International Journal of Innovative Computing, Information and Control
Volume **9**, Number **3**, March **2013**

ICIC International ©2013 ISSN 1349-4198
pp. 1201–1210

FEATURE-BASED 3D TEXTURE SYNTHESIS APPROACH

Chin-Chen Chang[1,*], Zen-Chung Shih[2], Chia-Wen Chang[2]
Wen-Kai Tai[3] and Der-Lor Way[4]

[1]Department of Computer Science and Information Engineering
National United University
Miaoli 360, Taiwan
*Corresponding author: ccchang@nuu.edu.tw

[2]Institute of Multimedia Engineering
National Chiao Tung University
No. 1001, University Road, Hsinchu 300, Taiwan

[3]Department of Computer Science and Information Engineering
National Dong Hwa University
No. 1, Sec. 2, Da Hsueh Road, Shoufeng, Hualien 97401, Taiwan

[4]Department of New Media Art
Taipei National University of Arts
No. 1, Hsueh-Yuan Road, Peitou, Taipei 112, Taiwan

Received January 2012; revised May 2012

Abstract. *Numerous 3D textures have been synthesized from 2D textures by image-based approaches. However, the quality problems still exist for 3D texture synthesis. Further improvements are required to extract more reliable texture features. A well-known texture feature extraction approach is the grey level co-occurrence probability (GLCP) approach. In this paper, a feature-based approach incorporating GLCP features from a 2D texture is presented for 3D texture synthesis. For color feature extraction, appearance vectors are used to replace RGB color values. For GLCP feature extraction, the statistical features including entropy, contrast, and correlation are extracted to exploit spatial relationships. Moreover, a weighting scheme is introduced to obtain weighted color and GLCP features for neighborhood matching in the synthesis process. The experimental results show that the proposed approach performs well in terms of the synthesis quality.*
Keywords: 3D texture, Texture analysis, Texture synthesis, Grey level co-occurrence probabilities

1. **Introduction.** Textures can be used to describe a wide range of surface properties. Texture analysis and synthesis is essential to computer vision, image processing and computer graphics [3,5,14,15,17]. Texture mapping [5,11] has been applied widely in computer graphics. It is the first method used for 3D surface texturing and an effective technique of simulating surface detail at relatively low cost. This technique can easily add the appearance of detail to a large variety of object surfaces. Without increasing 3D model complexity, it enhances the visual realism of 3D objects by adding fine texture details. However, it suffers problems such as distortion and discontinuity.

3D textures can be used to solve the above problems. A 3D texture is defined as colored points in 3D to represent a real-world material. Users do not need to find a parameterization for the surface of the object to be textured. Furthermore, 3D textures provide texture information inside the entire volume. Procedural approaches have been proposed to synthesize 3D textures. However, it is difficult to express procedurally a desired texture for users. It only can be defined for a limited set of textures. To address

these problems, numerous 3D textures have been synthesized from 2D textures by image-based approaches [2,4,8,10,13,16,20]. Several methods [2,4,10,16,20] used three orthogonal slices based on neighborhood matching to synthesize 3D textures. They are applicable to a wide variety of textures. However, the quality problems still exist for 3D texture synthesis. Further improvements are often required to extract more reliable texture features.

The grey level co-occurrence probability (GLCP) approach [3,6] is an important texture feature extraction method. It is a second-order approach for extracting texture features. Given an image window, the GLCPs characterize the probability of any grey level occurring spatially relative to any other grey level. From this distribution, several parameters are used for generating features. Most investigations do not consider the use of the GLCP texture features for 3D texture synthesis. In this paper, a feature-based approach incorporating GLCP features from a 2D texture is proposed to improve the quality of 3D texture synthesis. For color feature extraction, appearance vectors are used to replace RGB color values. For GLCP feature extraction, the statistical features including entropy, contrast, and correlation are extracted to exploit spatial relationships. Moreover, a weighting scheme is introduced for computing weighted color and GLCP features for neighborhood matching in the synthesis process. The proposed approach can synthesize desired 3D textures.

The rest of this paper is organized as follows: in Section 2, related works are reviewed. In Section 3, a feature-based approach for synthesizing 3D textures from a 2D texture is presented. Section 4 shows the results. Finally, conclusions are discussed in Section 5.

2. **Related Works.**

Several approaches have been proposed for synthesizing 2D textures. The approach developed by Ashikhmin [1] applied a texture synthesis algorithm to natural textures. This simple and efficient implementation allowed users to input interactive deformations during the synthesis process using a painting-like interface. Lefebvre and Hoppe [12] developed a texture synthesis algorithm based on neighborhood matching to achieve parallelism when deforming synthesis textures. Their approach included a coordinate up-sampling step and a correction approach. They also introduced a method of enhancing the resolution of coarse synthesized results. Turk [19] presented another method of synthesizing surface textures. In their approach, a hierarchy of points from low to high density is created over a given surface. The points are then connected to form a hierarchy of meshes. The user then specifies a vector field over the surface that indicates texture deformation.

Recently developed 3D texture synthesis approaches include Jagnow et al. [8], who used a stereological approach to synthesizing 3D textures from 2D textures. Their approach first analyzes the materials of spherical particles and then applies it to arbitrarily-shaped particles. This approach also provides a systematic method of predicting material structures. Wei [20] first adapted 2D neighborhood matching synthesis schemes to 3D textures. The key idea is to consider three 2D exemplars for each direction from an input 2D texture. In each voxel of the output 3D texture, three interleaved 2D neighborhoods are extracted. The best matches are found independently in each of the three 2D exemplars. Qin and Yang [16] presented a method for generating 3D textures from input examples. They used gray-level aura matrices for neighborhood matching. Their method characterizes each input example as a set of aura matrices and generates a 3D texture from multiple view directions. For each voxel of the output 3D texture, they only considered the pixels on the three orthogonal slices for neighborhood matching.

Kopf et al. [10] introduced a 3D texture synthesis method from 2D exemplars. They extended 2D texture optimization techniques to synthesize 3D textures. For each voxel,

they only considered the neighborhood coherence in three orthogonal slices, and iteratively increased the similarity between the output 3D texture and the input exemplar. Dong et al. [4] introduced a method to restrict synthesis to a subset of the voxels for 3D texture synthesis. They synthesized a volume from a set of pre-computed 3D-candidates. Their pre-computed 3D-candidates improve synthesis efficiency and reduce the dependency chain required to compute voxel colors. Chen and Wang [2] presented a method for synthesizing 3D textures from 2D exemplars. They adopted an optimization framework with the k-coherence search and the discrete solver for 3D texture synthesis. They integrated with two kinds of histogram matching methods called position and index histogram matching. Their approach can generate desired results. Jiang et al. [9] proposed a 3D texture synthesis approach using 2D exemplar and procedural noise. Their approach first uses procedural noise to design the movement path for exemplar in 3D space. Then, their approach uses the pixels of exemplar to color the 3D space through their trajectory for 3D texture synthesis.

From the above approaches, new advances in 3D texture synthesis approaches can synthesize 3D textures based on an input exemplar. Unfortunately, the quality problems still exist for 3D texture synthesis. Further improvements are often required to extract more reliable texture features.

3. **The Proposed Approach.** The flowchart of the proposed approach is shown in Figure 1. First, input a 2D texture and repeat the texture thrice as three directional exemplars T_x, T_y and T_z. Without loss of generality, exemplar T_x is only considered Then, in feature vector generation, weighted color and GLCP features are extracted from exemplar T_x. Third, for similarity set construction and 3D-candidates generation, it finds the three pixels most similar to each pixel and pre-computes a set of 3D-candidates for each pixel of exemplars T_x to build the relationship between the three exemplars. Finally, in synthesis process, the 3D pyramid synthesis method [4,7] is applied to obtain synthesis results.

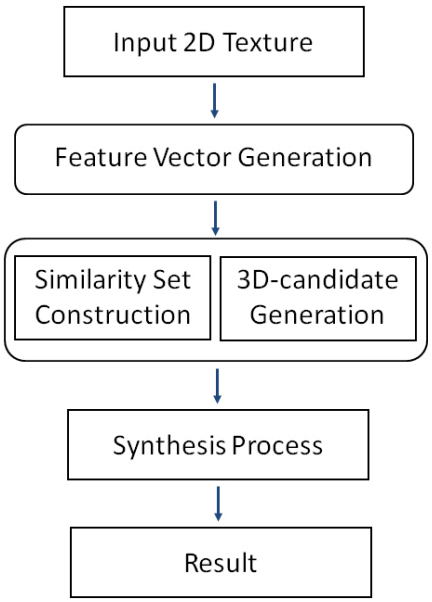

FIGURE 1. Flowchart of the proposed approach

3.1. **Feature vector generation.**

3.1.1. *Color features.* Traditionally, 3D texture synthesis using RGB colors for neighborhood matching needs larger neighborhood size and numerous data. If RGB colors are replaced by appearance vectors [12], quality of synthesis results is improved. Additionally, appearance vectors are continuous and low-dimensional than RGB colors for neighborhood matching. Therefore, the texture data values in color space are transformed into feature vectors in appearance space.

Hence, in the proposed approach, RGB colors in a $w_c \times w_c$ window of each pixel of exemplar T_x are used to construct color feature vectors. In the experiments, w_c is set as 5. Then, the principal component analysis (PCA) is performed to reduce the dimensions of the color feature vectors to form a transformed exemplar $T_x{}'$ in appearance space since PCA is commonly-used technique reducing the number of dimensions without much loss of information. In the experiments, color feature vectors are reduced to 8-dimensional feature vectors by PCA. Hence, a color feature vector with 8 dimensions is extracted for each pixel in $T_x{}'$.

3.1.2. *Grey level co-occurrence probability features.* Haralick et al. [6] introduced a GLCP method to extract texture features. Given a spatial window within a grey level image, the GLCP method computes the conditional joint probabilities, C_{ij}, of all pairwise combinations of grey levels i and j, given the inter-pixel displacement vector (δ, θ), which represents the relationship of the pixel pairs, where δ is the inter-pixel distance, and θ is the orientation.

The set of GLCPs is defined as

$$C_{ij} = \frac{P_{ij}}{\sum_{i,j=0}^{G-1} P_{ij}},$$

where P_{ij} is the frequency of occurrence between two grey levels, i and j, for the given displacement vector (δ, θ) and the given window size; G is the number of quantized grey levels. The probabilities are stored in a grey level co-occurrence matrix (GLCM) of size $G \times G$, where index (i, j) in the matrix is probability C_{ij}. Then, statistics are applied to the GLCM to generate texture features which are assigned to the center pixel of the spatial window.

In the proposed approach, exemplar T_x is first converted to a grey texture and then GLCP features from the grey image are extracted for each pixel in $T_x{}'$. Extracting GLCP features from the grey texture requires the parameters as follows: window size (n_x, n_y) image quantization (G), displacement vector (δ, θ) and statistic selection.

For window size (n_x, n_y), large window sizes are necessary to gather sufficient data to characterize local texture regions and small window sizes will result in poor co-occurrence probabilities. In the proposed approach, a spatial window of size 15×15 is used to compute GLCP features, as shown in Figure 2. For image quantization (G), larger values of G ($G \approx 64$) are excessive. Therefore, G is set as 32 in the experiment. The selection of displacement vector (δ, θ) depends on the characteristics of the grey texture for each direction and the influence of the distance of pixel pairs. In the proposed approach, the interpixel distance is set as 1 and four common orientations (0°, 45°, 90°, and 135°) are used by heuristics.

For statistic selection, many of the statistics suggested by Haralick et al. [6] produce highly correlated texture features which are not desirable. Clausi [3] studied the relationship of the statistical parameters and concluded that entropy, contrast and correlation are a preferred set of statistical parameters for extracting texture features Entropy means the level of spatial disorder of gray levels in the GLCM. Contrast indicates the contrast of

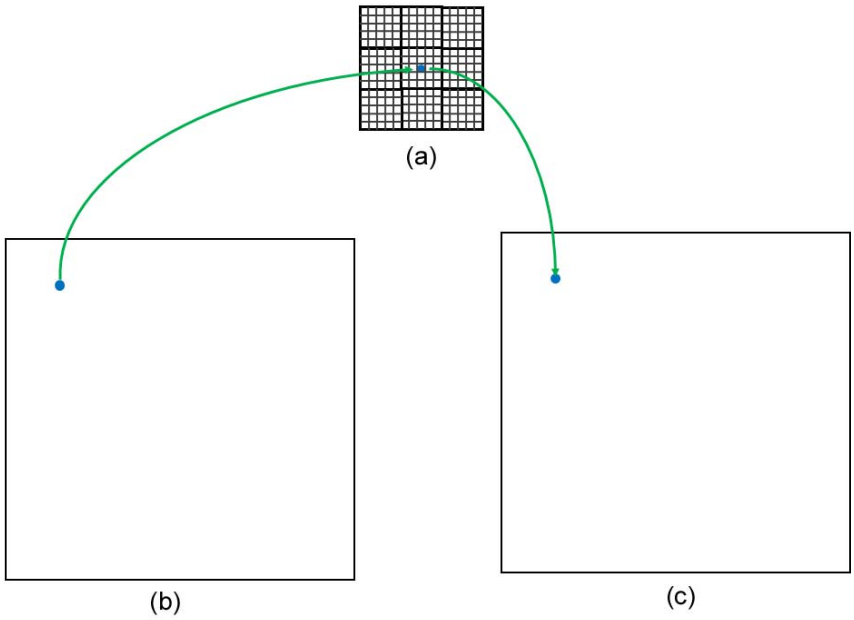

FIGURE 2. (a) Window of size 15×15 for each pixel, (b) grey texture converted from the input 2D texture, and (c) GLCP features computed from the window for each pixel

spatial disorder of gray levels in the GLCM. Correlation indicates the linear dependency of gray levels on those of neighboring pixels in the GLCM. Therefore, the three statistics are used in the proposed approach as follows:

$$Entropy: \quad -\sum_{i,j=0}^{G-1} C_{ij} \log C_{ij},$$

$$Contrast: \quad \sum_{i,j=0}^{G-1} C_{ij} \left(i-j\right)^2,$$

$$Correlation: \quad \sum_{i,j=0}^{G-1} \frac{\left(i-\mu_x\right)\left(j-\mu_y\right) C_{ij}}{\sigma_x \sigma_y},$$

where μ_x and μ_y are the means in the x- and y-directions, respectively; σ_x and σ_y are the standard derivations in the x- and y-directions, respectively.

Hence, a GLCP feature vector with 12 dimensions (4 orientations for each statistic parameter) is extracted for each pixel in $T_x{}'$.

3.1.3. *Weighted color and GLCP features.* After color and GLCP features are extracted, weighted color and GLCP features are proposed for 3D texture synthesis. Different weights are assigned to color and GLCP features. The weighted color and GLCP feature vector for each pixel in T_x' is defined as

$$WF = \left(weight_{CF} \cdot CF, \quad weight_{GLCPF} \cdot GLCPF\right),$$

where CF and $GLCPF$ are the color feature vector and GLCP feature vector, respectively; $weight_{CF}$ and $weight_{GLCPF}$ are the weights for color features and GLCP features, respectively. In the experiments, $weight_{CF}$ and $weight_{GLCPF}$ are set as 0.6 and 0.4 for

color textures, respectively; $weight_{CF}$ and $weight_{GLCPF}$ are set as 0.4 and 0.6 for grey textures, respectively.

3.2. **Similarity set construction and 3D-candidate generation.** For similarity set construction, a similarity set for each pixel in $T_x{}'$ is constructed to increase the effectiveness of neighborhood matching in the synthesis process. Based on the k-coherence search method [18], the k most similar pixels in $T_x{}'$ for each pixel p are searched. Moreover, based on the principle of coherence synthesis [1], searching candidates from the $n \times n$ neighborhoods of pixel p in T_x' can accelerate the synthesis process, where n is a user-defined parameter. Therefore, in the proposed approach, the k most similar pixels from the $n \times n$ neighborhoods of pixel p in $T_x{}'$ are searched to construct a similarity set $C^l_{1\ldots k}(p) = \{C^l_1(p), C^l_2(p), \ldots, C^l_k(p)\}$, where l is a pyramid level, $C^l_1(p) = p$ and k is a user-defined parameter. This technique can accelerate neighborhood matching because it does not need to search each pixel in $T_x{}'$ for neighborhood matching during synthesis process. In the experiments, k is set as 3 and n is set as 7.

For 3D-candidate generation, based on Dong et al. [4], a set of 3D-candidates for each pixel of transformed exemplars is pre-computed to build the relationship between the three exemplars.

3.3. **Synthesis process.** In the proposed approach, the 3D pyramid texture synthesis approach is applied to synthesize 3D textures. It is divided into three steps: the upsampling step, the jitter step, and the correction step.

For the upsampling step, the proposed approach synthesizes one voxel to an $m \times m \times m$ 3D texture, $S_0 - S_L$, where $L = \log_2^m$, where m is the size of the target 3D texture. A 3D texture S is synthesized from a voxel. Each voxel stores a triple of 2D coordinates of voxel v. A voxel is first constructed and the values of a triple of 2D coordinates $(1, 1)$, $(1, 1)$, $(1, 1)$ are assigned as its coordinates. The coordinates of parent voxels for the next level are then upsampled. Each of the 8 children is assigned parent coordinates plus a child-dependent offset.

For the jitter step, after upsampling the coordinates, the upsampled coordinates are jittered to achieve deterministic randomness. The upsampled coordinates at each level are added a jitter function value to perturb them.

The correction step uses the jittered coordinates and modifies them to recreate neighborhoods similar to those in transformed exemplars. For each voxel v, the feature vectors of warped neighborhoods are collected to obtain neighborhood vectors. Then, the voxel u most similar to voxel v from the transformed exemplars is searched based on neighboring matching by comparing the neighborhood vectors. The triple of voxel i'_1 is replaced with the triple of voxel u.

4. **Results.** Several experiments are conducted to evaluate the effectiveness of the proposed approach. The proposed algorithm is implemented in MATLAB running on a PC with 3.00GHz and 3.00GHz Core2 Extreme CPU and 8.0GB of system memory. In the experiments, input 2D textures are all 128×128 and the output 3D textures are all 128×128×128. In the experiments, it is not easy to adequately determine the parameters to obtain the optimal synthesis results for the proposed algorithm. The parameters are set to obtain the desired results by heuristics.

The input 2D texture in Figure 3(a) is a stochastic and marble-like texture and the input 2D texture in Figure 4(a) is a kind of stochastic textures. In Figure 3(c), the result synthesized by the proposed approach preserves more white features in whole area. But in Figure 3(b), it preserves less white features. Moreover, the result of the proposed approach is more colorful than that of the method using color features. In Figure 4(c),

the result of the proposed approach preserves complete features in whole area. But in Figure 4(b), it does not preserve features. From Figures 3 and 4, the proposed approach can preserve more features than the method using color features.

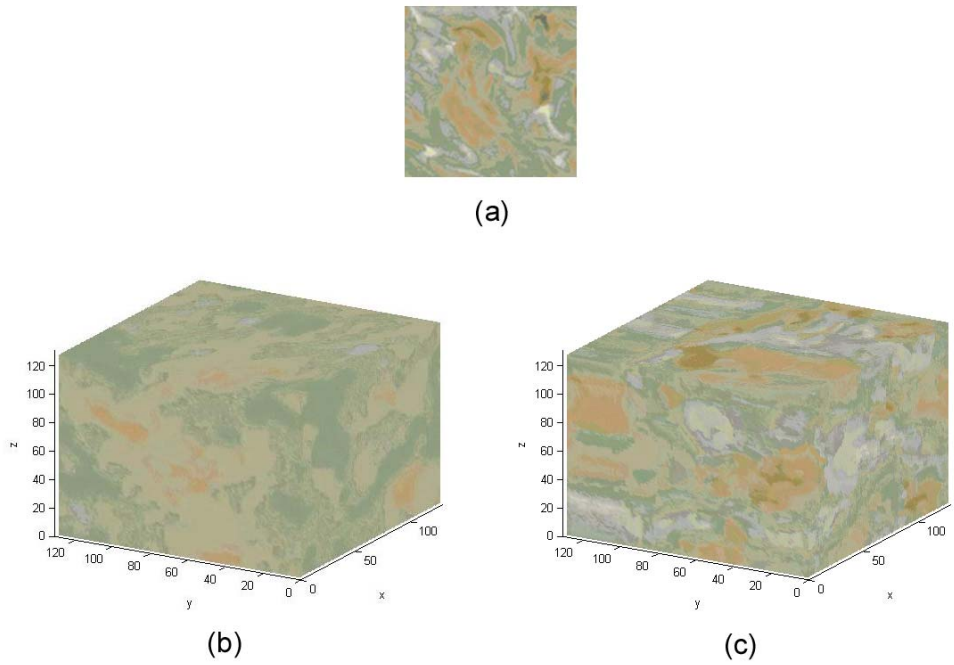

FIGURE 3. (a) Input 2D texture, (b) 3D texture synthesized by the method using color features, and (c) 3D texture synthesized by the proposed approach

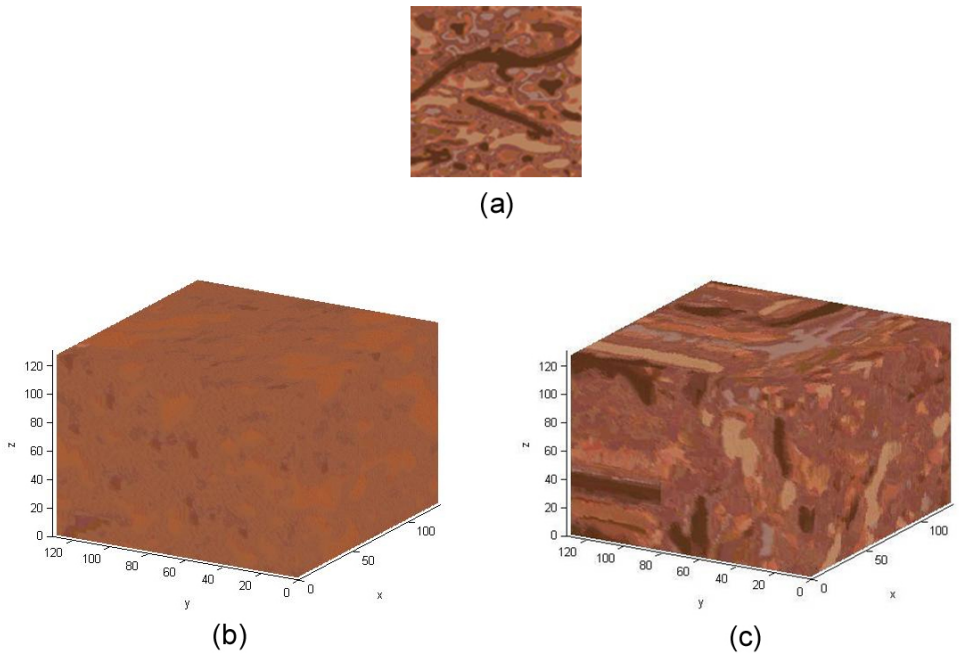

FIGURE 4. (a) Input 2D texture, (b) 3D texture synthesized by the method using color features, and (c) 3D texture synthesized by the proposed approach

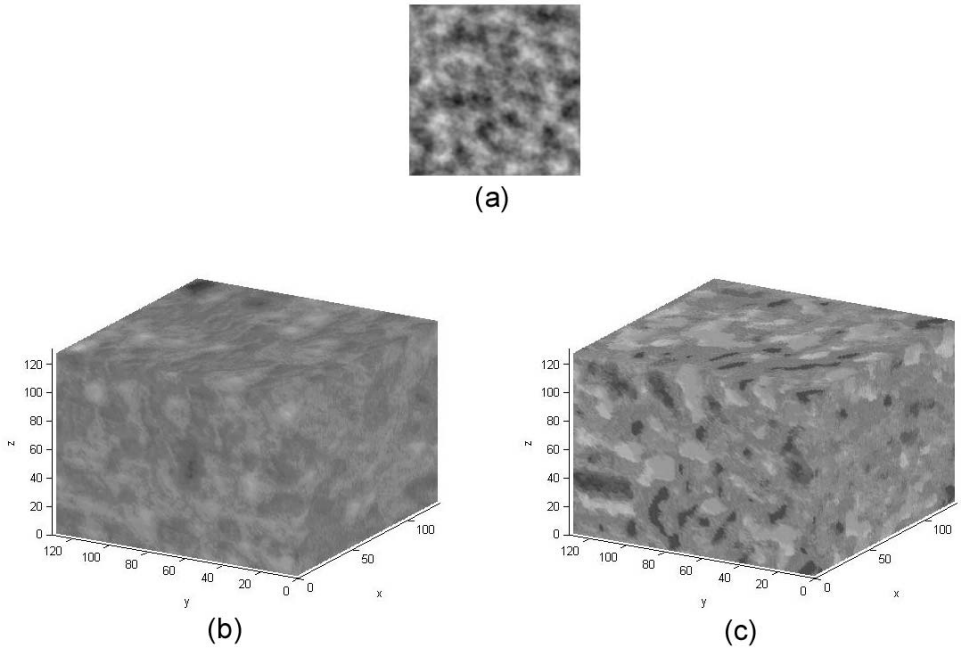

FIGURE 5. (a) Input 2D texture, (b) 3D texture synthesized by the method using color features, and (c) 3D texture synthesized by the proposed approach

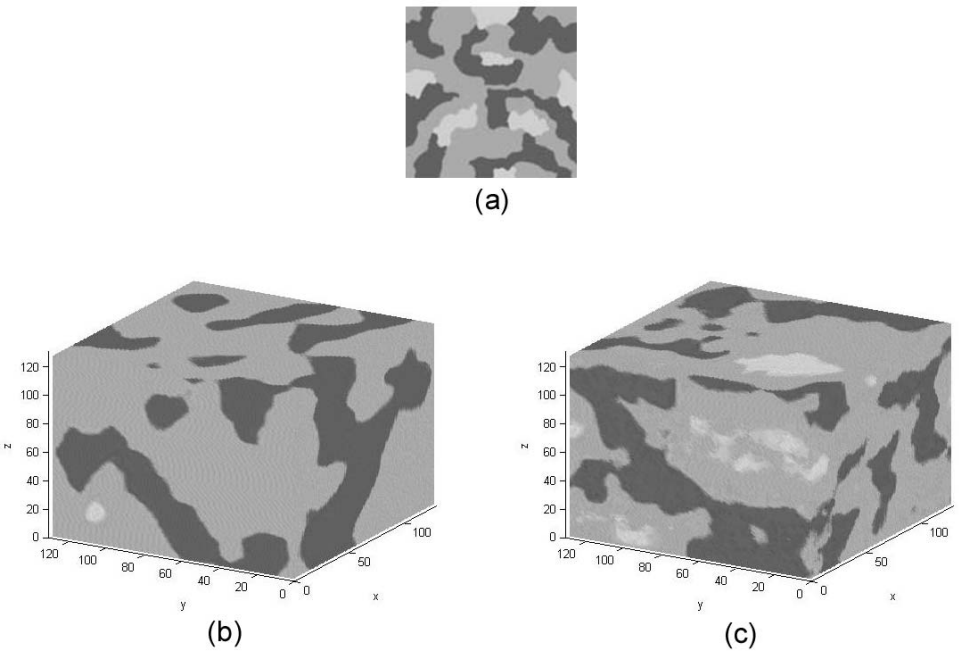

FIGURE 6. (a) Input 2D texture, (b) 3D texture synthesized by the method using color features, and (c) 3D texture synthesized by the proposed approach

The input 2D textures in Figure 5(a) and Figure 6(a) are grey level textures. They are stochastic patterns. As shown in Figure 5(c), the result of the proposed approach preserves more black features in whole area. However, in Figure 5(b), it preserves less black features. As shown in Figure 6(c), the result of the proposed approach preserves

white features in whole area. However, in Figure 6(b), it preserves less white features. From Figures 5 and 6, the quality of the proposed approach is better than that of the method using color features.

5. **Conclusions.** A feature-based 3D texture synthesis approach from a 2D texture has been proposed. The weighted color and GLCP features are introduced for more accurate neighborhood matching in the synthesis process. Different weights are assigned for color and GLCP features according to the characteristics of the input 2D texture. The proposed approach can synthesize desired 3D textures for a wide range of textures.

In the future, the proposed approach will be applied to synthesize textures changing with time in the 3D space. Moreover, an adaptive window of each pixel will be applied to extract more accurate GLCP features.

Acknowledgment. The authors would like to thank the National Science Council of Taiwan for financially supporting this research under Contract No. NSC 99-2221-E-239-033-.

REFERENCES

[1] M. Ashikhmin, Synthesizing natural textures, *ACM SIGGRAPH Symposium on Interactive 3D Graphics*, pp.217-226, 2001.

[2] J. Chen and B. Wang, High quality solid texture synthesis using position and index histogram matching, *The Visual Computer*, vol.26, no.4, pp.253-262, 2009.

[3] D. Clausi, An analysis of co-occurrence texture statistics as a function of grey level quantization, *Canadian Journal of Remote Sensing*, vol.28, no.1, pp.45-62, 2002.

[4] Y. Dong, S. Lefebvre, X. Tong and G. Drettakis, Lazy solid texture synthesis, *Eurographics Symposium on Rendering*, vol.27, no.4, 2008.

[5] D. S. Ebert, F. K. Musgrave, K. P. Peachey, K. Perlin and S. Worley, *Texturing & Modeling: A Procedural Approach*, 3rd Edition, Academic Press, 2002.

[6] R. Haralick, K. Shanmugam and I. Dinstein, Textural features for image classification, *IEEE Transactions on Systems, Man and Cybernetics*, vol.3, no.3, pp.610-621, 1973.

[7] D. J. Heeger and J. R. Bergen, Pyramid-based texture analysis synthesis, *ACM SIGGRAPH 1995*, vol.14, no.3, pp.229-238, 1995.

[8] D. Jagnow, J. Dorsey and H. Rushmeier, Stereological techniques for solid textures, *ACM SIGGRAPH 2004*, vol.23, no.3, pp.329-335, 2004.

[9] J. Jiang, Z. Huang and J. Zheng, Solid texture synthesis based on 2D exemplar and procedural noise, *Proc. of 2011 International Conference on Multimedia Technology*, pp.735-738, 2011.

[10] J. Kopf, C. W. Fu, D. Cohen-Or, O. Deussen, D. Lischinski and T. T. Wong, Solid texture synthesis from 2D exemplars, *ACM SIGGRAPH 2007*, vol.26, no.3, 2007.

[11] V. Kraevoy, A. Sheffer and C. Gotsman, Matchmaker: Constructing constrained texture maps, *ACM SIGGRAPH 2003*, vol.22, no.3, pp.326-333, 2003.

[12] S. Lefebvre and H. Hoppe, Appearance-space texture synthesis, *ACM SIGGRAPH 2006*, vol.25, no.3, 2006.

[13] N. Pietroni, P. Cignoni, M. Otaduy and R. Scopigno, A survey on solid texture synthesis, *Journal of Latex Class Files*, vol.6, no.1, 2007.

[14] N. Pietroni, P. Cignoni, M. Otaduy and R. Scopigno, Solid-texture synthesis: A survey, *IEEE Computer Graphics and Applications*, vol.30, no.4, pp.74-89, 2010.

[15] Y. Qiao, Z. Lu, C. Zhao and S. Sun, Feature based on modulus maxima of wavelet frame representation for texture retrieval, *International Journal of Innovative Computing, Information and Control*, vol.3, no.6(B), pp.1657-1666, 2007.

[16] X. Qin and Y. H. Yang, Aura 3D textures, *IEEE Transactions on Visualization and Computer Graphics*, vol.13, no.2, pp.379-389, 2007.

[17] M. T. Suzuki, T. Shibata, Y. Yaginuma and H. Kodama, Extended 3D HLAC pattern features for solid textures, *Proc. of 2011 IEEE International Conference on Signal and Image Processing Applications*, pp.541-546, 2011.

[18] X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo and H. Shum, Synthesis of bidirectional texture functions on arbitrary surfaces, *ACM SIGGRAPH 2002*, vol.21, no.3, pp.665-672, 2002.
[19] G. Turk, Texture synthesis on surfaces, *ACM SIGGRAPH 2001*, vol.20, no.3, pp.347-354, 2001.
[20] L. Y. Wei, *Texture Synthesis by Fixed Neighborhood Searching*, Ph.D. Thesis, Standford University, 2002.