Image Segmentation Using Discrete Cosine Texture Feature

Chi-Man Pun and Hong-Min Zhu

***Abstract*—** In this paper we propose a computational efficient approach for image segmentation based on texture analysis, a 2D discrete cosine transform (DCT) is utilized to extract texture features in each image block. We first split the input image into MxN blocks, calculate the distances between neighbor blocks by a set of largest energy signatures from DCT for each block. Then we merge blocks with smallest distances to form larger regions. The process will repeat until we got desired number of regions. Experimental results show that our proposed method outperforms the existing image segmentation method, especially on efficiency aspect.

***Keywords*—** image segmentation, texture analysis, split and merge, 2D discrete cosine transform.

I. Introduction

IN the framework of digital image engineering, the middle layer, image analysis has the objective of extracting information from an image via image segmentation, object representation and description, feature measurement, and even some higher level tasks such as object classification. Image segmentation is the first step and also one of the most critical tasks of image analysis, which has been extensively studied [1-4] for a few decades due to its applications in computer vision such as:

- Medical imaging (locate tumour).
- Object detection in satellite image.
- Face/fingerprint recognition.
- Traffic monitoring.
- Online image search engine.

While a great number of methods have been proposed in publications trying to handle image segmentation properly, it's still a complicated, unwell solved problem according to some specific view or requirement, such as for general images with no prior knowledge. Image segmentation responsible for extracting semantic foreground objects correctly from a given image, the performance of the subsequent image analysis procedures like retrieval will strongly dependent on the quality of the segmentation. The common image segmentation algorithms are either boundary-based or region-based, which can vary from k-means clustering algorithm to thresholding procedures, to heuristic region growing processes, and to the sophisticated hierarchical methods. In recent years, algorithms based on graph theory have drawn many attentions. Both tree-structured segmentation [5-11] and spectral clustering [12-17] commonly represent an image to be segmented as a graph. The nodes correspond to the image pixels, and the edges represent the relations between pixels with a weight indicates the (dis)-similarity between two pixels. The best known graph-based algorithm, normalized cut[16, 17], can detect object boundaries precisely. However, over-segmented problem will occur if the boundaries are ill-defined, and this approach is computationally expensive, it has been proved to be NP-hard.

In our proposed approach, a 2D discrete cosine transform (DCT) is applied to extract texture features of an image. Texture segmentation is an important task in many applications, or it can be combined with other low-level features segmentation to improve the performance. In our previous work, we have adopted 2D discrete wavelet transform (DWT) as an alternative texture analysis tool for image segmentation[18], which is shown to be efficient. Due to the character that the cosine transform can also represent the texture information, and more important is that it can achieve a high computational efficiency even compared to DWT based approaches, we adopt DCT in our image segmentation system to demonstrate that our implementation greatly improve the efficiency, while achieves a reasonable accuracy of segmented result.

The rest of the paper is organized as follows. In Section II, we give a review on cosine transform on which our segmentation approach is based. In Section III, our textural segmentation algorithm will be described in detail. Experimental results are presented in Section IV. and Conclusions are given in Section V.
.

II. 2D Discrete Cosine Transform

Discrete Cosine Transform (DCT) is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers to orderly express finitely data points in terms of a sum of cosine functions oscillating at different frequencies. DCT is equivalent to DFT of roughly twice the length, operating on real data with even symmetry, where in some variants the input and/or output data are shifted by half a sample. There are few types of DCT variants such as DCT-I, DCT-II, DCTIII-VIII and the most common one is the type

This work was supported in part by the Research Committee of University of Macau under the Grant: RG056/08-09S/PCM/FST.

C.-M. Pun and H.-M. Zhu are with the Department of Computer and Information Science, University of Macau, Macau S.A.R., China. (e-mail: {cmpun, ma86560}@umac.mo).

DCT-II. Its definition for an input image A and output image B is

$$B_{pq} = \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} A_{mn} \cos\frac{\pi(2m+1)p}{2M} \cos\frac{\pi(2n+1)q}{2N},$$

$$0 \le p \le M-1$$

$$0 \le q \le N-1$$

$$\alpha_p = \begin{cases} 1/\sqrt{M}, & p = 0 \\ \sqrt{2/M}, & 1 \le p \le M-1 \end{cases} \qquad \alpha_q = \begin{cases} 1/\sqrt{N}, & q = 0 \\ \sqrt{2/N}, & 1 \le q \le M-1 \end{cases}$$

where M and N are the row and column size of A, respectively. If one applies the DCT to real data, the result is also real. The DCT tends to concentrate information, making it useful for image compression applications.

.

III. PROPOSED IMAGE SEGMENTATION

The procedure involved in the proposed segmentation algorithm contains two main steps: block splitting and region merging. Block splitting process shown in Figure 2 is to split the input image into MxN blocks of small size, and apply DCT to calculate the features for each block which are used for distance calculation. The procedure shown in Figure 3 is to merge blocks to form a specified number of regions based on texture similarity between neighbor blocks.

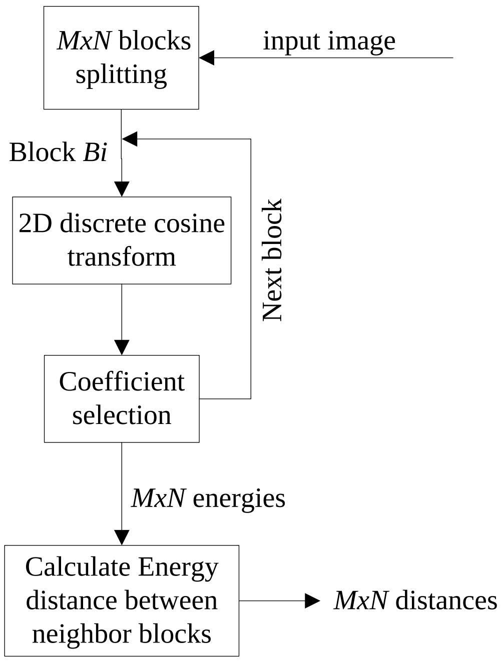

Figure 2 MxN block splitting process

A. *Image splitting and distance calculation*

We didn't include color feature in our algorithm, so the input image will be converted to a gray scale version in advance of any further process in case if it's a true color image. Also the input image will be resized to $2^n x 2^n$ to convenience the splitting process.

Image splitting and distance calculation algorithm split the input image into *MxN* blocks, and calculate the distance between each block and its upper/left neighbors. The number of blocks depends on the block size which is specified as a parameter. The algorithm is given as follows.

Algorithm I – Image Splitting and Distance Calculation

```
Input: image X, blocks size $S_0$, number of coefficients $C_0$
Step 1: calculate the number of blocks $MxN=size(X)/S_0$
Step 2: apply DCT to calculate the energy feature for each of MxN blocks, which is the sum of $C_0$ largest coefficients (exclude the upper-left one).
Step 3: calculate distance between neighbor blocks
    For each block Bi, and its upper/left neighbor Bj
        $dist(i, j) = | E_i - E_j |$    (1)
Output: MxN blocks with distances
```

Where E_i, E_j are energy of block *i, j* respectively.

B. *Region merging*

After the distances have been calculated between all neighbor blocks, region merging procedure will follow to form a specified number of regions. Here we use the notation region to indicate the image segment which contains multiple blocks. The algorithm is described as follows:

Algorithm II – Region Merging

```
Input: MxN blocks $R_0$ with distances $D_0$, minimal region size $N_0$, number of output regions Nr
Step 1: let $R_1=R_0$, if minSize $(R_0)<N_0$, then $R_1=\{ R_1 \mid size(R_1)< N_0, R_1 \in R_0\}$ which are regions with size < $N_0$.
Step 2: find union region $R = \{R_2 \cup R_3 \mid distance(R_2, R3) = min(D_0)\}$
Step 3: delete distance between Ri, Rj and their neighbors. $D_0(Ri, Rj)=[]$, where $Ri \in R$ or $Rj \in R$
Step 4: Region merging. Label(R) =new label L
Step 5: find blocks along common edges between R and its neighbor region.
    for each block Bi in R
        classify Bi's 4 neighbors Bj to regions Rn if Label(Bj)~=L
Step 6: calculate average distance between R and its neighbor regions
        $dist(R, \mathrm{Rn}) = \frac{1}{m}\sum_{k=1}^{m} dist(i, j)$    (2)
```

Step 7: if size (R_0)>*Nr*, go to step 1.
Output: segmented image with *Nr* regions.

The first two steps find two neighbor regions with minimal distance. In our intuitive experience, objects that we are interested are always not with small size compared to image size, to avoid isolated blocks appear as the final regions, we specify a parameter to indicate the minimal region size (the number of blocks should be in a region). So we tend to firstly merge those regions with size less than the minimal size. After such two regions found, we update the labels of blocks in these two regions to be the same, and destroy the distance relationships between these two regions and their neighbor regions which will be updated in Step 5 and step 6. We scan blocks in the new merged region to check their neighbor blocks, and classify all such blocks to corresponding common edges between regions based on their labels. So the distance between the merged region and its neighbor can be calculated by (2) as the mean of distances between block pairs along the common edge. The process above continues until we get required number of regions.

Comparison with the Adaptive Tree-structured Wavelet Transform for Feature Extraction

The development of our texture segmentation is motivated from Chang and Kuo's work, in their research they propose a multiresolution approach based on a tree-structured wavelet transform for texture analysis and classification to overcome the difficulty that the traditional pyramid-type wavelet transform cannot well represent the a texture with significant information often appears in the middle frequency channels. We adopt this technology in our segmentation algorithm according to its excellent efficiency and performance.

Figure 3 shows an example of a decomposition tree with decomposition structure encoding, and the feature vector size is 16. "114" means it's the diagonal coefficient at 3rd level, while its 1st and 2nd level parent nodes are approximation coefficients. The decomposition structure for each sub-image's decomposition will be preserved for comparison, which will be described in detail in section 3.

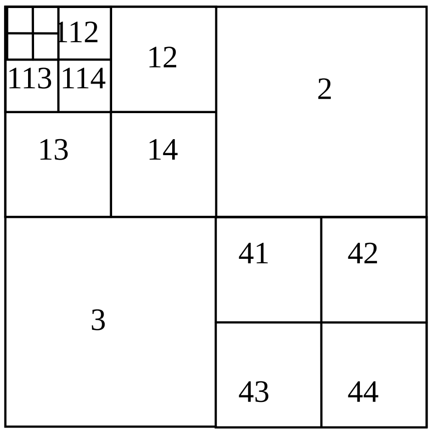

Figure 3 Decomposition structure example

The process to decompose a (sub-) image is shown below. Each decomposition is followed by log energy calculation for each coefficient, and the sub image with maximal log energy will be passed to next decomposition until we get a specified number of features.

Algorithm III – Feature Extraction by Adaptive Tree-structured Wavelet Transform

```
Input: image X, feature vector size fvsize
Step 1: initialize f=[X, logEnergy(X)]
Step 2: recursively decompose X
    While size(f) < fvsize
        subX= Sub image in f with max log energy
        [cA,cH,cV,cD]=dwt2 decomposition of subX
        f = f + [subX/2, logEnergy(cA)] //cH, cV, cD
        calculate decomposition structure
Step 3: Normalize log energies by corresponding sub image size.
Output: X's features of size fvsize, decompose structure
```

The procedure involved in the proposed segmentation algorithm contains two main steps: block splitting and region merging. Block splitting process shown in Figure 4 is to split the input image into *MxN* blocks of small size, and apply adaptive tree-structured wavelet transform to calculate the features for each block which are used for distance calculation. The procedure shown in Figure 5 is to merge blocks to form a specified number of regions based on texture similarity between neighbor blocks.

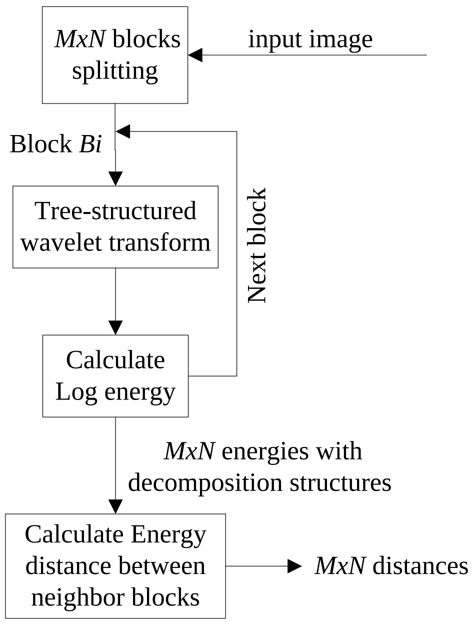
MxN blocks splitting
input image
Block Bi
Tree-structured wavelet transform
Next block
Calculate Log energy
MxN energies with decomposition structures
Calculate Energy distance between neighbor blocks
MxN distances

Figure 4 *MxN* block splitting process

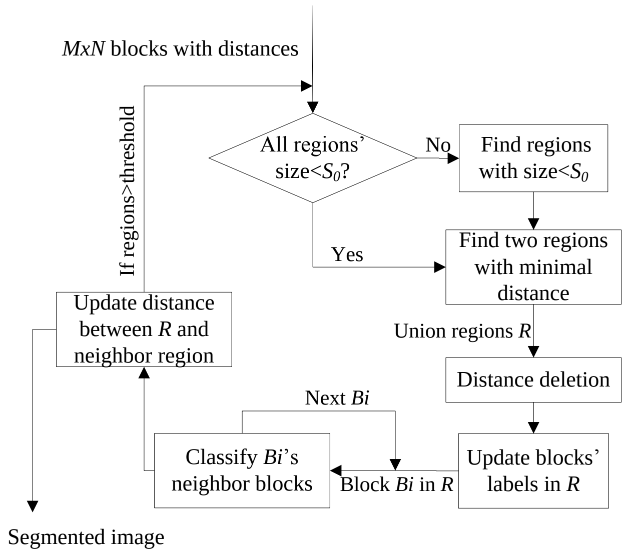
MxN blocks with distances
If regions>threshold
All regions' size<S0?
No
Find regions with size<S0
Yes
Find two regions with minimal distance
Update distance between R and neighbor region
Union regions R
Distance deletion
Next Bi
Classify Bi's neighbor blocks
Block Bi in R
Update blocks' labels in R
Segmented image

Figure 5 Region merging process

IV. Experimental Results

There are several parameters that can affect the performance and efficiency of the algorithm. So firstly we adjust the parameters to select the value that make the result to be the best.

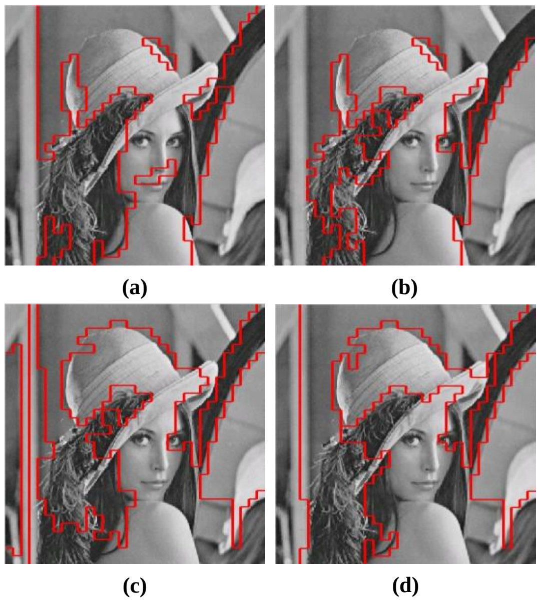

Figure 6 Image segmented into 5 regions with minimal region size of 15 blocks and initial bock size of 8x8. The numbers of DCT coefficients for energy calculation are: a) 10 with 1.39s. b) 20 with 1.35s. c) 30 with 1.45s. d) 40 with 1.35s.

As our first experiment, we concern on the number of DCT coefficients select for energy measurement. Step 2 in Algorithm showed how to calculate the energy of each block. According to the block size for splitting process, the number of largest DCT coefficients varies. We set the other parameters properly as we will demonstrate in following examples: the initial block size is 8x8 and the minimal region size is 15 blocks. As the block has totally 64 pixels and the DCT coefficient will be with the same size, we test selection with 10, 20, 30 and 40 largest DCT coefficients respectively for comparison. Figure 6 shows the segmentation results. We can see that figure 6(c) get the most reasonable segment which can roughly detect the primary boundaries in the Lena image. The preferable number of DCT coefficients is nearly 50 percent of the total.

Those regions with only a few blocks may cause confusion as the minimal region size should no less than 15 blocks. According to the algorithm, we notice that once there is a neighboring region with size less than 15 blocks (even contains 13, 14 blocks for example) and the distance is smaller than those regions of a single block, it will be merged in advance. So the precise definition to the region size should be like: regions with small size will be merged before any two neighboring regions with both sizes greater than the region size threshold.

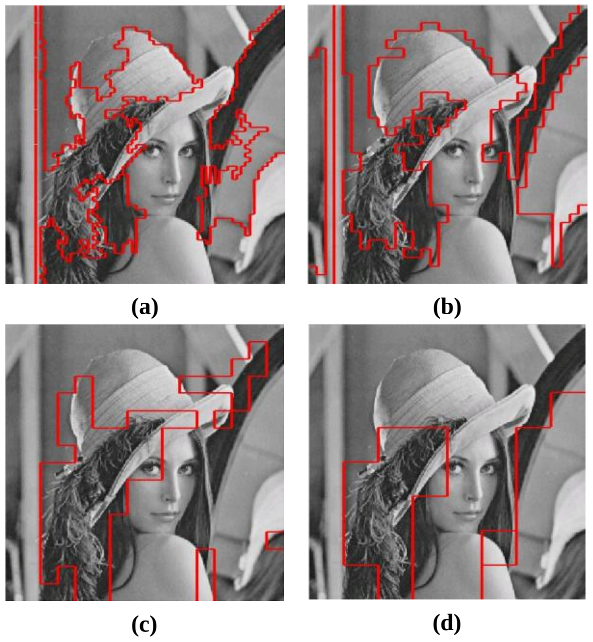

Figure 7 Image segmented into 5 regions with minimal region size of 15 blocks, 50% DCT coefficients are selected while use blocks of size a) 4x4 with 18.83s. b) 8x8 with 1.42s. c) 16x16 with 0.23s. d) 32x32 with 0.064s.

In image splitting and distance calculation process, input image will be splitted into *MxN* blocks where *M* and *N* depends on the specified block size. With large blocks, the following merging process will be quite efficient; however regions in the final segmented image can't achieve smooth boundaries. On the other hand, region merging with too small blocks will have efficiency problem. We adjust the block size to be a reasonable value 8x8 that the boundaries can be smoothly and efficiently segmented, there will be 32x32 blocks for the lena image of size 256x256. Figure 7 shows the segmentation results of different block size, (a) can achieve most smooth and accurate boundaries but it's computationally expensive, (b) is the reasonable one on both performance and efficiency. (c)-(d) with larger block size and can't detect a boundary precisely.

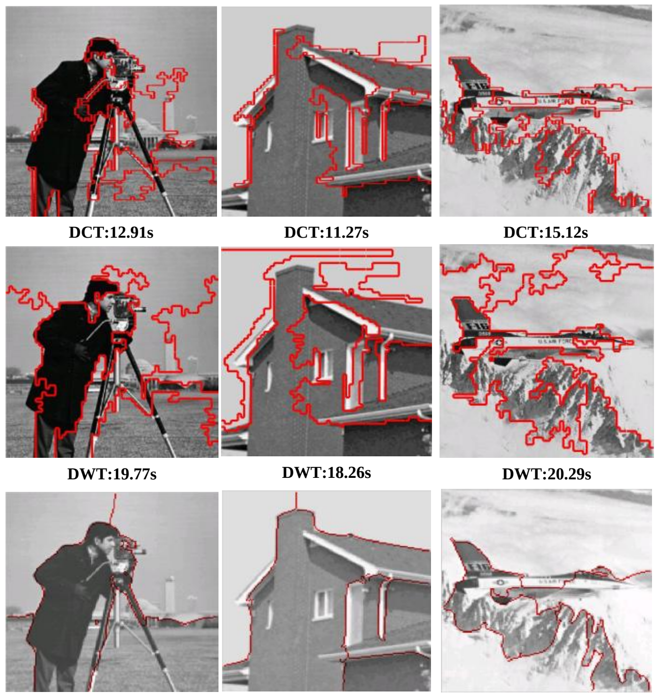

Figure 8 Result of different image segmentation approaches. We adopt the best parameters for our texture segmentation and DWT based approach. First row: our DCT based segment. Second: DWT based segment. Third row: Normalized cut approach. 5 regions are segmented in all the experiments, and the time cost is shown under each example.

In image retrieval applications, we typically try to search for images contain similar objects with the sample image, and the number of interested objects is always not too many. Figure 8 shows the comparison among three different image segmentation approaches: our DCT based texture segmentation; adaptive tree-structured 2D discrete wavelet transform (DWT) based texture segmentation; and the most well know graph segmentation algorithm -- normalized cut. 5 regions are segmented in all experiments and parameters are selected properly for both DCT and DWT based approaches, the sizes of the images are 512x512 compared to Lena image. From the third row in Figure 8 we can see that N-cut can get segmentation with more smooth and precise boundaries while it also has over segmentation problems. Moreover, the second and the third rows show that the DWT based approach greatly improves the efficiency compared to N-cut while can obtain a reasonable segmentation output, which has also been demonstrated in our previous work, and even that the image is resized to 160x160 in N-cut approach. We can also conclude from the first and the second row that the DCT based approach outperforms the DWT one on both efficiency and accuracy aspect. DCT segmentation achieves 30% faster averagely than DWT, while the segmented boundaries are also more precise.

There are several parameters that can affect the performance and efficiency of the DWT image segmentation algorithm. So firstly we adjust the parameters to select the value that make the result to be the best.

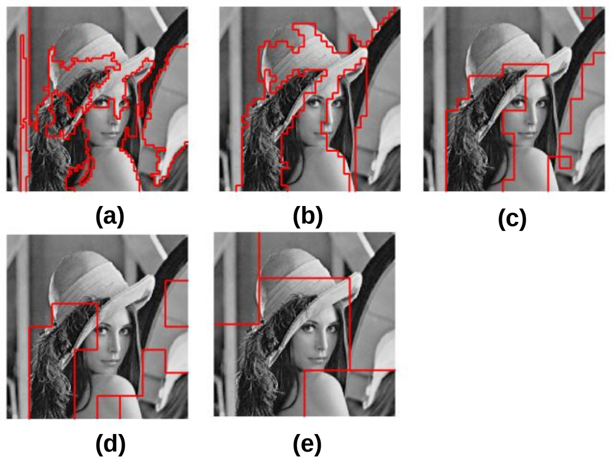

Figure 9 Image segmented into 5 regions with minimal region size of 25 blocks. a)4x4 with 30.25s. b) 8x8 with 3.09s. c) 16x16 with 5.67s. d) 32x32 with 1.48s. e) 64x64 with 0.45s

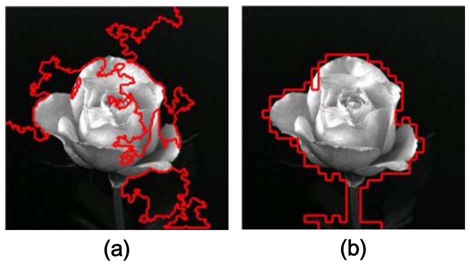

Figure 10 Segment high resolution image (1024x1024) by using a) 128x128 blocks of size 8x8, with 434.48s. b) 32x32 blocks of size 32x32, with 29.58s.

In image splitting and distance calculation process, input image will be splitted into *MxN* blocks where *M* and *N* depends on the specified block size. With large blocks, the following merging process will be quite efficient as there is less number of blocks; however regions in the final segmented image can't achieve smooth boundaries. And the feature vector size is also depends on the block size, large block (greater than 8x8) will be decomposed by structured wavelet transform for multiple times to get more features, which cause a high possibility that decomposition structures between two neighbor blocks are different and thus both blocks need to be re-decomposed based on each other's structure so that distance between can be calculated by the mean. On the other hand, there will be more blocks if the block tends to be small. More merge iterations are performed, the distance between the merged region and its neighbor should be updated in each iteration which is computational expensive.

For our first experiment on the lena image of size 256x256, we adopt different block sizes to compare the accuracy and efficiency, with other parameters set properly. Figure 9 shows the segmentation results, (a) can achieve most smooth boundaries but it's computationally expensive in region merging process, (b) is the reasonable one on both performance and efficiency where the most significant boundaries can be roughly detected. (c)-(e) with larger block size and can't detect a smooth boundary precisely. So it's reasonable to split an image of size 256x256 into 32x32 blocks with each block of size 8x8.

In case if image are of different resolutions, we keep the number of blocks fixed, while the block size proportional to the image size. Figure 10(a) shows the segment result using 8x8 blocks, which is quite inefficient because of the overload merging process, and the over segment is serious while the significant boundary of follower can not be detected. The result in (b) is much preferable even that multilevel decomposition is performed on each block.

Those regions with only a few blocks may cause confusion as the minimal region size should no less than 25 blocks. According to the algorithm, however, we notice that once there is a neighboring region with size less than 25 blocks (even contains 23, 24 blocks for example) and the distance is smaller than those regions of a single block, it will be merged in advance. So the precise definition to the "region size" should be like: regions with small size will be merged before any two neighboring regions with both sizes greater than the region size threshold.

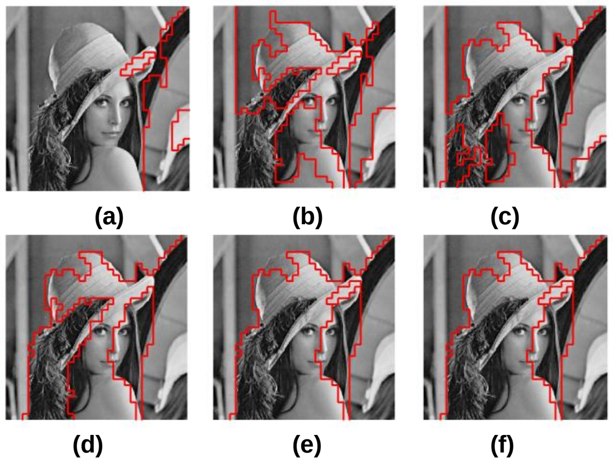

Figure 11 Image segmented into 5 regions, use 8x8 block size and different minimal region size. a) 5 blocks. b) 10 blocks. c) 15 blocks. d) 25 blocks. e) 35 blocks. f) 50 blocks

Another adjustable parameter is the minimal region size which indicates that in the region merging process, regions consist of less than a specified number of blocks are tends to be merged first. In most cases, an object presented in an image that we are interest has a large size. Figure 11 gives the corresponding results according to different minimal region sizes. Since the computation complexity only depends on the block size which was discussed above, here we don't explore it. Comparing region boundary accuracies in (a)-(f), we prefer the segmentation regions with minimal size of 25 blocks since it can detect the most significant boundaries which is not too much rugged and there is less over-segment.

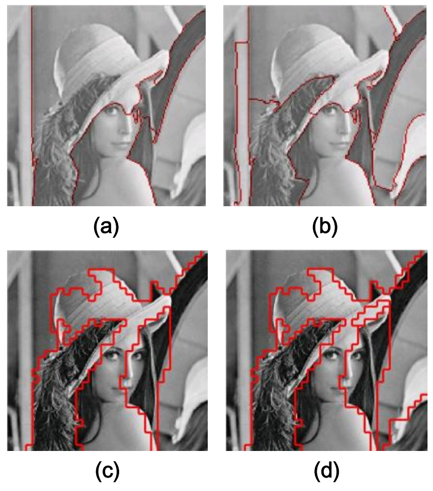

Figure 12 Compare with normalized cut algorithm according to efficiency. We adopt the best parameters for our texture segmentation. a) N-cut 5 regions with 38.13s. b) N-cut 10 regions with 37.77s. c) Texture segmentation 5 regions with 3.36s. d) Texture segmentation 10 regions with 3.52s

. Figure 12 shows the comparison between our wavelet texture segmentation and the most well know graph segmentation algorithm--normalized cut, images are segmented into 5 and 10 regions respectively. Results in (a) and (b) show that N-cut can get segmentation with more smooth and precise boundaries. However, comparing with (c) and (d) which are the results of texture segmentation, N-cut segments more regions on the background and also has over segmentation problem on Lena image. The texture segmentation can separate the hat from the background which N-cut cannot. Another attractive achievement we get is the great efficiency improvement, our proposed algorithm takes 11 times average less time than N-cut. Besides the lena image, Figure 12 presents the segmentation using our wavelet texture based solution and normalized cut on 4 different images, with 5 regions segmented. The first row are the results of our solution, the minimal region size is 25 blocks as well and the block size is proportional to the image size, as described in the first experiment. The second row shows all edges detected by N-cut, and edges that form the required number of regions are highlighted, as shown in the third row. The same as for lena image, boundaries of segmented regions in these images using our solution are less smooth than N-cut results but roughly define the regions locations. However, from the second and the third row we can see that N-cut regards some of the significant boundaries as in lower priority so that they are not highlighted, such as the left side of the camera man's body and the back of the baby head.

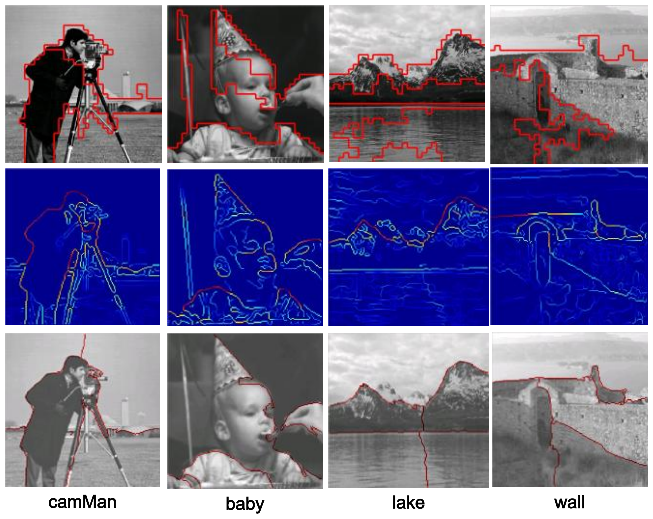

Figure 13 segment result of 4 different images (camera man, baby, mountain/lake, and a wall). Three rows are segment of wavelet texture, edges of N-cut, and segment of N-cut.

Table 1 summarizes the time cost of segmentations in Figure 13. The eigenvector calculation in N-cut is the most compute consuming process, and images are resized to 160x160 in advance since it can not even handle the image of size 256x256. While the number of splitted blocks is fixed (32x32), our texture segmentation on 256x256 images achieves an efficiency of 8 times faster on average compared to N-cut. For the high resolution baby image, multilevel decomposition is performed for each block and a high probability of re-decompositions are required, we can also conclude that our approach outperforms N-cut in the consideration of efficiency aspect.

Image	Image size	N-cut	Wavelet texture	
				Block size
camMan	256x256	30.98s	3.45s	8x8
baby	1024x1024	33.16s	25.58s	32x32
lake	256x256	28.03s	3.51s	8x8
wall	256x256	25.33s	3.50s	8x8

Table 1 comparison of efficiency between N-cut and texture segmentation.

V. Conclusion

We proposed a new method for image segmentation in this paper which applies 2D discrete cosine transform for texture analysis. The algorithm involves two stages: Image Splitting and Distance Calculation, and Region Merging. Proper parameter values are verified through testing in our experiments. We also showed that our proposed method outperforms the other two existing image segmentation methods especially on efficiency aspect, while the segment with reasonable accuracy can also be achieved. Future work may focus on integrate other features such as shape and/or color features in the proposed image segmentation for better performance.

Acknowledgments

The authors would like to thank the referees for their valuable comments. This work was supported in part by the Research Committee of the University of Macau.

References

[1] N. R. Pal, and S. K. Pal, “A review on image segmentation techniques,” *Pattern Recognit.*, vol. 26, no. 9, pp. 1277-1294, 1993.

[2] J. S. Suri, K. Liu, L. Reden *et al.*, “A review on MR vascular image processing algorithms: Acquisition and prefiltering: Part I,” *IEEE Trans. Inform. Technol. Biomed.*, vol. 6, no. 2, pp. 324-337, 2002.

[3] J. S. Suri, K. Liu, S. Singh *et al.*, “Shape recovery algorithms using level sets in 2-D/3-D medical imagery: A state-of-the-art review,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 6, no. 1, pp. 8-28, 2002.

[4] H. Trichili, M. S. Bouhlel, and F. Kammoun, “A review and evaluation of medical image segmentation using methods of optimal filtering,” *J. Test. Eval.*, vol. 31, no. 5, pp. 398-404, 2003.

[5] P. Felzenszwalb, and D. Huttenlocher, “Image segmentation using local variation,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 98-104, 1998.

[6] P. Felzenszwalb, and D. Huttenlocher, “Efficient graph-based image segmentation ” *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167-181, 2004.

[7] Y. Haxhimusa, and W. Kropatsch, “Segmentation graph hierarchies,” *Proc. Structural, Syntactic, and Statistical Pattern Recognition*, vol. 3138 LNCS, pp. 343-351, Aug. 2004.

[8] Y. Haxhimusa, A. Ion, W. Kropatsch *et al.*, “Evaluating minimum spanning tree based segmentation algorithms,” *Proc. CAIP*, vol. 3691, LNCS, pp. 579-586, 2005.

[9] A. Falcão, J. Stolfi, and R. Lotufo, “The image foresting transform: Theory, algorithms, and applications,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 19-29, Jan. 2004.

[10] A. Falcão, P. Felipe, and A. Paulo, “Image segmentation by tree pruning,” *Proc. 17th Brazilian Symp. Computer Graphics and Image Processing*, pp. 65-71, 2004.

[11] J. Ding, R. Ma, S. Chen *et al.*, “A fast directed tree based neighborhood clustering for image segmentation,” *Proc. 13th Int. Conf. Neural Information Processing, Part II*, vol. 4233, LNCS, pp. 369-378, 2006.

[12] R. Zabih, and V. Kolmogorov, "Spatially coherent clustering using graph cuts," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 437-444, 2004.

[13] Y. Y. Boykov, and M.-P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images," *Proc. IEEE Int. Conf. Computer Vision*, vol. 1, pp. 105-112, 2001.

[14] Y. N. Andrew, M. Jordan, and Y.Weiss, "On spectral clustering: Analysis and an algorithm," *Proc. Adv. Neural Information Processing Systems*, vol. 14, pp. 849-856, 2002.

[15] Y.Weiss, "Segmentation using eigenvectors:Aunifying view," *Proc. IEEE Int. Conf. Computer Vision*, vol. 2, pp. 975-982, 1999.

[16] J. Shi, and J. Malik, "Normalized cuts and image segmentation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 731-737, 1997.

[17] J. Shi, and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 8, pp. 888-905, Aug. 2000.

[18] C.-M. Pun, and H.-M. Zhu, "Image Segmentation Using Adaptive Tree-structured Wavelet Transform," *Proceedings of the 2009 Sixth International Conference on Computer Graphics, Imaging and Visualization*, pp. 290-294, Aug. 2009.

Chi-Man Pun received the B.Sc. and M.Sc. degrees from the University of Macau in 1995 and 1998 respectively, and Ph.D. degree in Computer Science and Engineering from the Chinese University of Hong Kong in 2002. He currently is an associate professor at the Department of Computer and Information Science of the University of Macau. His research interests include Content-Based Image Indexing and Retrieval, Digital Watermarking, Pattern Recognition, and Computer Vision.