FAST LEARNING AUTOMATON-BASED IMAGE EXAMINATION AND RETRIEVAL

B. John Oommen and Chris Fothergill

TR-208, JUNE 1992

School of Computer Science, Carleton University
Ottawa, Canada, KIS 5B6

FAST LEARNING AUTOMATON-BASED IMAGE EXAMINATION AND RETRIEVAL*

B. John Oommen and Chris Fothergill
School of Computer Science
Carleton University
Ottawa , ONT ., K1S 5B6
CANADA

ABSTRACT

In this paper we study the Image Examination and Retrieval Problem (IERP). Consider the scenario in which a user wants to browse through a database of images so as to retrieve a particular image which he is interested in. Rather than specifying the target image textually, we instead permit the user to access his image by using his *subjective* discrimination of how it resembles other images that are presented to him by the system. The IERP is not merely viewed as one involving recognition or classification, but instead as one that falls in the domain of classifying and partitioning the *set of images* in terms of their "visual" resemblances. In the process, we intend to not merely find images that match other images, but, in fact, to group all similar images together so that subsequent searches will be enhanced. The intelligent partitioning of the image database is done adaptively on the basis of the statistical properties of the user's query patterns. This is achieved using learning automata and does not involve the evaluation of any statistics.

I. INTRODUCTION

There currently exist, in many institutions, large collections of images with historical and scientific significance. For example, the Public Archives of Canada has over ten million photographs. Other organizations such as the National Air Photo Library, the National Aeronautics and Space Administration, the New York Public Library, and various university departments throughout the world, maintain large collections of photo, film, and microfiche images. Due to the large numbers of images, it is incredibly expensive and time-consuming to access these data sets, and retrieve specific information. This is called "*The Image Examination and Retrieval Problem*" (IERP). In this paper we shall present a new learning automaton solution for the IERP.

Image retrieval can be broken down into two components. There is the problem of indexing the image database so that a search can be done for any given element. There is also the problem of appropriately presenting the selected images to the user to confirm the correctness of the search.

Image presentation primarily depends on the technology used. One solution is that the images be represented by a digitized equivalent that can be viewed using computer monitors. Although

* Partially Supported by the Natural Sciences and Engineering Research Council of Canada. A preliminary version of this paper will appear in the Proceedings of the 1992 Second International Conference on Automation, Robotics, and Computer Vision, Singapore, September 1992.

there is a large conversion cost, this cost can be justified because the images may be converted in order to preserve them longer. Many older images will probably be digitized and stored on a compact medium such as CD-ROM or video disk. Also, in all likelihood, current images are in a digitized format. Thus any indexing system will have access to the digital format of many images.

This paper is concerned with the second and more important aspect of image retrieval, namely the indexing and searching of images in the database. Regardless of image storage technology, efficient methods of indexing and searching large numbers of images will have to be developed. In order to search for an object in any database, some form of search mask has to be specified. This requires that the database has a known indexing structure. In an analogous setting, text databases have a fixed format for each member record and the search mask is based upon the fields of the record. The constraints on valid field values are always known. For example, one particular field may be restricted to integers ranging from one to one hundred and another field to all valid names of 10 letters or less. The user forms a search mask with specific letter and number patterns. This mask takes the form of a text record which is in the same format as all the records in the database. Thus the search mask and the objects in the database share the same specific format.

The only way a similar interface for image databases could be attained is to have the user draw a picture of the image sought for. The system would then search the database for all images that match the user's description. However, this does not account for searches such as "Find all images of sport's heroes.", or "Find all images related to the Gulf War." There is no drawing tool that would properly allow the user to specify all the possible types of images covering various aspects of the Gulf War. Such searches would instead require some form of textual search mask that is matched with descriptions of every picture in the database. If the phrase "a picture is worth a thousand words" is valid, then there are a thousand different ways to describe a picture.

The problem of maintaining and querying image databases is very complex [1,5,6,12,14] -- it also has potential applications in multi-media systems [7,14,17]. Due to the fact that the image is a data type in its own right, some work has proceeded to incorporate it in object oriented databases [17]. However, querying the database is a far more complex problem than just maintaining it. Often, the picture that is to be retrieved is itself analyzed so as to render the search process efficient [2,13,16]. In other systems which use fancier user-interfaces, (such as in VIEW-station) [11] the V-Server does the image memory management with an objective specification of the query.

Present approaches for solving the IERP are based on one of two techniques. They use either a textual description or a visual description for both indexing images in the database, and defining the search criteria. In some cases a combination of the two methods is used. Textual descriptions of database objects are achieved through image feature classification as follows. Initially, the images in the system are given some text-based manually-assigned index. When a user searches for an image, he must first create an index for the image, and the system matches the user's textual index with the system indices to find the target image.

Visual classifications are obtained through the use of heuristic search methods. A heuristic scheme is a technique whereby a set of choices are evaluated and assigned values that reflect a given choice's desirability. In searching a set of images the user would define his search heuristic in terms of some mathematical or psychophysical measurement that can be applied to each image. The most desirable images are then selected as the result of the search. Heuristic search methods use a variety of approaches to examine and evaluate images. All the methods attempt to quantify the visual content of images. Unlike the text-based feature classification method, these approaches do not attempt to describe the contents in some language, but rather to describe the contents of the image mathematically. The basic principle is that the system evaluates the images using various mathematical criteria set out by the user. These approaches can be classified according to which of the following methods they use : the use of exemplars for searching, the clustering of images into discrete classifications, the application of problem solving algorithms derived from artificial intelligence, or the application of the Two Domain Theory.

There are various inherent disadvantages associated with text-based methods and with any heuristic-based methods. This is why we have attempted to develop a (hopefully superior) scheme to solve the IERP. This new method is based on mapping the IERP to the Object Partitioning Problem (OPP). Consequently, we shall attempt to solve the IERP by viewing it as one that falls in the domain of classifying and partitioning the *set of images* in terms of their "visual" resemblances. The goal is not just to find images that match other images, but, in fact, to group all similar images together so that subsequent searches will proceed much faster. Furthermore, instead of using some classification method that allows us to initially stipulate the membership of the images into groups, we allow the system to adaptively decide the grouping by examining the statistical properties of the user's query patterns. To do this, an algorithm is required that uses previous access patterns to intelligently partition the whole set of images so as to minimize future user access times.

There are many advantages to this approach. Unlike classification methods, there is no pre-interrogation of images to create indexes for each image. Instead, the images are indexed by the user during system use. Also, unlike heuristic methods, a given search does not simply impose it's own search criteria temporarily upon a set of images. Instead, the image database repartitioning is maintained after each search and this repartitioning also helps improve future searches. Finally, a user's interaction with such a scheme is also easier and faster than with the other two methods. The user simply keeps a *subjective* mental picture of the image he is searching for while viewing many images. Judgement of these images is made strictly on the basis of a quick visual impression.

In this paper we shall discuss the drawbacks of both the text-based systems and also the heuristic approaches. We shall then describe our solution which associates a simple learning automaton (rule) with every image. The automaton adaptively learns how the image database should be repartitioned as the users' subjective queries are examined. The design and implementation details of this system (a prototype of which has been built) are described in detail in this paper. The paper also describes the results of this implementation.

II. PREVIOUSLY KNOWN METHODS

In this section, we shall briefly describe the current methods of solving the IERP. Subsequently, our new scheme based on object partitioning will be presented.

II.1 Image Feature Classification

Image feature classification involves the assignment of keywords or phrases to describe each image. Thus the image database is matched with an proportionately sized text database whereby each image is represented by a text record. The first step in creating the image retrieval system is for all the images in the database to be classified. This would have to be done by a team of people operating with a set of guidelines.

The categories for description can be selected in quite a few ways. For example, images can be classified by describing any or all of the following: subject matter, component elements, viewing perspective, time period, colour scheme, or location. Thus, a picture of a car that is defined with some of the above categories may be described as follows:

Subject :	Automobile
Perspective :	Side_View
Components :	Tires, Glass, Metal
Colour :	Red

This type of description is intended for searches such as the following :

Query I : "Select all images where Subject = Automobile."

Query II: "Select all images where Colour = Red and Perspective = Side_View."

The text from the user's query would be matched with the above system database description to locate the desired target image. Thus the search process is identical to that which is used in standard textual databases.

One application of the above system is in Geographical Information Systems (GIS). A GIS may contain thousands of images, where all the images are maps, or other geographically coded images such as satellite or aerial photographs. These systems use text records that describe the geographical location of an image in terms of a bounding box around the image. Special structures can be used to index these bounding boxes and non-typical methods of defining a search mask are employed but the underlying principle is the same. For example, a search may be initiated by selecting a pixel location in one image, and the system internally creates a search mask based on the geographic location of the pixel. This mask then acts as a window for the system to look into the image database and all maps whose bounding box surrounds that location can be retrieved.

Another variation on text-based searches is used for complex images in which a hierarchical semantic image index is created [13]. The image is associated with a tree-like structure where the root of the tree is a general description of the image. Each child describes the components of the child's parent. In Figure I we see an outdoor scene that is mainly composed of the sky, the ground, a tree, and a car. The ground contains grass, and rock objects. The tree consists of a crown and a trunk, while the car consists of a body, windows, and tires.

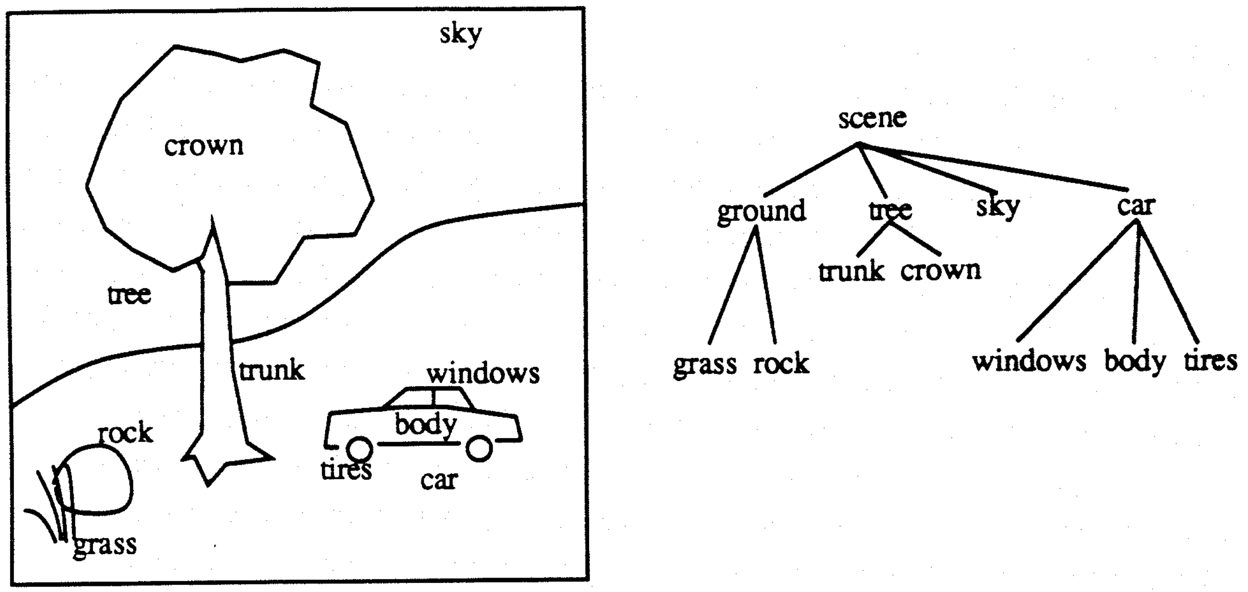


Figure I : An Image and its Semantic Hierarchical Indices

Searches in a database of this type require the user to specify the hierarchical structure of the target image, and this structure would be matched to the structures of the images in the system. Although there is now a comparison between two structural definitions of the images, the components of the structures are still text descriptors, and consequently, strictly speaking, this is just a variation on the original text-based system.

Some examples of text-based classification methods and languages found in the literature include the Extended Relational Model for Large, Flexible Picture Databases (ELF) [18], the Graphics-Oriented Relational Algebraic Interpreter (GRAIN) [2], the Multi-Sensor Image Database System (MIDAS) [6], and the Image Database System (IMDS) [5].

There are some inherent problems with text-based methods. Although the storage needed for text records is generally much less than that for images, such solutions require the system to use an additional database that must be both maintained and serviced. Also, the classification of possibly millions of images by human beings is an incredibly tedious, time consuming and expensive effort. Although guidelines for image classification may be specified, it still requires a subjective evaluation by people doing the classification, which may not match the subjective evaluation of the users of the system. Besides this, judging the visual contents of images is an imprecise task in itself. Finally, it is easy to see that the type of queries that can be performed on such systems is typically restricted to pre-defined text attributes. One is therefore unable to search for images that simply have visual similarities.

II.2 Heuristic Search Methods

Heuristic search methods use a variety of approaches to examine and evaluate images. All the methods attempt to somehow quantify the visual content of images. Unlike the text-based methods, these approaches do not attempt to describe the contents in some language, but instead attempt to describe the contents of the image mathematically. The basic principle is that the system evaluates the images using various mathematical criteria set out by the system user. These approaches can be classified according to which of the following methods they use: the use of exemplars for searching, the clustering of images into discrete classifications, the application of problem solving algorithms derived from artificial intelligence, or the application of the Two Domain Theory.

An exemplar is a restricted function that defines an image according to certain geometric properties possessed by the image. A function is used to map pixel intensities so as to assess the similarity of a collection of pixels to the desired pattern [4]. For example, a search for images of aircraft may be defined by the geometries of wing and tail connection to the fuselage. Unfortunately, the geometric description of one object can easily match the geometric description of another. For example the rectangular geometry of a book is similar to that of a desk top.

Clustering methods cluster images that possess similar pixel histograms or patterns into groups [3]. A proximity matrix would then be used for image patterns that share attributes with two or more pre-determined feature sets.

Problem solving algorithms have also been applied in solutions which aim to classify images [7]. For example, in analyzing satellite images, searching for sharp edges in images is used to distinguish human-made objects from backgrounds and natural phenomena. The problem with this approach is that the set of pictures that given algorithms distinguish between is typically, small.

The Two Domain Theory combines many of the strategies characterized in the other three heuristic methods [10]. The main difference is in how the user defines the search criteria. Instead of creating a set of machine based validity rules and meanings, the user is queried with several representative images and is asked to rate them on a scale according to the component break down of the images. The components of images include such things as lines, edges, colours, textures, and pixel distribution. Subsequently, clustering algorithms are used to select representative sets of pictures and to select subsets of images that are closest to the user's descriptors.

The problem with any heuristic-based search method is that it evaluates the visual content of the image and not the user's perception of the subject. For example the previous methods may be able to satisfy specific searches such as, "Find all images that contain aircraft.", but will fail on searches such as "Find all images related to air travel." The problem with the last search statement is that although one can mathematically define the exterior view of an aircraft, this definition does not apply to images of the inside of aircraft, stewardesses, control towers, or baggage claim carousels. The latter type of search is defined by the user's perception of the meaning of the image as opposed to solely the content of the image itself. There is also the possibility that the search method may be too complex for untrained users who do not have the time to understand various complex search mask definitions. Finally, applying mathematical formulas to analyze images, and asking users to provide image evaluation information is not only time consuming, but is also inaccurate. By it's very nature a heuristic is a best guess in evaluating the suitability of one image to match a search query. This is especially true since the heuristics involve the use of image recognition techniques which intrinsically have various "grey areas" of imprecision. Indeed, at what point does a block of pixels become an image of a house?

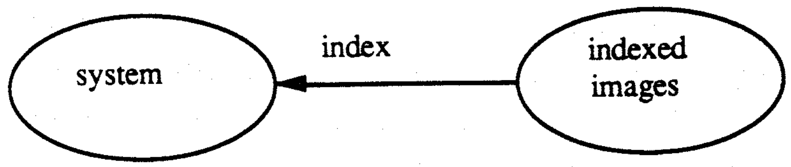
The most important drawback of both the text-based and the heuristic-based methods is the lack of intelligence required to speed up subsequent searches. The amount of time the system takes to process a given query is not decreased the next time an identical query is processed primarily because these systems do not learn, or improve with time and frequency of use.

III. A SOLUTION USING OBJECT PARTITIONING

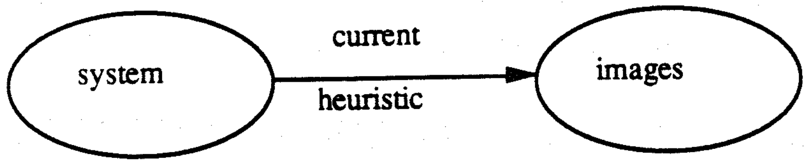
In this paper we propose that the IERP be solved by viewing the problem not as one of image recognition or image classification, but instead as one that falls in the domain of object partitioning problems. The goal is not just to find images that match other images, but to group all similar images together so that subsequent searches will proceed much faster. Instead of using some classification method which stipulates the membership of the images into groups, the system adaptively decides the grouping by extracting information about the statistical properties of the user's query patterns. The algorithm thus uses previous database access patterns to intelligently partition the whole set of images so as to minimize the response times for future queries.

There are many advantages to this approach. Unlike classification methods, there is no pre-interrogation of images to create indexes for each image. Instead, the images are indexed by the user while they are used. Also, unlike heuristic methods, a given search does not simply impose it's own search criteria temporarily upon a set of images. Instead, the database is conceptually re-partitioned after each search, and this repartitioning helps improve subsequent search times. As Figure II shows, the other two methods of image retrieval are unidirectional in the sense that the image indexing is either stagnant as in classification methods (Case A), or is only temporary as in heuristic methods (Case B). In contrast, the object partitioning method (Case C) can be described as a cyclical method of image retrieval in that the data indexing is dynamic and adaptive.

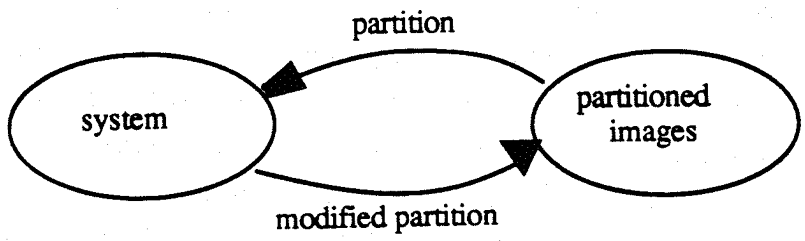
A user's interaction with an object partitioning method is also easier and faster than with the other two methods. This is because both the feature classification and heuristic methods expect the users to first translate their query into either some relational database type language, or into some mathematical and/or statistical representation. Certainly in the latter case, the methods for transforming a query into a heuristic search can be quite complicated. With the new method, these problems are avoided. The user simply keeps a subjective mental picture of the image he or she is searching for while viewing many images. Judgement of these images is made strictly on the basis of a quick visual impression.



Case A: Preclassification Method



Case B: Heuristic Method



Case C: Object Partitioning Method

Figure II : The System-Database Interaction for Three Image Retrieval Methods

The object partitioning problem (OPP) can be defined as follows: Let $A = \{A_1,...,A_W\}$ be a set of W objects to be partitioned into K classes $\{\pi_1,...,\pi_K\}$. The objects are partitioned such that the objects that are frequently accessed together lie in the same class. This implies that there is a correct partitioning, called a grouping, for the objects. In the most straightforward form of the OPP, objects are accessed in pairs $<A_i, A_j>$. A pair of accessed objects will be called a "Query". It is assumed that the objects within the same grouping are accessed more frequently together than with objects in other groupings. Thus, the solution to the OPP will process a sequence of queries so as to increase the likelihood that the pair of objects accessed will reside in the same class the next time an identical query is formed.

Throughout this study we consider the special case of the OPP where all the classes are of equal size. This problem is referred to as the Equi-Partitioning Problem (EPP). All objects are initially placed randomly into the various classes. It is assumed that the number of classes, K, is equivalent to the number of groupings of objects. In specific cases where it is not possible to conveniently equally partition objects into groups, some groups will simply have objects that do not correspond to an image. These objects have a NULL value.

There are a few well-known solutions to the EPP. The best non-automaton solution is the method due to Yu *et. al.* [19]. In this method, initially each member object, A_i, is associated with a real number X_i. As each query $<A_i, A_j>$ is processed, the quantities $\{X_i, X_j\}$ are moved towards their centroid by some amount δ_1. Then, a random pair $<X_p, X_q>$, $(1 \leq p,q \leq W)$ is moved away from its centroid by some amount δ_2. The result is that similar objects will gradually cluster together and dissimilar objects will separate.

III.1 Learning Automata Fundamentals

Our solution to the problem involves learning automata. Learning automata have been used in the literature to model biological learning systems and also to learn the optimal action which an environment offers. Learning is achieved by interacting with the environment and processing its responses to the actions that are chosen. Such automata have various applications such as parameter optimization, statistical decision making and telephone routing [10,19,20]. Since the literature on the field is extensive, we refer the reader to an excellent book on the field by Narendra and Thathachar [20] for a review of the families and applications of learning automata.

The learning process of an automaton can be described as follows: The automaton is offered a set of actions by the environment with which it interacts, and it is constrained to choose one of these actions. When an action is chosen, the automaton is either rewarded or penalized by the environment with a certain probability. A learning automaton is one which learns the optimal action, which is the action that has the minimum penalty probability. Hopefully, the automaton will eventually choose this action more frequently than other actions.

Stochastic learning automata can be classified into two main classes : (a) Fixed Structure Stochastic Automata (FSSA) and (b) automata whose structure evolves with time. Some examples of the former type are the Tsetlin, Krinsky and Krylov automata [9,15,20]. Although the latter automata are called variable structure stochastic automata, because their transition and output matrices are time varying, in practice, they are merely defined in terms of action probability updating rules [20].

A FSSA is a quintuple (α, Φ, β, F, G) where :

(i) $\alpha = \{\alpha_1, ..., \alpha_R\}$ is the set of actions that it must choose from.

(ii) $\Phi = \{\phi_1, ..., \phi_S\}$ is its set of states.

(iii) $\beta = \{0, 1\}$ is its set of inputs. The input '1' represents a penalty.

(iv) F is a map from Φ x β to Φ. It defines the transition of the state of the automaton on receiving an input. F may be stochastic.

(v) G is a map from Φ to α, and determines the action taken by the automaton if it is in state ϕ_i. With no loss of generality G is deterministic [15,20].

The selected action serves as the input to the environment which gives out a stochastic response $\beta(n)$ at time 'n'. $\beta(n)$ is an element of $\beta = \{0,1\}$ and is the feedback response of the environment to the automaton. The environment penalizes (i.e., $\beta(n) = 1$) the automaton with the penalty c_i, which is action dependent. On the basis of the response $\beta(n)$, the state of the automaton $\phi(n)$ is updated and a new action chosen at (n+1). Note that the $\{c_i\}$ are unknown initially and it is desired that as a result of interaction with the environment the automaton arrives at the action which presents it with the minimum penalty response in an expected sense.

The best known solution for the EPP is the Object Migrating Automaton (OMA) proposed by Oommen and Ma [9]. The OMA moves all the W objects around within its states as opposed to the automata simply moving from one state to another. Thus, when applied to the EPP a solution is not defined by the current state of the OMA, but instead by the entire structure of the automaton. The mapping function defining the transition of the automaton from one state to another, essentially defines the motion of two or three objects within the structure of the automaton. Whenever a query $<A_i, A_j>$ is received, if A_i and A_j are in the same class, they are both rewarded and moved towards the state which represents the state of *Maximum* Certainty for that class. This motion is executed one step at a time. However, if A_i and A_j are in different classes, they are moved towards the states of *Minimum* Certainty for their respective classes one step at a time, and whenever one of them is in the state of *Minimum* Certainty, it migrates towards the corresponding state of the other action. We shall later adapt this philosophy for the IERP and demonstrate how the migration can be executed efficiently so as to ensure an excellent partitioning.

The OMA is extremely accurate and fast. Experimentally, it converges to the true solution all the time, and does so in a speed which is an order of magnitude faster than the scheme due to Yu *et. al.* [19] especially when all the objects are initialized to be in the boundary states of their respective classes (the states of *Minimum* Certainty) instead of initializing to random states.

IV. THE IMAGE DATABASE LEARNING AUTOMATON

The learning automaton presented in this paper is called the Image Database Learning Automaton (IDLA) which is an interesting generalization of the OMA. The differences between the OMA and the IDLA will be clarified later, but suffice it to mention that the user does not have to describe his queries explicitly. The philosophy of the OMA is applied to the IERP by assuming that once the images are properly sorted and indexed, there is an underlying unknown grouping. For example, a given set of images based on transportation may divide into four image groups. These groups may represent cars, airplanes, boats, and trains. At system start-up these images may be randomly scattered amongst the four available partitions. As the system is used, and the user selects images by informing the system that he considers various images to be similar to his target image, the IDLA will utilize the "similarity" between these images intelligently. Consequently, the images will be migrated and partitioned in a fashion similar to the underlying grouping.

The IDLA deals with images that are partitioned equally into groups. These groups can be conceptually considered to be sub-databases, but in actuality, these sub-databases merely contain pointers to the images within their groups. Each image is thus represented by its abstract representation which is merely an index and a pointer which points to the location of the actual image. Effectively, the IDLA attempts to partition the abstract images based on the subjective information about the actual images themselves. Whenever an abstract image is in a certain action, it is equivalent to the image being in the class represented by that action. Whenever the search for the target image is initiated, the user is presented with a single image from each class which is the one closest to the state of *Maximum* Certainty of that class. The user then reports whether any of these is the target image. He also informs the system which of them is similar to the target image. This information is then utilized by the IDLA to reward or penalize the individual abstract images which are "moved" to further support this subjective information.

It is clear from the above discussion that the different states within a given class (action) quantify how certain we are that a given image belongs to that class. At system start-up all images are placed in the boundary state of their initially randomly chosen classes indicating that the system is initially uncertain of the placement of all the images. As the system is used, certain images will be rewarded for their being present in a given class and will move towards the most internal state of the class, indicating that the system is more certain that the images belong there. Likewise other images will be penalized and are either moved towards their boundary state or to another class, indicating the system's uncertainty in the appropriate class for a given image.

For the rest of this section we shall not differentiate between the actual images and their abstract representations. Let $\mathbf{A} = \{A_1,...,A_W\}$ be the set of images in the database. These images are to be partitioned into K groups. Let X be the user's target image. We define the Image Database Learning Automaton (IDLA) as a 6-tuple as below :

$$\left(\mathbf{A}\ ,\ \{\phi_1, \phi_2, ..., \phi_{KN}\},\ \{\alpha_1, \alpha_2, ..., \alpha_K\},\ \{0, 1\},\ Q(\cdot\ ,\cdot),\ \ G(\cdot) \right),$$ where,

(i) **A** is the set of images.

(ii) $\{\phi_1, \phi_2, ..., \phi_{KN}\}$ is the set of states.

(iii) $\{\alpha_1, \alpha_2, ..., \alpha_K\}$ is the set of K actions, each representing a certain class into which the elements of **A** must fall.

(iv) $\beta = \{0, 1\}$ is its set of inputs. The input '1' represents a penalty.

(v) Q, the transition function is quite involved and will be explained in detailed presently.

(vi) To describe the function G, we partition the set of states for the various classes. For each action α_j, there is a set of states $\{\phi_{(j-1)N+1}, ..., \phi_{jN}\}$, where N is the depth of memory and $1 \leq j \leq K$. Thus,

$$G(\phi_i) = \alpha_j \qquad \text{if} \quad (j-1)N + 1 \ \leq\ i\ \leq\ jN \qquad (1)$$

Observe that since this means that the automaton chooses α_1 if it is in any of the first N states, it chooses α_2 if it is in any of the states from ϕ_{N+1} to ϕ_{2N} , etc. The states of the automaton are automatically partitioned into groups, each group representing an action.With no loss of generality, we assume $\phi_{(j-1)N+1}$ to be the most internal state of action α_j, which is the state of *Maximum* Certainty, and ϕ_{jN} to be the boundary state, which is the state of *Minimum* Certainty.

As in the case of the OMA, we have **all** the elements of **A** move around in the states of the machine. If the image $a_i \in \mathbf{A}$ is in action α_j, it signifies that it is in class j (or in the jth. sub-database). Observe too that if the states occupied by the images are given, the actions chosen by the images of A can be trivially obtained using (1) above, and these specify the partition currently chosen by the IDLA.

Let $\xi_a(n)$ be the index of the state occupied by $a \in \mathbf{A}$ at the nth time instant. Also let,

$$\Xi(n)=\{\xi_a(n) \mid a \in \mathbf{A}\ \}.$$

Based on $\Xi(n)$, let $\Pi(n)$ be the current partition which the automaton decides. Observe that, as mentioned above, this partition is trivially obtained by repeatedly using (1) above on the various elements of $\Xi(n)$ to determine the action chosen by each image $a \in \mathbf{A}$. Using this notation we shall later describe the transition map of the IDLA. However, in what follows, in the interest of brevity, we shall omit the reference to the time instant 'n'.

We now briefly describe the actual operation of the prototype system. The user is always presented with a set of images on the view screen. At the initiation of a query, this set represents an overview of all the available classes of images. This overview set consists of *one single image* from each class, where that image is the one closest to the state of *Maximum* Certainty for the respective class. As the query search proceeds, the set presented to the user represents all the images of a particular chosen class.

Figure III illustrates the partitioning and viewing of images. In this case, four classes each containing four images are shown. In the centre of the diagram is an overview of the classes, in which one image from each class is displayed. These representative images are not *permanent* members of their classes -- they will be the ones closest to the states of *Maximum* Certainty.

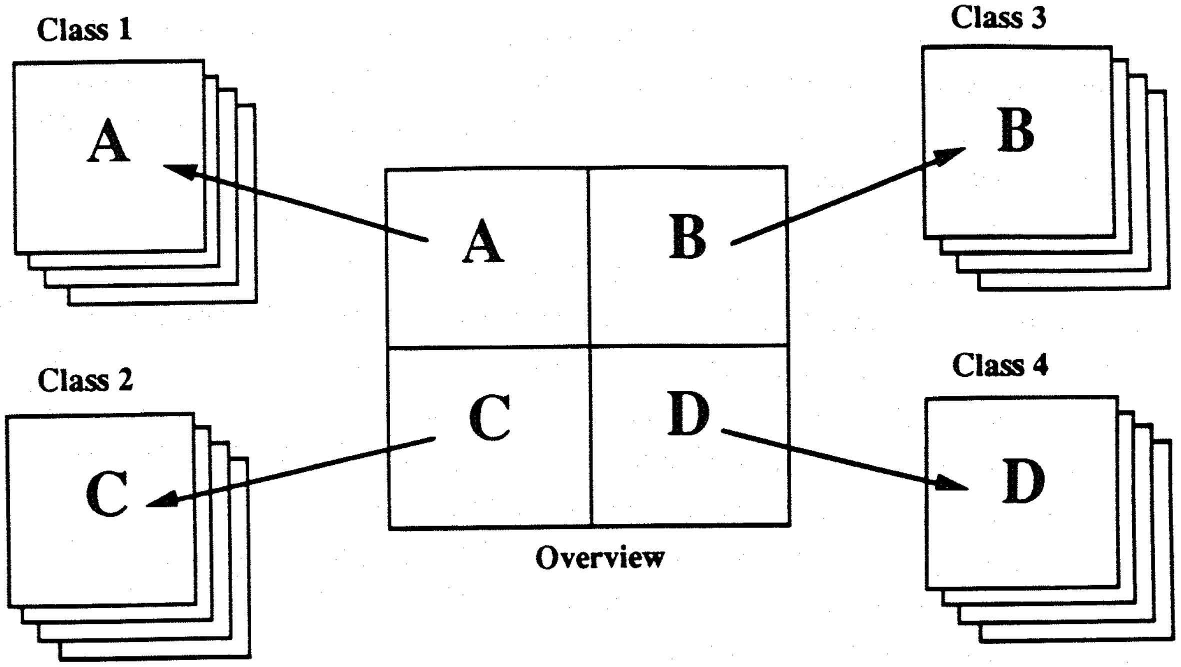


Figure III : An Example of the Partitioning and the Viewing of the Images

Initially, the IDLA begins its learning process by starting from an arbitrary partition $\Pi(0)$, which is obtained randomly or otherwise. Thereafter, for every query, the user is first presented with *ONE* representative image from each class -- the one closest to *its* most internal state. The user tags $M \geq 1$ of these images to be similar to X, his target image. Let $S = \{S_1, ..., S_M\}$ be these tagged images. He also associates a confidence value, λ_i, to his evaluation of each tagged image, S_i. The value λ_i is a real number in the interval $(0,1]$, and can be considered to be an objective measure of the user's confidence in the tagged image, S_i, being similar to X. It is thus a measure of the confidence with which he is willing to assert that X and S_i are similar.

Clearly, if one of these, say S_p is his target image, X, the search is terminated. If none of these is the target image, the images in the individual classes represented by the the elements of S are presented to the user, in the decreasing order of the $\{\lambda_i\}$. At every juncture, S, the set of similar images is updated and the corresponding confidence values associated with the images which are also recognized to be "similar" are also updated. Thus, at every instant, the IDLA is aware of the fact that all these elements are considered to be subjectively similar to each other, and the "degree of subjectivity" is also available.

The system utilizes this information as follows. Every pair $<S_i, X>$ is now considered by the IDLA. If they are both in the same class, they are both rewarded with a probability λ_i. If, however, they are in different classes, they are both penalized with a probability λ_i. Subsequently, every pair $<S_i, S_j>$ in S is considered by the IDLA. Again, if they are both in the same class, they are both rewarded, but this time with with a probability $\rho_{i,j}$, which takes into consideration the mutual similarity between S_i and S_j as inferred from $\{\lambda_i, \lambda_j\}$. Typically, the value $\rho_{i,j}$ can be computed using either a "fuzzy" relationship or a probabilistic one. In this paper,we shall assume that $\rho_{i,j}$ is probabilistically defined as :

$$\rho_{i,j} \leftarrow \lambda_i \cdot \lambda_j.$$

Analogously, if they are in different classes, they are both penalized with a probability $\rho_{i,j}$.

We now describe the actual transitions which are executed on selections being rewarded and penalized. Let us assume that the images in question are A_u and A_v. On being rewarded, since A_u and A_v are in the same class, say, α_j, both of them are moved toward the most internal state of that action, $\phi_{(j-1)N+1}$, one step at a time. See Figure IV(a). If A_u and A_v lie in different classes, say α_j and α_m respectively, (i.e. A_u is in state ξ_u, where $\xi_u \in \{\phi_{(j-1)N+1}, \ldots, \phi_{jN}\}$, and A_v is in state ξ_v, where $\xi_v \in \{\phi_{(m-1)N+1}, \ldots, \phi_{mN}\}$), they are moved away from $\phi_{(j-1)N+1}$ and $\phi_{(m-1)N+1}$ as follows:

a) If $\xi_u \neq \phi_{jN}$ and $\xi_v \neq \phi_{mN}$, then move A_u and A_v one state towards ϕ_{jN} and ϕ_{mN} respectively. See Figure IV (b).

b) If at least one of A_u or A_v is in the boundary state of *Minimum* Certainty, (i.e. either $\xi_u = \phi_{jN}$ or $\xi_v = \phi_{mN}$)[1], then move the object in the boundary state, say A_u, to ϕ_{mN}, the boundary state of α_m. In this case, since this will result in an excess of objects in α_m, one of the objects in α_m which is not accessed is moved to ϕ_{jN}, the boundary state of α_j. We choose to move the one closest to ξ_v. See Figure IV(c).

These migrations are exactly analogous to the ones used in the OMA. The primary difference is that these migrations are done for all pairs $<S_i, S_j>$ in S, and every pair $<S_i, S_j>$ is migrated with a probability $\rho_{i,j}$. Furthermore, all the migrations are executed only after the target image X is located. Thus, these pairs are not known as the query is being generated. Indeed, the set S is updated and stored as the system searches for the target image X, and all migrations are executed only *after* the latter has been identified.

[1]If both are in their respective boundary states of *Minimum* certainty, one of them is randomly chosen to be moved.

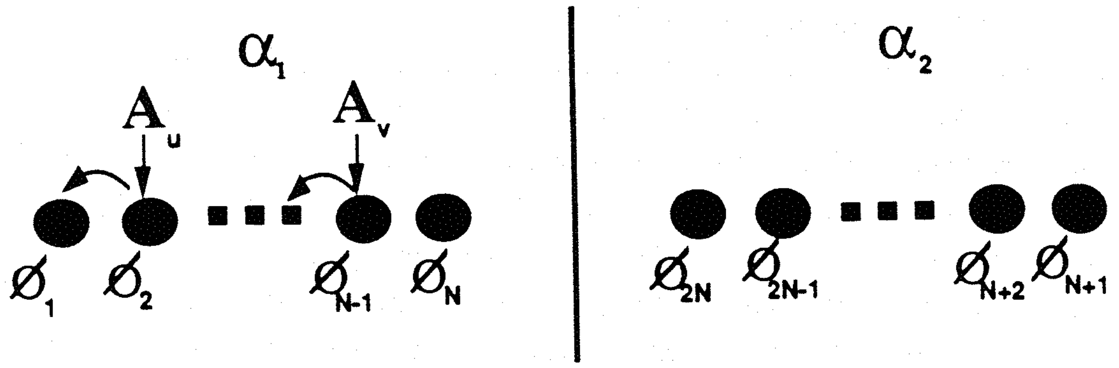


Figure IV (a) : Reward process for the 2N-state Image Database Learning Automaton.

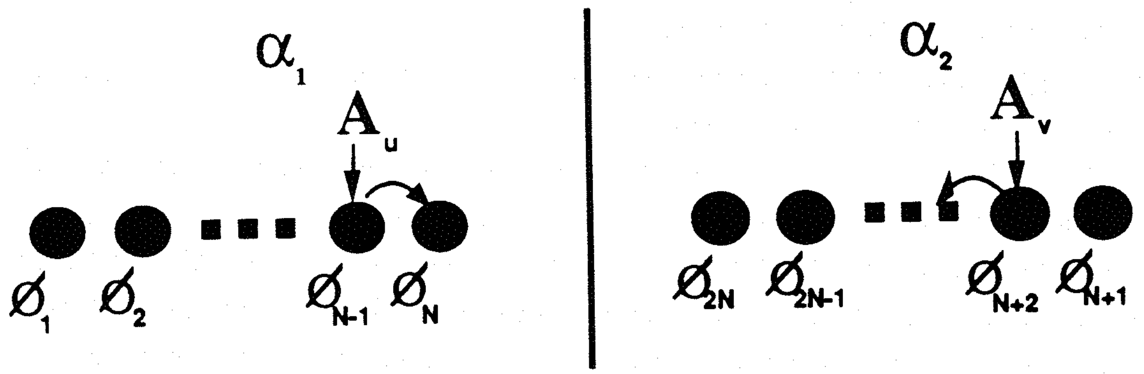


Figure IV (b) : Penalty process for the 2N-state Image Database Learning Automaton when neither of the images is in the boundary state.

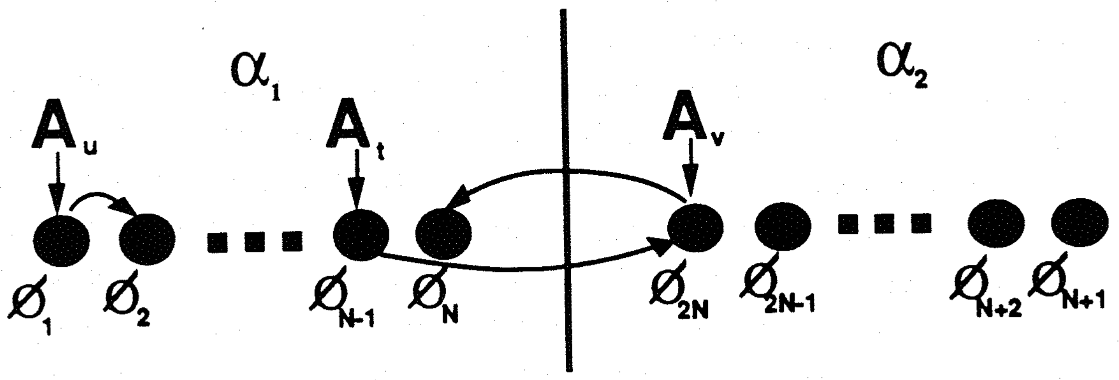


Figure IV (c) : Penalty process for the 2N-state Image Database Learning Automaton when one of the images (Av is in the boundary state).

Figure IV : The Transitions of the IDLA when the Images to be migrated are in α_1 and in α_2 Respectively.

4.1 System Design and Implementation Considerations

We shall now discuss objectives considered in designing the image retrieval system.

First of all, the system must correctly and efficiently implement the IDLA, which essentially implies that the Reward/Penalty Operations of the machine must be correctly executed. The system must be able to demonstrate the correctness by providing the user the partitioning of the objects as dictated by the Reward/Penalty Operations. This essentially involves intelligently providing the user the information about all the images in the system, their current classes, and their current states. In the design of the system we aimed at presenting these pieces of information in a straightforward way, with the capability of permitting the user to step through the scheme one query at a time and examine the dynamics of the machine. The user was also permitted the capability of executing several parallel experiments, and of having the individual experiments process queries indefinitely until the scheme converges.

In the prototype system which we implemented we assumed that the degree of confidence, λ_i, associated with every tagged image S_i was always unity.

We also spent a fair bit of effort on the system's user interface which presented the user with good graphical displays, pull-down menus, mouse capabilities, and easy file selection methods. The interface conformed to a well-acclaimed standard, and provided the ability to display colour images of various sizes. Thus, if the user is presented with a set of 16 images (and the resolution of the original images is 512 x 480), then each image is correspondingly scaled (or compressed) to be of resolution 128 x 120. The prototype system requires a machine with a supporting VGA card with at least 16 colours and a 320 x 200 resolution, which is the lowest accepted standard for graphics cards. The various options that are available at the top level menu are given in Figure V, and the screen layout for the manual operation mode in Figure VI.

With regard to implementation, the system was written so as to permit a "Simulation" mode, in which a Simulation module interacted with the IDLA module. In this mode the images and their similarities to the other presented images were randomly selected based on an underlying grouping which was *unknown* to the IDLA. The system and the user interface were implemented using the Microsoft Windows programming library, and thus, instead of spending a lot of time writing low-level graphics routines and higher-level menu routines, these were inherited and utilized to render the system more user-friendly. Windows currently supports a variety of high and low-resolution graphics cards, and so our prototype system can also be demonstrated on a large variety of systems with a potential future cross-platform capability.

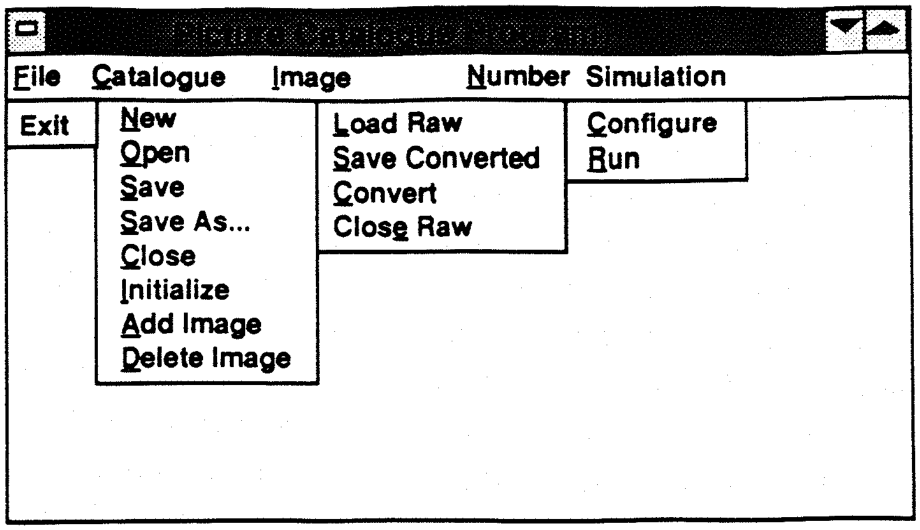
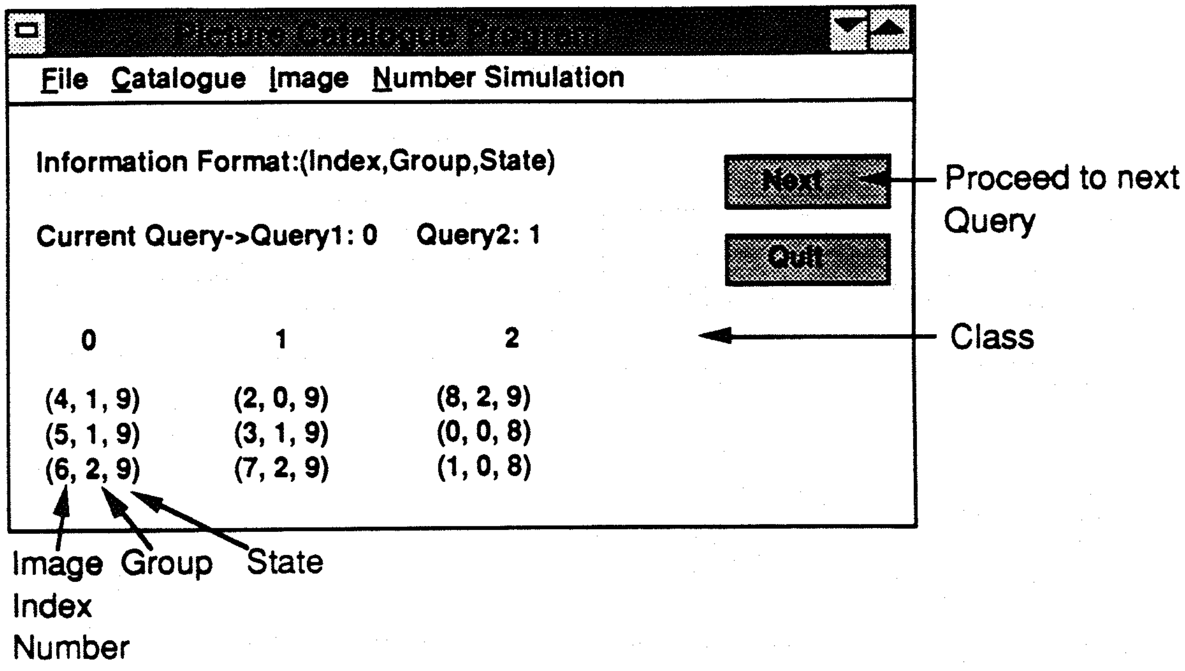


Figure V : The Various Menu Options Available on the Prototype System


Picture Catalogue Program
File Catalogue Image Number Simulation
Next
Quit
Proceed to next Query
Class
Image Index Number
Group
State

```
Information Format:(Index,Group,State)

Current Query->Query1: 0    Query2: 1


   0           1           2

(4, 1, 9)   (2, 0, 9)   (8, 2, 9)
(5, 1, 9)   (3, 1, 9)   (0, 0, 8)
(6, 2, 9)   (7, 2, 9)   (1, 0, 8)
```

Figure VI : The Screen Layout for the Manual Operation Mode of the Prototype.

Figure VII shows the various software modules involved. The "Menu Input" and "Display Output" modules are collections of user I/O routines that include Microsoft Windows functions from the Software Development Toolkit. The "Menu Input" module also handles switching between the various kinds of simulations that the system supports. At the heart of the system is the module which handles the Reward/Penalty Functions of the IDLA. It manipulates the automaton data structure and processes the queries made by either the "Simulation Module" or the "Image Catalogue" module. These modules provide three methods of testing the system, two in the former and one in the latter. In the former, the user is permitted to view the state of the automaton after processing each query or after the machine has converged (which occurs when all the images are in their states of *Maximum* Certainty). The "Image Catalogue" module also uses the automaton module to process the queries received from the "Menu Input" module. Finally, the "Image Catalogue" module interacts with the Catalogue Files for the purpose of re-configuration, and thus whenever the IDLA demands a change in the partitioning, the new partitioning information is passed to the "View Catalogue" module. This module uses the Image Files on disk to obtain the bit maps of the images and displays them by invoking the "Display Output" module.

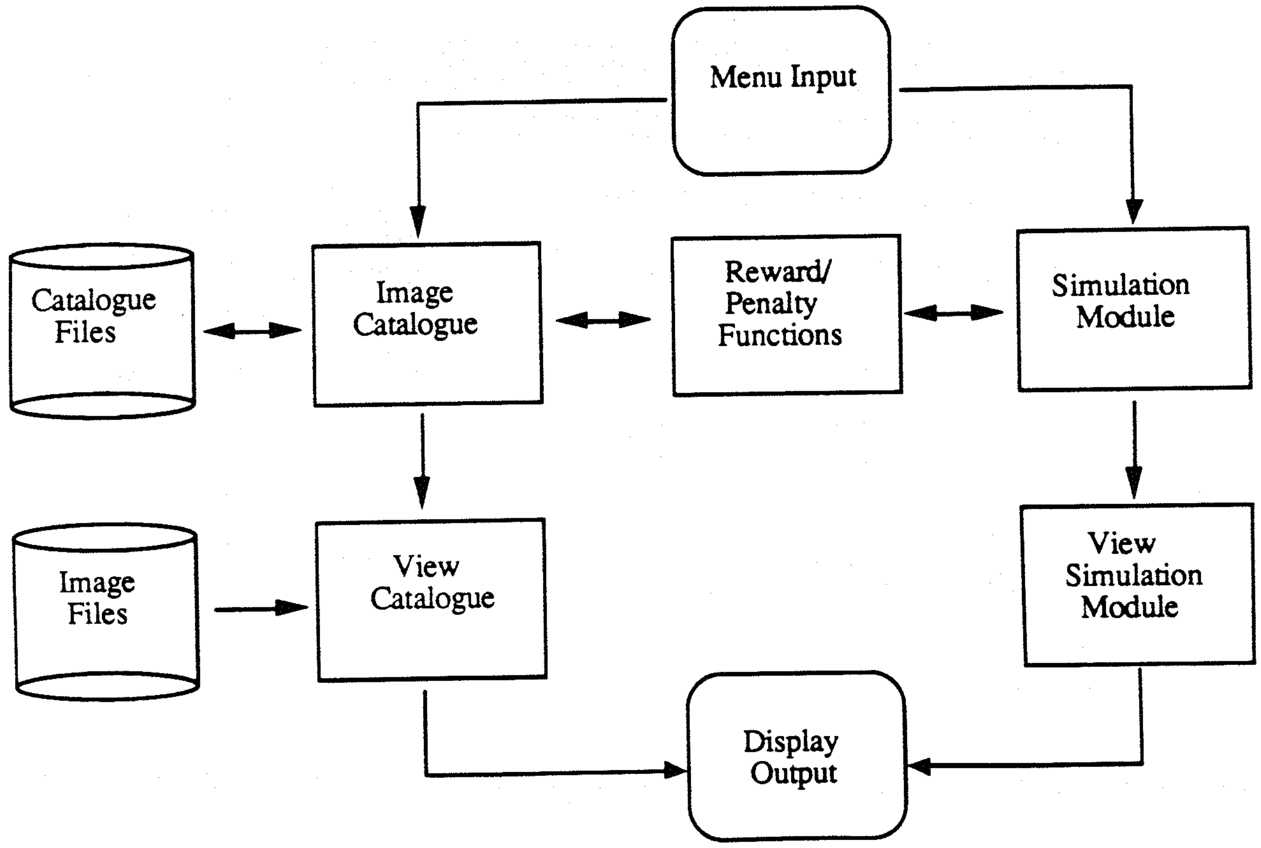


Figure VII : The Modules of the Image Retrieval System

The prototype system is both accurate and fast. The image database can currently contain up to 256 pictures which are to be partitioned into 16 sub-databases containing images of cars, boats, trucks, airplanes, hunting scenes etc. Typically, if the images are randomly assigned into the sub-databases, the system converges to the optimal configuration in a few dozen queries -- it rarely needs more than a hundred queries. It is difficult to objectively test the efficiency of the system because the input queries, the tagged images and the degrees of confidence are subjective. We are still in the process of investigating how rigid benchmark tests can be made on the system. We have some new ideas which we believe will help us objectively demonstrate the effectiveness of the system, but unfortunately these are yet to be rigorously verified.

The Pseudo-Code for the system is given in the Appendix.

V. CONCLUSIONS AND OPEN PROBLEMS

In this paper we have considered the scenario in which a user wants to browse through a database of images to retrieve a particular target image which he is interested in. This is called the Image Examination and Retrieval Problem (IERP). Rather than specifying the image textually, we permit the user to access his target image using his *subjective* discrimination of how the latter resembles the other images that are presented to him by the system. The query is processed by repeatedly presenting various representative images to the user who responds by tagging images which are "similar" to his target image. By extracting the information content in these responses, the IERP is not merely viewed as a problem involving recognition or classification. Instead it is viewed as one that falls in the domain of partitioning the *set of images* in terms of their "visual" resemblances. We have shown how queries can be used to not merely locate images, but, in fact, to group all similar images together so that subsequent searches will be enhanced. The intelligent partitioning of the image database is done adaptively on the basis of the statistical properties of the user's query patterns. This is achieved using learning automata and does not involve the evaluation of statistics.

We are currently investigating techniques by which we can objectively test the efficiency of our scheme. The question of partitioning the images into non-equally sized sub-databases remains open.

ACKNOWLEDGEMENT

The authors are extremely grateful to Mr. Ted de St. Croix of the School of Computer Science at Carleton for his assistance in preparing the manuscript.

REFERENCES

[1] Boursier, P., "Image Databases : A Status Report", *Proc. of the 1985 IEEE Workshop on Computer Architecture*, Miami Beach, Florida, Nov. 1985, pp. 355-358.

[2] Chang, S. K. and Lin, B.S., "GRAIN - A Pictorial Database Interface" in *IEEE 1980 Workshop on Picture Data Description and Management*, (1980), pp. 83 - 88.

[3] Dubes, R. and Jain, A. K., "Validity Studies in Clustering Methodologies", *Pattern Recognition*, (1976), Vol. 8 pp. 235-254.

[4] Lowe, D. G., *Perceptual Organization and Visual Recognition*, Kluwar Academic, Boston (1985).

[5] Lien, Y. E. and Utter, D. F., Jr., "Design of an Image Database", *Proc. of the Workshop on Picture Data Description and Management*, (Apr. 1977), pp. 131-136.

[6] McKeown, D. J. Jr. and Reddy, D.R., "A Hierarchical Symbol Representation for an Image Database", *Proc. of the Workshop on Picture Data Description and Management*, (Apr. 1977), pp. 40-44.

[7] Nagata, M. and Oonishi, Y., "Video Image Manipulation in Multi-media Pictorial Database Management", *Proc. of the 1985 IEEE Workshop on Computer Architecture*, Miami Beach, Florida, Nov. 1985, pp. 340-347.

[8] Nilsson, N. J., *Principles of Artificial Intelligence*, Tioga Publishing, Palo Alto, California, (1980).

[9] Oommen, B. J. and Ma, D. C. Y., "Deterministic Learning Automata Solutions to the Equipartitioning Problem" in *IEEE Transactions on Computers*, Vol. 37, No. 1. (Jan. 1988), pp. 2-13.

[10] Rorvig, M. E., "The Two Domain Theory of Image Collection Searching". To Appear in *Information Processing and Management*. Also available from Graduate School of Library and Information Science, The University of Texas at Austin, (1986).

[11] Sato, H., Okazaki, T., Yamamoto, H., and Tamura, H., "The VIEW-Station Environment : Tools and Architecture for a Platform Independent Image Processing Workstation", in the *Proc. of the IEEE Tenth International Conference on Pattern Recognition*, Atlantic City, June 1992, pp.576-583.

[12] Tamura, H. and Naokazu, Y., "Image Database Systems: A Survey" in *International Journal of Pattern Recognition*, Vol. 17, No. 1 (1984), pp. 29-43.

[13] Tanimoto, S. L., "Hierarchical Picture Indexing and Description", in *IEEE 1980 Workshop on Picture Data Description and Management*, (1980), pp. 103-105.

[14] Tojo, A. and Sato, T., "Interoperable database System : A New R & D Project and its impact on Multi-media Information Processing Technology", *Proc. of the 1985 IEEE Workshop on Computer Architecture*, Miami Beach, Florida, Nov. 1985, pp. 336-339.

[15] Tsetlin, M. L., *Automaton Theory and the Modelling of Biological Systems*, New York and London, Academic, (1973).

[16] Wakimoto, K., Shima, M., Tanaka, S., and Maeda, A., "An Intelligent User-Interface to an Image Database Using a Figure Interpretation Method", in the *Proc. of the IEEE Tenth International Conference on Pattern Recognition*, Atlantic City, June 1992, pp. 516-520.

[17] Weigand, H. "An Object-Oriented Approach in a Multimedia Database Project". in *Object-Oriented Databases : Analysis, Design and Construction (DS-4)*, (Ed. Meersman, R.A., Kent, W., and Khosla, S), Elsevier Publishers, 1991.

[18] Yamaguchi, K., Ohbo, N., Kunii, T. L., Kitagawa, H. and Harada, M., "ELF : Extended Relational Model for Large, Flexible Picture Databases", in *IEEE 1980 Workshop on Picture Data Description and Management*, (1980),pp. 95-102.

[19] Yu, C. T., Siu, M. D., Lam, D. and Tai, F., "Adaptive Clustering Schemes: General Framework" in *Proc. of the IEEE COMPSAC Conference*, (1981), pp. 81-89.

[20] Narendra, K. S. and Thathachar, M. A. L., *Learning Automata : An Introduction*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1989.

APPENDIX
THE IMAGE RETRIEVAL SYSTEM

Assumption : The system has W images, $\{A_1,...,A_W\}$, to be partitioned into K classes. Most of the procedures are self-explanatory. The central procedure, Process_IDLA, follows the Algorithm IDLA_System. With regard to notation, ξ_i is the state of the image S_i. It is an integer in [1..KN], where, if $(j-1)N+1 \le \xi_i \le jN$, then object S_i is assigned to α_j.

```
PROCEDURE  IDLA_System
    $S \leftarrow \{\ \}$                                        (* Initialize set of similar images S *)
    Initialize randomly $\xi_i$ for $1 \le i \le W$, to the boundary states of the classes,
        each class having W/K images
    Repeat
        Get ( User_Operation )
        Case ( User_Operation ) Of
            View_Top_Tevel :                                   (* Overview of all classes *)
                For (each of k image classes ) Do
                    Show image closest to the most internal class state
                EndFor
            End View_Top_Tevel

            View_Class
                Get_Class_No (k)
                For ( each image in class k ) Do
                    Show image
                EndFor
            End View_Class

            Select_Image_As_Similar :
                Get_Image_Index_and_Simiarity $(Y,\lambda)$
                $S \leftarrow S \cup Y$                              (*Add Y to similarity set S *)
            End Select_Image_As_Similar

            Select_Image_As_Found:
                Get_Image_Index (Y)
                Display_Image (Y)
                $S \leftarrow S \cup Y$
                For $i \leftarrow 1$ to Size (S) - 1 by 1 Do          (Migrate $A_i$ to $A_{|S|}$, with prob. $\lambda_i$)
                    Process_IDLA ( $A_i$, $A_{|S|}$, $\lambda_i$)
                EndFor
                For $i \leftarrow 1$ to Size (S) - 2 by 1 Do
                    For $j \leftarrow i + 1$ to Size (S) -1 by 1 Do
                        Process_IDLA ($A_i$, $A_j$, $\rho_{i,j}$)     (Migrate $A_i$&$A_j$, with prob. $\rho_{i,j} \leftarrow \lambda_i\lambda_j$)
                    EndFor
                EndFor

        EndCase
        $S \leftarrow \{\ \}$                                    (* Re-initialize S *)
        View_Top_Level
    ForEver                                                    (* Ends Repeat -- Wait next query *)
END PROCEDURE  IDLA_System
```

```
PROCEDURE Process_IDLA ($A_i$, $A_j$, $\pi$)
    $p \leftarrow$ Random(0,1)
    If ($p < \pi$) Then
        If (($\xi_i$ div N) $\leftarrow$ ($\xi_j$ div N)) Then          (*The partitioning is rewarded*)
            If (($\xi_i$ mod N) $\neq 1$) Then                        (*Move $S_i$ towards the internal
                $\xi_i \leftarrow \xi_i - 1$                                                   state*)
            EndIf
            If (($\xi_j$ mod N) $\neq 1$) Then                        (*Move $S_j$ towards the internal
                $\xi_j \leftarrow \xi_j - 1$                                    state*)
            EndIf
        Else                                                          (* The partitioning is penalized *)
            If ((($\xi_i$ mod N) $\neq 0$) and (($\xi_j$ mod N) $\neq 0$)) Then (* Both are in internal states *)
                $\xi_i \leftarrow \xi_i + 1$
                $\xi_j \leftarrow \xi_j + 1$
            Else  If ($\xi_i$ mod N $\neq 0$) Then                    (* $S_i$ is in an internal state *)
                    $\xi_i \leftarrow \xi_i + 1$
                    temp $\leftarrow \xi_j$                           (* Store the state of $S_j$ *)
                    $\xi_j \leftarrow \xi_i$                          (* Move $S_j$ to same group as $S_i$ *)
                    t $\leftarrow$ index of an unaccessed object in group of $S_i$
                        where $S_t$ is closest to $\xi_i$
                    $\xi_t \leftarrow$ temp                           (* Move $S_t$ to the old state of $S_j$ *)
                  Else                                                (* $S_i$ has to be moved *)
                    If ($\xi_j$ mod N) $\neq 0$) Then                 (* Move $S_j$ if necessary *)
                        $\xi_j \leftarrow \xi_j + 1$
                    EndIf
                    temp $\leftarrow \xi_i$                           (* Store the state of $S_i$ *)
                    $\xi_i \leftarrow \xi_j$                          (* Move $S_i$ to same group as $S_j$ *)
                    t $\leftarrow$ index of an unaccessed object in group of $S_j$
                        where $S_t$ is closest to $\xi_j$
                    $\xi_t \leftarrow$ temp
                  EndIf
            EndIf
        EndIf
    EndIf
END PROCEDURE Process_IDLA
```

School of Computer Science, Carleton University
Recent Technical Reports

TR-168 **Adaptive List Organizing for Non-stationary Query Distributions. Part I: The Move-to-Front Rule**
R.S. Valiveti and B.J. Oommen, January 1990

TR-169 **Trade-Offs in Non-Reversing Diameter**
Hans L. Bodlaender, Gerard Tel and Nicola Santoro, February 1990

TR-170 **A Massively Parallel Knowledge-Base Server using a Hypercube Multiprocessor**
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990

TR-171 **Parallel Processing of Quad Trees on the Hypercube (and PRAM)**
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990

TR-172 **A Note on the Load Balancing Problem for Coarse Grained Hypercube Dictionary Machines**
Frank Dehne and Michel Gastaldo, May 1990

TR-173 **Self-Organizing Doubly-Linked Lists**
R.S. Valiveti and B.J. Oommen, May 1990

TR-174 **A Presortedness Metric for Ensembles of Data Sequences**
R.S. Valiveti and B.J. Oommen, May 1990

TR-175 **Separation of Graphs of Bounded Genus**
Ljudmil G. Aleksandrov and Hristo N. Djidjev, May 1990

TR-176 **Edge Separators of Planar and Outerplanar Graphs with Applications**
Krzystof Diks, Hristo N. Djidjev, Ondrej Sykora and Imrich Vrto, May 1990

TR-177 **Representing Partial Orders by Polygons and Circles in the Plane**
Jeffrey B. Sidney and Stuart J. Sidney, July 1990

TR-178 **Determining Stochastic Dependence for Normally Distributed Vectors Using the Chi-squared Metric**
R.S. Valiveti and B.J. Oommen, July 1990

TR-179 **Parallel Algorithms for Determining K-width-Connectivity in Binary Images**
Frank Dehne and Susanne E. Hambrusch, September 1990

TR-180 **A Workbench for Computational Geometry (WOCG)**
P. Epstein, A. Knight, J. May, T. Nguyen, and J.-R. Sack, September 1990

TR-181 **Adaptive Linear List Reorganization under a Generalized Query System**
R.S. Valiveti, B.J. Oommen and J.R. Zgierski, October 1990

TR-182 **Breaking Substitution Cyphers using Stochastic Automata**
B.J. Oommen and J.R. Zgierski, October 1990

TR-183 **A New Algorithm for Testing the Regularity of a Permutation Group**
V. Acciaro and M.D. Atkinson, November 1990

TR-184 **Generating Binary Trees at Random**
M.D. Atkinson and J.-R. Sack, December 1990

TR-185 **Uniform Generation of Combinatorial Objects in Parallel**
M.D. Atkinson and J.-R. Sack, January 1991

TR-186 **Reduced Constants for Simple Cycle Graph Separation**
Hristo N. Djidjev and Shankar M. Venkatesan, February 1991

TR-187 **Multisearch Techniques for Implementing Data Structures on a Mesh-Connected Computer**
Mikhail J. Atallah, Frank Dehne, Russ Miller, Andrew Rau-Chaplin, and Jyh-Jong Tsay, February 1991

TR-188	**Generating and Sorting Jordan Sequences** Alan Knight and Jörg-Rüdiger Sack, March 1991
TR-189	**Probabilistic Estimation of Damage from Fire Spread** Charles C. Colbourn, Louis D. Nel, T.B. Boffey and D.F. Yates, April 1991
TR-190	**Coordinators: A Mechanism for Monitoring and Controlling Interactions Between Groups of Objects** Wilf R. LaLonde, Paul White, and Kevin McGuire, April 1991
TR-191	**Towards Decomposable, Reusable Smalltalk Windows** Kevin McGuire, Paul White, and Wilf R. LaLonde, April 1991
TR-192	**PARASOL: A Simulator for Distributed and/or Parallel Systems** John E. Neilson, May 1991
TR-193	**Realizing a Spatial Topological Data Model in a Relational Database Management System** Ekow J. Otoo and M.M. Allam, August 1991
TR-194	**String Editing with Substitution, Insertion, Deletion, Squashing and Expansion Operations** B John Oommen, September 1991
TR-195	**The Expressiveness of Silence: Optimal Algorithms for Synchronous Communication of Information** Una-May O'Reilly and Nicola Santoro, October 1991
TR-196	**Lights, Walls and Bricks** J. Czyzowicz, E. Rivera-Campo, N. Santoro, J. Urrutia and J. Zaks, October 1991
TR-197	**A Brief Survey of Art Gallery Problems in Integer Lattice Systems** Evangelos Kranakis and Michel Pocchiola, November 1991
TR-198	**On Reconfigurability of Systolic Arrays** Amiya Nayak, Nicola Santoro, and Richard Tan, November 1991
TR-199	**Constrained Tree Editing** B. John Oommen and William Lee, December 1991
TR-200	**Industry and Academic Links in Local Economic Development: A Tale of Two Cities** Helen Lawton Smith and Michael Atkinson, January 1992
TR-201	**Computational Geometry on Analog Neural Circuits** Frank Dehne, Boris Flach, Jörg-Rüdiger Sack, Natana Valiveti, January 1992
TR-202	**Efficient Construction of Catastrophic Patterns for VLSI Reconfigurable Arrays** Amiya Nayak, Linda Pagli, Nicola Santoro, February 1992
TR-203	**Numeric Similarity and Dissimilarity Measures Between Two Trees** B. John Oommen and William Lee, February 1992
TR-204	**Recognition of Catastrophic Faults in Reconfigurable Arrays with Arbitrary Link Redundancy** Amiya Nayak, Linda Pagli, Nicola Santoro, March 1992
TR-205	**The Permutational Power of a Priority Queue** M.D. Atkinson and Murali Thiyagarajah, April 1992
TR-206	**Enumeration Problems Relating to Dirichlet's Theorem** Evangelos Kranakis and Michel Pocchiola, April 1992
TR-207	**Distributed Computing on Anonymous Hypercubes with Faulty Components** Evangelos Kranakis and Nicola Santoro, April 1992
TR-208	**Fast Learning Automaton-Based Image Examination and Retrieval** B. John Oommen and Chris Fothergill, June 1992