A Novel Texture Synthesis Algorithm Using Patch Matching by Fuzzy Texture Unit

G.Venkata Rami Reddy, Associate professor
School of Information Technology
JNT University Hyderabad,
Hyderabad,India
gvr_reddi@yahoo.co.in

Dr.V Vijaya Kumar
Dean & Prof. Dept. of Computers,
G.I.E.T, Rajahmundry,
A.P., India.
vakulabharanam@hotmail.com

B. Sujatha, Research Scholar
& Assoc. Prof., Dept. of CSE,
G.I.E.T, Rajahmundry,
A.P., India.
birudusujatha@gmail.com

***Abstract*— Texture is an important spatial feature useful for identifying objects or regions of interest in an image. This paper presents a novel texture characterization method based on Fuzzy Texture Unit (FTU) for texture synthesis. The proposed fuzzy texture characterization approach, takes into account the vagueness introduced by noise and the different caption and digitization processes, for defining the texture unit, by which texture synthesis is obtained. FTU extracts local texture information from the pixels for characterizing the textural aspect of a digital image and FTU detects the classes having highest relevance for deciding the existence of a concrete textural feature within an image by which patches can be identified easily. A patch that satisfies the minimum error criterion computed using L2norm is selected as the candidate patch. Edging two patches along the minimum error boundary cut is considered to be one of the most efficient candidate patches till recently. However, in real textures, especially the semi-stochastic ones, perfect cuts do not exist; therefore, the defects due to the inconsistency between the patches become an important problem for the synthesized texture quality. For this reason the present paper adopts a modified MEBC determine the boundary between the patches. Experimental results on wide range of texture samples demonstrate that the performance of the present approach is superior to many other existing methods of texture synthesis.**

Keywords- Fuzzy Texture Unit, Error Tolerance, Minimum Error Boundary Cut.

I. INTRODUCTION

Generally, the goal of texture synthesis is to generate a new synthetic texture according to a given sample texture so that this new synthetic texture is (1) similar to the original texture in terms of human perception and (2) sufficiently different from the given texture. Since texture is an important cue in human visual perception, texture processing has been one of the most actively studied areas in computer graphics, computer vision and image processing [1-7]. Technologies used for texture synthesis can be divided to two categories [8]: procedural texture synthesis [9] and example-based texture synthesis [10]. Procedural texture synthesis, using an algorithm, intends to create a realistic representation of natural elements such as wood, marble, granite, metal, stone, and others, so they obtained texture is a computer generated image. The example-based texture synthesis generates a high resolution texture from a small input example. The continuity and similarity of texture structures are assured in the example-based texture synthesis.

Example-based texture synthesis can be categorized as [11]: pixel-based methods, patch-based methods and tiling-based methods. Pixel-based algorithms, the synthesized texture are generated pixel by pixel. Each pixel is the result of a search of the given example, using the neighborhood already synthesized. The main advantage of pixel-based methods [12–22] is the ability to control textures at pixel level. However, these methods are sensitive to the size of the window during the search for pixels in the example images and when deciding a pixel's neighbors in the synthesized texture. More recent work [23] provides a controllable parallel method at pixel level that can utilize modern GPUs. Patch-based algorithms, the texture is generated patch by patch and cutting paths are sought which minimize overlap errors between patches. Specific algorithms include image quilting using dynamic programming [24], graph-cut texture [25], iterative optimization based on Markov Random Fields [26] and the earlier lapped textures technique [27]. Patch-based algorithms tend to preserve the global structure, but they sometimes introduce local artifacts in the regions of overlap between neighboring patches. Patch-based texture synthesis has been shown to produce high-quality textures faster than pixel-based approaches.

As a typical representative of patch-based synthesis algorithms, image quilting [28] solves the problem of stitching neighboring blocks by means of Minimum Error Boundary Cut (MEBC). The MEBC algorithm has been used to achieve a better transition between the adjacent patches. For each synthesis unit, the overlapping block error between the old synthesized block and the candidate block in its neighboring position is estimated, and a block with minimum error is chosen as the newly synthesized patch. Then the MEBC through the error surface can be computed via dynamic programming and the newly chosen block with an irregular optimal path is made. Although image quilting algorithm holds a good performance particularly for semi-structured textures and stochastic textures, there still exists some typical problems especially when the input texture does not contain enough variability. Fuzzy Texture Unit (FTU) can capture the micro-structure of local brightness variations and therefore can describe complementary features based on the distribution of local spatial patterns. Inspired by the local properties of FTU and the peculiarity of MEBC we combine both features together and present in this paper a new method for texture synthesis. The main idea is to synthesize a new texture by taking FTU image as the temporary image for selecting overlap regions satisfies MEBC. Regions block matching and the best match boundary by using FTU. The proposed algorithm is to improve synthesizing efficiently.

The rest of the paper is organized as follows. In section 2, FTU is briefly described. Section 3 introduces the novel texture synthesis method using FTU and MEBC. In section 4, we present experimental results. Finally, we conclude in section 5.

II. Fuzzy Texture Unit

Greater or lesser quantities are further quantized using fuzzy logic based approach as follows. Here, two more levels of comparison are introduced. A texture unit is represented by eight elements, each of which has only five possible values {0, 1, 2, 3 and 4} obtained from a neighborhood of 3×3 image region. In Base5, Eqn. (1) is used to determine the elements, Ei of texture unit. The texture number representation is shown in Fig.1:

$$E_i = \begin{cases} 0 & \text{if } V_i < V_0 \text{ and} V_i < x \\ 1 & \text{if } V_i < V_0 \text{ and} V_i > V_x \\ 2 & \text{if } V_i = V_0 \\ 3 & \text{if } V_i > V_0 \text{ and} V_i > y \\ 4 & \text{if } V_i > V_0 \text{ and} V_i < y \end{cases} \quad \text{for } i = 1,2,\ldots 8 \qquad (1)$$

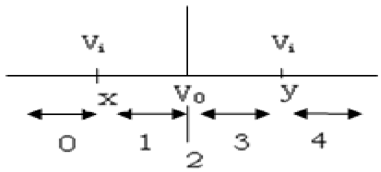

Figure 1. Fuzzy Texture Number (base-5) Representation

where x, y are user-specified values.

The FTU number (FTUn5) for the pixel p is computed in base5 as given in Eqn.2.

$$FTU_{n5}(p) = \sum_{i=1}^{8} E_i * 5^{(i-1)/4} \qquad (2)$$

The total texture units range from 0 to 194.

For example

90	130	145
160	140	200
100	140	250

→

0	1	4
4		3
0	2	3

FTU= {0, 1, 4, 3, 3, 2, 0, 4}, FTU_{n5} = 124

Figure 2. Transformation of a 3×3 neighborhood to a FTU

III. TEXTURE SYNTHESIS BASED ON FTU

A novel method for patch based synthesis using a combination of image quilting and the local properties of FTU is proposed in this paper. During the process of texture synthesis, FTU image of the input texture sample is firstly generated and used as a temporal image for overlapping block error estimation and block matching. To reduce visual artifacts, the adjacent patches are overlapped along the boundary regions. Because of overlapping, the successively synthesized patches will be slightly differed from the standard N×N pixels. In the following, we will firstly show the entire description of the proposed method.

A. Methodology

Let I_i be the input texture and the output image is IO. For a synthesized texture of 2N×2N pixels, the synthesis starts with a patch size of N/4×N/4 pixels. The proposed approach picks the size of overlapped areas between adjacent blocks by using FTU image. The complete process is for synthesizing a new texture image I_O is as follows:

Step 1: Calculate the FTU image using Eqn.2 for the original image.

Step 2: 2a) For every location, search in the FTU image for a set of blocks that satisfy the overlap constraints by using modified MEBC along horizontal and vertical direction as given in Algorithm 3.3.1 with minimum error tolerance. Pick the two blocks from the corresponding locations of the input texture image.

2b) Paste the patches from the input texture sample Ii on to the output image IO based on scan line order.

Step 3: Select the next region such that it overlaps with the previous selected region. If the new selected regions satisfy MEBC then take the patch from the corresponding location of the original image and paste in the output image based on scan line order.

Step 4: Repeat step 3 until the whole output image IO is synthesized.

B. Calculate the Minimum Error Surface by using L2-Norm

Before manipulating with the overlap region of the new patch and the constructed patch, an error surface is calculated. Different cost functions will result in different error surfaces; typically used metrics are L2-norm and squared root difference. In our paper L2-norm is considered. The formula for this distance between a point x (x1, x2, etc.) and a point y (y1, y2, etc.) is given in Eqn.3:

$$e_{ij} = \sqrt{\sum_{i,j=1}^{n} (x_{i,j} - y_{i,j})^2} \qquad (3)$$

For color image texture synthesis, the FTU code of each color channel is calculated separately and therefore the resulted FTU code for the input texture is a color image. For the calculation of the minimum error, Euclidean distance metric is used on three color channels as given in Eqn.4:

$$e_{ij} = \sqrt{\sum_{i,j=1}^{n} + R_{(x_{ij}-y_{ij})^2} + G_{(x_{ij}-y_{ij})^2} + B_{(x_{ij}-y_{ij})^2}} \qquad (4)$$

Here, R, G, B represent values of three color channels, i and j are two dimensional indices for two boundary zones.

C. Determine Minimum-Cost Boundary Cut

Once the error surface is constructed, the minimum-cost boundary cut algorithm is applied to it. Once the best patch is chosen the Minimum Error Boundary Cut (MEBC) operation is performed to determine the boundary between the patches. Efros and Freeman (2001) [1] performed three types of MEBC operation to locate the boundary between the adjacent patches in the output texture. They are MEBC in vertical direction, MEBC in horizontal direction and L shaped MEBC. The determination of MEBC along the vertical and horizontal directions is easier where the L shaped MEBC is difficult. The computation difficulty in the L shaped MEBC is overcome by performing the MEBC operation in two stages, first along the horizontal direction then along the vertical direction. By performing this extended MEBC, the quality of the output texture has been increased to a greater extent in the region of the horizontal overlap. This is because a better boundary along the horizontal direction is obtained, compared with the conventional L-shaped MEBC operation.

The Minimum Error Boundary Cut (MEBC) is calculated both along the vertical and horizontal directions. In general, the MEBC along the vertical direction is calculated using the formula given in Eqn.5

$$E_{ij} = e_{ij} + \min(E_{i-1,j-1}, E_{i-1,j}, E_{i-1,j}) \quad (5)$$

Similarly the MEBC along the horizontal direction is calculated using the formula given in Eqn.6

$$E_{ij} = e_{ij} + \min(E_{i-1,j-1}, E_{i-1,j}, E_{i+1,j}) \quad (6)$$

where Eij is the cumulative minimum error in the ith and jth pixel.

The modified MEBC operation is performed in two different types. They are MEBC along horizontal direction and MEBC along vertical direction.

D. Algorithm for MEBC along Horizontal Direction

Step1: Find the pixel-to-pixel difference between the pixels in the boundary regions of the previously synthesized patch and the candidate patch.

Step 2: With the pixel-to pixel difference values obtained in step 1 form a new patch called difference patch.

Step 3: Identify the pixels with the lowest value in the last row of the difference patch.

Step 4: Locate three pixels in the previous row adjacent to the pixel identified in step (3).

Step 5: Choose the pixel with the lowest value from the identified three pixels of step 4.

Step 6: If the pixels identified in step 5 touches the left or right boundary of the difference patch then select the two adjacent pixels of previous row.

Step 7: Repeat step 4 and 5 for the remaining rows of the difference patch.

Step 8: A boundary line is made along the vertical direction through the pixels of the lowest difference values.

Step 9: The pixels on the left side of this line form the boundary portions of the previously synthesized patch and on the right side are the boundary portions of the candidate patch.

E. Algorithm for MEBC along Vertical Direction

Step 1: Find the pixel- to- pixel difference between the pixels in the boundary regions of the previously synthesized patch and the candidate patches.

Step 2: With the pixel-to pixel difference values obtained in step 1 form a new patch called difference patch.

Step 3: Identify the pixels with the lowest value in the last column of the difference patch set.

Step 4: Locate three pixels in the previous column adjacent to the pixel identified in step 3.

Step 5: Choose the pixel with the lowest value from the identified three pixels of step 4.

Step 6: If the pixels identified in step 5 touches the top or bottom boundary of the difference patch then select the two adjacent pixels of the previous column.

Step 7: Repeat step 4 and 5 for the remaining columns of the difference patch.

Step 8: A boundary line is made along the horizontal direction through the pixels of the lowest difference values.

Step 9: The pixels on the top side of this line form the boundary portions of the previously synthesized patch and on the bottom side are the boundary portions of the candidate patch.

IV. EXPERIMENTAL RESULTS

The proposed method is tested on tested on VisTex and other standard texture databases of color and gray images, each of size N×N. For the input image of size N×N the experimental results generated the synthesized image of size 2N×2N, by considering the patch size of N/4×N/4 with the overlap region is of six pixels width. Figure 3 shows some of the results for image texture synthesis. In Fig. 3, the input texture images are in the left side and the synthesized results generated by the proposed method are in the right side.

The quantifying parameters such as relative entropy and mean of the average difference in histogram are used to measure the performance of the texture synthesis algorithms. In order to validate the synthesized results measurably, we use metric called relative entropy. The relative entropy of P with respect to Q, is called the Kullback-Leibler distance, is defined by Eqn.7

$$D(P, Q) = \sum_k P_k \log \quad (7)$$

where E is the relative entropy computed by original image P and the synthesized result Q. P(x) and Q(x) are the probability of each gray scale in image P and Q separately. Although E(P,Q)   E(Q,P), so relative entropy is therefore not a true metric, it satisfies many important mathematical properties.

The obtained results for the various textures under FTU patch based texture synthesis method are shown in Table 1. From Table 1, it is observed that the entropy parameter is approximately same for the input and synthesized images. The second parameter which is mean of the average difference in histogram is very low. Therefore the inference is that input and output images are very much similar.

TABLE I. PERFORMANCE COMPARISON OF PATCH BASED TEXTURE SYNTHESIS USING FTU

Texture Type	Input Relative Entropy	Output Relative Entropy	Mean of Average difference in Histogram
Fruit-1	5.67	5.06	0.00054
Fruit-2	5.98	5.59	0.00043
Fruit-3	5.66	5.33	0.00032
Fruit-4	5.87	5.96	0.00022
Fruit-5	5.91	5.56	0.00012
Fruit-6	5.18	4.99	0.00034
Stone-1	6.23	6.12	0.00041
Stone-2	6.75	5.95	0.00032
Stone-3	6.34	6.73	0.00036
Metal-1	5.76	5.54	0.00047
Metal-2	6.01	5.89	0.00043
Metal-3	5.99	5.01	0.00025
Water	5.67	5.43	0.00026
Brick	5.45	5.12	0.00051
Grass	6.02	5.87	0.00065
Leaves-1	6.23	6.67	0.00045
Leaves-2	5.99	6.09	0.00068
Leaves-3	6.01	6.54	0.00061

V. CONCLUSIONS

While the algorithm is particularly effective for semi-structured textures (which were always the hardest for statistical texture synthesis), the performance is quite good on stochastic textures as well. Based on the results proposed approach are at par with state-of-the-art patch-based synthesis techniques like image quilting and graph cuts. On the other hand, proposed technique maintains the shape as well as orientation of the keys as it rotates the texture. The validation test results are shown in Table 1. Two parameters entropy and mean of the average difference in histogram are considered for testing. The test results are satisfactory and the proposed algorithm works well for any texture image.

In this paper we have introduced a novel patch based texture synthesis algorithm which uses the effective combination of properties for FTU descriptor and the modified MEBC. FTU feature can provide with more abundant information of local micro structure variability about the input texture. A FTU based approach used for selecting patches in patch based texture synthesis. The best patch selection is done with L2-norm and also a modified minimum error boundary cut is used to preserve the texture feature along the boundaries and also synthesis conflicts between neighboring patches were substantially reduced.

Figure 3. Results for texture synthesis. For each texture, the input is on the left and the output on the right

ACKNOWLEDGMENT

I would like to thank to Prof. Rameswara Rao, Vice Chancellor for encouraging research Programmes. The authors would like to express their gratitude to Sri K.V.V. Satyanarayana Raju, Chairman, and Sri K. Sasi Kiran Varma, Managing Director, Chaitanya group of Institutions for providing necessary Infrastructure. Authors would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] D. J. Heeger, J. R. Bergen, "Pyramid-based texture analysis/synthesis," AGM Proceedings of SIGGRAPH,pp. 229-238, 1995.
[2] J. S. De Bonet, "Multiresolution sampling procedure for analysis and synthesis of texture images," ACM Proceedings of SIGGRAPH, pp. 361-368, 1997.
[3] G. R. Cross and A. K. Jain, "Markov random field texture models," IEEE Rans. On Potte? Analysis and Mochzne Intell., vol. 5, No. 1, pp. 25-39, 1983.
[4] Efros and T. Leung, "Texture synthesis by nonparametric sampling," IEEE Int. Conf. on Computer Vision, vol. 2, pp. 1033-1038, 1999.
[5] L. Y. \Vei and hl. Levoy, "Fast texture synthesis using tree-structured vector quantization," ACM Proceedings of SIGGRAPH, 2000.
[6] Y. Q. Xu, B. N. Guo and H. Shum, "Chaos mosaic: fast texture synthesis," Microsoft Research Report, 2000.
[7] P. J. Burt and E. H. Adelson, "Multiresolution spline with application to image mosaics," AChl Transactions on Graphics, 2(4), pp 217-236, 1983.
[8] Chongyang Ma, Li-Yi Wei, Baining Guo, Kun Zhou. ‘‘Motion Field Texture Synthesis’’, ACM Transactions on Graphics (SIGGRAPH Asia 2009).
[9] EBERT D. S., MUSGRAVE F. K., PEACHEY. D., PERLIN K.,WORLEY S.: Texturing and Modeling: A Procedural Approach. Morgan Kaufmann Publishers Inc., 2002.
[10] KWATRA V., WEI L.-Y., LEFEBVRE S., TURK G.: Course 15: Example-based texture synthesis. In SIGGRAPH ’07 courses (2007).
[11] X. Zhang and Y. J. Kim, ‘‘Efficient texture synthesis using strict wang tiles,’’ Graph. Models, vol. 70, no. 3, pp. 43-56, 2008.
[12] D. Garber, Computational models for texture analysis and texture synthesis, in: PhD Dissertation, University of Southern California, 1981.
[13] A. Efros, T. K. Leung, Texture synthesis by non-parametric sampling, in: Proceedings of ACM SIGGRAPH, 1999, pp. 1033–1038.
[14] K. Popat, R. Picard, Novel cluster-based probability model for texture synthesis,Visual Communication and Image Processing 2094 (1993) 756–768.
[15] L. Wei, M. Levoy, Fast texture synthesis using tree-structured vector quantization, in:Proceedings of SIGGRAPH, 2000, pp. 479–488.
[16] M. Ashikhmin, Synthesizing natural texture, in: Proceedings of the Symposium onInteractive 3D Graphics, 2001, pp. 217–226.
[17] X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, H.-Y. Shum, Synthesis of bi-directionaltexture functions on arbitrary surfaces, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH) 21 (3) (2002) 665–672.
[18] S. Zelinka, M. Garland, Towards real-time texture synthesis with the jump map, in: Proceedings of the 13th Eurographics workshop on Rendering, 2002, pp. 99–104.
[19] J. S. DeBonet, Multiresolution sampling procedure for analysis and synthesis of texture image, in: Proceedings of ACM SIGGRAPH, 1997, pp. 361–368.
[20] Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, D. H. Salesin, Image analogies, in: Proceedings of ACM SIGGRAPH, 2001, pp. 327–340.
[21] J. Zhang, K. Zhou, L. Velho, B. Guo, H.-Y. Shum, Synthesis of progressively-variant textures on arbitrary surfaces, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH) 22 (3) (2003) 295–302.
[22] L. Tonietto, M. Walter, Towards local control for image-based texture synthesis, in: Proceedings of the 15th Brazilian Symposium on Computer Graphics and Image Processing, 2002, p. 252.
[23] S. Lefebvre, H. Hoppe, Parallel controllable texture synthesis, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH) 24 (3) (2005) 777–786.
[24] A. Efros, W. T. Freeman, Image quilting for texture synthesis and transfer, in:Proceedings of ACM SIGGRAPH, 2001, pp. 341–346.
[25] KWATRA, A. SCHODL, I. ESSA, G. TURK, A. BOBICK, GRAPHCUT TEXTURE: IMAGE AND VIDEO SYNTHESIS USING GRAPH CUTS, ACM TRANSACTIONS ON GRAPHICS (PROCEEDINGS OF ACMSIGGRAPH) 22 (3) (2003) 277–286.
[26] V. Kwatra, I. Essa, A. Bobick, N. Kwatra, Texture optimization for example-based synthesis, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH) 24 (3)(2005) 795–802.
[27] E. Praun, A. Finkelstein, H. Hoppe, Lapped textures, in: Proceedings of ACM SIGGRAPH, 2000, pp. 465–470.
[28] A Efros, and W.T. Freeman, “ Image quilting for texture synthesis and transfer,” ACM Transactions on Graphics (SIGGRAPH 2001), 20(3), Aug. 2001, pp. 341-346.

AUTHORS PROFILE

G.Venkata rami reddy received the M.Tech. (CSE) degree from JNT University Hyderabad in 1998. He is working in JNT University since 2000. Presently he is working as an Associate Professor in Dept of CSE in School of Information Technology, JNT University Hyderabad. He is more than 11 years of experience in teaching and Software Development. . He is pursuing his Ph.D. in the area of Image processing from JNT University Hyderabd in Computer Science and Engineering under the guidance of Dr. M. Anji Reddy. He is presented more than 6 National and International journal and conference. His areas of interests are image processing, computer networks, analysis of algorithms.

Vakulabharanam Vijaya Kumar received integrated M.S. Engg, degree from Tashkent Polytechnic Institute (USSR) in 1989. He received his Ph.D. degree in Computer Science from Jawaharlal Nehru Technological University (JNTU) in 1998. He has served the JNT University for 13 years as Assistant Professor and Associate Professor and taught courses for M.Tech students. He has been Dean for Dept of CSE and IT at Godavari Institute of Engineering and Technology since April, 2007. His research interests include Image

Processing, Pattern Recognition, Network Security, Steganography, Digital Watermarking, and Image retrieval. He is a life member for CSI, ISTE, IE, IRS, ACS and CS. He has published more than 150 research publications in various National, Inter National conferences, proceedings and Journals.

B.Sujatha received the B.Tech. degree from JNT University, Kakinada in 1997 and received her M. Tech. (Computer Science & Engineering) from Andhra University in 2002. She is having 10 years of teaching experience. Presently she is working as an Assoc. Professor in GIET, Rajahmundary. She has published 15 research publications in Inter National Journal. She is a member of SRRF-GIET, Rajahmundry. She is pursuing her Ph.D from Mysore University in Computer Science under the guidance of Dr. V.Vijaya Kumar. Her research interest includes Image Processing and Pattern Recognition. She is a Life member of ISCA.