Texture Synthesis Based on Multiple Seed-blocks and Support Vector Machines

Junyu Dong Ran Wang Xinghui Dong
College of Information Science and Engineering
Ocean University of China
Qingdao, China
dongjunyu@ouc.edu.cn

***Abstract*—We introduce a new method for texture synthesis based on multiple seed-blocks and support vector machines (SVM). First the sample texture is used to train the SVM model with class labels assigned to gray levels. During the synthesis process, each time we generate one patch in the left-to-right order in the result texture. The size of each patch is smaller than that of the sample, and we search a seed-block in the already generated patches to ensure the synthesized patch has similar texture characteristics as the sample. Support vector machines are used to generate pixel values within each patch. The advantage of using SVM is that the sample is not required during the synthesis stage since it has been modeled by a linear model. Unlike previous work in [8], which can only synthesize highly structured texture, the proposed method can successfully synthesize both random and structured textures. It is also extended to synthesize 3D surface texture or Bidirectional Texture Functions (BTF).**

Keywords-texture synthesis; multiple seed-blocks; support vector machines; cutting curve; 3D surface texture

I. INTRODUCTION

Research into texture synthesis is normally concerned with learning and generation of 2D texture images [1, 2]. Recent approaches based on non-parametric models, e.g. synthesis based on a tiles set [13, 14], can produce good results for many types of texture, especially for random texture. However, the disadvantage is that they require the input sample and the set of tiles during the whole synthesis process. In contrast, parametric methods, which normally employ statistical models and can provide compact representation for the sample, may have problems to synthesize highly structured textures.

As a machine learning method, support vector machines (SVM) has been widely used for classification and regression [3, 4]. It was firstly introduced for texture synthesis in [8]. The advantage of using support vector machines is that the sample can be simply modeled by a linear model and the input-output mapping function can be generated from a set of labeled training data. Meanwhile, the sample is not required during the synthesis stage. However, the original method in [8] failed to synthesize texture with random patterns (see Figure 1).

This paper proposes a new method that combines support vector machines with multiple seed-blocks to synthesize new texture. The training data is generated after extracting features and is used to train the SVM model. Meanwhile, we optimize the paramaters by means of cross validation. There are several kernel functions for choice and the polynomial kernel is employed based on a variety of experiments. The synthesis process is in the raster order and a left-to-right manner. Each time when we synthesize an output patch in the result texture, we use a small seed-block retrieved from the already generated patches to guide the SVM model to predict pixel values within the patch. This constraint ensures the output patch has the same texture characteristics as the input sample. Adjacent patches are connected using an optimal cutting curve.

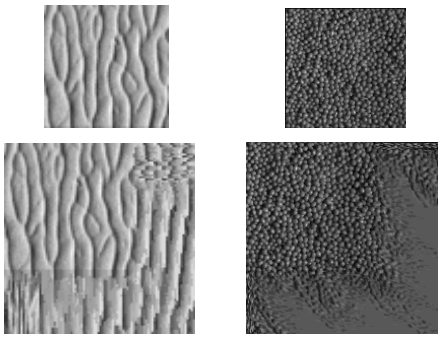


Figure 1. Failed synthesis results produced in [8]. The samples are random textures.

The proposed method can successfully synthesize both random and structured textures. Since real-world texture normally comprises rough surface geometry and complex reflectance, we also extend the method for 3D surface texture synthesis.

The rest of this paper is organized as follows: Section 2 introduces 2D texture synthesis based on multi-seeded patches and support vector machines. Section 3 presents image extrapolation through multi-seeded patches and SVM. Section 4 describes the extension of the 2D synthesis algorithm for 3D surface texture synthesis. We show the experimental results in section 5, and finally we make conclusion in section 6.

978-1-4244-6516-3/10/$26.00 ©2010 IEEE

II. TWO-DIMENSIONAL TEXTURE SYNTHESIS BASED ON MULTI-SEEDED PATCHES AND SVM

Support vector machines were originally introduced for texture synthesis in [8], where the input sample is used to train a SVM model and pixel gray values are used as class labels. The synthesis process is simply converted to a classification process. However, the original method failed to synthesize random textures while being able to successfully generate highly structured textures. The reason is that pixel values in a random texture are more stochastic, and the SVM model can only accurately predict the pixel values in a patch whose size is smaller than that of the sample.

In order to synthesize both random and structured textures, we propose to combine multi-seeded patches with support vector machines. We employ optimal seed-blocks for synthesizing output patches in the output texture. Figure 2 shows the flow chart of the proposed algorithm which comprises the five stages: (1) extracting features and generating training data, (2) training the SVM model, (3) predicting pixel values in the output patch through SVM, (4) finding the best cutting curve in the overlap region between adjacent output patches, and (5) synthesizing the next seeded patch. This process is repeated until all patches in the result texture are generated.

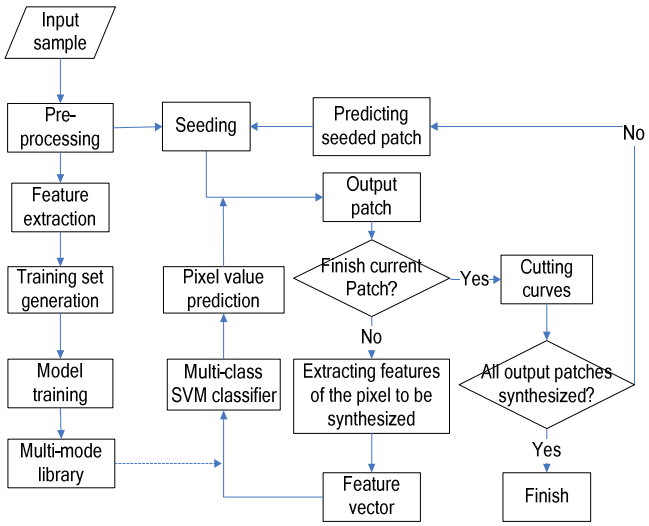


Figure 2. Flow chart of the 2D synthesis algorithm.

A. Feature Extraction and Training Data generation

We extract the feature and generate the training data in the same manner as [8]. We simply use all pixel values in the pre-defined neighborhood to form a feature vector so as to avoid expensive computation of additional feature extraction. The size of the neighborhood can be different for various kinds of texture images, whereas, the neighborhood must contain primary texture elements. The pixel intensity is used as the class level for texture synthesis. In this paper we also use the "L" shaped neighbor as in [5] to form the corresponding feature vector, and the synthesis process starts from the top-left location in each patch and in a raster order.

B. SVM Model Training

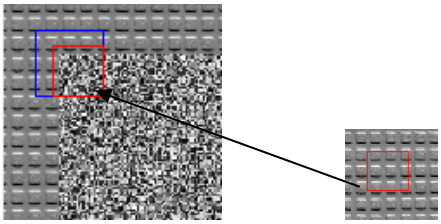
After generating the training data set, we train the multi-class SVM classifier and use the cross-validation method to optimize the parameters so that an SVM model can be produced. In this paper, we employ the K-fold cross-validation. Firstly the original sample is randomly partitioned into K subsamples, one of the subsamples is retained as the test data and the other K-1 subsamples are remained as training data. The cross-validation process is repeated K times. Each subsample is used once as the test data. The cross-validation process optimizes the parameters and considers all the different attributes of the input data to avoid overfitting and underfitting.

C. Output Patch Synthesis

We initialize the output texture with the pre-defined size. As shown in Figure 1, we find that the results are not good by only using SVM for synthesis of random texture. However, those pixels whose coordinates are within the sample's range can be predicted accurately. Thus, if we want to correctly synthesize a pixel value in the output, we must make sure that the pixel and its neighborhood is contained the coordinate range of the sample.

In our method, we repetitively generate an output patch of a fixed size, and we ensure the size of each individual output patch is not beyond the sample's.

For the first patch to be synthesized, we select a small block which contains at least two texture elements from the upper-left corner of sample image and seed the block in the output patch. Pixels in this output patch are synthesized in a raster order and all pixel values in the L-shape neighbor are used to form a feature vector. The SVM model is then used to predict pixel values. When the first patch is synthesized, we use it to search for another seed-block to be used for synthesizing the second patch. This process is illustrated in Figure 3.



(a) (b)

Figure 3. The process of searching a seed-block. (a) Partly synthesized texture. The texture patch filled with random noise is the one to be synthesized. (b) The seed block is searched in the already synthesized texture. The seed block will be treated as already generated values in (a) and the rest pixels in the patch will be synthesized using SVM.

D. Seeded Patch Prediction and Image Quilting

As shown in Figure 3, after we synthesize part of the output texture, we need to find a seed-block for the adjacent patch (shown as random noise). We use pixels in the blue frame as the initial values to generate the candidate seed-block which is shown in the red frame through SVM. Then we compare the candidate seed-block with the patches of the same size in the already generated patches. Since we have successfully

synthesized the first patch, we simply use it for searching seed blocks. The best matched block in this patch is selected as the next seed-block. In the overlap region between neighbor output patches, we employ the image quilting method proposed in [1] to search the best cutting curve. In this way, adjacent patches can be smoothly connected. The process is repeated until all pixels in the output texture are assigned values.

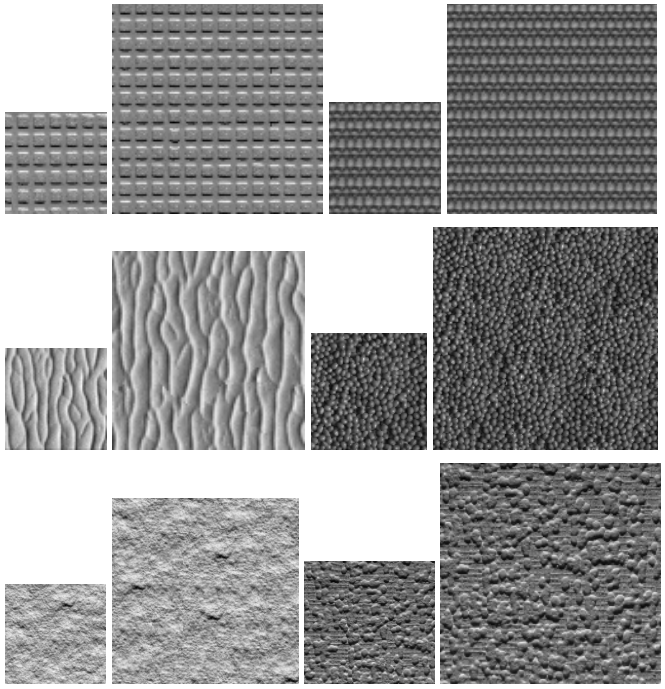


Figure 4. Texture synthesis results. The small image is the sample, and the larger one is the output texture synthesized by the proposed method based on multiple seed-blocks and SVM.

III. IMAGE EXTRAPOLATION

In [9], Efros and Leung introduced an image extrapolation method. Wei and Levoy further improved the algorithm and modified the synthesis order in a spiral way to avoid the directional bias [5]. However, these methods essentially perform exhaustive search within the existing image for best match for the neighborhood surrounding the pixel to be generated.

We propose to generate a training data set from four directions of the sample, and then it is used to train the SVM model. Next we synthesize the output texture from four directions. In this way, image extrapolation can be achieved by our extrapolating approach based on multiple seed-blocks and support vector machines.

IV. THREE-DIMENSIONAL SURFACE TEXTURE SYNTHESIS

Recent research into texture synthesis is normally concerned with learning and generation of 2D intensity texture images. However, 2D texture synthesis techniques cannot provide the information required for 3D surface texture such as the original illumination and viewpoint conditions. There are not many of publications regarding 3D surface texture synthesis [6, 7]. Similar to the work in [8], the proposed method can also model the representation image set in a parametric way, and the extension from 2D to 3D surface synthesis is same as in [8].

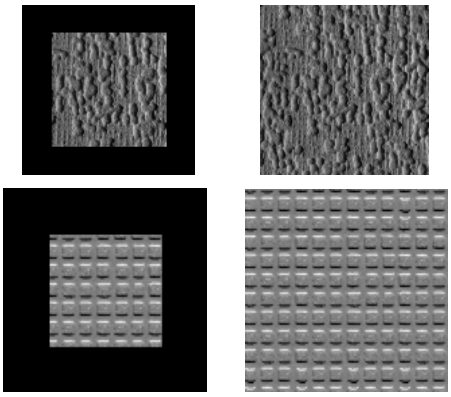


Figure 5. In each texture group, the left one is the original sample with black borders, and the right one is the result obtained from the proposed image extrapolation algorithm.

The first stage is to extract the representation images of the sample surface texture under a wide range of illumination directions. Three-dimensional surface texture requires a great number of images to capture its appearances under varied illumination directions or view points [11]. It is therefore important to extract a compact representation of the sample texture, as it is impractical to operate directly on the original data set [6]. In this paper we use 3I method to represent 3D surface texture with Lambertian reflectance and captured under many different lighting directions. The 3I method uses three images of the sample texture taken at an illumination slant of 45° and tilt angles of 0°, 90° and 180° [6]. These three images can further generate new images with arbitrary illumination directions.

Secondly, we train a model for each sample image in the representation data set and this process is identical to the training stage in the 2D case. The three images in the 3I representation method are modeled separately.

Thirdly, the three images are synthesized separately like a 2D texture. Finally the synthesized result representation images are relit through a linear combination and new texture images under arbitrary illumination directions may be generated [6].

V. EXPERIMENTAL RESULTS

In our experiments, the LibSVM package is used for SVM implementation [10] and the texture samples are from the PhoTex texture database [12].

Figure 4 shows the 2D texture synthesis results. It can be seen that the propose method can generate good results for both structured and random textures.

Figure 5 shows two groups of examples results generated by our image extrapolating approach based on SVM and quilting.

Figure 6 shows the results produced by the 3I representation. The small images are the samples and blow arrows are used to indicate illumination directions.

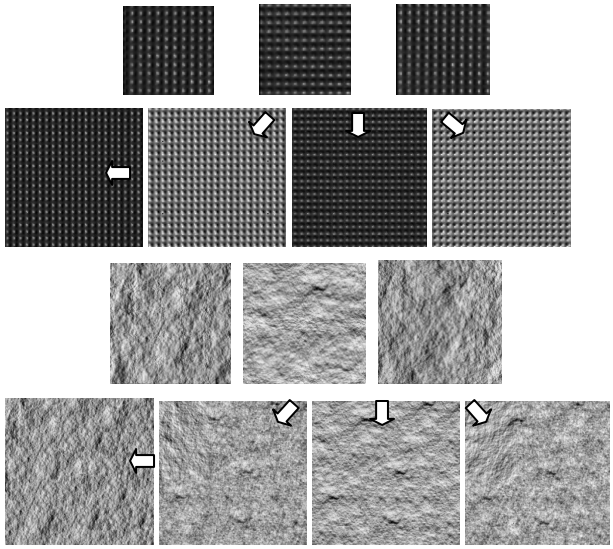


Figure 6. Results of 3D surface texture synthesis by 3I representation. For each group, the top row is the samples. The bottom row shows the synthesis results and the block arrows show illumination directions.

VI. CONCLUSION AND FUTURE WORK

This paper introduces a new texture synthesis method based on multiple seed-blocks and SVM. After feature extraction and model training, the input sample can be modeled by a linear model and the sample is not needed during the synthesis stage. Each time a patch with a small seed-block is synthesized using SVM. We employ the image quilting method in the overlap region of two adjacent patches so that they can be smoothly jointed. This method is further extended to synthesize 3D surface texture. Experimental results show that the method is applicable to both structured texture and random texture, and it also can be used for image extrapolating.

It should be noted that one drawback of this method is the high computation cost since each time only one pixel value is predicted by the SVM model. Future work may improve the computation efficiency.

ACKNOWLEDGMENT

The work was supported by a grant from National Nature Science Foundations of China (No. 60702014). We would like to thank Guimei Sun and Yuanxu Duan for sharing their previous work based on support vector machines.

REFERENCES

[1] Efros, A.A. and Freeman W.T.. Image quilting for texture synthesis and transfer. Proceeding of ACM SIGGRAPH, 341- 346, 2001.

[2] LIANG L., LIU C., XU Y.-Q., GUO, B., SHUM H.-Y. Real time texture synthesis by patch-based sampling. ACM Transactions on Graphics, 20(3): 127-150, 2001.

[3] Vapnik V N. The Nature of Statistical Learning Theory. NY: Springer-Verlag, 1995.

[4] Cristianini N, Shawe-Taylor J. Kernel Methods for Pattern Recognition. Cambridge: Cambridge University Press, 2004.

[5] L Y Wei, M Levoy. Fast texture synthesis using tree-structured vector quantization. Proceeding of ACM SIGGRAPH, 479-488, 2000.

[6] Junyu Dong and Mike Chantler. Capture and synthesis of 3D surface texture. International Journal of Computer Vision (IJCV), 62:177-194, 2005.

[7] Liu, X.; Yu, Y. and Shum, H.Y. Synthesizing Bidirectional Texture Functions for Real-World Surface. Proceedings of the 28th annual conference on Computer graphics and interactive techniques, 97-106, 2001.

[8] Dong, J.; Duan, Y.; Sun, G. Texture Synthesis through Support Vector Machines. Proceedings of 19th International Conference on Pattern Recognition, 2008.

[9] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. Proceedings of ICCV, 1033-1038, September 1999.

[10] Fan, P.-H. Chen, and C.-J. Lin. Working set selection using the second order information for training SVM. Journal of Machine Learning Research 6, 1889-1918, 2005.

[11] Dana, K. J.; Van Ginneken, B.; Nayar, S. K. and Koenderink, J. J. Reflectance and Texture of Real-World Surfaces. ACM Transactions on Graphics, 18(1):1-34, 1999.

[12] http://www.macs.hw.ac.uk/texturelab/resources/databases/Photex/index.htm

[13] Tuen-Young Ng, Conghua Wen, Tiow-Seng Tan,Xinyu Zhang, Young J. Kim. Generating an ω–Tile set for Texture Synthesis. Computer Graphics International 2005.

[14] F Cohen Michael, S Jonathan, H Stefan, D Oliver. Wang Tiles for image and texture generation. ACM transactions on graphics, 2003.