Patchwork texture synthesis

Paul Harrison
(pfh@csse.monash.edu.au)

July, 2002

Abstract

In this technical report, a simple and fast method is described for the synthesis of texture from a sample. This method produces results comparable to those of more sophisticated methods. The basic principle employed is to piece together patches of a sample into a larger output texture. These patches are chosen at random, and have random shape. A method to disguise the edges of the patches is described. The texture synthesis method is applied to still images and video. Its application to texture transfer is also described.

Keywords: texture synthesis, texture transfer, video texture

1 Introduction

Synthetic textures are widely used in computer graphics, for example to increase the realism of computer generated images. One method of synthesizing texture is from a sample. For example, given a picture of grass, a larger picture of grass might be synthesized. In this technical report, image and video texture are considered. A simple and fast technique for texture synthesis is described that gives high quality results.

Many papers have been published on the topic of texture synthesis from a sample. One technique pieces together the texture pixel by pixel, for each pixel choosing the *best fit* to the part of the pixel's neighbourhood which has already been synthesized. [4, 5]. This can produce high quality results, but is slow. To produce each pixel, the entire input image must be searched. A number of variations on this best fit method have been published [1, 6, 12].

Another technique examines relationships between elements in a feature hierarchy. For example, it might be determined that in some texture, certain smaller features occur in the presence of certain large features. Examples of feature hierarchies that have been used are wavelets [11] and image pyramids [2, 7, 12].

The majority of these techniques directly copy large sections of the sample texture to the output. Notable exceptions are Heeger and Bergen's image pyramid method [7] and Portilla and Simoncelli's complex wavelet method [11], all other cited techniques have this property.

A number of texture synthesis methods make the copying process explicit. For example, in Efros and Freeman's *image quilting* method [3, 9] an overlapping grid of tiles is produced. Each tile is simply copied from the sample texture. Tiles are chosen so that their overlapping regions match as closely as possible.

Xu et al.'s *chaos mosaic* method [13] takes the idea of copying from the sample one step further. Tiles chosen randomly from the sample texture are randomly pasted onto the output image. Feathering is used to try to disguise tile borders. No attempt is made to match tiles to each other.

This technical report examines whether copying random sections of a sample texture is all that is necessary for texture synthesis. To do this, a simple method of texture synthesis is proposed that operates using this principal. The texture synthesis method is first applied to images, then to video. Finally, its extension to texture transfer is examined.

$O(x)$	Ouput image
$I(x)$	Input image
p_i	Set of points in the output image
q_i	Set of points in the input image
f	Parameter specifying patch border sharpness
$d(a, b)$	Euclidean distance between a and b

Table 1: List of symbols.

2 Method

The texture synthesis method presented in this technical report simply copies random patches of the input to the output, so that the output is completely covered. The edges of these patches are blended together so as to be invisible.

The difficulty lies in making edges within the output invisible. Several types of edges are illustrated in Figure 1. As can be seen, a sharp transition (Figure 1a) is clearly visible, while a gradual transition (Figure 1b) looks duller than the original texture. This kind of transition is sometimes called feathering, and was used by Xu et al. [13]. This dullness can be removed by controlling the standard deviation of each pixel in the transition, producing an invisible join (Figure 1c).

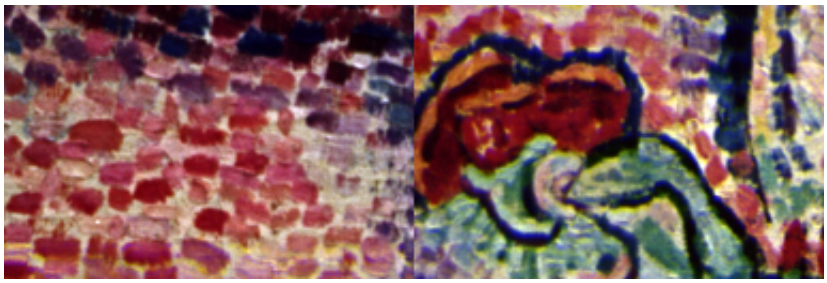
An irregular pattern of patches is used, as the human eye is skilled at discerning regular patterns. A Voronoi diagram, based on a set of randomly chosen points, produces a patchwork that is sufficiently irregular.

3 Image texture synthesis

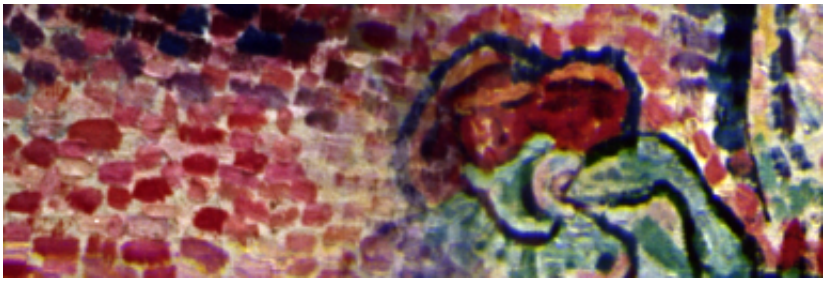
In this section, the texture synthesis method is applied to images. Symbols used in this section are listed in Table 1.

First, define the Voronoi cells that are to be used as patches. Choose n points p_i in the output image $O(x)$. The points p_i form the centers of Voronoi cells, and should be chosen to cover the output evenly. One simple way to achieve

(a)



(b)



(c)

Figure 1: Joining one texture patch to another by (a) a sharp transition, (b) a gradual transition and (c) a gradual transition controlling for standard deviation.

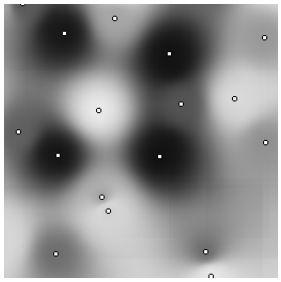
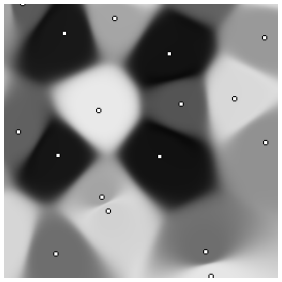
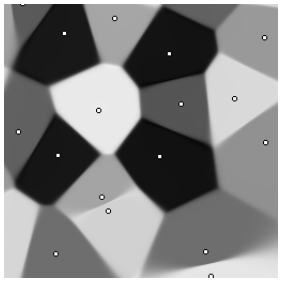
(a)  (b)  (c) 

Figure 2: Example of patch shapes about the points p_i for (a) $f = 2$, (b) $f = 6$ and (c) $f = 20$

this is to choose jittered grid points. The size of the grid determines the approximate size of each cell, as well as the number of points required.

Also choose n points q_i from the input image $I(x)$ at random. These are used to specify the part of the texture sample copied into each patch. The points must be chosen such that for all points x in or near the Voronoi cell of p_i the point $x - p_i + q_i$ lies within the input image. This is necessary to ensure that the portion of the texture sample copied to each patch lies within the texture sample's border.

Let $d((x_1, y_1), (x_2, y_2))$ be the Euclidean distance between two points. Let a be the average color of the input image. Let f be a parameter defining the sharpness of the edges of the patches (the effect of this parameter is illustrated in Figure 2). Assign each Voronoi cell a weight $w_i(x)$ at each point x in the output image

$$w_i(x) = d(x, p_i)^{-f} \tag{1}$$

Then the value of each pixel in the output image may be calculated using

$$O(x) = \frac{\sum_{i=1}^{n} (I(x - p_i + q_i) - a) w_i(x)}{\sqrt{\sum_{i=1}^{n} w_i(x)^2}} + a \tag{2}$$

In this equation, the divisor rescales the result so that its standard deviation is the same as that of the input image. Subtracting out and then adding

back the average a ensures that the output image has the same average color as the input. For efficiency, terms with small weight may be omitted from the summations.

3.1 Results

A test suite of images was chosen from the VisTex texture database [10]. These are shown in Figure 3. They include simple and complex textures, textures with elongated features, tiling patterns and several non-textures. The results of synthesis are shown in Figure 4. Many of the synthesized textures (S1, S2, C1, E1, E2, E3) do not contain any obvious artifacts and are not obviously synthetic. Best results were obtained where the texture did not have large structures, such as objects (C3), cracks (C2, C4), or tiling elements (T1, T2). Random choice of patches is therefore not sufficient for synthesis of all textures, but is practical for a large subset of textures.

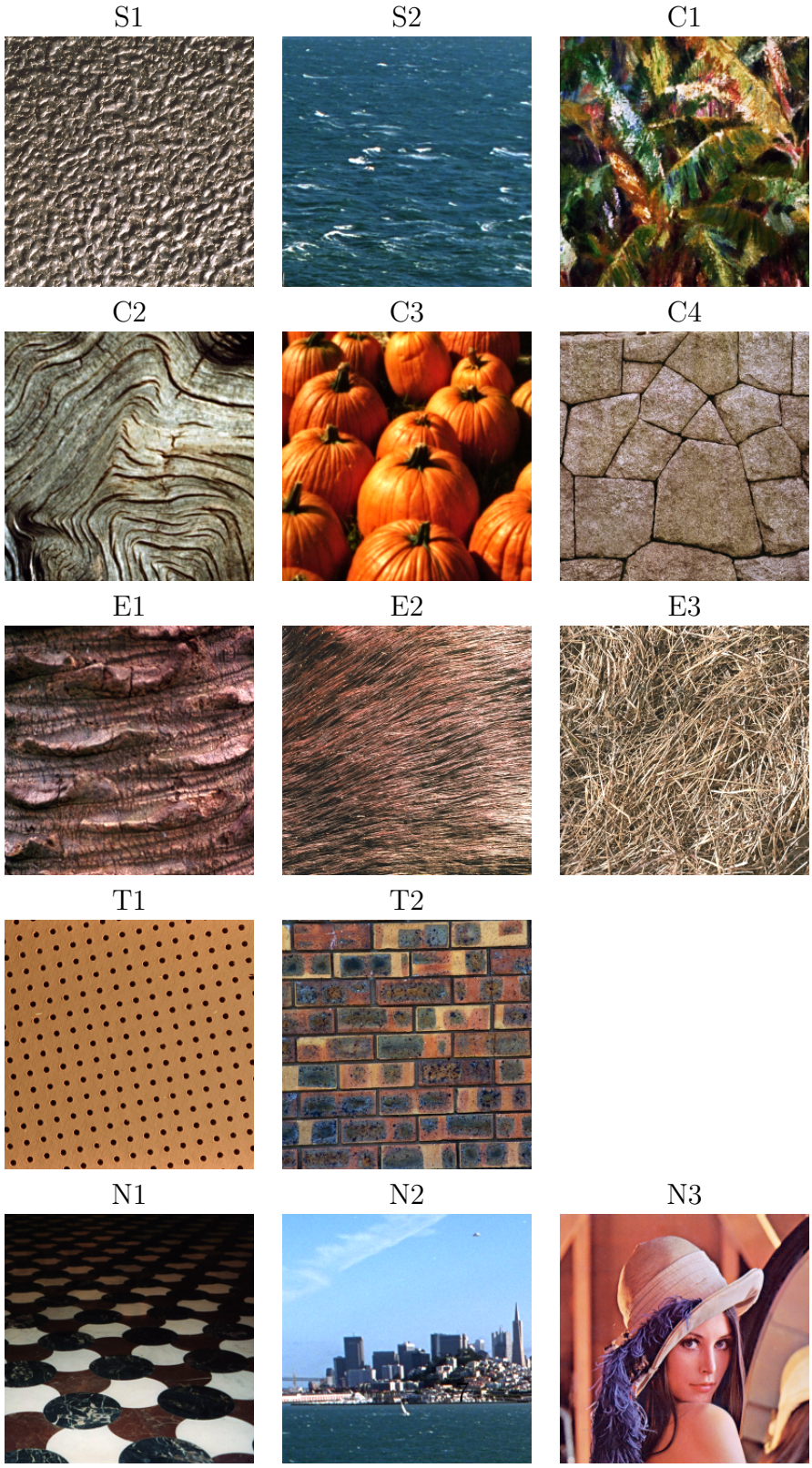


Figure 3: Test suite.

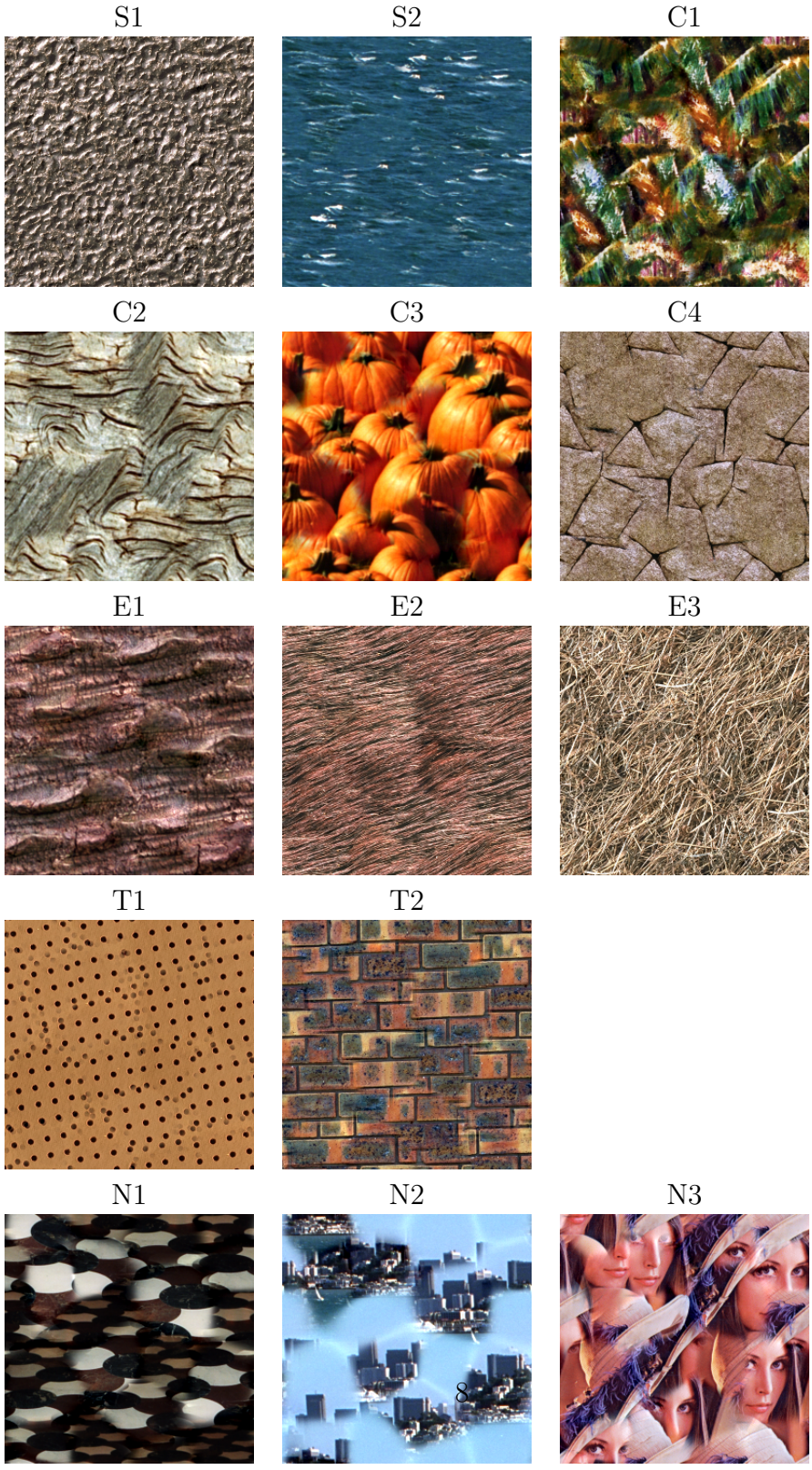
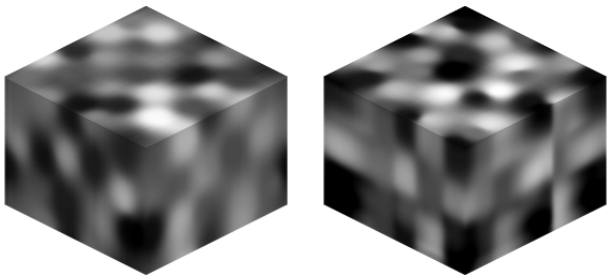


Figure 4: Results with $f = 6$, and a grid size of 64 pixels.



patch-method-1.mpg (a) patch-method-2.mpg (b)

Figure 5: Video cubes illustrating the cell shapes of two different video synthesis methods (the vertical axis is time). The videos are available on the CD-ROM accompanying this technical report.

4 Video texture synthesis

In this section, the texture synthesis method is extended to video. The obvious way to approach video texture synthesis is to treat the temporal dimension as just another spatial dimension, producing a three dimensional mosaic of patches. This is illustrated in Figure 5a. It has the drawback that any static elements in the video will become transient in the output. It is therefore proposed to treat the time dimension separately. Patches are fixed between frames, and a separate compositing step is performed in the time dimension, as illustrated in Figure 5b. This will preserve any static elements in the texture.

The video texture synthesis method consists of two stages. In the first stage, a set of video sequences are constructed from frames of the input video. In these intermediate sequences, no re-arrangement within each frame takes place, just a re-arrangement of the frames themselves. In the second stage, each frame of the output sequence is constructed from the corresponding frames of the intermediate sequences, using the image synthesis method described in the previous section.

To generate each intermediate sequence, n frames p_i are first chosen in the intermediate sequence, and n corresponding frames q_i are chosen in the input sequence. The intermediate sequence is generated as a series of transitions

between each frame p_i and its successor p_{i+1}.

First, define weights for a smooth transition between p_i and p_{i+1}

$$w_1 = \frac{1}{2} + \frac{1}{2}cos(\pi u) \quad (3)$$

$$w_2 = \frac{1}{2} - \frac{1}{2}cos(\pi u) \quad (4)$$

where u is the position within the interval $[p_i, p_{i+1}]$

$$u = \frac{t - p_i}{p_{i+1} - p_i} \quad (5)$$

Also calculate the average value of each pixel in the input sequence over time, forming an average image $A(x)$. Then the intermediate sequence $N(x,t)$ may be synthesized from the input sequence $I(x,t)$ within the interval using

$$N(x,t) = \frac{w_1 I(x, t - p_i + q_i) + w_2 I(x, t - p_{i+1} + q_{i+1}) - A(x)}{\sqrt{w_1^2 + w_2^2}} + A(x) \quad (6)$$

In this equation, each pixel in each intermediate frame is produced by combining pixels from two source sequences. The standard deviation of each pixel is preserved in the result.

Piecing all the intervals together gives a full intermediate sequence.

The output sequence is generated from a set of intermediate sequences. Each frame of the output is constructed by piecing together patches sampled from the corresponding frames of the intermediate sequences as in the image synthesis method. Each intermediate frame contributes one patch to the output frame. Each frame in the output is synthesized using the same arrangement of patches.

For efficiency the two stages of synthesis can be combined, so that the pixel values in the intermediate sequences are only ever calculated as needed.

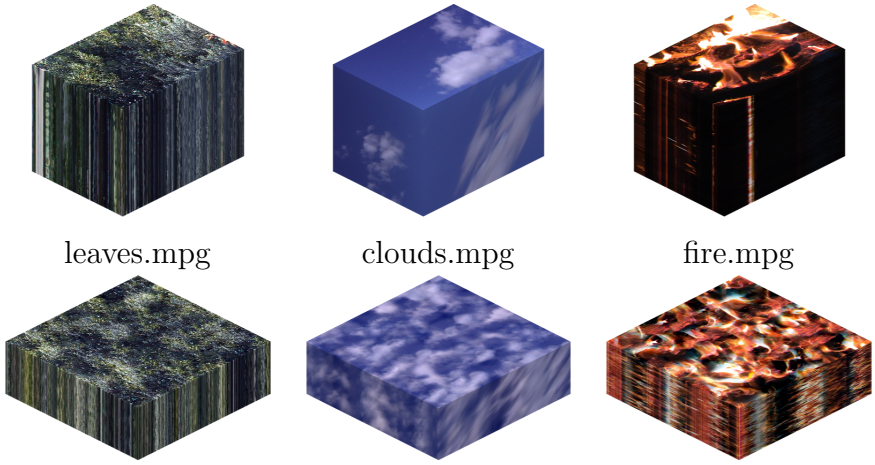


Figure 6: Input video cubes (top row) and corresponding synthetic video cubes (bottom row). The videos are available on the CD-ROM accompanying this technical report.

4.1 Results

Results are shown in Figure 6. All results are tilable video loops. The first video produced good results. That is, there are no obvious artifacts and the video is not obviously synthetic. The second was less satisfactory, the patches are clearly visible. This is because the movement of the clouds reveals the pattern of patches. The third video also produced reasonable results, although there are some artifacts where bright flame occured at the edge of a patch.

5 Texture transfer

Texture transfer is the process of giving an item such as an image a particular texture. For example, a photograph could be given the texture of a painting. This technique was first described by the author in the context of the best fit texture synthesis method [6], and later by Hertzmann et al. [8] and by Efros and Freeman [3]. In this section, the texture synthesis method described in

the previous sections will be extended to texture transfer. This will first be described in the domain of images, then generalized to video.

To perform texture transfer, two images are required. As before, a texture image is required ($I(x)$). Additionally a pattern image is necessary ($P(x)$). The pattern image defines the pattern of texture desired in the output. For example, using a painting as the image texture and a photograph as the pattern image a synthetic painting of the photograph could be created.

Patches are again defined in terms of a Voronoi diagram. However, the points q_i in the input texture are no longer chosen randomly. Instead, each is chosen to minimize the sum of squares difference of the patch of the pattern image around p_i and the patch of input texture around q_i.

Finding the exact best match to each patch often means that a particular area of the texture sample is repeated many times in the output. A solution to this problem is to consider only a random subset of points in the image as possibilities for each point q_i. This also yields a substantial speed increase.

The output to be synthesized is no longer homogeneous. The average pixel value varies from place to place. During synthesis this means that rather than using an overall average, a localized average must be used. One way to do this is to calculate an average a_i of each patch. The output can then be calculated from

$$O(x) = \frac{\sum_{i=1}^{n}(I(x - p_i + q_i) - a_i)w_i(x)}{\sqrt{\sum_{i=1}^{n} w_i(x)^2}} + \frac{\sum_{i=1}^{n} a_i w_i(x)}{\sum_{i=1}^{n} w_i(x)} \tag{7}$$

This is very similar to Equation 2, the only difference being the calculation of the average.

This transfer technique may also be applied to video. For example, to produce an image with animated texture, the pattern image $P(x)$ might be matched against the average frame $A(x)$. The video texture synthesis method described in section 4 can then be applied.

5.1 Results

Results are shown in Figures 8 and 9, based on the images in Figure 7. As a pre-processing step, the pattern images were rescaled so that they had the same average color and color range as the texture images. These images took longer to produce than the texture synthesis images, as selecting each patch required searching the texture image.

An example of video texture transfer is shown in Figure 10. The texure used was the video of leaves in Figure 6.

(a) 

(b) 

Figure 7: Pattern images.

Figure 8: Textures and corresponding results of texture transfer with $f = 2$, and a grid size of 16 pixels.

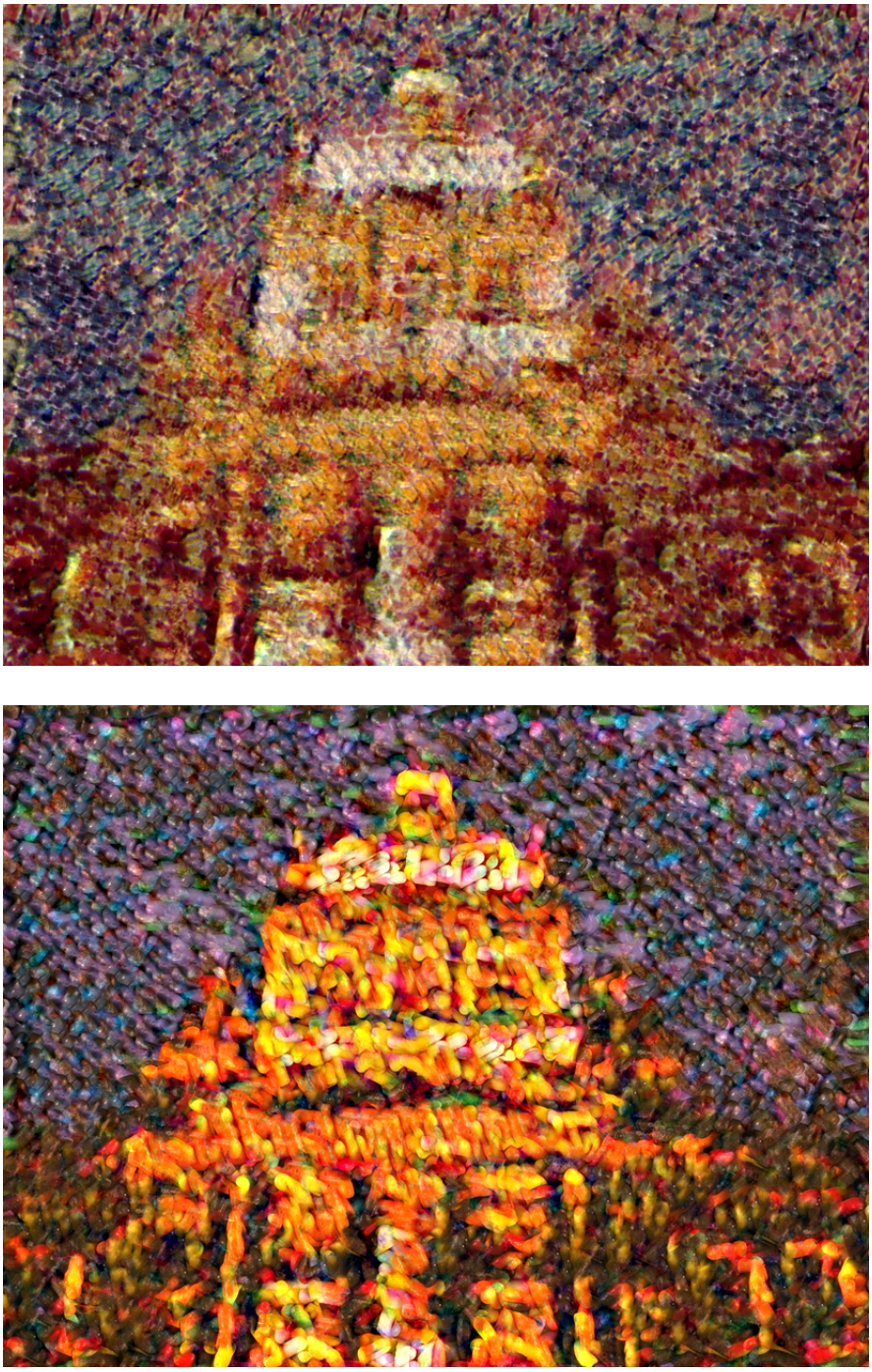
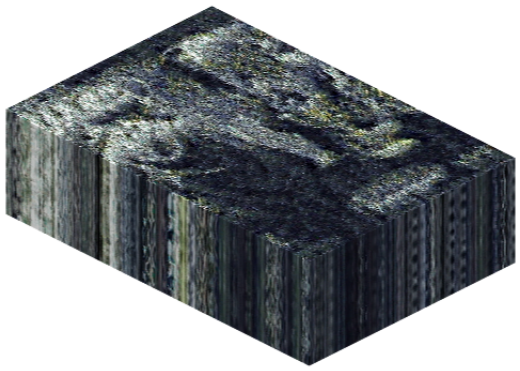


Figure 9: Additional texture transfer results.



tt.mpg

Figure 10: Example of video texture transfer.

6 Discussion

The main contribution of this technical report is the method of blending edges invisibly. The texture synthesis technique described in this technical report is simple and fast, and gives good results for many textures.

Could the technique be improved by choosing the patches more carfully, as in image quilting [3]? Interestingly, the answer is no. The method used to disguise edges relies on patches being uncorrelated. If patches are chosen so that they are correlated, the edges will be over-emphasized and become visible.

In the introduction it was noted that many texture synthesis methods copy sections of the input, either implicitly or explicitly. If this property is considered separately from the selection of the patches that are copied, their relative importance can be compared. In the technique described in this technical report, the selection of patches was random. Excellent results were obtained with some textures, but others produced poorer results. In particular, tiling patterns and textures with distinct objects produced poor results. It can therefore be concluded that best fit selection of patches can only improve textures such as these.

The complexity of texture synthesis using this technique is $O(n)$ in the num-

ber of pixels. As a consequence, it is practical to use the technique with high resolution digital photographs and digital movies.

7 Acknowledgements

I thank Dr. Alan Dorin, my PhD supervisor, for his support and his comments on drafts of this technical report. His input made this technical report considerably more readable.

References

[1] M. Ashikhmin. Synthesizing natural textures. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, pages 217–226. ACM Press, 2001.

[2] J. S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proceedings of SIGGRAPH 1997*, pages 361–368. ACM Press, 1997.

[3] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of SIGGRAPH 2001*, pages 341–346. ACM Press, 2001.

[4] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, pages 1033–1038. IEEE, 1999.

[5] D. Garber. *Computational Models for Texture Analysis and Texture Synthesis*. PhD thesis, University of Southern California, 1981.

[6] P. F. Harrison. A non-hierarchical procedure for re-synthesis of complex textures. In *WSCG'2001*, pages 190–197, Plzen, Czech Republic, 2001. University of West Bohemia.

[7] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of SIGGRAPH 1995*, pages 229–238. ACM Press, 1995.

[8] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proceedings of SIGGRAPH 2001*. ACM Press, 2001. Available: `http://mrl.nyu.edu/projects/image-analogies/` (Accessed: 2002, May 31).

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

[10] MIT Vision Texture database. Available: `http://www-white.media.mit.edu/vismod/imagery/VisionTexture/vistex.html` (Accessed: 2002, May 27).

[11] J. Portilla and E. P. Simoncelli. Texture modeling and synthesis using joing statistics of complex wavelet coefficients. In *Proceedings of the IEEE Workshop on Statistical and Computational Theories of Vision*, 1999. Available: `http://www.cis.ohio-state.edu/~szhu/SCTV99.html` (Accessed: 2000, September 7).

[12] L. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of SIGGRAPH 2000*. ACM Press, 2000. Available: `http://graphics.stanford.edu/projects/texture/` (Accessed: 2000, September 7).

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