TRANSPARENT TEXTURE TRANSFER

Chan-Tai Yeh, Ting-Hui Tsai, Hwann-Tzong Chen

Department of Computer Science, National Tsing Hua University, Taiwan.

ABSTRACT

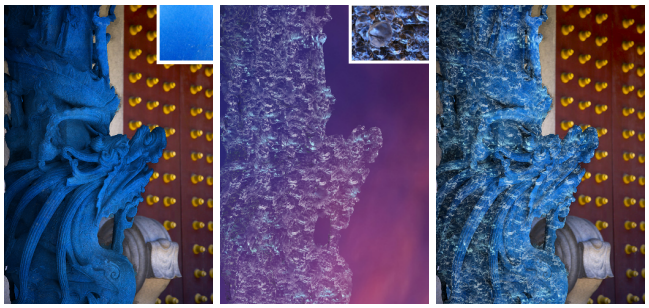
This paper presents a toolchain that can transfer transparent, multi-layer textures onto some unknown surface in an image. We propose *alpha quilting*, which is able to generate a large texture map and its corresponding alpha matte from a small template of natural texture. The synthesized texture map can then be transferred to a surface according to the estimated surface normal. The procedure of texture transfer is called *texture matting*, which layers multiple textures onto a surface with respect to the transparency and multi-layer characteristics of natural textures. The experimental results show that our method is able to produce visually plausible results of pasting opaque and transparent textures on surfaces.

Index Terms— texture transfer, texture matting

1. INTRODUCTION

Previous approaches to image-based texture synthesis often treat a texture as a single-layer image consisting of a regular pattern or some less regular but repeated image features [2, 5, 9, 12]. Algorithms for image-based texture synthesis generally synthesize the textures regardless of their intrinsic properties, except that certain parameters in the algorithms might depend on the scale of texture. However, many natural textures, such as flames or clouds, are not just purely flat and opaque. They may comprise layers of substances and elements, and often have a transparent appearance. In this work, we try to take into consideration the layered nature and the transparent property of textures, and by doing so we are able to deal with more complex textures that cannot be well handled by previous texture synthesis algorithms. We develop a tool for transferring multi-layer natural textures onto an unknown surface shown in an image. The process of transferring textures can be divided into three parts: First, we extract the alpha matte from a small template of natural texture, and synthesize a large foreground-texture map using the small templates of texture and its corresponding alpha matte. Second, we recover the surface normals from the input image. Third, we transfer the large texture map onto the surface according to the recovered surface normals.

To produce visually plausible results of texture transfer, we use *texture matting* to take into account the transparency of the synthesized texture map. By texture matting we mean that the sample textures used for synthesis are not necessarily opaque but may be associated with some alpha mattes that define the opacity (or the transparency) of the textures. The proposed technique is able to synthesize a new foreground-texture map and its corresponding alpha matte. The texture can then be transferred and layered onto the target surface to produce plausible blending results based on the alpha matte. Fig. 1 illustrates an example of two-layer texture matting using the opaque texture Glacier and transparent texture Ice. Our texture matting method is able to produce a visually appealing synthetic image with shading and transparent effects.



(a) (b) (c)

Fig. 1. (a) Texture transfer using opaque texture Glacier. (b) Alpha quilting using transparent texture Ice. Note that a random background is overlaid with the alpha quilting result for visualization. (c) An example of two-layer texture matting.

1.1. A Concise Review of Related Work

Efros and Leung [5] introduce Markov random fields (MRFs) and non-parametric sampling to the modeling of textures. Wei and Levoy [12] also use an MRF model but present a more efficient multi-scale algorithm to predict the states of pixels. They use nearest-neighbor search to find the best match for predicting the pixel states instead of estimating the states by non-parametric sampling. Ashikhmin [2] presents an algorithm that is capable of synthesizing structural textures by constraining the candidates in nearest-neighbor search. Hertzmann *et al*. [8] combine the previous two algorithms [2, 12], and introduce the idea of *image analogies* for various applications of texture synthesis and image

processing. Another way to preserve the texture structure and reduce the computational cost is to synthesize a large texture 'patchwise' rather than 'pixelwise'. The *image quilting* algorithm [4] follows a patch-based synthesis scheme and attempts to find a minimum-cost boundary to stitch two neighboring patches. Kwatra *et al*. [9] also use a patch-based procedure and perform graph cuts to find optimal seams for merging patches. Guo *et al*. [6] present a method for two-layer texture synthesis. They model the similar elements (textons) in a texture image, and learn a generative model for synthesizing new textures. They only consider either "zero" or "full" transparency in the template texture, for indicating that a pixel in the texture belongs to foreground or background. Regarding transferring textures onto a surface, *e.g*. [11], we adopt the approach proposed in [7] for estimating surface normals. Surface normals are clustered into segments, and the textures can be mapped onto the surface according to the normal vector at the center of a segment.

2. TEXTURE TRANSFER WITH ALPHA MATTES

The main problem to be addressed in the texture transfer application is how to paste a template of natural texture onto an unknown surface. Consider we have two images that are given as the input: The first image is a (not-fully-opaque) sample texture that we try to use for texture synthesis. The second image consists of an unknown surface and we are interested in transferring the sample texture onto it. We assume that the user provides the information for deriving the alpha matte from the sample texture, in a form such as scribbles [10] or trimaps (foreground, background, and unknown areas [3]). The silhouette of the target surface is also assumed to be specified by the user. Given the two input images and the user-provided information, we want to solve the problem of layering the natural texture onto the surface, subject to the transparency of the texture and the curvature of the surface. Our aim is to create visually plausible results of texture transfer, where the output images exhibit realistic transparency of the textures, and the multi-layer characteristics are vividly simulated on the target surface.

2.1. Alpha Quilting: Synthesizing Textures and Alpha Mattes

Efros and Freeman [4] propose *image quilting* for patch-based texture synthesis. To improve the effectiveness of texture synthesis and texture transfer for multi-layer textures, we propose a new algorithm called *alpha quilting*, which is able to produce a large foreground-texture map and its corresponding alpha matte. The main idea of our approach is to synthesize textures with side information derived from the alpha values. The input to the alpha quilting algorithm comprises a sample foreground-texture and its alpha matte. The foreground-texture and the alpha matte are obtained by applying the alpha estimation technique of [10]. Our alpha quilting algorithm goes thorough the texture map to be synthesized in raster-scan order, and synthesizes the texture map 'patch by patch' with overlaps. For every location, we have to search the sample foreground-texture for a set of candidate patches that satisfy the overlap constraints based on some error tolerance. To account for the transparency property, the alpha quilting algorithm uses both the color and the alpha values in the criteria of selecting candidate patches.

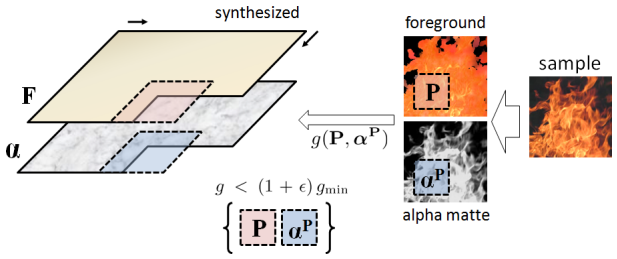


Fig. 2. The alpha quilting algorithm proceeds with a patch-based synthesis scheme. We go thorough the to-be-synthesized texture map in raster-scan order, and synthesize the texture map 'patch by patch' with some overlaps. Given a patch $\mathbf{P}$ from the sample foreground-texture and its alpha mattes $\boldsymbol{\alpha}^{\mathbf{P}}$, we compute the error function $g(\mathbf{P}, \boldsymbol{\alpha}^{\mathbf{P}})$ with respect to the overlapping region at the current location. The overlapping region is an L-shape area, and the patch overlaps at the left and the above with the previously synthesized foreground-texture $\mathbf{F}$. We select a set of candidate patches whose error g satisfies $g < (1+\epsilon)\, g_{\min}$ for some $\epsilon > 0$, where $g_{\min}$ is the minimum value over all patches.

2.1.1. *Selecting Candidate Patches*

The procedure of selecting candidate patches is illustrated in Fig. 2. Given a patch $\mathbf{P}$ in the sample foreground-texture and its corresponding alpha matte $\boldsymbol{\alpha}^{\mathbf{P}}$, we define the error function g as a combination of two terms:

$$g(\mathbf{P}, \boldsymbol{\alpha}^{\mathbf{P}}) = \sum_{\{i,i'\}\in\Omega} \|\alpha_i F_i - \alpha^{\mathbf{P}}_{i'} P_{i'}\|^2 + \lambda \left(1 - \rho(\hat{h}, h(\boldsymbol{\alpha}^{\mathbf{P}}))\right), \tag{1}$$

where F_i is a pixel of the previously synthesized foreground-texture inside the overlapping region Ω, and $P_{i'}$ is a pixel in $\mathbf{P}$. The alpha values of F_i and $P_{i'}$ are α_i and $\alpha^{\mathbf{P}}_{i'}$. The function ρ computes the Bhattacharyya coefficient between two distributions $\hat{h}$ and $h(\boldsymbol{\alpha}^{\mathbf{P}})$, which we will discuss later. The patch's upper and left parts are overlapped with the previously synthesized foreground-texture to form an L-shape overlapping region Ω. The index pairs $\{i, i'\} \in \Omega$ denote an index i to the overlapping pixel in the previously synthesized foreground-texture and an associated index i' to the corresponding pixel in the patch.

The first term in Eq. (1) measures the dissimilarity in appearance between the patch and the synthesized foreground-texture, by taking into account the alpha values. This term tends to favor patches of small alpha values, and therefore we have to add a regularization term to prevent from choosing patches that contain only a small portion of the foreground. The prior for an alpha matte can be modeled by beta distributions, as shown in [1]. In this work, we directly estimate the distribution $\hat{h}$ of alpha values in the sample texture, and expect that the synthesized foreground-texture would exhibit a similar distribution of alpha values (denoted by $h(\boldsymbol{\alpha}^{\mathbf{P}})$). The dissimilarity between the two distributions is measured by $1-\rho(\hat{h}, h(\boldsymbol{\alpha}^{\mathbf{P}}))$, where ρ is the Bhattacharyya coefficient

$$\rho(\hat{h}, h(\boldsymbol{\alpha}^{\mathbf{P}})) = \int \sqrt{\hat{h}\, h(\boldsymbol{\alpha}^{\mathbf{P}})}\, d\alpha\,. \tag{2}$$

The value of ρ is within $[0,1]$, and it is equal to 1 if the two distributions are identical. In practice, we use a histogram of 100 bins to approximate the distribution of alpha values for the computation of the Bhattacharyya coefficient.

Some patches might have very distinctive appearance, and so the differences between these patches and others would be quite significant. If such patches happen to be chosen, it is very likely that only a small number of suitable candidates will be included in the next iteration. Therefore, instead of picking the best patch for every location, we search the sample foreground-texture for a set of candidate patches whose values of g in Eq. (1) are within error tolerance $(1+\epsilon)\, g_{\min}$ for some $\epsilon > 0$, where $g_{\min}$ is the minimum value over all patches. We use ϵ to expand the range of candidates, and in our implementation we set $\epsilon = 0.02$.

2.1.2. *Minimum Error Boundary*

For each patch belonging to the candidate set, we use dynamic programming to find the minimum-error seam inside the overlapping region. We discuss below the case of computing vertical seam, and the case of finding horizontal seam can be solved similarly. For finding the vertical seam, we consider only the difference in the appearance of the foreground. Here we change the notation for the subscripts. We use two-dimensional indexes for the sake of explanation. The error surface at a location (l,m) with respect to a candidate patch $\mathbf{P}$ and its alpha matte $\boldsymbol{\alpha}$ is defined by

$$e_{l,m}(\mathbf{P}, \boldsymbol{\alpha}^{\mathbf{P}}) = \|\alpha_{l,m} F_{l,m} - \alpha^{\mathbf{P}}_{l',m'} P_{l',m'}\|^2\,, \tag{3}$$

where $F_{l,m}$ is the previously synthesized foreground-texture at location (l,m), and the indexes (l',m') denote the corresponding location in $\mathbf{P}$. To find the vertical minimum-error seam through this error surface, we traverse e, from top to bottom, and compute the minimum cumulative error map E for all seams using dynamic programming

$$E_{l,m} = e_{l,m} + \min\{E_{l-1,m-1}, E_{l,m-1}, E_{l+1,m-1}\}\,. \tag{4}$$

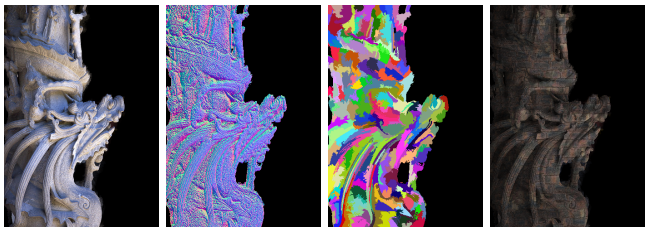
The minimum cumulative value of the last row of E represents the total error of the seam, and the corresponding path that starts from the first row of e to an endpoint at the bottom row indicates the minimum-error vertical seam. A similar procedure is applied to horizontal overlaps.

2.1.3. *Choosing the Patch by Factored Sampling*

For each patch in the candidate set, we compute a probability based on its minimum cumulative error:

$$p(\mathbf{P}, \boldsymbol{\alpha}^{\mathbf{P}}) \propto \exp\{-\eta\, E(\mathbf{P}, \boldsymbol{\alpha}^{\mathbf{P}})\}\,. \tag{5}$$

We sample a patch from the above probability distribution of candidate patches, and stitch the selected patch into the synthesized foreground-texture based on the vertical and horizontal minimum-error seams. We also merge the alpha matte of the selected patch into the existing alpha matte according to the seams.



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

Fig. 3. (a) The input image with the background being cut out. (b) Estimating the normal vector of each pixel on the object surface. (The x, y, and z components of a normal vector are respectively converted to R, G, and B colors for displaying.) (c) Grouping adjacent pixels of similar normal-vectors using k-means. (e) Transferring a new texture onto the reconstructed surface according to the approximate surface normals.

3. SURFACE NORMAL RECONSTRUCTION FOR TEXTURE MATTING

The results of alpha quilting will be layered on a surface, which is reconstructed from a static image using normal-vector recovery. Many approaches to surface normal recovery have been proposed in the literature of graphics and machine vision. Our attempt is to reconstruct the surface of a specified object from a static image, and to apply texture transfer and matting to the surface. We do not need to recover the exact 3D information about the surface. We only need the approximate surface normals that are good enough to account for surface-based texture transfer. The simple surface-normal recovering technique presented in [7] is employed for our application. After estimating the normal vector of each pixel on the surface, we may group adjacent pixels of similar normal

(a) (b) (c) (d) (e) (f)

Fig. 4. (a) The original image. (b) Texture transfer using opaque texture Bricks. (c) Alpha Quilting of Erosion. (A random background is added for visualization.) (d) Erosion + Bricks. (e) Alpha Quilting of Clouds. (f) Clouds + Erosion + Bricks.

vectors to form clusters of surface normals. In our implementation, we use the k-means algorithm to do clustering. We create for each pixel a 5D data point that comprises the three components of the normal vector and the x and y coordinates of the pixel, and then apply k-means to cluster the 5D data points. The final step is to layer the texture patches on the surface based on the clusters of surface normals. To make the appearance look more natural, each texture patch must be deformed to match the orientation at its target location on the surface, using the technique described in [7]. The procedure is shown in Fig. 3.

After the above procedure, the texture image would seem to be pasted along the surface of the object, but the luminance information is lost in the result. We compute the inner product between the normal vector of each point and the estimated light source vector $\hat{S}$ to recover the luminance. We multiply the recovered luminance to the tiled textures to get the output of texture transfer.

We can further combine the results of texture transfer to produce multi-layer effects. Suppose that texture $\mathbf{T}_1$ contains a transparent foreground and texture $\mathbf{T}_2$ is a previously synthesized texture-transfer result. We may then take $\mathbf{T}_1$ as foreground and $\mathbf{T}_2$ as background, and use the alpha values of $\mathbf{T}_1$ to combine $\mathbf{T}_1$ and $\mathbf{T}_2$.

3.1. Implementation Details

The light source is estimated by least squares, but if we know the correct direction of the light source beforehand, we can directly apply its direction and skip the step of light source estimation. Furthermore, the estimation error on the light source and on the normal vectors might sometimes lead to unnatural approximations. To prevent negative values of the Z component of the normal vectors, we require that the value of Z component should not be lower than a threshold ($= 0.1$). We rectify the Z component values according to the threshold, and then re-normalize the vectors to ensure that every surface normal remains a unit vector.

4. EXPERIMENTAL RESULTS

This section shows the experimental results of alpha quilting and texture matting. We present the results of pasting multi-layer textures onto the estimated surface of a stone sculpture. In our experiments we use two samples of opaque textures (Bricks and Glacier) and three samples of transparent textures (Erosion, Cloud, and Ice). The synthesis results of transparent textures and opaque textures are combined to produce the multi-layer texture effects.

Fig. 4 depicts the results of three-layer texture matting using varied texture samples. At the bottom layer we use an opaque texture Bricks, and at the upper two layers we use two different transparent textures, Erosion and Cloud. Based on the alpha mattes of textures produced by alpha quilting, we are able to create various combinations of matting effects, and the synthesis images exhibit the transparency and multi-layer characteristics of the textures. Fig. 1 illustrates another example of two-layer texture matting using the opaque texture Glacier and transparent texture Ice.

5. CONCLUSIONS

The goal of this paper is to improve the effectiveness of texture synthesis in reproducing multi-layer characteristics of opaque and transparent textures on a surface. The alpha matte of texture provides a useful cue for merging and modulating the appearance of overlapping region within the two neighboring patches. We have shown that the proposed alpha quilting and texture matting techniques perform well in creating interesting visual effects. A possible extension of this work is to enable the textures to "grow" outside the surface boundary in the image.

Acknowledgment. This work was supported in part by NSC grant 101-2628-E-007-020-MY3 and MOST grant 103-2221-E-007-045-MY3.

6. REFERENCES

[1] N. Apostoloff and A. W. Fitzgibbon. Bayesian video matting using learnt image priors. In *CVPR (1)*, pages 407–414, 2004.

[2] M. Ashikhmin. Synthesizing natural textures. In *The proceedings of 2001 ACM Symposium on Interactive 3D Graphics*, pages 217–226, 2001.

[3] Y.-Y. Chuang, B. Curless, D. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *CVPR (2)*, pages 264–271, 2001.

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

[5] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *ICCV*, pages 1033–1038, 1999.

[6] C. en Guo, S. C. Zhu, and Y. N. Wu. Modeling visual patterns by integrating descriptive and generative methods. *International Journal of Computer Vision*, 53(1):5–29, 2003.

[7] H. Fang and J. C. Hart. Textureshop: texture synthesis as a photograph editing tool. *ACM Trans. Graph.*, 23(3):354–359, 2004.

[8] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. Salesin. Image analogies. In *SIGGRAPH*, pages 327–340, 2001.

[9] V. Kwatra, A. Schödl, I. A. Essa, G. Turk, and A. F. Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Trans. Graph.*, 22(3):277–286, 2003.

[10] A. Levin, D. Lischinski, and Y. Weiss. A closed form solution to natural image matting. In *CVPR (1)*, pages 61–68, 2006.

[11] G. Turk. Texture synthesis on surfaces. In *SIGGRAPH*, pages 347–354, 2001.

[12] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH*, pages 479–488, 2000.