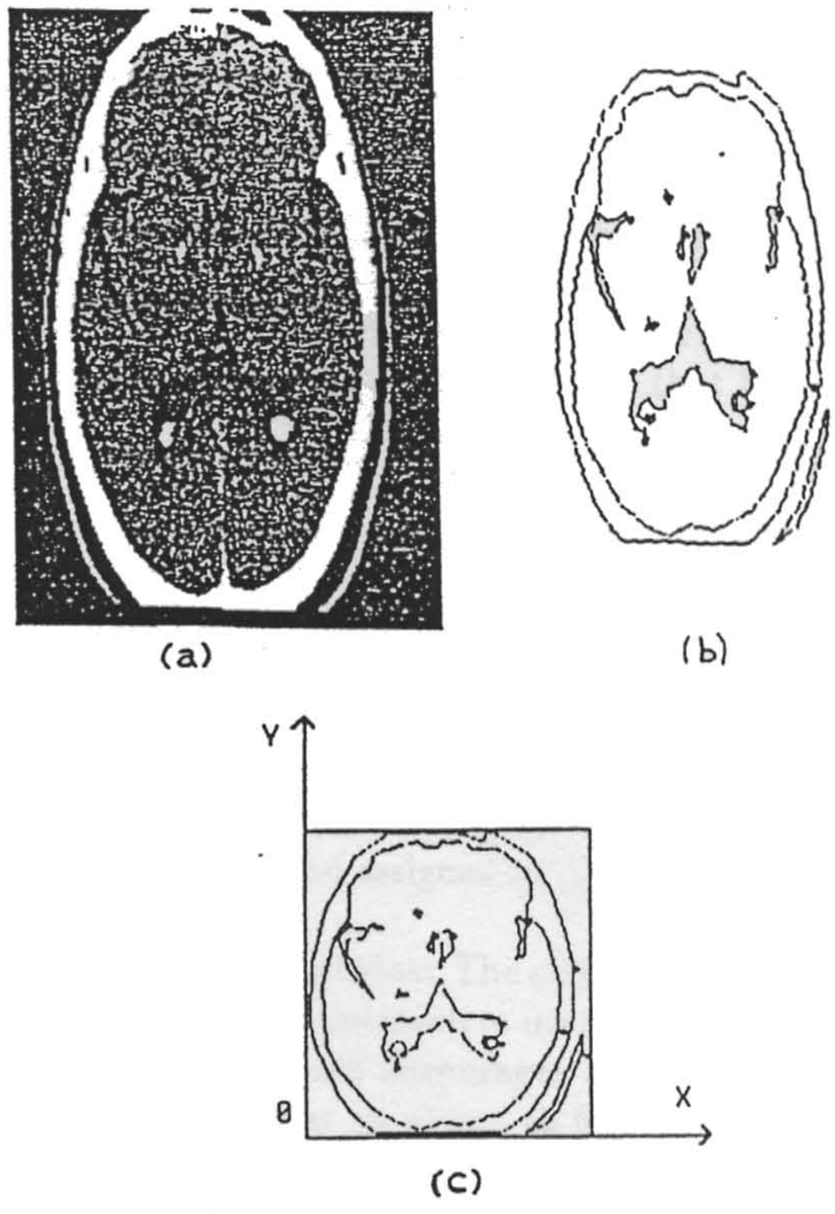

Figure 2: a) Example image. b) Segments of example image. c) Internal representation

The problem here is analogous to the problem of object recognition and localization in Computer Vision, where one must find the translation (dX,dY), rotation, scaling factor (Sx,Sy), and identity of object(s) of interest. The first five parameters can easily be identified using a reference contour for a specific image. First, moments are used for the determination of the orientation. The correctly oriented picture is scaled in the X and Y direction (Sx,Sy) and translated so that the minimum enclosing rectangle of the reference segment (outer body contour) becomes a square of fixed size and whose lower-left corner is the origin of the coordinate system. This constitutes the new IDB representation of the picture and will be used for the retrieval of images based on pictorial content. Spatial relationships between segments are not needed because the spatial reasoning is done in absolute coordinates regardless of distortions, translations and rotations of our image. This reduces dramatically the combinatorial complexity of the matching process, and the representation without relationships is more suitable for machine learning algorithms.

Regions, edges and other features are combined to generate image descriptions (see Fig. 2). Selected features that can be used to represent the content of an image for image database applications are :

1. Segments (or objects): characterized by their absolute dimensions (Xmin, Xmax, Ymin, Ymax) ; area ; perimeter ; position ; orientation (moments) ; shape factor ; covering with its enclosing rectangle ; compactness ; elongation ; Fourier descriptors (for closed segments); texture class ; gray value ; color ; gradient etc.

2. Parts of segments: segments are reduced to points (linear segment midpoints, skeleton end-points, high curvature points, and other characteristic points) and may be characterized by a list of attributes such as length, orientation, position, gray value, gradient etc.

3. Relationships: The only relationship among objects used is the contain or inclusion relationship. Given a "normalized" representation, spatial relationships like relative orientation, minimum distance, and relative position are not needed explicitly.

Images can be regarded as consisting of parts (segments) that in turn are composed of subparts, which have their own properties, etc. Thus an image can be described as a hierarchical structure using classes of "segments" that contain other classes of segments at a lower level of abstraction. The multilevel description

of the image has the advantage of modularity and incremental construction of the system. This means that we can begin from a flat structure and incrementally add more and more attributes and features at different levels of abstraction. It is obvious that this kind of representation of image content uses complex internal structures and may require a larger number of properties.

3.3 Retrieval - matching

Matching, i.e. finding a correspondence between a query image and a set of images in the database, even with the use of very simple attributes that are easy to compare (simple similarity metrics) [9], very efficient matching techniques [10] working at the level of edge segments, techniques which use hierarchical decision-tree matching, or a matching process which is based on hierarchical structures that capture increasing levels of detail about the objects of interest, leads to a combinatorial explosion in the case of large image databases (100.000 to 1.000.000 images). Therefore, the organization of the database should be such as to facilitate matching and retrieval operations. Specifically:

- The number of possible candidate images should be minimized as much as possible. Pictures which are "similar" should be clustered together. Documents will be classified according to a hierarchy (classes) at various levels of specificity in order to partition efficiently the image database.

- The number of segments that would be checked for correspondence between the query image and the set of possible candidate images should be kept to a minimum. Thus, we do not try to match every segment in the image, but use prior knowledge to restrict matching to a high level of abstraction (not full depth matching) for parts of the image or to match a limited subset of the image segments (try to match "differences"). Models should be hierarchical and object knowledge could be used to spatially constrain the search in the image, and limit the number of image abstractions. One can also limit the number of attributes for specific segments (only relevant and discriminant attributes are used).

Modules that use specific techniques for matching could be implemented using the OO paradigm and activated using message passing at different levels of abstraction (levels of the hierarchy).

Below we consider the matching procedure :

Low level matching: Segments are reduced to points The use of spatial access methods is very fast and easy to implement when the dimension of the space (number of attributes) is not very large.

Matching metrics: Weights may be used to define how relevant an attribute is and the importance of a specific segment in a given query. In general, a significant weight is associated with segments that belong to the UNKNOWN class (pathology).

Complexity: The use of an image representation that does not use explicit relationships between segments reduces dramatically the complexity of the matching process. The complexity is O(I*S) where I is the number of candidate images and S is the number of segments used for the retrieval (usually 1–3 segments). The matching process can also be parallelized, i.e. the search space can be partitioned and assigned to multiple processors.

Possible query scenarios: The object-oriented model is associated with an interactive user-friendly environment. The model itself encourages and simplifies the concept of pointing at an object to find out the allowable operations on the object and then select a desired operation. A number of different modes of access are outlined below:

- Access to an object through its class.
- Access through the aggregation hierarchy (e.g. requests for descendants).
- Access through relationships (relationships can be used to move from one object to another).
- Access through specific features of intrinsic data (e.g. search for specific patterns).
- Combinations of the above.

In most cases users don't know what they are looking for. When a user opens a document ("object") he can look at it at various levels of abstraction (resolution), examine iteratively lower levels of abstraction, navigate to other objects etc. A user needs query mechanisms which support "opening up" complex objects and searching for qualifying parts of the object without having to have a different operator for each particular search.

To support the above, *indexing* collections of objects leads to extremely fast retrievals by avoiding long, sequential searches. *Clustering* allows objects to be physically placed on disk in a way that minimizes retrieval time for specific data access patterns.

4 KNOWLEDGE REPRESENTATION

4.1 Data representation

The data representation model used in our work is the Frame model, which is one of the essential knowledge representation techniques used in AI [11]. The primitive element is the unit. Each unit has a name and consists of a number of slots, which in turn consist of a list of aspects called facets. Slots are used for describing the unit they belong to (attributes). Facets are needed for proper specification of the slot and its value. Thus units are formed by an aggregation of slots and facets. A unit can be a class-object or a member-object. Relations (links) relate member-objects to class-objects. As data structures, frames benefit from slot access procedures and value inheritance. The frame is a control mechanism as well as a data representation. The slots may have constraints of their own (cardinality, value class, etc) or constraints that reference other slot instances or other objects implemented in the form of active values or rules (applicable rules facet). Figure 3 shows some slots of a segment instance description in KEE (Knowledge Engineering Environment).

```
Unit : S1.1
Superclasses : CLOSED.SEGMENTS
Member of : SKULL
-----------------------
Own Slot : HISTORY from SEGMENTS
 Inheritance : OVERRIDE.VALUES
 Value Class : (UNION IMAGE.PROCESSING IMAGE.DISPLAY)
 Cardinality.Max : 100
 Variable Type : MULTIPLE.ARGUMENTS
 Values : FMOY (FFIL 20) (TEXT 5)

Own Slot : TEXTURE from SEGMENTS
 Inheritance : OVERRIDE.VALUES
 Value Class : (ONE.OF BLANK SHADED STRIPED CROSSED WAVY)
 Cardinality.Max : 1
 AvUnits : TEXTURE.AV
 Confidence : .9
 Time Stamp : Nov 1 09:20:03
 Variable Type : NOMINAL
 Applicable Rules : TEXTURE.RULES
 Values : SHADED

Own Slot : GRAY.LEVEL from SEGMENTS
 Inheritance : OVERRIDE.VALUES
 Value Class : #[Interval: [0 255]]
 Cardinality.Max : 1
 AvUnits : GRAY.LEVEL.AV
 Confidence : .9
 Time Stamp : Nov 1 09:20:04
 Variable Type : LINEAR
 Applicable Rules : GRAY.LEVEL.RULES
 Mean Value : 155
 Synonyms : LUMINOSITY
 Values : 203

Own Slot : DISPLAY from SEGMENTS
 Inheritance : OVERRIDE.VALUES
 Value Class : (UNION IMAGE.PROCESSING IMAGE.DISPLAY)
 Variable Type : MULTIPLE.ARGUMENTS
 Values : (DS 256 256 1 10 W)
```

Figure 3: Some slots of a segment instance description.

4.2 Rules

The system should allow the integration of background knowledge (knowledge about the specific domain) such as constraints, assertions that describe a class, procedures and production rules that change and manipulate the schema. The use of prior knowledge will lead to the elimination of many search paths and will reduce the system response time during retrievals.

Knowledge is encapsulated in if-then rules which interact with the body of data [12]. Rules are also considered as objects. Because of the large number of rules, additional control has been added by structuring the rule base into classes, and attaching methods to specific classes of the system via the APPLICABLE.RULES facet. Using rules, we can build several models for a specific class of images.

Classification rules: They use a priori knowledge to classify segments of the image [13].
Consistency rules: Knowledge is primarily used to check the consistency of our data. This class of rules applies spatial and contextual constraints to update or modify values of slots for the segments of a specific image and to check the consistency of the database schema. They can also be used for constraint propagation or dynamic classification (database schema evolution).
Strategies: Strategies are control structures that will also be implemented in the form of rules. They will be used to encode knowledge about which kind of methods to apply and in what order (e.g. sequence of image analysis tasks).
Retrieval rules: These rules encode knowledge and retrieval strategies for a specific class of images. Retrieval strategies reduce the complexity of a query graph

4.3 Active values

Active values are pieces of code which are executed when a specific change in data (slot value; see Fig. 3) takes place. We can use this code to check for consistency, update something in the database or simply for debugging purposes.

The combination of active values with rules can facilitate the construction of the system's knowledge-base. An active value can be attached to specific data in our system, and when these data change the necessary rules are triggered so that the state of our knowledge will be always consistent.

5 SYSTEM DESCRIPTION

5.1 Logical organization

The database is organized hierarchically, with only large-grained or high level of abstraction objects at the top level. Figure 4 shows the organization of the system and Figure 5 shows the hierarchy of image and segment classes. The high level of abstraction is on the left and the low level of abstraction is on the right in the figure. For clarity, only one class is expanded at each level of the hierarchy.

IMAGE.PLANE : Image classes are defined depending on the part of the body and the image plane position and orientation [14]. The lower levels represent classes based on image content, and classes of image segments for a specific class of images.

SEGMENTS : The instances of this class represent image segments (refer to previous paragraph on "image representation" (attributes : slots).ACCESS : defines the different classes of users, time and location organization ; data, procedures (methods, rules etc) and parameters (for display, processing and retrieval) can be inherited to a specific class of image content.

IMAGE.TOOLS : procedures for the display and processing (interactive and automatic) of images are considered as complex objects (methods + parameters + rules).

MODALITY, OPERATIONS : like ACCESS the data, procedures and parameters can be inherited to a specific class of image content.

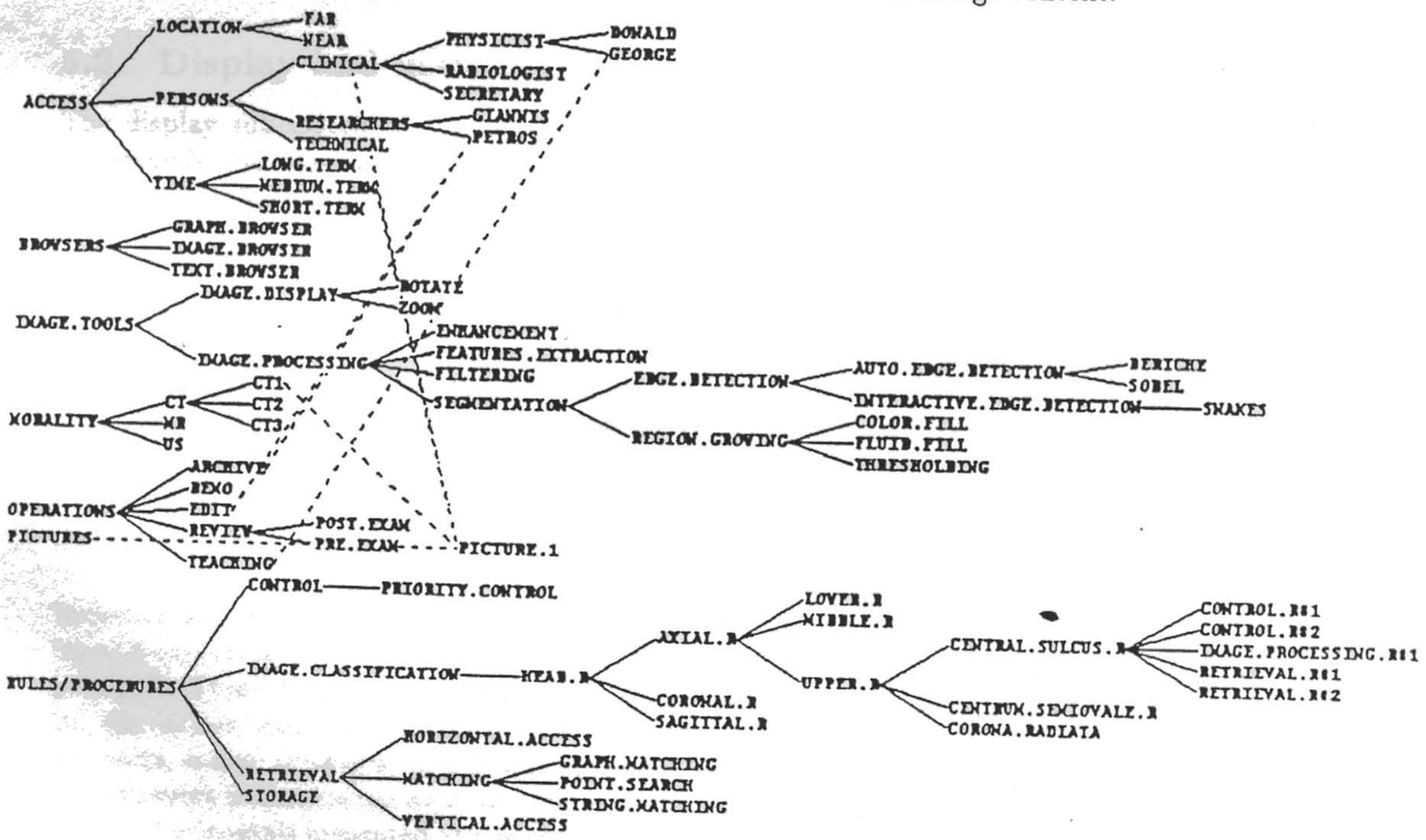

Figure 4: Organization of the system.

Figure 5: Hierarchy of image and segment classes.

RULES/PROCEDURES : see previous paragraph on "knowledge representation".

Control: We use a simple control mechanism. In order to alleviate the overhead of many choices for the user, the system has a priority control that defines the most important functions that will appear on the menus or will be activated automatically.

Memory Organization :

Long term: This part of the system contains descriptions of all permanent objects (definitions of classes, rules, methods etc) and is the "static" part of the system. For efficiency it is resident in virtual memory [7].
Short term: This part of the system includes the large number of objects that are instances of a class (leaves) and is the part of the system that changes dynamically. For space efficiency a DB implementation will be used.

5.2 Display and user interface

The display sub-system can be built around a family of methods and messages; each method (routine) may consist of an iterative scheme for scanning and accessing the object using messages, and the code needed to display the object(s) on the screen(s). A small set of display functions can be used on a wide variety of objects. The use of virtual objects enhances and simplifies the display system. Rather than having options in the display routine, a virtual object can be created that captures the desired functionality. The different parameters of the display routines could be stored in the database hierarchy.

Browsers are an interactive query mechanism for finding and exploring objects and relationships between objects [15]. Three different types of browsers can be used for the access and retrieval of images: text oriented browsers, image browsers, and graph browsers. All of the browsers maximize the amount of information (vertical or horizontal) presented at one time and provide a means to interactively apply functions to select objects.

Text browser: the text oriented browser works with any object and provides an easy interface to applying functions interactively. The display can be done horizontally : display one object at a time with one field per line, maximizing the depth of information displayed about a particular object, or vertically when we are looking for a specific set of objects.

Image browser: the image browser displays reduced resolution pictures of objects. Usually a text browser is used to narrow down the number of images to browse with an image browser.

Graph browser: the graph browser displays objects and the relationships between them (e.g. parent and child links, relationship objects etc). KEE has built-in text and graph browsers.

5.3 Retrieval scenario

Retrievals of images based on their content, supported by the system under development, proceed in four steps:

1. Formulation of a query based on the definition of segments which are representative of image content (see section 3.1).

2. Narrowing of the search space by definition of classes and subclasses using a variety of tools. Tools which have been developed include: direct input of image class (manual or text browser based), a graphical definition of the image plane, a browser based on a hierarchy tree, automatic class selection (using machine learning techniques) and selection of the most likely class based on comparison of reference query segments with segments of representative images of each class.

3. Low level matching to retrieve specific images by content.

4. Display or listing of retrieved images on a visual or text browser. As mentioned earlier, the system (through its image processing module) also supports further interactive processing of retrieved images.

A typical screen of the system is shown in Figure 6.

6 CONCLUSION

An object-oriented approach to the flexible design and incremental implementation of image database systems has been presented. Using this approach, a system has been developed and implemented in which the user interface, the image database, and the image retrieval modules are fully integrated. The object-oriented approach has permitted an implementation which is incremental in that, at first, only a hierarchy tree browser was implemented and subsequently more features were added. These features include a reference image library, and mechanisms for integrating knowledge for each image class (e.g. image processing parameters and operators, retrieval strategies and rules, etc.). The system, which is currently in the final stages of development and evaluation in our laboratory, is highly interactive and facilitates the quick and intelligent access of images based on their content. Finally, such a system can be seen as a standalone system as well as a layer of a PACS and offers image access tools that are not routinely available on a PACS.

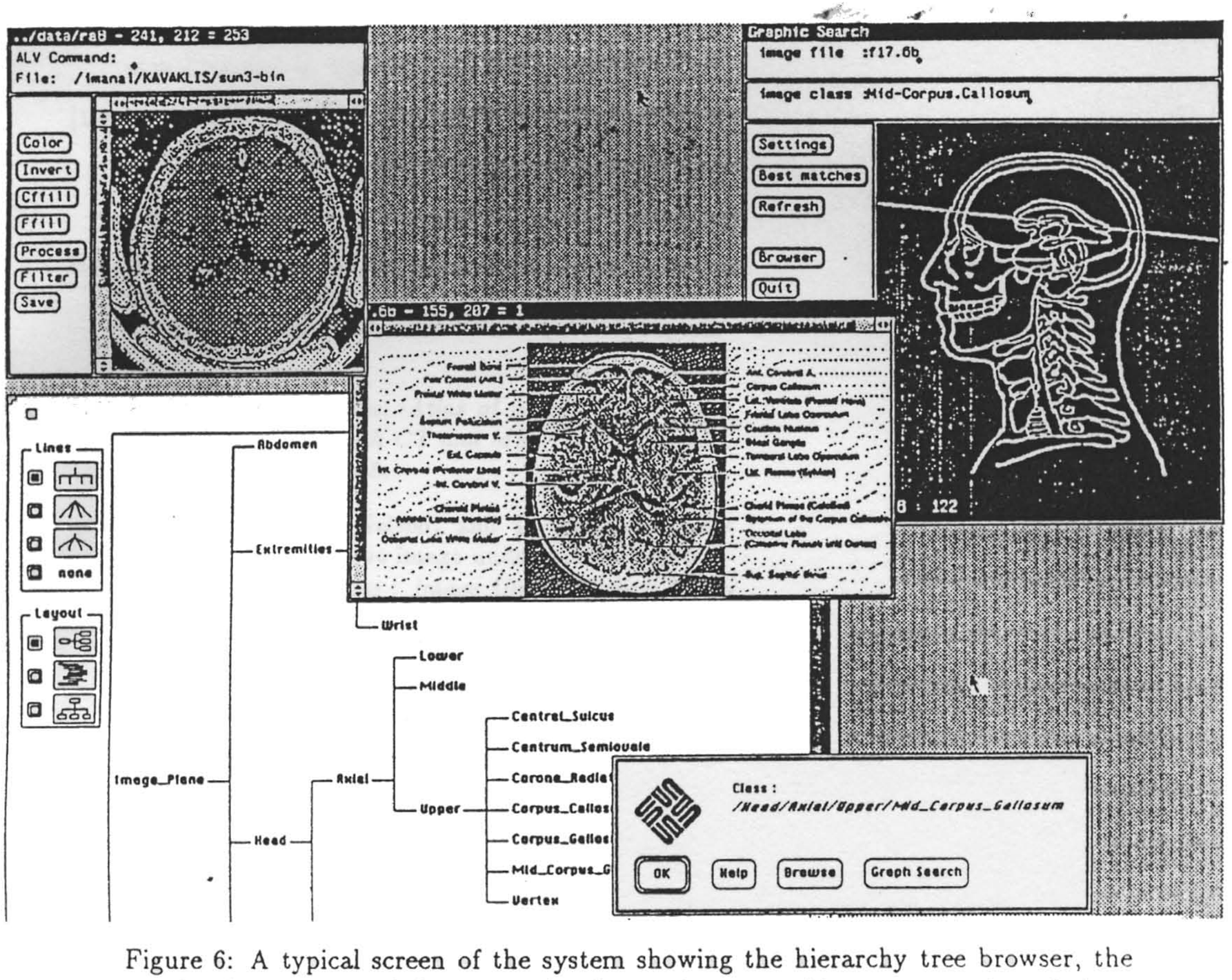

Figure 6: A typical screen of the system showing the hierarchy tree browser, the graphical image plane definition tool, reference image, and image processing window.

ACKNOWLEDGEMENTS

This research was supported in part by the Commission of the European Communities under project AIM-1008. We would like to thank the members of the PRIMIS group (AZ-VUB Brussels) and the Department of Diagnostic Imaging, Yale University School of Medicine, for supplying most of the medical images.

References

[1] S. C. Orphanoudakis, E. G. Petrakis, and P. Kofakis. A medical image database system for tomographic images. In *Proceedings of the International Symposium CAR'89*, pages 618–622, Springer - Verlag, Berlin, June 25-28 1989.

[2] Kim W. Woelk D. Multimedia information management in an object-oriented database system. In *Proc. of the 13th VLDB Conference*, pages 319–329, Brighton, 1987.

[3] L. Mohan and R.L. Kashyap. An Object-Oriented Knowledge Representation for Spatial Information. In *IEEE Trans. on Software Eng.*, pages 675–681, 1988.

[4] F. Banchillon. Object-Oriented Database Systems. In *ACM SIGART/SIGMOD/SIGACT, Proceedings of the 7th Symposium on Principles of Database Systems*, pages 152–162, Austin, Texas, March 1988.

[5] Nguyen G. and Rieu D. Schema evolution in object-oriented database systems. In *INRIA research report No 947*, December 1988.

[6] I.R. Young D.A. Bell, D.H.O Ling. Medical image databases : state of the art. In *Internal document , HIPACS AIM project deliverable 01*, August 1989.

[7] Thatte S.M. Persistent memory : a storage architecture for object-oriented database systems. In *1986 Int. workshop on Object-Oriented Database Systems*, pages 148–158, Asilomar-California, Sept. 1986.

[8] Gupta R., Cheng W.H., Gupta R., Hardonag I., and Breuer M. An object-oriented VLSI CAD framework : a case study in rapid prototyping. In *IEEE-Computer*, pages 28–36, May 1989.

[9] D.H. Ballard and C.M. Brown. *Computer Vision*. Prentice Hall, 1982.

[10] Ayache N. Efficient registration of stereo images by matching graph descriptionms of edge segments. In *INRIA research report No 559*, August 1986.

[11] M. Mitschang. Towards a unified view of design data and knowledge representation. In *Proc. of 2nd Int. Conf. on Expert Database Systems*, pages 33–49, Sheraton Premiere at Tysons Corner, VA, 1988.

[12] B.A. Draper, R.T. Collins, J. Brolio, A.R. Hanson, and E.M. Riseman. The Schema System. In *International Journal of Computer Vision, 2*, pages 209–250, 1989.

[13] D.M. McKeown, W.A. Harvey, and J. McDermott. Rule-based interpretation of aerial imagery. In *IEEE Transactions on PAMI, Vol. PAMI-7, No. 5*, September 1985.

[14] A. John Christoforidis. *Atlas of Axial, Sagittal, and Coronal Anatomy with CT and MRI.* W.B. Saunders Co, Philadelphia, 1988.

[15] C. McConnell, P. Nelson, and D. Lawton. Constructs for cooperative image understanding environments. In *Proceedings : Image Understanding Workshop*, pages 497–506, February 1987.