Morphing Textures with Texton Masks

Leandro Tonietto
tonietto@exatas.unisinos.br
UNISINOS
PIPCA - Mestrado em Computação Aplicada

Marcelo Walter
marcelow@exatas.unisinos.br
UNISINOS
PIPCA - Mestrado em Computação Aplicada

Abstract

Image morphing has been extensively studied in computer graphics. Given two input images, morphing algorithms produce a sequence of inbetween images which transforms the source image into the target image in a visually pleasant way. In this paper, we propose an algorithm, based on recent advances from texture-from-sample ideas, which synthesizes a morphing sequence targeted specifically for textures. We use the idea of binary masks (or texton masks*) to control the morphing sequence and guarantee a coherent transition from the source texture to the target texture.*

1 Introduction

The transformation of one source image into a target image is a technique generally referred to in computer graphics as two dimensional morphing, or just *morphing*. Since the introduction of the first morphing algorithm in the 1992 Black and White video [3], the power of this technique is being extensively explored in the entertainment industry (movies, games and television). One of the main issues related with these morphing techniques is the control of results. In order to achieve a visually smooth transition (for instance, when morphing two faces, an eye in the source image should map also to an eye in the target image) most techniques make use of user-defined features which will control the morphing process and guarantee a pleasant result. In this paper we address the morphing of textures [1], which are not amenable for control by features, since there are no relevant features to start with. For example, consider the two textures shown in Figure 1. Intuitively, one might expect that each green structure should morph into a blue structure. It is also obvious that a simple interpolation from source pixels into target pixels will not, in general, result in a good visual result, in the sense that the structures are not preserved.

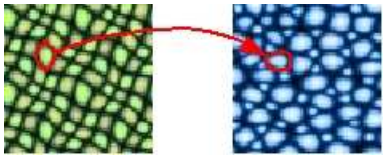


Figure 1. Two example textures.

A general solution to this problem for any two arbitrary input textures is very hard. In [15] Liu and colleagues presented a solution to this problem. Their solution synthesizes smooth morph sequences, however the user must specify a correspondence between a pattern in the source image and a pattern in the target image. We explore in this paper an alternative approach without user intervention. Our main idea is to use texton masks to control the transition of the source texture into the target. The texton masks can be automatically computed from the original textures and help identify elements in both textures. These elements, once identified, can be made to morph into one another. This idea explores recent advances in image-based texture synthesis, particularly the synthesis of progressively variant textures (PVTs) [24] in the context of morphing textures. At the same time, since our solution is based on texture from sample ideas, the output resolution of the inbetween images can be arbitrarily established. In Figure 2 we illustrate a result from our algorithm.

2 Related Work

Our work combines research from texture synthesis from samples and image morphing. Therefore we present the relevant work in both areas.

Texture synthesis. Texture synthesis is an old research subject in both Image Processing and Computer Graphics

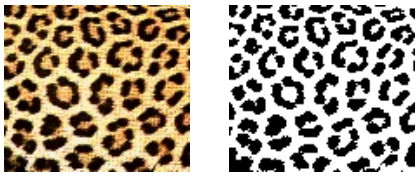
[1] It is relevant here to explain how images in general differ from textures. As presented in [21], for an image, different regions defined through a moving square are perceived to be different, while for textures they are perceived to be similar.

Figure 2. Example of morphing sequence result. Image resolution = $64x64$**.** $N(p_S) = 11$**,** $N(p_T) = 15$**. For meaning of parameters see Section 3.**

fields, with research going back as far as the late seventies [16]. The many different models and approaches have always tried to generate textures either to validate texture models (mostly in Image Processing tasks) or simply to use the result in some application. The idea of using a sample as input information to create the result has always been present (see for instance [16, 17, 4, 8]). Until recently, despite progress, such techniques were either too slow to be of practical use or the results were not general enough to be useful [9, 5, 18].

The work of Efros and Leung presented in 1999 [7] introduced a new simple way of looking at this problem by "growing" a texture one pixel at a time from an initial seed. The color of a given pixel is determined by scanning over square patches of the sample texture that are similar to the patch on the texture being generated. A random patch in the sample is selected among the few satisfying the similarity criterion. The similarity is measured with a $L2$ norm (sum of squared differences) weighted by a Gaussian kernel. The original Efros and Leung's algorithm is slow and recent extensions have improved its performance, particularly the work of Wei and Levoy [21]. They have used a raster scan ordering to transform noise pixels into the result texture and have also improved the performance of the algorithm by using a multi-scale framework and vector quantization. Their approach also minimizes the $L2$ norm in RGB space but without any weighting. Efros and Freeman introduced yet another way of synthesizing image-based textures by stitching together random blocks of the sample and modifying them in a consistent way [6]. They call the technique "image quilting". The idea improves dramatically on the one-pixel-at-a-time approach since it builds the texture at a much coarser scale while being able to keep high frequencies of the sample. The same idea of using patches from the sample to synthesize the result was explored by Liang *et al* [13]. In this work they were able to achieve real-time generation of large textures using special data structures and optimization techniques.

More recently, Zhang and colleagues introduced an image-based texture synthesis method for rendering of *Progressively Variant Textures* (PVTs) [24]. Although not formally defined in the paper, the concept of PVT is an important one for texture synthesis, since it captures the class of textures where the texture elements vary in a progressive fashion, typical examples being mammalian fur patterns (e.g., leopard skin, Figure 3(a)). From an homogeneous texture sample they were able to synthesize a PVT. One main idea in their work, explored in our approach, was the notion of *texton masks*. A texton mask is a binary image marking prominent features or texture elements in the texture. In their synthesis algorithm, the texton masks prevent the disintegration of texture elements during synthesis. In Figure 3 we illustrate a texton mask for a leopard skin texture.



(a) Texture (b) Mask

Figure 3. Example of Leopard Texture and corresponding Texton Mask.

Image Morphing Image morphing techniques were introduced in the movie Willow [19] and in the Black and White video [3] and have become a very popular technique for special effects. The main idea is to compute, given a source and a target image, a sequence of images such that the user has the feeling that somehow the source image transformed (or it was *morphed*) smoothly into the target image. For images in general the problem can be broken down into three steps [22, 23]: feature specification, warp generation, and transition control. Feature specification is used to compute a mapping function that defines the spatial relationship between the points in the source and target images; warp generation is concerned with how the feature specification is used to derive changes for the whole image; transition control specifies the rate of warping and color blending across the morph sequence. Many advances have been made since the introduction of the first morphing techniques, particularly for computing the warp functions. The main ap-

proaches are mesh warping [19], field morphing [3], radial basis functions [2], thin plate splines [10], energy minimization [11], and multilevel freeform deformations [12].

Texture morphing can be viewed as a particular case of image morphing. There is not much work targeted specifically for this task. In [20], a solution is presented for a related case where the images contain a single pattern element and the morphing sequence maintains the pattern coherence throughout the transformation (for example, a baby deer morphing into an adult deer). The closest approach to ours is the work of Liu *et al* [15]. They presented a solution for morphing a source texture into a target texture with the user defining how a *pattern* from the source texture maps to a *pattern* into the target texture. A pattern is defined as a "...semantic unit that composes a group of feature points" [15]. Their solution suffers from a common drawback common to all feature-based transformations, which is the specification of features. For textures in general there are no relevant features that could be used for controlling the transition. This makes the problem more interesting. Simple interpolation will not, in general, map textons from the source image into textons of the target image. We use the definition of texton as a prominent group of pixels in the texture, as presented in [24]. We illustrate in Figure 4 what we mean by textons.

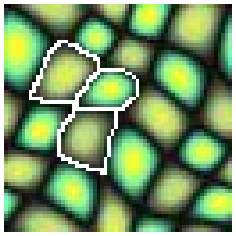


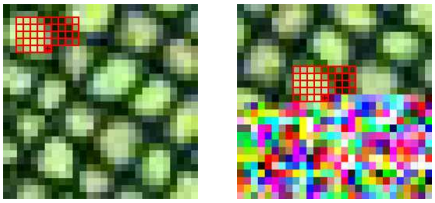
Figure 4. The idea of Textons.

Our approach computes a texton mask for the source and target images and uses this mask to control the interpolation, as presented in the next section.

3 Our Solution

The main idea behind our solution is the use of texton masks to control the morphing process. Basically, we want to guarantee that a texton from the source image will map to a texton in the target image. We augment the texture synthesis process of Wei&Levoy [21] (here called the *WL algorithm*) with the information from the masks, inspired on the use of masks from [24].

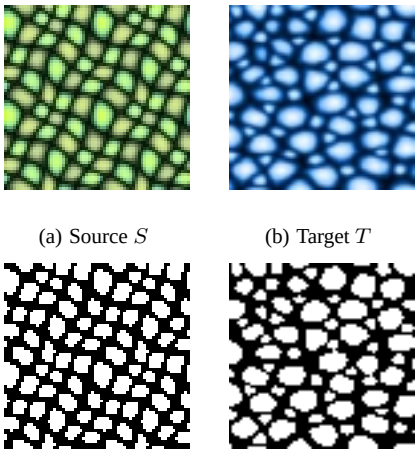
Our algorithm is an extension on the basic algorithm of Wey and Levoy [21] and therefore we start this section with a brief overview of their approach. Figure 5 illustrates the general idea.



(a) (b)

Figure 5. Illustration of Wey and Levoy's synthesis algorithm. (a) Sample being scanned for the best match (b) Noise pixels being transformed into the final texture. The images have been enlarged to illustrate the algorithm.

The algorithm starts with a small texture sample as input (Figure 5 (a)). This sample will be used to construct a texture with the same overall visual appeal as the sample. Even though it is not strictly necessary, the synthesized texture is usually larger than the sample. In order to do this a noise texture is transformed in a raster scan ordering one pixel-at-a-time (Figure 5 (b)). Starting with the pixel at the upper leftmost corner, the algorithm searches on the sample for the best match measured with a $L2$ norm in RGB space. The search uses a user-defined neighborhood size. The sample is searched using continuous boundary conditions (toroidal).



(a) Source S (b) Target T

(c) Mask M_S for (a) (d) Mask M_T for (b)

Figure 6. Example of input for our algorithm.

Now we present our solution. Our problem can be stated as: given an input source texture S, a corresponding input texton mask M_S for S, an input target texture T, and

a corresponding input texton mask M_T for T, compute m textures O_m which will be called the *morph sequence*. In general, S, T, and O_m do not necessarily have the same resolution. For each texture O_m being computed, we build a corresponding mask M_m which is a weighted combination from masks M_S and M_T. In Figure 8 we illustrate two of these masks. The weight is linearly proportional to the distance from the source. This mask is responsible for mantaining the coherence between frames and therefore for the whole morphing sequence. In Figure 6 we illustrate a possible input for our algorithm.

3.1 Computing the Texton Masks

The computation of the texton masks is automatic and simple. We compute a binary image by color thresholding the grayscale image of the input textures S and T. We convert the color texture into a grayscale luminance value using the following formula [1]: $Y = 0.2125R + 0.7154G + 0.0721B$. The greyscale luminance is converted to either a black or white pixel using a user-specified threshold for each input texture. Although very simple, the computed masks capture the texton structures, as we can see in Figure 6(c)(d). A slightly more elaborated solution, as presented in [24], asks the user to specify a foreground and a background color for thresholding.

3.2 Synthesis

Once the masks are computed, our synthesis procedure follows the WL algorithm. The algorithm follows a raster scan ordering, starting in the upper left corner. To generate the O_k frame, the algorithm requires S, M_S, T, M_T, O_{k-1} and M_{k-1}. Exceptionally for the first frame, O_{k-1} and M_{k-1} are noise images. Figure 7 shows a schematic of our morphing textures algorithm.

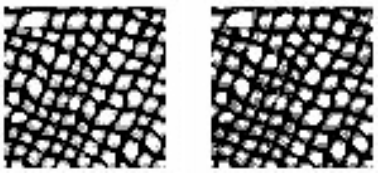


Figure 8. Example of intermediate masks.

Below we summarize our synthesis procedure for a given intermediate texture O_k:

1. Find best match in M_S following the WL algorithm on mask M_{k-1}. Call this best match pixel b_S
2. Find best match in M_T following the WL algorithm on mask M_{k-1}. Call this best match pixel b_T
3. Compute $b = prob_k * b_S + (1 - prob_k) * b_T$
4. Update M_k with the b gray value. Note that this mask M_k will be used in the $M_(k + 1)$ synthesis.
5. Find best match in S following the WL algorithm on texture S, taking into account b_S. Call this best match pixel p_S
6. Find best match in T following the WL algorithm on texture T, taking into account b_T. Call this best match pixel p_T
7. Compute final best pixel $p = prob_k * p_S + (1 - prob_k) * p_T$
8. Return O_k and M_k.

The main departure from the basic WL algorithm is the use of information from the masks (steps 1,2,3) to decide on the final best match (steps 4,5,6). When searching for the best match in the source and target textures, this best match will have to be of the same type as the information b computed in step 3. Intuitively, this means that the best match will have to satisfy both a color criterion (norm $L2$ in RGB space) and a "mask" criterion, that is, the best match must be of the same type as its mask, expressed by the value b.

Steps 3 and 6 make use of a probability $prob_k$ for computing the final best pixel p. This probability captures, for the morph sequence, how far we are on the morphing process from the source texture, based on the number m of desired intermediate textures. For an intermediate texture O_k, $k = 0, \ldots, m$ $prob_k$ is computed as follows: $prob_k = \frac{m-k}{m}$. When searching for the best match, the WL algorithm uses a user-specified neighborhood $N(p)$ around the pixel p. For our solution the user will have to specify two neighborhoods, one for the input texture $N(p_S)$ and one for the input texture $N(p_T)$.

4 Results

Here we present some results with our approach. For the first sequence of results, in Figure 9, we used the same source and target textures as presented in [15]. We can see that our approach achieved a similar result while not demanding feature specification from the user.

For comparison we present in Figure 10 the result of simple interpolation between corresponding pixels in the same source and target images from Figure 9. Even though it is difficult to assess the differences, we can see that in our approach, particularly the 4th and 5th frames, the texton structure is more visible than for the interpolation case (the textures are more blurred).

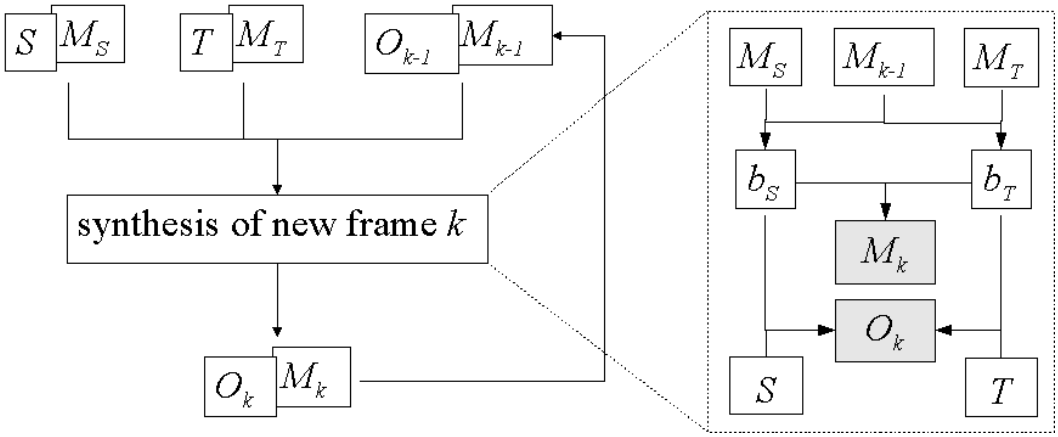


Figure 7. Schematic representation of our texture synthesis algorithm with texton masks.

In general we can see that our solution generates visually adequate morphing sequences, similar to the results presented in [15], although in their case, since the user controls how the mapping is established, more elaborated transitions are possible.

5 Conclusions

We presented a morphing technique specifically for textures. Our solutions automatically produces a series of transition textures between a source and a target textures. For both source and target textures we compute a texton mask, and from these masks we derive the transitions. Since our approach combines texture morphing with texture synthesis from samples, we can generate arbitrarily sized morphing sequences, a feature not presented in [15]. An interesting challenge would be to enhance our solution using a patch-based texture synthesis approach, as presented in [14].

Acknowledgements

We acknowledge the partial financial support from CAPES and useful discussions with Soraia R. Musse.

References

[1] T. Akenine-Moller and E. Haines. *Real-Time Rendering*. A K Peters, Ltd., 2002.

[2] N. Arad, N. Dyn, D. Reisfeld, and Y. Yeshurun. Image warping by radial basis functions: Application to facial expressions. *CVGIP: Graphical Models and Image Processing*, 56(2):161–172, Mar. 1994.

[3] T. Beier and S. Neely. Feature-based image metamorphosis. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 35–42. ACM Press, 1992.

[4] G. Cross and A. K. Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(1):25–39, Jan. 1983.

[5] J. S. de Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In T. Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 361–368. ACM SIGGRAPH, Addison Wesley, Aug. 1997. ISBN 0-89791-896-7.

[6] A. Efros and W. Freeman. Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001*, pages 341–346, August 2001. ISBN 1-58113-292-1.

[7] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *International Conference on Computer Vision*, volume 2, pages 1033–1038, 1999.

[8] A. Gagalowicz and S. Ma. Model driven synthesis of natural textures for 3-D scenes. In *Eurographics '85*, pages 91–108. 1985.

[9] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In R. Cook, editor, *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 229–238, Aug. 1995.

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

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

[12] S.-Y. Lee, K.-Y. Chwa, S. Y. Shin, and G. Wolberg. Image metamorphosis using snakes and free-form deformations. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, pages 439–448, Aug. 1995.

[13] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics*, 20(3):127–150, July 2001. ISSN 0730-0301.

[14] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics (TOG)*, 20(3):127–150, 2001.

Figure 11. Another example of morphing sequence result.

[15] Z. Liu, C. Liu, H.-Y. Shum, and Y. Yu. Pattern-based texture metamorphosis. In *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, pages 184–191. IEEE Computer Society, 2002.

[16] S. Lu and K. Fu. A syntactic approach to texture analysis. *Computer Graphics and Image Processing*, 7(3):303–30, June 1978.

[17] J. Monne, F. Schmitt, and D. Massaloux. Bidimensional texture synthesis by markov chains. *Computer Graphics and Image Processing*, 17(1):1–23, Sept. 1981.

[18] E. Simoncelli and J. Portilla. Texture characterization via joint statistics of wavelet coefficient magnitudes. In *Fifth IEEE International Conf on Image Processing*, volume I, pages 62–66, Chicago, Illinois, October 1998. IEEE Computer Society.

[19] D. Smythe. A two-pass mesh warping algorithm for object transformation and image interpolation. Technical Report TR1030, ILM Computer Graphics Dept., 1990.

[20] A. Tal and G. Elber. Image morphing with feature preserving texture. In P. Brunet and R. Scopigno, editors, *Computer Graphics Forum (Eurographics '99)*, volume 18(3), pages 339–348. The Eurographics Association and Blackwell Publishers, 1999.

[21] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 479–488, July 2000.

[22] G. Wolberg. *Digital Image Warping*. IEEE Computer Society Press, 1990.

[23] G. Wolberg. Image morphing: a survey. *The Visual Computer*, 14(8-9):360–372, 1998.

[24] J. Zhang, K. Zhou, L. Velho, B. Guo, and H.-Y. Shum. Synthesis of progressively variant textures on arbitrary surfaces. *ACM Transactions on Graphics*, 22(3):295–302, July 2003.

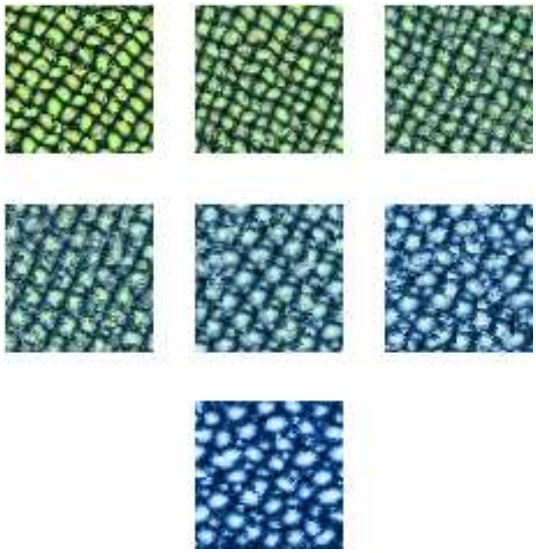


Figure 9. Example of morphing sequence result. Test case taken from [15].

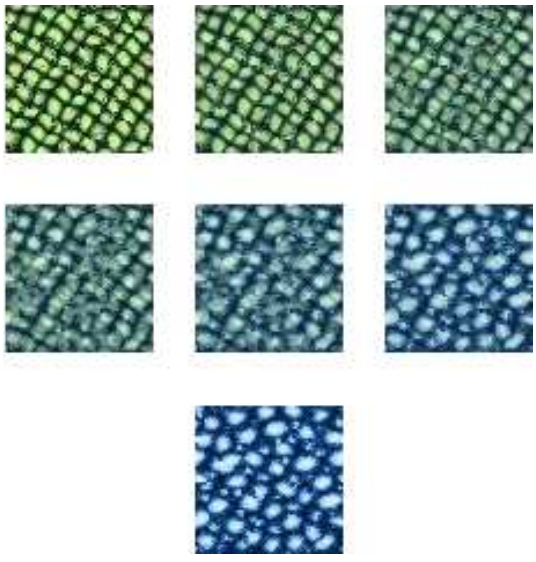


Figure 10. Example of morphing sequence with linear interpolation.