A Rapid Texture Synthesis Algorithm Based on Clustering Preprocessing*

Kong De-Hui†, *Yin Bao-Cai*††, *Li Yan*†††, *Shi Yun-Hui*††††

Key Laboratory of Multimedia and Intelligent Software, College of Computer Science in Beijing University of Technology, Beijing 100022, China

Summary
This paper presents a novel technique using clustering preprocessing for the challenging problem in the field of computer graphics, which is to speed up texture synthesis without losing of synthesis quality and adding of method complexity. An algorithm of rapid texture synthesis based on clustering preprocessing is designed to speed up both of pixels-based and patch-based local region-growing methods maintain original level of texture quality or gaining better. Experiments show that efficiency of pixel-based texture synthesis method with respect to stochastic texture sample increases by 80 percent with better synthesis quality, and efficiency of patch-based texture synthesis method with respect to structured texture sample increases by 70 percent. Due to avoiding the step of blending the overlap pixels needed by patch-based texture synthesis the synthesis quality is more steady than original results.

Key words:
Texture synthesis; Local region-growing methods; clustering; preprocessing

1. Introduction

As one of the key technique of computer graphics, sample-based texture synthesis has become one of the hottest issue in the field with the greatly spreading application of computer graphics[1][2][3][4][5][6][7][8][9][10][11][12] [13]. This technique has been developing basing on the Markov random field property of the given texture samples, which means that the samples satisfy the locality and stationarity. Locality implies that the color at a pixel is dependent only on the neighbor pixels around it, while stationarity implies that this dependency is independent of the actual location of the pixel. In generally, the main texture synthesis algorithms can be broadly classified into two categories: global optimization-based methods [14] and local region-growing methods [10].

Global methods evolve the entire texture as a whole, based on some criteria for evaluating similarity with the input sample. Energy of synthesized texture with respect to the texture of input samples can be measured by comparing local neighborhoods in the two textures and used as a kind of the criteria, which equals to the sum of energies of all of individual synthesized neighborhood that is defined as its distance to the closest neighborhood in the texture of input sample. Powerful controllable feature of the synthesis process is the advantage of this category of methods, however the disadvantages of the methods are brought on by the complexity of energy function and its dependency to the shape of neighborhood makes the synthesis result unsteady. Furthermore, iterative optimization process needed by the methods is a time-consuming process.

Local region-growing methods, called local matching methods also, grow the texture one pixel or one patch at a time with the goal of maintaining coherence of grown region with nearby pixels. Thus, the key steps of the methods are to find the pixel or the patch in the input sample that matches with the neighborhood texture in the synthesizing texture. In this paper a strategy based on clustering preprocessing is introduced to improve these methods on both aspects of quality and speed. The paper is constructed as following: Part Ⅱ reviewed some work related with local region-growing methods and abstract the contribution in this paper. Part Ⅲ introduces the clustering preprocessing in details. Part Ⅳ describes the new algorithm supported by clustering preprocessing. Part Ⅴ shows the experiments and the conclusions is made in the next part.
For detailed information for authors, please refer to [1].

2. Related work

Local region-growing methods are further classified into two categories by the amount of the grown pixels: pixel-based and patch-based matching methods. Efros etc.[10] in 1999 proposed one of the earliest pixel-based matching method that used a region of rectangle shape as the texture neighborhood. It was improved in 2000 by Wei and levoy[12] by replacing rectangle shape with a ruler of L shape and introducing multi-resolution and vector quantifying techniques to speed up matching process. These improvements made the method become one of the most typical pixel-based synthesis method. After that, many achievements including Ashikhmin[11][2001] are reported. In general, pixel-based methods are simple and easy to be accomplished and suitable to the input sample of stochastic texture. However the characteristic of synthesizing only one pixel at a time makes them be convinced as inefficient methods; moreover the methods are not suitable to the input samples with complex or structural details. The methods synthesizing one patch of pixels at a time were presented in 2001 by Efors[6] and

* Supported partly by the National Natural Science Foundation of China under Grant No.60572104, No.60375007, and No.60533030

Liang etc.[7] respectively, that improved the efficiency of pixel-based methods and maintained more local property of input by using more neighborhood pixels when matching. After that, Cohen1[15][2003], Nealen etc.[9][2003] and Nealen etc.16][2004] developed the idea and proposed a series of patch-based methods that improved the quality of the synthesizing texture and increased the speed of the matching process. After all, blending of overlap part of two neighbor patches is required by the patch-based methods and as a result the quality of the synthesizing texture is decreased.

Besides of extending growing region from a pixel to a patch of pixels, many others efforts have been done to improve the efficiency of synthesis methods, including : Ashikhmin[11] introduced the strategy of limiting the area on the input sample where the matching neighborhood of pixels was searched to speed up matching process, Wei and levoy[12] made use of multi-resolution and vector coding techniques to speed up synthesis, XiaoGang Xu[13] improved Efors [10] by searching matched patch along a spirality order instead of scan-line order and introduced multi-seed strategy to speed up matching, Liang etc.[7]combined PCA(Principal Components Analysis), Quad-Tree Pyramid, and Optimized KD-Tree data structure to speed up synthesis, Zelinka etc. [17] [2004] bypassed matching process by using a preprocessing skill, Lefebvre etc. [18] [2005]] improved pyramid model to accomplish a parallel synthesis algorithm and made use of GPU programming technique to further speed up synthesis. In general, limiting matching area, optimizing matching rulers, simplifying the computing of matching error and adopting parallel technique are the current clues to speed up texture synthesis. While all of the accelerating skills achieve their aims by losing either the accuracy of matching or the simplicity of the synthesis algorithm, worse depending on the hardware performance.

This paper presents a novel strategy that speeds up local region-growing methods without losing of matching accuracy and synthesizing simplicity, for which a new idea is introduced that the problem of synthesis is reconsidered as a problem related with recognizing rather than a problem related with matching. In details, the set constructed by all of the individual neighborhood of pixels enumerated from the input sample are divided into some sub-classes according to the similarities between neighborhoods, and this step is called clustering preprocessing. After that, to match synthesizing neighborhood of pixels on input sample, the key step of local region-growing method, can be reconsidered as to recognize a matched element from one of the sub-classes. Supposing that the whole set of neighborhood of pixels is divided into three sub sets evenly, quantity of matching operator needed by original algorithms would be reduced by two-third, that means one third quantity of recognizing operator would be needed by improved algorithms. In fact, this analysis result is powerfully proved by our experiments.

Compared with the current speedup skills the new strategy contributes as following: (ⅰ) The clustering preprocessing can be used to accomplish an accurate rapid algorithm that increase algorithm efficiency by more than 70 percent without losing quality of synthesis. (ⅱ) The clustering preprocessing unifies the synthesis process of both pixel- and patch-based synthesis methods. (ⅲ) Avoiding the blending of overlap pixels needed by patch-based methods and resulting in more steady quality of synthesis.

3. Clustering preprocessing of neighborhood of pixels

For describing the process more accurately, following terms used are defined in advance.

Matched Neighborhood of Pixels (MNPs): a pattern of a group of pixels defined according to the feature of the input sample, that can be used to recognize the group of pixels in the input sample that is similar to the group of pixels in the synthesizing texture image. It was represented by the grey pixels in Fig.1.

Synthesized Pixel /Patch of Pixels (SP/SPPs): a pixel /or a patch of pixels in the input sample that is a neighboring pixel (or a patch of pixels) of Matched Neighborhood of Pixels defined above, shown as the red pixels in Fig.1. When a MNPs is found in the input sample, the SP/SPPs will be rendered on the corresponding location in the synthesizing texture image.

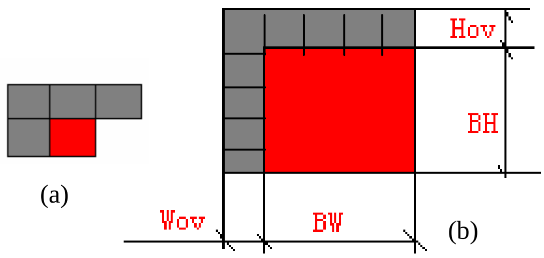


Fig.1 MNPs (shown as grey pixels)and SP/SPPs(shown as red pixels) (a) for stochastic input sample; (b) for non-stochastic input sample

According to the different features of stochastic and non-stochastic input samples, two kinds of MNPs and the corresponding SP/SPPs are defined respectively. For the former a SP is defined and a SPPs of size of BH×BW pixels is defined for the latter. Both of the MNPs satisfy L shape but with different dimensions, shown as Fig.1.

Initial Space of MNPs (ISMNPs) : according to the define of MNPs, individual MNPs can be enumerated in the input sample along the scan-line order and all of them constructs a set. When every MNPs is represented as a vector of color, the set can be considered as a space of vectors, called Initial Space of MNPs.

Clustered Space of MNPs (CSMNPs) : after clustering with respect to MNPs is executed in ISMNPs according to its similarity, some sub-spaces come into being. Supposing the amount of the sub-spaces is R, the group of the sup-spaces is called R-Clustered Space of MNPs.

Supported by the above terms, a clustering preprocessing of neighborhood of pixels can be described as following three steps.

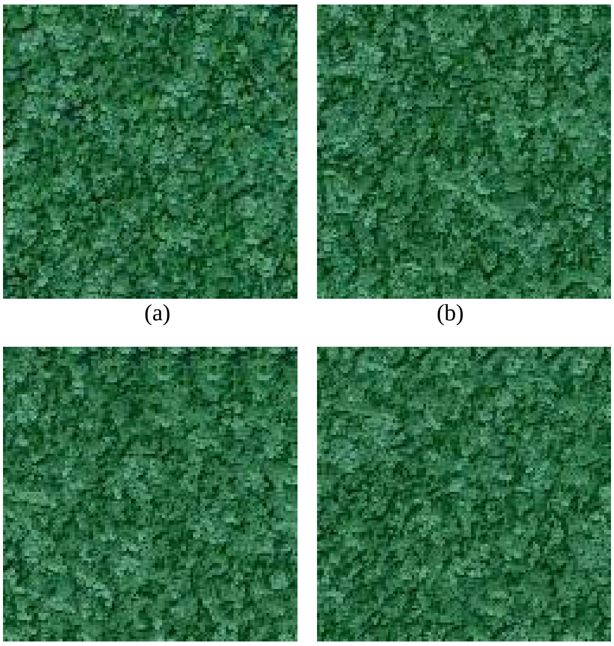
i) To define the MNPs and corresponding SP/SPPs according to the feature of input sample.

ii) To construct the ISMNPs.

iii) To cluster MNPs in ISMNPs and construct CSMNPs. In this paper RGB based Euclidean distance is used to measure the similarity between MNPss and K-means clustering strategy is chosen to compute the R-CSMNPs and the kernel vectors of R sub-spaces.

The function of clustering preprocessing is effectively reducing the domain where MNPs is searched and absolutely maintaining the matching accuracy in the meantime. Thus a rapid synthesis algorithm can be expected with losing neither matching accuracy nor synthesis simplicity. Fig.2 shows the result images of texture synthesized respectively using the algorithms of Wei 和 levoy[12], shown in Fig.2(a), and our algorithms using 3/5/8-CSMNPs respectively, shown in Fig.2(b/c/d), where the input is sample (a) shown in Fig.4.

Combined other information summed up in Tab.1 a conclusion appears that the algorithms using clustering preprocessing reduce more than half synthesizing time under the same synthesizing quality with respect to the original algorithm. Furthermore, the more sub-spaces are clustered, the more synthesizing time (22 percent) is reduced and the better synthesizing quality is expected.



(a) (b)

(c) (d)

Fig.2(a) texture image synthesized by Wei and Levoy[12] algorithm Fig.2(b) texture synthesized by our algorithm using 3-CSMNPs Fig.2(c) texture synthesized by our algorithm using 5-CSMNPs Fig.2(d) texture synthesized by our algorithm using 8-CSMNPs

Table.1: specifications of algorithms applied to synthesize the textures in Fig.2
(the sizes of input and images are measured by pixel)

	Wei&levoy[12] Algorithm	Our algorithms using R-CSMNPs		
		R＝3	R＝5	R＝8
Size of input sample	64×64	64×64	64×64	64×64
Size of MNPs	5×5	5×5	5×5	5×5
SP/SPPs	One pixel	One pixel	One pixel	One pixel
Size of synthesizing image	124×124	124×124	124×124	124×124
Synthesizing quality	norm	norm	better	best
Synthesizing time(s)	42.215	19.219	12.47	9.282

More valuable properties of the clustering preprocessing can be explored by recursively using it in sub-spaces, consequently multi-level CSMNPs are constructed and the efficiency of synthesis algorithm can be greatly increased. Besides that, simplifying the computing of similarities between MNPs by representing the MNPs as a sparse vector that equals the difference between it and the kernel of the sub-space is also a path to speed up the algorithm.

When clustering preprocessing is finished, relationship between MNPs and SP/SPPs should be recorded to be used in following synthesis method.

4. A rapid texture synthesis algorithm based on clustering preprocessing

Using above clustering preprocessing, existing local region-growing methods, including both pixel- and patch-based algorithm, can be improved. In this part a rapid texture synthesis algorithm using R-CSMNPs is described.

4.1 The algorithm using clustering preprocessing

The synthesizing process is shown in Fig.3: The grey part in Fig.3(a) shows a MNPs that is found in the input sample that matches with the current MNPs in synthesizing map shown as the same grey part in Fig.3(b). The corresponding SP/SPPs shown as red part in Fig.3(a) is then rendered in the synthesizing map, which is shown in Fig.3(c). The current MNPs is updated progressively to repeat the steps until the synthesizing map is finished. The blue part in Fig.3 represents the synthesized part of pixels in the synthesizing map.

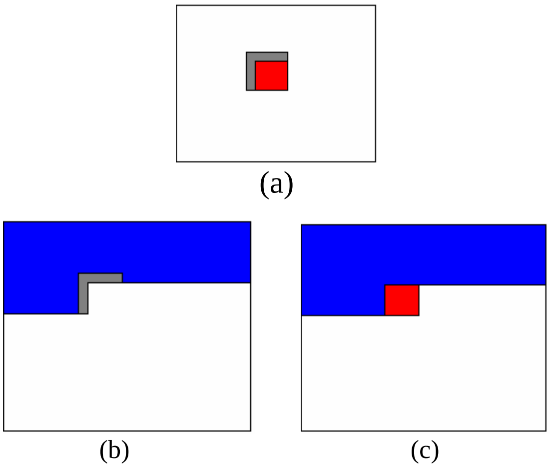


Fig.3 synthesizing process: (a)a MNPs found in the input (b)MNPs in the synthesizing map (c) SP/SPPs renderedin the synthesizing map

The steps of the algorithms including:

Step1: The clustering preprocessing. After that the R-CSMNPs and their kernels, $\boldsymbol{C}_i$(i = 1,2,…,R), are computed. In the meantime, relationship between the MNPs and its corresponding SP/SPPs is recorded as ***L*** ;

Step2: Initializing the synthesizing map as ***I*** ;

Step3: Updating the current MNPs in synthesizing map as ***P*** along the scan-line order;

Step4: Computing the distances between ***P*** and $\boldsymbol{C}_i$(i = 1,2,…,R) and then, the sub-space whose kernel is the nearest vector from the current MNPs is recorded as the current CSMNPs, sign as C***S***. If the multi-level CSMNPs was clustered in Step1, then this step should be repeated until finding out the lowest level of sub-space, still sign as the CS;

Step5: Searching out the MNPs in C***S*** matched with ***P***. And then using ***L*** to find the corresponding SP/SPPs that should be rendered on the corresponding location of ***I***;

Step6: Repeating the Step3~Step6 until the synthesizing map is finished.

Where Step1 can be executed off-line and treated as the same input parameters with the input sample.

4.2 Discussion on the application of the algorithm

In general, existing pixel-based matching methods expend large amount of time to achieve synthesizing map with excellent effect; while patch-based methods expend less time but reach more bad effect than pixel-based methods. Further more, due to using larger size MNPs, patch-based methods encounter barrier to speed up greatly. The steps of our algorithm are the same for both pixel- and patch-based methods, so the application of clustering preprocessing unifies two categories methods and speeds up them greatly.

For pixel-based methods, which is used to stochastic input samples, MNPs and SP are defined as the patterns shown in Fig.1(a) and the patterns ensure the granularity uniformity of CSMNPs. Thus the searching of MNPs in the input sample can be executed accurately in one of R-CSMNPs whose dimension is much less than the original searching domain, this makes sure the coinstantaneous improvement on both effect and speed of methods.

For patch-based methods, MNPs and SPPs are defined as the patterns shown in Fig.1(b) and the patterns demonstrate the local feature of the samples clearly, which is used to non-stochastic input samples. Indeed the granularity of level-one CSMNPs is not uniform, then multi-level MNCPs can be applied to solve the problem. The similarity between the MNPss ensures that the corresponding SPPs produces synthesizing map of better effect than patch matching strategy. On the other hand, the dimension of the MNPs is less than that of the patch used in patch matching methods, so that the searching of MNPs can be executed rapidly, especially in the multi-level CSMNPs.

In one word, our improved algorithm using clustering preprocessing unifies the pixel-based and patch-based methods without losing the simplicity of methods, and the efficiency of the methods is improved on both speed and effect of synthesis.

5. Results of experiments

Four input samples (64×64 pixels) used in our experiments are shown in Fig.4 and the platform is PC(P4 2.8GHz/512M). In Fig.2 the synthesized maps synthesized by pixel-based methods and our improved algorithm are shown and contrasted.

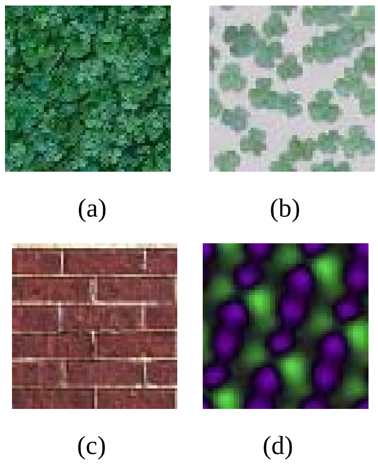
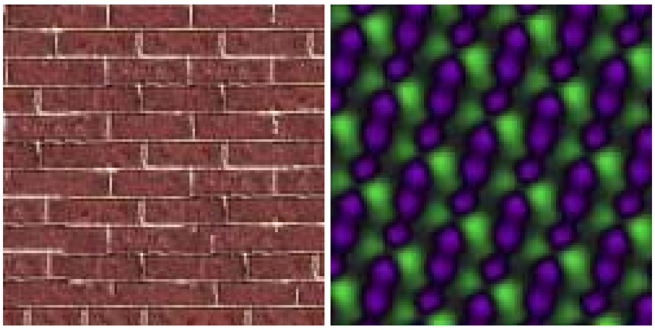


Fig.4 The input samples

In this part one of patch-based methods, Efros [3] patch matching method, is chosen to synthesize maps using the five inputs and the results are shown in Fig.5. The size of the result map is 124× 124pixels.

In the meantime, its improved form, our algorithm using the clustering preprocessing, is contrastively accomplished and the results are shown in Fig.6. The size of the result map is 124× 124pixels.

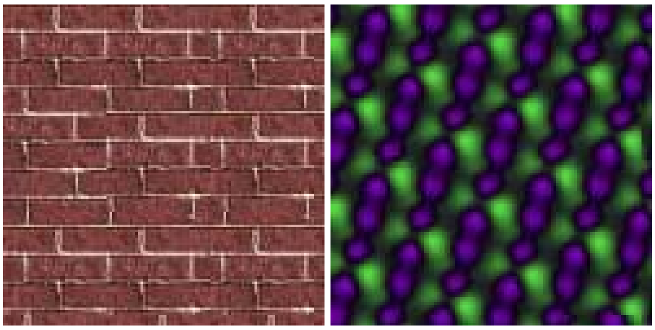
(a) (b)



(c) (d)

Fig.5 Synthesized maps using Efors patch matching method

(a) (b)



(c) (d)

Fig.6 Synthesized maps using our algorithm

The efficiencies of two algorithms are contrast in Tab.2. A conclusion can be reached that under the same effect of synthesizing maps, our algorithm increases the speed of synthesis by 70 percent than original method.

Table.2: Related information of Efors patch matching method and our algorithm (the dimension of patch and MNPs/SPPs is pixel)

Input sample	Efors patch matching method			Our algorithm			
	Dimension of patch	Overla p pixel	**Time for synthesizing(s)**	MNPs	SPPs	Time for Clustering(s)	**Time for synthesizing(s)**
(a)	28×28	4	**0.5**	4×4	30×30	14.09	**0.14**
(b)	28×28	4	**0.39**	4×4	30×30	9.78	**0.14**
(c)	28×28	4	**0.406**	4×4	30×30	12.35	**0.172**
(d)	28×28	4	**0.718**	10×10	40×40	5.97	**0.110**
(e)	28×28	4	**0.422**	4×4	30×30	10.9	**0.125**

6. Conclusion

Increasing the speed of texture synthesis as high as possible maintaining the effect of synthesis images is a challenge issue in the field of realistic computer graphics in the recent years. The strategy of clustering preprocessing and the algorithms using it presented in this paper improve the existing local region-growing methods, including pixel- and patch-based methods by both aspects of speed and effect of texture synthesis. Experiments prove that our algorithm can be used to handle both stochastic and non- stochastic input samples with the efficiency increase by 70 percent or more. Furthermore, some skills, such as multi-level clustering and sparse vector etc., can be used to extend the achievement.

Experiments also show that the efficiency of texture synthesis is related with the definition of MNPs, so exploring the clustering preprocessing to define the MNPs adaptively and choose more suitable clustering means are future topics.

References

[1] http://www.bjpu.edu.cn/sci/multimedia/mul-lab/Chinese/kong.htm

[1] HEEGER, D. J., AND BERGEN, J. R. Pyramid-based texture analysis/synthesis. In Proceedings of SIGGRAPH 95 (Anaheim, California, August 6–11, 1995) (August 1995), R. Cook, Ed., Computer Graphics Proceedings, Annual Conference Series, 229–238.

[2] DEBONET, J. S. Multiresolution sampling procedure for analysis and synthesis of texture images. Proceedings of SIGGRAP 97 (August 1997), 361–368.

[3] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman. Texture Mixing and Texture Movie Synthesis Using Statistical Learning. IEEE Trans. on Visualizationand Computer Graphics, 2001,7(2):120-135

[4] PORTILLA, J., AND SIMONCELLI, E. P. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision* , 2000 , 40 (10) : 49-71.

[5] Wei Li-yi. Texture synthesis by fixed neighborhood searching [D]1 Palo Alto : Stanford University , 2001.

[6] Efros A .A , Freeman W .T. Image quilting for texture synthesis and transfer [A]1 In : Computer Graphics Proceedings , Annual Conference Series , ACM SIGGRAPH , Los Angeles , California , 2001, 341-346.

[7] Liang .L , Liu .C , Xu .Y, *et al*1 Real-time texture synthesis by patch-based sampling [R]Beijing : Microsoft Research Asia , 2001.

[8] XU, Y., GUO, B., AND SHUM, H.-Y. Chaos mosaic: Fast and memory efficient texture synthesis. Tech. Rep. MSR-TR- 2000-32, Microsoft Research, 2000.

[9] A. Nealen and M. Alexa. Hybrid texture synthesis. In Rendering Techniques2003, 14th Euro graphics Workshop on Rendering, EurographicsAssociation, 2003, 97–105.

[10] EFROS, A., AND LEUNG, T. Texture synthesis by nonparametric sampling. International Conference on Computer Vision 2 (September 1999), 1033–1038.

[11] M. Ashikhmin. Synthesizing natural textures. In Symposium on Interactive 3D Graphics, 2001, 217–226.

[12] WEI, L-Y., AND LEVOY, M. Fast texture synthesis using tree-structured vector quantization. Proceedings of SIGGRAPH 2000 (July 2000), 479–488.

[13] XU Xiaogang, YU Jinhui, *et al.* Fast texture synthesis using multi-seeds. Journal of Image and Graphics, 2002, 7(10), 994-999. (In Chinese)

[14] Vivek Kwatra, Irfan Essa, Aaron F. Bobick, Nipun Kwatra Texture Optimization for Example-based Synthesis. Proceedings of SIGGRAPH 2005.

[15] M. F. Cohen,J. Shade, S. Hiller, and O. Deussen. Wang tiles for image and texture generation. In Proceedings of ACM SIGGRAPH 2003 (ACM Transactions on Graphics, 2003.) , 287–294.

[16] A. Nealen and M. Alexa. Fast and High Quality Overlap Repair for Patch-Based Texture Synthesis. In Rendering Techniques2004, 14th Euro graphics Workshop on Rendering,, Eurographics Association, 2004, 582-585.

[17] Steve Zelinka and Michael Garland. Jump Map-based interactive Texture Synthesis. ACM Transaction on Graphics, 2004, 23(4): 930-962.

[18] Sylvain Lefebvre, Hugues Hoppe, Parallel Controllable Texture Synthesis, Proceedings of SIGGRAPH 2005.

Kong Dehui received the B.S. degree in mathematics from Inner Mongolia University in 1989, and M.S. and Ph.D degrees in Computer from Beijing University of Aeronautics and Astronautics in 1992 and 1996, respectively. From 1996, she has been a staff of College of Computer Science in Beijing University of Technology and devoted to the research on the fields of Computer Graphics and its related domains.

Baocai Yin received his B.A.Sc., M.A.Sc, and Ph.D. degree from the Dalian University of Technology, Dalian, China, in 1985, 1988, and 1993 respectively. He was with the department of Computer Science, Harbin Institute of Technology, between 1993 and 1995, and he was Senior Software Engineer with Motorola-NCIC Joint R & D Laboratory for Advanced Computer and Communication Technologies, between 1996 and 1998. Since 1996, he has been with College of Computer Science and Technology, Beijing University of Technology, Beijing China. His current research interests are Facial image recognition and synthesis, Model Based Video Coding and Computer Graphics.