Pattern-based Texture Metamorphosis

Ziqiang Liu*† Ce Liu* Heung-Yeung Shum* Yizhou Yu‡

*Microsoft Research Asia †Zhejiang University ‡University of Illinois at Urbana-Champaign
zqliu@msrchina.research.microsoft.com {i-celiu, hshum}@microsoft.com yyz@cs.uiuc.edu

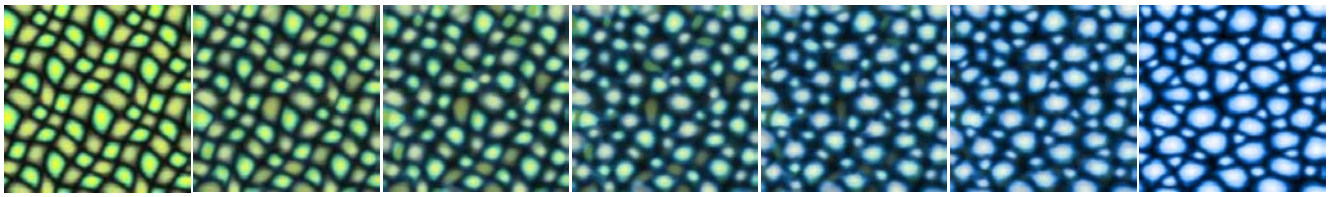


Figure 1. A texture metamorphosis sequence is generated from the leftmost to the rightmost.

Abstract

In this paper, we study texture metamorphosis, or how to generate texture samples that smoothly transform from a source texture image to a target. We propose a pattern-based approach to specify the feature correspondence between two textures, based on the observation that many texture images have stochastically distributed patterns which are similar to each other. First, the user selects a pattern in the source and target textures, and establishes the "local feature correspondence" between these two patterns by specifying landmarks. Then, repeated patterns are automatically detected and localized in the source and target textures. The "pattern correspondence" between two textures is formulated as an integer programming problem and solved using the Hungarian algorithm. Finally, we obtain a warp function between two textures by combining "local feature correspondence" and "pattern correspondence". Experiments demonstrate that our technique produces visually appealing morphing sequences, with moderate amount of user interaction.

1. Introduction

Image metamorphosis (a.k.a. image morphing) has become a powerful tool for generating special visual effects in films and television. Image metamorphosis refers to the process of morphing one image to another smoothly. In general, image morphing consists of two steps, specifying correspondence (e.g., points or lines) between two images, and computing a warp function that defines where each pixel in one image should move to in the other image. For example, to morph between two human faces, one needs to specify the correspondence between the noses, lips etc.

Inspired by the fascinating visual effect of general image morphing and the recent extensive work on texture synthesis, analysis and application, we are especially interested in texture morphing. However, texture metamorphosis is different from image metamorphosis because a texture image either is composed of many discernable and similar patterns or is highly random with irregular features. Thus, it is very difficult to manually specify correspondences between two textures. Unlike face morphing that needs feature correspondence on a small number of edges and corners, texture morphing would require feature correspondence on a very large number of feature points in the textures. Moreover, users may be confused about how to extract features and establish correspondences between them.

In this paper, we propose a pattern-based approach to address this problem for textures containing discernable and similar patterns. A pattern is a semantic unit that composes a group of feature points. The user only needs to select one representative pattern for each texture image and specify feature point correspondences for these patterns; our algorithm handles the remaining work of finding all similar patterns, establishing correspondences and generating morphing sequences. In brief, we decouple the correspondence problem between two texture images into "local feature correspondence" between two matched patterns, and "pattern correspondence" to map the patterns in one texture to the other. Deformable object detection and alignment techniques are used to find the patterns that are similar to what the user has specified. Then pattern correspondence is automatically established using the Hungarian algorithm to minimize the cost of the morphing path. Furthermore, we use the standard multilevel free-form deformation (MFFD)

algorithm [17] to compute the warp function which is then used to generate the morphing sequences.

This paper is organized as follows. In Section 2 we introduce some related work. Then, we present our approach to texture morphing in Section 3. Details of the algorithm are described in Section 4. In Section 5, extensive experiments are provided. We conclude this paper in Section 6.

2. Related Work

Texture morphing is related to both texture synthesis/editing and image morphing.

Texture synthesis. The pioneering work of Julesz suggested that two textures appear the same if they share some common statistics [13]. Heeger and Bergen [11] proposed to analyze and synthesize textures in terms of histograms of multi-channel filtering. Zhu et. al. [27] devised a sophisticated framework, FRAME, to learn the Gibbs potential function of a texture and synthesize samples by Gibbs sampling. They produced good results for highly stochastic textures, but were less successful on structured textures. Recent texture sysnthesis approaches focus on enforcing statistics locally. De Bonet [4] used a multi-resolution feature-based approach to sample each pixel conditioned on its "parents structures" at the coarser scales. Efros and Leung [8] developed a method of "growing" a texture pixel by pixel using nonparametric sampling (NPS) based on a spatial neighborhood. Wei and Levoy [22] proposed new methods to speed up the NPS algorithm. There are also works on synthesis from multiple source textures such as [2] and [21], where the texture created has the combined visual appearance of all the inputs.

Patterns and textons. Although the concept of "textons" was proposed by Julesz some twenty years ago to represent the basic elements in texture [14], it is not very clear how textons are constructed. Recently, Guo et. al. proposed a generative model to learn the textons including the basic form and the distribution of the texton map [10]. Their results support that a small number of patterns are sufficient to generate a texture image. Xu et. al. also observed that texture synthesis can be done by simply re-distributing texture patches that are visually meaningful [26]. Good synthesis results by Efros and Freeman [7] and Liang et. al. [18] further confirm the existence of repeated patterns in texture images.

Texture editing. A recent emerging interest is the development of tools and algorithms that enable users to edit textures. Hertzman et. al. [12] learnt the statistics of a pair of images in a nonparametric way, and output a novel image for input analogous to the learnt pair. Efros et. al. [7] demonstrated the effect of texture transfer, i.e. rendering an object with a texture taken from a different object. Brooks et. al. [5] proposed a system to edit textures by transferring the user's operation at a local area to the entire texture image.

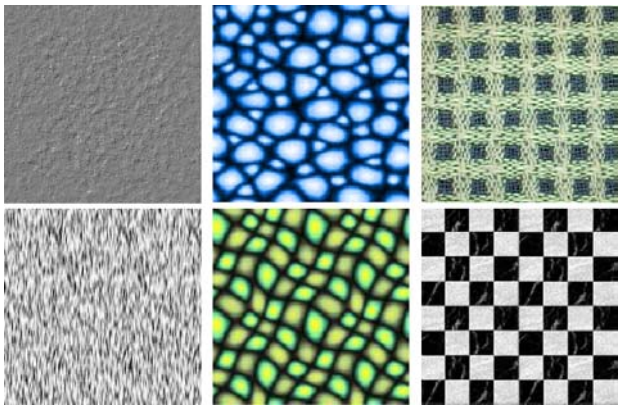


Figure 2. Three types of textures: (a) highly random (b) semi-structured (c) regular repeated. Above and below are pairs for texture morphing.

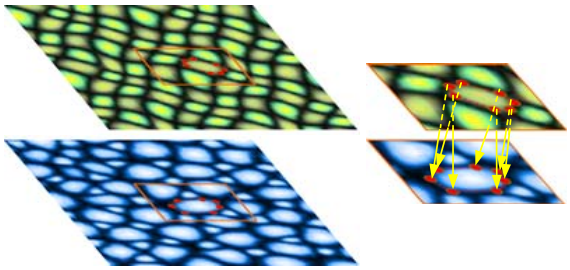
Image morphing. Image morphing consists of three steps – feature and correspondence specification, warp generation, and transition control [24], in order to generate high-quality metamorphosis sequences. Most existing morphing algorithms, including Mesh warping [23], field morphing [3], radial basis functions [1], thin plate splines [15, 19], energy minimization [16], and multilevel free-form deformations [17], focus on generating warp functions. Little work has been done on how computers can aid in building feature correspondence. Gao et. al. [9] present an algorithm for generating the warp function between two similar images with little human interaction.

3. Our Approach

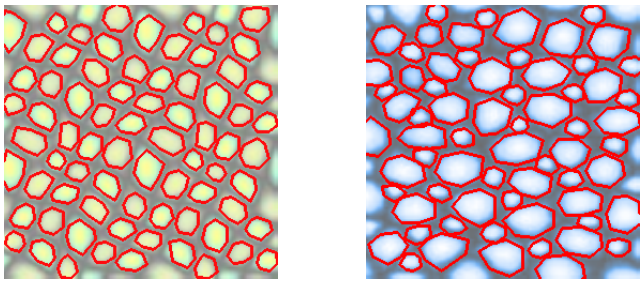
3.1 Observations

We classify texture images into three categories, highly random, semi-structured and regular repeated, as shown in Figure 2. We focus our attention on metamorphosis for regular repeated and semi-structured textures in this paper.

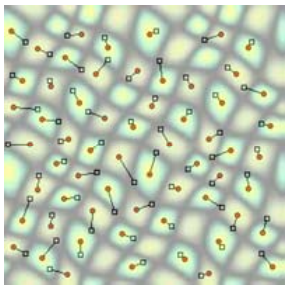
But what kind of correspondence between two textures can generate a smooth and reasonable morphing sequence? Since texture is composed of randomly repeated patterns, it is natural to assume that in the morphing process the patterns in the source image should smoothly warp to those in the target. Thus, the global feature correspondence between the two images is decoupled to a "local feature correspondence" that describes how two local patterns correspond,



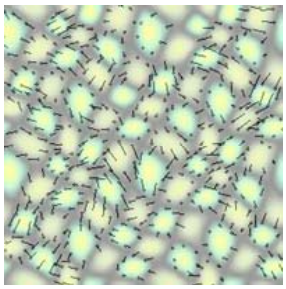
(a) User specifies the representative patterns and the associated correspondence by landmarks (section 4.1)



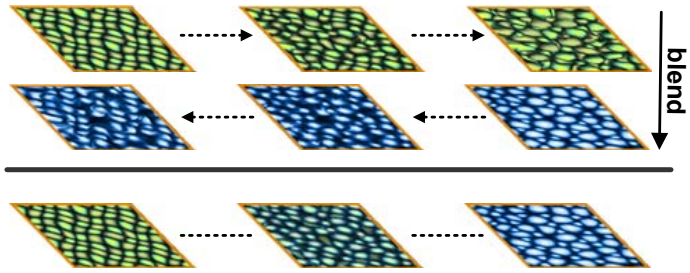
(b) Pattern detection and alignment (section 4.2)



(c) Pattern correspondence (section 4.3)



(d1) Landmark flow (section 4.4)



(d2) Forward, inverse and synthesized sequences acquired based on the warp field (section 4.4).

Figure 3. System overview. (a) The user selects patterns and specifies correspondence by landmarks. (b) Pattern sets obtained interactively in source and target textures by detection and alignment algorithms. (c) Correspondence between patterns (red circle) in source texture and patterns (white square) in target texture. (d1) Landmark flow generated by combining pattern correspondence and pattern alignment. (d2) Warping and morphing

and a "pattern correspondence" to map the patterns from the source to the target. It is the user who selects the most representative patterns and specifies the "local feature correspondence" between them. The "pattern correspondence" can also be manually specified. To reduce user's work, however, we devise an automatic algorithm to find the "pattern correspondence". The problem of texture metamorphosis is formulated as follows.

3.2 Problem Formulation

We use $\mathbf{L}$ and $\mathbf{L}'$ to denote the image lattice for the source and target textures, respectively. The essential task of image morphing is to find a *warp function* [24]

$$Z' = \mathbb{W}(Z), \quad Z \in \mathbf{L} \text{ and } Z' \in \mathbf{L}', \tag{1}$$

where Z and Z' are 2-d coordinates. The warp function is always constructed by feature correspondence. We define the notation of correspondence as $[\cdot, \cdot]$, where "$\cdot$" can be a feature point, a line, a point set or an image. In general, the "global correspondence" of two images is specified by N feature points

$$[\mathbf{L}, \mathbf{L}'] = \{[Z_i, Z_i'],\ i = 1, \cdots, N\}, \tag{2}$$

from which we may find the warp function by interpolation.

Patterns in texture image. In texture morphing, we define the most representative pattern from texture images by m landmarks, for source and target respectively: $U = \{C, Y_1, \cdots, Y_m\}$ where C is the centroid of U, Y_j $(j = 1, \cdots, m)$ is the coordinate relative to C, and $U' = \{C', Y_1', \cdots, Y_m'\}$. Once the patterns are selected, the "local feature correspondence" is obtained

$$[U, U'] = \{[Y_j, Y_j'],\ j = 1, \cdots, m\}. \tag{3}$$

Note that the correspondence can be modified when the index sequence of the feature points changes.

By scanning the whole images, we can get a set of patterns similar to that selected for the source and target respectively

$$\begin{aligned}\Omega_U &= \{U_i | d(U_i, U) < \varepsilon, U_i \subset \mathbf{L}\} \\ \Omega_{U'} &= \{U_i' | d(U_i', U') < \varepsilon', U_i' \subset \mathbf{L}'\}\end{aligned} \tag{4}$$

where $d(\cdot,\cdot)$ is a distance measure between a pattern and the user-specified pattern, and $\varepsilon, \varepsilon'$ are thresholds. $U_i = \{C_i, Y_{i1}, \cdots, Y_{im}\}$. Let the element numbers of Ω_U and $\Omega_{U'}$ be n and n', respectively. Without generalization, we assume $n < n'$. Then the "pattern correspondence" is established upon the pattern pairs

$$[\Omega_U, \Omega_{U'}] = \{[U_i, U_{k_i}'], i = 1, \cdots, n\} \tag{5}$$

where $\{k_i, i = 1, \cdots, n\}$ is an index set.

Pattern-based texture morphing. The global feature correspondence of $\mathbf{L}$ and $\mathbf{L}'$ are established by propagating the local feature correspondence $[U, U']$ to the pattern correspondence $[\Omega_U, \Omega_{U'}]$

$$\begin{aligned}[\mathbf{L}, \mathbf{L}'] &= [\Omega_U, \Omega_{U'}] \otimes [U, U'] \\ &= \{[Z_{ij}, Z_{k_i j}']\}_{i=1\ j=1}^{n\ m}\end{aligned} \tag{6}$$

where $\otimes$ is a tensor product. $Z_{ij} = C_i + Y_{ij}$ is the absolute coordinate of Y_{ij} in $\mathbf{L}$. Figure 3 shows the flowchart of our system, which involves four steps:

(a) The user selects the most representative patterns by landmarks and specifies their correspondence.

(b) Find the pattern set by automatically searching the images. User interaction might be needed to refine the detection.

(c) Find the pattern correspondence automatically by minimizing the cost function of morphing flow.

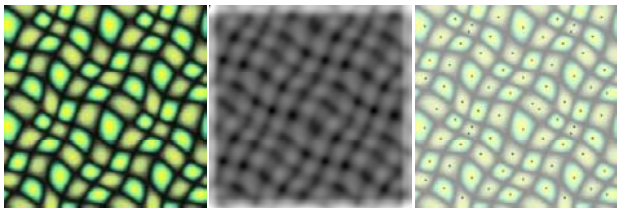
(d) Establish the global feature correspondence by combining the local feature correspondence and pattern correspondence. Generating the texture morphing sequence is the same as that in conventional image morphing.

Details of these steps are presented in the next section.

4. Technical Details

4.1 Pattern and Correspondence Specification

The user must specify a pattern in the source texture and a corresponding one in the target. As shown in Figure 3 (a), the user sequentially specifies the key points $\{Y_1, \cdots, Y_m\}$ in a local region in the source and the corresponding sequence $\{Y_1', \cdots, Y_m'\}$ in the target. U and U' are constructed respectively (the centroid coordinates are computed as means). The pairs Y_j and Y_j' are assumed to make a match in our system, i.e. $[U, U'] = \{[Y_j, Y_j']\}_{j=1}^m$. We may change the indexing order to modify the correspondence. Obviously different users would have different choices of interesting patterns for the same texture. Three factors in this step affect the final outcome of the system: the pattern that the feature points represent, the number of feature points and the order of feature points in the sequence. We will see how these three factors contribute later in the following experiment section.



(a) (b) (c)

Figure 4. (a) Original image. (b) Hough vote intensity image. (c) Candidate pattern centers found by thresholding and searching for local minima.

4.2 Pattern Detection and Alignment

The task of pattern detection is to find the patterns similar to the selected pattern in the source and the target, respectively. For simplicity, we shall discuss how to extract the pattern set Ω_U from the source L.

4.2.1 Distance Measurement

The distance between a pattern U_i and the specified U $d(U_i, U)$ is a weighted sum of shape distance $d_s(U_i, U)$ and local feature distance $d_f(U_i, U)$. The shape distance is computed by scaling U_i to the size of U and summing the Euclidean distance between corresponding landmarks. It has been demonstrated that band-pass oriented filters are capable of capturing the similarity/disimilarity of textures [27]. By filtering we obtain a feature vector for each pixel. Since the pattern is modeled by a deformable template, $d_f(U_i, U)$ is computed by summing the feature distances of all landmark pairs.

4.2.2 Pattern Detection

In the deformable object detection/localization literature, the generalized Hough transform (GHT) is often used to give an initialization [20]. GHT is a bottom-up process to estimate the object parameters from local evidence such as edge, corner or feature detection. Here we choose feature

detection to provide local evidence. If the local feature similarity of a point in the texture to the jth landmark in the specified pattern is above a confidence threshold, it is a candidate for the jth landmark. Each candidate jth landmark votes to a possible centroid area with Gaussian distributed weights. Summing the weights over all possible candidate landmarks, we can get a Hough voting intensity image as shown in Figure 4 (b). The darker the area, the more likely it is to be a pattern at that location. After smoothing the voting intensity image, the rough estimation of pattern number and position is achieved by thresholding and local minimization, as shown in Figure 4 (c).

4.2.3 Pattern Alignment

GHT alone is not accurate enough to extract patterns. Therefore, a top-down verification process is needed to refine the positions of the patterns by minimizing the distance of each candidate pattern to the specified pattern. The strategy adopted here is very similar to Active Shape Model (ASM) [6]. Each pattern is separately aligned. First, we independently update each landmark to minimize the corresponding feature distance, then the shape is updated by a step toward the shape of the specified pattern. These two steps are iteratively performed.

4.3 Pattern Correspondence

Pattern correspondence needs to be established to provide a morphing path for each pattern. To satisfy human visual comfort, we try to make the morphing path as smooth as possible. We assume each pattern U_i in the source should match a pattern U'_i in the target with minimal shift, i.e. the distance between them. Thus, we can obtain the pattern correspondence by minimizing the summed distance of the morphing path.

The distance between two patterns in the source and target images respectively can be simply measured by the distance between their centroids. As discussed in Section 3.2, the pattern number in the source image is less than the target number $n < n'$. Thus, every pattern $U_i \in \Omega_U$ has a match in $\Omega_{U'}$. To compute the pattern map field $[\Omega_U, \Omega_{U'}]$ is indeed to find the index set $\{k_i\} \subset \{1, \cdots, n'\}$ in Eqn.(5), by minimizing the cost function of morphing path

$$\{k_i^*\}_{i=1}^n = \arg\min \sum_{i=1}^{n} ||Z_i - Z'_{k_i}||^2. \tag{7}$$

This problem can be easily solved by integer programming [25]. Let $d_{ij}^2 = ||Z_i - Z'_j||^2$ denote the squared distance of patterns U_i and U'_j. Since $n < n'$ we add $n' - n$ virtual patterns after $\{U_i\}_{i=1}^n$ to make the pattern number of the source and target identical, and define $d_{ij}^2 = 0$ for $i > n$. By introducing a coefficient matrix $[a_{ij}]_{n' \times n'}$ where each element is a binary variable $a_{ij} = \{0, 1\}$, the cost function Eqn.(7) is reformulated as

$$[a_{ij}]^* = \arg\min \sum_{i=1}^{n'} \sum_{j=1}^{n'} a_{ij} d_{ij}^2 \tag{8}$$

with constraints

$$\begin{aligned} &\sum_{i=1}^{n'} a_{ij} = 1, \text{ for } j = 1, \cdots, n' \\ &\sum_{j=1}^{n'} a_{ij} = 1, \text{ for } i = 1, \cdots, n'. \end{aligned} \tag{9}$$

Namely there is only one $a_{ij} = 1$ for each row and column. The Hungarian algorithm [25] is used to solve this problem. Finally, $\{k_i^* = j | a_{ij}^* = 1\}_{i=1}^n$ generates the global feature correspondence by Eqn.(6).

Two adjacent patterns may appear overlapping along their morphing paths if the patterns are too crowded. To address this problem, we may scale down the size of source patterns before morphing, and scale them up to the target size after that.

4.4 Generating Morphing Sequence

Combining the manually specified local feature correspondence and the automatically optimized pattern correspondence, we may generate the global feature correspondence between the source and target by Eqn.(6). Then the remaining work is the same as in general image morphing. A sparse points interpolation technique MFFD [17] is adopted to build up a warp function or a warp field. This field is further used to produce a morphing effect by color interpolation. In addition, median filtering can be chosen to smooth out some inconsistency in the morphing process.

5. Experiments

We have applied our system to a number of texture pairs, shown in Figures 5 and 1. Figure 5 (a) is a simple case with identical pattern numbers in both the source and target. In the morphing process, the position of each pattern remains invariant, while the shape smoothly transforms to the target. Since we merely select the major patterns and ignore smaller ones, they appear or disappear in the sequence. Figure 5 (b) is more difficult because the pattern numbers in the source and target are different. Therefore, some patterns in the source may gradually vanish into the background of the target. Note that manually specifying feature correspondences is infeasible for this case. In Figure 5 (c), we select the vertical edge between two adjacent bricks with a short

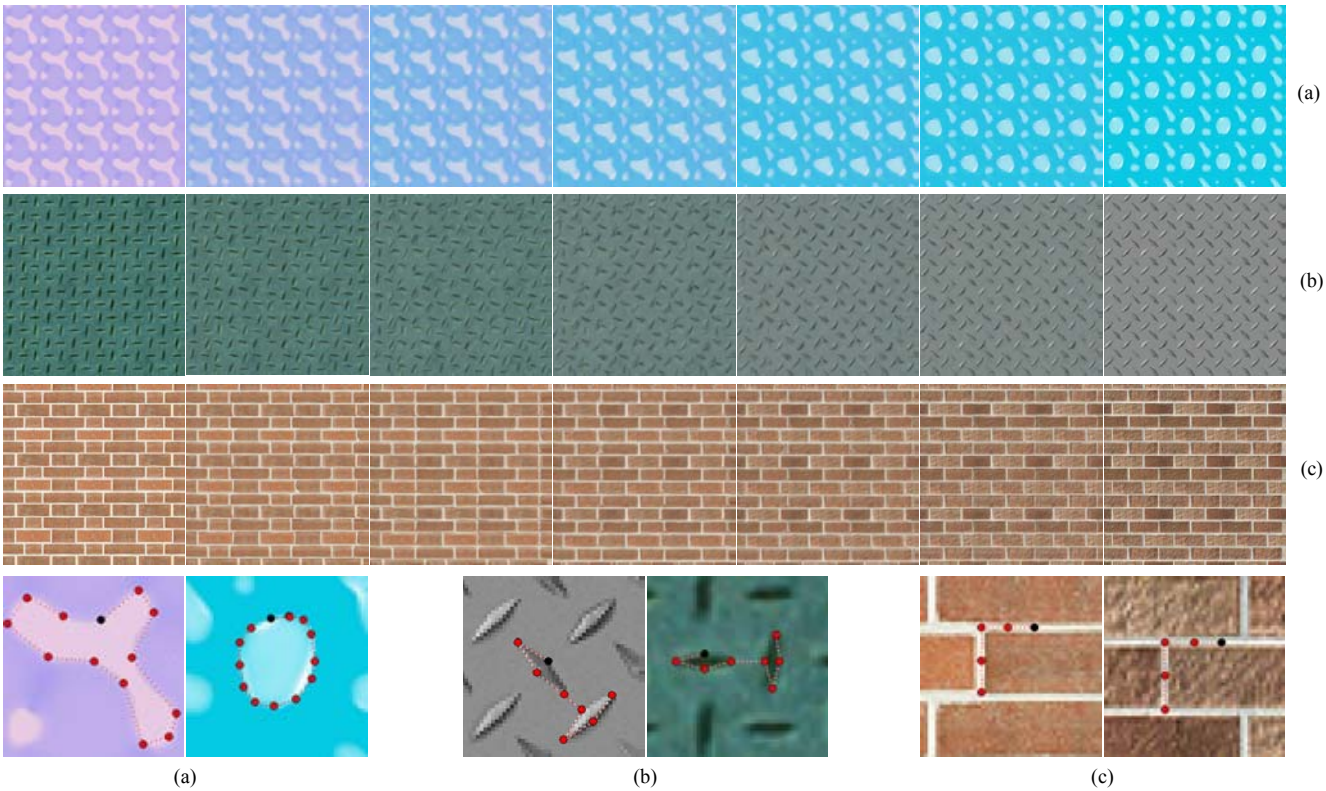


Figure 5. Experimental results of texture morphing and the patterns selected by the user.

horizontal edge as the pattern. In this way the horizontal and vertical structures of the source bricks are retained in the morphing sequence and smoothly transformed to the target.

A very challenging example is shown in Figure 1, where the distributions of both the source and target are stochastic and different from each other. Since the scales of the two patterns are distinguished, we have to enlarge one of them to keep their scales similar. The patterns distributed in the source and target are both crowded, which would produce overlapping effects if we directly apply our algorithm. As suggested in Section 4.3, we first scale down the size of the patterns in the source, then perform the standard texture morphing algorithm, and finally scale up the patterns to the target size. A median filter is applied at the post-processing stage to improve the consistency of the morphing sequence. The synthesized morphing sequence demonstrates that our algorithm deals with semi-structured textures very well.

The choice of the patterns plays an essential role in synthesizing the morphing sequence. As Figure 5 (c) has demonstrated, a pattern is not necessarily a polygon, but could be a composition of landmarks which can represent various 2D objects. This allows for a wider range of application. Different selections of patterns lead to visually different effects. Figure 6 (a) denotes the pattern specification in the source, whereas (b) and (c) indicate two different patterns in the target. Corresponding morphing sequences are shown in Figure 7.

Two other significant aspects are the number of landmarks and the order of their correspondence. A comparison is given in Figure 8. Although the selected patterns in both the source and target are dark squares, (a) and (c) are specified with four corner points, while (b) and (d) with eight landmarks (four corners plus four middle points, each lying between two adjacent corners). The orders of the local correspondence in (a) and (b) are different from those in (c) and (d). Therefore, (a) and (b) appear as shifting, and (c) and (d) rotating. We observe that the more landmarks used, the more accurately the user's intention is modeled. Eight landmarks produce a smoother morphing sequence than four landmarks.

6. Conclusion

In this paper we have proposed a pattern-based approach for metamorphosis between two textures, each containing discernable and similar patterns. Global feature correspondence between two texture images is decomposed to local feature correspondence between two selected patterns from

the source and target respectively, and pattern correspondence which maps the patterns in the source image to the target. An advantage of our approach is that only a moderate amount of user interaction is required for texture morphing. Extensive experiments demonstrate that our algorithm can generate some interesting texture morphing sequences.

References

[1] N. Arad, N. Dyn, D. Reisfeld, and Y. Yeshurun. Image warping by radial basis functions: applications to facial expressions. *CVGIP: Graph Models Image Processing*, 56:161–172, 1994.

[2] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman. Texture mixing and texture movie synthesis using statistical learning. *IEEE Transactions on Visualization and Computer Graphics*, 7(2), 2001.

[3] T. Beier and S. Neely. Feature-based image metamorphosis. *SIGGRAPH'92*, pages 35–42, 1992.

[4] J. D. Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. *SIGGRAPH'97*, 1997.

[5] S. Brooks and N. Dodgson. Self-similarity based texture editing. *SIGGRAPH'02*, 2002.

[6] T. Cootes and C. Taylor. Statistical models of appearance for computer vision. Technical report, University of Machester, 2000.

[7] A. Efros and W. Freeman. Image quilting for texture synthesis and transfer. *SIGGRAPH'01*, 2001.

[8] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. *ICCV*, 1999.

[9] P. Gao and T. Sedergerg. A work minimization approach to image morphing. *Visual Computer*, 1998.

[10] C. E. Guo, S. C. Zhu, and Y. N. Wu. Visual learning by integrating descriptive and generative methods. *ICCV*, 2001.

[11] D. J. Heeger and J. R. Bergen. Pyramid based texture analysis/synthesis. *SIGGRAPH'95*, pages 229–238, 1995.

[12] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. *SIGGRAPH'01*, 2001.

[13] B. Julesz. Visual pattern discrimination. *IRE Transactions of Information Theory*, pages 84–92, 1962.

[14] B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97, 1981.

[15] S. Lee, K.-Y. Chwa, J. Hahn, and S. Shin. Image morphing using deformable surfaces. In *Computer Animation'94*, pages 31–39, Geneva, Switzerland, 1994.

[16] S. Lee, K.-Y. Chwa, J. Hahn, and S. Shin. Image morphing using deformation techniques. *The Journal of Visualization and Computer Animation*, 7(1):3–23, 1996.

[17] S. Lee, K.-Y. Chwa, S. Shin, and G. Wolberg. Image metamorphosis using snakes and free-form deformations. *SIGGRAPH'95*, pages 439–448, 1995.

[18] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and Heung-Yeung. Real-time textures synthesis by patch-based sampling. *TOG*, 20(3):127–150, July 2001.

[19] P. Litwinowicz and L. Williams. Animating images with drawings. *SIGGRAPH'94*, pages 409–412, 1994.

[20] C. Liu, H.-Y. Shum, and C. S. Zhang. Hierarchical shape modeling for automatic face localization. *ECCV*, 2002.

[21] L.-Y. Wei. *Texture synthesized by fixed neighborhood searching*. PhD thesis, Stanford, 2002.

[22] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. *SIGGRAPH'00*, 2000.

[23] G. Wolberg. *Digital image warping*. IEEE Computer Society Press, Los Alamitos, Calif, 1990.

[24] G. Wolberg. Image morphing: a survey. *Visual Computer*, pages 360–372, 1998.

[25] L. Wolsey and G. L. Nemhauser. *Integer and Combinatorial Optimization*. Wiley Press, June 1999.

[26] Y.-Q. Xu, B. Guo, and H.-Y. Shum. Chaos mosaic: Fast and memory efficient texture synthesis. Technical Report MSR-TR-2000-32, Microsoft Research, April 2000.

[27] S. C. Zhu, Y. N. Wu, and D. Mumford. Minimax entropy principle and its application to texture modeling. *Neural Computation*, 9(8), November 1997.

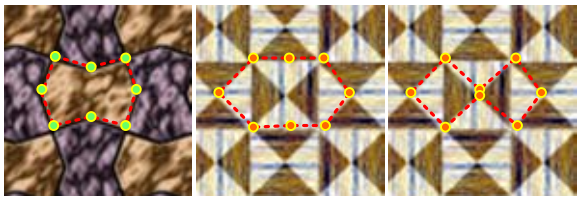


Figure 6. User may specify different patterns in target as shown in (b) and (c).

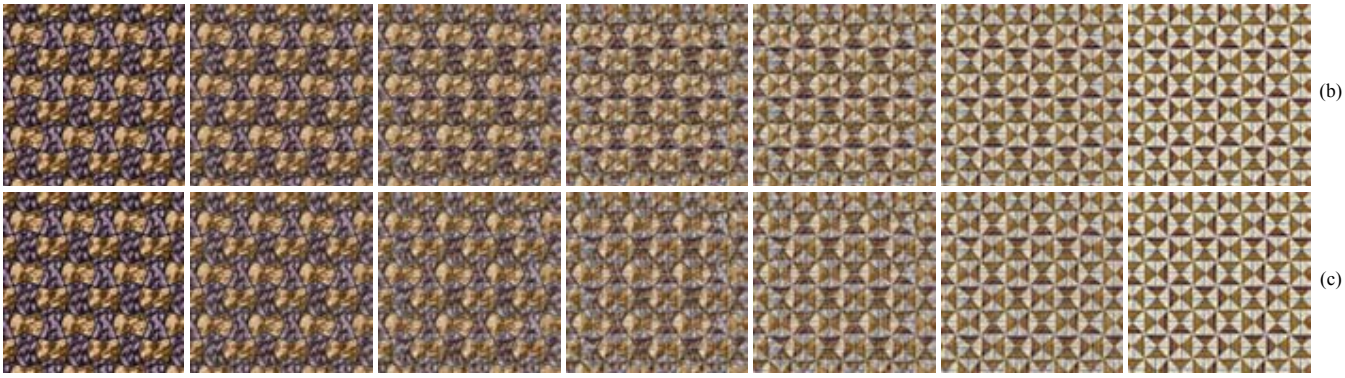


Figure 7. The morphing sequences with different selected pattern in Figure 6.

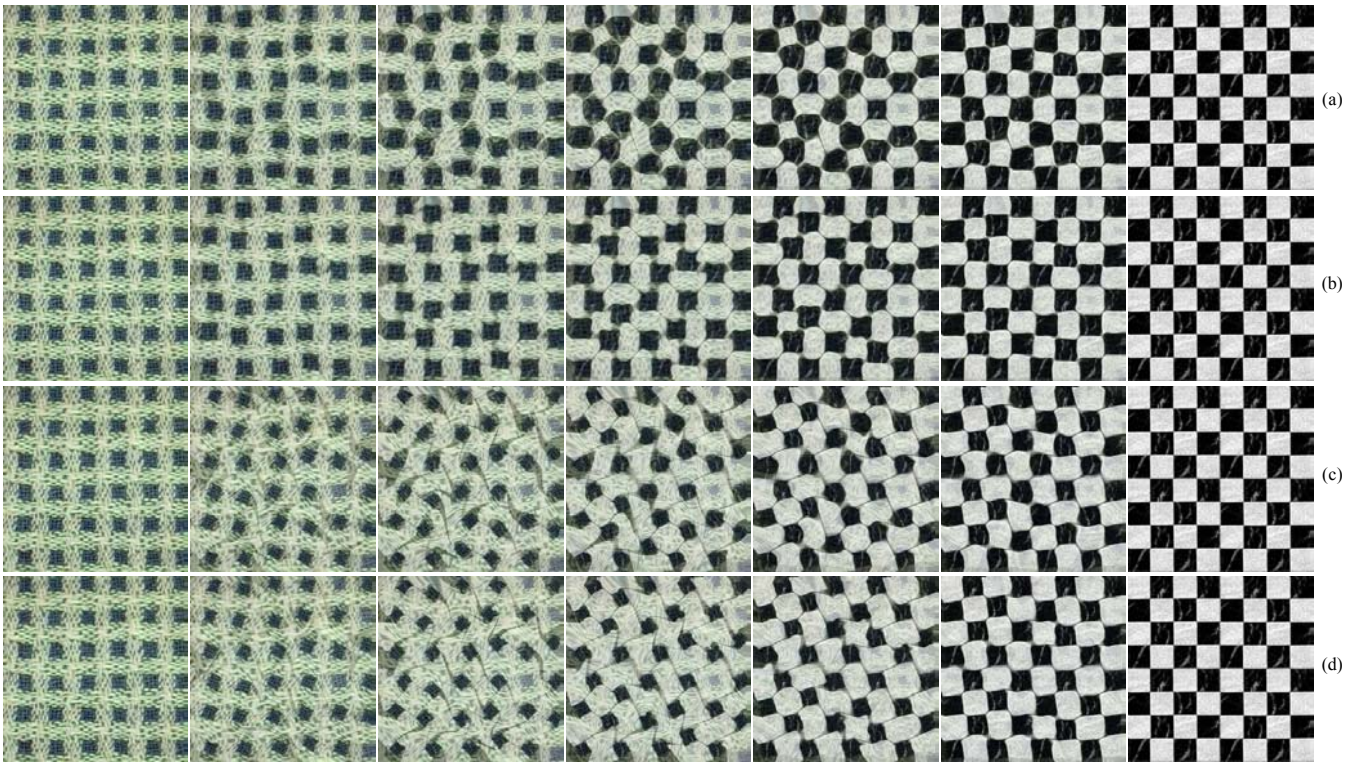


Figure 8. The effects of landmark number and local correspondence. (a) and (c) are specified with four landmarks, while (b) and (d) with eight landmarks. The local correspondences of (a) and (b) are different with (c) and (d).