Image and Vision Computing 27 (2009) 713–724

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Query by low-quality image

Mohammad Faizal Ahmad Fauzi [a,*], Paul H. Lewis [b]

[a] Faculty of Engineering, Multimedia University, Jalan Multimedia, 63100 Cyberjaya, Selangor, Malaysia
[b] School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, UK

ARTICLE INFO

Article history:
Received 29 October 2005
Received in revised form 21 July 2008
Accepted 25 July 2008

Keywords:
Content-based image retrieval
Low-quality image analysis
Wavelet transform

ABSTRACT

The motivation for research on low-quality images comes from a requirement by some museums to respond to queries for pictorial information, submitted in the form of fax messages or other low-quality monochrome images of works of art. The museums have databases of high-resolution images of their artefact collections and the person submitting the query is asking typically whether the museum holds the artwork shown or perhaps some similar work. Often the query image will have no associated metadata and will be produced from a low-resolution picture of the original artwork. The resulting poor quality image, received by the museum, leads to very poor retrieval accuracy when the fax is used in standard query by example searches using, for example, colour, spatial colour or texture matching algorithms. We propose a special retrieval algorithm in order to improve the retrieval accuracy in query by low-quality image application and evaluate it in comparison with more conventional algorithms. Throughout this paper, fax images will be used as the main source of low-quality image for query by low-quality image experiments. Nonetheless, some other forms of low-quality image will also be considered.

1. Introduction

Content-based image retrieval (CBIR) [1,2] is a retrieval system where images are indexed based on their own visual content instead of the traditional text-based keyword or metadata information. Colour, texture and shape are the most used features for the extraction of information in CBIR. Most published work on image retrieval assumes a reasonable quality query image, captured digitally from camera or video, or obtained from scanning a good quality image [3–7]. There are actually other image acquisition approaches which can provide query images, but with a rather poor quality; a fax machine is a very good example. This section briefly explains the idea of *query by low-quality image* and why it is necessary.

The motivation for research on low-quality images comes from a requirement by some museums to respond to queries for pictorial information, submitted in the form of fax messages or other low-quality monochrome images of works of art. The museums have databases of high-resolution images of their artefact collections and the person submitting the query is asking typically whether the museum holds the artwork shown or perhaps some similar work. The query image may have no associated metadata and may be produced from a low-resolution picture of the original artwork. The resulting image, received by the museum, leads to very poor retrieval accuracy when used in standard *query by example* searches using, for example, colour or texture matching algorithms. Some examples of genuine fax images received by the museum together with the corresponding high-resolution images they represent are shown in Figs. 5 and 6. It is clear that fax images are of very low quality and even the best quality fax machine cannot prevent some loss of information in the image produced. If this image is to be used as a query for retrieval, special algorithms need to be developed in order to improve the retrieval accuracy.

With the advance of imaging and communication technology these days, one might wonder if anyone would still use fax machines as a medium to transfer images. The answer is yes. There are museums which hold large databases of images that frequently receive fax images from around the world asking if they have the artworks shown. They have problems in fulfilling requests as a result of poor retrieval results using standard *query by example* searches. Moreover, even images captured by camera or scanner sometimes carry noise, although they are not as bad as in fax images. Therefore this specific area of CBIR needs to be explored more comprehensively. Our proposed algorithm is based on the wavelet transform and binary image matching and experiments show that it is suitable for a wide range of low-quality image retrieval applications.

This paper is organized as follows. The following section briefly describes the analysis of low-quality images and previous work concerning its use in image retrieval. Then a brief description of the wavelet transform follows. A novel algorithm for coping with low-quality query images is explained in Section 4, with a few

* Corresponding author. Tel.: +60 3 8312 5420; fax: +60 3 8318 3029.
E-mail addresses: faizal1@mmu.edu.my (M.F. Ahmad Fauzi), phl@ecs.soton.ac.uk (P.H. Lewis).

0262-8856/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2008.07.007

available methods for comparison presented in Section 5. Section 6 presents all experimental results and finally conclusions and future work are presented in Section 7.

2. Analysis of low-quality images

This section begins with an analysis of fax images and their properties. This is followed by a description of several other types of low-quality images which may be used as a query in CBIR applications.

2.1. Fax images

The fax machine is one of our most important communication machines. In almost any office, big or small, you will find a fax machine. Connected to a normal phone line, a fax machine allows the transmission of a document image to someone else instantly. Even with the popularity of e-mail, it is almost impossible to do business without the ability to use fax. There are many factors that can degrade the quality of the received document from that of the original. Most fax machines convert colour documents to monochrome documents, and certain colours such as yellow, greenish yellow and light blue are not well recognised by the machine. Also, if either fax machine is less than clean, the document may be smudged or have black lines though it.

The resolution and contrast setting of the machine is also important. For most machines, several resolution settings can be chosen such as super fine, fine, standard and half tone. Higher resolution gives better quality but takes a longer time to transmit the document, while lower resolutions have the opposite effect. The contrast setting also tend to be set high since that will improve the quality of a transferred text document. But for an image document, the received image will tend to be binary-like. Since fax machines operate using telephone lines, noise and distortion can also occur during transmission. Finally, the printing quality of the receiving fax machine will also contribute to the quality of the received document. Some fax machines also use special paper for printing, which tends to turn yellow or brown after a period.

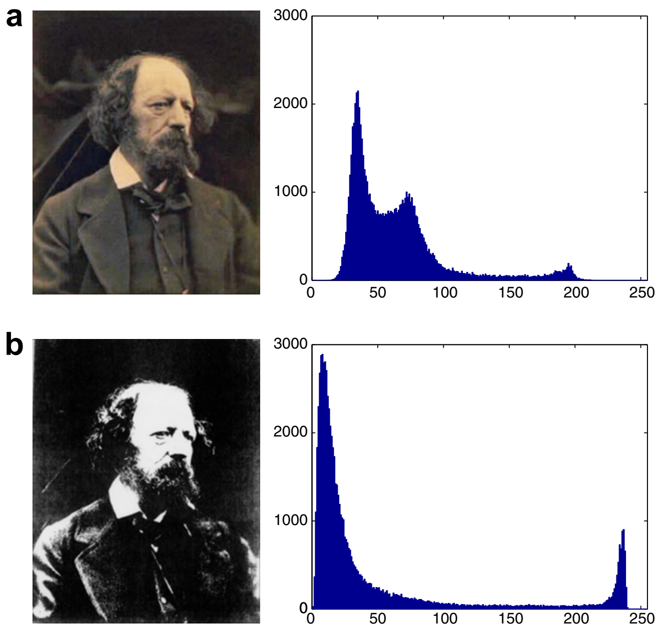

Fig. 1. Difference in histogram distribution between an image and its corresponding fax version. (a) An image and its grey-level histogram. (b) A fax version of the image and its grey-level histogram.

All of these factors may contribute to the poor quality of fax images. Fig. 1 shows an original image and a corresponding fax together with their intensity histograms. From the histogram, we can see that the pixel distribution of the fax image is pushed towards both ends of the histogram. If the distribution is pushed further towards the two extremes the result is essentially binary in nature. The histogram also suggests that no image enhancement or image restoration algorithms could be used to restore the fax image to the original. Therefore if the fax image is to be used as a query for CBIR, we have to work with the inadequate information. However, the fact that the fax image is almost binary in nature may be exploited for CBIR applications.

2.2. Other low-quality image examples

Besides fax images, there are other forms of low-quality image. This includes images of inappropriate contrast and brightness, highly-compressed images, low-resolution images, quantized images, as well as noisy images. Each of these forms of low-quality image will be considered briefly and used as queries for the query by low-quality image algorithms discussed in the following section.

2.2.1. Images of inappropriate brightness and contrast

There are many image acquisition devices available on the market. One of the most popular is the digital camera. But even with a high quality digital camera we can still get a rather poor quality image. For example, an image can be overexposed or underexposed and the resulting images can be either too bright or too dark [8,9]. The over and underexposed images are also difficult to enhance. When an image is overexposed, its pixels tend to skew toward the high end of the histogram, and simple histogram modification is sometimes inadequate to recover the image. The same can be said about underexposed images. An image can also be of low or high contrast. A low contrast image will have a very narrow histogram, while a high contrast image will have pixels clustered at both the low and high end of the histogram, as for fax images.

2.2.2. Highly compressed images

In order to save storage space, most digital images are compressed when stored in a computer. The compression technique is often lossy for standard image collections [8,10]. This may be achieved by eliminating data that is visually unnecessary and by taking advantage of the redundancy that is inherent in most images. Standard lossy compression techniques can achieve up to 50 times reduction in storage space for still images. However there is a tradeoff in achieving such a high compression ratio. Higher compression means more data being removed and hence leads to a drop in image quality. When the compression ratio is too high, a significant degradation in image quality is observed. This reduction in quality will be crucial when the image is used as a query in CBIR applications.

2.2.3. Low-resolution images

Images can have a wide range of resolutions [9]. The higher the resolution, the larger the potential information carried by the image. In the case of a low-resolution image, if the resolution is too low, there might not be enough information in the image. The number of pixels determines the resolution of an image, and it is a subjective issue to decide how many pixels are considered to be low resolution. If these images are used as the query in a CBIR system, problems might arise due to the inadequate information in the image. Various interpolation algorithms are available in order to increase the resolution of the image, but interpolation algorithms do not increase the information in the image. No matter

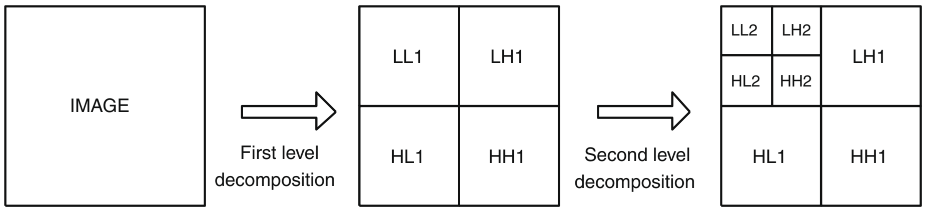

Fig. 2. An illustration of a 2-level wavelet decomposition of an image.

how good the algorithm, it is impossible to recreate data reliably that is non-existent in the first place.

2.2.4. Quantized images

The standard number of grey level values used for monochrome images is 256 as studies have shown that the human visual system can differentiate no more than 256 grey levels [9]. However, in some applications, images are quantized to some smaller number of grey levels [8]. The smallest number of grey levels for an image is 2, which is the case for binary images. In between these values, users can always set their own number of grey levels for an image, usually a number which is a power of 2. However, lower grey values mean a reduction in information that can be carried in the image. Using these kinds of images in CBIR applications can also raise a problem because of the lack of information, as in the case for low resolution images.

2.2.5. Noisy images

Images can sometimes be accompanied by noise. Noise is any undesired information that contaminates an image [8,9]. Noise appears in images from a variety of sources. The digital image acquisition process, which converts an optical image into a continuous electrical signal that is then sampled, is the primary process by which noise appears in digital images. At every step in the process there are fluctuations caused by natural phenomena that add a random value to the exact brightness value for a given pixel. The signal-to-noise ratio determines the amount of noise in an image. The lower the signal-to-noise ratio, the poorer the quality of the image.

2.3. Previous work on query by low-quality image

As far as CBIR is concerned, there is no comprehensive work on query by low-quality image applications. In most CBIR systems, if a low-quality image is encountered, either as query or database image, they usually enhance the particular image first before processing the information in the image, for example by noise reduction [11–13] or histogram manipulation [14,15]. Although this may be a good approach in some cases, it is usually effective only on “not so low”-quality images. As we have seen previously, the images we are working on are particularly poor in quality, and enhancing the image will not increase the retrieval accuracy of the system. In the following sections a simple but effective method for dealing with *query by low-quality image* problems is proposed and evaluated.

3. The wavelet transform

Wavelets were invented to overcome the shortcomings of the Fourier Transform. The fundamental idea is not only to analyse according to frequency but also according to scale, which is not covered by the Fourier transform. Most standard wavelets are based on the dilations and translations of one mother wavelet, ψ, which is a function with some special properties.

In image processing terms, the dilations and translations of the mother functions ψ are given by the wavelet basis function

$$\psi_{(s,l)}(t) = 2^{-s/2}\psi(2^{-s}x - l).$$

The variables s and l are integers that dilate and orientate the mother function ψ to generate a family of wavelets, such as the Daubechies wavelet family [16,17]. The scale index s indicates the wavelet’s width, and the location index l gives its position in 2D.

Wavelets can be divided into orthogonal and non-orthogonal. They are orthogonal if their basis functions are mutually orthogonal. Because of the orthogonality, there exist a series of coefficients that can represent the wavelet decomposition for the whole family of a particular wavelet. The series of coefficients is named the quadrature mirror filter (QMF). This concept, along with the theory of filter banks, is the basis for producing the famous fast pyramidal wavelet transform (PWT) decomposition proposed by Mallat [18].

A QMF is a filter that can be either low-pass or high-pass just by changing sign and rearranging its coefficient. In the 2D wavelet decomposition, the QMFs are used as both highpass and lowpass in both horizontal and vertical directions, followed by a 2 to 1 subsampling of each output image. This will generate four wavelet coefficient images, i.e., the *low–low*, *low–high*, *high–low* and *high–high* (*LL*, *LH*, *HL*, *HH*, respectively) subbands. The process is repeated on the *LL* channel, and the 2D PWT decomposition is constructed. Fig. 2 illustrates the PWT decomposition of an image up to two levels. For detailed information on wavelet theory, please refer to [19–21].

In this paper, the PWT wavelet coefficients are used to compute the feature vector of an image. The information obtained from this is of great importance as it shows the dominant frequency channels of a particular image and is particularly useful in similarity matching.

4. Query by low-quality image (QBLI) algorithm

Our *query by low-quality image* algorithm is based on the wavelet transform. There are several reasons for the use of wavelet transforms in the algorithm. First, the wavelet transform is of low computational intensity, which is an essential property in the proposed algorithm. Second, the wavelet transform is known to have very good discrimination when used in image classification, at least for texture retrieval [22–27]. Finally, the feature vector produced by the wavelet transform is compact, as we use only the mean energy of each wavelet channel as the feature. The number of channels of the wavelet transform can be controlled by adjusting the number of decomposition levels during the transform. The compact feature vector is also an essential property for using the proposed algorithm. These properties make the wavelet transform a suitable image analysis tool for our *query by low-quality image* application.

There are a number of varieties of wavelet transform. These include the standard or PWT, tree-structured wavelet transform (TWT), discrete wavelet frames (DWFs) and Gabor wavelet transform, among others. However, not all of these varieties of wavelet transform have the three properties mentioned above. In fact, the standard wavelet transform is the only technique that offers all three advantages. The TWT transform results in a much higher number of channels, and is quite complex compared to the standard wavelet transform. Moreover, for a non-textured image, the frequency content of an image is concentrated in the low-frequency region, thus image decomposition is needed just for the low-frequency band. The DWFs technique, although it has the same number of channels, is more computationally intensive than the standard wavelet transform. The Gabor wavelet and all other continuous wavelet transforms have an even higher computational complexity. Therefore, the PWT will be used in the proposed QBLI algorithm to compute the feature vector of an image.

Since the quality of the query image is so low that it differs substantially from its original, applying the PWT to the good images in the database will not produce a feature vector close enough to the feature vector of the query. To avoid this problem, we exploit the fact that the low-quality image is almost binary in nature. The low-quality query image is first converted to an actual binary image, before the PWT is applied. Since the low-quality images by themselves are almost binary, the binarisation process does not really reduce too much information from the image. In fact most of the removed information are actually distorted contents that prevent correct matching between the low-quality images with their corresponding high quality versions. Binarisation paved the way for a better content-based matching between them.

A similar conversion to binary is applied to each of the database images, choosing a threshold which makes them as close as possible to the binary of the query image, before the PWT is applied. Hence, during matching, the comparison is actually made between the binary versions of both the query image and the database images. Using an appropriate threshold, the binary image of the query should be similar to the binary image of its original. If two binary images are almost the same, their feature vectors should also be similar, hence resulting in correct retrieval of the original image. One simple approach to compute the threshold value for binary conversion of the database images is to use the percentage of black or white pixels of the query binary. For example, if after binarisation, the percentage of white pixels of the query image is x, then all the database images can be converted to binary in such a way that the percentage of white pixels is also x.

At first, this method may seem to be unsuitable for use with an interactive retrieval system, since in order to compute the feature vector of the database images, the percentage of white pixels is required and that depends on the query. It is highly ineffective to compute the feature vectors of the database images during retrieval as it makes the response time large for a large database. An effective retrieval system computes the feature vectors of the database images in advance, so that only the matching process is needed during retrieval process. However, due to the compact nature of the wavelet signatures, an algorithm suitable for interactive retrieval can be proposed. The algorithm consists of two steps; *binary image thresholding* and *feature vector computation and comparison*, and these are explained below.

4.1. Binary image thresholding

As stated earlier, since the query images are almost binary, it is better to compute feature vectors in the binary domain. The query image can be converted to binary by thresholding in the middle of the grey scale range covered by the image. In order for the feature vector of a database image to be compared fairly with the feature vector from the query, the database image must also be converted to binary. But the choice of threshold is not immediately obvious. For the original database image corresponding to the query, an appropriate threshold is probably one that produces a binary image with the same percentage of black (or white) pixels as the binary form of the query image. We could use this percentage for all the database images, but it varies from query to query. How can the percentage be matched if we are to precompute the feature vectors from the binary versions of all the database images? Note that since the query image is the target and already effectively binary, it is the original database image that must be made as close as possible to the binary query and not vice versa.

One way to solve the problem is to convert each database image into a set of different binary images corresponding to different percentages of black pixels from 0 to 100%. If sufficient binaries are created, the binary query image will then be very similar to one of these binaries for the original image. We propose that 99 binaries are created for each database image corresponding to percentages of black pixels from 1 to 99 in steps of 1% (note that binaries of 0% and 100% are, respectively, black and white images, and thus are not needed). One might argue that 99 binaries for the database images might be too much or too little. In the experimental section, we discuss the optimum number of binaries to be created for each database image. At the moment, for the sake of explanation and as an initial estimate, 99 binaries representing 99 threshold percentages are used.

However, the binaries do not need to be stored. Calculating the feature vectors for the database involves calculating the PWT for each of the binary images for each image in the database. This is implementable since the PWT is a fast algorithm and, more importantly, the feature vectors for each binary image have only a relatively small number of coefficients. There will be 99 sets of feature vectors for each database image, but during matching, once the percentage of pixels of the query is known, only one of these 99 sets will be used for comparison, namely the set associated with the same pixel percentage as the binary query image. Figs. 3 and

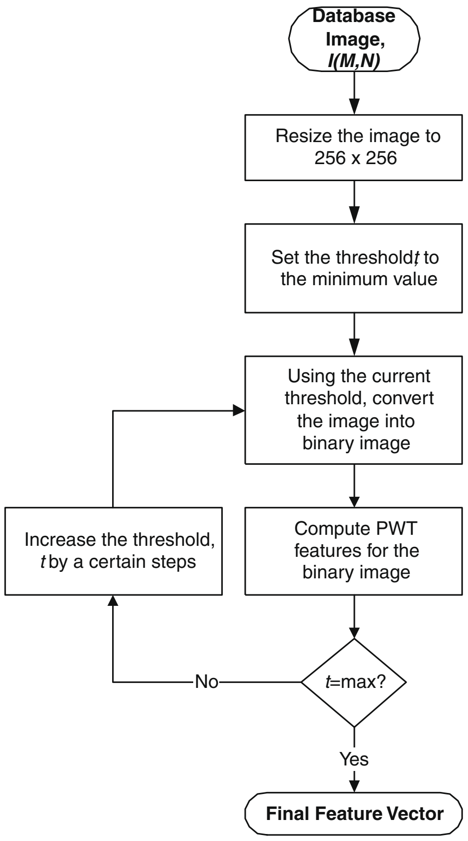
Database Image, I(M,N)
Resize the image to 256 x 256
Set the threshold t to the minimum value
Using the current threshold, convert the image into binary image
Compute PWT features for the binary image
Increase the threshold, t by a certain steps
t=max?
No
Yes
Final Feature Vector

Fig. 3. Flowchart of the proposed algorithm for database feature extraction stage.

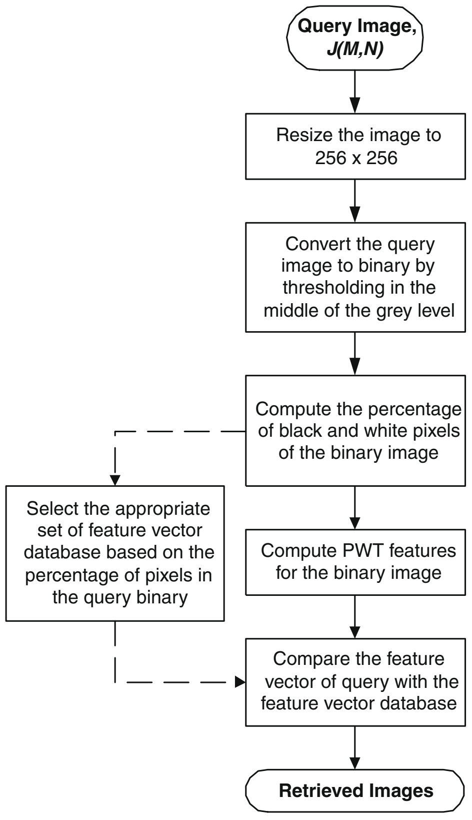
Query Image, J(M,N)
Resize the image to 256 x 256
Convert the query image to binary by thresholding in the middle of the grey level
Compute the percentage of black and white pixels of the binary image
Select the appropriate set of feature vector database based on the percentage of pixels in the query binary
Compute PWT features for the binary image
Compare the feature vector of query with the feature vector database
Retrieved Images

Fig. 4. Flowchart of the proposed algorithm for interactive retrieval stage.

4 show the flowchart of the proposed algorithm for the feature extraction stage and the retrieval stage, respectively.

Some might argue that the proposed binary image thresholding technique will have problems if there exist many images with similar binary layout in the database. However, the way the binarisation process is carried out, it is quite unlikely for two or more database images with totally different content, even after thresholding to the same pixel percentage, to end up having similar binary layout. Even if the binary layouts happen to be similar, the similarity is exclusively for that particular pixel percentage only, with minimal effect on the overall system performance. If two or more images share similar binary layouts for several different pixel percentages, the images will most probably share the same image content or structure as well.

4.2. Feature vector computation and comparison

As described earlier, the PWT is used as a tool for feature vector extraction. Since the PWT is usually applied on dyadic square images, the binary images are all resized to 256×256. Other image sizes can also be used although bigger sizes will increase the computational load while smaller sizes will reduce the information within the image. The 256×256 size is a suitable compromise. For simplicity, nearest neighbour interpolation and decimation is used for the resizing process. The resizing can also be done before the binary conversion. The PWT algorithm is applied and the image is decomposed into four sub-images (*LL*, *LH*, *HL* and *HH*). The *LL* band is decomposed further until the desired number of decomposition levels is achieved. For a start, we set the smallest sub-images (sub-images corresponding to the highest level) are of size 4×4, which means six levels of decomposition for an image of size 256×256. This results in 19 different sub-images or sub-bands.

Once the wavelet coefficients of a binary image are available, features are computed from each sub-band, resulting in 19 features for each binary image. The mean μ is the energy measure used as the features. Let the image sub-band be $W_{mn}(x,y)$ where mn denotes the specific sub-band, m is the decomposition level and $n = 1,2,3,4$ indicates the *LL*, *LH*, *HL*, *HH* bands, respectively, then μ_{mn} is calculated by

$$\mu_{mn} = \frac{1}{N_{mn}^2}\sum_y \sum_x |W_{mn}(x,y)|, \tag{1}$$

where N is the length of a particular sub-band mn. There are several other energy measures that can be used as the feature such as the standard deviation, the number of zero-crossings etc., but in order to keep the algorithm simple, only the mean energy is used. Incorporating any other energy measure with the mean energy will increase the total number of features for a particular database image quite drastically, and hence is not very suitable for the proposed algorithm. The feature vector f for a particular binary image is therefore

$$\begin{aligned} f &= [\mu_{mn}],\ n \neq 1 \text{ except for the coarsest level, i.e., } m = 6, \\ &= [\mu_{12}, \mu_{13}, \mu_{14}, \mu_{22}, \ldots, \mu_{61}, \mu_{62}, \mu_{63}, \mu_{64}]. \end{aligned} \tag{2}$$

The feature vectors for the database images will have $99 \times 19 = 1881$ coefficients, although only 19 will be used for comparison in each retrieval task. During matching, a distance classifier is used to compute the similarity between the query and each database image. There are many possible distance classifiers and the performance for this particular algorithm is investigated later. Initially, the widely used minimum Euclidean distance is utilised. The minimum Euclidean distance between two features i and j is given by

$$d_{\mathrm{Euclidean}}(i,j) = \sum_m \sum_n \left[\mu_{mn}^{(i)} - \mu_{mn}^{(j)}\right]^2. \tag{3}$$

The square root is omitted for computational efficiency as it does not affect the order of similarity. Once the distances are computed, the images will be retrieved in order of increasing distance from the query image.

5. Available methods for comparison

As previously mentioned, there is little published research specifically on query by low-quality images available in the literature. Thus, it is difficult to compare the performance of the proposed algorithm with other published algorithms. However, in this section we describe four techniques which will be compared against the proposed algorithm. One of the techniques is a simple non-real-time pixel matching algorithm which we develop simply for the sake of comparison with the proposed QBLI method. The second technique is to simply use the wavelet transform to extract features from the low-quality image without any binarisation processes. This will show the benefit of the inclusion of the binarisation stage in solving the particular retrieval problem. The final two methods are shape-based methods in the forms of Hu moments and Fourier descriptor, where the effect of shape distortion in the low quality images will be investigated.

5.1. Pixel matching algorithm

The pixel matching algorithm is almost identical to the proposed QBLI algorithm except that instead of using PWT coefficients of binary images as features, pixel by pixel matching is used between the binary query and the database images. It is intended to show the effect of using PWT coefficients as features in the QBLI algorithm. For a particular database image, 99 binaries are created and stored as features. During retrieval, the query is converted to binary and the percentage of black pixels is computed. A pixel by pixel matching is then performed between the query binary and the database binary corresponding to the percentage of black pixels. The number of matching pixels, as a percentage of the total number of pixels, is used as the similarity measure and the database images are then retrieved in decreasing order of similarity.

This method, however, uses a large amount of storage. A 256×256 image requires $(256 \times 256 \times 99)/8\ \text{bit} \approx 800\ \text{kB}$ of memory to store all the features. An alternative way to perform the pixel matching algorithm is by performing binary conversion during the retrieval process. The query image is converted to be totally binary, and the percentage of black pixels is computed. The database image to be compared to the query is then converted to binary in such a way that the percentage of its black pixels is the same as in the binary query image. Both binaries are re-sampled to the same size and a pixel by pixel comparison is made. By performing the pixel matching algorithm this way, there is no need for feature storage at all, although the computational load involved during interactive retrieval is very high.

Based on the description above, the pixel matching algorithm is not appropriate for interactive retrievals since it either involves a large storage or a very slow interactive retrieval. However, as will be shown later, this method does give a very high accuracy in retrieving images based on a low-quality query. This method is used as a yardstick for the evaluation of the proposed wavelet-based QBLI algorithm since it gives such good retrieval accuracy. In the experiment section, the second approach to pixel matching algorithm will be used because there is not much difference in computational speed with the increasing size of re-sampled images, as oppose to linear increase in storage space for the first approach.

5.2. Pyramidal wavelet transform

Since the proposed QBLI method uses the PWT as the feature extractor, it is reasonable that the PWT algorithm is compared to the new algorithm. By comparing it with the feature extractor itself, we can evaluate the true effectiveness of the binarisation stage in the QBLI. We can also observe what the retrieval accuracy would be if we are using standard feature extraction techniques without considering the high difference between the low-quality image and its original.

The procedure for the retrieval using the PWT is relatively straight forward. The entire database image is first resized to a dyadic square image. In this experiment, we use the same size as in the proposed QBLI method, which is 256×256. The pyramidal wavelet transform is applied to the database image, and 19 channels are produced. The mean energy (as in Eq. (1)) is computed from each channel. During the retrieval process, the query image is also resized to 256×256 and features are computed in the same way as for the database image. The similarity between the query and database image is computed using the function in Eq. (3), and the images are retrieved in order of increasing distance.

5.3. Shape-based techniques

To check whether shape information can be used to address the *query by low-quality image* problem, two shape-based techniques are used, namely the Hu moments and the Fourier descriptor. Hu moments describe the image content with respect to its axes based on seven moments that are rotation, scaling, translation and skew invariant. Fourier descriptor extract the shape information by applying Fourier transform to the boundary co-ordinates of an object expressed as complex numbers. The descriptors can be normalized to make them independent of translation, scale and rotation.

Before Hu moments and Fourier descriptor are applied, the images are first pre-processed to extract the dominant shape in the image. The pre-processing operations include edge detection using Canny operator, followed by morphological closing, filling and opening. Seven features are extracted from Hu moments, while 64 features are extracted from Fourier descriptor. As with the wavelet transform, the similarity between the query and database image is computed using the function in Eq. (3), except that the distance is normalized in the case of Hu moments. The images are then retrieved in order of increasing distance.

6. Experimental analysis

The database used in our experiment consists of 1062 images of various types and sizes, taken from the Victoria and Albert Museum collection. From the 1062 images, 20 are selected as target images in the experiments, and are shown in Fig. 5. As mentioned before, fax images are the primary sources for the low-quality images in this paper, hence genuine fax versions of the 20 selected images are used as queries for the retrieval experiments. The 20 fax images are shown in Fig. 6. The evaluation is based on the ability of the algorithm to retrieve the original image when the fax version of the original is used as the query. Experiments conducted include the evaluation of the effectiveness of the proposed algorithm, the performance using different distance classifiers, as well as different numbers of decomposition levels and wavelet bases, investigation of the optimum number of binaries to be created, and finally the performance of the algorithm on other types of low-quality images.

6.1. Evaluation of the algorithm

In this particular experiment, the number of binaries created for each database image is set to 99, representing 99 different percentages of black pixels. The wavelet basis used for wavelet transform decomposition is the Daubechies 8-tap wavelet. The distance metric used is the minimum Euclidean distance, and the number of decomposition levels is set to six, which means 19 features for each binary image. The results for the QBLI algorithm are shown in Table 1, where the retrieval position of the original image among the 1062 database images is shown. The table also shows performance comparison with the pixel matching algorithm, basic PWT, Hu moments and Fourier descriptor techniques.

It can be seen that the basic PWT *query by example* algorithm is particularly poor for fax queries, but the retrieval results obtained using the QBLI algorithm are very encouraging, and the most consistent among the five methods. All the original images are retrieved within the top 5. This is a good result considering the poor quality of the fax images and the reasonably large image database used. The result suggests the importance of the binarisation process in the algorithm. It also suggests that the distances between the fax images and their originals are very close and should still produce good results for a larger image database. It is also interesting to notice that some similar images to the target image were also retrieved as the top rank, which suggest the algorithm serves similarity retrieval quite well. The two shape-based algorithms also failed to produce reliable results, mainly because the shape information in some low-quality images is so badly distorted it directly and severely affects the retrieval accuracy.

Table 1 also shows the results using the pixel matching algorithm. The result was obtained using the second approach to pixel matching on the binaries with the re-sampled size set to 256×256. As expected, the pixel matching algorithm gives very

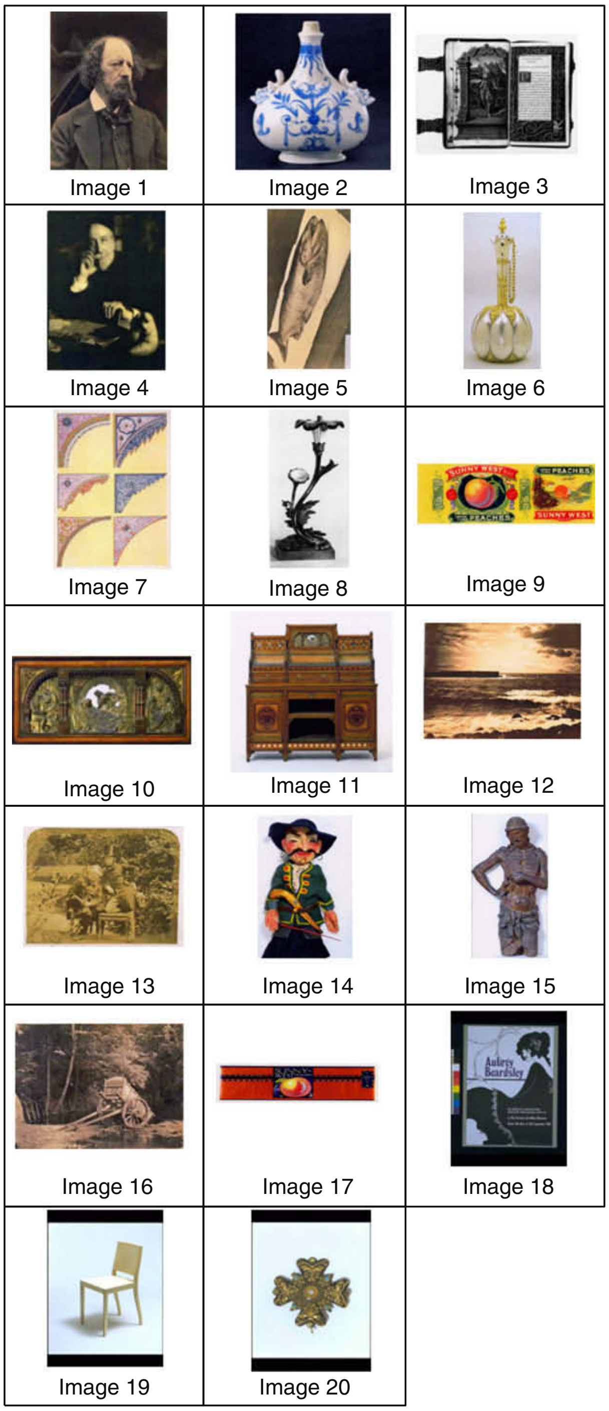

Fig. 5. 20 selected images from the database.

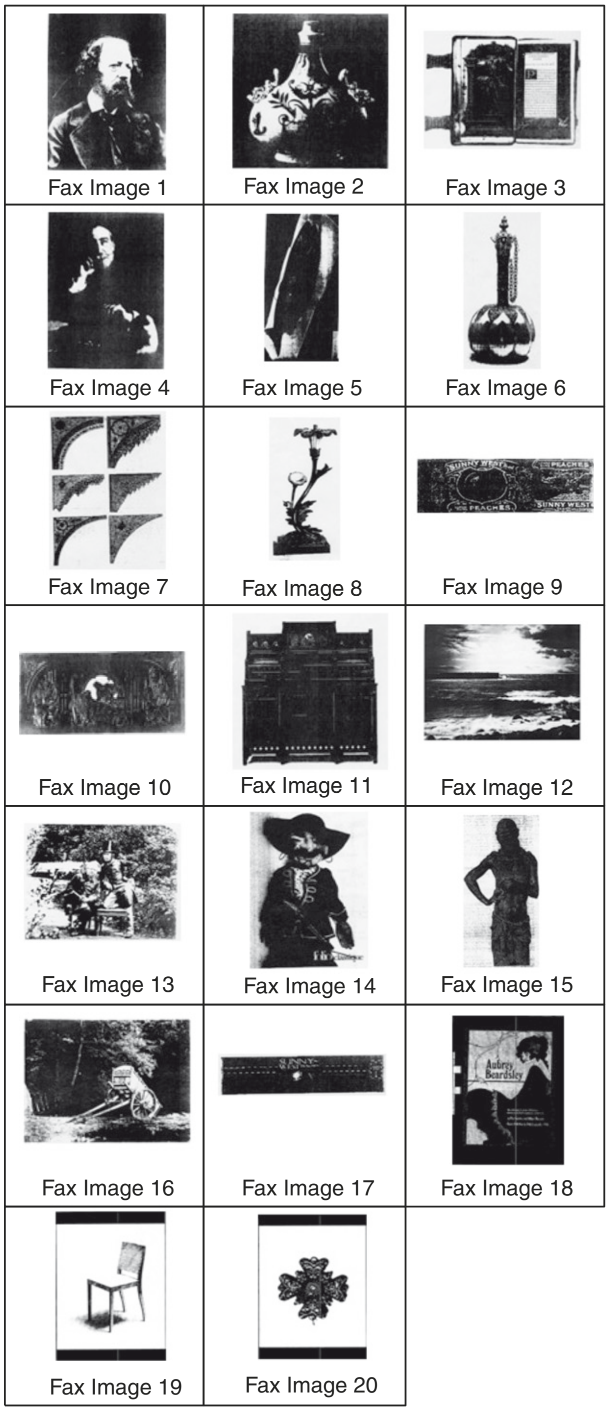

Fig. 6. 20 fax images corresponding to the images in Fig. 5.

good retrieval results. All the originals were returned as the first match, except for one case only, which fails because of the poor state around the edges of that particular query image, which leads to inaccurate computation of the percentage of black pixels. The result of the QBLI algorithm for that particular query is, however, very promising. This indirectly shows that the pixel matching algorithm, while giving excellent performance, is very sensitive to noise. The QBLI algorithm in comparison is much more robust. It was also observed that the size of the re-sampled image is quite critical for the pixel matching algorithm. The bigger the re-sampled image, the better the accuracy, with the best result observed when the re-sampled images are set to 256×256 or bigger.

In terms of the speed of the algorithm, the average time taken for retrieving images from the database of 1062 images using the proposed technique is about 1 second, which is the same as the speed using the basic PWT method, twice the speed of the shape-based algorithms, and more than 100 times faster than the pixel matching algorithm. The times were observed on a 700 MHz Xeon processor. From the accuracy and speed observation, it can be seen that the proposed QBLI algorithm almost equals the pixel matching algorithm in terms of retrieval performance, but involves a much smaller computational load, and is therefore much better for interactive queries. It is also important to point out the fact that the PWT algorithm applied on binary images helps

Table 1
Retrieval results using 20 fax images on a database of 1062 images

Query image	Rank of original				
	Pixel matching	Basic PWT	Hu moment	Fourier descriptor	Proposed QBLI
1	1	104	266	987	1
2	1	369	308	14	1
3	1	15	1	1	1
4	1	21	741	217	3
5	1	272	9	276	1
6	1	130	1	2	1
7	1	258	1	819	1
8	1	2	1	1	3
9	1	502	106	427	1
10	15	302	54	16	2
11	1	603	1	1	1
12	1	299	49	28	1
13	1	60	203	614	1
14	1	495	224	759	4
15	1	500	1	466	2
16	1	339	270	803	1
17	1	15	33	810	2
18	1	264	606	405	4
19	1	1	10	1	1
20	1	1	42	1	1

Table 2
Retrieval results using three different distance metrics

Query image	Rank of original		
	Manh.	Norm. eucl.	Norm. manh.
1	1	1	2
2	1	1	1
3	1	1	4
4	3	1	1
5	1	1	1
6	1	1	1
7	1	1	1
8	2	1	1
9	1	1	1
10	2	1	1
11	1	1	1
12	1	1	1
13	1	1	1
14	3	5	2
15	2	1	3
16	1	1	1
17	1	1	1
18	3	61	61
19	1	1	1
20	1	1	1

in minimising computation time. Logically, a computation based on just ones and zeros is much quicker than a computation based on real values. The fast QBLI method integrates the high accuracy of the pixel matching method with the low computational load and the robustness of the basic PWT method.

6.2. The effect of the distance metric

In the previous experiment, we used the minimum Euclidean distance as distance classifier for the proposed algorithm. There are, however, several other distance metrics that can be used with the algorithm such as the Manhattan distance, Bayesian distance and Mahalanobis distance. However, with the exception of Manhattan distance, these distance metrics are rather complex in nature. For the sake of computational simplicity, we restrict our distance metrics only to simple functions, which means only the Euclidean and Manhattan distance are tested. However, we also introduce normalisation in distance computation to observe if the addition of normalisation can further boost the retrieval performance. The idea of normalising the feature vectors arises due to different range of energy values between different wavelet channels, and it is feared that this might lead to contributions from some channels becoming dominant. In the following experiment, we will observe the performance of three different distance classifiers, namely the Manhattan distance, the normalised Euclidean distance and the normalised Manhattan distance, compared to the Euclidean distance. The three additional distance metrics are given below:

$$d_{\text{Manhattan}}(i,j) = \sum_m \sum_n \left|\mu_{mn}^{(i)} - \mu_{mn}^{(j)}\right|, \tag{4}$$

$$d_{\text{Norm.Euclidean}}(i,j) = \sum_m \sum_n \left[\frac{\mu_{mn}^{(i)} - \mu_{mn}^{(j)}}{\sigma_{mn}}\right]^2, \tag{5}$$

$$d_{\text{Norm.Manhattan}}(i,j) = \sum_m \sum_n \left|\frac{\mu_{mn}^{(i)} - \mu_{mn}^{(j)}}{\sigma_{mn}}\right|, \tag{6}$$

where σ_{mn} is the standard deviation of a particular feature over the entire database.

Experimental results are shown in Table 2. From the table, there is not much difference between the three distance metrics compared to the Euclidean distance. However, closer experimental inspection suggests that the standard Euclidean and Manhattan distance metrics give somewhat more stable retrieval results. A good example is on the ranking of fax image 18, where in both the normalised cases, the ranking is outside the top 60. Careful inspection shows that fax image 18 has noise distributed across it in the form of black dots on a brighter background. After binarisation, it appears that the noise is still present in the binary image. This noise is usually of high frequency, thus it will be captured by the high-frequency channels of the wavelet decomposition.

The high-frequency channels of the wavelet decomposition have a small range of energy values compared to the other channels. By normalising the features, the noise in the high-frequency channels is amplified and therefore results in high dissimilarity between the fax and the original. Without the normalisation process, the dissimilarity measure will not be too big because the high-frequency channels contribute only a small portion of the overall measurement. Since low-quality images are very likely to contain noise as in fax image 18, it is better not to normalise the features. Further inspection shows that although the features have a wide range of energy value, none of the individual feature differences really dominate in such a way that a particular feature difference is completely superior to the others. Hence, there is no need for normalisation for the dissimilarity computation.

Between the non-normalised Euclidean and Manhattan distance, the Manhattan distance appears to be better than the Euclidean distance, where all the retrieved targets are within the top 3. The reason for this is that with Euclidean distance, the individual feature difference is squared, which further reduces the contribution of smaller feature differences. By using Manhattan distance, the individual feature differences are added together which preserves the contribution of each individual feature. Using this metric, none of the feature differences are amplified or neglected, hence resulting in a better retrieval performance. The Manhattan distance will therefore be used as the distance metric for the QBLI algorithm.

6.3. The effect of different numbers of decomposition levels, L

The number of decomposition levels is an important factor in the wavelet transform. It decides how many frequency channels a particular transform will have. The bigger the number of decomposition levels, the more the detail in the image is captured. How-

Table 3
Retrieval results using four different numbers of decomposition levels

Query image	Rank of original			
	$L = 6$	$L = 5$	$L = 4$	$L = 3$
1	1	2	5	11
2	1	1	1	1
3	1	3	19	106
4	3	1	2	1
5	1	1	3	18
6	1	9	5	1
7	1	1	1	2
8	2	2	1	5
9	1	1	1	2
10	2	1	14	2
11	1	1	2	83
12	1	1	3	5
13	1	1	1	2
14	3	57	143	387
15	2	6	3	8
16	1	1	2	6
17	1	1	1	1
18	3	16	41	178
19	1	1	2	53
20	1	1	1	2

Table 4
Retrieval results using six different wavelet bases

Query image	Rank of original					
	D4	D8	C6	S8	H	B
1	1	2	3	1	1	184
2	1	2	1	1	2	26
3	1	2	1	1	2	302
4	3	1	1	1	1	19
5	1	1	1	1	1	228
6	1	1	1	1	1	5
7	1	1	1	1	1	1
8	3	3	1	1	1	3
9	1	1	1	1	1	168
10	2	29	1	1	8	104
11	1	1	1	1	1	335
12	1	1	1	1	1	84
13	1	2	3	1	1	53
14	4	38	1	7	10	605
15	2	1	1	1	1	448
16	1	2	1	1	2	38
17	2	1	1	1	1	1
18	4	59	1	1	9	248
19	1	1	1	1	1	301
20	1	1	1	1	1	1

D4, Daubechies 4-tap; D8, Daubechies 8-tap; C6, Coiflet 6-tap; S8, Symmlet 8-tap; H, Haar; and B, binary wavelet transform.

ever, since the pyramidal wavelet transform decimates its output by a factor of 2, if the decomposition is continued all the way, we will obtain channels with only 1 pixel resolution. This might lead to inaccurate readings and unstable features, and will increase the number of features to be computed. Therefore, we limit the smallest resolution for a particular channel, and it was found experimentally that 4×4 pixels is appropriate for the smallest channel resolution. For a 256×256 image, this leads to a maximum of six levels of decomposition, the one used in previous experiments.

However, we might want to use fewer decomposition levels since it involves fewer computations and fewer features. Table 3 summarizes the results. From the table, clearly the retrieval performance decreases with the decrease of the number of decomposition levels. The decrease in performance is quite drastic for some query images, while steady in most other queries, depending on the quality of the queries. Using three decomposition levels for example results in five of the target images to be ranked outside of the top 50, which is not a good performance. Hence, since using six levels of decomposition gives the best performance, and its corresponding number of features (19 features) is still compact, we adopt these parameters for the QBLI algorithm.

6.4. The effect of using different wavelet bases

In the previous experiments, the Daubechies 8-tap wavelet is used as the wavelet basis for decomposition. Four more wavelet bases were tested in order to observe the effect of using different wavelet bases in the algorithm. In addition, a special binary wavelet transform decomposition is also tested since it is interesting to see if total binary operation on binary images might give better results. Six decomposition levels are applied for each case, and the Manhattan distance is used as classifier. The bases for the non-binary wavelets are the ones used in Matlab's wavelet toolbox [17], and for the binary wavelet transform are the ones suggested by Swanson and Tewfik [28]. The results are summarized in Table 4.

From the table, the binary wavelet transform particularly gives a very poor and inconsistent result. The other non-binary wavelet transform provides comparable performance. The poor retrieval results for the binary wavelet transform are due to the lack of information carried by the output of the binary wavelet transform. The binary wavelet transform performs a special filtering operation which results in a binary output. Since we are using mean energy as the feature, computing this feature on binary channels does not provide enough information, which leads to poor discrimination between images. While the binary wavelet transform is good in some application such as the compression of binary images, it is not suitable for image retrieval.

The other four wavelet bases give a good performance, especially the Coiflet and Symmlet bases. It can be concluded that the choice of wavelet bases is not very crucial in the QBLI algorithm, although based on this particular test, the Coiflet wavelet basis gives the best result (all but two are returned as the first match, the rest are in the top three) and will be used as the basis for the algorithm. One might argue that the Haar wavelet should give better performance because of the binary-like nature of its filter. This might be true, but since we are dealing with low-quality images, in which the query binary does not really resemble the target binary completely due to noise etc., a bit of tolerance is needed in computing the features. In this case, the Haar wavelet may be too accurate in representing the binary images so that when noise is present, the dissimilarity measure increases dramatically. The Daubechies 8-tap, Coiflet 6-tap and Symmlet 8-tap wavelets although they may not be very accurate in representing binary images, they are better for the reasons above.

6.5. Optimum threshold for binarisation

It is debatable whether the choice of creating 99 different binaries for each database image is optimal. Different numbers of binaries are tested for the algorithm to find the optimal number of binaries to be created. Steps of 0.5%, 2%, 5% and 10% (which creates 199, 49, 19 and 9 binaries, respectively) are tested and the results are shown in Table 5. Note that there is quite a large reduction in the number of features to be stored by reducing the number of binaries to be created. Experiments are conducted using Coiflet 6-tap wavelet basis, six decomposition levels and Manhattan distance as classifier.

From the table, increasing the number of binaries from 99 to 199 does not appear to improve the retrieval performance. For each case of the query image, the same result as using only 99 binaries is observed. Hence, there is no need for increasing the

Table 5
Retrieval results using different number of binaries

Query image	Rank of original				
	0.5%	1%	2%	5%	10%
1	3	3	6	3	155
2	1	1	6	23	9
3	1	1	1	1	2
4	1	1	1	18	55
5	1	1	2	1	7
6	1	1	1	1	1
7	1	1	1	1	1
8	1	1	1	1	3
9	1	1	1	1	3
10	1	1	1	3	3
11	1	1	1	1	1
12	1	1	1	1	1
13	3	3	3	4	5
14	1	1	1	1	1
15	1	1	1	1	1
16	1	1	1	1	1
17	1	1	12	167	146
18	1	1	2	2	2
19	1	1	1	5	58
20	1	1	1	1	1

The percentage indicates the increment in which a binary image will be created, which in turns indicates the total number of binaries for an image.

numbers. The performance of using 49 binaries does not differ much from that of using 99 binaries, except for some specific queries. All the retrieved targets are still within the top 15, and the fact that it halves the number of features to be stored might make 49 binaries useful. As the number of binaries decreases to 19 and 9, the performance continued to drop, which suggests that the steps of 5% and 10% are too big and result in poor binary matching.

In actuality, as long as the query binaries resemble closely enough the target binary, we will obtain good results. The problem arises when the percentage of pixels of the binary queries appear in the middle of two target binaries, say 85% of query binary compared to 80% and 90% of target binaries which can lead to a quite different binary matching. In this case, the 5% and 10% steps proved to be too large a gap for some queries and therefore should not be used. This experiment also suggests that in order to get the best possible result, we only have to use a maximum of 99 binaries as increasing them does not improve the performance. The choice of optimum number of binaries is therefore a tradeoff between retrieval performance and the computational speed. If the speed constraint is not very crucial, using 99 binaries is better, otherwise using 49 binaries offers a comparable alternative. Furthermore using 99 binaries will also scale better, thus we chose to use 99 binaries for the QBLI algorithm.

6.6. Using other forms of low-quality image as query

We now consider other types of low-quality image besides fax images to establish whether the proposed algorithm works for these particular types of images. As mentioned previously, we have listed five other low-quality image types which are the image of inappropriate brightness and contrast, highly compressed images, low-resolution images, quantized images and noisy images. The same twenty images as in Fig. 5 are used as the target for retrieval with the query being the modified low-quality version of them. All experiments are conducted using 99 binaries for each image, Coiflet 6-tap wavelet basis, having six decomposition levels and utilizing Manhattan distance as distance classifier, as this combination is found to be the best in the previous section.

6.6.1. Images of inappropriate brightness and contrast

The 20 selected images are modified by varying their brightness and contrast in order to make 20 queries for the experiment. We have conducted five different experiments using five different brightness/contrast modifications. The modifications are high brightness (75% increase in original brightness), low brightness (−75% of original brightness), high contrast (75% of original contrast), low contrast (−75% of original contrast), and a random combination of the brightness and contrast of the target image.

Experimental results show that for all queries, the target images are retrieved as the first rank. Clearly from the result, the proposed algorithm works well with this kind of low-quality images. Note that the queries are generated by performing some operation on the histogram of its originals. As long as the operation does not entirely change the histogram, the proposed QBLI algorithm lends itself well in solving this type of problem. Problems will only arise when a non-linear operation is performed on the original resulting in the pixel distribution of the image being completely modified, in which case the modified image will probably be very different from its original.

6.6.2. Highly compressed images

Five different compression quality levels are used to obtain low-quality queries. Each of these queries are produced using MATLAB, and are compressed using the standard JPEG compression. The quality setting in JPEG compression is measured by values between 0 and 100, with 0 corresponding to the highest compression (worst quality) and 100 corresponding to the lowest (best quality). The default quality setting for JPEG compression is 75. In this experiment, the five different compression qualities take the values of 0, 5, 10, 20 and 30.

From experimental observations, it can be concluded that the proposed algorithm works well using this kind of low-quality image. Even using the lowest compression quality as query (compression quality value of 0), the algorithm is still able to rank the target images within the top 3. The queries having compression quality of 0 and 5 in particular are very poor, yet the algorithm still manages to achieve very promising retrieval. As the compression quality increases, the performance increases unsurprisingly because the better the quality, the more it resembles the queries' originals.

6.6.3. Low-resolution images

In this experiment, the 20 selected images are resized to a lower resolution to produce low-quality query images. However, since in the algorithm we resize all images to 256×256, the 20 selected images need to be resized to a resolution smaller than 256, otherwise there will not be the same amount of information between the queries and the targets, and the queries cannot be considered as low-quality images. We choose to resize the selected images to 32, 64 and 128 pixels along their longest dimension. Table 6 shows the retrieval results for the low-resolution queries.

The result of resolution 32 is quite poor compared to the other resolutions. This is understandable since an image having 32 pixels in its longest dimension is actually a very small image and not much information is carried. We also have to notice that query images 9, 10 and 17 have a high width to height ratio. If their longest dimension is 32, their other dimension will have far less pixels resulting in a very small amount of information along that particular dimension. Increasing the resolution means increasing the amount of information carried, hence the reason for the much better results for resolution 64 and 128. Apart from the very small resolution of query images 9, 10 and 17 which is responsible for the very poor retrieval results, the other queries give quite a good performance, although not as good as the two previous types of low-

Table 6
Retrieval results for low-resolution images, where r is the resolution of image's longest dimension

Query image	Rank of original		
	$r = 32$	$r = 64$	$r = 128$
1	3	1	1
2	1	1	1
3	4	4	1
4	1	1	1
5	2	1	1
6	4	1	1
7	10	2	1
8	1	1	1
9	398	18	1
10	487	4	1
11	3	1	1
12	1	1	1
13	28	3	1
14	1	1	1
15	1	1	1
16	45	1	1
17	632	134	2
18	39	2	1
19	262	79	13
20	9	3	2

Table 7
Retrieval results for noisy images

Query image	Rank of original			
	Gaussian, σ=			Blurred images
	0.005	0.01	0.02	
1	1	1	12	1
2	1	1	1	1
3	1	1	1	7
4	1	1	1	1
5	1	1	1	1
6	1	1	4	1
7	1	1	1	1
8	1	1	1	1
9	2	3	20	335
10	1	2	41	6
11	1	1	1	1
12	1	1	1	1
13	4	14	76	3
14	1	1	1	1
15	1	1	1	1
16	1	1	8	45
17	1	1	17	308
18	1	1	1	44
19	8	1	7	4
20	1	1	1	2

quality images. Hence, the proposed algorithm is also suitable for handling this type of low-quality image query.

6.6.4. Quantized images

In this experiment, four different quantized images are produced for each target, having 2 (binary), 4, 8 and 16 grey levels, respectively. In all cases, the quantized images are poor compared to the originals.

Experimental results show that all target images are retrieved as the first match, hence the performances of all quantized cases are very good. The binary queries (two levels of grey values) are in fact not much different from the fax images, and since the proposed algorithm operates in binary, it lends itself very well to the binary queries. It is also worth mentioning that the performance of the other three quantized images are exactly the same as the binary cases, because they are all converted to the same binary version by the algorithm. Hence, these quantized images are handled very well by the QBLI algorithm.

6.6.5. Noisy images

The 20 selected images shown in Fig. 5 have noise added to produce noisy image queries. Gaussian noise having zero mean with three different strengths are chosen. The strength of the noise is characterized by their standard deviation, and values of $\sigma = 0.005$, 0.01 and 0.02 were used. In addition a second type of noise, in terms of image blurring, will also be tested. Pixel averaging of radius five pixels are performed on the selected images to produce blur images. Table 7 shows the result of the retrieval experiment.

For the Gaussian noise, the retrieval rates are not significantly affected for $\sigma = 0.005$. As the strength of the noise increases, the performance of the algorithm decreases, although they are still giving acceptable retrieval results. For $\sigma = 0.02$, apart from a few images which was heavily distorted because of the noise, the algorithm manages to rank most of the target images within the top 10. This shows that the proposed algorithm is also suitable in tackling noise related problems. As for the blurred images, except for a few queries, the proposed algorithm works well to retrieve the target images. The few poor results are for queries 9 and 17, and careful inspection shows that the particular blurred images are quite distorted compared to their originals.

6.7. Performance on very large database

The 20 fax images are now used as query images for retrieving their originals from a database consisting of 16,959 images. This is a full collection of images contributed to our server by the Victoria and Albert Museum. Six levels of decompositions are used to perform wavelet decomposition on 99 binaries per database image using Coiflet 6-tap wavelet basis, while the Manhattan distance metric are used to compute the dissimilarity measure.

It was found that the proposed algorithm has no problems in adjusting to a larger database. All of the target originals are retrieved within the top 6. Considering there are almost 17,000 different images within the database, top 6 performance is very good. In fact 13 of the 20 target originals are retrieved as the first ranked image, which suggests that the proposed algorithm manages to produce very similar feature vectors between the fax and the originals. Hence, even with many more images in the database to confuse the system, the algorithm is still able to find the correct target.

The time taken to perform the retrieval is recorded to be around 80 seconds on average on a 700 MHz Xeon processor. This is a reasonable performance considering there is no multidimensional indexing employed to index such a large number of feature vectors. An addition of a suitable multi-dimensional indexing technique can help with the increasing size of database and improve the speed of retrieval, and make the CBIR system friendlier for the users. In addition, a suitable fast searching algorithm could also further improve the speed of the system.

7. Conclusion and future work

In this paper, a novel *query by low-quality image* technique has been proposed which uses binary matching together with a pyramidal wavelet transform. We also developed a simple but time consuming technique which we call the pixel matching technique, used as a yardstick for the evaluation of the proposed QBLI algorithm. The pixel matching technique gave very good retrieval accuracy, while the proposed novel algorithm gave a comparable performance. This indicates that the QBLI algorithm is almost as good as the pixel matching algorithm, but has the advantage of being much faster computationally, making it more suitable for interactive use. The QBLI method illustrates the importance of the wavelet-based feature extractor in image analysis, since in the PWT, we have a very fast algorithm and compact feature vectors. These two constraints are important in using the QBLI method where multiple feature extractions as well as multiple feature vectors are necessary for each database image. Other feature extraction techniques are either slow or have large feature vectors.

Another important observation from the experiments conducted is that the binary image matching and thresholding method proposed is an excellent way to search using low-quality images. This is shown by the fact that the basic PWT technique gives a very poor retrieval accuracy when used for this kind of problem. This suggests that other feature extraction methods might also fail to deliver a good performance if not accompanied by a special preprocess like the binary image thresholding and matching technique. This is supported by the experiments using shape-based algorithms where both Hu moments and Fourier descriptor failed to produce reliable results. Further experiments suggested the use of Manhattan distance as the distance classifier in order to get the best results. Normalising the distance function appears to over-amplify the noise associated with the queries and therefore is not suggested. The best number of decomposition levels is found to be 6, resulting in 19 features, while the best basis was found to be Coiflet 6-tap wavelets although this parameter is not very crucial. The optimum number of binaries to be created for each image is found to be either 49 or 99, depending on whether accuracy or speed is more important to the system. Finally the proposed algorithm lends itself quite well to solving query by other low-quality image types.

Although the proposed algorithm gives good performance in solving *query by low-quality image*, it still offers plenty of room for improvement. For example, the system might face a problem if the object to background ratio of the image differs significantly from the target images. In this case, a special pre-processing algorithm to detect plain background could be developed so that all images will have their default object to background ratio, hence solving the above problem. The accuracy of the QBLI algorithm could also be further improved if some sort of text annotation can be associated with it, i.e., to use both text-based and content-based image retrieval (TBIR and CBIR). The TBIR could act as a pre-filter for the content-based retrieval process applied at later stage.

Acknowledgements

The authors are grateful to the Faculty of Engineering, Multimedia University, Malaysia and the School of Electronics and Computer Science at the University of Southampton, UK for financial support. We are also grateful to the EU for their support under Grant No. IST_1999_11978 (The Artiste Project), and to one of our collaborators, the Victoria and Albert Museum (London, UK) for use of their images.

References

[1] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, Content-based image retrieval at the end of the early years, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 1349–1380.

[2] Yong Rui, Thomas S. Huang, Image retrieval: current techniques, promising directions, and open issues, Journal of Visual Communication and Image Representation 10 (1999) 39–62.

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

[4] J.R. Bach, C. Fuller, A. Gupta, et al., Virage image search engine: an open framework for image management, Proceeding SPIE Storage and Retrieval for Image and Video Databases (1996) 76–87.

[5] A. Pentland, R.W. Pickard, S. Sclaroff, Photobook: content-based manipulation of image databases, International Journal of Computer Vision (1996).

[6] J.R. Smith, S.-F. Chang, Visualseek: a fully automated content-based image query system, Proceeding ACM Multimedia (1996) 87–98.

[7] T. Gevers, A.W.M. Smeulders, PicToSeek: a color invariant retrieval system, Image Databases and Multimedia Search (1997) 25–37.

[8] S.E. Umbaugh, Computer Vision and Image Processing, Prentice-Hall, Englewood Cliffs, NJ, 1998.

[9] R.C. Gonzales, R.E. Woods, Digital Image Processing, 3rd ed., Addison-Wesley, Reading, MA, 1992.

[10] Z.-N. Li, M.S. Drew, Fundamentals of Multimedia, Prentice-Hall, Englewood Cliffs, NJ, 2004.

[11] M. Nachtegael et al., A comparative study of classical and fuzzy filters for noise reduction, Proceedings of The 10th IEEE International Conference on Fuzzy Systems (2001) 11–14.

[12] R.A. Peters, A new algorithm for image noise reduction using mathematical morphology, IEEE Transactions on Image Processing 4 (1995) 554–568.

[13] M. Yoshioka, S. Omatu, Noise reduction method for image processing using genetic algorithm, Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (1997) 2650–2655.

[14] Q. Zhang, P.A. Mlsna, J.J. Rodriguez, A recursive technique for 3-D histogram enhancement of color images, Proceedings of IEEE Southwest Symposium on Image Analysis and Interpretation (1996) 218–223.

[15] H.-S. Wong, J.-H. Wang, Contrast enhancement based on divided histogram manipulation, Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (2000) 1551–1555.

[16] A. Graps, An introduction to wavelets, IEEE Computational Science and Engineering 2 (1995) 50–61.

[17] The MathWorks Inc., Wavelet Toolbox for Matlab version 6.1.

[18] S.G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (1989) 674–693.

[19] I. Daubechies, The wavelet transform, time-frequency localization and signal analysis, IEEE Transactions on Information Theory 36 (1990) 961–1005.

[20] T. Ngunyen, D. Gunawan, Wavelet and wavelet-design issues, Proceedings of ICCS (1994) 188–194.

[21] S. Mallat, Wavelets for a vision, Proceedings of IEEE 84 (1996) 604–614.

[22] J.-F. Aujol, G. Aubert, L. Blanc-Feraud, Wavelet-based level set evolution for classification of textured images, IEEE Transactions on Image Processing 12 (2003) 1634–1641.

[23] C.-M. Pun, M.-C. Lee, Extraction of shift invariant wavelet features for classification of images with different sizes, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 1228–1233.

[24] M. Unser, Texture classification and segmentation using wavelet frames, IEEE Transactions on Image Processing 4 (1995) 1549–1560.

[25] T. Chang, C.C.J. Kuo, Texture analysis and classification with tree-structured wavelet transform, IEEE Transactions on Image Processing 2 (1993) 429–441.

[26] A. Laine, J. Fan, Texture classification by wavelet packet signatures, IEEE Transactions on Pattern Analysis and Machine Intelligence 15 (1993) 1186–1191.

[27] W.Y. Ma, B.S. Manjunath, A comparison of wavelet transform features for texture image annotation, Proceedings of IEEE International Conference on Image Processing (1995) 256–259.

[28] M.D. Swanson, A.H. Tewfik, A binary wavelet decomposition of binary images, IEEE Transactions on Image Processing 5 (1996) 1637–1650.