Int J Digit Libr (1997) 1: 311–328

INTERNATIONAL JOURNAL ON
Digital Libraries
© Springer-Verlag 1997

Original articles

Content-based image indexing and searching using Daubechies' wavelets*

James Ze Wang[1], **Gio Wiederhold**[2], **Oscar Firschein**[2], **Sha Xin Wei**[3]

[1] Department of Computer Science and School of Medicine, Stanford University, Stanford, CA 94305, USA
[2] Department of Computer Science, Stanford University, Stanford, CA 94305, USA
[3] Stanford University Libraries, Stanford University, Stanford, CA 94305, USA

Abstract. This paper describes WBIIS (Wavelet-Based Image Indexing and Searching), a new image indexing and retrieval algorithm with partial sketch image searching capability for large image databases. The algorithm characterizes the color variations over the spatial extent of the image in a manner that provides semantically meaningful image comparisons. The indexing algorithm applies a Daubechies' wavelet transform for each of the three opponent color components. The wavelet coefficients in the lowest few frequency bands, and their variances, are stored as feature vectors. To speed up retrieval, a two-step procedure is used that first does a crude selection based on the variances, and then refines the search by performing a feature vector match between the selected images and the query. For better accuracy in searching, two-level multiresolution matching may also be used. Masks are used for partial-sketch queries. This technique performs much better in capturing coherence of image, object granularity, local color/texture, and bias avoidance than traditional color layout algorithms. WBIIS is much faster and more accurate than traditional algorithms. When tested on a database of more than 10 000 general-purpose images, the best 100 matches were found in 3.3 seconds.

Key words: Content-based Retrieval – Image Databases – Image Indexing – Wavelets

1 Introduction

Searching a digital library [21] having large numbers of digital images or video sequences has become important in this visual age. Every day, large numbers of people are using the Internet for searching and browsing through different multimedia databases. To make such searching practical, effective image coding and searching based on image semantics is becoming increasingly important.

In current real-world image databases, the prevalent retrieval techniques involve human-supplied text annotations to describe image semantics. These text annotations are then used as the basis for searching, using mature text search algorithms that are available as freeware. However, there are many problems in using this approach. For example, different people may supply different textual annotations for the same image. This makes it extremely difficult to reliably answer user queries. Furthermore, entering textual annotations manually is excessively expensive for large-scale image databases.

Image feature vector indexing has been developed and implemented in several multimedia database systems such as the IBM QBIC System [7, 15] developed at the IBM Almaden Research Center, the Virage System [10] developed by Virage, Inc., and the Photobook System developed by the MIT Media Lab [16, 17]. For each image inserted into the database, a feature vector on the order of 500 elements is generated to accurately represent the content of the image. This vector is much smaller in size than the original image. The difficult part of the problem is to construct a vector that both preserves the image content and yet is efficient for searching. Once the feature vectors are generated, they are then stored in permanent storage. To answer a query, the image search engine scans through the previously computed vector indexes to select those with shortest *distances* to the image query vector. The distance is computed by a measure such as the vector distance in Euclidean space. For partial sketch queries, usually a mask is computed and applied to the feature vector.

In the WBIIS project, we developed a new algorithm to make semantically-meaningful comparisons of images efficient and accurate. Figure 1 shows the basic structure of the system. To accurately encode semantic features of images we employ wavelets based on continuous functions, as described by Daubechies [5]. Using these wavelets and statistical analysis, our algorithm produces

* Work supported in part by the Stanford University Libraries and Academic Information Resources. For original color figures in this paper, check out the URL: `http://www-db.stanford.edu/` ~wangz/project/imsearch/ IJODL97/

Correspondence to: wangz@cs.stanford.edu

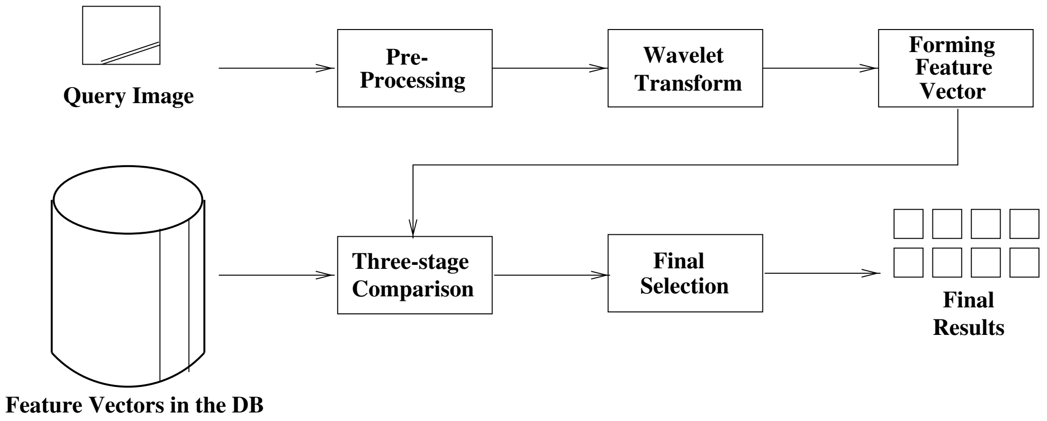


Fig. 1. Basic structure of the WBIIS system

feature vectors that provide a much better frequency localization than other traditional color layout coding algorithms. The localization of wavelets can be fine-tuned to deliver high resolution for higher frequencies and lower resolution for lower frequencies. We use a novel multistep metric to compute the distance between two given images. Promising results have been obtained in experiments using a database of 10 000 general-purpose images.

2 Preprocessing the images in the database

Many color image formats are currently in use, e.g., GIF, JPEG, PPM and TIFF are the most widely used formats. Because images in an image database can have different formats and different sizes, we must first normalize the data. For our test database of relatively small images, a rescaled thumbnail consisting of 128 × 128 pixels in Red-Green-Blue (RGB) color space is adequate for the purpose of computing the feature vectors.

Bilinear interpolation is used for the rescaling process. This method resamples the input image by overlaying the input image with a grid with 128 × 128 points. This gives one grid point for each pixel in the output image. The input image is then sampled at each grid point to determine the pixel colors of the output image. When grid points lie between input pixel centers, the color values of the grid point are determined by linearly interpolating between adjacent pixel colors (both vertically and horizontally).

This rescaling process is more effective than a Haar-like rescaling, i.e., averaging several pixels to obtain a single pixel to decrease image size, and replicating pixels to increase image size, especially when the image to be rescaled has frequent sharp changes such as local texture. It is necessary to point out, however, that the rescaling process is in general not important for the indexing phase when the size of the images in the database is close to the size to be rescaled. The sole purpose for the rescaling is to make it possible to use the wavelet transforms and to normalize the feature vectors. Here, we assume the images in the database to have sizes close to 128 × 128. In fact, images may be rescaled to any other size as long as each side length is a power of two. Therefore, to obtain a better performance for a database of mostly very large images, we would suggest using a bilinear interpolation to rescale to a large common size, with side lengths being powers of two, and then apply more levels of Daubechies' wavelets in the indexing phase.

Since color distances in RGB color space do not reflect the actual human perceptual color distance, we convert and store the image in a component color space with intensity and perceived contrasts. We define the new values at a color pixel based on the RGB values of an original pixel as follows:

$$\begin{cases} C_1 = (R + G + B)/3 \\ C_2 = (R + (\max - B))/2 \\ C_3 = (R + 2 * (\max - G) + B)/4 \end{cases} \tag{1}$$

Here max is the maximum possible value for each color component in the RGB color space. For a standard 24-bit color image, max $= 255$. Clearly, each color component in the new color space ranges from 0 to 255 as well. This color space is similar to the opponent color axes

$$\begin{cases} RG = R - 2 * G + B \\ BY = -R - G + 2 * B \\ WB = R + G + B \end{cases} \tag{2}$$

defined in [1] and [20].

Besides the perception correlation properties [11] of such an opponent color space, one important advantage of this alternative space is that the C_1 axis, or the intensity, can be more coarsely sampled than the other two axes on color correlation. This reduces the sensitivity of color matching to a difference in the global brightness of the image, and it reduces the number of bins and subsequent storage in the color histogram indexing.

3 Multiresolution color layout image indexing using wavelets and the fast wavelet transform

Many end-users are interested in searching an image database for images having similar image semantics with respect to a given query image or a hand-drawn sketch. Although it is not yet possible to fully index the image semantics using a computer vision approach, there are several ways to index the images so that semantically-meaningful queries can be performed by comparing the indexes. The color histogram is one of the many ways to index color images. However, while a global histogram preserves the color information contained in images, it does not preserve the color locational information. Thus, using similarity of histograms as a measure, two images may be considered to be very close to each other even though they have completely unrelated semantics. Shape and texture-based detection and coding algorithms are other techniques of indexing images. They both have substantial limitations for general-purpose image databases. For example, current shape detection algorithms only work effectively on images with relatively uniform backgrounds. Texture coding is not appropriate for non-textural images.

Storing color layout information is another way to describe the contents of the image. It is especially useful when the query is a partial sketch rather than a full image. In traditional color layout image indexing, we divide the image into equal-sized blocks, compute the average color on the pixels in each block, and store the values for image matching using Euclidean metric or variations of the Euclidean metric. It is also possible to compute the values based on statistical analysis of the pixels in the block. Both techniques are very similar to image rescaling or subsampling. However, they do not perform well when the image contains high frequency information such as sharp color changes. For example, if there are pixels of various colors ranging from black to white in one block, an effective result value for this block cannot be predicted using these techniques.

Work done by the University of Washington [12] applies the Haar wavelet to multiresolution image querying. Forty to sixty of the largest magnitude coefficients are selected from the $128^2 = 16\,384$ coefficients in each of the three color channels. The coefficients are stored as $+1$ or -1 along with their locations in the transform matrix. As demonstrated in the cited paper, the algorithm performs much faster than traditional algorithms, with an accuracy comparable to traditional algorithms when the query is a hand sketch or a low-quality image scan.

One drawback of using the Haar transform to decompose images into low frequency and high frequency is that it cannot efficiently separate image signals into low frequency and high frequency bands. From the signal processing point of view, since the wavelet transform is essentially a convolution operation, performing a wavelet transform on an image is equivalent to passing the image through a low-pass filter and a high-pass filter [9]. The low-pass and high-pass filters corresponding to the Haar transform do not have a sharp transition and fast attenuation property. Thus, the low-pass filter and high-pass filter cannot separate the image into clean distinct low frequency and high frequency parts. On the other hand, Daubechies wavelet transform with longer length filters [5] has better frequency properties. Because in our algorithm we rely on image low frequency information to do comparisons, we applied the Daubechies wavelet transform instead of the Haar transform.

Moreover, due to the normalization of functional space in the wavelet basis design, the wavelet coefficients in the lower frequency bands, i.e., closer to the upper-left corner in a transform matrix, tend to be more dominant (are of larger magnitude) than those in the higher frequency bands. Coefficients obtained by sorting and truncating will most likely be in the lower frequency bands. For the Haar case,

$$F_0(x(n)) = \frac{1}{\sqrt{2}}(x(n) + x(n+1)) \tag{3}$$

$$F_1(x(n)) = \frac{1}{\sqrt{2}}(x(n) - x(n+1)) \tag{4}$$

coefficients in each band are expected to be $2/\sqrt{2}$ times larger in magnitude than those in the next higher frequency band, i.e., those in one level previous to the current level. For a 128×128 image, we expect the coefficients in the transform to have an added weight varying from 1 to 8 before the truncation process. As indicated in Eq. (3), the low frequency band in a Haar wavelet transform is mathematically equivalent to the averaging color block or image rescaling approach in traditional layout algorithms mentioned above. Thus, the accuracy is not improved when the query image or the images in the database contain high frequency color variation.

Although the University of Washington approach can achieve a much faster comparison by storing only 40 to 60 coefficients for each color channel as a feature vector, much useful information about the image is discarded. Thus, it is possible for two images having the same feature vector to differ completely in content. In addition, two pictures with similar content but different locations of sharp edges may have feature vectors that are far apart in feature space. This is why the University of Washington algorithm has a sharp decrease in performance when the query image consisted of a small translation of the target image.

We have developed a color layout indexing scheme using Daubechies' wavelet transforms that better represents image semantics, namely, object configuration and local color variation, both represented by Daubechies' wavelet coefficients. For large databases, feature vectors obtained from multi-level wavelet transforms are stored to speed up the search. We apply a fast wavelet transform (FWT) with Daubechies' wavelet to each image in the database, for each of the three color components. Some coefficients of the wavelet transform, and their standard deviations, are stored as feature vectors. Given a query image, the search is carried out in two steps. In the first step, a crude selection based on the standard deviations stored is carried out. In the second step, a weighted version of the Euclidean distance between the feature coefficients of an image selected in the first step and those of the querying image is calculated, and the images with the smallest distances are selected and sorted as matching images to the query. We will show below that this algorithm can be used to handle partial hand-drawn sketch queries by modifying the computed feature vector.

3.1 Daubechies' wavelets and fast wavelet transform

When processing signals, the prime consideration is the localization, i.e., the characterization of local properties, of a given basis function in time and frequency. In our case, the signals we are dealing with are 2-D color images, for which the time domain is the spatial location of certain color pixels and the frequency domain is the color variation around a pixel. Thus, we seek a basis function that can effectively represent the color variations in each local spatial region of the image. In this subsection, we examine the various transforms and their properties to arrive at a transform that has attractive properties for the image retrieval problem.

Spline-based methods are efficient in analyzing the spatial localization for signals that contain only low frequencies. Traditional Fourier-based methods [4, 8], such as the Discrete Cosine Transform (DCT) aim to capture the frequency content of the signal. The Discrete Fourier Transform and its inverse are defined as

$$F[k] = \sum_{n=0}^{N-1} f[n] \mathrm{e}^{-j2\pi nk/N} \tag{5}$$

$$f[n] = \frac{1}{N} \sum_{k=0}^{N-1} F[k] \mathrm{e}^{j2\pi nk/N}. \tag{6}$$

Discrete Fourier Transforms are currently used effectively in signal and image processing because of the frequency domain localization capability. They are ideal for analyzing periodic signals because the Fourier expansions are periodic. However, they do not have the spatial localization property because of their infinite extensibility.

Two mathematical methods are available for non-periodic signals, the Windowed Fourier Transform (WFT) and the wavelet transform. The WFT analyzes the signal in both spatial and frequency domains simultaneously by encoding the signal through a scaled window related to both location and local frequency. Therefore, signals are easily underlocalized or overlocalized in spatial domain if the spatial behavior is inconsistent with the frequency of the signal. Wavelets are basis functions that have some similarities to both splines and Fourier series. They have advantages when the aperiodic signal contains many discontinuities or sharp changes.

Wavelets, developed in mathematics, quantum physics, and statistics, are functions that decompose signals into different frequency components and analyze each component with a resolution matching its scale. Applications of wavelets to signal denoising, image compression, image smoothing, fractal analysis and turbulence characterization are active research topics [22, 18].

Wavelet analysis can be based on an approach developed by Haar [14]. Haar found an orthonormal base defined on $[0, 1]$, namely $h_0(x), h_1(x), \ldots, h_n(x), \ldots$, other than the Fourier bases, such that for any continuous function $f(x)$ on $[0, 1]$, the series

$$\sum_{j=1}^{\infty} \langle f, h_j \rangle h_j(x) \tag{7}$$

converges to $f(x)$ uniformly on $[0, 1]$. Here, $\langle u, v \rangle$ denotes $\int_0^1 u(x)\overline{v(x)}dx$ and $\overline{v}$ is the complex conjugate of v.

One version of Haar's construction [14,2,3] can be written as follows:

$$h(x) = \begin{cases} 1, & x \in [0, 0.5) \\ -1, & x \in [0.5, 1) \\ 0, & \text{elsewhere} \end{cases} \tag{8}$$

$$h_n(x) = 2^{j/2} h(2^j x - k) \tag{9}$$

where $n = 2^j + k$, $k \in [0, 2^j)$, $x \in [k2^{-j}, (k+1)2^{-j})$.

There are problems with Haar's construction. For example, Haar's base functions are discontinuous step functions and are not suitable for analyzing continuous functions with continuous derivatives. If we consider images as 2-D continuous surfaces, we know that Haar's base functions are not appropriate for image analysis.

Another basis for wavelets is that of Daubechies. For each integer r, Daubechies' orthonormal basis [5, 6, 13] for $L^2(\mathbb{R})$ is defined as

$$\phi_{r,j,k}(x) = 2^{j/2} \phi_r(2^j x - k), \; j, k \in \mathbb{Z} \tag{10}$$

where the function $\phi_r(x)$ in $L^2(\mathbb{R})$ has the property that $\{\phi_r(x-k) | k \in \mathbb{Z}\}$ is an orthonormal sequence in $L^2(\mathbb{R})$.

Then the *trend* f_j, at scale 2^{-j}, of a function $f \in L^2(\mathbb{R})$ is defined as

$$f_j(x) = \sum_k \langle f, \phi_{r,j,k} \rangle \phi_{r,j,k}(x). \tag{11}$$

The *details* or *fluctuations* are defined by

$$d_j(x) = f_{j+1}(x) - f_j(x). \tag{12}$$

To analyze these details at a given scale, we define an orthonormal basis $\psi_r(x)$ having properties similar to those of $\phi_r(x)$ described above.

$\phi_r(x)$ and $\psi_r(x)$, called the *father wavelet* and the *mother wavelet*, respectively, are the wavelet prototype functions required by the wavelet analysis. Figure 2 shows some popular mother wavelets. The family of wavelets such as those defined in Eq. (10) are generated from the father or the mother wavelet by change of scale and translation in time (or space in image processing).

Daubechies' orthonormal basis has the following properties:

- ψ_r has the compact support interval $[0, 2r+1]$
- ψ_r has about $r/5$ continuous derivatives
- $\int_{-\infty}^{\infty} \psi_r(x)dx = \cdots = \int_{-\infty}^{\infty} x^r \psi_r(x)dx = 0.$

Daubechies' wavelets give remarkable results in image analysis and synthesis due to the above properties. In fact, a wavelet function with compact support can be easily implemented by finite length filters. This finite length property is important for spatial domain localization. Furthermore, functions with more continuous derivatives analyze continuous functions more efficiently and avoid the generation of edge artifacts. Since the mother wavelets are used to characterize details in the signal, they should have a zero integral so that the trend information is stored in the coefficients obtained by the

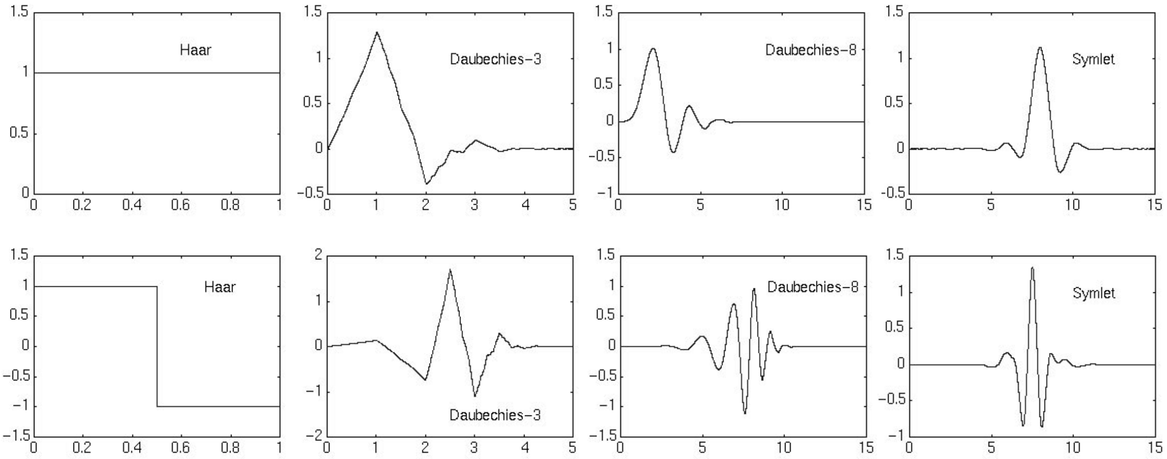


Fig. 2. Plots of some analyzing wavelets. First row: father wavelets, $\phi(x)$. Second row: mother wavelets, $\psi(x)$

father wavelet. A Daubechies' wavelet representation of a function is a linear combination of the wavelet function elements.

Daubechies' wavelets are usually implemented in numerical computation by quadratic mirror filters [14]. Multiresolution analysis of trend and fluctuation is implemented using convolution with a low-pass filter and a high-pass filter that are versions of the same wavelet. For example, if we denote the sampled signals as $x(n), n \in \mathbb{Z}$, then Eq.(3) and Eq.(4) are quadratic mirror filters for Haar's wavelet. In fact, average color block layout image indexing is equivalent to the Haar transform with high-pass filtering neglected. Daubechies' wavelets transform is more like a weighted averaging which better preserves the trend information stored in the signals if we consider only the low-pass filter part. Although Daubechies' wavelets may not be better than Haar's for all image analysis applications, various experiments and studies [22] have shown that Daubechies' wavelets are better for dealing with general-purpose images.

Figures 5 and 6 show comparisons of the Haar wavelet, which is equivalent to average color blocks, and Daubechies' wavelets. In Fig. 5, we notice that the signal with a sharp spike is better analyzed by Daubechies' wavelets because much less energy or trend is stored in the high-pass bands. Daubechies' wavelets are better suited for natural signals or images than a flat Haar wavelet. In layout image indexing, we want to represent as much energy in the image as possible in the coefficients of the feature vector. When using the Haar wavelet, we lose much trend information in the discarded high-pass bands. Figure 6 shows the reconstruction of two images based only on the feature vectors of traditional layout indexing (same as Haar) and those of WBIIS using Daubechies' wavelets. Clearly, images reconstructed by saved Daubechies' coefficients are closer to the original images than those reconstructed by saved Haar's coefficients. Here, we use image reconstruction to compare information loss or encoding efficiency between Haar and Daubechies in the course of truncating discrete wavelet representations. Although these two examples in themselves do not imply that a searching scheme using Daubechies' wavelets is better than that using Haar's wavelet, they may help explain observations on how the schemes function. Figures 11 and 10 show the results of the searches using the two different wavelet bases. Saved Haar wavelet coefficients do not capture high frequency local texture as effectively as the saved Daubechies' wavelet coefficients.

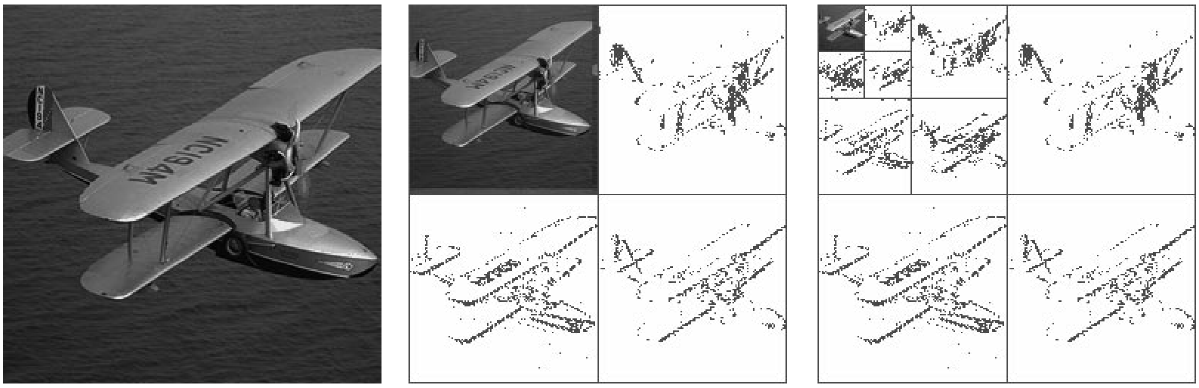
Because the original signal can be represented in terms of a wavelet expansion using coefficients in a linear combination of the wavelet functions, similar to Fourier analysis, data operations can be performed using just the corresponding wavelet coefficients. If we truncate the coefficients below a threshold, image data can be sparsely represented.

The wavelet transform offers good time and frequency localization. Information stored in an image is decomposed into averages and differences of nearby pixels. The information in smooth areas is decomposed into the average element and near-zero difference elements. The wavelets approach is therefore a suitable tool for data compression, especially for functions with considerable local variations. For example, the basis functions are very flexible with respect to both scale index j and position index k. We may decompose the image even further by applying the wavelet transform several times recursively. Figure 3 shows the multi-scale structure in the wavelet transform of an image.

3.2 Wavelet image layout indexing in WBIIS

The discrete wavelet transform (DWT) we described can be directly used in image indexing for color layout type queries. Our algorithm is as follows:

For each image to be inserted to the database, obtain 128×128 square rescaled matrices in (C_1, C_2, C_3) components following Eq. (1) in Sect. 2. Compute a 4-layer 2-D fast wavelet transform on each of the three matrices using Daubechies' wavelets. Denote the three matrices obtained from the transforms as $W_{C_1}(1:128, 1:128)$,



original image (256 × 256) *1-level wavelet transform* *3-level wavelet transform*

Fig. 3. Multi-scale structure in the wavelet transform of an image. Dots indicates non-zero wavelet coefficients after thresholding. Daubechies-8 wavelet is used for this transform

$W_{C_2}(1:128,1:128)$ and $W_{C_3}(1:128,1:128)$[1] Then the upper-left 8×8 corner of each transform matrix, $W_{C_i}(1:8,1:8)$, represents the lowest frequency band of the 2-D image in a particular color component for the level of wavelet transform we used. The lower frequency bands in the wavelet transform usually represent object configurations in the images and the higher frequency bands represent texture and local color variation. The three 8×8 submatrices (namely, $W_{C_i}(1:8,9:16)$, $W_{C_i}(9:16,1:8)$ and $W_{C_i}(9:16,9:16)$) closest to the 8×8 corner submatrix $W_{C_i}(1:8,1:8)$ represent detailed information in the original image to some extent, though most of the fluctuation information is stored in the thrown-away higher frequency band coefficients. Extracting a submatrix $W_{C_i}(1:16,1:16)$ of size 16×16 from that corner, we get a semantic-preserving compression of 64:1 over the original thumbnail of 128×128 pixels. We store this as part of the feature vector.

Then we compute the standard deviations, denoted as $\sigma_{c_1}, \sigma_{c_2}, \sigma_{c_3}$, of the 8×8 corner submatrices $W_{C_i}(1:8,1:8)$. Three such standard deviations are then stored as part of the feature vector as well. Figure 4 shows two images with the upper-left corner submatrices of their 2-D fast wavelet transforms in (C_1, C_2, C_3) color space. Notice that the standard deviation of the coefficients in the lowest frequency band obtained from the first image differs considerably from that obtained from the second image. Since the standard deviations are computed based on the wavelet coefficients in the lowest frequency band, we have eliminated disturbances arising from detailed information in the image.

We also obtain a 5-level 2-D fast wavelet transform using the same bases. We extract and store a submatrix of size 8×8 from the upper-left corner. Thus, we have stored a feature index using the multiresolution capability of the wavelet transform.

Because the set of wavelets is an infinity set, different wavelets may give different performance for different types of image. One should take advantage of this characteristic in designing an image retrieval system. To match the characteristics of the signal we are analyzing, we used a Daubechies-8 or Symmlet-8 wavelet for the DWT process. Symmlets were designed by Daubechies [6] to be orthogonal, smooth, nearly symmetric, and non-zero on a relatively short interval (compact support). Wavelet subclasses are distinguished by the number of coefficients and by the level of iteration. Most often they can be classified by the number of vanishing moments. The number of vanishing moments is weakly linked to the number of oscillations of the wavelet, and determines what the wavelet does or does not represent. The number of vanishing moments for the subclass of our Symmlet wavelet is 8, which means that our wavelet will ignore linear through eighth degree functions.

Wavelets perform better than traditional layout coding because the coefficients in wavelet-created compression data actually contain sufficient information to reconstruct the original image at a lower loss rate using an inverse wavelet transform.

3.3 Wavelet image layout matching in WBIIS

When a user submits a query, we must compute the feature vector for the querying image and match it to the pre-computed feature vectors of the images in the database. This is done in two phases.

In the first phase, we compare the standard deviations stored for the querying image with the standard deviations stored for each image in the database.

Figure 7 demonstrates the histograms of the standard deviations we computed for general-purpose images. Studying the three histograms, we found that the standard deviations of the intensity component are a lot more diverse than those of the other two. We would consider σ_{C_1} more dominant than σ_{C_2} or σ_{C_3} alone. Also, more images in this general-purpose image database have lower standard deviations. For any given standard deviation computed for the query, we want to find roughly the same number of images having standard deviations close to those of the query. Based on the trends shown in the histograms, we have developed the following selection criterion for the first step.

[1] Here we use MATLAB notation. That is, $A(m_1:n_1, m_2:n_2)$ denotes the submatrix with opposite corners $A(m_1, m_2)$ and $A(n_1, n_2)$

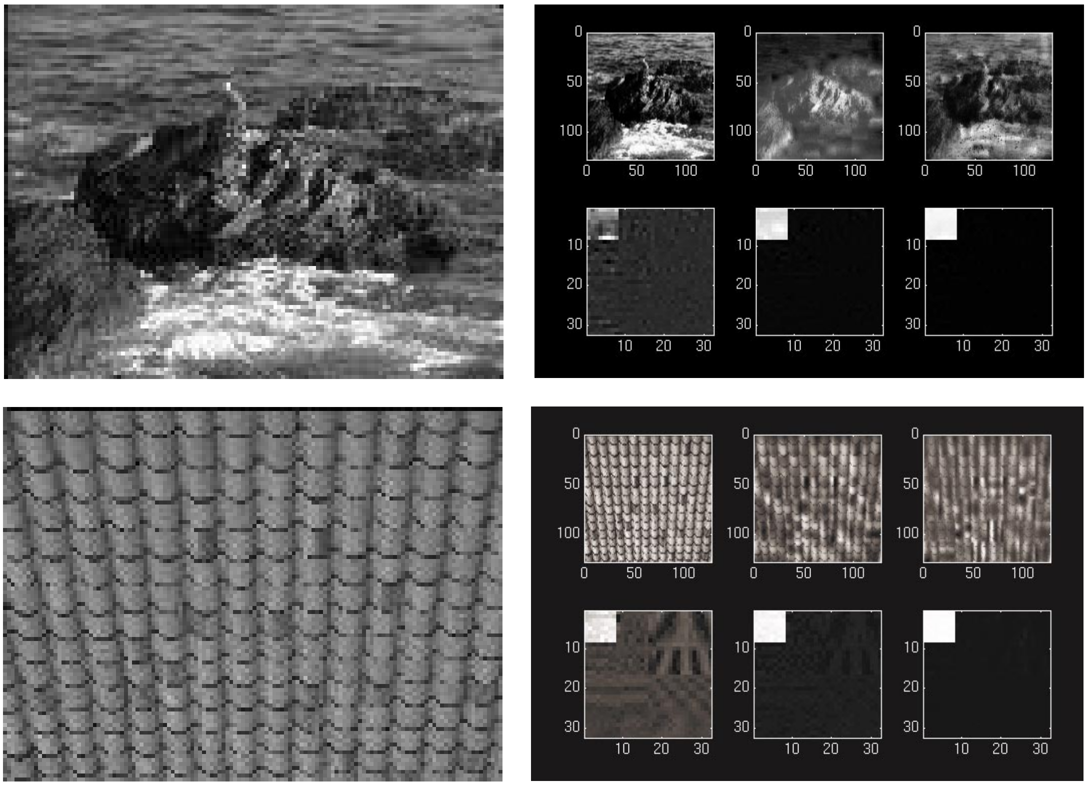


Fig. 4. Two images with the upper-left corner submatrices of their fast wavelet transforms in (C_1, C_2, C_3) color space. The standard deviations we stored for the first image are $\sigma_{C_1} = 215.93$ $\sigma_{C_2} = 25.44$, and $\sigma_{C_3} = 6.65$ while means of the coefficients in the lowest frequency band are $\mu_{C_1} = 1520.74$, $\mu_{C_2} = 2124.79$, and $\mu_{C_3} = 2136.93$. The standard deviations we stored for the second image are $\sigma_{C_1} = 16.18$, $\sigma_{C_2} = 10.97$, and $\sigma_{C_3} = 3.28$ while means of the coefficients in the lowest frequency band are $\mu_{C_1} = 1723.99$, $\mu_{C_2} = 2301.24$ and $\mu_{C_3} = 2104.33$

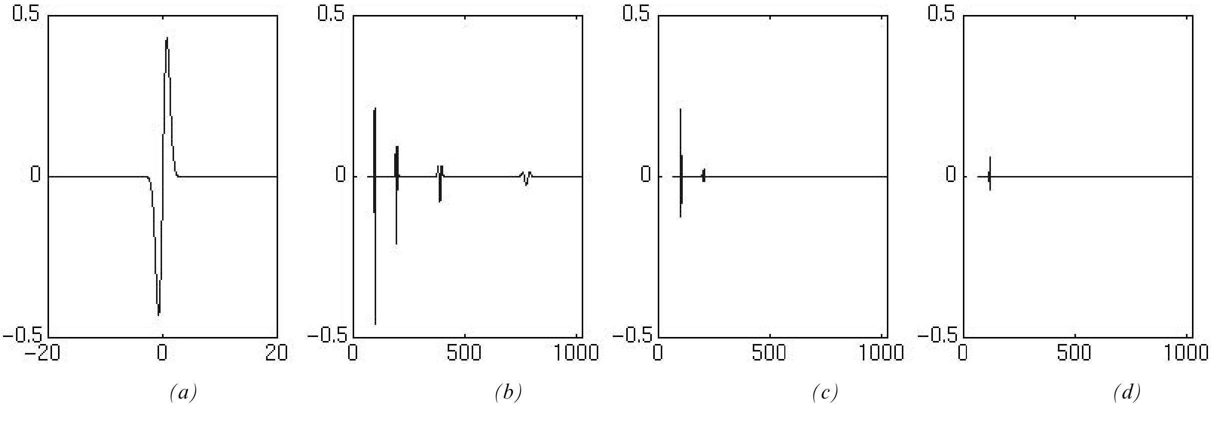


Fig. 5. Comparison of Haar's wavelet and Daubechies wavelets on a 1-D signal. (a) original signal (xe^{-x^2}) of length 1024 (b) coefficients in high-pass bands after a 4-layer Haar transform (c) coefficients in high-pass bands after a 4-layer Daubechies-3 transform (d) coefficients in high-pass bands after a 4-layer Daubechies-8 transform

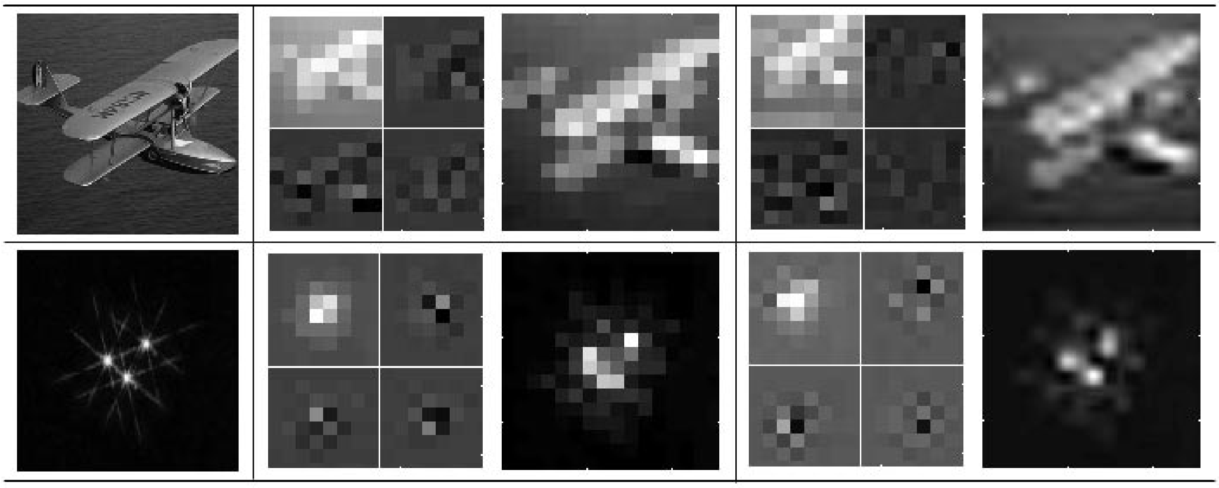
Denote the standard deviation information computed for the querying image as σ_{c_1}, σ_{c_2} and σ_{c_3}. Denote the standard deviation information stored in the database indexing for an image as σ'_{c_1}, σ'_{c_2} and σ'_{c_3}.

If the acceptance criteria[2]

$$\left(\sigma_{c_1}\beta < \sigma'_{c_1} < \frac{\sigma_{c_1}}{\beta}\right) \; || \; \left[\left(\sigma_{c_2}\beta < \sigma'_{c_2} < \frac{\sigma_{c_2}}{\beta}\right) \;\&\&\; \left(\sigma_{c_3}\beta < \sigma'_{c_3} < \frac{\sigma_{c_3}}{\beta}\right)\right]$$

fails, then we set the distance of the two images to 1, which means that the image will not be further considered in the matching process. Here, $\beta = 1 - \frac{percent}{100}$ and *percent* is a threshold variable set to control the number

[2] Here we use standard C notation. That is, | | denotes OR and && denotes AND.

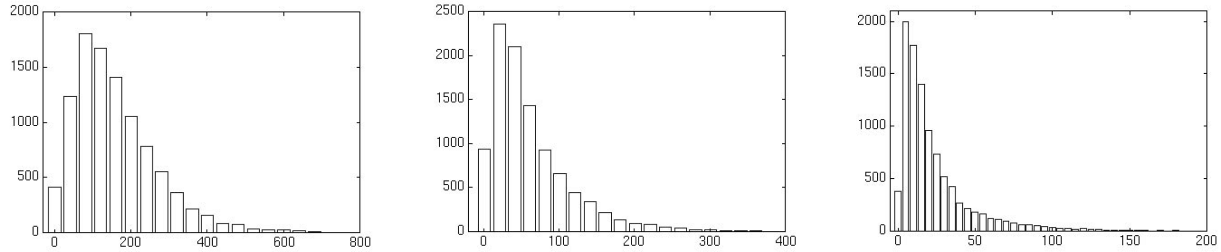


Original image (128 × 128)

'saved Haar's coeff. (16 × 16) *and its image recpmstruction*

'saved Daubechie's coeff. (16 × 16) *and its image recpmstruction*

Fig. 6. Comparison of Haar's wavelet and Daubechies-8 wavelet



histogram of σ_{C_1}

histogram of σ_{C_2}

histogram of σ_{C_3}

Fig. 7. Histogram of the standard deviations of the wavelet coefficients in the lowest frequency band. Results were obtained from a database of more than 10 000 general purpose images

of images passing the first matching phase. Usually it is set to around 50. Note that the above acceptance criteria holds if and only if

$$\left(\sigma'_{c_1}\beta < \sigma_{c_1} < \frac{\sigma'_{c_1}}{\beta}\right) \; || \; \left[\left(\sigma'_{c_2}\beta < \sigma_{c_2} < \frac{\sigma'_{c_2}}{\beta}\right) \;\&\&\; \left(\sigma'_{c_3}\beta < \sigma_{c_3} < \frac{\sigma'_{c_3}}{\beta}\right)\right]$$

holds.

Having first a fast and rough cut and then a more refined pass maintains the quality of the results while improving the speed of the matching. Usually about one fifth of the images in the whole database passes through the first cut. That means we obtain a speed-up of about five by doing this step. For a database of 10 000 images, about 2000 images will still be listed in the queue for the Euclidean distance comparison. Although it is possible that the first pass may discard some images that should be in the result list, in most cases the quality of the query response is slightly improved due to this first pass. In fact, an image with almost the same color, i.e., low standard deviation, is very unlikely to have the same semantics as an image with very high variation or high standard deviation.

A weighted variation of Euclidean distance is used for the second phase comparison. If an image in the database differs from the querying image too much when we compare the $8 \times 8 \times 3 = 192$ dimensional feature vector, we discard it. The remaining image vectors are used in the final matching, using the $16 \times 16 \times 3 = 768$ dimensional feature vector with more detailed information considered. Let $w_{1,1}$, $w_{1,2}$, $w_{2,1}$, $w_{2,2}$, w_{c_1}, w_{c_2} and w_{c_3} denote the weights. Then our distance function is defined as

$$\begin{aligned} &Dist(Image, Image') \\ &= w_{1,1}\sum_{i=1}^{3}(w_{c_i} \parallel W_{C_i,1,1} - W'_{C_i,1,1} \parallel) \\ &+ w_{1,2}\sum_{i=1}^{3}(w_{c_i} \parallel W_{C_i,1,2} - W'_{C_i,1,2} \parallel) \\ &+ w_{2,1}\sum_{i=1}^{3}(w_{c_i} \parallel W_{C_i,2,1} - W'_{C_i,2,1} \parallel) \\ &+ w_{2,2}\sum_{i=1}^{3}(w_{c_i} \parallel W_{C_i,2,2} - W'_{C_i,2,2} \parallel) \end{aligned}$$

where

commercial algorithm

WBIIS

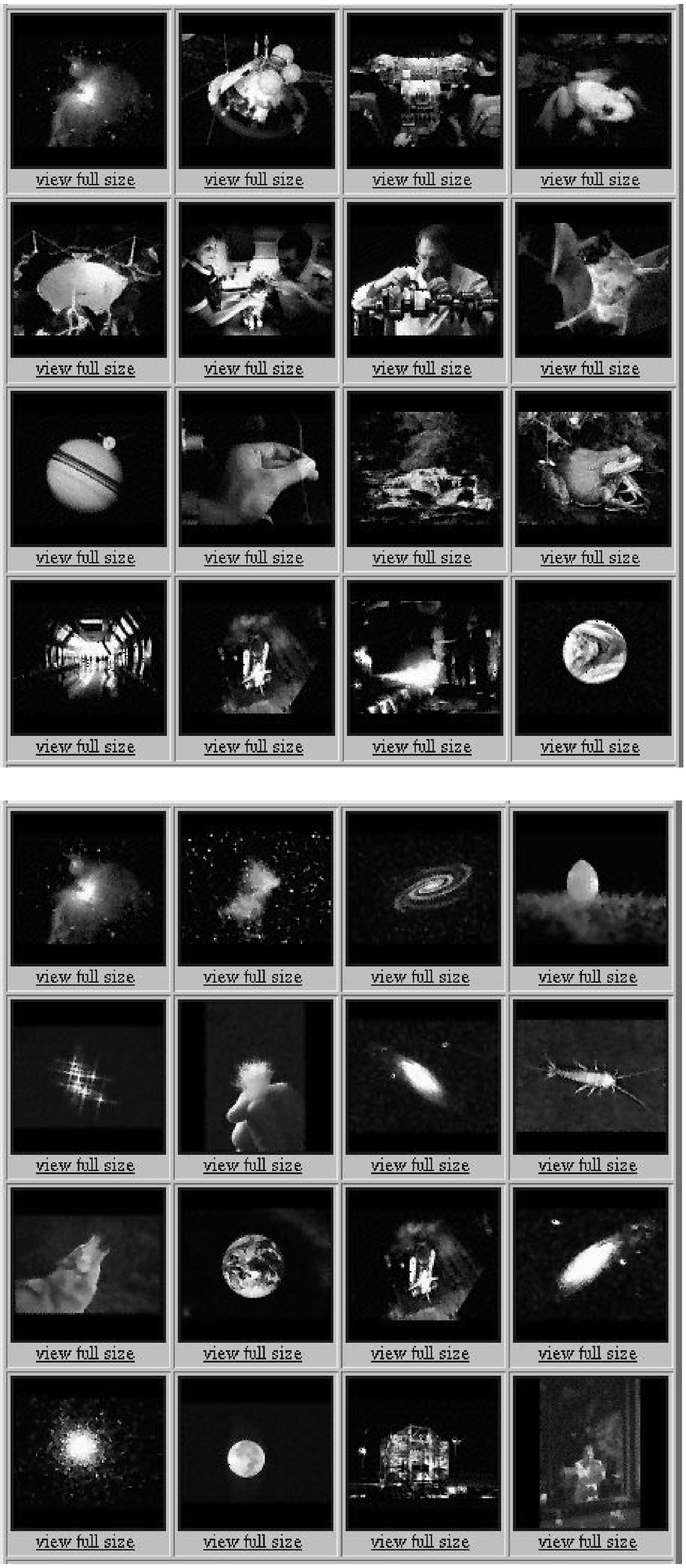
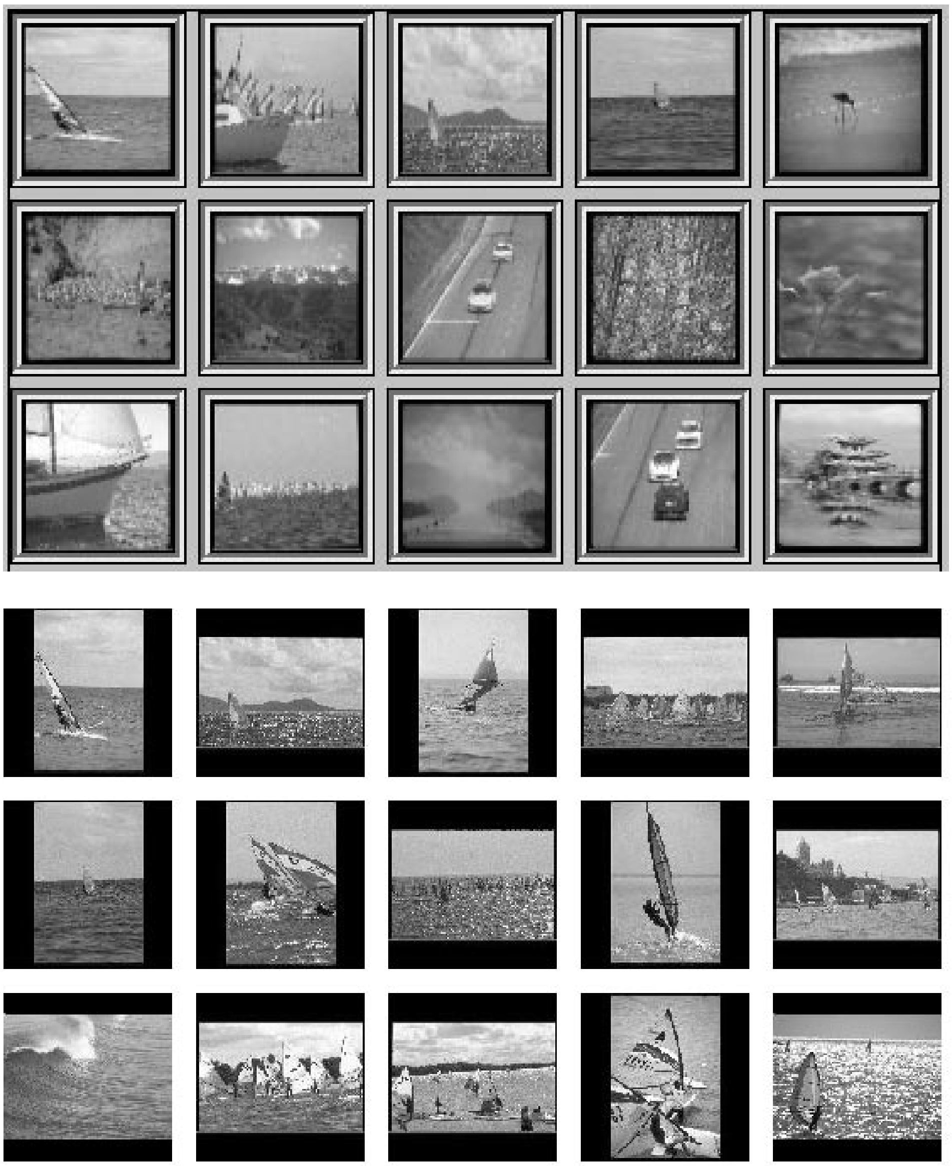


Fig. 8. Comparisons with a commercial algorithm on a galaxy-type image. Note that many images unrelated to the galaxy query image are retrieved by the commercial algorithm. The upper-left corner image in each block of images is the query. The image to the right of that image is the best matching image found. And so on. Results were obtained from a database of approximately 10 000 images



algorithm by University of Washington

WBIIS

Fig. 9. Query example. Many images unrelated to a water scene are retrieved by the University of Washington algorithm. The upper-left corner image in each block of images is the query. Results were obtained from a database of approximately 10 000 images

$$
\begin{aligned}
W_{C_i,1,1} &= W_{C_i}(1:8,1:8)\\
W_{C_i,1,2} &= W_{C_i}(1:8,9:16)\\
W_{C_i,2,1} &= W_{C_i}(9:16,1:8)\\
W_{C_i,2,2} &= W_{C_i}(9:16,9:16)
\end{aligned}
$$

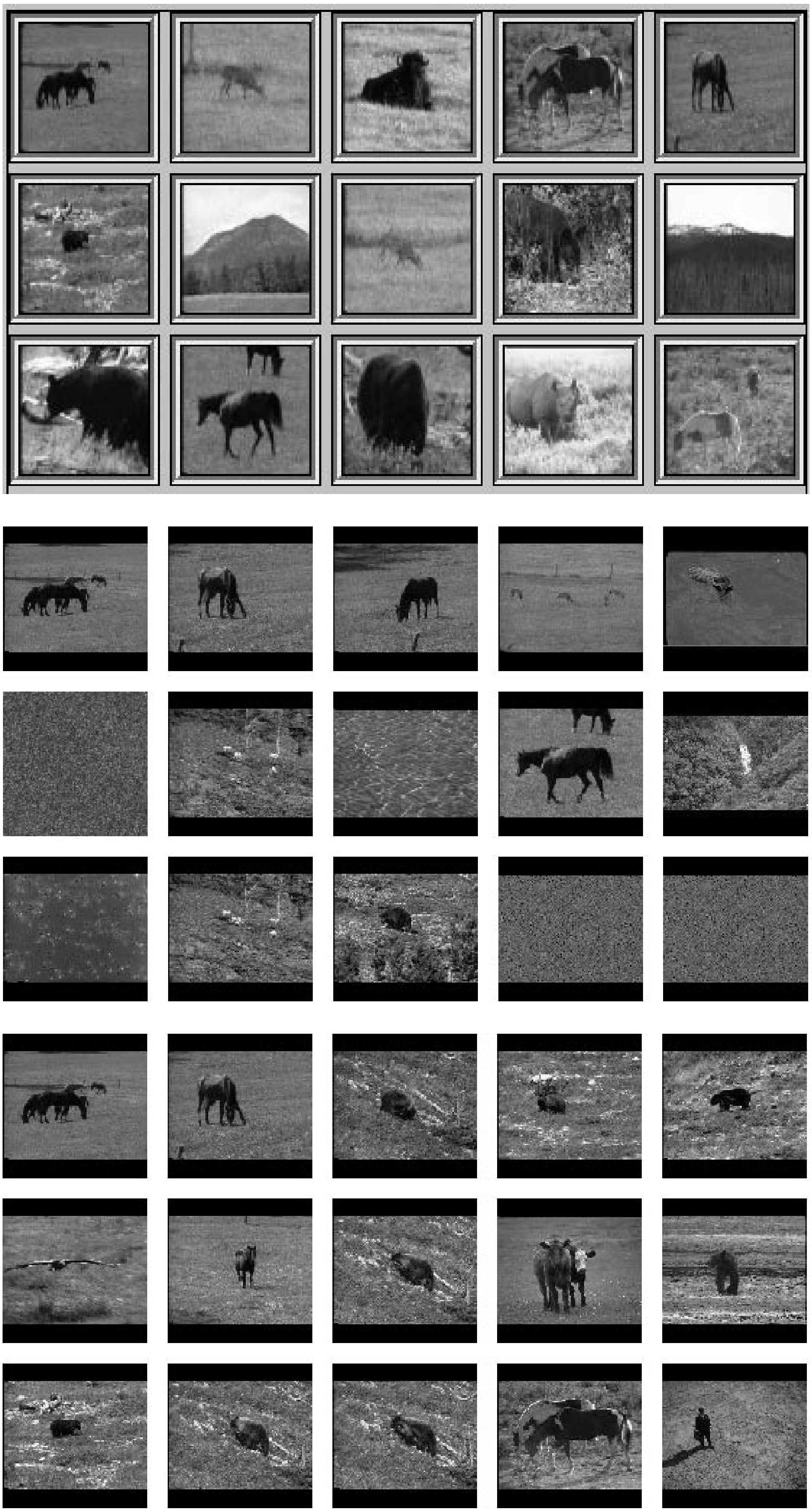
and $\| u - v \|$ denotes the Euclidean distance. In practice, we may compute the square of the Euclidean distances instead in order to reduce computation complexity. If we let $w_{j,k} = 1$, then the function $Dist(I_1, I_2)$ is the Euclidean distance between I_1 and I_2. However, we may raise $w_{2,1}$, $w_{1,2}$, or $w_{2,2}$ if we want to emphasize the vertical, horizontal or diagonal edge details in the image. We may also raise w_{c_2} or w_{c_3} to emphasize the color variation more than the intensity variation.

To further speed up the system, we use a component threshold to reduce the amount of Euclidean distance computation. That is, if the difference at any component within the feature vectors to be compared is higher than a pre-defined threshold, we set the distance of the two images immediately to 1 so that the image will not be further considered in the matching process.

The angle of any two feature vectors in the n-dimensional feature vector space is an alternative measure to the Euclidean distance we discussed above. The cosine value of the angle can be obtained by computing the vector dot product in a normalized vector space. This alternative measure reduces the sensitivity to color or brightness shift.

3.4 Wavelet partial query layout matching in WBIIS

A partial image query can be based on an image of low resolution, a partial image, a very low resolution block

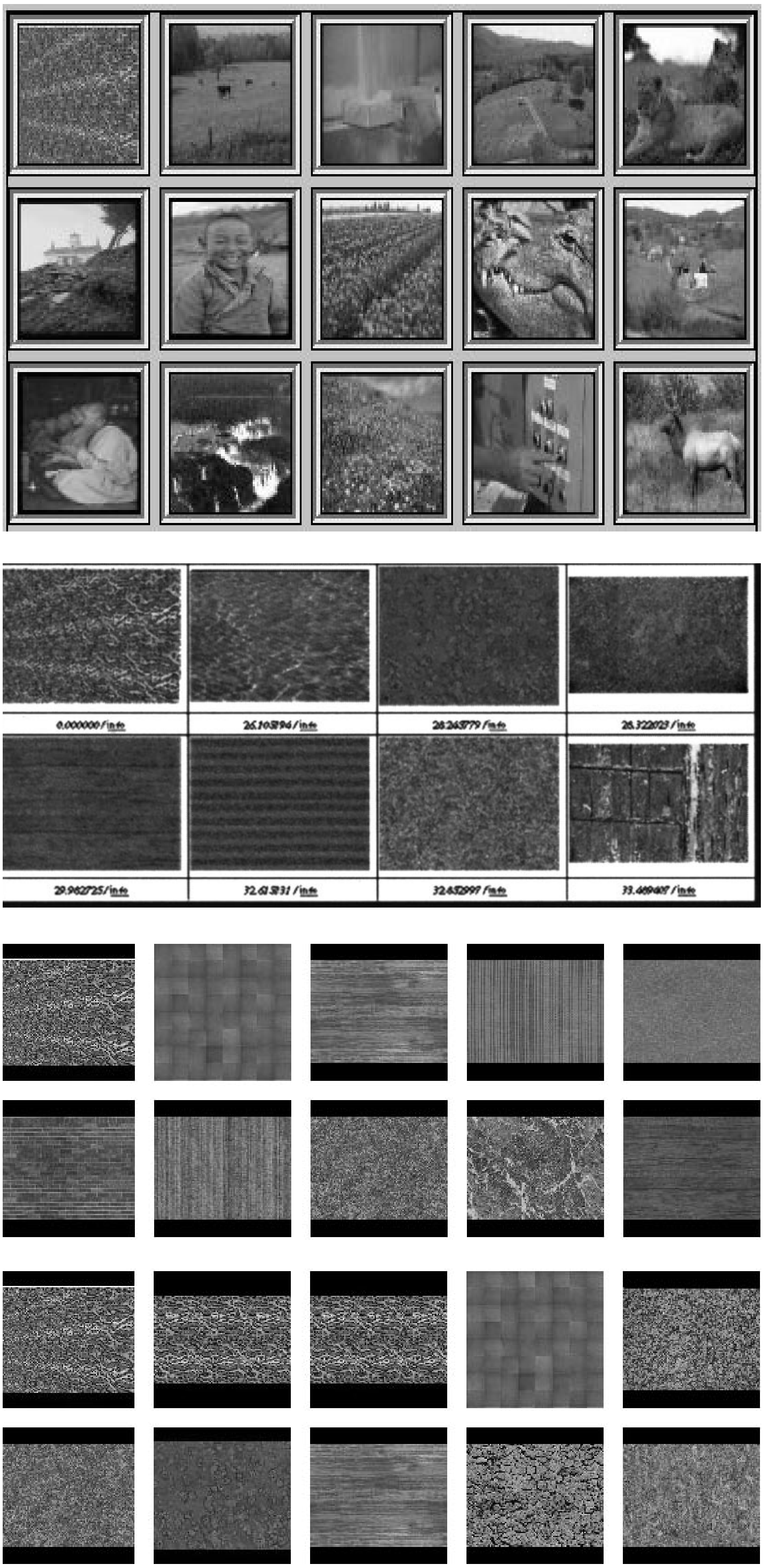


algorithm by University of Washington

WBIIS with Haar Wavelet

WBIIS with Daubechies' Symmlet-8 Wavelet

Fig. 10. Another query example. The upper-left corner image in each block of images is the query. Results were obtained from a database of approximately 10 000 images



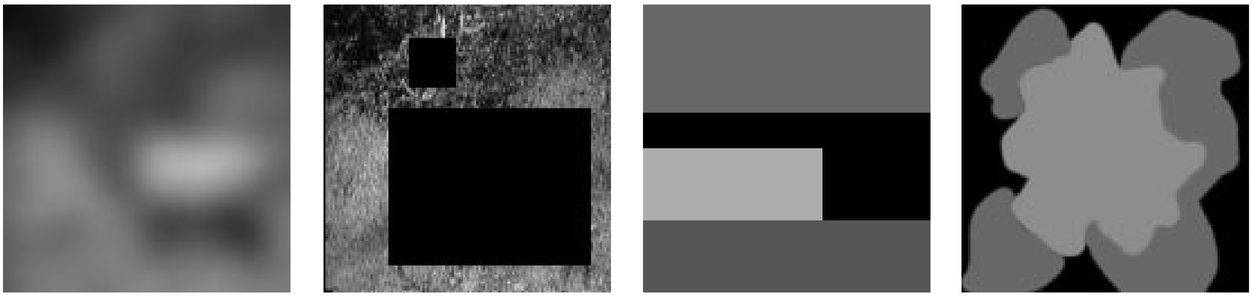
commercial algorithm

algorithm by University of Washington

WBIIS with Haar Wavelet

WBIIS with Daubechies' Symmlet-8 Wavelet

Fig. 11. Comparison on a texture image. The upper-left corner image in each block of images is the query. Results were obtained from a database of approximately 10 000 images



low resolution *partial image* *block sketch* *hand-drawn sketch*

Fig. 12. Types of partial sketch queries our WBIIS system aims to handle. Black areas in a query image represent non-specified areas

sketch or a hand-drawn sketch. Figure 12 shows the different types of partial image queries our system is designed to handle. We assume that the users do not care about the non-specified areas, but are only interested in finding images in the database that best match the specified areas of the query image. This kind of query is very useful in real-world digital libraries. For example, if a user wants to find all images with a racing car of any color in the center of an image, the user may simply form a query by cutting off the center area of an image with a white car.

To handle partial image queries, spatial localization of the feature vector is crucial. For example, if we use some variations of the color moments to represent images, we would not be able to answer partial sketch queries because each element in a feature vector is a function of all pixels in the image. Due to the spatial localization properties of our wavelet-based image indexing, we can implement a retrieval algorithm for partial sketch queries with ease.

When a user submits a partial image query, we first rescale the query image into a 128 × 128 rescaled image. At the same time, the non-specified areas are rescaled to fit in the 128 × 128 rescaled image. A binary mask, denoted initially as $M_0(1:128, 1:128)$ is created to represent the specified areas. Then we compute the feature vector of the rescaled query image using the wavelet-based indexing algorithm we discussed above with the non-specified areas being assigned as black. Here, the standard deviations are computed based on the wavelet coefficients within an 8 × 8 mask $M_4(1:8, 1:8)$ which is a subsample of $M_0(1:128, 1:128)$.

Comparison of the query feature vector with the stored vectors for the image database is done in two phases.

In the first phase, we compare the standard deviations computed for the querying image with the standard deviations within the mask for the wavelet coefficients stored for each image in the database. That is, we need to first re-compute the standard deviations of the wavelet coefficients in the masked areas for each image in the database. In cases where the users specify a majority of pixels in the query, we may simply use the pre-computed and stored standard deviation information. Then a similar distance measure is used to compare the standard deviation information.

A masked weighted variation of the Euclidean distance is used for the second phase comparison. The distance function is defined as[3]

$$\begin{aligned}
&Dist(Image, Image') \\
&= w_{1,1} \sum_{i=1}^{3} (w_{c_i} \parallel M_4 \ .* W_{C_i,1,1} - M_4 \ .* W'_{C_i,1,1} \parallel) \\
&+ w_{1,2} \sum_{i=1}^{3} (w_{c_i} \parallel M_4 \ .* W_{C_i,1,2} - M_4 \ .* W'_{C_i,1,2} \parallel) \\
&+ w_{2,1} \sum_{i=1}^{3} (w_{c_i} \parallel M_4 \ .* W_{C_i,2,1} - M_4 \ .* W'_{C_i,2,1} \parallel) \\
&+ w_{2,2} \sum_{i=1}^{3} (w_{c_i} \parallel M_4 \ .* W_{C_i,2,2} - M_4 \ .* W'_{C_i,2,2} \parallel)
\end{aligned}$$

If an image in the database differs from the querying image too much when we compare the $8 \times 8 \times 3 = 192$ dimensional feature vector, we discard it. The remaining image vectors are used in the final matching, using the $16 \times 16 \times 3 = 768$ dimensional feature vector. The measure is the same as discussed in the previous subsection except that we usually assign different weights in the three color components for partial queries with low resolution. In fact, when the resolution in the partial sketch is low, we need to emphasize the color variation rather than the intensity variation. For example, a red block (i.e., R = 255, G = 0, B = 0) shows the same color intensity as a green block (i.e., R = 0, G = 255, B = 0). As a result, we raise w_{c_2} and w_{c_3} to about twice the setting for w_{c_1}.

4 Results

4.1 Performance issues

This algorithm has been implemented by embedding it within the IBM QBIC multimedia database system. The discrete fast wavelet transforms are performed on IBM RS/6000 workstations. To compute the feature vectors

[3] Here we use standard MATLAB notation. That is, ‘ . * ’ denotes component-wise product

for the 10 000 color images in our database requires approximately 2 hours of CPU time.

The matching speed is very fast. Using a SUN Sparc-20 workstation, a fully-specified query takes about 3.3 seconds of response time with 1.8 seconds of CPU time to select the best 100 matching images from the 10 000 image database using our similarity measure. It takes about twice the time to answer a partially specified query.

There are many ways to further speed up the system for very large image databases. For example, we may presort and store the standard deviation information within the feature vectors of the images in the database because we must compare this information for each query. Also, we may use a better algorithm to find the first k matching images if k is smaller than $\log_2(n)$ if the database contains n images. In fact, an algorithm of execution time of $O(kn)$ can be constructed for this task to replace the quick-sort algorithm with run time $O(n\log(n))$ we are currently using.

Figures 8–11 show accuracy comparisons of our wavelet algorithm with the color layout algorithms in IBM QBIC and Virage, two of the most popular commercial multimedia databases, and the system developed by University of Washington. Figures 13–15 show the query results obtained from partial sketch image queries. Default parameters are used for the University of Washington's algorithm. In all cases, the number of reasonably similar images retrieved by our algorithm within the best matches is higher. In our comparisons of query results, we consider one retrieval algorithm as better than another if the number of similar images among a fixed number of best matching images is higher. We do not attempt to compare two images which are both very similar to a query image because we do not have a quantitative measure of the similarity between two images. When several images are all very close to the query image, it is meaningless to rank their similarities to the query image since subjective opinions often dominate and the distances are too close to make ranking orders simply based on sorting results. For example, in Fig. 9, the second and third images retrieved by our algorithm are both very close to the first image (the query image). Some people may favor the second one because the color of the boat is the same as that of the boat in the query image; on the other hand, some may favor the third one since its view is broader in vertical direction, just as the query image.

With our resources, it is impossible for us to compare and quantify the accuracy of the algorithms on all 10 000 query images in the image database. In general, our wavelet-based algorithm outperforms the above mentioned algorithms by returning more semantically-meaningful images in the set of best matching images, especially when the image contains large local color variations.

4.2 Limitations of the search

WBIIS is designed to be invariant to scale and aspect ratio changes since query images and all images in the database are normalized to the same size and aspect ratio before the matching step. Color and intensity shift can be handled by the alternative measure discussed at the end of Sect. 3.3.

The WBIIS system is designed to handle color layout type queries. Because of the nature of color layout search, WBIIS has limitations in certain types of applications when high degrees of rotation and translation invariance are important. However, WBIIS can handle small amount of rotation and translation changes. In the searching phase, a global measure, i.e., the set of the standard deviations of the saved wavelet coefficients, is utilized to measure the image coherence. The multi-scale indexing scheme is also used to avoid bias. Experiments have shown that WBIIS with Daubechies' wavelets is well capable of handling a maximum rotation of 20 degrees and a maximum translation around 20% in general. In Fig. 9, for instance, the system successfully finds images with wind surfers in various parts, many of which differ a lot from that of the query. Similar situation can be found in Figs. 10, 11 and 14. The system is more sensitive to rotation and translation changes when performing partial sketch search with large non-specified areas. Currently the system cannot handle queries based on subregions.

5 Conclusions and future work

In this paper, we have explored some alternatives for improving both the speed and accuracy of traditional color layout image indexing algorithms used in large multimedia database systems. An efficient wavelet-based multi-scale indexing and matching system using Daubechies' wavelets developed by us has been demonstrated.

It is possible to improve the searching accuracy by fine-tuning the algorithm, e.g., using a perceptually-comparable color space and adjusting weights for different wavelet coefficients when computing the distance between two images. Sensitivity to rotation and translation changes can be further reduced by introducing a more sophisticated matching metric. It is also possible to make the searching faster by developing a better algorithm for storing and matching the feature vectors. We are also working on shape-based image indexing and searching algorithms using only the high-pass wavelet filters. Experiments with our algorithm on a video database system could be another interesting study.

Finally, we are working on applying this technique to different types of image databases such as medical images and scanned art images. We are actively working with the Stanford University Libraries to integrate our image querying system into digital library systems such as the MediaWeaver system [19]. Our wavelet-based image search engine is currently being used at the Stanford University Library to assist teaching and research projects in liberal art departments.

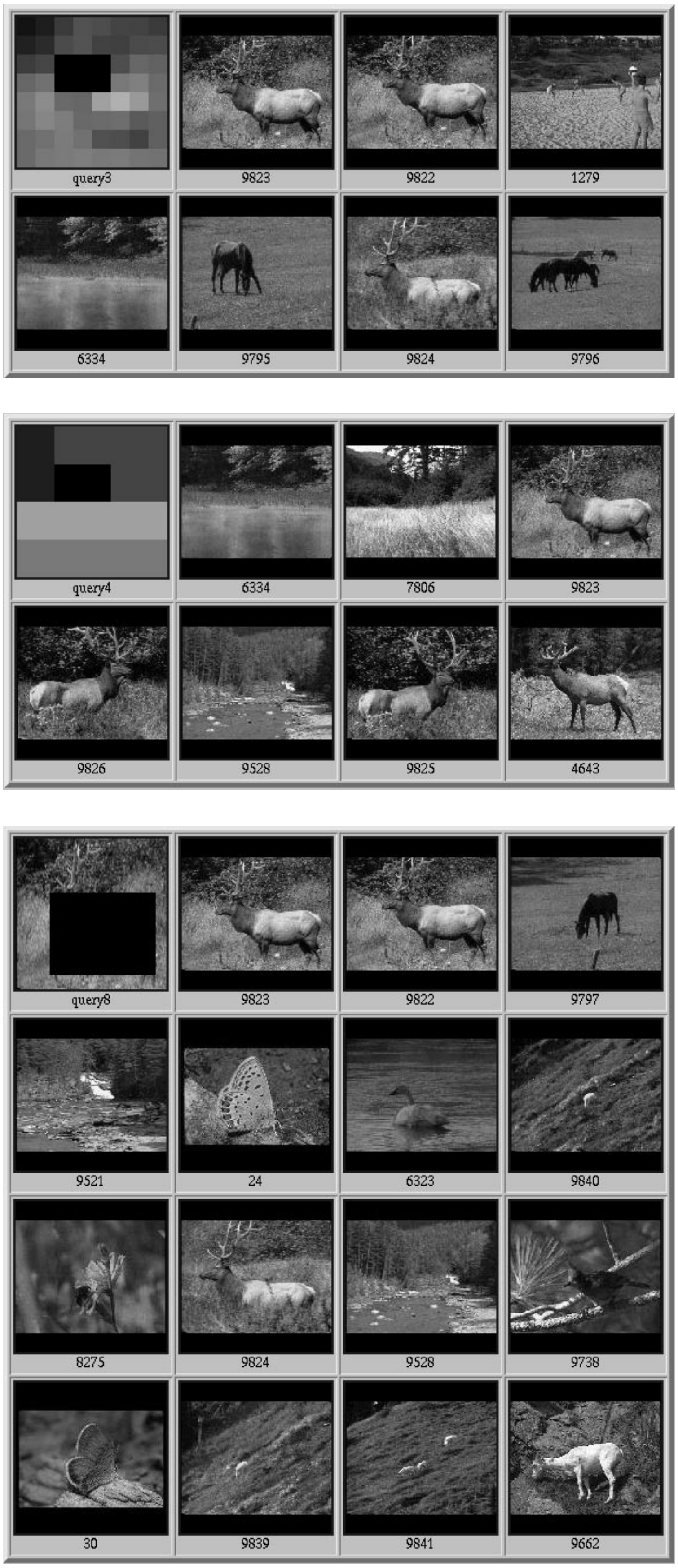


Fig. 13. Partial sketch queries in different resolutions. The upper-left corner image in each block of images is the query. Black areas in a query image represent non-specified areas. Results were obtained from a database of approximately 10 000 images

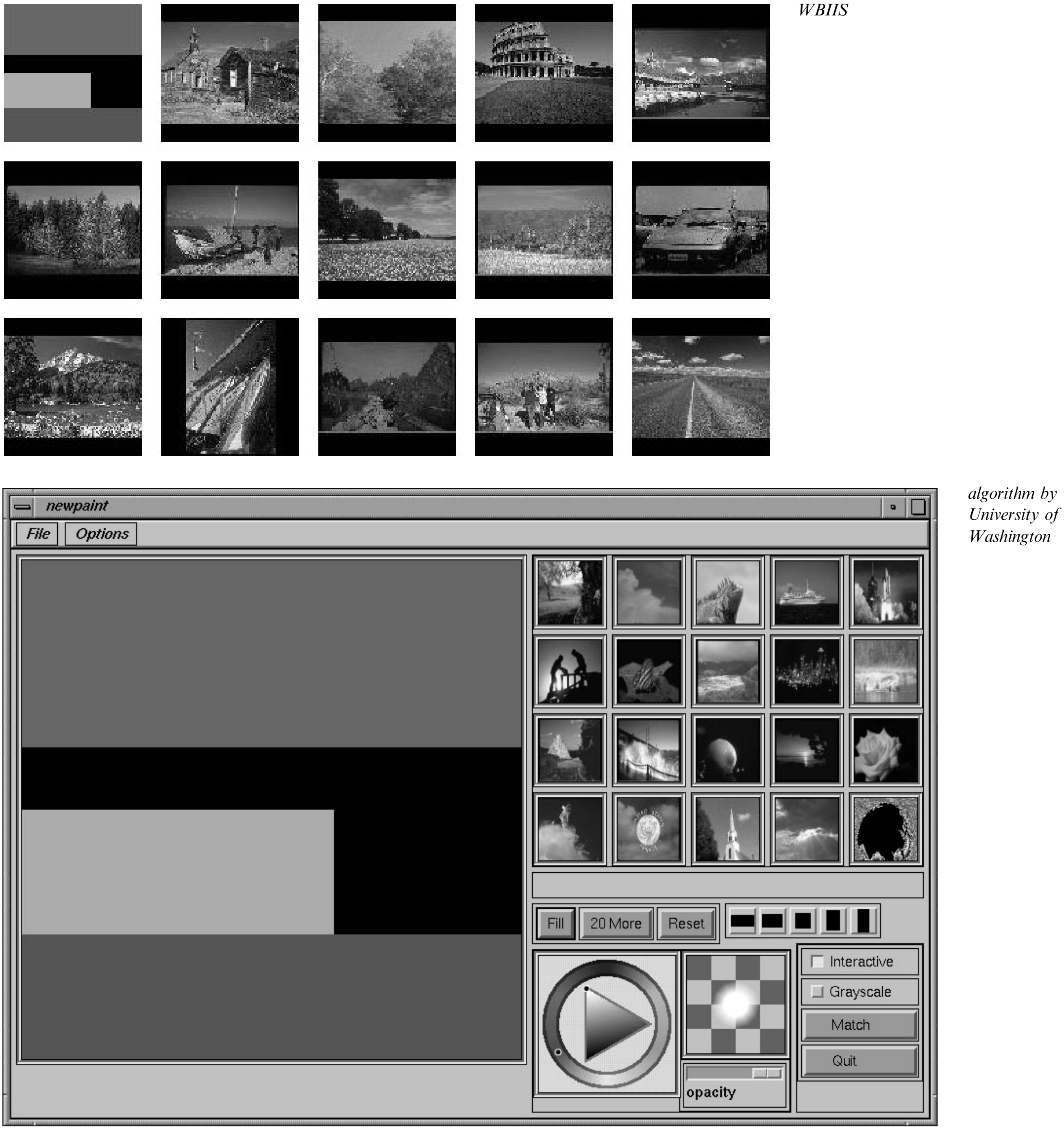


Fig. 14. Query results on a hand-drawn query image. Black areas in a query image represent non-specified areas. Equivalent query for the two systems. Results were obtained from a database of approximately 10 000 images

Acknowledgement. The project could not be completed without the valuable discussions with Xiaoming Huo of Stanford University Statistics Department, Tom Minka of MIT Media Lab Photobook group, various researchers including Dragutin Petkovic and Wayne Niblack in the QBIC group at the IBM Almaden Research Center, Tong Zhang of Stanford University Computer Science Department, Jia Li of Stanford University Information Systems Laboratory, and Ying Liang of Stanford University Electrical Engineering Department. The authors would like to thank IBM for allowing us to use the QBIC system, and D. H. Salesin's group for allowing us to use the system developed by the University of Washington, for comparing our wavelet-based algorithm with their existing indexing and searching algorithms. The detailed comments of anonymous reviewers were also most useful to help improve the paper's presentation. Likewise, we should mention the use of the WaveLab package from the Stanford University Statistics Department at an earlier development stage of the project.

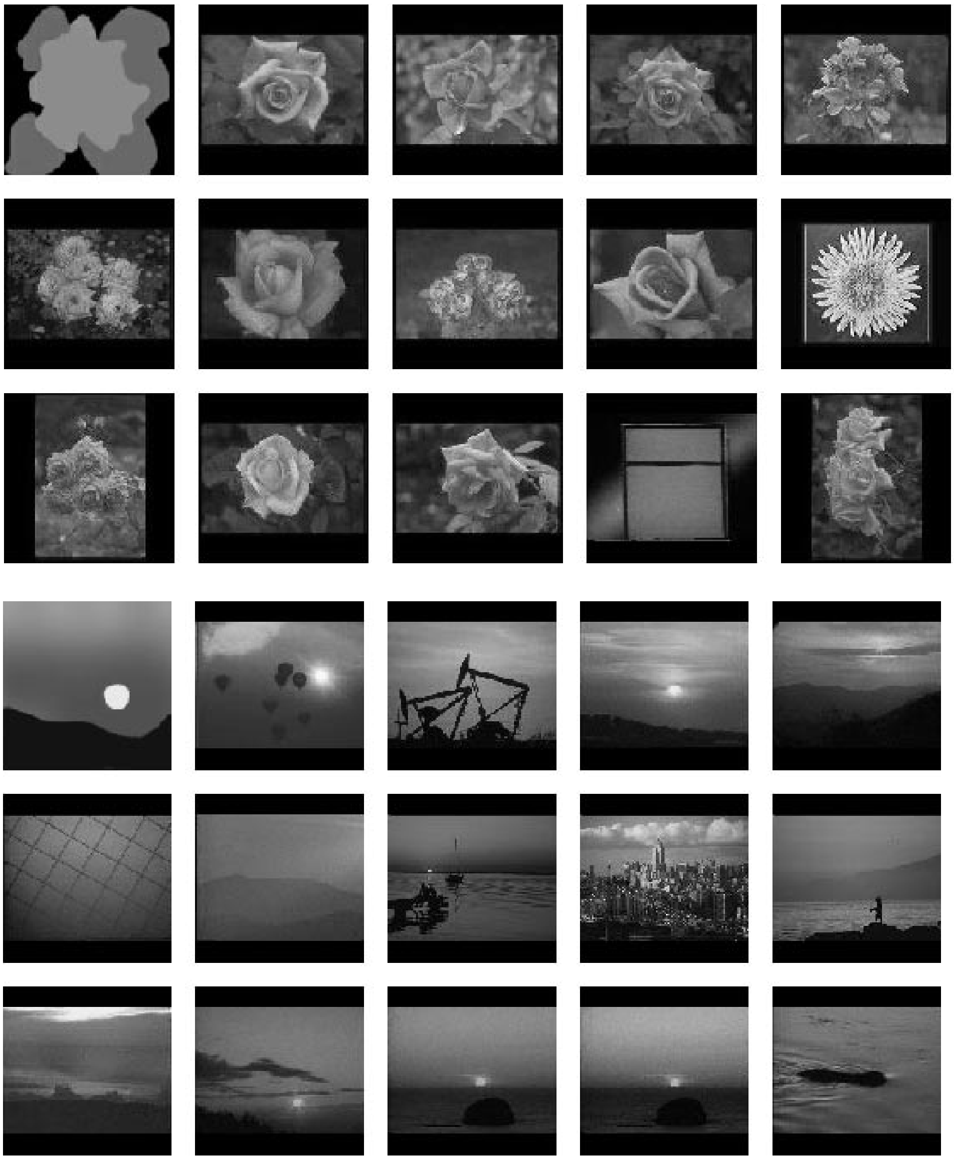


Fig. 15. Two other query examples using WBIIS. The upper-left corner image in each block of images is the query

References

1. Dana H. Ballard, Christopher M. Brown, *Computer Vision*, Prentice-Hall, Englewood Cliffs, NJ, 1982
2. Charles K. Chui, *An Introduction to Wavelets*, Academic Press, San Diego, CA, 1992
3. Charles K. Chui, *Wavelets: A Tutorial in Theory and Applications*, Academic Press, San Diego, CA, 1992
4. Ruel V. Churchill, *Fourier Series and Boundary Value Problems*, McGraw-Hill, New York, 1987
5. Ingrid Daubechies, Orthonormal bases of compactly supported wavelets, *Communications on Pure and Applied Mathematics*, 41(7):909–996, 1988
6. Ingrid Daubechies, *Ten Lectures on Wavelets*, CBMS-NSF Regional Conference Series in Applied Mathematics, 1992
7. C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic and W. Equitz, Efficient and Effective Querying by Image Content, *J. of Intelligent Information Systems*, 3:231–262, 1994
8. Gerald B. Folland, *Fourier Analysis and Its Applications*, Pacific Grove, CA, 1992
9. Robert M. Gray, Joseph, W. Goodman, *Fourier Transforms: An Introduction for Engineers*, Kluwer Academic Publishers, 1995
10. Amarnath Gupta and Ramesh Jain, Visual Information Retrieval, *Coms. ACM*, 40(5) pp. 69–79, 1997
11. L. M. Hurvich and D. Jameson, An Opponent-process Theory of Color Vision, *Psychological Review*, 64, 384–390, 1957
12. C. E. Jacobs, A. Finkelstein, D. H. Salesin, Fast Multiresolution Image Querying, *Proceedings of SIGGAPH 95*, *in Computer Graphics Proceedings*, *Annual Conference Series*, pp. 277–286, August 1995
13. Gerald Kaiser, *A Friendly Guide to Wavelets*, Birkhauser, Boston, 1994
14. Yves Meyer, *Wavelets: Algorithms & Applications*, SIAM, Philadelphia, 1993
15. W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos and G. Taubin. The QBIC project: Query image by content using color, texture and shape, SPIE *Storage and Retrieval for Image and Video Databases*, pp. 173–187, San Jose, CA, 1993
16. R. W. Picard, T. Kabir, Finding Similar Patterns in Large Image Databases, *IEEE ICASSP*, Minneapolis, Vol. V, pp. 161–164, 1993

17. A. Pentland, R. W. Picard, S. Sclaroff, Photobook: Content-Based Manipulation of Image Databases, *SPIE Storage and Retrieval Image and Video Databases II*, San Jose, CA, 1995
18. A. Said, W. A. Pearlman, An Image Multiresolution Representation for Lossless and Lossy Image Compression, *IEEE Trans. Image Processing*, Sept. 1996
19. Sha Xin Wei, MediaWeaver – A Distributed Media Authoring System for Networked Scholarly Workspaces, special issue on *Multimedia Tools and applications*, *Multimedia Systems Journal*, ACM, 1996
20. M.J. Swain, D. H. Ballard, Color Indexing, *Int. Journal of Computer Vision*, 7(1):11–32, 1991
21. Gio Wiederhold, Digital Libraries, Value, and Productivity, *Com. ACM*, 38(4), pp. 85–96, 1995
22. Special Issue on Wavelets and Signal Processing, *IEEE Trans. Signal Processing*, Vol. 41, Dec. 1993