Patch-based Synthesis for Non-frontal-parallel Textures (NFPT)

Chung-Ren Yan and Tong-Yee Lee*
Dept. of CSIE
National Cheng-Kung University, Taiwan
tonylee@mail.ncku.edu.tw

Chao-Hung Lin
Dept. of Geomatics
National Cheng Kung University, Taiwan
linhung@mail.ncku.edu.tw

Abstract

In this paper, we propose a new patch-based texture synthesis method. The proposed method consists of two main components: a feature-weighted similarity and a pixel-based re-synthesis step. Most existing patch-based methods can only handle for frontal-parallel textures (FPT) and generate poorly synthesized results for non-frontal-parallel textures (NFPT). To better synthesize NFPT, we enhance the proposed method with a view warping technique. As a result, the proposed method can handle well for FPT and NFPT.

1. Introduction

Texture synthesis is a popular research topic in computer graphics. The kernel of the previous works is structural similarity matching. Generally, there are two classes of methods: pixel-based [1, 2, 6, 7, 8] and patch-based [3, 4, 5, 9, 10, 11] methods, respectively. Pixel-based algorithms synthesize only one pixel at a time. Efros et al.[2] synthesize a new pixel by searching the sample image and finding the pixel that has the most similar neighborhood. This algorithm works fine if a larger neighborhood size is chosen to measure similarity. Therefore, it is very slow. Wei et al. [7] utilize a pyramid synthesis and a tree-structured vector quantization method to accelerate [2]. Later, Ashikhmin [1] extends [7] by reducing the searching space of the input image to several candidates. Hertmann [6] presents a method to learn painting styles, based on [7]. Zelinka et al. [8] create a novel jump map, which stores a set of matching input pixels (jumps) for each input pixels. It allows their algorithm to synthesize texture in real time.

In contrast to pixel-based methods, patch-based methods synthesize a patch at a time. Xu et al. [3] synthesize new textures by random patch pasting. The problem with random patch pasting algorithm is that the discontinuity and artifacts at the overlapped region of the adjacent patches. Efros et al. [4] present a minimum-error-boundary-cut within the overlapped regions to reduce the discontinuity and artifacts. Liang et al. [5] apply feathering technique on the overlapped regions and accelerate the search by a quad-tree pyramid data structure. In contrast to most patch-based methods, Nealen et al. [10] adaptively splits patches to suitable sizes while satisfying user specified error for the mismatching in the overlapped regions. A pixel-based re-synthesis is then applied to the mismatched pixels in overlapped regions. To reduce the problem of broken features at the boundary of two adjacent patches, Wu et al. [9] propose a feature map to guide texture synthesis. For objects with regular or near-regular textures (NRT), Liu el al. [11] use a deformation technique to regularize an input NRT into a RT, then perform synthesis on this RT and finally utilize deformation information to convert this synthesized RT to the original near-regular structure.

In this paper, we propose a patch-based texture synthesis method. This method consists of two main components: 1) a feature-weighted similarity measure to search for the best match and 2) a pixel-based re-synthesis is used to reduce discontinuity at the boundary of adjacent patches. Liang et al [5] mention that most patch-based algorithms can not handle non-frontal-parallel sample textures that are usually obtained by camera lens with non-vertical camera axis. In this paper, we solve this problem using a view-warping technique [12]. Therefore, the proposed method still works well for synthesizing non-frontal-parallel input textures. The rest of the paper is organized as follows. In Section 2, we present the proposed patch-based method. In Section 3, we introduce our handling for non-frontal-parallel textures. The texture synthesized results for non-frontal-parallel textures are demonstrated in Section 4. Finally,

* Corresponding author.

Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC'06)
0-7695-2616-0/06 $20.00 © 2006 IEEE

(a) (b) (c) (d)

Figure 1. (a): The W_d. function varies smoothly from high importance at the patch boundary, i.e., black border, to low importance at the blue border of the L-shape neighborhood; (b) finding a minimum-error-cut path, (c) defining a repairing area, (d) mismatched pixels in green color.

conclusions and suggestions for future work are given in Section 5.

2. Texture Synthesis

The proposed patch-based texture synthesis method includes two major steps: 1) search in a sample texture for the best match for the current output neighborhood and 2) re-synthesize the mismatched pixels at the boundary of two adjacent patches. In the following, we outline our method and then describe these two major steps.

1. Select an initial patch from input texture randomly and paste it to the left top corner of the output texture.
2. Search an adjacent patch that has the best matched neighborhood in the input texture constrained by a L-shape.
3. Paste the best matched patch to the output image in a scan-line order patch by patch.
4. Find a minimum-error-cut path.
5. Repeat step 2 to 4 until we complete output image.
6. Re-synthesize the repairing regions.
7. Gaussian filtering on re-synthesized pixels (optional).

Our method generates output texture in a scan-line order from top to bottom. Initially, we randomly select a 2D texture patch p_i from 2D input texture and paste p_i into the desired output texture. Then, we search the best possible patch p_j in the input texture constrained by a L-shape neighborhood.

Usually, most previous techniques compute L2 distance in color space to measure the similarity of two neighborhoods. From our observation, we see that the pixels closer to patch boundaries have more influence on the resulting textures and therefore we would like to strengthen their importance by a distance-weighted function W_d in Eq.(1).

$$W_d(i,j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d(i,j)^2}{2\sigma^2}} \quad (1)$$

In above equation, *d(i,j)* is the distance form a pixel *(i,j)* to the closest patch boundary. We visualize the value of this distance-weighted function for each pixel in the L-shape neighborhood in Figure 1(a). In addition to Eq. (1), we would like to strengthen the importance of local texture structure perceived in the L-shape neighborhood using W_s in Eq. (2).

$$W_s(i,j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{[|M_{color} - P(i,j)| - 255]^2}{2\sigma^2}} \quad (2)$$

For a local L-shape, we can find its histogram of color distribution. In above equation, M_{color} represents the color with maximum number of pixels in the histogram and *P(i, j)* is the color of the pixel at *(i, j)*. In some sense, M_{color} can be considered as a background color in a local L-shape. The difference between *P(i, j)* and M_{color} is simply treated as the strength of features detected in our method. Finally, the proposed similarity measure is defined in Eq. (3).

$$Sim = \sum_{i,j \in L-shape} P_{diff}(i,j)(W_d + W_s) \quad (3)$$

The step4-7 is illustrated in Figure 1(b)-(d), and more details can refer to [14].

3. Handling Non-frontal-parallel Textures (NFPT)

In literature, Liang et al [5] may first mention that most patch-based algorithms can not handle non-frontal-parallel sample textures. Non-frontal-parallel textures (NFPT), i.e., obtained when a camera is parallel to a planar texture, are very common today. In Liang et al's paper, many failure examples of patch-based method are demonstrated. In this paper, we solve this problem using a well-known view warping technique [12] depicted in Figure 2. In Figure 2, there

Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC'06)
0-7695-2616-0/06/$20.00 © 2006 IEEE

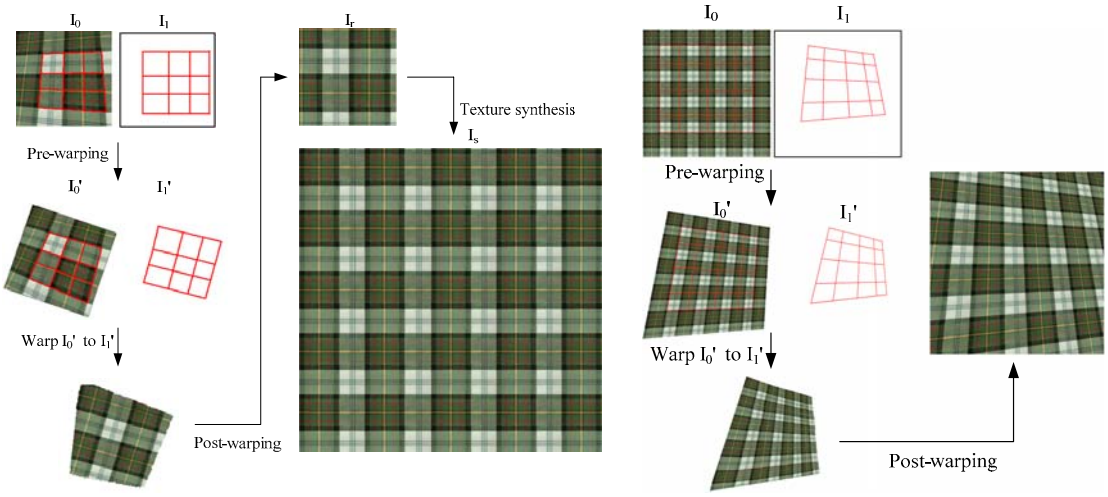


Figure 2. Warping system for non-frontal-parallel texture (NFPT) synthesis. (a): create a rectified FPT for a given NFPT and texture synthesis, (b): generate a different view of the synthesized texture.

are two major parts: 1) to generate a rectified FPT for a given NFPT and texture synthesis in Figure 2(a), and 2) to create a different view of texture synthesis in Figure 2(b), i.e., create another NFPT. The steps are described as follows. First, the input NFPT, I_0 will be transformed into a FPT, I_1. At this stage, the user must specify a set of corresponding feature lines on I_0 and a blank image I_1. We compute the intersections of these feature lines as feature points, i.e., 8 or more corresponding points are required on both I_0 and I_1. Then, the fundamental matrix of two views can be determined for pre-warping I_0 and I_1 into two parallel views, I_0' and I_1' [12]. This pre-warping also changes positions of corresponding points. Remember I_1' is a still empty image at this moment. But, we have corresponding points on I_0' and I_1'. Next, with these corresponding points and I_0', we use [13] to generate an I_0'. Thereafter, we proceed a post-warp, i.e., inverse of pre-warp, to warp I_1' to the original view of I_1. This warped image called I_r can be treated as a rectified image of I_0. Consequently, we use I_r as an input sample for texture synthesis described in Section 2 and obtain a new output texture I_s. Finally, to generate different views of I_s, we apply the similar scenario to warp I_s to the desired view in Figure 2(b). The texture synthesis for Non-frontal-parallel Textures (NFPT) is finished.

4. Experiment Results

In this section, we show our synthesized results for handling NFPT in Figure 3-6. These results demonstrate that our method does very well for NFPT. In addition, we also show a failure example of handling NFPT using our patch-based method without view-warping in Figure 3(b) an 4(b). This shows the need of view-warping for handling NFPT. Most existing methods will generate similar failure results without considering perspective effect on sample textures.

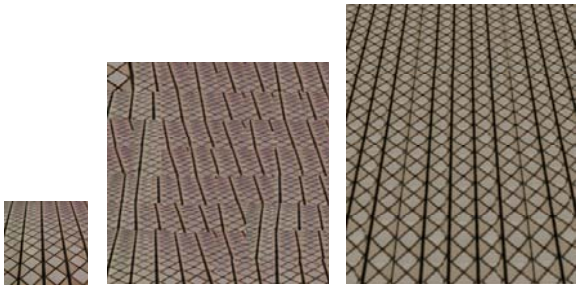
5. Conclusion and Future Work

In this paper, we introduce a patch-based method with a view-warping to handle both frontal-parallel and non-frontal-parallel textures. A comparison between our method and other existing algorithms is demonstrated to verify the proposed method. In near future, we plan to research acceleration method to speed up our texture synthesis. Other extension for synthesizing texture on 3D surface or better/fast method for handling NFPT will be investigated soon. In addition, we would like to apply our technique to synthesize texture for the expression change of head animation [15] or for 3D morphing applications [16].

6. Acknowledgements

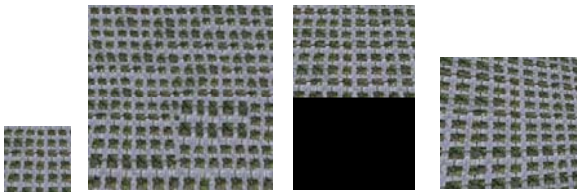
This project is supported by National Science Council, Taiwan, R.O.C under contract No. 94-2213-E-006-063 and 94-2213-E-006-005.

7. References



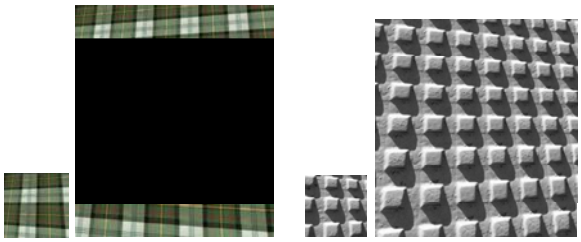
(a) (b) (c)

Figure 3. (a)original texture, (b)synthesis result without view-warping, (c)synthesis result with view-warping.



(a) (b) (c) (d)

Figure 4. (a)original texture, (b)synthesis result without view-warping, (c)(d)synthesis result with view-warping.



(a) (b) (c) (d)

Figure 5. NFPT synthesis results

(a) (b) (c) (d)

Figure 6. NFPT synthesis results

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

[2] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling", *Intl. Conf. Computer Vision*, 1999, pp. 1033-1038.

[3] Y. Xu, B. Guo, And H.-Y Shum, "Chaos mosaic: Fast and memory efficient texture synthesis", *Microsoft Res. Tech. Rep. MSR-TR-2000-32*, April 2000.

[4] A. Efros, and T. Freeman, "Image Quilting for Texture Synthesis and Transfer", *Proceedings of SIGGRAPH*, 2001, pp. 27-34.

[5] L. Liang, C. Liu, Y. Xu, B. Guo, and H.-Y Shum, "Real-time texture synthesis using patch-based sampling", *ACM Trans. Graphics*, Vol. 20, No. 3, 2001, pp. 127-150.

[6] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, and D. Salesin, "Image analogies", *Proceedings of SIGGRAPH*, 2001, pp. 327-340.

[7] L.-Y. Wei And M. Levoy, "Fast texture synthesis using tree-structured vector quantization", *Proceedings of SIGGRAPH*, 2000, pp. 479-488.

[8] S.Zelinka and M. Garland, "Towards real-time texture synthesis with the jump map", *Proceedings of the Thirteenth Eurographics Workshop on Rendering Techniques*, 2002, pp. 99-104.

[9] Q. Wu and Y. Yu, "Feature Matching and Deformation for Texture Synthesis", *ACM Transactions on Graphics*, Vol. 23, No. 3, 2004, pp. 362-365.

[10] A. Nealen and M. Alexa, "Hybrid Texture Synthesis", *Proceedings of Eurographics Symposium on Rendering*, 2003, pp. 97-105.

[11] Y. Liu, W.-C. Lin, and J. HaysG, "Near Regular Texture Analysis and Manipulation", *ACM Transactions on Graphics*, Vol. 23, No. 3, 2004, pp. 368-376.

[12] S. Seitz, and C. Dyer, "View Morphing", *Proc. SIGGRAPH* 96.

[13] T. Beier, and S. Neely, "Feature-based image metamorphosis", *Proc. SIGGRAPH 92. In Computer Graphics*, 1992.

[14] T.Y. Lee and C.R. Yan, "Feature-based Texture Scheme", *Lecture Notes on Computer Science (LNCS 3482)*, volume 3482, Springer-Verlag, 2005, pp. 1043-1049.

[15] T.Y. Lee, P.H. Lin, and T.H. Yang, "Photo-realistic 3D Head Modeling Using Multi-view Images", *Lecture Notes on Computer Science (LNCS 3044)*, Springer-Verlag, May 2004, pp. 713-720.

[16] C.H. Lin, and T.Y. Lee, "Metamorphosis of 3D Polyhedral Models Using Progressive Connectivity Transformations", *IEEE Transactions on Visualization and Computer Graphics*, Vol. 11, No.1, Jan-Feb. 2005, pp. 2-12.