Journal of Universal Computer Science, vol. 16, no. 3 (2010), 402-423
submitted: 27/7/09, accepted: 29/1/09, appeared: 1/2/10 © *J.UCS*

Block-based Against Segmentation-based Texture Image Retrieval

Mohammad Faizal Ahmad Fauzi
(Multimedia University, Cyberjaya, Selangor, Malaysia
faizal1@mmu.edu.my)

Paul H. Lewis
University of Southampton, United Kingdom
phl@ecs.soton.ac.uk)

Abstract: This paper concerns the best approach to the capture of local texture features for use in content-based image retrieval (CBIR) applications. From our previous work, two approaches have been suggested, the multiscale block-based approach and the automatic texture segmentation approach. Performance comparison as well as advantages and disadvantages of the two methods are presented in this paper. The databases used are the Brodatz and VisTex databases, as well as three museum image collections of various sizes and contents, with each collection presenting different challenges to the CBIR systems. Experimental observations suggest that the two approaches both perform well, with the multiscale technique having the edge in retrieval performance and scale invariance, while the segmentation technique has the edge in lighter computational complexity as well as having the shape information for later purposes. The choice between the two approaches thus depends on application.

Key Words: texture, texture segmentation, multiscale technique, content-based image retrieval, discrete wavelet frames
Category: H.3.1, H.3.3

1 Introduction

In *query by texture*, the main goal of the retrieval system is to be able to retrieve images containing similar texture to the given query texture. Many texture feature extraction techniques are available in the literature including the grey level co-occurence matrices [Haralick et al. 1973], simultaneous autoregressive model [Kashyap and Chellappa 1983, Mao and Jain 1992], fractal dimension [Chaudari and Sarkar 1995, Chen and Bi 1999, Kaplan 1999], Markov random field [Krishnamachari and Chellappa 1997, Wang et al. 1998], the Gabor transform [Dunn and Higgins 1995, Mittal et al. 1999], Law's texture feature [Laws 1980], and wavelet-based techniques [Hsin 2000, Lee and Chen 2002, Muneeswaran et al. 2005, Chang and Kuo 1993, Mallat 1989], with each having their own advantages and disadvantages.

Fig. 1 shows the architecture of a simple content-based image retrieval (CBIR) system using texture features. In natural scene images, texture usually appears only in some part of the image. Applying the feature extractor to the image

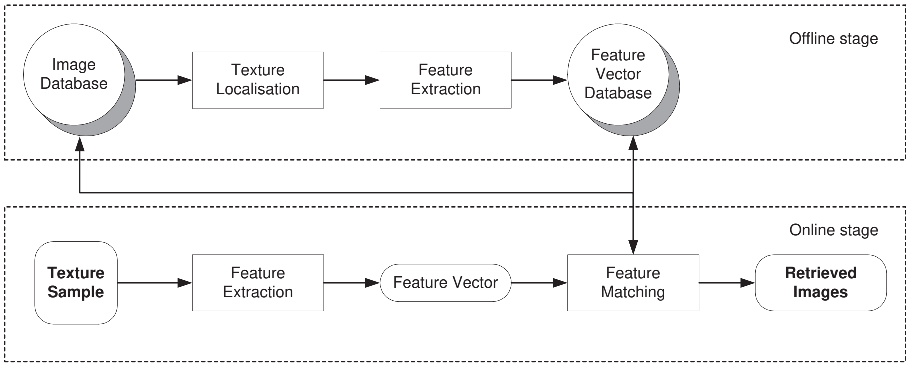


Figure 1: A simple content-based image retrieval system architecture for retrieving image with similar textures to the query

globally will result in an incorrect representation for the textures within the image. Instead of global features, local features from the texture regions should be extracted. It is thus very important to find the best approach, in terms of accuracy and computational speed, in utilizing the texture feature extraction method to obtain local statistics. Two approaches are considered in this paper, block oriented decomposition and automatic texture segmentation.

Using the block decomposition approach, features are extracted from several sub-image blocks [Chan et al. 2001, Natsev et al. 1999, Smith and Chang 1994, Manjunath and Ma 1996, Guo and Zhang 1997]. Assuming the sub-image blocks are small enough and efficiently structured, this method will produce feature vectors for each and every texture present in the image. However, since there are many blocks, and the textured and non-textured blocks are treated as equal, there is more risk of confusing the algorithm and reducing the discrimination accuracy.

Using the automatic texture segmentation approach, the image is first segmented into several textured and non-textured regions, and the feature vectors are then computed only on the textured regions [Porter and Canagarajah 1996, Liu and Zhou 2004, Perry and Lowe 1989, Yang et al. 2004]. By automatic segmentation, we mean a segmenter which does not need a priori knowledge either on the type of the texture or the number of textures present in the image. Once the image is segmented, texture features can then be extracted from the texture regions alone. However, as is the case with all segmentation problems, it is difficult to obtain an accurate segmentation, especially around the boundary of the texture, which can later affect the retrieval result.

We have conducted a comprehensive experimentation on several state of the art texture feature extraction techniques and found that the discrete wavelet frames (DWF) is the best technique to be used in our application domain, that is museum digital image collections of various kinds. In our previous work [Fauzi 2004, Fauzi and Lewis 2008], we have also studied several block oriented decompositions and proposed an improvement over the hitherto best found method, resulting in the multiscale block decomposition algorithm. In another work [Fauzi 2004, Fauzi and Lewis 2006], we have developed a novel automatic texture segmenter, which was subsequently improved to become a texture identifier, as an alternative to the block-based method in content-based image retrieval system. However since the approaches were developed and published independently, no direct comparison was made between the two approaches in the previous papers.

In this paper, a comprehensive evaluation of the two approaches in content-based image retrieval, based on the DWF technique, is presented. The performance of the two approaches is compared, especially the retrieval performance and computational complexity. To the best of our knowledge, there has not been any work elsewhere comparing these two approaches, hence we hope this research provides a good barometer for future texture-based CBIR applications. Most CBIR systems use the block oriented decomposition for extracting local texture since there are only a limited number of automatic texture segmentation algorithms available. Therefore it would also be interesting to see if segmentation is really needed in content-based image retrieval applications.

This paper is organized as follows. The next section will briefly describe the discrete wavelet frames technique, followed by the two approaches to be used, the multiscale block decomposition algorithm and the automatic texture segmenter. Section 3 describes the different image databases used in the experiment, with the experimental observations presented in section 4. Finally conclusions are presented in section 6.

2 Review of the Algorithms Involved

This section describes the two approaches to texture-based image retrieval. However the multiscale and segmentation-based methods are used only to capture local texture regions of an image. To extract the texture feature itself, the enhanced discrete wavelet frames technique is used as it was found that the technique is the most suitable for our application domain. This section summarizes the two approaches together with the enhanced discrete wavelet frames method.

2.1 Discrete Wavelet Frames

The discrete wavelet frames (DWF) is nearly identical to the standard wavelet transform, except that one upsamples the filters, rather than downsamples the image. While the frame representation is over-complete, and computationally

```
Initialize the feature vector FV,

H = height of the image,
W = width of the image,
L = minimum of ( H,W),

while ( L >= 64)

Divide image into several 64x64 sub-images S,

for i=1: S
        Get sub-image i,
        Perform DWF decomposition,
        Compute feature vector,
        Update FV,
end

K = nearest dyadic integer smaller than L,
Reduce image by a factor of K/L ,
H = new height of the image,
W = new width of the image,
L = minimum of ( H,W)

end
```

Figure 2: Pseudocode of the multiscale-based CBIR algorithm

more intensive than the standard wavelet transform, it holds the advantage of being translationally invariant [Unser 1995].

Given an image, the DWF uses a pair of 1D lowpass and highpass filters to filter the image horizontally and verticaly. The filtering process is performed without the sub-sampling process, resulting in four filtered images, or channels, of the same size as the input image. The decomposition is then continued in the LL channel only, but since the image is not sub-sampled, the filter has to be upsampled by inserting zeros in between its coefficients. The total number of filtered images generated by DWF decomposition is the same as in the standard wavelet transform, which is $3l-1$, where l is the number of decomposition levels. Usually the energy in each channel is used as the feature to describe the image.

2.2 Multiscale Block Decomposition

In [Fauzi and Lewis 2008], we proposed a multiscale texture retrieval technique for use with content-based image retrieval applications. Fig. 2 shows the pseudocode of the proposed algorithm. Using this algorithm, we start with one database image and divide it into several 64×64 overlapping blocks. For each block, the discrete wavelet frames method is used to extract features from the local region. After all the blocks generated for this level have been processed, the image is reduced in size by a certain factor, representing the next resolution image.

The above process is repeated for all the blocks in this resolution. Finally when the length of either dimension of the images reached 64, the process is stopped, and all the feature vectors created are stored. During retrieval, the feature vector of the query texture is compared with all the feature vectors of the image, and the one with the smallest distance is considered as the best match for that particular image. This is repeated for all images in the database and the distances are sorted so that retrieval can be based on increasing distance.

There are two approaches considered when generating the overlapping blocks, namely *case 1 overlapping* and *case 2 overlapping*. Using *case 1 overlapping*, an overlapping range between 0 and 63 pixels is allowed, while using *case 2 overlapping*, the range is improved to between 32 and 63 pixels. For a square sub-image, the number of sub-images generated, K for *case 1* and *case 2 overlapping* for any single level is calculated as:

$$K_{case1} = \left(\left\lceil \frac{M}{64} \right\rceil \right)^2 \tag{1}$$

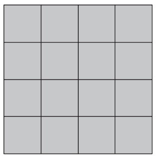
$$K_{case2} = \left(\left\lceil \frac{M}{64} \right\rceil * 2 - 1 \right)^2 \tag{2}$$

where M is the length of the sub-image, and $\lceil . \rceil$ is defined to be the smallest integer greater than or equal to a given number. The $\lceil . \rceil$ operator ensures the sub-images are interconnected and no sections of the image will be left out. Figure 3 shows an example of sub-images generated for a 256×256 image for both *case 1* and *case 2 overlapping*, and Figure 4 shows the variation of overlapping amount with different sizes of image for the two approaches. Note for dyadic size images, the overlapping between sub-images for the two cases is at its minimum, i.e. 0 and 32 pixels respectively.

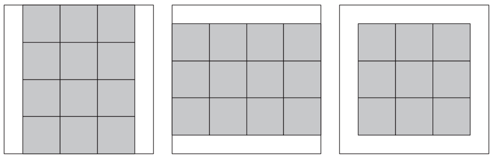
In terms of sub-image coverage, the minimum coverage of *case 2 overlapping* ($\geq \frac{9}{16}$) is more than twice that of *case 1 overlapping* ($\geq \frac{1}{4}$) (for detail mathematical derivation, readers are referred to [Fauzi and Lewis 2008]). Even though *case 2 overlapping* has a much better coverage than *case 1 overlapping*, the total number of sub-images generated however increases almost quadratically with the increase in the length of the image, which may effect the computation time considerably.

2.3 Automatic Texture Segmenter

In [Fauzi and Lewis 2006], we proposed an automatic texture segmentation algorithm for content-based image retrieval application. Figure 5 shows the pseu-



(a)



(b)

Figure 3: (a) Sub-images generated by *case 1 overlapping*, showing 0 pixel overlapping for dyadic image size (b) Additional sub-images generated for *case 2 overlapping*, ensuring minimum of 32 pixels overlapping

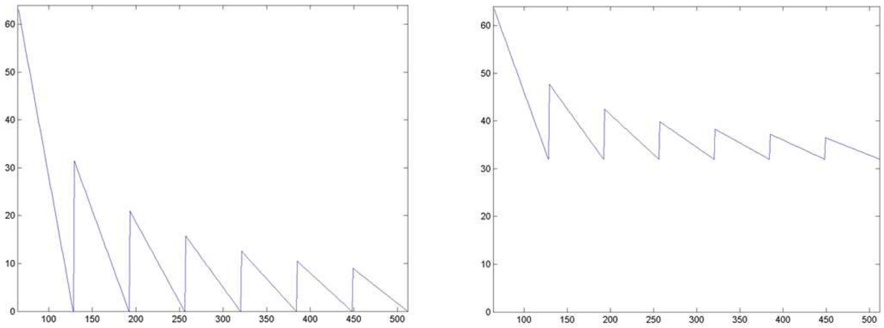


Figure 4: Amount of pixels overlapping for various image sizes (left) *case 1 overlapping*, and (right) *case 2 overlapping*

docode for the automatic segmentation algorithm itself, while Fig. 6 shows the pseudocode for incorporating the automatic segmentation into a CBIR system. The algorithm uses a modified DWF technique together with mean-shift clustering technique to automatically segment different textured regions without the need for information of either the type of texture or the number of textures which exist within the image.

The modified DWF is proposed for better clustering of data points as compared to the standard DWF and the standard wavelet transform. It uses the richer coefficients of the DWF to generate the pyramid structure of the wavelet transform. Instead of throwing away every other coefficients during the sub-

```
L = number of decomposition levels,

Perform DWF decomposition up to L levels, followed by data reduction,

Apply adaptive smoothing for every channels,

Data = 4 channels from the highest level,

while data != base level
        Apply mean shift algorithm,
        Apply fuzzy clustering to obtain membership function, U,
        Interpolate U to the next resolution,
        Data = Interpolated U + 3 channels of the next level,
end

Assign each pixel to the class it has the highest probability of.
```

Figure 5: Pseudocode of the detailed segmentation algorithm

```
Identify the number of segments J,

Initialize the feature vector FV,

for i = 1: J
        Get segment i,
        T = (Identify if the segment is textured),
        if T is true
                Perform DWF decomposition for the segment,
                Compute the feature vector,
                Update FV,
        end
end
```

Figure 6: Pseudocode of the segmentation-based CBIR algorithm

sampling stage as in the case for standard wavelet transform, new wavelet coefficients are computed by taking the mean energy within distinct blocks of size $2^k \times 2^k$, where k corresponds to the decomposition level. Mean shift clustering, on the other hand, is used to automatically determine the number of segments within the image through the modified multi-resolution DWF data space. It consists of five stages, namely data sampling, mode seeking, cluster center derivation, cluster center validation and cluster delineation.

The segmentation process consists of two stages, the top-down decomposition using DWF followed by bottom-up segmentation using mean shift clustering. During the top-down phase, the image is decomposed into pyramid structured DWF up to a certain level, l, followed by adaptive smoothing to reduce variances. In the subsequent bottom-up phase, starting from the highest level (lowest resolution), the four DWF channels at that level are used as the input to the mean-shift clustering process. The output is a 2D membership function describing the probability of every pixel belonging to the different clusters identified. The membership function is then interpolated and combined with the 3 DWF channels at the next resolution, and the whole process is repeated. At the base level, the membership function is of the same size as the original image, and the segmented image can be obtained by assigning every pixel to the class for which it has the highest probability.

To use the texture segmentation algorithm in CBIR, it is extended to become a texture identifier. For a particular segment, by comparing the total energy in the middle frequency channel with the ones in the low frequency channels, we can estimate whether or not the segment is textured. The feature extraction can then be performed on the textured regions only, and the feature vectors for each texture segment are stored. During retrieval, the feature vectors of the query texture will be compared to all feature vectors of all images inside the database, and the retrieval is carried out based on increasing distance.

3 Experimental Databases

Several databases were used throughout the experimental section. For quantitative evaluation of the performance of the DWF, multiscale block decomposition and the segmentation algorithms, the Brodatz texture database [Brodatz 1966] as well as the VisTex database [Picard et al. 1995] are used. To evaluate different aspects of the techniques, different sub-sets of databases are derived from these two databases. Since the ground truth of the data is known, quantitative evaluation can easily be obtained. The different databases used throughout the experiments are summarized in Table 1.

For the experiments on the discrete wavelet frames technique, the 112 textures (of size 512×512) from the Brodatz collection are divided into 16 non-overlapping sub-images (of size 128×128), resulting in 1792 images in total corresponding to 112 classes (DB1). To test the algorithm on color texture database, the VisTex database is used, where 16 non-overlapping images are extracted from the 167 parent images, yielding 2672 images in total (DB2). The ability of the algorithm to retrieve images from the same class of the query is recorded. The retrieval is conducted for all images, and the average recall rate is recorded.

For the multiscale-based retrieval experiment, nine overlapping sub-images of size 256×256 were extracted from each of the 112 Brodatz images, resulting

Table 1: Summary of different databases used throughout the experiments

Database	Description
DB1	Brodatz textures of 128×128 size, 16 textures per class × 112 classes = 1792 images.
DB2	Vision textures of 128×128 size, 16 textures per class × 167 classes = 2672 images.
DB3	Brodatz textures of size 256×256, 9 textures per class × 112 classes = 1008 images. 10 Vision textures of size 80×80 were pasted onto the first 90 Brodatz textures to create 90 target images.
DB4	Same as DB3, but the Vision textures pasted is increased to 140×140.
DB5	Same as DB3, but the Vision textures pasted is increased to 200×200
DB6	Same as DB3, but the Brodatz textures can be of any size.
DB7	National Gallery image collections, totalling 1462 images.
DB8	Victoria and Albert Museum image collections, totalling 16959 images.
DB9	C2RMF image collections, totalling 2500 images.

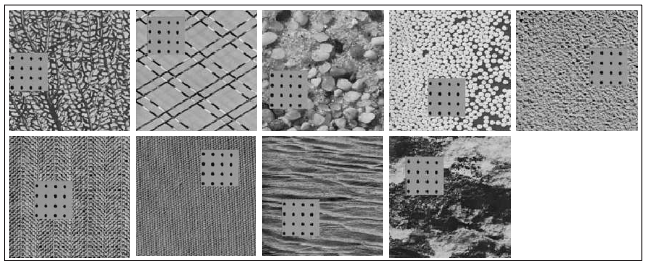


Figure 7: Example of target images created from the Brodatz and Visual texture databases.

in 1008 images. Out of this, 90 images are selected as target images, and ten Visual Texture images are each cut-and-pasted on nine different locations in the 90 selected Brodatz textures (DB3, DB4, DB5). The ten Visual texture images are used as query images, and the ability of the multiscale algorithm to retrieve the Brodatz images which contain the pasted VisTex texture is recorded. Figure 7 shows an example of the nine target images for one of the query image. Besides the dyadic size image database, the experiment was also conducted on arbitrary size Brodatz image collections (DB6).

For the segmentation-based retrieval experiment, the same setup as in the multiscale approach above is used, and the ability of the algorithm to retrieve Brodatz images in which one of its segments contains similar texture to the query is recorded. As for evaluating the segmentation algorithm itself, several multi-textured images are created from the Brodatz datasets and the ability of the algorithm to correctly segment the images are recorded. The multi-texture images contain between two to five different textures, arranged in a manner in which calculation of misclassified pixels is possible.

For evaluation of the algorithms in a real application area, museum databases are used. The images from our three collaborators, which are the collection from the National Gallery (NG), the Victoria and Albert Museum (VA) and the C2RMF databases will be used in the retrieval experiments, and is named DB7, DB8 and DB9 respectively. The three databases carry quite different challenges. The VA database, with 16959 images, has mainly images of objects hence is not too complex, but the database size is the largest. The NG database, with 1462 images, contains more complex images because all of its images are paintings of scenes. Finally the C2RMF database, with 2500 images, offers the most difficult challenge since many of its images are quite "smudgy" paintings (because they are originally for use in restoration research) and also not many textures are significantly visible.

4 Experimental Results and Discussion

In this section, experimental results for the discrete wavelet frames, multiscale-based and segmentation-based techniques on the different databases are presented. Throughout the section, the precision and recall are used to measure the retrieval performance:

$$Precision = \frac{Number\ of\ retrieved\ images\ that\ are\ relevant}{number\ of\ retrieved\ images} \tag{3}$$

$$Recall = \frac{Number\ of\ relevant\ images\ that\ are\ retrieved}{number\ of\ relevant\ images} \tag{4}$$

For each experiment, only the top N retrieved images are considered, where N is the number of relevant images. The precision and recall will therefore record similar measurements. Throughout the section, we will refer to this measurement as the recall rate, except for the last experiment on real museum images where the precision for the top 10 retrieved images are recorded instead because of the lack of ground truth data.

4.1 Discrete Wavelet Frames Experiments

Several state of the art texture feature extraction techniques have been experimented, which includes the multiresolution simultaneous auto-regressive model

Table 2: Percentage of recall rate for different texture methods

Texture Method	Average Recall Rate	Computational Speed (sec)	
		128×128	256×256
PWT	70.48	0.09	0.21
TWT	70.42	0.45	0.95
DWF	71.56	0.19	0.58
Gabor 6×3	71.77	2.23	53.30
Gabor 6×4	74.34	2.78	59.11
DCT	55.96	0.06	0.57
Law's	64.49	0.17	2.23
GLCM	44.99	1.27	3.68
MRSAR	74.10	15.01	61.16

(MRSAR), Gabor filters (18 and 24 filters), discrete wavelet frames, pyramid and tree-structured wavelet transform (PWT and TWT respectively), Law's method, discrete cosine transform (DCT) and grey-level co-occurrence matrix (GLCM). The performance of the nine techniques in retrieving 100 homogeneous textures from the Brodatz database DB1 is presented in Table 2.

From the table, it can be observed that the best recall rate is obtained by the Gabor 6×4 filters and the MRSAR methods, followed by the Gabor 6×3 filters and DWF methods. However from the computational complexity point of view, Gabor and MRSAR is computationally very heavy. The DWF method, on the other hand, is computationally very light, which suggests that it is the optimal method for texture retrieval, with high recall rate and low computational complexity.

The second experiment was conducted to find the optimal value for all the parameters of the DWF, such as the wavelet basis, the number of decomposition levels and the image padding type, as well as the distance metric used during the retrieval stage. In the previous experiment, three levels of decomposition is used, based on Daubechies wavelet, using periodic padding. In this experiment, several wavelet bases were tested, including the Haar wavelet, Symlet with 4, 8 and 16 taps, Coiflet with 6, 12 and 18 taps, and Daubechies with 8 and 16 taps. Up to five levels of decomposition were tested, as well as the periodic, symmetric and zero padding configuration. 4 different distance metrics were tested, which are the Manhattan, Euclidean, normalized Manhattan and normalized Euclidean metrics. It was found that the parameters used in the previous experiment actually generated the best performance. The choice of wavelet basis is actually not very crucial but nonetheless the Daubechies 8-tap wavelet is chosen. As for the distance metric, the normalized Euclidean outperforms the other metrics.

The final experiments were conducted to further improve the performance of

Table 3: Summary of the optimal parameters for DWF

Parameters	Optimal Parameter Value/Type
Level of decomposition	3
Wavelet basis	Not crucial, but Daubechies 8-tap is chosen
Padding type	Periodic
Distance metric	Normalized Euclidean
Features	Standard deviation of energy and zero-crossings
Channels	All channels

the DWF. Two approaches were considered. Firstly, instead of computing the energy in each of the DWF channel, several other statistical features were tested. These include the standard deviation of coefficients, standard deviation of energy, maximum and minimum value, maximum energy, maximum row and column sum energy, and the number of zero-crossings. It was found that combining the standard deviation of energy with the number of zero-crossings gives the highest recall rate. Secondly, several channel combinations were tested to see if dropping any channel would affect the performance of the DWF. It was found that in order to get the highest performance, all channels must be used. For color textures, all the textures are first converted to gray scale before DWF is applied. The choice of conversion operator does not seem to affect the performance of the algorithm substantially.

Table 3 summarizes the optimal parameters recorded for DWF. These DWF settings yield 20 features per texture image, and manages to improve the recall rate to 80.67% for database DB1. Experiments on the VisTex database DB2, which consists of color textures, recorded an average recall rate of 68.58%. This is quite a high recall rate considering the level of confusion the VisTex database brings.

4.2 Multiscale-based Texture Retrieval Experiments

Three sets of experiments were conducted to test the robustness of the multiscale algorithm. In the first experiment, the sensitivity of the sub-image locations is tested in order to compare *case 1 overlapping* with *case 2 overlapping*. For this experiment, the evaluation is based on a single scale only, i.e. the algorithm is expected to retrieve sub-images that correspond to the original scale only. Database DB3 was used for this experiment. The target sub-images pasted on the database images of size 256×256 is of size 80×80 pixels while the size of the query images is 64×64. It was found that the performance of *case 2 overlapping* doubles the recall rate for *case 1 overlapping* (80% to 40%). This is a huge gap as far as retrieval performance is concerned. We can conclude that

Table 4: Recall rate (%)for five different scales of query images

Database	Scale 1 (Original)	Scale 2	Scale 3	Scale 4	Scale 5
DB3	100	71.1	18.9	4.4	5.5
DB4	96.7	78.8	37.7	38.9	100
DB5	100	86.7	46.7	41.1	94.4

case 1 overlapping, although it has a much lower computational intensity, is far behind *case 2 overlapping* in terms of retrieval performance.

The second set of experiments is aimed at testing the multiscale nature of the multiscale algorithm, using *case 2 overlapping* approach. However because of the nature of the multiscale algorithm, where the image is downsampled to get to the next scale, database DB3 which uses 80×80 target textures is not very suitable. This is because, for such a small target region, the probability of the algorithm to capture the region at multiple scales is slim. This can be seen in Table 4 for the DB3 database, where the recall rate decreases with different scales. Because of this, databases DB4 and DB5, which employ 140×140 and 200×200 target textures respectively, were also used. Five different scales were tested for the query images. The size of the query image remains the same at 64×64 pixels. The five different scales for the query images are produced from the original VisTex texture by appropriate resizing of the original images. From Table 4, it is observed that, for the databases DB4 and DB5, the multiscale algorithm does reduce the scale dependent nature of DWF. Without the multiscale algorithm, if the query image is of different scale to the target texture, the recall rate will be very low. However with the multiscale algorithm, it can be seen that similar but different scales textures can also be retrieved by the DWF. This is true especially for scale 5, where the scale is close to half the scale of the target texture. The difference will be more obvious if the database images used are larger than 256×256, since more scales will be involved.

The results reported in the previous experiments actually represents the worst case result for the multiscale algorithm. Remember that the database images used are all of dyadic size, hence the overlapping is at the minimum. If a database consisting of a collection of random size images is used, the retrieval performance will be better. To prove this, another set of experiments was conducted. For this experiment, database DB6 was used. From each 512×512 scanned Brodatz texture image, nine randomly sized images are produced, which can be of any height, width and location within the parent images. This results in 1008 random sized images in the database. Next, for each ten Vision textures, target images of size 80×80 are randomly pasted onto 9 different database images, resulting in 90 modified database images consisting of target textures. Each

Table 5: Percentage of correctly detected number of textures

No. of textures in an image	No. of images tested	Images with correctly detected no. of textures	No. of textures detected for the incorrect case			
			2	3	4	5
2	16	15		1		
3	12	11				1
4	13	12				1
5	9	7			2	
Total	50	45 (90%)	0	1	2	2

of the ten VisTex textures (of size 64×64) is used as the query for the retrieval experiment. An average of 88% recall rate is reported from the experiment. If we compare this rate with the rate using the dyadic image sizes, where the rate is 80%, the difference is rather significant.

4.3 Segmentation-based Texture Retrieval Experiments

Before we can evaluate the retrieval results using the segmentation-based approach, it is important to first evaluate the segmentation algorithm itself. The performance of the segmentation algorithm is evaluated by its ability to identify the correct number of textures in the image, as well as its precision in defining the boundaries of the segmented images. The precision is measured by computing the percentage of misclassified pixels in the segmented images. We applied our texture segmentation algorithm to several images of composite textures with size 256×256 pixels.

Textures from the Brodatz album are used to make up the composite texture images, ranging from two to five textures per image, by a cut-and-paste technique. Textures pasted are of either rectangular or square shape in order to make the computation of misclassified pixels easier. None of the textures used in our experiment can be discriminated by grey level values alone. All together, we have applied our algorithm to 50 composite textures, and the results are summarized in Table 5. A return of 90% correctly detected number of textures is very promising. Except for one of the three-textured images, which the algorithm detected to have five textures, all other incorrect results only miss by plus/minus one texture. Experiments on synthetic textures and real scene image also shows good segmentation accuracy.

The texture segmentation algorithm, after extension to become texture identifier, is evaluated quantitatively in the same manner as in the evaluation of the multiscale algorithm. From visual inspection of the conducted experiments, a

Table 6: Recall rate (%)for the segmentation-based retrieval

Database	Scale 1 (Original)	Scale 2	Scale 3	Scale 4	Scale 5
DB3	4.4	3.3	6.7	1.1	2.2
DB4	38.9	60	38.9	15.6	11.1
DB5	35.6	60	38.9	16.7	10

textured region usually gives a ratio of less then 10, while a non-textured region can be from 10 up to infinity. However this is not always true as our collection of images is very large and it is impossible to visually inspect the ratio of all the textures. To avoid a situation where a genuine texture is missed, a threshold of 20 is used.

For the retrieval experiment, the same databases used in the multiscale experiments, i.e. database DB3, DB4, and DB5, were used. Obviously the number of textures in any particular image in this database is two at the maximum, even though the algorithm is able to segment many more segments. However for this quantitative evaluation of the algorithm, we restrict our images to a maximum of two textures, so that a fair comparison can be made. The segmentation-based approach for images having more than two textures per image will be addressed in the museum databases in the next section.

A summary of the retrieval results for the segmentation-based approach is presented in Table 6. From the table, the performance on DB3 in particular is very poor. For all scales of the query, the average recall rate is well below 10%. The main reason for this poor result is the small region of the target textures. For database DB3, the target textures are of size 80×80 in a 256×256 image. This constitutes less than 10% of the total image area, which means the segmenter itself in the first place will have problems segmenting the small regions. An investigation on the segmentation performance on the 90 target textures revealed that only 41% of them gave satisfactory segmentation results.

As the size of the target textures is increased (by using databases DB4 and DB5), the texture segmenter is able to work much better, where satisfactory segmentation results increased to 65%. This in turn results in a much improved retrieval result. Another observation that can be deduced from Table 6 is that, unlike the multiscale approach, the retrieval system is not scale invariant. If the scale of the target textures is different from the query texture, they will not be retrieved, which explains the poor recall rate for scale 4 and 5.

In general, the segmentation-based retrieval depends a lot on the accuracy of the segmenter itself. For the experiments discussed above, the average retrieval rate is not very high, and several factors have been identified. Firstly, if two textures in an image are quite similar, there is a high chance that the two regions will not be segmented. The two textures will be regarded as one, and the features extracted are compromised. Secondly, some Brodatz textures are non-homogeneous and complex (e.g. D039, D040, D041, D042, D043, D044, D045, D062, D067, D086, D090, please refer to [Brodatz 1966]), resulting in over-segmentation. Thirdly if one of the over-segmented regions is statistically similar to the target texture, then it will be regarded as one texture, and again the features extracted from the target texture are compromised. Finally some Brodatz textures are coarse in general but with fine textures inside (e.g. D011). The lack of multiscale property in the segmenter means they can only segment either one of these coarse or fine textures.

4.4 Comparison on Brodatz and VisTex Databases

Using the results in the previous sections, a comparison between the multiscale-based and the segmentation-based retrieval approaches can be drawn. Based on the recall rate obtained from the ground truth data, the multiscale-based method clearly outperforms the segmentation-based method. Not only does the multiscale-based approach give a better recall rate, it also enjoys the multiscale property that can reduce the scale dependence, which the segmentation-based approach lacks.

In terms of speed, they can be divided into 2 types, online retrieval response, and offline feature extraction response. Both are important although, since the online response involves users, it may be deemed more important. While the speed can be improved using faster computers, finding the approach with lighter computational load is still important in case hardware requirement is an issue. For the offline feature extraction stage, the segmentation approach recorded a lower computational load compared to the multiscale approach. Depending on the image size, the multiscale approach can take up to a few minutes to perform the feature extraction of several hundreds generated sub-images. For the same image, the segmentation approach recorded a much faster response, hence can be advantageous when the retrieval system involves a very large database size. As an example, to extract features from museum images of size 768×512, the multiscale and segmentation approaches recorded an average of 110 and 25 seconds respectively.

During the online retrieval stage, again the segmentation approach gives a faster response. This is because the number of feature vectors of the segmentation-based algorithm for a particular image is much lower than for the multiscale-based algorithm. For the multiscale method, up to hundreds of feature vectors need to be matched with the query for a single database image, whereas for the segmentation method, only N feature vectors need to be compared, where N is the number of textures which exist within the image. As an example, the multiscale and segmentation approaches recorded an average of 11 and 1.5 seconds

Figure 8: Example of retrieval result for National Gallery database. The query image is located at the top left, followed by the top ten retrieved images using the multiscale-based approach, and the top ten retrieved images using the segmentation-based approach

respectively for retrieving similar images from the DB6 database.

4.5 Comparison on Real Museum Databases

The multiscale-based and the segmentation-based approaches are both evaluated on the three museum databases (DB7, DB8, DB9). The retrieval performance will be based mainly on human visual inspection as there is no ground truth associated with these databases. However due to a limitation in space, it is not possible to give too many examples of the retrieval performance of the algorithms. The precision rate will also be used to provide some rough quantitative measurements of the retrieval performances by considering the top 10 retrieved images. Fig. 8, 9 and 10 show examples of retrieval results using both approaches on the DB7, DB8 and DB9 databases respectively.

Figure 9: Example of retrieval result for Victoria and Albert Museum database. The query image is located at the top left, followed by the top ten retrieved images using the multiscale-based approach, and the top ten retrieved images using the segmentation-based approach

The most important observation from all the experiments conducted on museum images is that both algorithms work well with all databases, quantitatively and qualitatively. From Fig. 8, 9 and 10, both algorithms manage to retrieve visually similar texture, even for the DB9 database, although its performance is not as good as for the DB7 and DB8 databases. This is because the ”smudgy” nature of images within the C2RMF images brings a level of confusion. The DB7 and DB8 databases on the other hand show a much better retrieval result. Even for a database size of 16959 images for the Victoria and Albert Museum database, both *query by texture* algorithms manage to retrieve similar textures to the query, implying that the algorithms are not affected by the size of the databases. In terms of the precision, the multiscale-based approach recorded 60%, 100% and 60% rates for the DB7, DB8 and DB9 respectively. The segmentation-based approach, on the other hand, recorded 50% precision each for all 3 databases. Both algorithms can be said to be applicable for content-based image retrieval

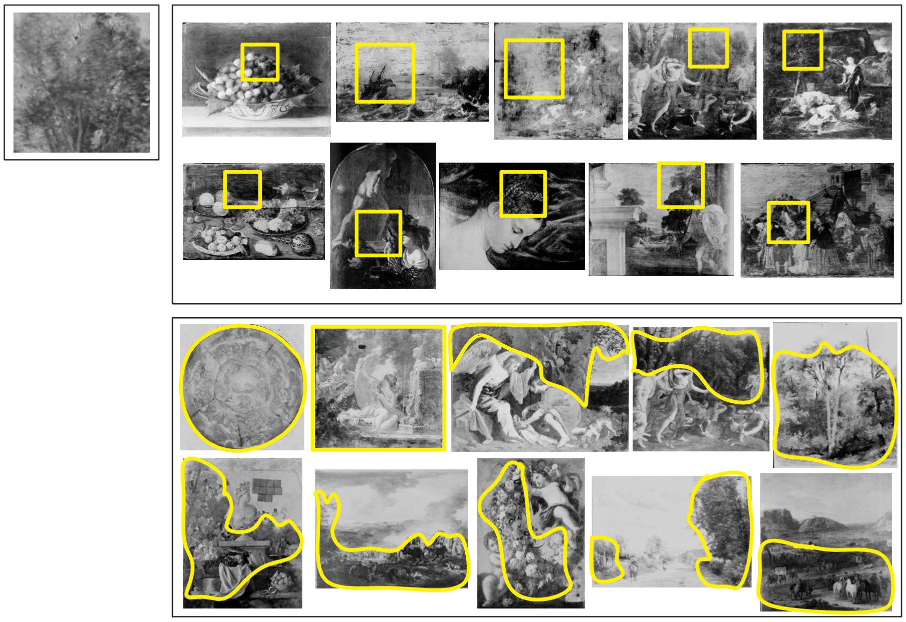


Figure 10: Example of retrieval result for C2RMF database. The query image is located at the top left, followed by the top ten retrieved images using the multiscale-based approach, and the top ten retrieved images using the segmentation-based approach

of museum collections.

The next observation is to compare the performance between the multiscale-based and the segmentation-based approaches. From the experiments, it was observed that the multiscale-based approach gives better performance than the segmentation-based approach. One example can be seen from Fig. 8. In this particular example, the multiscale-based approach manages to retrieve two images which have exactly the same texture as the query (images ranked first and third) while the segmentation-based approach can only manage to retrieve one of them. This is probably caused by a fault in the segmentation process or the lack of multiscale property within the segmentation-based approach. It was also observed that in general the patches retrieved by the multiscale-based approach are also much more similar to the query. The algorithm however can also suffer from some odd retrieval result. For example, when two textures are captured by a sub-image, the resulting feature vector might be similar to a feature vector of a completely different query texture, and hence will be retrieved as one of the top matches.

The segmentation-based algorithm has the advantage of retrieval speed, although the accuracy is not as good as the multiscale-based approach. The computational load for the feature extraction process is also lower than that of the multiscale-based approach. If it takes the multiscale-based algorithm more than one minute on average to process each database image during the feature extraction process, then for a database of 16959 images as in the VA database, it will take a few days to create the feature vectors for the entire database. Even though this process is carried out in advance and does not involve the users, it is still a problem, at least for the system administrator. Hence the segmentation-based approach holds the advantage in this particular respect. Although this approach lacks the multiscale property, it can on the other hand provide the shape of the segmented objects. This might be useful for later purposes such as object identification or shape-from-texture processes. On the downside, this approach can suffer from a fault during the segmentation process, where either a texture is classified as insignificant or a texture is classified together with another texture.

5 Conclusion

Overall, the advantages and disadvantages of both algorithms can be summarized as follows. The multiscale-based approach has the advantage of much better accuracy at the expense of computational load. The multiscale nature of this approach also adds to its advantages as it has been proven to be useful in capturing both coarse and fine textures. The choice between the multiscale-based and the segmentation-based approaches therefore depends on application. If the user is willing to trade a longer time for a better accuracy and scale invariance, then the multiscale-based approach will be suitable. However if the user requires quick retrieval and is willing to tolerate a slightly less reliable outcome, then the segmentation-based approach is more suitable. An algorithm which combines the advantages of the multiscale-approach with the advantages of the segmentation-based approach to produce a better texture-based CBIR system would be worthy of exploration.

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 number IST_1999_11978 (The ARTISTE Project), and to our collaborators, the Victoria and Albert Museum (London, UK), the National Gallery (London, UK) and the Research and Restoration Center for the Museum of France (Paris, France) for use of their images.

References

[Brodatz 1966] Brodatz, P.: "Textures: A Photographic Album for Artists Designers"; Dover Publications Inc., New York (1966).

[Chan et al. 2001] Chan, S., Martinez, K., Lewis, P., Lahanier, C., Stevenson, J.: "Handling sub-image queries in content-based retrieval of high resolution art images"; In Proceedings of International Conference in Cultural Heritage and Technologies, (2001), 157–163.

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

[Chaudari and Sarkar 1995] Chaudhuri, B.B., Sarkar, N.: "Texture segmentation using fractal dimension"; IEEE Transactions on Pattern Analysis and Machine Intelligence 17, (1995), 72–77.

[Chen and Bi 1999] Chen, Y.Q., Bi, G.,: "On Texture Classification Using Fractal Dimension"; International Journal of Pattern Recognition and Artificial Intelligence 13, (1999), 929–943.

[Dunn and Higgins 1995] Dunn, D., Higgins, W. E.: "Optimal Gabor filters for texture segmentation"; IEEE Transactions on Image Processing 4, (1995), 947–964.

[Fauzi 2004] Fauzi, M.F.A.: "Content-based image retrieval of museum images"; Ph. D. Thesis, University of Southampton (2004).

[Fauzi and Lewis 2006] Fauzi, M.F.A., Lewis, P.H.: "Automatic texture segmentation for content-based image retrieval application"; Pattern Analysis and Applications 9, (2006), 307–323.

[Fauzi and Lewis 2008] Fauzi, M.F.A., Lewis, P.H.: "A multiscale approach to texture-based image retrieval"; Pattern Analysis and Applications 11, (2008), 141–157.

[Guo and Zhang 1997] Guo, J., Zhang, A.: "Image decomposition and representation in large image database systems"; Journal of Visual Communication and Image Representation 8, (1997), 167–181.

[Haralick et al. 1973] Haralick, R.M., Shanmugam, K., Dinstein, I.: "Textural features for image classification"; IEEE Transactions on Systems, Man, and Cybernetics SMC-3, (1973), 610–621.

[Hsin 2000] Hsin, H.-C.: "Texture segmentation using modulated wavelet transform"; IEEE Transactions on Image Processing 9, (2000), 1299–1302.

[Kaplan 1999] Kaplan, L.M.: "Extended fractal analysis for texture classification and segmentation"; IEEE Transactions on Image Processing 8, (1999), 1572–1585.

[Kashyap and Chellappa 1983] Kashyap, R.L., Chellappa, R.: "Estimation and choice of neighbors in spatial-interaction models of images"; IEEE Transactions on Information Theory 29, (1983), 60–72.

[Krishnamachari and Chellappa 1997] Krishnamachari, S., Chellappa, R.: "Multiresolution Gauss-Markov Random Field models for texture segmentation"; IEEE Transactions on Image Processing 6, (1997), 251–267.

[Laws 1980] Laws, K.I.: "Textured Image Segmentation"; Ph. D. Thesis, University of Southern California (1980).

[Lee and Chen 2002] Lee, K.-L., Chen, L.-H.: "A New Method for Extracting Primitives of Regular Textures Based on Wavelet Transform"; International Journal of Pattern Recognition and Artificial Intelligence 16, (2002), 1–25.

[Liu and Zhou 2004] Liu, Y., Zhou, X.: "Automatic texture segmentation for texture-based image retrieval"; In Proceedings of the International Conference on Multimedia Modelling, (2004), 285–290.

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

[Manjunath and Ma 1996] Manjunath, B.S., Ma, W.Y.: "Texture features for browsing and retrieval of image data"; IEEE Transactions on Pattern Analysis and Machine Intelligence 18, (1996), 837–842.

[Mao and Jain 1992] Mao, J.C., Jain, A.K.: "Texture classification and segmentation using multiresolution Simultaneous AutoRegressive models"; Pattern Recognition 25, (1992), 173–188.

[Mittal et al. 1999] Mittal, N., Mital, D.P., Chan, K.L.: "Features for texture segmentation using Gabor filters"; In Proceedings of Seventh International Conference on Image Processing and Its Applications, (1999), 353–357.

[Muneeswaran et al. 2005] Muneeswaran, K., Ganesan, L., Arumugam, S., Soundar, K.R.: "Texture classification with combined rotation and scale invariant wavelet features"; Pattern Recognition 38, (2005), 1495–1506.

[Natsev et al. 1999] Natsev, A., Rastogi, R., Shim, K.: "WALRUS: A similarity retrieval algorithm for image databases"; In Proceedings of ACM SIGMOD International Conference on Management of Data, (1999), 395–406.

[Perry and Lowe 1989] Perry, A., Lowe, D.G.: "Segmentation of textured images"; In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (1989), 319–325.

[Picard et al. 1995] Picard, R. et. al.: "Vision Texture 1.0."; MIT Media Lab, at http://www-white.media.mit.edu/vismod/imagery/VisionTexture/vistex.html. (1995).

[Porter and Canagarajah 1996] Porter, R., Canagarajah, N.: "A robust automatic clustering scheme for image segmentation using wavelets"; IEEE Transactions on Image Processing 5, (1996), 662–665.

[Smith and Chang 1994] Smith, J.R., Chang, S.-F.: "Quad-tree segmentation for texture based image query"; In Proceedings of the 2nd Annual ACM Multimedia Conference, (1994), 279–286.

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

[Wang et al. 1998] Wang, L., Liu, J., Li, S.Z.: "Texture classification using wavelet decomposition with Markov Random Field models"; In Proceedings of the 14th International Conference on Pattern Recognition, (1998), 1613–1615.

[Yang et al. 2004] Yang, G., Hou, Y., Huang, C.: "Texture segmentation algorithm based on wavelet transform and kd-tree clustering"; In Proceeding of the Conference on Robotics, Automation and Mechatronics, (2004), 987–990.