Metadatabase and Search Agent for Multimedia Database Access over Internet

Wendy Chang[†], Deepak Murthy, Yousong Mei[†], and Aidong Zhang
Department of Computer Science
State University of New York at Buffalo
Buffalo, NY 14260

[†]Department of Electrical and Computer Engineering
State University of New York at Buffalo
Buffalo, NY 14260

Abstract

Various multimedia repositories are now distributed throughout the Internet and can be accessed via the World Wide Web tools. In such an environment, a challenging problem is to find the multimedia databases at distributed locations that are the most relevant to the user query. In this paper, we investigate approaches to the creation of a metadatabase and design of a search agent in a metaserver which supports the integrated access to various multimedia databases. The creation of the metadatabase formulates the metadata on the types of media data each multimedia database at a remote site houses and its query capabilities. The design of the search agent at metaserver accesses the metadatabase and control the distribution of user queries to relevant database sites through the site server. The search agent also directs interactive dialogue between the client and multimedia databases. The proposed metadatabase and search agent is designed and implemented in a web-based multimedia information retrieval environment.

1 Introduction

With the technology of the Internet, the availability of digital media, and the advent of the World Wide Web (WWW), it is now possible to access large multimedia repositories distributed throughout the world. Such access is becoming increasingly important in a number of applications, including medical diagnostics, manufacturing, education, surveillance, and distributed publishing. In such applications, information from multimedia databases distributed at remote locations must be made available at other locations for purposes ranging from diagnostics to distance learning. Accessing these repositories in a distributed setting over the public Internet poses a number of challenges. In a typical scenario, access to a multimedia database at a remote database site may be initiated by a client machine running a browser such as Netscape or Microsoft's Internet Explorer. The query is processed by the browser and sent to a metaserver. This metaserver selects target multimedia database site(s) and poses the query to these databases in an acceptable form. The search mechanism of each database then searches its repository for possible answers to the posed query. Answers are fed back to the metaserver for eventual transmission to the client.

Some previous work has addressed resource discovery systems in the context of text databases. In such work, techniques from information retrieval [ODL93, SEKN93] have been applied to intelligent resource site selection. Example of such systems include GLOSS (from Stanford) [GGMT94, GGM95], WHOIS++ (from Bunyip Corporation), and HARVEST (from University of Colorado) [BDH+94]. These systems employ statistical approaches to record the frequency of occurrence of text keywords from known sites; this information is used to construct an index of relevant sites for query direction. In particular, the GLOSS (Generalized Glossary of Servers) server [GGM95, GGMT94] keeps statistics on the available databases and estimates their potential applicability to a given query using Boolean and vector-space retrieval models of document retrieval. Another approach uses inference networks to assist with text database discovery [CLC95]. This summarizes databases using document frequency information for each term together with the inverse collection frequency of different terms. An inference network then uses this information to rank the databases for a given query [CLC95]. Finally, the HARVEST [BDH+94] system provides a flexible architecture for accessing information over the Internet. "Gatherer" modules collect information about data resources, and this is supplied to "broker" modules. A structured representation of these broker modules is kept in the harvest server registry which, in a sense, becomes a metadatabase of information about the individual database sites.

A number of new technical issues arises as a consequence of the nature of multimedia data. Traditional database access methods, in which text annotations are assigned to image and video data, are both insufficient to unambiguously describe the media content and involve the time-consuming task of manually tagging the multimedia data. Many techniques have been proposed to handle content-based multimedia data retrieval [ACF+93, CYDA88, TPF+91, HLHC92, BPJ93, WN94, SC94, JM95, SNA+95, SZ94, COM95]. Given individual multimedia databases located in various places over the internet, the issues of global access to these databases have to be taken into consideration. In particular, the process of selecting the relevant databases for a given query remains a challenging problem. As multimedia information grows rapidly over Internet or WWW, a careful selection of databases becomes necessary. In addition, content-based querying of each image database is inherently complex and computationally intensive. Furthermore, queries posed to several databases may yield answers which must be consolidated and summarized before they can be presented to a user. Finally, since users may not be aware of the specific capabilities of each remote database, queries may need to be transformed into a more suitable format, a process which may require additional user input.

This paper will focus on the integration of multimedia databases for use in a distributed web-based environment. The architecture of the system is shown in Figure 1. As can be seen from this figure, the three main components are multimedia database systems at web sites, a metaserver consisting of a search agent and a metadatabase at the web server, and a set of web applications at web clients. Each multimedia database system manages the insertion, deletion, and update of the media data stored in the local database.

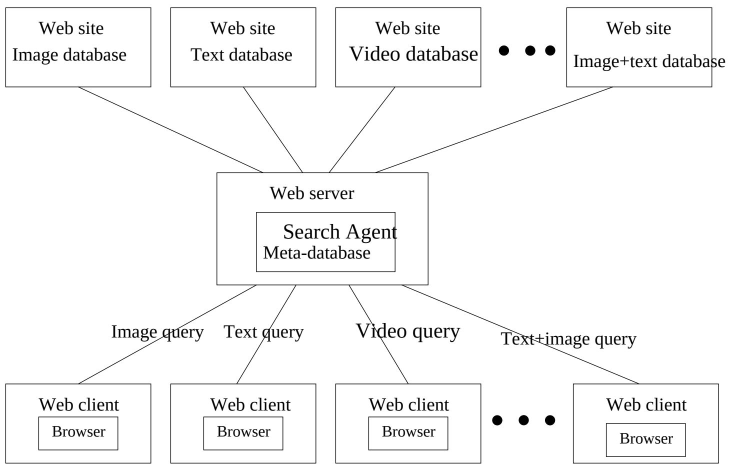


Figure 1: The web-based multimedia data management and retrieval system architecture.

We will particularly investigate approaches to the creation of a metadatabase and the design of a metaserver search agent which supports integrated access to various multimedia databases. The creation of the metadatabase will involve the formulation of the metadata needed for database site selection. Such metadata are based on the types of media data housed at each multimedia database site and on their query capabilities. The metadata is generated by a refining module which observes the pattern of responses of various databases to user queries. The search agent located at the metaserver will access this metadatabase and control the distribution of user queries to relevant database sites. Thus, a novel indexing mechanism must be supported. The search agent also directs interactive dialogue between the client and multimedia databases. The proposed metadatabase and search agent will be designed and implemented in a Web-based multimedia information retrieval environment. The intent of this system is to allow users to access information from multiple multimedia databases using the richer information presentation capabilities of Web browsers. The effectiveness of the proposed approaches will be demonstrated using a prototypical system that consolidates information from relevant multimedia database sites in response to user queries.

The rest of the paper is organized as follows. In Section 2, a metadatabase design framework is presented that makes use of the analogy between image selection within a database and the selection of databases themselves. Section 3 presents the design of the search agent, including similarity measurement, database ranking algorithm, and a method for user query redirection. In Section 4, we present the design of the prototype system which implements the proposed approaches in a Web-based environment. Experimental results are also reported. In Section 5, we discuss the resource discovery procedure. Concluding remarks are offered in Section 6.

2 Metadatabase Design

In this section, we will formulate the metadata necessary for the metaserver to select appropriate component multimedia databases to perform user queries. We will then propose an approach to the organization of the metadata. The metadatabase will be designed to maintain its *extensibility*; that is, it can be extended to include new metadata without changing the existing metadata structure. Extensibility is considered to be essential to accommodate the rapid growth of data resources. In addition, performance and storage requirements are also considered in the design of the metadatabase.

2.1 Database Grouping

The metadatabase is organized on two different levels. At the first level, database sites are grouped according to templates (or icons), including keywords, texture pattern, color pattern, and shape pattern. The information needed for group assignment is supplied at the time the database registers with the metaserver. A database site can belong to one or more template group. Probability data for each database site with respect to templates is also supplied at the time of database registration. As all groupings are based on degree of similarity, we must begin with a definition of the term "similarity." We must introduce a parameter called the "similarity threshold".

We consider image A to be "similar" to image B if the similarity rate of A to B is greater than a "similarity threshold". In our previous work, we were able to compare various images to a base image and assign a similarity rate, ranging from 0.0 to 1.0, to each image [ZCA95, ZCA96, RSZ96, RSZSM96], with 0.0 representing minimum and 1.0 representing maximum similarity. Related work from other researchers in the community has also appeared [SNA+95, SC94, JM95]. This work assumes the availability of similarity rates among images.

We will use the following notation in the ensuing discussions:

- **SimRate(i,t):** the similarity rate between image i and template t, where $0.0 \leq SimRate(i,t) \leq 1.0$.
- **th:** the similarity threshold, where $0.0 \leq th \leq 1.0$.
- **NuImg(db,t,th):** number of similar images of a database db, with respect to template t and similarity threshold th.
- **Mean(db,t,th):** the mean similarity rate of similar images within database db, with respect to template t and similarity threshold th. This value is calculated using the equation

$$Mean(db,t,th) = \frac{\Sigma SimRate(i,t)}{NuImg(db,t,th)}, \tag{1}$$

where $i \in db$ and $SimRate(i,t) > th$.

- **Var(db,t,th):** the standard deviation of similarity rate of similar images within database db, with respect to template t and similarity threshold th. This value is calculated using the equation

$$Var(db,t,th) = \sqrt{\frac{\Sigma(SimRate(i,t) - Mean(db,t,th))^2}{NuImg(db,t,th) - 1}}, \quad (2)$$

where $i \in db$ and $SimRate(i,t) > th$.

Under each template t, only those databases satisfying the following condition will be included:

$$NuImg(db,t,th) > 0. \quad (3)$$

For a database db which is included in template t, metadata are recorded with which the search agent to determine the probability of retrieving information for any user query in db. These metadata include $NuImg(db,t,th)$, $Mean(db,t,th)$, $Var(db,t,th)$, and time, as well as the cost involved in performing a query. Note that a database site may associate with one or more templates. Each template is associated with a utilization, and its value is defined as number of times a query is executed. A refining module is applied periodically to delete templates with a low utilization value. Such operations should be performed off-line, during low-use periods, rather than during user queries. New templates can be added by the metaserver.

2.2 Template Organization

At the second level, images templates are organized into a hierarchy, with the root representing a dummy node and the first-level nodes representing the most general cluster classification, including various categories of color, texture, shape, text, spatial constraint, and motion. Each category may then contain a sub-tree of specific image icons. The hierarchical organization of the templates is similar to the image clustering organization presented within an individual multimedia database [ZCA96]. Figure 2 shows a sub-tree of geographical environment image icons.

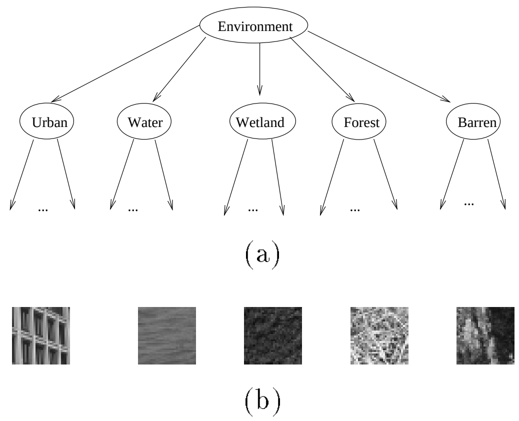


Figure 2: (a) A hierarchical clustering of an environment image setting. (b) Clustering icons.

Combining both levels, Figure 3 shows a conceptual view of the metadatabase. Using this approach, the metaserver will assemble the metadatabase as follows. Given a set of databases at various sites, an initial metadatabase is constructed from pre-defined templates returned by the individual databases. Such a template contains metadata about the database, including the type of media data housed, expected query form, specialized algorithms supported, and statistical data associated with each multimedia template. Data such as monetary cost and latency of database sites are also stored to enable early pruning of costly sites. These templates can be periodically updated by the metaserver and relayed to component databases. The initial categorization of databases in the metadatabase is used to direct queries to relevant sites. A record of recently returned responses and the associated queries is maintained at the server to avoid redundant searches.

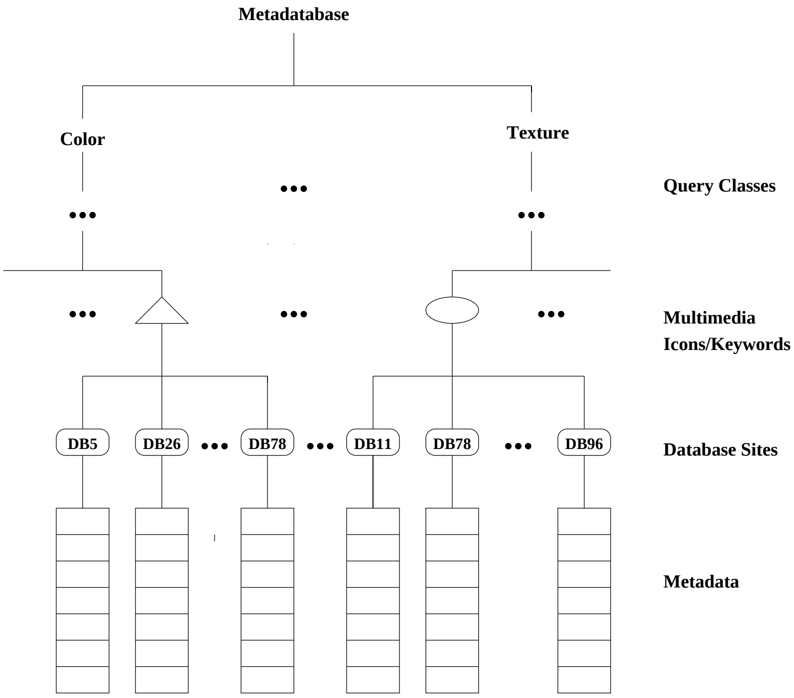


Figure 3: A conceptual view of the metadatabase.

In this approach, while the data in the metadatabase is simple (namely, the relevance probability of templates), the mechanisms for searching the metadatabase can be complex. A query-initiated search will involve the selection of relevant templates, which are then be used to index the appropriate database sites. This strategy could also be used to select text-only database sites, as the task of selecting relevant database sites is based directly on keywords. For non-text data, the search will be performed on the feature-abstracted representation of media data rather than directly on the media data itself. Approaches to supporting the search agent will be given in the next section.

3 Search Agent

In this section, we will introduce approaches to conducting a user query search. We will evaluate the characteristics of similarity and define methods of mean measurement, a ranking algorithm, and a query search procedure.

3.1 Similarity Comparisons

The two level indexing approach used in the metadatabase design results in the comparisons of two similarity rates during the search procedure of database resources. One is the similarity rate between query pattern and the template, and the other is the similarity rate between the template and database images. Through these two steps of similarity comparisons, we determine the degree of similarity between the query pattern and database images. To better understand the relationships between these two similarity rates, we have conducted experiments on various types of images. Using the Brodatz Collection [Bro66], we have demonstrated in [ZCA96] that images can be classified into different degree of clusters based on their similarity to a given image icon. We observed that if two highly relevant images are compared to the same template, it is highly possible that they will be given the similar similarity rates.

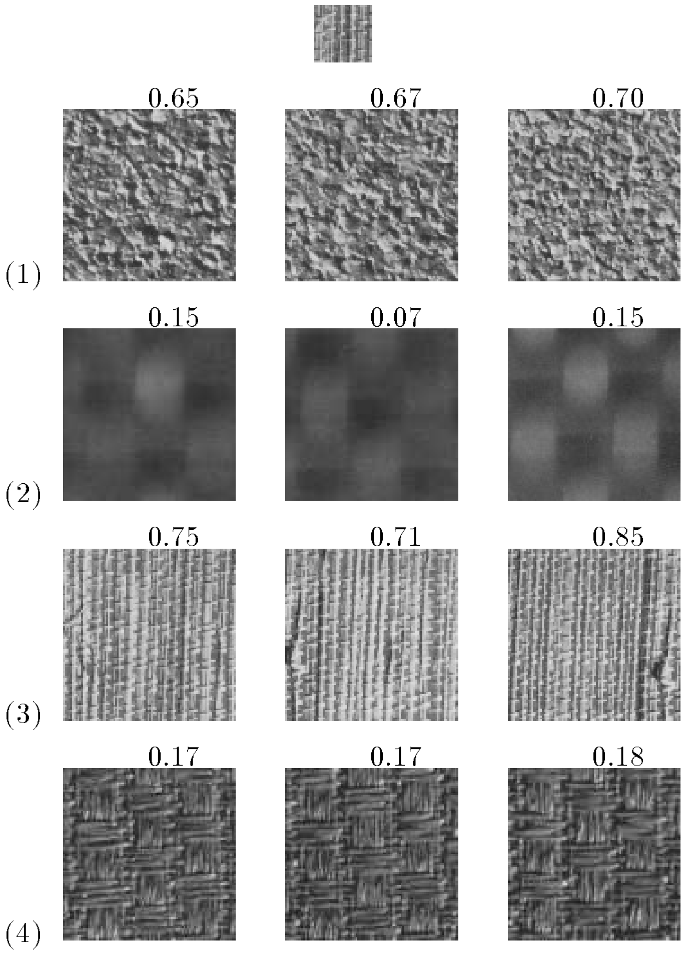


Figure 4: Iconic and database images from Brodatz Album.

Figure 4 given in [ZCA96] illustrates the above situation. In this figure, there are four different clusters of images. The similarity rate between the image icon and the image is listed on top of the image. We can see that a different cluster has a different range of similarity rates. For images in the same cluster, their similarity rates to the image icon is bound within a particular range.

For example, cluster (2) is in the range of $[0.07, 0.15]$ and cluster (3) is in the range of $[0.71, 0.85]$. Images belonging to cluster (3) are considered to have similar features to that of the image icon.

Thus, we assume that if two images are similar to a template, the similarity rates of these two images to the template will be closely related. Although we cannot guarantee that, for any two images A and B which have closely related similarity rates, A and B will be similar. The experiments demonstrate that there is a high probability such that A and B are similar.

The above observation a key point for the development of the database selection algorithm in the search agent. We will now be able to say that, for a given query q, template t, and a $SimRate(q,t)$, there exists an image i with the $SimRate(i,t) \approx SimRate(q,t)$, then q is similar to i. We further expanding this observation by introducing an offset value of similarity rate, together with $SimRate(q,t)$, it forms a "relevant range" so that the similarity rates of majority exact matching or highly relevant images will fall within this range. Later, we will examine this property more and determine the offset value experimentally.

3.2 Mean Measurement

We will now propose using the randomly generated mean similarity rate to give us an estimate of the mean similarity rate of each database site. Several factors must be considered when using the mean estimation method:

- *Sample selection*: As we are looking at possible thousands of images at each database site, it is impractical to observe all its values entirely. It is also unnecessary, as we are only interested in the images that are relevant to the query data pattern, possibly having the similar $SimRate$ value. To narrow the scope of calculations, we decided to select our samples based on a similarity threshold. We will only consider the images that have similarity rates greater than or equal to the threshold, and calculate the mean similarity rate value of these samples. We feel that the images obtained using this measurement are most relevant to our objective. Note that as the contents of each database site are constantly changing, the mean of similarity rates of the samples obtained only represents a random instance of the database site.

- *Sample distribution*: Most statistics methods we looked at requires knowing the distribution of the sample, a requirement impossible to be measured by us. Fortunately, this requirement doesn't pose a problem when the sample size is large, owing to the central limit theorem, which states that when the sample size is large enough, the distribution function approaches the standard normal distribution. In practice, the mean confidence interval statistics we used is highly reliable, even when the distribution of sample is considerably different from normal.

- *Mean interval*: We use the sample mean similarity rate to give us an estimate of the similarity of the database. The estimation is at 99% confidence level [Tri82]. Thus, for a given database DB, with a mean of μ, the mean interval is estimated, as follows:

$$Mean(db,t,th) - E(\alpha,db,t,th) < \mu < Mean(db,t,th) + E(\alpha,db,t,th) \tag{4}$$

where

$$E(\alpha,db,t,th) = \frac{z_{\alpha/2} * Var(db,t,th)}{\sqrt{NuImg(db,t,th)}}, \tag{5}$$

and $\alpha = 0.01$ and $z_{\alpha/2}$ is the normal distribution value with $NuImg > 30$. When $NuImg \leq 30$, we substitute $z_{\alpha/2}$ with Student-t distribution value $t_{\alpha/2}$ with $NuImg - 1$ degree of freedom [Tri82].

- *Sample size*: The selection of the "similarity threshold" is critical for the mean estimation method, as it will impact the sample size (i.e., number of similar images). Statistically, the larger the sample size, the more reliable the estimation of the population mean. Researchers have suggested having a sample size of at least 30 for a good approximation. Ideally, we would like to set the minimum sample size to be at least 100. For a sample size less than 30, the normal distribution parameter should be replaced with student-t distribution parameters.

3.3 Ranking Algorithm

Several approaches are considered when we develop our ranking algorithm. A simple one is to calculate the ranks of the databases based on the number of similar images $NuImg$ for matched templates. This seems to work well when the query is made directly on the template (i.e., the similarity rate between query and template is 1.0), but it falls short when the query pattern is a non-template image. Here, we propose a different approach based on the mean measurement introduced in the previous section. As the population of a normal distribution tends to center around its mean value, we would like the mean of a database site to fall within the "relevant range" of each query. We feel this will give us the highest probability of a relevant match.

Equation 6 illustrates our ranking algorithm. For a given query q and template t, we select database db, if:

$$\{m_1, m_2\}_{db} \in \{a, b\} \tag{6}$$

where

$$m_1 = Mean(db,t,th) - E(\alpha,db,t,th) \tag{7}$$

$$m_2 = Mean(db,t,th) + E(\alpha,db,t,th) \tag{8}$$

$$a = SimRate(q,t) - offset \tag{9}$$

$$b = SimRate(q,t) + offset \tag{10}$$

$$SimRate(q,t) \geq th. \tag{11}$$

Note that our method gives a good measurement of relevancy probability between database sites but it doesn't give an indication of the query result size. Although, the method does favor the sites with more images above the similarity threshold, as the larger the $NuImg$ value, the smaller the mean interval and the more likely it will fall within the "relevant range" completely. Nevertheless, when more than one database site are selected, the site with larger $NuImg$ value is recommended to be searched first.

We can extend this method to boolean operations. For an "and" operation, the selection will require that both mean intervals of a database site must fall into the relevant ranges. For an "or" operation, at least one must fall into the relevant range.

We illustrate our method in the following example. Consider the four databases in Table 1, with the mean and standard deviation information. Assume the similarity rate between query data pattern and icon X and icon Y are 0.82 and 0.83, respectively, and the relevant range offset is set at 0.05. Also the similarity threshold is set at 0.7. The mean intervals for four databases are calculated and shown in Table 2.

	A		B		C		D	
	Icon X	Icon Y	Icon X	Icon Y	Icon X	Icon Y	Icon X	Icon Y
NuImg	50	100	200	90	0	150	30	270
Mean	0.80	0.82	0.79	0.80	0.00	0.85	0.85	0.76
Var	0.05	0.11	0.07	0.09	0.00	0.13	0.10	0.12
E	0.0182	0.0283	0.0128	0.0244	0.000	0.0273	0.0470	0.0188

Table 1: Statistics information for four databases.

	A	B	C	D
Icon X	0.782-0.818	0.777-0.803	0.000-0.000	0.803-0.897
Icon Y	0.792-0.848	0.776-0.824	0.823-0.877	0.741-0.779

Table 2: Mean estimates for four databases.

To find an image that is similar to query data pattern based on the icon X, we can safely first discard database C, since there is no image that is similar to the icon X. For the remaining A, B, and D, we can see that A and B falls into $[0.77, 0.87]$ range. Similarly, for a query based on icon Y, we see that A and C falls into $[0.78, 0.88]$ range.

The final answer given by our search agent to the query depends on user's interest. Let DB be the set of databases. There are at least three possible user interests:

- *All relevant*: the intention of the submitter is to search *all* databases containing relevant images. For this type of interest, the answers returned by the search agent for a given query q over a set of databases DB are:

$$Answer_{all}(q, DB) = \{db \in DB | NuImg(db, t, th) > 0\}, \tag{12}$$

where t is the template and th is the similarity threshold.

For the above example, $Answer_{all}(q, DB) = \{B, A, D\}$ for icon X and {C,A,D,B} for icon Y.

- *Some relevant*: the intention of the submitter is to find *some* databases containing relevant images without searching all databases. For this type of interests, we define our answers as:

$$Answer_{some}(q, DB) = \{db \in DB | \{m_1, m_2\}_{ab} \in \{a, b\}\}, \tag{13}$$

For the above example, $Answer_{some}(q, DB) = \{B, A\}$ for icon X and {C,A} for icon Y.

- *More relevant*: the intention of the submitter is to search the databases which contain the higher number of relevant images than any others. As the information kept by the metaserver about each database is much less detailed than that kept at each database site, it would be very hard for our search agent to accurately guess the number of relevant images in the answer to a given query. Nevertheless, we define our answers to this interest as:

$$Answer_{more}(q, DB) = \{db \in DB | \{m_1, m_2\}_{db} \in \{a, b\} \wedge NuImg(db, t, th) = \max_{db' \in DB}(NuImg(db', t, th))\} \tag{14}$$

For the above example, the $Answer_{more} = \{B\}$ for icon X and {C} for icon Y.

Continue the example with boolean operations, to find an image that is based on both icons X and Y, we first discard database C, as C contains no similar images to X. For the remaining databases, A falls into both relevant ranges. Thus, $Answer_{more} = Answer_{some} = \{A\}$ and $Answer_{all} = \{A, B, D\}$. We put B ahead of D because of higher possible number of images of B (at most 90) than D (at most 30).

To find an image that is based on either icon X or icon Y, we select A, B and C. Thus $Answer_{more} = \{B\}$, $Answer_{some} = \{B, A, C\}$, and $Answer_{all} = \{B, A, C, D\}$. We put B ahead of A and C because of higher possible number of images of B (at most 290) than A and C (at most 150).

3.4 Query Search

Reflecting the two-level metadata organization of database sites in the metadatabase, a user query will be first compared with the templates. One or more query class categories may be derived by selecting the closest templates. The mapping of each user query to templates may be determined by a set of rules. This may require finding a semantic equivalence between the user's description and the template indices for the metadatabase [CM+90] or from features extracted from the query data pattern (for visual queries).

The ranking algorithm is then run by the search module to retrieve the relevant database sites. If the initial site selection does not offer a sufficient pruning of the database sites[1], the search module can further prune these sites based on user feedback before distributing to relevant database sites. Also, if no template can be found for the query data pattern, the metaserver will have the option to create new templates. Under this scenario, all databases will be asked to return the information with respect to the new templates.

4 Experiments

In this section, we present the implementation of the metadatabase and search agent in a web-based system. Experimental results will then be presented.

4.1 The System Implementation

The design approach to access the web-based environment involves a Java-enabled browser accepting a query issued by a user. The user interface is a java applet and the user query is in the form of an image. This image query is forwarded to the metaserver program running on the web server. The results (selected database sites) are displayed to the user to select from. The final query results (names of retrieved images) are also displayed to the user and the images desired by the user can be directly retrieved from the specific databases.

4.1.1 The Metaserver

Whenever a user query comes in, the metaserver must selectively forward it to the relevant databases for efficient searching. Given a query, the metaserver runs a local search to come up with templates matching the query. It then calculates the potential for each remote database by calculating its rank among the databases using the approach proposed in Section 3. The metaserver then selects the top N databases to route the query to.

The functionality of the meta-server is split across two modules. The first module *register module* is responsible for collecting and updating information about the remote databases. The second module *selection module* is responsible for making decisions based on the information in the metadatabase and is also responsible for routing the query to the remote databases. The register module sends each template to every remote database mentioned in the configuration file and stores the retrieved information in files which will subsequently be used by the selection module to make decisions based on the query. The register module takes two configuration files as input. The template configuration file specifies the location of the image data and the location to which information concerning the template should be put whereas the database configuration file contains data about the location (including host and port specifications) of the remote databases.

[1]Note that, since multiple databases could reside at one web site, the word database site refers to the actual database.

At startup, the register module reads information from the configuration files and sends the image data corresponding to each template to all the remote databases. The databases in turn return the results based on the matching images for this template. The register module stores these results in files corresponding to each template. Section 5 will provide the details on this aspect.

The selection module makes decisions based on the information collected by the register module. It stores the template information in a hash table and the database information in an array and listens for requests from users. Upon receiving a user query, a local search is performed to find the matching templates. These are used to lookup the hash table for information about these templates at different database sites. The databases are ranked and the query is forwarded to the top N databases.

Both the register and selection modules support dynamic configuration. Databases and templates can be added dynamically by modifying the configuration files. There are individual server programs running at the remote database sites aiding the metaserver in performing image retrieval. The information server generates information about individual templates passed by the metaserver. The database server retrieves the images matching the user query and sends them back to the metaserver.

A testbed with three databases has been constructed from the Brodatz Collection [Bro66], the VisTex database obtained from the MIT media lab, and a geographical database containing remotely sensed satellite images. Iconic images are randomly selected from image cuts.

The result in the form of retrieved multimedia data is validated by the end user. Key performance factors like *precision and recall* also aid in quantifying the effectiveness of the above approach. Specifically, the search agent at the web server is evaluated by its accuracy in directing the user queries to relevant database sites. Given a query and a set of multimedia databases, we examine the effectiveness of the search agent in two aspects:

- **Precision of selection:** all selected database sites should actually contain the information relevant to what is to be retrieved by the query.
- **Recall of selection:** all database sites which contain the information relevant to the query should be selected.

Assume $sites\_of\_chosen$ to be the set of database sites selected in response to the query and $sites\_of\_relevant$ be the set of database sites that have been judged to contain images that are relevant to the query. To precisely conduct the above evaluation for user queries, we will use the following precision and recall for the selection of database sites:

$$Precision = \begin{cases} \frac{sites\_of\_chosen \cap sites\_of\_relevant}{sites\_of\_chosen} & \text{if } sites\_of\_chosen > 0 \text{ ,} \\ 1 & \text{otherwise.} \end{cases} \tag{15}$$

$$Recall = \begin{cases} \frac{sites\_of\_chosen \cap sites\_of\_relevant}{sites\_of\_relevant} & \text{if } sites\_of\_relevant > 0, \\ 1 & \text{otherwise.} \end{cases} \tag{16}$$

In the following sections, we use P_{all}, P_{some} and P_{more} to represent precision values for different user semantics, and R_{all}, R_{some}, and R_{more} for recall values.

4.2 Testing and Validation

We will now examine the performance of our ranking algorithm. The similarity rates between database images and each template are calculated using the fractal-based similarity comparison technique developed in [ZCA96].

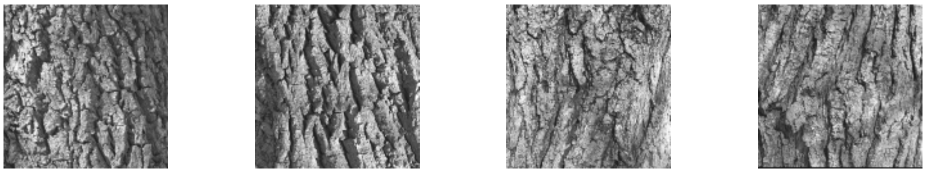
4.2.1 Determining Relevant Offset

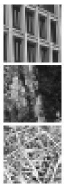
In Section 3, we proposed a construction of "relevant range" that is centered around the similarity rate between query image and template, so that majority of the similarity rates between relevant images and the same template will fall within this range. To determine this range, we have randomly selected several sets of relevant images and compared them to various templates. The results are shown in Table 3.

Image Set	Icon	SimRate of Relevant Images	SimRate Offset
Fense Signs	Bark	0.69,0.70,0.80	0.11
	Flower	0.72,0.75,0.75	0.03
	Food	0.52,0.58,0.61	0.09
Buildings	Grass	0.50,0.52,0.52,0.53	0.03
	Waldo	0.73,0.77,0.81,0.83	0.10
	Painting	0.55,0.56,0.59,0.61	0.06
Barks	Building	0.77,0.77,0.80,0.80	0.03
	Painting	0.52,0.53,0.55,0.56	0.04
	Grass	0.78,0.78,0.89,0.91	0.13
Bricks	Tile	0.91,0.91	0.00
	Flower	0.67,0.77	0.10
	Wood	0.77,0.80	0.03

Table 3: The similarity rates of relevant images.

In seven out of the twelve experiments, we have similarity rates falling within a range of 0.06. For the five trials that have ranges greater than 0.06, four of them have at least half of the images falling within a 0.06 range. The only exception is *Bricks* vs. *Flower*. Thus, we have temporarily chosen 0.06 as the range offset for our experiments. Later, we will try different offset values and see how it affect the performance of our ranking algorithm.





0.77	0.77	0.80	0.80
0.52	0.53	0.55	0.56
0.78	0.78	0.89	0.91

Figure 5: Iconic and database images from VisTex database

4.2.2 Evaluation over Pairs of Databases

In this section, we analyze the ability of our ranking algorithm to distinguish between two databases. We arrange 220 images into two databases A and B, so that A contains all relevant images to the specific query and B contains no relevant images to the query. Here we define the query image and the database image to be "relevant" if their similarity rate is greater than 0.7. With this condition, we know that the correct answers for the various query should always be $Answer_{all} = Answer_{some} = Answer_{more} = \{A\}$. The results of our experiments are shown in Table 4. The average precision and recall values for all 15 queries are shown in Table 5.

The values for R_{some} and R_{more} are generally lower than P_{some} and P_{more}. This is because our ranking algorithm chooses the most promising databases, not all of the ones potentially containing matching documents. Therefore, some matching databases may be missed. The value for P_{all} is also slightly low. This is not surprising, as in this case, our ranking algorithm returns all non-empty databases, and not everyone of them will contain relevant images.

We also notice that when SimRate ≥ 0.80, four out of five times our ranking algorithm is able to choose A as the relevant database. The results are mixed when the SimRate is between 0.79 and 0.60. When the SimRate is below 0.60, no databases are selected, as our ranking algorithm can no longer distinguish the two databases. The reason for this is that when the similarity between query data pattern and the template becomes less similar (generally below 0.8), the template is no longer a good base to be used in the estimation. The statistics data of the tests with SimRate ≥ 0.80 are shown in Tables 6, 7, and 8. Table 9 shows the average P and R values with respect to different ranges of SimRate, and it clearly shows a deterioration of the performance with SimRate < 0.80.

4.2.3 Evaluation over Three Databases

In this section, we examine our ranking algorithm over three databases. We add a third database C. Unlike database B, which contains no relevant images to the query, database C does contain some relevant images but not as many as A. Also, the images in C have no similarity rates greater than 0.92 to the queries. With this condition, we know that the correct answers for the various queries should always be $Answer_{all} = \{A, C\}$, $Answer_{some} = \{A, C\}$, and $Answer_{more} = \{A\}$. We

Query Pattern	Media Icon	SimRate	$Answer_{all}$	$Answer_{some}$	$Answer_{more}$
Waldo	Food	**0.88**	A,B	A	A
Fense Sign	Tile	0.88	A,B	A,B	A
Grass	Bark	**0.86**	A,B	A	A
Building	Waldo	**0.83**	A,B	A	A
Grass	Flower	**0.81**	A,B	A	A
Wood	Grass	0.78	B,A	B,A	B
Bark	Building	**0.77**	B,A	B,A	B
City	Terrain	0.75	A,B	A,B	A
Terrain	Sand	0.73	A,B	-	-
Water	Brick	0.70	A,B	-	-
Flower	Fense Sign	0.67	B,A	A	A
City	Metal	0.62	A,B	A	A
Flower	Wood	0.62	A,B	A,B	A
City	Water	0.56	A,B	-	-
Fense Sign	Grass Land	0.56	A,B	-	-

Table 4: The experimental results with two databases.

	All	**Some**	**More**
Precision	**0.5**	**0.8333**	**0.8667**
Recall	**1.0**	**0.7333**	**0.6000**

Table 5: Parameters P and R for the two databases configuration.

report the experimental results in Table 10. The average precision and recall values are shown in Table 11. The statistics data of the database C with SimRate $\geq$ 0.80 are shown in Tables 12 and 8.

Again, we observe good test results with SimRate $\geq$ 0.80. The high P_{some} and P_{more} values (0.8889 and 0.8667, respectively) indicate that our ranking algorithm is useful for exploring some of the relevant/best databases.

4.2.4 Selecting Threshold

Selecting the right "similarity threshold" is a critical task. To determine a proper threshold to accommodate various queries, we start with a "similarity threshold" as low as 0.1, then gradually increase the value with an increment of 0.025 until all $Mean$ values are as close to the $SimRate$

	Database A				
Icon	Food	Tile	Bark	Waldo	Flower
Threshold	0.80	0.75	0.73	0.67	0.70
NuImg	37	56	44	97	38
Mean	0.865	0.874	0.843	0.822	0.791
Var	0.056	0.0555	0.0704	0.080	0.076
E	0.0237	0.0191	0.0273	0.0210	0.0317

Table 6: The statistics information for database A.

	Database B				
Icon	Food	Tile	Bark	Waldo	Flower
Threshold	0.80	0.75	0.73	0.67	0.70
NuImg	17	44	30	128	38
Mean	0.843	0.870	0.822	0.788	0.764
Var	0.0478	0.0589	0.0652	0.0825	0.0574
E	0.0336	0.0229	0.0307	0.0188	0.024

Table 7: The statistics information for database B.

as possible, but not exceeding the $SimRate$ value. The preliminary results indicate that no single threshold can be used for all queries. Rather, the threshold is a function of SimRate. We plot the values used in our tests in Figure 6.

Except the last test, the threshold seems to generally fall between 80% to 90% of the SimRate value.

4.2.5 Adjusting Range Offset

In this section, we analyze the impact of range offset on the performance of our ranking algorithm. Tables 13 and 14 show the test results of offsets 0.4 and 0.8 over two databases with SimRate ≥ 0.80.

Clearly, when an offset becomes bigger, more databases will fall within the "relevant range." Thus, the precision value is decreased. Similarly, when an offset becomes smaller, less databases will fall within the "relevant range", thus decreases the recall value. We believe that an ideal offset should be as small as possible but still covers the majority of the relevant images. Different methods used in calculating the similarity rates and different image types may result in different offset values. We have found that, for our test images, 0.06 seems to be an adequate choice.

Icon	Food	Tile	Bark	Waldo	Flower
Database A	**0.841-0.889**	**0.855-0.893**	**0.816-0.870**	**0.801-0.843**	**0.759-0.823**
Database B	**0.809-0.877**	**0.847-0.893**	**0.791-0.853**	**0.769-0.807**	**0.740-0.788**
Database C	**0.828-0.887**	**0.851-0.894**	**0.804-0.865**	**0.760-0.826**	**0.742-0.820**
Relevant Range	**0.82-0.94**	**0.82-0.94**	**0.80-0.92**	**0.77-0.89**	**0.75-0.87**

Table 8: The interval estimates and relevant ranges.

SimRate	P_{all}	R_{all}	P_{some}	R_{some}	P_{more}	R_{more}
0.80 - 1.00	0.5	1.0	0.9	1.0	1.0	1.0
0.60 - 0.79	0.5	1.0	0.75	0.75	0.75	0.5
0.00 - 0.59	0.5	1.0	1.0	0.0	1.0	0.0

Table 9: Parameters P and R for two databases configuration vs SimRate ranges.

4.2.6 Discussion

We have reported some preliminary test results on our ranking algorithm. We found that, with careful selections of templates and range offset, our ranking algorithm performs well for the *all relevant* and *some relevant* queries. The performance deteriorate when the SimRate decreases. Although we have mentioned that our ranking algorithm is not intended to support the *more relevant* queries, we have been getting very good test results. The reason for this is that even though we will not know the result sizes without actually searching the databases, it is usually the case that the relevancy probability is higher if a database contains more images similar to the templates. We also give an estimate of the threshold as a function of SimRate.

5 Filling and Refinement of Metadatabase

In this section, we will discuss the filling process of the metadatabase. In this process, the addition of new metadata and refinement of the existing metadata will be considered.

5.1 Database Registration

When the metadatabase is first constructed or when a new database is added, each database reports initial relevant metadata. First, the templates representing the multimedia data in the database are added to the metadatabase if the templates do not exist in the metadatabase. Second, the number of "similar" images with respect to each template in the database is reported, which is represented by $NuImg$. To supply this information, all images in the database should have been compared to the given template and those that have a similarity rate greater than a given "similarity threshold",

Query Pattern	Media Icon	SimRate	Answer$_{all}$	Answer$_{some}$	Answer$_{more}$
Waldo	Food	**0.88**	A,C,B	A,C	A
Fense Sign	Tile	0.88	A,C,B	A,C,B	A
Grass	Bark	**0.86**	A,C,B	A,C	A
Building	Waldo	**0.83**	B,A,C	A	A
Grass	Flower	**0.81**	A,B,C	A	A
Wood	Grass	0.78	B,A,C	B,A,C	B
Bark	Building	**0.77**	B,A,c	B,A,C	B
City	Terrain	0.75	A,C,B	A,C,B	A
Terrain	Sand	0.73	A,C,B	-	-
Water	Brick	0.70	A,C,B	-	-
Flower	Fense Sign	0.67	B,A,C	A	A
City	Metal	0.62	A,C,B	A,C	A
Flower	Wood	0.62	A,C,B	A,C,B	A
City	Water	0.56	A,C,B	-	-
Fense Sign	Grass Land	0.56	A,C,B	-	-

Table 10: The experimental results with three databases.

	All	**Some**	**More**
Precision	**0.667**	**0.8889**	**0.8667**
Recall	**1.0**	**0.6333**	**0.6000**

Table 11: Parameters P and R for the three databases configuration.

th, are considered. The mean ($Mean$) of the similarity rates of these selected images as well as the standard deviation Var are calculated and reported. The values of $NuImg$, $Mean$ and Var are used to calculate the estimated mean interval. The resulting mean estimates are then stored in the metadatabase. Same procedures are applied to all pre-defined templates.

When new templates are added into the metadatabase, the relationships between these templates and all databases should be established. This is performed by submitting the templates to the databases which have not been polled using these templates and collecting the values of $NuImg$, $Mean$ and Var. Note that such processes should be performed off-line and during non-busy machine hours.

Much work has been done on resource discovery over the Internet [Web, LD95, ODL93, SEKN93, BDH$^+$94, GGM95, GGMT94, CLC95]. Since this aspect is not the focus of the paper, we assume

	Database C				
Icon	**Food**	**Tile**	**Bark**	**Waldo**	**Flower**
Threshold	**0.80**	**0.75**	**0.73**	**0.67**	**0.70**
NuImg	**23**	**48**	**36**	**40**	**23**
Mean	**0.857**	**0.872**	**0.834**	**0.793**	**0.781**
Var	**0.0504**	**0.0577**	**0.0710**	**0.0804**	**0.067**
E	**0.0295**	**0.0215**	**0.0305**	**0.0327**	**0.0392**

Table 12: The statistics information for database C.

Query Pattern	**Media Icon**	**SimRate**	**$Answer_{all}$**	**$Answer_{some}$**	**$Answer_{more}$**
Waldo	Food	**0.88**	A,B	A	A
Fense Sign	Tile	0.88	A,B	A,B	A
Grass	Bark	**0.86**	A,B	-	-
Building	Waldo	**0.83**	A,B	A	A
Grass	Flower	**0.81**	A,B	-	-

Table 13: Experimental results of two databases with 0.4 offset.

the discovery of new database resources are done manually. Once discovered, the new database will be asked to register with the metaserver following the steps outlined above.

5.2 Metadatabase Refinement

The databases involved in the metadatabase may be updated. In such a case, the initial metadata recorded may become inaccurate in representing the databases. Thus, the metadatabase must be dynamically updated. Metadatabase refinement involves two aspects:

- Updating existing metadata based on database updates, and
- Updating existing metadata based on query feedback.

As mentioned in Subsection 5.1, new templates may be reported from the databases. In addition, users accessing the system may also request to add new templates. For each new template added to the metadatabase, poll will be conducted by submitting the templates to the databases and collecting the values of $NuImg$, $Mean$ and Var. An utilization value is also maintained for each template. The value is defined as number of times the query is executed via the template. The refining module is applied periodically to delete templates with low utilization values.

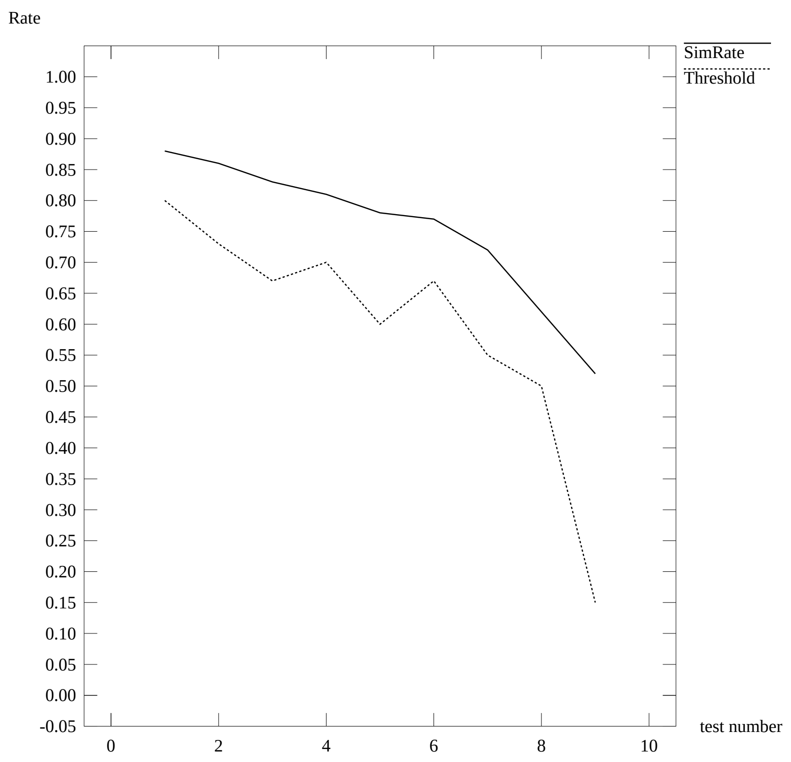


Figure 6: Comparisons of threshold and SimRate.

Refinement of the current metadatabase when the database is updated is a difficult task. Since we consider it impractical to require the databases to report their status every time they are updated, the refinement of the metadatabase can only be based on careful evaluation and validation of the query results. This is proven to be non-trivial, since we have no knowledge regarding whether the results are indeed relevant to the query without the actual inputs from the user. Furthermore, the correctness of the produced answer depends on the user's semantics for the query. Thus, the refinement process must consider user semantics when validating the query results. To resolve these issues, some artificial intelligence techniques must be employed. Since this is beyond the scope of this research, we currently rely on the users to validate the query feedback. We analyze the user inputs periodically and ask selected sites to resubmit statistics when the performance is not of satisfactory. The actual re-submission and updation of the metadata will be performed off-line.

To keep track of all the updates of the metadata, a log file is maintained by the refining module. The log file will record the latest submissions of each database and last time it was contacted by the meta server. The log file will also include information such as template additions/deletions and new database registrations.

Query Pattern	Media Icon	SimRate	$Answer_{all}$	$Answer_{some}$	$Answer_{more}$
Waldo	Food	**0.88**	A,B	A,B	A
Fense Sign	Tile	0.88	A,B	A,B	A
Grass	Bark	**0.86**	A,B	A,B	A
Building	Waldo	**0.83**	A,B	B,A	B
Grass	Flower	**0.81**	A,B	A,B	A,B

Table 14: Experimental results of two databases with 0.8 offset.

6 Conclusion

We have developed a system framework for the creation of a metadatabase and the design of a metaserver search agent which supports integrated access to various multimedia databases. The creation of the metadatabase involved a two-level organization of templates and metadata on component databases. Such metadata are based on the types of media data housed at each multimedia database site and on their query capabilities. The search agent located at the metaserver accesses the metadatabase and control the distribution of user queries to relevant database sites. The proposed metadatabase and search agent have been designed and implemented in a Web-based multimedia information retrieval environment. Preliminary experiments have demonstrated the effectiveness the approaches.

We plan to further perform experimental analysis on the proposed approaches using larger number of databases and images. We would also evaluate the sensitivity of various parameters set up in the proposed mechanisms.

References

[ACF+93] Manish Arya, William Cody, Christos Faloutsos, Joel Richardson, and Arthur Toga. QBISM: A Prototype 3-D Medical Image Database System. *IEEE Data Engineering Bulletin*, 16(1):38–42, 1993.

[BDH+94] M. Bowman, P. Danzig, D. Hardy, U. Manber, and M. Scwartz. Harvest: A scalable, customizable discovery and access system. Technical Report CU-CS732-94, Department of Computer Science, University of Colorado-Boulder, 1994.

[BPJ93] J.R. Bach, S. Paul, and R. Jain. A Visual Information Management System for the Interactive Retrieval of Faces. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):619–628, 1993.

[Bro66] P. Brodatz. *Textures: A Photographic Album for Artists and Designers.* Dover, New York, 1966.

[CLC95] J. Callan, Z. Lu, and B. Croft. Searching Distributed Collections with Inference Networks. In *Proceedings of the 18th Annual SIGIR Conference*, 1995.

[CM+90] N. Cercone, M. Morgenstern, et al. Resolving semantic heterogeneity. In *Sixth IEEE International Conference in Data Engineering*, 1990.

[COM95] Special Issue on Content-Based Image Retrieval Systems. *IEEE Computer*, 28(9), 1995.

[CYDA88] S.K. Chang, C.W. Yan, Donald C. Dimitroff, and Timothy Arndt. An Intelligent Image Database System. *IEEE Transaction on Software Engineering*, 14(5):681–688, May 1988.

[GGM95] L. Gavarno and H. Garcia-Molina. Generalizing Gloss to Vector-Space Databases and Broker Hierachies. In *Proceedings of the 21st International Conference on Very Large Data Bases*, pages 78–89, 1995.

[GGMT94] L. Gavarno, H. Garcia-Molina, and A. Tomasic. The Effectiveness of Gloss for the Text Database Discovery Problems. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Minneapolis, May 1994.

[HLHC92] Tai-Yuan Hou, Peiya Liu, Arding Hsu, and Ming-Yee Chiu. Medical Image Retrieval by Spatial Features. In *IEEE Conference on Systems, Man, and Cybernetics*, 1992.

[JM95] R. Jain and S.N.J. Murthy. Similarity Measures for Image Databases. In *Proceedings of the SPIE Conference on Storage and Retrieval of Image and Video Databases III*, pages 58–67, 1995.

[LD95] C. Lagoze and J. Davis. Dienst: An architecture for distributed document libraries. *Communications of ACM*, 38(4):47, April 1995.

[ODL93] K. Obraczka, P. Danzig, and S-H Li. Internet Resource Discovery Services. *IEEE Computer*, 26(9), 1993.

[RSZ96] E. Remias, G. Sheikholeslami, and Aidong Zhang. Block-Oriented Image Decomposition and Retrieval in Image Database Systems. In *Proceedings of the 1996 International Workshop on Multi-media Database Management Systems*, Blue Mountain Lake, New York, August 1996.

[RSZSM96] E. Remias, G. Sheikholeslami, Aidong Zhang, and Tanveer F. Syeda-Mahmood. Supporting Content-Based Retrieval in Large Image Database Systems. *The International Journal on Multimedia Tools and Applications*, 1996. (to appear).

[SC94] John R. Smith and Shih-Fu Chang. Quad-Tree Segmentation for Texture-Based Image Query. In *Proceedings of ACM Multimedia 94*, pages 279–286, San Francisco, California, October 1994.

[SEKN93] M. Scwartz, A. Emtage, B. Kahle, and C. Neuman. A Comparison of Internet Resource Discovery Approaches. *IEEE Computer*, 26(9), 1993.

[SM93] T.F. Syeda-Mahmood. *Attentional Selection in Object Recognition.* PhD thesis, Department of Computer Science, MIT, May 1993.

[SNA+95] M. Flickner H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by Image and Video Content: The QBIC System. *IEEE Computer*, 28(9), 1995.

[SZ94] S.W. Smoliar and H.J. Zhang. Content-Based Video Indexing and Retrieval. *IEEE Multimedia*, 1(2):62–72, 1994.

[TPF+91] A. Turtur, F. Prampolini, M. Fantini, R. Guarda, and M.A. Imperato. IDB: An Image Database System. *IBM Journal of Research and Development*, 35(1):88–96, January 1991.

[Tri82] Kishor Trivedi. *Probability & Statistics with Reliability, Queuing, and Computer Science Applications.* Prentice Hall, 1982.

[Web] K. Webster. Cornell project saves documents,books – and makes them accessible. *Advanced Imaging*, pages 42–46.

[WN94] Jian Kang Wu and Arcot Desai Narasimhalu. Identifying Faces Using Mutiple Retrievals. *IEEE Multimedia*, 1(2):27–38, 1994.

[ZCA95] Aidong Zhang, Biao Cheng, and Raj Acharya. An Approach to Query-by-texture in Image Database Systems. In *Proceedings of the SPIE Conference on Digital Image Storage and Archiving Systems*, pages 338–349, Philadelphia, October 1995.

[ZCA96] Aidong Zhang, Biao Cheng, and Raj Acharya. A Fractal-Based Clustering Approach in Large Visual Database Systems. *The International Journal on Multimedia Tools and Applications*, 3(3):225–244, November 1996.