ELSEVIER

The Journal of Systems and Software 64 (2002) 131–149

The Journal of Systems and Software

www.elsevier.com/locate/jss

Heterogeneous image database selection on the Web

Deok-Hwan Kim [a], Seok-Lyong Lee [b], Chin-Wan Chung [c,*]

[a] Department of Information and Communication Engineering, Korea Advanced Institute of Science and Technology, 373-1, Kusong-dong, Yusong-gu, Taejon 305-701, Republic of Korea

[b] School of Industrial and Information Systems Engineering at Hankuk University of Foreign Studies, 89, Wangsan-ri Mohyun, Yongin-si, Kyounggi-do, Republic of Korea

[c] Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, 373-1, Kusong-dong, Yusong-gu, Taejon 305-701, Republic of Korea

Received 24 February 2001; received in revised form 18 July 2001; accepted 9 May 2002

Abstract

Image databases on the Web have heterogeneous characteristics since they use different similarity measures and queries are processed depending on their own schemes. In the content-based image retrieval from distributed sites, it is crucial that the metaserver has the capability to find objects, similar to a given query object in terms of the global similarity measure, from different image databases with different local similarity measures. In this paper, we investigate the problem of finding databases, which contain more objects relevant to a given query than other databases, from many image databases dispersed on the Web. This problem is referred to as a database selection problem.

We propose a new selection method to determine candidate databases. The selection of databases is based on the hybrid estimator using a few sample objects and compressed histogram information of image databases. Extensive experiments on a large number of image data demonstrate that our proposed method improves the effectiveness of distributed content-based retrieval in a heterogeneous environment.

Keywords: Database selection; Similarity search; Hybrid estimator; Image database

1. Introduction

Emerging new multimedia applications, such as digital libraries, medical diagnostic systems, remote site education, distributed publishing and electronic commerce, need to access information from image databases distributed at remote locations and process queries in a distributed manner. Currently, there are a number of databases on the Web which contain visual objects such as images and video frames, and it is increasingly important to retrieve them.

Contrary to the traditional databases such as relational databases which retrieve relevant tuples to a query, multimedia databases retrieve visual objects using the content-based retrieval method called 'similarity query'. The similarity query retrieves visual objects similar to a query object q using the similarity measure. For example, let $\mathrm{sim}(q,x)$ be the similarity function which maps the similarity between two visual objects, q and x, into a real number whose range is $[0,1]$. The larger the value is, the more similar two objects are. Then the similarity query is to find a set of objects x to satisfy $\mathrm{sim}(q,x) > T$, where T is the threshold value.

We call a similarity query in the distributed environment like the Web as 'distributed similarity query'. There are different databases on the Web which have visual objects. In such an environment, we need metaservers in order to handle a distributed similarity query efficiently. The scenario of the distributed similarity query on the Web is as follows: A user gives a query with a query object and a global threshold (GT) to a metaserver. Then the metaserver sends the user's query to the image databases. After the query is evaluated at each

* Corresponding author.
E-mail addresses: dhkim@islab.kaist.ac.kr (D.-H. Kim), sllee@hufs.ac.kr (S.-L. Lee), chungcw@islab.kaist.ac.kr (C.-W. Chung).

0164-1212/02/$ - see front matter © 2002 Elsevier Science Inc. All rights reserved.
PII: S0164-1212(02)00032-8

database, the metaserver merges query results from databases, and presents them in a sorted order to the user.

If the metaserver searches all databases on the Web with respect to the user's query, it will take too much time to complete the query. To avoid such an exhaustive process, the metaserver has to provide a way to narrow down the search scope to a few candidate databases. This is called 'database selection problem'. Until now, various approaches have been attempted to solve the database selection problem on text databases while only a few researches have been made for image databases in spite of the importance of the image database selection problem. In the paper, we focus on the image database selection problem in the distributed environment like the Web.

1.1. Problem definition

There are various kinds of image databases. In order to collect image data from them efficiently through a distributed similarity query, we must know their characteristics and functions well. The difficulty of distributed similarity search is that these image databases have different characteristics and functions among them. That is, they have heterogeneous characteristics and functions. Moreover, they are not designed to be performed under some predefined network architecture. They have their own autonomous data management systems. In order for them to be used for distributed similarity search by the metaserver, we must resolve the heterogeneity and autonomy. Let us start with the discussion of those characteristics.

Heterogeneity: It is usual that the similarity of two images is derived from the distance between their feature vectors in the feature space. A feature vector is defined for a particular attribute such as color, texture and shape. Databases in the distributed environment may have different attributes. In the case of color, there are various approaches for the color representation such as the color histogram, the average color, and the major color (Crane, 1997; Wyszecki and Stiles, 1982). The color histogram is one of the most widely used visual features. Various color spaces like RGB, HSV, and YCbCr can be used to represent the color histogram of images. The distance between two color feature vectors is frequently used to measure the color similarity. In RGB and YCbCr color spaces, the Euclidean distance is generally used for a distance measure. In HSV color space, however, the angular distance is used to measure the color similarity since features in the space are represented in the form of a corn using the polar coordinate (Kanai, 1998). It is natural that the image databases at different sites may have different similarity measures. Before the queries are performed on the heterogeneous image databases, the mismatches of the similarity measures must be resolved.

Autonomy: Most of image databases are autonomous in their data managements in the context that they are not designed to be controlled by a certain metaserver. They may have their own storages and index structures to store image data, and the summary data to optimize similarity queries. This summary information is not provided to the other site, that is, a metaserver. However, the metaserver must maintain metadata collected from image databases in order to perform a distributed similarity query efficiently and effectively. It means we need the architecture which collects the information necessary for the query processing from autonomous image databases. There are various ways to do this. For example, we force the existing image databases to have predefined query interfaces in order for a metaserver to collect the information. Or, the metaserver can send an agent program to each image database so that the agent program run a job to collect information from each image database needed for distributed similarity search. The autonomous databases with these interfaces or agent programs are referred to as the *semi-autonomous* databases.

Content evolution: The content of databases may change after some updates. The metaserver must keep the up-to-date content summary information. Otherwise, the query to the old content of the metaserver may leads to wrong answers.

With these considerations, the problem we address in this paper is how to select image databases which are the most relevant to a given query from multiple image databases that are semi-autonomous and heterogeneous. The term 'relevant database' means the database that has more objects similar to the query object than others. The term 'global similarity measure' means the similarity measure of the metaserver and the term 'local similarity measure' means that of an image database. Let a metaserver get a list of image databases db_i ($1 \leqslant i \leqslant S$), sample objects o_{ij} ($1 \leqslant j \leqslant n$) from each database db_i, and the compressed histogram information for the selectivity estimation from each database, where S is the number of image databases and n is the number of sample objects from db_i. Then, we define the problem in this paper formally as follows:

- Given: a query object q, the global similarity measure, a global similarity threshold GT and the number of image databases M to be selected ($M \leqslant S$).
- Target: Select M image databases based on the relevance to q.

1.2. Brief sketch of our method

A metaserver constructs and maintains the summary information regarding image databases. However, it is impractical for a single metaserver to collect metadata from all databases on the Internet in a timely manner.

Therefore, we consider that several metaservers keep their own metadata to balance the load and to avoid the single-point-failure. In this case, we assume that a metaserver can duplicate and delegate metadata to others and remove them easily. The focus of this paper, however, is not the architecture of the Web database but the database selection. Therefore, we assume that there exists one metaserver. We also assume that local databases are semi-autonomous, that is, they export content summaries such as compressed histogram information and local similarity measures to the metaserver through the interfaces only. However they do not routinely provide fetching of sample objects. The metaserver should use the query interface in order to fetch them from image databases.

Our approach is composed of two phases: *preprocessing* and *database ranking*. A brief sketch of those phases is illustrated in Fig. 1, and a brief explanation follows.

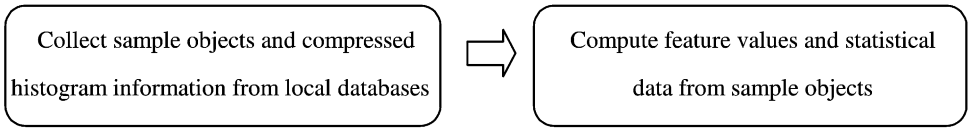
Preprocessing phase: The summary information of image databases is collected background whenever a new database is registered to the metaserver or a registered database is updated largely. Multi-dimensional histograms are generated from image feature data at image databases in order to estimate the selectivity of each database for a given query. We use the compressed histogram information using the discrete cosine transformation (DCT) proposed in Lee et al. (1999) since it reduces the storage overhead and the network transmission cost. In addition, the metaserver extracts sample objects using the progressive query-based sampling method, which slightly modifies the sampling method in (Callan et al., 1999; Provost et al., 1999), and computes statistical metadata such as correlation coefficients, the mean, and the standard deviation from the feature set of sample objects.

Database ranking phase: To solve the image database selection problem, we propose a hybrid selectivity estimation method by using a few sample objects and the histogram information of each image database. The metaserver captures statistical data on the similarity distribution of sample objects using the regression analysis. A given user's global threshold (GT) is translated into the local threshold (LT) of each image database using statistical data since image databases use their own similarity measures in the heterogeneous environment. The metaserver estimates the result size of a query on each image database using its histogram information and LT. The result size of the query estimated by the local similarity measure of the image database may be different from that estimated from the global similarity measure of the metaserver. A sample selectivity compensation (SSC) technique is developed to compensate for the difference between them using sample objects. Based on this hybrid estimator, the result size of the query using GT is estimated, which is the key criterion to rank and select databases.

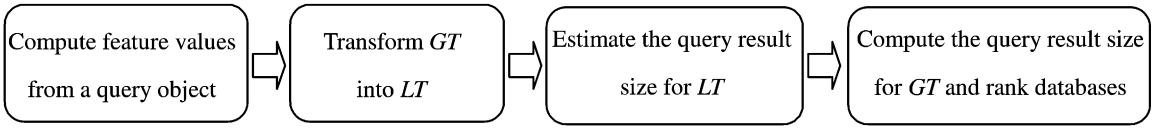
1.3. Contribution

The database selection process at a metaserver is an essential work for efficient retrieval of image data from a number of remote locations. In this paper, we propose a user transparent database selection method, based on the hybrid estimator of the result sizes of the similarity queries. This method has the following desirable properties:

(1) It is designed to rank image databases more exactly from the global similarity measure's point of view when the global similarity measure and local similarity measures are different. Different similarity measures cause the difference of the estimated result size of a query. To solve this issue, we propose the SSC technique to compensate for the difference of the estimated size between them.
(2) It provides an efficient and effective method for estimating the number of relevant objects to the query by a small size of sample objects and a small number of DCT coefficients. Therefore, the storage and time



(a) Preprocessing phase



(b) Database ranking phase

Fig. 1. Overview of the image database selection.

requirement of a metadatabase is reasonable for large image databases.

(3) The method can reflect the update at image databases timely. The compressed histogram information using DCT is used to keep the up-to-date information of image databases. Since it is small-sized even for a multi-dimension and updated dynamically when the images of the database are changed, it can be transferred to the metaserver from an image database timely, with low transmission cost.

1.4. Paper organizations

The remainder of this paper is organized as follows: Section 2 provides a survey of related work with a brief discussion on database selection. In Section 3 we define simple distance measures and show our observations about the relationship between heterogeneous similarity measures. The database selection process is described in Section 4 with a hybrid scheme to estimate the number of objects relevant to the query at each image database and a selection algorithm to determine candidate databases. Section 5 explains the framework of our experiment and presents a series of experimental results. We give conclusions in Section 6.

2. Related work

A lot of studies have been made for the database selection problem on text databases (Callan et al., 1995; Gravano et al., 1994; Gravano and Garcia-Molina, 1995; Meng et al., 1998; Meng et al., 1999; Xu et al., 1998; Yuwono and Lee, 1997). Gravano et al. (1994, 1995), proposed a keyword-based distributed database broker system based on a Boolean and vector-space retrieval model to estimate the number of potentially relevant documents in a database to a given query. Callan et al. (1995) presented a probabilistic model of information retrieval based on the inference network. Meng et al. (1998, 1999) also proposed methods for estimating the usefulness of text databases based on the probabilistic model. These traditional methods for the text database selection, however, may not be applicable for image databases since there exists a semantic gap between vectors of text databases and feature vectors of image databases.

There are different image databases on the Web such as QBIC (Flickner et al., 1995), Virage (Bach et al., 1996), WebSEEk (Smith and Chang, 1997), and VisualSEEk (Smith and Chang, 1996), to name a few. A recent work for the image database selection was made by Chang et al. (1998). They proposed mean-based and histogram-based selection approaches that use the visual similarity of the query with respect to templates which are representative images from image clusters in local databases, and the statistical data of clusters associated with templates. The mean-based approach uses (1) the number of samples and (2) the mean and variance of the similarity distribution of database images with respect to a template, to determine the likelihood of a cluster to a given visual query. The histogram-based approach is based on not only the statistics of the similarity distribution (represented as a histogram) of database images but the locations of the images within a image cluster. However, they assume that image databases support the same feature extraction method and distance function as the metaserver. Thus, this approach can be used restrictively in a realistic environment since most databases on the Web use different similarity measures.

On the other hand, Benitez et al. (1998) proposed a content-based meta-search engine for images called the MetaSEEk. Given a visual query, the MetaSEEk ranks image databases using the historical data of the relevance feedback made by users. However, in this approach, if a specific database was changed much, the past information may not be valid anymore since the historical data is not updated dynamically, which leads to an incorrect selection of databases.

3. Heterogeneous similarity measures

In this section, we define simple distance measures for similarity retrieval of large image databases on the Web, and describe the relationship between the global similarity measure and a local similarity measure by using various statistical data.

3.1. Distance and similarity

Color histograms are popular methods to represent the distribution of colors in images where each histogram bin represents a color in one of various color spaces (RGB, YCbCr, HSV, etc). However, the distance measure for the color histogram is computationally expensive during query processing since the histogram represents a high-dimensional distribution (at least 32 or 64 color bins).

Therefore, we present a low dimensional distance measure, called the regional average color distance, which is the function with respect to the distance between the average color distribution of a region of an image and that of the corresponding region of another image. Each image is partitioned into k subimages of an equal size. Let $\vec{C} = [\vec{C}_1, \ldots, \vec{C}_k]$ represent the color of an image and each regional color, $\vec{C}_i$, be composed of $3 \times p$ matrix whose jth column is the color $c_{ij} = [\alpha_{ij}, \beta_{ij}, \gamma_{ij}]^{\mathrm{T}}$, where $j = 1, \ldots, p$ represents a bin number and $\alpha_{ij}, \beta_{ij}, \gamma_{ij}$ represent the magnitude of color components (for example, H, S, V) for the jth bin of the ith region. Given p-dimensional color histograms of k subimages for each

of two images, $\vec{x}_1,\ldots,\vec{x}_k$ and $\vec{y}_1,\ldots,\vec{y}_k$, 3×1 regional average color vectors for the $2k$ subimages are

$$\vec{x}_{\mathrm{avg},i}=\vec{C}_i\vec{x}_i \quad \text{and} \quad \vec{y}_{\mathrm{avg},i}=\vec{C}_i\vec{y}_i \quad \text{for } i=1,\ldots,k$$

Definition 1. The regional average color distance in RGB, YCbCr color space is defined as:

$$\begin{aligned} d^2_{\mathrm{avg}} &= \sum_{i=1}^{k}[\vec{x}_{\mathrm{avg},i}-\vec{y}_{\mathrm{avg},i}]^{\mathrm{T}}[\vec{x}_{\mathrm{avg},i}-\vec{y}_{\mathrm{avg},i}] \\ &= \sum_{i=1}^{k}\Big[(x_{\mathrm{avg},\alpha_i}-y_{\mathrm{avg},\alpha_i})^2+(x_{\mathrm{avg},\beta_i}-y_{\mathrm{avg},\beta_i})^2 \\ &\quad +(x_{\mathrm{avg},\gamma_i}-y_{\mathrm{avg},\gamma_i})^2\Big] \end{aligned} \tag{1}$$

where $\alpha_i=r_i$, $\beta_i=g_i$, $\gamma_i=b_i$ for RGB color space and $\alpha_i=y_i$, $\beta_i=Cb_i$, $\gamma_i=Cr_i$ for YCbCr color space, $i=1,\ldots,k$.

Definition 2. The regional average color distance in HSV color space is defined as (Kanai, 1998):

$$\begin{aligned} d^{2'}_{\mathrm{avg}} &= \sum_{i=1}^{k}[\vec{x}_{\mathrm{avg},i}-\vec{y}_{\mathrm{avg},i}]^{\mathrm{T}}[\vec{x}_{\mathrm{avg},i}-\vec{y}_{\mathrm{avg},i}] \\ &= \sum_{i=1}^{k}\Big[(x_{\mathrm{avg},v_i}-y_{\mathrm{avg},v_i})^2+\Big(x_{\mathrm{avg},v_i}x_{\mathrm{avg},s_i}\cos\Big(\frac{x_{\mathrm{avg},h_i}\pi}{3}\Big) \\ &\quad -y_{\mathrm{avg},v_i}y_{\mathrm{avg},s_i}\cos\Big(\frac{y_{\mathrm{avg},h_i}\pi}{3}\Big)\Big)^2+\Big(x_{\mathrm{avg},v_i}x_{\mathrm{avg},s_i}\sin\Big(\frac{x_{\mathrm{avg},h_i}\pi}{3}\Big) \\ &\quad -y_{\mathrm{avg},v_i}y_{\mathrm{avg},s_i}\sin\Big(\frac{y_{\mathrm{avg},h_i}\pi}{3}\Big)\Big)^2\Big] \end{aligned} \tag{2}$$

where $\vec{x}_{\mathrm{avg},i}=(x_{\mathrm{avg},h_i},x_{\mathrm{avg},s_i},x_{\mathrm{avg},v_i})$, $\vec{y}_{\mathrm{avg},i}=(y_{\mathrm{avg},h_i},y_{\mathrm{avg},s_i},y_{\mathrm{avg},v_i})$, $i=1,\ldots,k$.

Note that the regional average color of sample objects can be precomputed in the preprocessing phase and then organized easily into a metadatabase. Furthermore, the regional average color distance is the lower bound on the histogram distance measure (Hafner et al., 1995). Therefore, it, as a cheaper distance measure, can be used to rank relevant databases to the query without any false dismissal.

The regional average color distance is converted into the similarity using the inter-feature normalization technique suggested in MARS (Ortega et al., 1998). A brief description is as follows:

(a) Compute the distance between all pairs of sample images.
(b) Calculate and store the mean μ^* and the standard deviation σ^* of distances.
(c) Calculate distance values between database images and a given query image.
(d) Apply the Gaussian normalization to the distance values obtained in (c) using μ^* and σ^* so that 99 percent of distance values fall in the range $[-1,1]$. Let d be one of these values.
(e) Let $s=(d+1)/2$ in order to map d to a value ranged $[0,1]$.
(f) Then similarity $=1-s$.

3.2. Relationship between similarity measures

Two similarity terms, global and local, are formally defined as follows:

Definition 3. The *global similarity*, $\mathrm{sim}_{\mathrm{global}}(q,o)$, is defined as the similarity value between a query image q and an image o that is calculated by the similarity measure of the metaserver. The *local similarity*, $\mathrm{sim}_{\mathrm{local}_i}(q,o)$, is defined as the similarity value between a query image q and an image o calculated by the similarity measure of an image database db_i.

Since databases on the Web are heterogeneous, their feature extracting methods and distance functions may be different and so do the similarity measures, although their attributes used in the similarity search are the same. Therefore the local similarity value between a query image and an image object is different from the global similarity value between them. Following three examples illustrate this:

Example 1. The metaserver and the image database support the similarity search using the color attribute. The metaserver extracts average color features from the color histogram in HSV color space while the image database extracts them in RGB color space. The metaserver measures its similarity value against a query image as the image database does. Fig. 2(a) shows the scatter diagram of global similarity values (y-coordinate) and local similarity values (x-coordinate) for 4716 pairs of images selected from the set of 4716 images. Each of 4716 images is selected as the first element of a pair, and the second element of the pair is selected arbitrarily among 4716 images. In this case, the diagram shows that the shape of a graph is a straight line. Fig. 2(b) shows the diagram for the case that the image database extracts average color features from the color histogram in YCbCr color space and measures its similarity value as the metaserver does.

Example 2. The metaserver and the image database support the similarity search using the texture attribute. The metaserver extracts texture features from second moment of the color histogram in HSV color space while the image database extracts them in RGB color space. The metaserver measures its similarity value against a query image as the image database does. The scatter diagram of the global similarity values (y-coordinate) and the local similarity values (x-coordinate) for

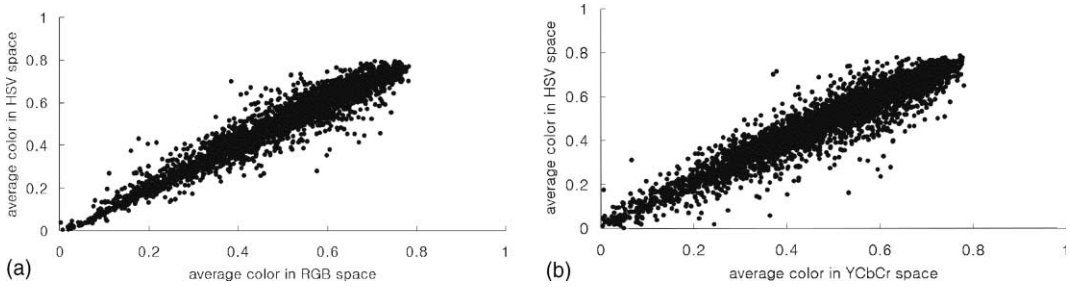


Fig. 2. Scatter-diagram of similarity values for different color features. Scatter-diagram of (a) average color in RGB space vs. average color in HSV space and (b) average color in YCbCr space vs. average color in HSV space.

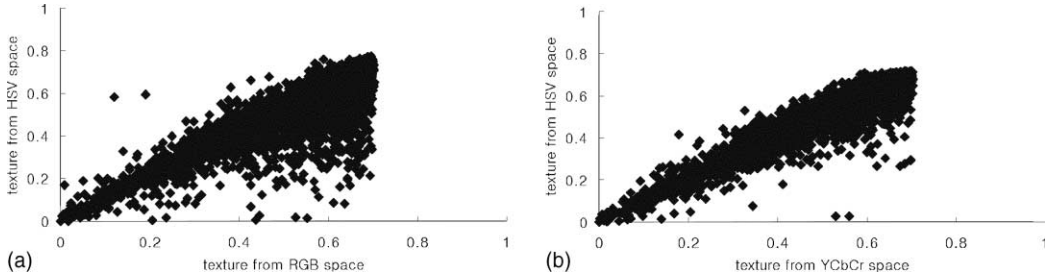


Fig. 3. Scatter-diagram of similarity values for different texture features. Scatter-diagram of (a) texture in RGB space vs. texture in HSV space and (b) texture in YCbCr space vs. texture in HSV space.

4716 images is shown in Fig. 3(a). The diagram shows the shape of a straight line. Fig. 3(b) shows the diagram for the case that the image database extracts texture features from second moment of the color histogram in YCbCr color space and measures its similarity value as the metaserver does.

Example 3. In Fig. 4(a), the similarity values of the y-coordinate are obtained using the average color from the color histogram in HSV color space while those of the x-coordinate are obtained using the texture extracted from second moment of the color histogram in RGB color space. Contrary to the previous cases, the scatter diagram does not show any relationship between two similarity measures with different attributes. Fig. 4(b) shows the diagram for the case that the image database extracts texture features from second moment of the color histogram in YCbCr color space and measures its similarity value as the metaserver does.

Observation 1. Although similarity measures are different between the metaserver and image databases, the scatter diagrams of similarity values of some pairs of similarity measures show the shape of a straight line.

Since we cannot prove that two different similarity measures with the same attribute shows the linear relationship, we have made extensive experiments that show various cases that satisfy the linear relationship.

The statistical linear regression method is used to obtain the equation of a straight line and the test of statistical hypothesis is used to verify the linear rela-

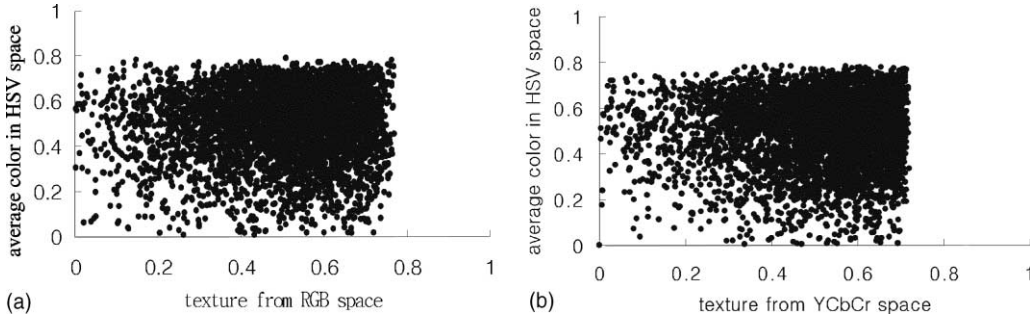


Fig. 4. Scatter-diagram of similarity values for color and texture features. Scatter-diagram of (a) average color in HSV space vs. texture in RGB space and (b) average color in HSV space vs. texture in YCbCr space.

Table 1
The description of features to be used in similarity measures

Feature name		Feature description
Color feature	Feat1	Average color feature from histogram in RGB color space and the Euclidean distance function is used.
	Feat2	Average color feature from histogram in HSV color space and the angular distance function is used.
	Feat3	Average color feature from histogram in YCbCr color space and the Euclidean distance function is used.
Texture feature	Feat4	Texture feature from histogram in RGB color space and the Euclidean distance function is used.
	Feat5	Texture feature from histogram in HSV color space and the angular distance function is used.
	Feat6	Texture feature from histogram in YCbCr color space and the Euclidean distance function is used.

tionship between two similarity measures. As test indicators, we used the scatter diagram, the sample coefficient of determination (r^2), and the analysis of variance (F_0, $F(\alpha)$) where r^2 is given by (sum of squares due to linear regression)/(total variance), F_0 is given by (mean square due to linear regression)/(mean square of residual), $F(\alpha)$ is obtained from F-distribution for a level of significance α. If the linear regression model is effective for two similarity measures, the scatter diagram must show the shape of a straight line, r^2 ($0 \leqslant r^2 \leqslant 1$) must be near to 1 and F_0 must be larger than $F(\alpha)$ (Hillier and Lieberman, 1977; Park, 1985). Table 1 describes features and distance functions to be used in these experiments. There are two groups of features: feat1–feat3 are average color features, feat4–feat6 are texture features.

Table 2 shows the result of experiments for two similarity measures when similarity values are measured for pairs of images. We made two kinds of pairs of features: one is that two features are chosen from the same group, the other is that two features are chosen from different groups.

Since the similarity search is performed for a given query objects, we experimented for the case in which similarity values are measured between a fixed query image and arbitrary images. Table 3 shows the comparison of the case in which similarity values are measured for pairs of images and the case in which similarity values are measured for a fixed query image and arbitrary images. In Table 3, ρ is a correlation of the distribution and β_0, β_1 are the statistical values used in the linear regression line $y = \beta_0 + \beta_1 x$. As shown in Table 2, in the case of similarity measures from the same group, the scatter diagram shows the shape of a straight line, r^2 value is near to 1, and F_0 is much larger than $F(\alpha)$.

However, in the case of similarity measures from different groups, the scatter diagram does not show the shape of a straight line and r^2 value is near to 0. And F_0 in this case is much smaller than F_0 of the case that the linear relationship is satisfied even though F_0 is larger than $F(\alpha)$. In this case, we can say that two similarity measures do not satisfy the linear relationship. As shown in Table 3, we can observe the fact that there are no severe difference between the case that similarity values are measured for arbitrary pairs of images and the case that similarity values are measured for a fixed given query image and arbitrary images.

For any two similarity measures, if they satisfy the linear relationship, we can use that property for the distributed similarity search.

4. Database selection

In this section, we present our proposed hybrid estimation method for ranking databases and choosing candidate databases to submit a query q. Table 4 summarizes the notation used in our database selection approach.

4.1. Hybrid estimator

We are interested in identifying "good candidate" databases, which produce large result sizes with respect to a query. Therefore, we focus on estimating result sizes of the query from image databases that use heterogeneous similarity measures. Let I_i be the set of images in the ith image database.

Table 2
Test of statistical hypothesis for linear regression between two similarity measures using different features and distance functions

Features	Scatter diagram	Correlation, ρ	r^2	F_0	F (0.05)	Result
Feat1:Feat2	Straight line	0.9711	0.943	78 046	0.000	Linear
Feat1:Feat3	Straight line	0.9814	0.963	123 486	0.000	Linear
Feat2:Feat3	Straight line	0.9498	0.902	43 504	0.000	Linear
Feat1:Feat4	Scattered	0.0623	0.004	18.346	0.000	Non-linear
Feat2:Feat5	Scattered	0.0115	0.001	0.629	0.428	Non-linear
Feat4:Feat5	Straight line	0.8830	0.780	16 680	0.000	Linear
Feat4:Feat6	Straight line	0.9153	0.838	24 331	0.000	Linear
Feat5:Feat6	Straight line	0.9544	0.911	48 239	0.000	Linear

Table 3
Comparison of the case in which the similarity is measured for pairs of images and the case in which the similarity is measured between a fixed query image and an arbitrary one

	ρ	β_0	β_1	r^2	F_0	F (0.05)
Feat1:Feat2						
Pairs of images	0.971	0.0003	1.007	0.943	78 046	0.000
Fixed query1	0.969	−0.037	1.069	0.939	72 015	0.000
Fixed query2	0.991	−0.020	0.987	0.982	256 073	0.000
Fixed query3	0.971	−0.029	1.061	0.942	77 141	0.000
Fixed query4	0.975	0.010	1.001	0.950	90 441	0.000
Fixed query5	0.913	0.056	0.870	0.833	23 537	0.000
Feat2:Feat3						
Pairs of images	0.950	0.242	0.949	0.902	43 504	0.000
Fixed query1	0.912	0.083	0.877	0.832	23 281	0.000
Fixed query2	0.972	0.065	0.882	0.945	80 463	0.000
Fixed query3	0.971	−0.029	1.061	0.942	77 141	0.000
Fixed query4	0.948	0.025	0.986	0.899	41 970	0.000
Fixed query5	0.934	0.046	0.950	0.872	32 231	0.000
Feat4:Feat5						
Pairs of images	0.883	0.036	0.930	0.780	16 680	0.000
Fixed query1	0.826	0.122	0.695	0.682	10 131	0.000
Fixed query2	0.963	0.044	0.999	0.928	60 417	0.000
Fixed query3	0.856	0.072	0.900	0.732	12 869	0.000
Fixed query4	0.862	0.034	0.985	0.743	13 612	0.000
Fixed query5	0.896	0.069	0.912	0.802	19 129	0.000

Table 4
Notations used in database selection

Symbol	Meaning
DB	A set of all image databases $= \{db_1, \ldots, db_s\}$
db_i	The ith image database
q	Query
M	Number of databases to be selected
GT	Global threshold that a user specifies
LT_i	Local threshold of the ith image database corresponding to GT
n	Number of random sample objects
y	Global similarity value
x	Local similarity value

Definition 4. The global query result size of the ith image database for q and GT, $\mathrm{gnum}(db_i, q, GT)$, is defined as follows:

$$\mathrm{gnum}(db_i, q, GT) = |\{o \in I_i | \mathrm{sim}_{\mathrm{global}}(q, o) \geqslant GT\}| \quad (3)$$

Definition 5. The local query result size of the ith image database for q and LT_i, $\mathrm{lnum}(db_i, q, LT_i)$, is defined as follows:

$$\mathrm{lnum}(db_i, q, LT_i) = |\{o \in I_i | \mathrm{sim}_{\mathrm{local}_i}(q, o) \geqslant LT\}| \quad (4)$$

Definition 6. The global query selectivity of the ith image database for q and GT, $\mathrm{gsel}(db_i, q, GT)$ is defined as follows:

$$\mathrm{gsel}(db_i, q, GT) = \mathrm{gnum}(db_i, q, GT)/|I_i| \quad (5)$$

where $|I_i|$ is the number of objects in database db_i.

Definition 7. The local query selectivity of the ith image database for q and LT_i, $\mathrm{lsel}(db_i, q, LT_i)$, is defined as follows:

$$\mathrm{lsel}(db_i, q, LT_i) = \mathrm{lnum}(db_i, q, LT_i)/|I_i| \quad (6)$$

The ideal rank of databases is then determined by sorting databases according to their $\mathrm{gnum}(db_i, q, GT)$ for a query q. But the computation of $\mathrm{gnum}(db_i, q, GT)$ may not be possible because an image database does not provide a histogram for global features but for local features. Therefore we need to compensate for the difference between $\mathrm{gnum}(db_i, q, GT)$ and $\mathrm{lnum}(db_i, q, LT_i)$. To do that, we introduce two selectivity compensations as follows:

Definition 8. Population selectivity compensation (PSC_i) of the ith image database

$$PSC_i(db_i, q, GT) = \mathrm{gsel}(db_i, q, GT) - \mathrm{lsel}(db_i, q, LT_i) \quad (7)$$

However, we cannot compute an exact value of PSC_i. Instead, we define the Sample Selectivity Compensation to estimate PSC_i.

Definition 9. Sample selectivity compensation ($SSC_{i,n}$) of the ith image database.

Let $\mathrm{gsel}_{\mathrm{sample},n}(db_i, q, GT)$ denote a global query selectivity estimated using random samples of size n and

$\mathrm{lsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, \mathrm{LT}_i)$ denote a local query selectivity estimated using random samples of size n. GT and LT_i holds a linear relation of $\mathrm{GT} = \widehat{\alpha}_i + \widehat{\beta}_i \times \mathrm{LT}_i$ which is obtained using the linear regression analysis for the local similarity and the global similarity of sample objects. $\mathrm{SSC}_{i,n}$ is defined as an offset value to compensate for the difference between $\mathrm{gsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, \mathrm{GT})$ and $\mathrm{lsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, \mathrm{LT}_i)$ of a query q.

$$\mathrm{SSC}_{i,n}(\mathrm{db}_i, q, \mathrm{GT}) = \mathrm{gsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, \mathrm{GT}) - \mathrm{lsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, \mathrm{LT}_i) \tag{8}$$

Let $\mathrm{lsel}'(\mathrm{db}_i, q, \mathrm{LT}_i)$ be the local query selectivity estimated using the histogram information of the image database db_i. Then, the estimated global query selectivity of db_i, gsel', for q can be obtained from the hybrid estimator as follows:

$$\mathrm{gsel}'(\mathrm{db}_i, q, \mathrm{GT}) = \mathrm{lsel}'(\mathrm{db}_i, q, \mathrm{LT}_i) + \mathrm{SSC}_n(\mathrm{db}_i, q, \mathrm{GT}) \tag{9}$$

Likewise, the estimated global query result size of $\mathrm{db}_i, \mathrm{gnum}'$, for q as follows:

$$\mathrm{gnum}'(\mathrm{db}_i, q, \mathrm{GT}) = \mathrm{gsel}'(\mathrm{db}_i, q, \mathrm{GT}) \times |I_i| \tag{10}$$

$\mathrm{gsel}'(\mathrm{db}_i, q, \mathrm{GT})$ and $\mathrm{gnum}'(\mathrm{db}_i, q, \mathrm{GT})$ are used as approximations of $\mathrm{gsel}(\mathrm{db}_i, q, \mathrm{GT})$ and $\mathrm{gnum}(\mathrm{db}_i, q, \mathrm{GT})$, respectively.

4.2. Database histogram information

In order to select image databases relevant to a given query, we need histograms to estimate the result sizes of the query for image databases. For image databases, the multi-dimensional histograms are constructed from the feature vectors of image objects. Feature values of an object o in image database db_i are values in a real data space $[-\infty, \infty]$. But the histogram information as the content summary of image database db_i is generated in the normalized data space $[0, 1]^p$ with dimension p. Therefore, in order to construct the histogram information, the feature data in each database should be normalized as follows:

For an object o, let $F = \{f_1, f_2, \ldots, f_k, \ldots, f_p\}$ be the corresponding p dimensional feature vector, where f_k is the k-th feature value in feature vector F. If there are N objects in database db_i, $N \times p$ feature matrix $\Im$ can be formed. Each column $\Im_k$ of $\Im$ is a set of k-th feature values of length N. Assuming that the set $\Im_k$ has a normal distribution, we can compute the mean m_k and the standard deviation σ_k of the set $\Im_k$. Then we perform the *intra-feature normalization* (Ortega et al., 1998) for feature vectors with the dimension p. It is formulated as follows:

$$f'_k = \frac{\frac{f_k - m_k}{3\sigma_k} + 1}{2} \tag{11}$$

According to the above formula, the probability that a feature value falls in the range [0, 1] is approximately 99 percent. The parameters $m_1, m_2, \ldots, m_k, \ldots, m_p$ and $\sigma_1, \sigma_2, \ldots, \sigma_k, \ldots, \sigma_p$ are stored as metadata. They can be used to transform points in the real data space into corresponding points in the normalized data space. In order to construct the histogram information of HSV color features, we transform features in the polar coordinate system into those in the rectangular coordinate system.

Then the normalized data space is partitioned into several rectangular buckets and the frequency values associated with the buckets are compressed using DCT technique. The DCT coefficients are referred to as histogram information.

It is important to keep the up-to-date information of image databases. When the number of data updates reaches a certain threshold, histograms should be reconstructed entirely since their accuracy become low. In contrast, our database histogram method can reflect dynamic data updates with reasonable overhead since the linearity property of DCT enables this by processing only the update data (Lee et al., 1999). When the number of inserted objects reaches a certain threshold, the values of DCT coefficients for the inserted objects only are computed, transferred to the metaserver and then added into the existing DCT coefficients of the database histogram.

4.3. Selectivity estimation using compressed histogram information

Lee et al. (1999) proposed the multi-dimensional selectivity estimation method for a rectangular range query. The method is briefly described below. Let q_p be a p-dimensional rectangular range query. We assume the data space is normalized as $(0, 1)^p$. The range of the query q_p is $a_i \leqslant x_i \leqslant b_i$ for $1 \leqslant i \leqslant p$ and x_i coordinate is divided into N_i partitions.

Lemma 1. *Let f be a function which is represented using the inverse DCT function. The selectivity of the p-dimensional query whose range is represented as the hyper-rectangle is expressed as follows*:

$$\text{Selectivity of a query } q_p = \int_{a_p}^{b_p} \cdots \int_{a_2}^{b_2} \int_{a_1}^{b_1} f(x_1, x_2, \ldots, x_p)\, \mathrm{d}x_1\, \mathrm{d}x_2 \cdots \mathrm{d}x_p \tag{12}$$

$$= \sqrt{\frac{2}{N_1}} \cdots \sqrt{\frac{2}{N_p}} \sum_{g(u_1,\ldots,u_p) \in Z} k_{u_1} \cdots k_{u_p} g(u_1, \ldots, u_p) \times \int_{a_1}^{b_1} \cos(u_1 \pi x_1)\, \mathrm{d}x_1 \cdots \int_{a_p}^{b_p} \cos(u_p \pi x_p)\, \mathrm{d}x_p \tag{13}$$

where Z is the set of selected coefficients from zonal sampling, and $g(u_1,\ldots,u_p)$ is a p-dimensional DCT coefficient, and

$$k_{u_i} = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u_i = 0 \quad u_i = 0,\ldots,N_i-1 \\ 1 & \text{for } u_i \neq 0 \end{cases}$$

4.4. Selectivity estimation for the spherical similarity query

In image databases, the similarity query can be represented as a hyper-sphere in the real data space. Since the method of Section 4.3 can estimate only the selectivity of a rectangular query, the hyper-sphere of a query q should be approximated into the hyper-rectangle(s) in the real data space. In this section, we describe two selectivity estimation techniques, based on the histogram information using DCT, to estimate the result size of a spherical similarity query as follows:

Single rectangle approximation (SRA): The selectivity of a query q can be obtained from the selectivity of the hyper-rectangle which has the same volume and center as the hyper-sphere of query q. An approximated hyper-rectangle is plotted in Fig. 5, i.e., the dotted rectangle.

Multiple rectangle approximation (MRA): It generates v hyper-rectangles, R_i, $i = 1,\ldots,v$, within the hyper-sphere whose center is $(0,\ldots,0)$ and radius 1 in advance. To generate hyper-rectangles efficiently, we give the condition that an overlapped region is made from at most two hyper-rectangles. an overlapped area is regarded as another hyper-rectangle. Let s_i be the selectivity of hyper-rectangle R_i, λ_i =(the volume of the common region of R_i and the hyper-sphere)/(the volume of R_i) and ρ =(the volume of the hyper-sphere)/(the total volume of inner regions of R_i). We adjust λ_i value of a new rectangle to be more than a predefined value τ. In our experiment, we empirically choose τ to be 0.7 in three dimension or 0.5 in six dimension since the MRA technique performs well with these ratios. These hyper-rectangles are scaled and transformed properly to locate them within the query range and the selectivity of each hyper-rectangle, s_i, ρ and λ_i is computed. Then the selectivity of q is $\rho\sum_{i=1}^{v} s_i\lambda_i$.

In both cases, the volume of the hyper-sphere or the volume of the common region of the hyper-rectangle and the hyper-sphere should be calculated. However, it is difficult to compute analytically the volume of the hyper-sphere in a real data space since the boundary effect (Berchtold et al., 1997) may occur. In this case, we can calculate the volume approximately by applying the Monte-Carlo method (Kalos and WhitRock, 1986) as follows: the volume of the hyper-sphere = the volume of the hyper-rectangle circumscribing the hyper-sphere × (the number of random points within the hyper-sphere/ the total number of random points in the hyper-rectangle).

The histogram information for an image databases is generated in the normalized data space $[0,1]^p$ where p is the dimension of the feature vector of an image database. As shown in Fig. 5, therefore, the hyper-sphere and the hyper-rectangle in the real data space must be transformed into the hyper-oval and the hyper-rectangle, respectively, in the normalized data space by using Equation (11). Then the selectivity of the spherical similarity query can be estimated using Equation (13).

4.5. Sample selectivity compensation

Different similarity measures cause the difference Diff_i between $\mathrm{gnum}(\mathrm{db}_i, q, \mathrm{GT})$ and $\mathrm{lnum}(\mathrm{db}_i, q, \mathrm{LT}_i)$. In order to estimate Diff_i, $\mathrm{PSC}_i(\mathrm{db}_i, q, \mathrm{GT})$ in Equation (7) can be used. But PSC_i cannot be computed at the metaserver. Instead, we fetch sample objects from image databases to the metaserver, extract their features, and compute $\mathrm{SSC}_{i,n}$. To do it, we use the *progressive query-based sampling* method in the preprocessing phase. It is a method to acquire samples from an image database across the Internet without special cooperation between the metaserver and the image database. We assume that a metaserver can issue range queries to each database to retrieve images. Although random sampling is not possible, it can be approximated with queries carefully selected either from SAMP or QP, where SAMP is a set of collected objects obtained by using the progressive query-based sampling and QP is a temporary query pool with randomly selected objects either from a metaserver

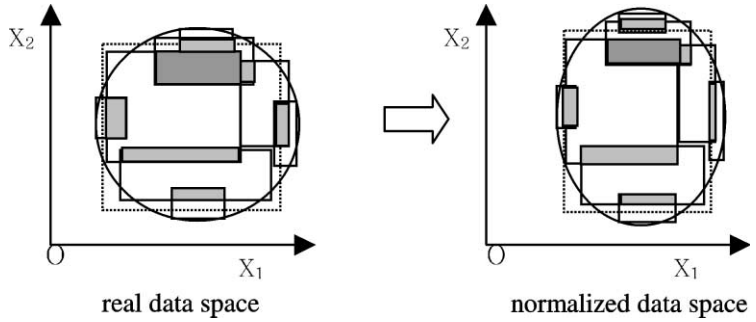


Fig. 5. Real data space and normalized data space.

or other image databases ($|QP| > 30$). The objects in QP are selected such that they are uniformly distributed in the feature space. A detailed sampling algorithm for an image database is as follows:

(1) Select an initial query object randomly from QP; SAMP $\leftarrow \phi$; $n \leftarrow 0$.
(2) Submit the query object to the ith image database.
(3) Retrieve objects from the ith image database using a range query, and select randomly δ samples among retrieved objects such that δ is 10% of retrieved objects, and then add them to SAMP.
(4) Replace QP with SAMP if the number of sample objects in SAMP is sufficiently large.
(5) Calculate a stopping criterion value based on the characteristics of the retrieved objects.
(6) If a stopping criterion has not been reached yet,
   (a) Select a new query object from QP; $n \leftarrow n + \delta$; and
   (b) Go to Step (2).

The algorithm involves two important issues such as how to select query objects (from Step (1) to Step (4)), and when to stop collecting samples from a database (from Step (5) to Step (6)). The distribution of the population should be learned and query objects should be selected carefully in order to provide random samples of images.

In Step (1), a query object is randomly selected from QP since QP has a uniform distribution, that is, the objects in QP can evenly cover the data space of an image database. In Step (3), the metaserver retrieves objects from the ith image database using a range query, and selects randomly 10% samples among the retrieved objects, and then adds them to SAMP. Then δ is in proportion to the density of the given query region. A range query retrieves many objects from the dense area while it retrieves a few objects from the sparse area of the population. A sample set SAMP learns progressively the distribution of the whole data set according to retrieved objects by repeating a range query using a randomly selected query object from QP. In Step (4), the query object is randomly selected from QP until the size of the sample set SAMP becomes sufficiently large (for example, $|SAMP| > 30$). Initially, samples in SAMP would be biased strongly in the feature space since they are collected from a few query objects. A solution is to select query objects from a uniform temporary query pool QP in order to get a more random set of query objects. When the size of the sample set SAMP becomes sufficiently large, a random query object can be selected from SAMP since SAMP has approximately learned the data distribution of an image database.

In Steps (5) and (6), a stopping criterion is evaluated to decide when the learned sample set SAMP is sufficiently accurate and the improving rate of the accuracy of $SSC_{i,n}$ becomes very small. As for the accuracy of SAMP, the whole data distribution can be estimated from the compressed histogram information of each database. For the computational efficiency, we assume that dense areas in the data space can be found by inspecting the frequency values of all buckets in the compressed histogram information of the ith database. We use the range of the bucket whose frequency value is close to the average of histogram frequency values as a target range for estimating the accuracy of SAMP. Let the sample size be n, the database size be N, the sample frequency of the target range be n_t, and the histogram frequency of the same range be N_t, respectively. If the sample frequency ratio n_t/n of the target range is sufficiently similar to the histogram frequency ratio N_t/N of the same range, it is reasonable to conclude that the learned sample set is not biased.

On the other hand, we have to show that $SSC_{i,n}$ converges to PSC_i to show the accuracy of the estimation method using $SSC_{i,n}$.

In order to compute $SSC_{i,n}$ of a query using sample objects of size n from the ith image database db_i, the statistical metadata such as sample regression line coefficients $(\hat{\alpha}, \hat{\beta})$, and the sample coefficient of determination(r^2) are used. As shown in Fig. 6, we can estimate the regression line, $E(y|x) = \hat{\alpha} + \hat{\beta}x$, from the bivariate distribution of the local similarity value x and the global similarity value y of a given query using sample objects.

Let the points $(x_i, E(y|x_i))$, $i = 1, \ldots, n$ on the estimated regression line be *threshold points*. The following lemmas and theorem show that $SSC_{i,n}$ converges to a constant value as the sample size n increases.

Lemma 2. *The local query selectivity estimated using random samples of size n with the local similarity threshold x converges stochastically to the local query selectivity estimated using all objects in db_i as the sample size n increases.*

Lemma 3. *The global query selectivity estimated using random samples of size n with the global similarity threshold $\hat{y}(= E(y|x))$ converges stochastically to the global query selectivity estimated using all objects in db_i as the sample size n increases.*

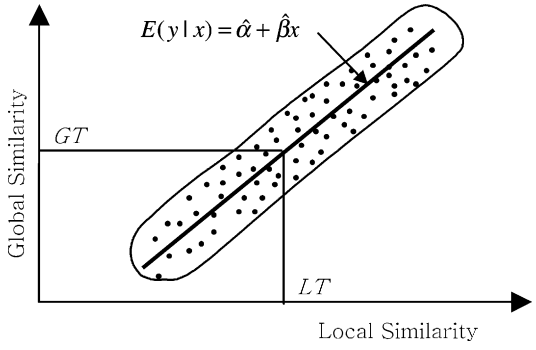


Fig. 6. Regression line.

Theorem 1. *For a given query*, $\mathrm{SSC}_{i,n}(\mathrm{db}_i, q, E(y|x))$ *with threshold point* $(x, E(y|x))$ *converges stochastically to* PSC_i *as the sample size n increases.*

The proofs of the above lemmas and theorem can be found in Appendix A. Now, let us explain how to determine the size n of sample objects. Since $\mathrm{SSC}_{i,n}$ converges to PSC_i as n increases, both $\mathrm{SSC}_{i,n}$ and $\mathrm{SSC}_{i,n+\delta}$ are close to PSC_i when $\mathrm{SSC}_{i,n}$ is sufficiently accurate. Actually, we choose n when $|\mathrm{SSC}_{i,n} - \mathrm{SSC}_{i,n+\delta}|/|\mathrm{SSC}_{i,n}| \leqslant 0.1$. As the sample coefficient of determination r^2 becomes high enough, we can observe the following:

Observation 2. If r^2 $(0 \leqslant r^2 \leqslant 1)$ is the sample coefficient of determination in the bivariate distribution of the global similarity value and the local similarity value with respect to a query, Fig. 7 illustrates that $\mathrm{SSC}_{i,n}$ decreases as r^2 increases. It means that $\mathrm{lnum}(\mathrm{db}_i, q, \mathrm{LT})$ is getting close to $\mathrm{gnum}(\mathrm{db}_i, q, \mathrm{GT})$.

4.6. Database ranking algorithm

Our proposed database selection criteria is as follows:

Select top M databases in a ranked list $G(q, \mathrm{GT}) = (\mathrm{db}_{g_1}, \mathrm{db}_{g_2}, \ldots, \mathrm{db}_{g_S})$ where $G(q, \mathrm{GT})$ is ranked according to their estimated global query result size $\mathrm{gnum}'(\mathrm{db}_i, q, \mathrm{GT})$ for a qiven query q.

Algorithm Database_Ranking (q,GT,DB)
(1) for each $\mathrm{db}_i \in DB$, $i = 1, \ldots, S$;
(2) get $\hat{y} = \hat{\alpha} + \hat{\beta}x$ using the linear regression analysis;
(3) calculate $\mathrm{SSC}_{i,n}(\mathrm{db}_i, q, \mathrm{GT})$ using the sample objects of the size n;
(4) $\mathrm{LT}_i = (GT - \hat{\alpha})/\hat{\beta}$;
(5) compute the estimated local query selectivity, $\mathrm{lsel}'(\mathrm{db}_i, q, LT)$, using the histogram information;
(6) $\mathrm{gsel}'(\mathrm{db}_i, q, \mathrm{GT}) = \mathrm{lsel}'(\mathrm{db}_i, q, \mathrm{LT}_i) + \mathrm{SSC}_{i,n}(\mathrm{db}_i, q, \mathrm{GT})$;
(7) $\mathrm{gnum}'(\mathrm{db}_i, q, \mathrm{GT}) = \mathrm{gsel}'(\mathrm{db}_i, q, \mathrm{GT}) \times |I_i|$;
(8) end for
(9) Rank databases according to $\mathrm{gnum}'(\mathrm{db}_i, q, \mathrm{GT})$.

In Steps (2) and (3), the run time of the regression analysis and calculating $\mathrm{SSC}_{i,n}(\mathrm{db}_i, q, \mathrm{GT})$ will be proportional to the number of sample objects, n, for each database. Let p be the dimension, α be the computation time of the sin function, and z be the number of DCT coefficients. v rectangles are used for MRA. Since the selectivity computation time of the rectangular range query is obtained as $2p\alpha z$ from Eq. (13), the run time complexity of computing $\mathrm{lsel}'(\mathrm{db}_i, q, \mathrm{LT}_i)$, in Step (5), is given by $\mathrm{O}(v2p\alpha z)$ when MRA is used or $\mathrm{O}(2p\alpha z)$ when SRA is used. Then the complexity of the algorithm is $\mathrm{O}(S(n + v2p\alpha z))$ for a set of S databases when n sample objects and MRA are used.

4.7. Pure sampling-based method

Our proposed database selection method is regarded as the method using an hybrid estimator, since it uses the histogram information together with a small number of sample objects.

As an alternative to estimate the result size of a query according to the global similarity measure, the pure sampling-based estimator can be used. It ranks databases by using only $\mathrm{gsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, \mathrm{GT})$. It provides a reasonable size estimation for any data distributions and its accuracy for the size estimation becomes high in proportion to the amount of samples taken. There is a trade-off between the estimation accuracy and the amount of samples.

Fig. 8 depicts the relationship between the sample size and the accuracy of the size estimation. The horizontal axis represents the number of samples. N is the total number of available objects in an image database. The vertical axis represents the accuracy of the size estimation when a sample set of size n is given.

The pure sampling-based method incurs the run-time cost for sampling since the random sampling occurs in the query processing time, and requires more storage of the metadatabase since it needs a sufficient size of samples so as to achieve high accuracy of the size esti-

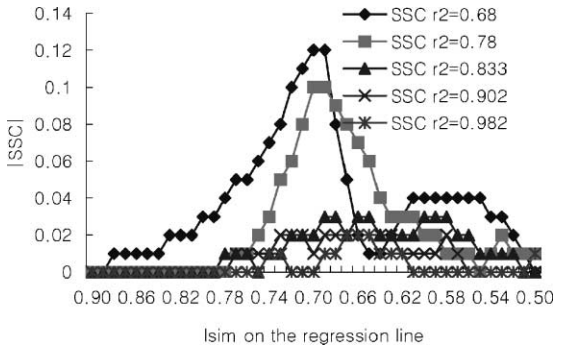


Fig. 7. $\mathrm{SSC}_{i,n}$ on threshold points.

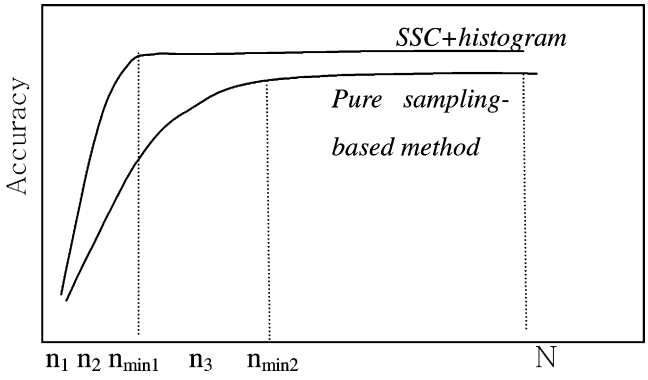


Fig. 8. Learning curves and progressive samples.

mation. Even though the proposed method uses a small size of samples($n_{\min 1}$), it can achieve higher accuracy than the pure sampling-based method. We use the progressive query-based sampling in the preprocessing phase since the sampling through Internet incurs high cost. The experimental results are shown in Fig. 11 of Section 5.2.

5. Experiments

In order to measure the effectiveness of our proposed method, we have conducted comprehensive experiments over a large number of image data and various queries. Our experiments focus on showing the accuracy of the selection method to rank image databases with respect to a given query. The system is written in Microsoft VC++ under Windows NT, on the HP NetServer.

5.1. Experimental framework

The test data consists of 83 476 images with 256-color bitmaps. We constructed 10 image databases based on a semantic categorization since visual databases usually contain particular types of images. Each database uses a different feature extraction method and a different distance function. To acquire visual features that characterize images, we extracted the average color and texture from various color spaces using a color histogram method. For each image, the average color (μ_1, μ_2, μ_3) is used to represent the average intensity of each color component and the texture (σ_1, σ_2, σ_3) is used to represent the relative smoothness of each color component. In our experiments, we use the average color and texture in HSV color space as the features of the metaserver. Table 5 shows feature extraction methods, database sizes, and semantic categories for all image databases. The experiments for the database selection are conducted by varying the parameters in Table 6.

Table 5
The test data set environment

Sites	Feature extraction methods	Size	Semantic categories
1	Average color/texture in RGB color space	8888	Scene, photo
2	Average color/texture in RGB color space	8819	Animal, zoo collection
3	Average color/texture in HSV color space	8526	Art and design
4	Average color/texture in RGB color space	8328	Background, pattern
5	Average color/texture in YCbCr color space	7940	Flower, plant
6	Average color/texture in YCbCr color space	7835	Clip art
7	Average color/texture in YCbCr color space	8328	Background, pattern
8	Average color/texture in HSV color space	9908	People
9	Average color/texture in HSV color space	8328	Background, pattern
10	Average color/texture in YCbCr color space	8819	Animal, zoo collection

Table 6
Test parameters for database selection

Parameter	Meaning
Dimension D	Dimension of image features ($D = 3$, 6 are used)
SR	Number of sample objects divided by the total number of objects in an image database (SR = 0.1–4.4%)
M	Number of databases to be selected ($M = 1, 2, \ldots, 10$)
Threshold T	Range of GT values ($T = 0.7$, 0.6, 0.5 are used)

5.2. Experimental results

Our experiments include the comparison of two rectangle approximation techniques used for the hybrid estimator: SRA and MRA. We have executed 30 queries for each test using various parameters and averaged their results. When the linear regression is performed, 99.9% confidence level is used to estimate the prediction interval of the global similarity y. For MRA, we generate 13 small hyper-rectangles to approximate a normal hyper-sphere in advance. We first observe an appropriate number of sample objects, which provides a sufficiently accurate global selectivity estimation. To measure the accuracy, the relative error (E) is defined as follows:

$$E = \frac{|\text{query result size} - \text{estimated result size}|}{\text{query result size}} \times 100\%$$

Fig. 9(a) and (b) shows the relative error of SRA and MRA for $D = 3$ and $D = 6$, respectively. We use the same databases (sites 4, 7, and 9) but different similarity measures (RGB, YCbCr, HSV color spaces). We can observe that the relative error of SRA-YCbCr is 12.5–16.2% for $D = 3$ and 24–26.5% for $D = 6$ and that of MRA-YCbCr is 2.8–6.6% for $D = 3$ and 9.5–14.7% for $D = 6$. The relative error of MRA for the same range of the sample ratio (SR) shows better results than that of SRA for $D = 3$, 6. The relative error of SRA-HSV shows constantly 14.2% and 29.9% in the whole range because the global similarity measure is the same as the local one (HSV color space) and, therefore, SSC = 0 for the whole range. The experiments also illustrate that the relative errors change little as the SR increases above 1.1%. Consequently, we will fix the SR to 1.1% for the next experiments.

Fig. 10(a)–(d) show the results of the sample selectivity compensation($SSC_{i,n}$) for given four queries. We can observe that $SSC_{i,n}$ fluctuates when the SR is very

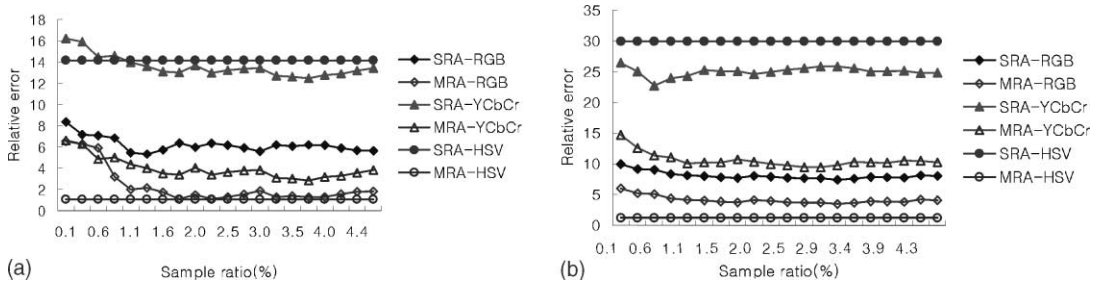


Fig. 9. Relative error (a) for D = 3 and (b) for D = 6.

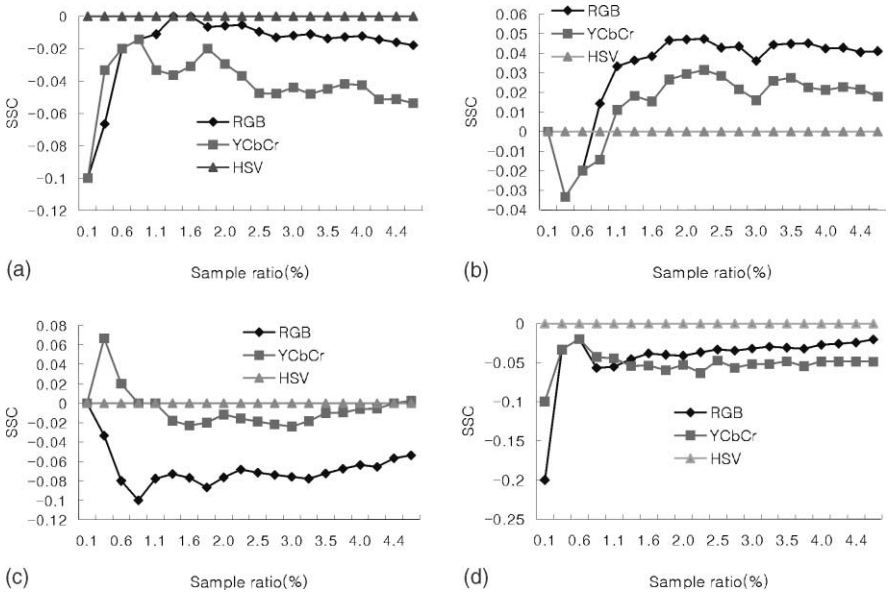


Fig. 10. $SSC_{i,n}$: (a) $SSC_{4,n}$ for query 1, (b) $SSC_{4,n}$ for query 2, (c) $SSC_{4,n}$ for query 3 and (d) $SSC_{4,n}$ for query 4.

small but $SSC_{i,n}$ converges stochastically to a constant value when the SR increases. The result shows that we can significantly offset the difference between the global and local selectivities utilizing $SSC_{i,n}$ when the SR is larger than 1.1%.

We also conducted comprehensive experiments to compare (1) the pure sampling-based method and (2) the histogram-based method and (3) proposed two hybrid methods (SRA + SSC, MRA + SSC) in order to measure the accuracy and efficiency of the proposed method in estimating the result sizes of queries. Fig. 11 shows that,as compared with pure sampling-based method, the accuracy for estimation of the result size of the query can be significantly improved by the hybrid methods even when the sample size is small. The result also shows that the hybrid methods can make a substantial saving in the sample size for the same accuracy.

In order to illustrate that the progressive query-based sampling method provides unbiased samples, we compared the distribution of the whole data set with that of the samples in Fig. 12. The color features of images are represented in the 3 dimensional data space. To visualize

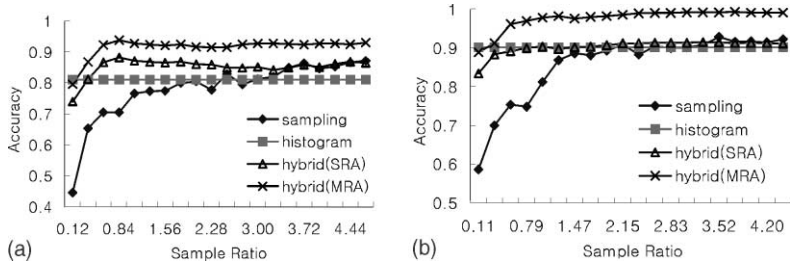


Fig. 11. Accuracy (a) for database 6 and (b) for database 2.

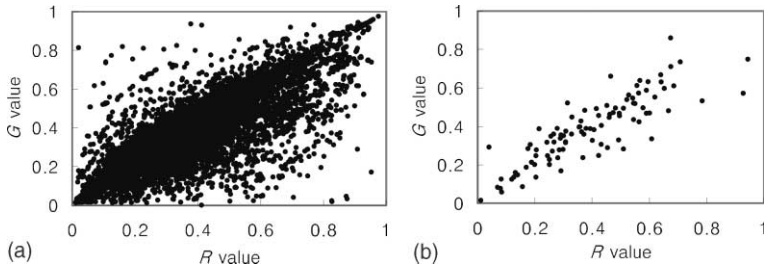


Fig. 12. Scatter-diagram of color features for whole data set and the samples. Scatter-diagram of (a) whole data set and (b) the samples.

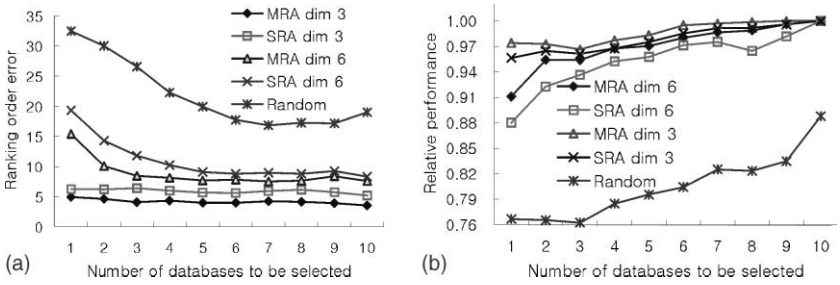


Fig. 13. Performance of methods: (a) ranking order error and (b) relative performance.

them, we use R, G values. Fig. 12(a) shows the scatter diagram of R values (x-coordinate) and G values (y-coordinate) for the whole data set of site 4 (8328 images). Fig. 12(b) shows the diagram for samples that are extracted from the same database using the progressive query-based sampling method. When a SR is 1.1%, the experimental result shows that the samples are not biased.

We examine the performance of the proposed database selection method. Two performance measures are used. First, the ranking order error (E') for test queries is used to compare the ranks returned by the proposed method against the ideal ranks (Callan et al., 1995). The ideal ranks of each database are determined by measuring the global similarity between a given query and each database image. The ranking order error for the test query q_j is calculated as follows: $E'_j = (1/|G|)\sum_{\mathrm{db}_i \in G}(\mathrm{IR}_{i,j} - \mathrm{ER}_{i,j})^2$ where $\mathrm{IR}_{i,j}$ is the ideal rank of the database db_i based on the actual result size for q_j, $\mathrm{ER}_{i,j}$ is the estimated rank of db_i determined by the ranking algorithm for q_j, and G is the set of databases selected by the proposed database selection method. Given k test queries $q_1, q_2, \ldots, q_k$, E' is computed by the expression $E' = (1/k)\sum_{j=1}^{k} E'_j$. Second, the relative performance (P) derives the accuracy of the database selection method by computing the ratio between the result size of the global queries returned by the database selection method and that returned by the ideal choices of databases. Formally, the relative performance P_j contributed by a test query q_j ($1 \leqslant j \leqslant k$) is defined as: $P_j = \sum_{\mathrm{db}_i \in G} \mathrm{gnum}(\mathrm{db}_i, q_j, \mathrm{GT}) / \sum_{\mathrm{db}_i \in B} \mathrm{gnum}(\mathrm{db}_i, q_j, \mathrm{GT})$ where B is the set of databases selected by the ideal choice. Given k test queries $q_1, q_2, \ldots, q_k$, P is computed by the expression $P = (1/k)\sum_{j=1}^{k} P_j$.

Fig. 13(a) and (b) shows the performance of the database selection using two hybrid estimation methods against the number of databases to be selected (M) for $D = 3$ and 6. The performance of a random database selection, which selects arbitrary M databases out of S databases, was used as the baseline reference for comparison. Our proposed techniques always outperform the random database selection and MRA yields better performance than SRA in these figures.

In addition, we examine the accuracy of the database selection by comparing two proposed methods with the optimal one. Fig. 14 shows the result size of the global queries estimated for each database. MRA is better than SRA, when $D = 3$, 6 respectively.

Table 7 shows the volume of the hyper-sphere with center $(0, \ldots, 0)$ and radius 1, the total volume of the hyper-rectangle(s), the total-in-volume (= sum of the volumes of hyper-rectangles within the hyper-sphere) and the total-in-volume ratio(= the total-in-volume/the volume of the hyper-sphere) for SRA, MRA in case of $D = 3$ and 6, respectively. We can observe that MRA shows better performance than SRA since MRA has higher total-in-volume ratio than SRA. But the accuracy of MRA and SRA degrades as the dimension of image features increases. This is the natural result because the total-in-volume ratio between the hyper-rectangles and the hyper-sphere decreases as the dimension increases.

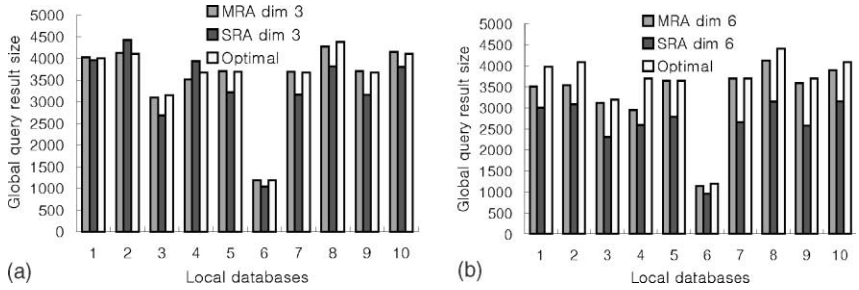


Fig. 14. Global query result size (a) for $D = 3$ and (b) for $D = 6$.

Table 7
Volumes of the hyper-sphere and hyper-squares for SRA and MRA

		Sphere volume	Total rectangle-vol	Total-in-volume	Total-in-volume ratio (%)
SRA	$D = 3$	4.19	4.19	3.53	84.2
	$D = 6$	5.17	5.17	3.49	67.5
MRA	$D = 3$	4.19	5.45	4.08	97.4
	$D = 6$	5.17	7.72	4.26	82.5

5.3. Storage and time requirement of the metadatabase

The size of the metadatabase largely depends on the number of sample objects fetched from image databases and the size of database histogram information. For each database, the size of the statistical metadata ranges from 96 bytes for three dimensional image features to 144 bytes for six dimensional ones since they consist of the mean, the standard deviation, the correlation coefficient of global and local similarity values of sample objects for transforming the global similarity threshold to local one, and the mean, the standard deviation of the feature set for constructing the database histogram information. The storage requirement of the database histogram information depends on the number of DCT coefficients. In our experiments, we use 2012 DCT coefficients in three dimension or 2499 DCT coefficients in six dimension. The size of the database histogram information is 2012×8 bytes (16 kbytes) in three dimension or 2499×8 bytes (20 kbytes) in six dimension.

Using the progressive query-based sampling approach, we collected sample images from each database. The SR for each database is described in Table 8 when the number of retrieved and randomly selected sample objects for each range query, δ, is variable (i.e., δ is 10% of retrieved objects for a range query) and the small range size 0.01 is used. Each sample image is stored as a feature vector in the metadatabase. The size of the feature vector depends on the type of feature extraction method used by the application. In our study, the size of the color or the texture feature vector is 12 bytes in three dimension or 24 bytes in six dimension, and 30 bytes for index information and 42 bytes for header information. When we use the average color in three dimension as the image feature, the size of global and local feature vectors of sample images is at most (42 bytes $\times$ 155 images (1.56%) + 42) $\times$ 2 bytes (13 104 bytes). The total storage requirement of metadatabase with just one registered database ranges from 28.6 kbytes in three dimension to 36.8 kbytes in six dimension. The storage required by the metadatabase is less than 1% of the size of the image database. The more databases are registered to the metaserver, the more sample images are added. The size of the metadatabase will increase as follows:

$$\text{growth factor} = \frac{\text{size(histogram information)} \times |\text{DB}| + \text{size}(\text{fv}_{\text{sample}})}{\text{DBSize}} \tag{14}$$

where DBSize is the storage space for all registered databases and $\text{fv}_{\text{sample}}$ is feature vectors of all sample images and |DB| is the number of image databases.

Table 9 shows the construction costs and the selectivity estimation costs of the database histogram information. It contains actual timings taken for two different sizes of input data(the sample size $S = 4000$ and 8000) and different values for the space (D, the number of DCT coefficients in three and six dimensions) allocated to the histogram information. We performed the cost evaluation on an HP Netserver and averaged the results of five runs. These timings do not include the time taken to extract feature vectors from sample objects and to compute the statistics of them. The construction cost of histogram information for SRA only

Table 8
Sample ratio for each database

	1	2	3	4	5	6	7	8	9	10
DBSize (Mbytes)	94.1	72.7	72.6	66.9	233	580	66.9	175	66.9	72.7
Sample ratio (%)	1.1	0.79	1.6	1.1	0.9	0.84	1.1	1.56	1.1	1.47

Table 9
Construction and evaluation costs of database histogram information

Histogram information	Time taken (ms)					
	Three dimension			Six dimension		
	$S = 4000$, $D = 2012$	$S = 8000$, $D = 2012$	$S = 8000$, $D = 4000$	$S = 4000$, $D = 2499$	$S = 8000$, $D = 2499$	$S = 8000$, $D = 4501$
SRA						
Construction	1970	2580	4830	2080	2690	5050
Selectivity estimation	50	60	60	110	110	110
MRA						
Generation of approximated rectangles		49 164			93 450	
Construction	1970	2580	4830	2080	2690	5050
Selectivity estimation	110	110	110	170	170	170

includes the computing time of DCT coefficients. In addition to computing time of DCT coefficients, we have to consider the generation time of approximated rectangles to evaluate the total construction cost of histogram information for MRA. Even though the generation time is relatively long, the generation is performed only one time in the preprocessing phase. The more the number of DCT coefficients, the larger the construction cost is. The experiment also shows that the number of sample objects has little effect on the time taken for constructing the histogram information. We note that construction costs of histogram information are reasonable and have no difference between SRA and MRA. As can be seen from Table 9, the selectivity estimation costs are much smaller than the construction costs of database histogram information.

6. Conclusions

In this paper, we have investigated the database selection from a large number of image databases on the Web. To solve the problem, we proposed a new hybrid estimation method, which can accurately estimate the number of result objects, globally similar to a given query, from image databases with different similarity measures. It uses the database histogram information which provides the spherical selectivity estimation, and a small number of sample objects which compensate for the selectivity difference between the metaserver and each image database.

The database selection mechanism has been implemented within the metaserver. The metaserver requires low storage to store the summary information of each image database since the hybrid estimator uses a small number of sample objects and the small size of compressed histogram information. As a further advantage, the metaserver can keep the up-to-date information of image databases by using the linearity property of DCT coefficients even though the content of image databases are frequently updated.

We have performed a series of experiments with a large number of real image data and examined the retrieval effectiveness of the proposed method. Our method has shown a sufficient accuracy to select relevant databases with respect to a query. As a future work, we plan to extend our work to the collection fusion of heterogeneous image databases for range queries.

Acknowledgements

This work was supported by Korea Research Foundation Grant (KRF-2000-041-E00262). We would like to thank Dr. Ju-Hong Lee for useful discussions.

Appendix A. Proofs of Lemma 2, Lemma 3, and Theorem 1

In the area of mathematical statistics, if the size of the collection, from which the sample is chosen, is finite but large enough compared to that of the sample, the limit notation can be used as a standard practice (Hogg and Craig, 1978).

Lemma 2. *For every fixed* $\epsilon > 0$, $\lim_{n\to\infty} \mathrm{Prob}\{|\mathrm{lsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, x) - \mathrm{lsel}(\mathrm{db}_i, q, x)| < \epsilon\} = 1$.

Proof. Let o_{ij} be the jth object of db_i. $X_1 = \mathrm{sim}_{\mathrm{local}_i}(q, o_{i1}), X_2 = \mathrm{sim}_{\mathrm{local}_i}(q, o_{i2}), \ldots, X_n = \mathrm{sim}_{\mathrm{local}_i}(q, o_{in})$ are a set of random samples of size n from the cumulative distribution $F(x) = \mathrm{lsel}(\mathrm{db}_i, q, x)$ of a query q. x is a threshold. We define $F_n(x; X_1, X_2, \ldots, X_n)$ as $F_n(x) = \mathrm{lsel}_{\mathrm{sample},n}(\mathrm{db}_i, q, x) = |T|/n$, where $T = \{o_{ij} | \mathrm{sim}_{\mathrm{local}_i}(q, o_{ij}) \geqslant x, j = 1, \ldots, n\}$. Then $F_n(x)$ is a random variable and its distribution is as follows:

$$\mathrm{Prob}\left\{F_n(x) = \frac{j}{n}\right\} = \binom{n}{j}[F(x)]^j[1 - F(x)]^{n-j}, \quad j = 0, 1, \ldots, n$$

The mean and the variance of its distribution is derived as follows:

$E[F_n(x)] = F(x), Var[F_n(x)] = F(x)[1 - F(x)]/n$

From the Chebychev inequality (Hogg and Craig, 1978),

$$\text{Prob}\{|F_n(x) - F(x)| \geqslant \epsilon\} \leqslant \frac{\text{Var}[F_n(x)]}{\epsilon^2} = \frac{[F(x)][1 - F(x)]}{n\epsilon^2}$$

If we take the limit, as n becomes infinite, we have $\lim_{n\to\infty} \text{Prob}\{|F_n(x) - F(x)| \geqslant \epsilon\} = 0$. □

Lemma 3. *For every fixed* $\epsilon > 0$, $\lim_{n\to\infty} Prob\{|\text{gsel}_{\text{sample},n}(\text{db}_i, q, E(y|x)) - \text{gsel}(\text{db}_i, q, E(y|x))| < \epsilon\} = 1$.

Proof. $Y_1 = \text{sim}_{\text{global}}(q, o_{i1}), Y_2 = \text{sim}_{\text{global}}(q, o_{i2}), \cdots, Y_n = \text{sim}_{\text{global}}(q, o_{in})$ are a set of random samples of size n from the cumulative distribution $G(E(y|x)) = \text{gsel}(\text{db}_i, q, E(y|x))$ of a query q, where $E(y|x) = \hat{\alpha} + \hat{\beta}x$. $E(y|x)$ is a threshold. We define $G_n(E(y|x))$ as

$$G_n(E(y|x)) = \text{gsel}_{\text{sample},n}(\text{db}_i, q, E(y|x)) = |Z|/n$$

where $Z = \{o_{ij} | \text{sim}_{\text{global}}(q, o_{ij}) \geqslant E(y|x)\}$. Then we can prove that $G_n(E(y|x))$ converges to $G(E(y|x))$ as in the proof of Lemma 2. □

Theorem 1. *Let* $L = \{(x, E(y|x)) | E(y|x) = \hat{\alpha} + \hat{\beta}x\}$. *For any threshold point* $(x, E(y|x)) \in L$ *and every fixed* $\epsilon > 0, \lim_{n\to\infty} \text{Prob}\{|\text{SSC}_{i,n}(\text{db}_i, q, E(y|x)) - \text{PSC}_i(\text{db}_i, q, E(y|x))| < \epsilon\} = 1$.

Proof. It is given that $\text{SSC}_{i,n}(\text{db}_i, q, E(y|x)) = G_n(E(y|x)) - F_n(x)$ and $\text{PSC}_i(\text{db}_i, q, E(y|x)) = G(E(y|x)) - F(x)$. We are to prove that $\lim_{n\to\infty} \text{Prob} \times \{|\text{SSC}_{i,n}(\text{db}_i, q, E(y|x)) - \text{PSC}_i(\text{db}_i, q, E(y|x))| < \epsilon\} = 1$ for every $\epsilon > 0$. Now

$$\begin{aligned}
&\text{Prob}\{|\text{SSC}_{i,n}(\text{db}_i, q, E(y|x)) - \text{PSC}_i(\text{db}_i, q, E(y|x))| \geqslant \epsilon\} \\
&\quad = \text{Prob}\{|[G_n(E(y|x)) - F_n(x)] \\
&\qquad - [G(E(y|x)) - F(x)]| \geqslant \epsilon\} \\
&\quad = \text{Prob}\{|[G_n(E(y|x)) - G(E(y|x))] - [F_n(x) \\
&\qquad - F(x)]| \geqslant \epsilon\} \leqslant \text{Prob}\{|[G_n(E(y|x)) - G(E(y|x))]| \\
&\qquad + |[F_n(x) - F(x)]| \geqslant \epsilon\} \leqslant \text{Prob}\{|G_n(E(y|x)) \\
&\qquad - G(E(y|x))| \geqslant \epsilon/2\} + \text{Prob}\{|F_n(x) \\
&\qquad - F(x)| \geqslant \epsilon/2\} \leqslant \frac{Var[G_n(E(y|x))]}{(\epsilon/2)^2} + \frac{Var[F_n(x)]}{(\epsilon/2)^2} \\
&\quad = \frac{4[G(E(y|x))][1 - G(E(y|x))]}{n\epsilon^2} + \frac{4F(x)[1 - F(x)]}{n\epsilon^2}
\end{aligned}$$

(by Lemma 2 and 3). If we take the limit, as n becomes infinite, we have

$$\lim_{n\to\infty} \text{Prob}\{|\text{SSC}_{i,n}(\text{db}_i, q, E(y|x)) - \text{PSC}_i(\text{db}_i, q, E(y|x))| \geqslant \epsilon\} = 0.$$ □

References

Bach, J.R. et al., 1996. The virage image search engine: An open framework for image management. SPIE Storage and Retrieval for Still Image and Video Databases IV, 76–87.

Benitez, A.B., Beigi, M., Chang, S.-F., 1998. A content-based image meta-search engine using relevance feedback. IEEE Internet Computing 2 (4), 59–69.

Berchtold, S., Bohm, C., Keim, D.A., Kriegel, H.-P., 1997. A cost model for nearest neighbor search in high-dimensional data space. In: Proceedings of the ACM Symposium on Principles of Database Systems, pp. 78–86.

Callan, J., Connell, M., Du, A., 1999. Automatic discovery of language models for text databases. In: Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 479–490.

Callan, J., Lu, Z., Croft, W., 1995. Searching distributed collection with inference networks. In: Proceedings of the Eighteenth Annual International ACM/SIGIR Conference, pp. 21–28.

Chang, W., Sheikholeslami, G., Wang, J., Zhang, A., 1998. Data resource selection in distributed visual information systems. IEEE Transactions on Knowledge and Data Engineering 10 (6), 926–946.

Crane, R., 1997. Simplified Approach to Image Processing. Prentice-Hall, Englewood Cliffs, NJ.

Flickner, M., Sawhney, H., Niblack, W., et al., 1995. Query by image and video content: The QBIC system. IEEE Computer Magazine 28 (9), 23–32.

Gravano, L., Garcia-Molina, H., Tomasic, A., 1994. The effectiveness of GlOSS for the text database discovery problem. In: Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 126–137.

Gravano, L., Garcia-Molina, H., 1995. Generalizing GlOSS to vector-space databases and broker hierarchies. In: Proceedings of International conference on Very Large Data Bases.

Hafner, J., Sawhney, H.S., Equitz, W., Flickner, M., Niblack, W., 1995. Efficient color histogram indexing for quadratic form distance functions. IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (7), 729–736.

Hillier, F., Lieberman, G., 1977. Introduction to Operations Research. McGraw-Hill, New York.

Hogg, R.V., Craig, A.T., 1978. Introduction to Mathematical Statistics, fourth ed. Collier Macmillan, New York.

Kalos, M.H., WhitRock, P.A., 1986. Monte Carlo Methods. Wiley, NewYork.

Kanai, Y., 1998. Image segmentation using intensity and color information. In: Proceedings of the Visual Communications and Image Processing'98, Part 2, pp. 709–720.

Lee, J.H., Kim, D.H., Chung, C.W., 1999. Multi-dimensional selectivity estimation using compressed histogram information. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 205–214.

Meng, W., Liu, K.L., Yu, C., Wang, X., Chang, Y., Rishe, N., 1998. Determining text databases to search in the Internet. In: Proceedings of International Conference on Very Large Data Bases, pp. 14–25.

Meng, W., Liu, K.L., Yu, C., Wu, W., Rishe, L., 1999. Estimating the usefulness of search engines. In: Proceedings of International Conference on Data Engineering, pp. 146–153.

Ortega, M., Chakrababarti, K., Porkaew, K., Mehrotra, S., 1998. Supporting ranked Boolean similarity queries in MARS. IEEE Transactions on Knowledge and Data Engineering 10 (6), 905–925.

Park, S.H., 1985. Regression Analysis. DaeYoung Co.

Provost, F., Jensen, D., Oates, T., 1999. Efficient progressive sampling. In: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 23–32.

Smith, J.R., Chang, S.F., 1996. VisualSEEk: A fully automated content-based image query system. In: Proceedings of the ACM International Multimedia Conference, pp. 87–98.

Smith, J.R., Chang, S.F., 1997. Visually searching the web for content. IEEE Multimedia, 12–20.

Wyszecki, G., Stiles, W.S., 1982. Color Sciences: Concept and Methods, Quantitative Data and Formula, second ed. Wiley, New York.

Xu, J., Cao, Y., Lim, E.-P., Ng, W.-K., 1998. Database selection techniques for routing bibliographic queries. In: Proceedings of ACM Digital Libraries International Conference, pp. 264–273.

Yuwono, B., Lee, D.L., 1997. Server ranking for distributed text retrieval systems on the Internet. In: Proceedings of International Conference on DB Systems for Advanced Applications, pp. 391–400.

Deok-Hwan Kim received the B.S. degree in computer science and statistics from Seoul National University in 1987, and MS degree in information and communication engineering from KAIST in 1995. He is currently working toward the Ph.D. degree at KAIST. He was a senior engineer in Telecommunication R&D at LG Electronics from 1987 to 1996. He is currently an assistant professor in Department of Internet Information at Dongyang Technical College. His research interests focus on multimedia information retrieval and Web data mining.

Seok-Lyong Lee received B.S. degree in mechanical engineering in 1984 and MS degree in computer science in 1993 from Yonsei University, and Ph.D. degree in information and communication engineering from KAIST in 2001. He was an Advisory S/W Engineer in S/W Development Institute at IBM Korea from 1984 to 1995. He is currently an assistant professor in School of Industrial and Information Systems Engineering at Hankuk University of Foreign Studies. His major research interests include multimedia databases, data mining and warehousing, and Web information retrieval.

Chin-Wan Chung is professor and chair of the Division of Computer Science at the Korea Advanced Institute of Science and Technology (KAIST). From 1983 to 1993, he was a senior research scientist and a staff research scientist in the Computer Science Department at the General Motors Research Laboratories (GMR). He received a Ph.D. from the University of Michigan in 1983. While at GMR, he developed DATAPLEX, a heterogeneous distributed database management system integrating relational databases and hierarchical databases. At KAIST, he developed a full scale object-oriented spatial database management system called OMEGA, which supports ODMG standards. His current research interests include multimedia databases, XML, OLAP and spatio-temporal databases.