EUROGRAPHICS 2017 / L. Barthe and B. Benes
(Guest Editors)

Volume 36 (2017), Number 2

Analysis and Controlled Synthesis of Inhomogeneous Textures

Yang Zhou[1,2] Huajie Shi[2] Dani Lischinski[3] Minglun Gong[4] Johannes Kopf[5] Hui Huang[1,2†]

[1]Shenzhen University [2]SIAT [3]The Hebrew University of Jerusalem [4]Memorial University of Newfoundland [5]Facebook

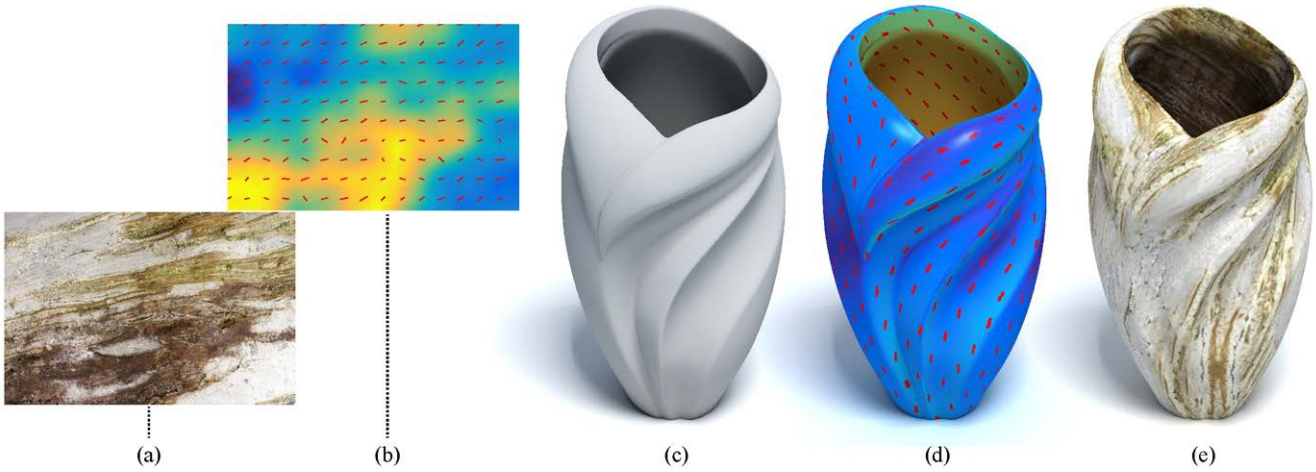


Figure 1: *Given an inhomogeneous and anisotropic texture exemplar (a), we automatically extract the corresponding* source guidance map *(b), comprising a scalar progression channel (rendered in pseudo-color) and a direction field (overlayed in red). A* target guidance map *(d) may then be used to synthesize a texture for a specific 3D object (c). The final texture-mapped result is shown in (e).*

Abstract

Many interesting real-world textures are inhomogeneous and/or anisotropic. An inhomogeneous texture is one where various visual properties exhibit significant changes across the texture's spatial domain. Examples include perceptible changes in surface color, lighting, local texture pattern and/or its apparent scale, and weathering effects, which may vary abruptly, or in a continuous fashion. An anisotropic texture is one where the local patterns exhibit a preferred orientation, which also may vary across the spatial domain. While many example-based texture synthesis methods can be highly effective when synthesizing uniform (stationary) isotropic textures, synthesizing highly non-uniform textures, or ones with spatially varying orientation, is a considerably more challenging task, which so far has remained underexplored. In this paper, we propose a new method for automatic analysis and controlled synthesis of such textures. Given an input texture exemplar, our method generates a source guidance map comprising: (i) a scalar progression channel that attempts to capture the low frequency spatial changes in color, lighting, and local pattern combined, and (ii) a direction field that captures the local dominant orientation of the texture. Having augmented the texture exemplar with this guidance map, users can exercise better control over the synthesized result by providing easily specified target guidance maps, which are used to constrain the synthesis process.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—color, shading, shadowing, and texture I.4.7 [Image Processing and Computer Vision]: Feature Measurement—texture

1. Introduction

During the past two decades, there has been tremendous progress in example-based texture synthesis methods [WLKT09]. Most of these methods inherently assume that the input exemplar textures

† Corresponding author: Hui Huang (hhzhiyan@gmail.com)

Figure 2: *Inhomogeneous and spatially variant directional textures (left in each pair) do not lend themselves well to uncontrolled synthesis. Lack of control over the progression or orientation across the texture yields synthesis results that do not look natural (right in each pair). These results were synthesized using self-tuning texture optimization [KNL*15].*

are homogeneous (stationary). However, many surfaces in the real world are, in fact, inhomogeneous, and contain some form of spatial variation that manifests itself in perceptible changes in color, lighting, pattern, and the size and orientation of texture elements, which may gradually evolve across the spatial extent of the exemplar. We henceforth refer to such spatially variant behaviors as *progressions*. Most existing synthesis methods operate in a local fashion and are therefore not well equipped to *automatically* handle these more global phenomena, as demonstrated in Figure 2.

Furthermore, texture artists, the intended users of texture synthesis, are rarely interested in merely synthesizing a larger texture from an exemplar; rather, their typical goal is to produce textures intended for specific 3D models, which requires good control over the synthesized result. However, example-based texture synthesis methods often lack simple and intuitive means of user control. In particular, little attention has been paid over the years to controlling the synthesis of inhomogeneous textures.

A common way to add control to existing methods is by providing manually created *feature* or *guidance channels* both for the exemplar *and* the output texture [HJO*01]. The guidance channels dictate, in a "texture-by-numbers" style, where specific content from the exemplar should be placed in the output texture. The manual nature of this workflow is tedious, especially when attempting to annotate continuous progressions. The recent self tuning texture optimization approach [KNL*15] automatically computes a guidance channel, however, it is designed to help preserve large scale structures, such as curve-like features, rather than to control gradual spatial changes.

In this work we present a new approach for automatic analysis and controlled synthesis for general inhomogeneous and directional textures. Ideally, we would have liked to analyze spatial progressions in a semantic fashion (i.e., detect changes in lighting, reflectance, material, weathering degree, etc.). However, such a targeted analysis can be extremely difficult to accomplish in practice, as well as limited to specific phenomena. Instead, we automatically extract from a texture exemplar a *guidance map* that consists of two independent components: (i) a scalar *progression channel*, which captures low-frequency spatial changes in color, lighting, and local pattern combined, and (ii) a *direction field* that captures the dominant local orientation (see Figure 1(b)). Our reasoning here is that the orientation is often less correlated with the other changes, and can be reliably factored out. Augmenting the texture with this source guidance map makes it possible to exercise spatial control over the synthesized result via automatically generated or user-supplied *target guidance maps*.

Specifically, we employ manifold learning (using the Isomap method [TdSL00]) to perform non-linear dimensionality reduction of high-dimensional local texture descriptors in order to obtain the scalar progression channel. In addition, we use histograms of oriented gradients (HoG) descriptors [DT05] in order to extract a vector field describing the local dominant orientation.

Our automatically extracted source guidance maps are used to make example-based texture synthesis more controllable, by augmenting the exemplar's color channels. To this end, we use the texture optimization framework [KEBK05, KNL*15] with a suitably modified patch distance metric. Our implementation builds upon the available code of self tuning texture optimization [KNL*15], but other algorithms could be adapted as well.

While manifold learning has been used before for analyzing time-varying appearance of materials [WTL*06, XWT*08], these approaches were restricted to continuous changes in reflectance, and did not capture other texture variations. We demonstrate that our use of high-dimensional texture descriptors coupled with Isomap dimensionality reduction is able to capture more general variations, and yields superior results.

There has also been previous work on analyzing the local orientation and using the result to control the synthesis [FM02, LFA*15]. Our approach is novel in that it enables controlling both spatial progression and local orientation across the synthesized texture, both individually, and jointly. To our knowledge, this is the first example-based synthesis method to offer such capabilities.

Precisely specifying the target guidance channels can be a tedious task in itself. It is much easier for the user to provide simple, smooth guidance channels (Figure 4, middle), but they may yield artificially looking results, and tend to cause repetition artifacts. We present a new method for refining such smooth target guidance channels to make them more compatible with the exemplar (Figure 4, right). The refinement is done by injecting procedurally generated noise, adapted to match certain statistics extracted from the source guidance map. This typically leads to more naturally looking synthesis results. In addition, we demonstrate that the target guidance channels can also be automatically generated based on the geometry of the target 3D model.

In summary, our main contributions include:

- automatically augmenting the input exemplar with a guidance map describing dominant low-frequency changes of various visual properties, as well as the local orientation across the texture;
- independent and simultaneous control of spatially variant visual attributes and of local orientation;
- refinement of smooth target guidance channels for more natural synthesis results;
- adapting the texture optimization framework to enable controlled synthesis via user-provided or automatically generated target guidance channels.

We show numerous results and several applications for which our approach is useful throughout the paper. These include dominant texture synthesis, generation of time variant weathering sequences, and 3D model targeted synthesis. Some of these applications have been the subject of entire research papers in their own right, and the effectiveness of our approach for these applications is a testament of its usefulness and versatility.

2. Related work

Example-based texture synthesis has been extensively researched for over twenty years, and we refer the reader to Wei et al. [WLKT09] for a comprehensive survey. The relevant class of example-based synthesis methods are non-parametric methods, which include pixel-based methods [EL99, WL00], stitching-based methods [EF01, KSE*03, LL12], optimization-based methods [KEBK05, HZW*06, WSI07], and appearance-space texture synthesis [LH06]. Image melding [DSB*12] unifies and generalizes patch-based synthesis and texture optimization, leveraging PatchMatch, a fast randomized patch search algorithm [BSFG09].

Kaspar et al. [KNL*15] describe a self-tuning texture optimization approach, which uses image melding with automatically generated and weighted guidance channels. These guidance channels are designed to help reproduce the large-scale structures present in the texture exemplar. They are defined as the distance transform from a set of automatically extracted large scale salient contours, and offer no attempt to analyze or control spatial changes in appearance or orientation. In this work, we augment Kaspar's method with automatically generated progression and orientation guidance channels, and use an automatically refined target guidance channel as a soft constraint that helps control the synthesis result.

While non-parametric methods are typically able to reproduce small scale structure, they assume a stationary Markov Random Field (MRF) model, which makes it difficult for them to cope with highly inhomogeneous textures, which violate this assumption. In order to control large scale structure, Ashikhmin [Ash01] proposed to guide the synthesis process by a user-provided target image, which specifies the local average colors across the target texture. *Texture-by-Numbers* [HJO*01] extends this idea further by augmenting the input exemplar with a label map, where regions with distinct texture are distinguished by different labels. A suitable label map may be painted manually by the user, or created automatically using unsupervised image segmentation. To synthesize a new image, a target label map is provided, which indicates how the different textures should be arranged in the resulting image. However, that work addressed neither the issue of automatically generating a continuous label map for progressive textures, nor the automatic refinement of the target control map, as we do in this work.

Many other subsequent works made use of control maps when synthesizing non-stationary textures, for example [ZZV*03, WTL*06, GTR*06, LGG*07, WHZ*08]. However, in these works the control map for the input texture exemplar is either obtained from user input [ZZV*03], or derived from a set of measurements from static [WTL*06] or time-varying [GTR*06] material samples. Lu et al. [LGG*07] compute their control maps from a specific model of texture formation across a 3D surface. Wu et al. [WWY14] use the level set method to separate art patterns and textures with curvilinear feature into layers. In contrast to these methods, we analyze an input texture exemplar to yield several guidance channels completely automatically and without assuming any specific model, and propose a way to refine the target guidance channels for improved synthesis.

Rosenberger et al. [RCOL09] model inhomogeneous textures as a stacking of several layers, resulting in a discrete label map, and propose an example-based approach to synthesize realistic boundaries for these layers for the target control map. Lockerman et al. [LSA*16] propose an automatic hierarchical segmentation method for textures, also yielding a discrete label map. Our analysis differs from a discrete segmentation, since it results in continuous progression and orientation maps. In a discrete segmentation, the boundaries between segments are not always accurate, especially in textures that exhibit gradual changes, and the texture may still exhibit considerable changes inside each segment. In Section 5, we compare our continuous guidance maps to the discrete label maps generated by [RCOL09] and [LSA*16], and demonstrate that using continuous maps enables provides better control over the synthesis.

Wang et al. [WTL*06] model the time-variant surface appearance from a flat sample of partially weathered material, across which they capture the spatially-variant 7D BRDF. A neighborhood graph (appearance manifold) is then constructed, by connecting samples with similar reflectance vectors. The graph is constructed similarly to Tenenbaum et al. [TdSL00], but no Isomap dimensionality reduction takes place. Instead, they infer the weathering degree of each point on the sample from its geodesic distance in the graph to the least and most weathered locations, which are user specified. This work was extended to editing and manipulating weathering effects in a regular image by Xue et al. [XWT*08], where the manifold is constructed using the CIE Lab color of each pixel. Both when using 7D BRDFs, and when using colors, the resulting weathering degree maps are limited to smooth temporal variations in reflectance, and both papers admit that they cannot handle well other changes in texture, such as peeling and cracks.

In contrast, our input is an ordinary RGB texture, rather than BRDFs. We do not assume that the inhomogeneity results from any particular process, nor do we require any user input. Instead we rely on high-dimensional texture descriptors which have proven themselves in previous texture analysis tasks, and use the Isomap method to project the samples onto a one-dimensional manifold. In Section 5 we demonstrate that these design choices result in significantly more effective progression maps, compared to [XWT*08].

Lu et al. [LDR09] describe a method for detecting dominant texture samples based on a manifold generated using the diffusion distance. This enables them to synthesize uniform textures from "contaminated" exemplars. However, they report that textures with smooth transitions pose a challenge to their approach. While we also assume that texture features are located on a low-dimensional manifold in a high-dimensional feature space, our method is designed to cope with such progressive textures, and enables their controlled synthesis. As we demonstrate in Section 5.1, our approach subsumes [LDR09], since it can also be used to identify dominant texture regions and synthesize from them (see Figure 11).

A variety of methods have been proposed specifically for synthesis of weathering phenomena by assuming and simulating a physical model [DH96, DEJ*99, MDG01, BPMG04, DGA04, DRS08]. While such methods have produced some highly realistic results, they are not geared towards matching a particular appearance given by an example. Also, controlling the results of the synthesis typically involves specifying a large number of parameters, which are not always intuitive. In contrast, our approach is example-based, and involves an easy to specify control map.

In a recent work, Bellini et al. [BKCO16] propose a method for estimating an "age map" for a partially weathered texture, which enables synthesizing a sequence mimicking its time-varying evolution process. Their approach is based on the assumption that the unweathered portion of the texture is stationary, consisting of repetitive patterns. Iizuka et al. [IEKM16] derive a similar "weathering degree" map from a small amount of user input, and use it to simulate aging of surfaces in images. The progression maps extracted by our approach do not rely on such an assumption, nor do they require user input. Nevertheless, we demonstrate that they can effectively play the role of age maps, also enabling the synthesis of time-varying weathering sequences (Section 5.1).

Many previous works discussed ways for extracting the local orientation across an image. In particular, Feng et al. [FM02] perform a PCA analysis of multiscale gradients at each local neighborhood, similarly to our analysis of HoG features. However, the orientation detection method in itself is not the focus of our work. Lukáč et al. [LFA*15] also use the results of direction analysis in order to enable interactive painting with directional textures. The desired local texture direction is added as a hard constraint into the texture optimization framework: during synthesis source patches are simply rotated exactly by the difference between their original orientation and the target field orientation. Their results indeed exhibit good control over the direction, but the input exemplars they use are mostly homogeneous otherwise. In contrast, our approach introduces a soft constraint on the orientation, attempting to balance between the target scalar guidance map and the target direction field. Our contribution lies in introducing the ability to simulatenously control spatial inhomogeneity and spatially variant orientation. We compare our results to those of Lukáč et al. [LFA*15] in Section 5.

3. Texture Analysis

As explained earlier, in order to support controllable synthesis of inhomogeneous textures, we first automatically analyze the spatial inhomogeneity and anisotropy patterns that may be present in a given texture exemplar.

Specifically, we currently detect and extract: (i) dominant low-frequency changes in color and in local texture pattern, which we encode as a continuous scalar progression map; and (ii) dominant orientation of the local texture patterns, which we encode as a 2D direction field. We refer to the extracted progression map and direction field collectively as the *source guidance map*. During synthesis each of these visual characteristics may be controlled separately by the corresponding components in a *target guidance map*.

3.1. Scalar progression map extraction

Previous research in texture analysis and classification has shown that textures may be locally characterized by a variety of high-dimensional descriptors, such as color histograms, filter bank responses [MBSL99, VZ05], and the joint distribution of intensity values over small patches [VZ09]. Thus, a texture exemplar may be represented as a collection of points in a high-dimensional feature space. We further assume that the feature points are located on a low-dimensional manifold embedded in the feature space [LDR09], and apply manifold learning in order to project the feature points onto the manifold. The assumption is that the dominant changes across the texture will be reflected by the coordinates of each point on the learnt manifold.

Specifically, we use the Isomap method [TdSL00] to compute the non-linear dimensionality reduction that projects the high-dimensional texture descriptors to a low-dimensional manifold. The descriptors that we use for this purpose consist of two components: (i) a color histogram computed over a local texture window (we use a 3D color histogram with $64 = 4 \times 4 \times 4$ bins), and (ii) a histogram of filter bank response over the same window (we use the rotationally invariant MR8 filter bank [VZ05], also with 64 bins). We chose to apply Isomap on a full matrix of pairwise distances after experimenting with several alternatives, including using distances to only nearest neighbors from each sample, and using t-SNE [MH08] instead of Isomap, but found that our method of obtaining the scalar guidance channel yielded the best results.

Note that differently from the appearance manifolds of Wang et al. [WTL*06], our high-dimensional feature vectors are texture descriptors, computed over neighborhoods of substantial size, rather than pointwise reflectances (measured BRDFs or CIE Lab colors). Thus, in contrast to [WTL*06, XWT*08] our manifolds capture not only changes in color, but also changes in other aspects, such as scale, pattern, and more. This is demonstrated in Figures 5 and 6.

Our Isomap-based progression analysis proceeds as follows:

- Extract the texture descriptors (3D color histograms and filter bank responses) using a sliding window. In our experiments we used windows of size 48×48, with a stride of 8.
- Construct a distance matrix between pairs of feature vectors, where the distance is given by the sum of Earth Mover's Distances (EMD) [RTG00] between the corresponding histograms.
- Apply the Isomap method on the distance matrix to obtain the

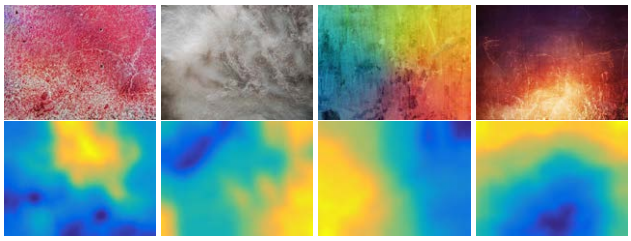


Figure 3: *Top row: several spatially inhomogeneous textures; Bottom row: corresponding scalar progression channels computed by our method.*

coordinates of the embedding of each feature vector on a one-dimensional manifold.

- Create a scalar map, by assigning to the location from which each feature vector was extracted its corresponding one-dimensional coordinate.
- Upsample the Isomap-generated map to a full resolution scalar progression channel.

Several textures exhibiting various kinds of spatial progressions are shown in Figure 3, along with the extracted scalar progression channels.

It is worth noting that other distance measures, including the L_2 distance and the Chi-square measure, were also tested, but found to perform less well than EMD.

3.2. Orientation field extraction

As may be seen in Figure 2, some textures exhibit strong local directionality (anisotropy), which may also vary spatially. Attempting to perform constrained synthesis from such textures, without accounting for local directionality might fail to produce satisfactory results. Thus, our goal is to detect anisotropy of the local texture patterns, so that they may be controlled during synthesis.

Local anisotropy. We measure local anisotropy by computing the histogram of oriented gradients (HoG) descriptor [DT05] (using 9 bins) for non-overlapping $n \times n$ texture patches ($n = 16$), and obtain the dominant orientation through PCA analysis on the corresponding set of direction vectors. This is similar to a method described by Feng and Milanfar [FM02]. The eigenvector corresponding to the largest eigenvalue defines the dominant orientation. To account for the magnitude of gradients, we scale the orientation vectors (i.e., the eigenvectors) by the corresponding eigenvalue. Thus, in strongly anisotropic regions, the orientation vectors have larger magnitudes than in weakly anisotropic ones. As may be seen in Figure 1(b), the orientations thus obtained tend to form a smooth vector field over the exemplar, while also reflecting the strength of the local anisotropy.

Note that the dominant orientations computed through PCA analysis are in fact undirected, i.e., each orientation can rotate by 180°. We found that assigning harmonized directions to all orientation vectors can be tricky in some cases and is not needed for the purpose of synthesizing textures based on target guidance maps.

4. Controlled Synthesis

Our controlled texture synthesis approach is based on the self-tuning texture optimization method [KNL*15]. Texture optimization (originally introduced by Kwatra et al. [KEBK05]) optimizes the similarity between the synthesized texture T and the exemplar S, measured over a set of overlapping local patches:

$$\min_{\{\mathbf{t}_i,\mathbf{s}_i\}} \sum_{i \in T} d(\mathbf{t}_i, \mathbf{s}_i), \tag{1}$$

where $\mathbf{t}_i$ is a square $N \times N$ patch in T with top-left pixel i, and $\mathbf{s}_i$ refers to an approximate nearest neighbor of $\mathbf{t}_i$ in the exemplar S. The approximate nearest neighbors are computed efficiently using the PatchMatch algorithm [BSFG09]. Similarity between patches in (1) are measured as the sum of squared color distances, i.e.,

$$d(\mathbf{t}_i, \mathbf{s}_i) = \|\mathbf{t}_i - \mathbf{s}_i\|_2^2. \tag{2}$$

We make two major modifications to the above method. First, given the source guidance channels (automatically computed as described in Section 3), we modify the Euclidean distance metric in (2) in order to constrain the synthesis to comply with the target guidance channels provided by the user:

$$\begin{aligned} d(\mathbf{t}_i, \mathbf{s}_i) \;=\; & \|\mathbf{t}_i - \mathbf{s}_i\|_2^2 + \\ & \alpha(\beta d_p(\mathbf{t}_i, \mathbf{s}_i) + (1-\beta) d_o(\mathbf{t}_i, \mathbf{s}_i)), \end{aligned} \tag{3}$$

where d_p and d_o measure the similarity of the progression channels and the direction fields, respectively:

$$d_p(\mathbf{t}_i, \mathbf{s}_i) = \|p(\mathbf{t}_i) - p(\mathbf{s}_i)\|_1, \tag{4}$$

$$\begin{aligned} d_o(\mathbf{t}_i, \mathbf{s}_i) &= \sum_{x \in \mathbf{s}_i, y \in \mathbf{t}_i} |\boldsymbol{v}_x||\boldsymbol{v}_y|(1 - \cos\theta) \\ &= \sum_{x \in \mathbf{s}_i, y \in \mathbf{t}_i} (|\boldsymbol{v}_x||\boldsymbol{v}_y| - |\boldsymbol{v}_x \cdot \boldsymbol{v}_y|). \end{aligned} \tag{5}$$

Here, $p(\mathbf{t}_i)$ and $p(\mathbf{s}_i)$ are the vectors of progression channel values of patches $\mathbf{t}_i$ and $\mathbf{s}_i$, $\boldsymbol{v}_x$ and $\boldsymbol{v}_y$ are the direction vectors at pixels x and y, and θ is the angle between $\boldsymbol{v}_x$ and $\boldsymbol{v}_y$. The linear combination coefficient β balances between the scalar progression and the direction field constraints. We currently set it to $\beta = 0.5$ for textures that exhibit both spatial inhomogeneity and local anisotropy. The constraint weight α balances between the color difference and the guidance channel difference. All of our results were obtained with $\alpha = 0.9$. Note that we do not normalize the vectors $\boldsymbol{v}_x$ and $\boldsymbol{v}_y$, since we want to make our orientation control adaptive to the magnitude of local anisotropy. In practice, the user provided target vectors are typically normalized, so $\boldsymbol{v}_y$ usually has unit length, while, as mentioned before, the extracted source vectors have different lengths according to the strength (eigenvalue) of local anisotropy. Thus, eq. (5) will only play an important role in areas of strong anisotropy, and will not significantly affect the synthesis in areas where the anisotropy is weak (and therefore the extracted orientation may be less reliable).

Second, we modify the PatchMatch algorithm to allow for rotations when searching for the approximate nearest neighbor. Such an extension is already used in the generalized PatchMatch algorithm [BSGF10]. However, rather than simply propagating the rotation angle from a pixel to its neighbor (as done in [BSGF10]), we predict the rotation angle using the target guidance vector field.

Specifically, let θ denote the rotation angle at pixel i, then the propagated rotation angle at pixel $i+1$ is set to $\theta \pm \delta$, where δ is the difference in the orientation between pixels i and $i+1$ of the target guidance field. Following the propagation, random search is used to improve both the predicted offset and the predicted rotation.

Unlike our approach, Lukáč et al. [LFA*15] do not include patch rotation in the propagation and the random search stages of PatchMatch. Instead, they directly compute the rotation angle between $\mathbf{t}_i$ and $\mathbf{s}_i$, and rotate the local frame of $\mathbf{s}_i$ to align with the orientation of $\mathbf{t}_i$ when computing the patch distance (2). As a result, they only search the offset space, and force the orientation of source patches to align with the desired target orientation. In contrast, our approach amounts to a soft constraint on the orientation, and includes it in the search along with the offset, resulting in a larger search space. Figure 9 compares our method of orientation control to [LFA*15], demonstrating slightly better results. However, the true advantage of our approach is that it enables simultaneous control of orientation and spatial inhomogeneity.

4.1. Target guidance refinement

User-provided target guidance channels are typically smoother and exhibit less spatial detail than our automatically extracted source guidance channels. Overly smooth target guidance channels may result in artificially looking patterns and tend to cause repetition artifacts, thus it is often helpful to enrich them with additional detail. We address this issue by injecting suitably adapted multi-scale noise into the user-provided smooth guidance channels.

Our approach is inspired by the ubiquity of Perlin noise [Per85] in procedural generation of realistic looking natural patterns, and by the effectiveness of manipulating statistics at multiple scales [HB95]. Specifically, we compute two Laplacian pyramids [BA83], one for the source guidance channel P_{src} and another for the target guidance channel P_{tgt}. Recall that the coarsest layer in a Laplacian pyramid is a low-pass filtered and downsampled version of the image, while the remaining layers are the differences between succesively low-pass filtered versions. We modify the coarsest level of P_{tgt} by matching its histogram to that of the corresponding level of P_{src}. To each of the remaining band-pass layers we add Perlin noise of the appropriate scale, scaled such that its variance matches that of the values at the corresponding layer of P_{src}. The refined guidance field is obtained by reconstruction from the modified P_{tgt}. Thus, the refined guidance field contains noise at multiple scales, whose statistics at each scale are derived from those of the source guidance field.

The benefits of our target guidance refinement technique are demonstrated in Figure 4. As may be seen in this figure, synthesis results produced directly using a smooth bump-shaped guidance channel tend to produce somewhat artificially looking results with some noticeable repetitions. The results produced with the refined guidance channels are considerably improved.

5. Results and Applications

We have implemented all of our texture analysis in Matlab. It takes roughly 60 minutes to extract the source guidance channels from a 600×400 texture exemplar. Most of this time is spent on constructing the matrix of the distances between the feature vectors, which is then provided as input to the Isomap algorithm.

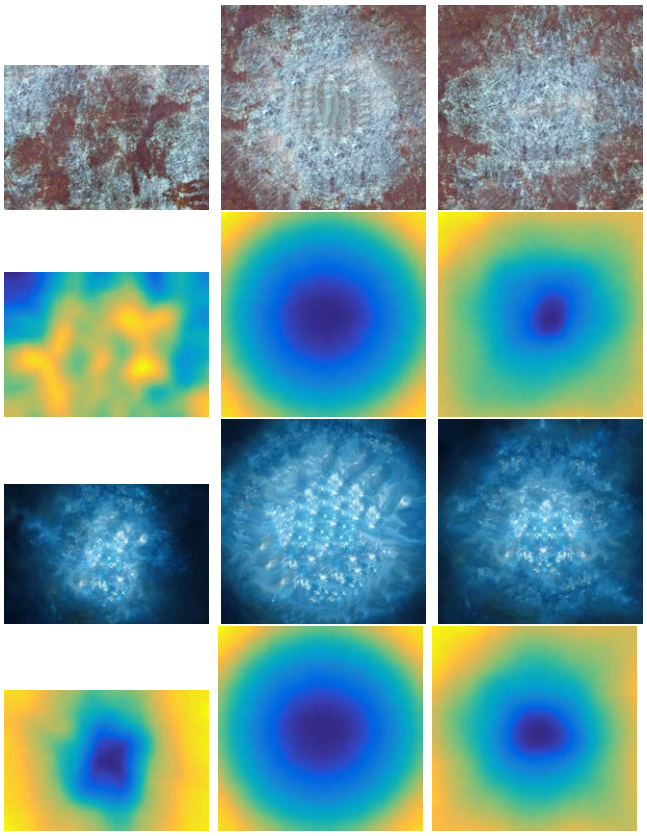


Figure 4: *Guidance refinement: Left column: exemplars with their extracted progression maps; Middle: a smooth bumpshaped target guidance produces a less natura looking texture with visible repetitions; Right: results synthesized from refined guidance channels.*

Our texture synthesis implementation is built upon the available self tuning texture optimization code [KNL*15]. It takes roughly 25 minutes to synthesize a 512×512 texture using our guided synthesis approach. All the times were measured on an Intel Xeon E5-2687W CPU running at 3.4GHz with 8GB of RAM.

Figure 5 shows a gallery of textures exhibiting significant spatial changes in luminance, hue and/or saturation, scale, and pattern. Next to each exemplar (1st and 4th columns) we show the scalar guidance channel extracted by our analysis (2nd and 5th columns). The textures in this figure are not strongly anisotropic, and the source vector field is therefore not shown. The 3rd and 6th columns show our guided synthesis results. In all of the examples, the target guidance field was a smooth Gaussian bump (or an inverted version of it), which was automatically refined as described in Section 4.1. The refined target guidance channels, as well as many additional synthesis results are included in the supplementary materials. It should be noted that the bump-shaped target map was chosen arbitrarily in order to demonstrate the robustness of our method. In

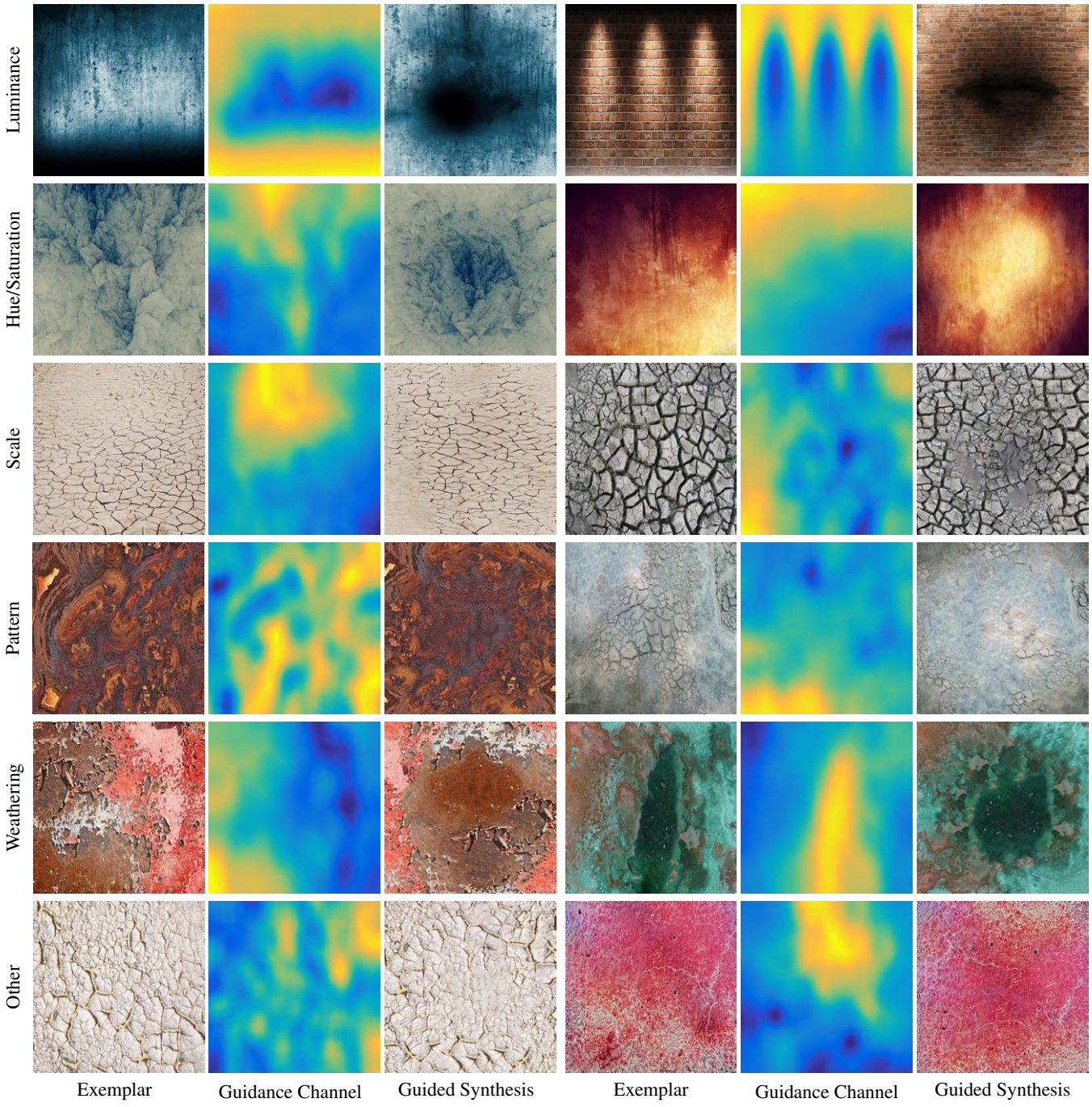


Figure 5: *Spatial progression control: a variety of textures exhibiting different kinds of spatial progressions. Each row shows two different textures with a similar kind of spatial progression. For each texture, we show the input exemplar, followed by the source guidance channel, and a result synthesized using a bump-shaped target guidance channel.*

a practical scenario, a custom designed target map would typically be created for each texture, as demonstrated below in Section 5.1.

As described in Section 2, Xue et al. [XWT*08] also compute a weathering degree map based on a single image. We compared our approach to theirs both in terms of the feature vectors used (high-dimensional texture descriptors vs. pixelwise CIE Lab colors) and dimensionality reduction (Isomap vs. ratio of geodesic distances to two user-specified extremes). A 2×2 comparison matrix was formed consequently. As shown in Figure 6, when only using pixel colors as feature vectors, the resulting degree maps fail to cap-

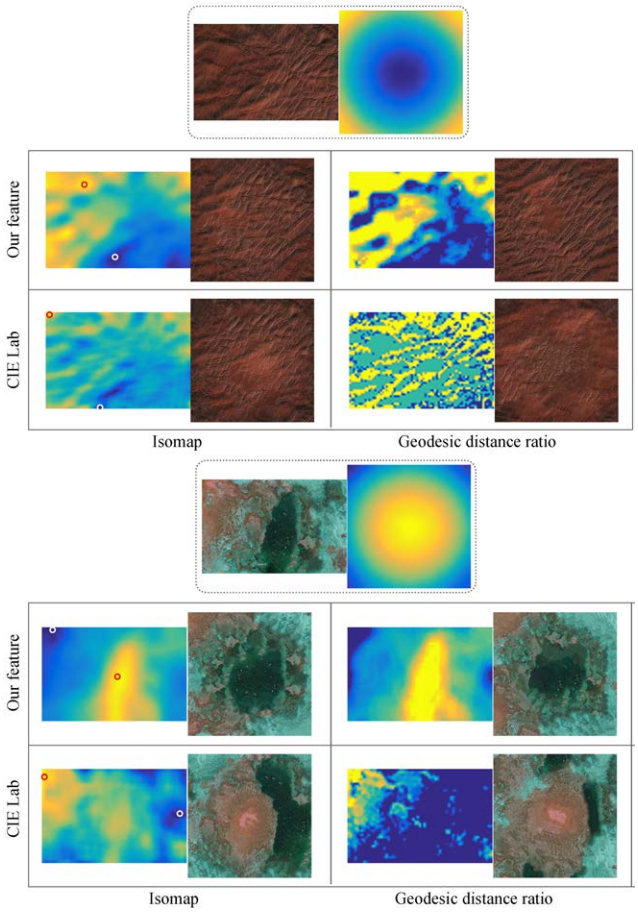


Figure 6: *Comparison with Xue et al. [XWT*08] in terms of the feature vectors and the dimensionality reduction used. The input texture exemplar and the corresponding target progression map are shown above each table. For each texture, we show a 2×2 comparison matrix, where the vertical dimension corresponds to different feature vectors, and the horizontal dimension to different dimensionality reduction methods. We can observe that our approach is able to better capture progressions that cannot be characterized using pixelwise colors alone, and yields better controllability. Note that when using geodesic distance ratios, Xue et al. made use of user-specified extrema. In this comparison, we automatically set these extrema to the extrema of the Isomap-based maps in the leftmost column (indicated using small circles).*

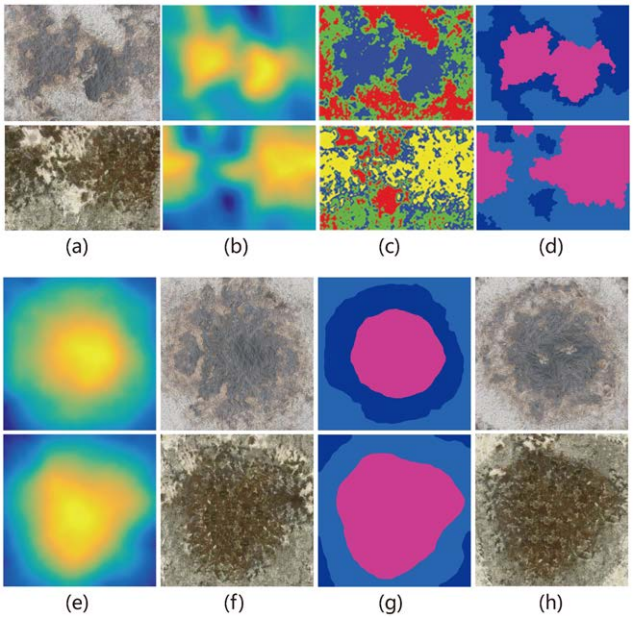


Figure 7: *Comparison with discrete label maps. (a) input exemplar, (b) our progression map, (c) discrete label map from [RCOL09], and (d) discrete label map at coarsest scale generated by [LSA*16]. (e,f) continuous target guidance map and corresponding synthesis result; (g) target map obtained by discretization of (e), and the corresponding result (h). For a fair comparison, the discretization was done such that the relative area of each label in (g) is the same as in the maps in (d).*

ture the pattern progression, leading to less successfully controlled synthesis results. In contrast, when using our high-dimensional histogram-based feature vectors with EMD distances, the guidance maps capture such progressions well. As for dimensionality reduction, we can see that the progression maps generated by Isomap are smoother than those produced with geodesic distance ratios, again yielding better controllability. In addition, our approach is fully automatic, while geodesic distance ratios require user interaction, which may introduce instability.

Figure 7(a–d) shows qualitative comparisons between our continuous progression maps and the discrete label maps of Rosenberger et al. [RCOL09] and Lockerman et al. [LSA*16]. As discussed in Section 2, discrete label maps are less suited for describing inhomogeneous textures that exhibit gradual changes, while using our continuous progression maps yield better control over the result. This is demonstrated in Figure 7(e–h), where we compare our results to those obtained using Lockerman's label map as source guidance and a suitably discretized version of our target map as the target guidance. Note, the unnatural looking dark ring formed when using the discrete label map in the top result in (h), and the less natural boundaries between the three areas in the bottom result in (h).

Figure 8 demonstrates our ability to control local orientation in strongly directional textures. The middle column shows the input exemplar superimposed with the vector field extracted as described in Section 3. In this figure we control only the local orientation using a wave-shaped target vector field, and no scalar guidance channel was used. Specifically, we set β in equation (3) to zero.

Figure 9 compares our method of orientation control to [LFA*15], demonstrating that our soft constraints produce slightly more continuous results, compared to the hard constraints used in [LFA*15]. However, the true advantage of our approach is that different spatially variant properties may be controlled *independently and simultaneously* with the local orientation. This is demonstrated in Figure 10. To our knowledge, our method is the first to offer such control.

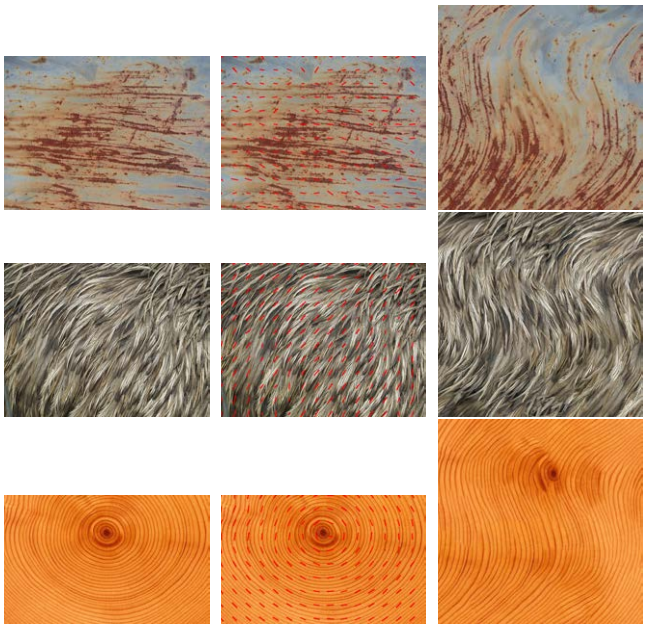


Figure 8: *Orientation control: given a texture exemplar with noticeable local anisotropy (left) we extract a vector field that describes it (middle). A wave-shaped target vector field is then used to control the local direction across the synthesized result (right).*

Figure 9: *Comparison with the orientation control method of Lukáč et al. [LFA*15]. Left: input exemplar, middle: result using a hard orientation constraint [LFA*15], right: our result (soft constraint). As highlighted in the middle column, using hard constraint sometimes introduces discontinuities in color and linear features, especially when the target field becomes more complex. In contrast, using soft constraints helps produce smoother orientation changes and better preserves color and linear features.*

Figure 10 also shows a self-evaluation of our method. Although there is no "ground truth" that our progression channels could be compared to, we show a qualitative comparison between the progression channels extracted by our method from a texture that was synthesized using a progression map to begin with. As we can see in columns (a) and (e) in Figure 10, the recovered guidance channels are indeed qualitatively close to the target guidance used to generate the textures, which demonstrates the consistency of our analysis. Note that exact recovery cannot be expected in this experiment, since the target guidance is only used as a soft constraint by our synthesis process.

5.1. Applications

Lu et al. [LDR09] detect dominant texture areas in an exemplar based on a manifold generated using the diffusion distance. They demonstrate that their method enables them to synthesize uniform textures from "contaminated" exemplars. Our scalar progression channel may be simply yet effectively used for the same purpose, as demonstrated in Figure 11. From the source progression channel we obtain a binary map using Otsu's automatic thresholding [Ots79] (shown superimposed over the progression channel in the middle of each row). The larger among the two resulting segments is considered to contain the dominant texture, and synthesis is then carried out using only patches contained therein. Additional examples are included in the supplementary materials.

Although our approach was not explicitly designed to assess the degree of weathering, our scalar progression channel can nevertheless function as an effective "age map", when extracted from a texture depicting a partially weathered surface. Thus, by providing a suitable target guidance channel we are able to synthesize textures that appear less or more weathered than the original exemplar. This is demonstrated in Figure 12, which shows several sequences similar to those that may be produced using the recent method of Bellini et al. [BKCO16] that was specifically designed with (de-)weathering in mind.

To achieve the results shown in Figure 12, we first threshold the source guidance channel G to identify a "clean" region of the input texture T. A clean texture T_c is then synthesized using only patches from the clean region (similarly to our dominant texture synthesis described earlier). However, while synthesizing T_c, we also synthesize a new guidance channel G_c from the source guidance channel G. Another threshold t_w is used to detect the fully weathered regions of T (areas for which $G < t_w$). Next, we generate a sequence of guidance maps G_i, by gradually transitioning from G_c to G, and then continuing to gradually darken those regions of the guidance channel which have not yet reached t_w, while freezing the already weathered regions. Synthesizing textures using this sequence of guidance maps gives us the desired time varying weathering sequence.

Finally, we demonstrate the use of our technique to synthesize textures targeting a specific 3D model, where the control that we provide proves crucial. Two such examples with user specified target guidance maps are shown in Figure 13. A UV map was created for each model in 3ds Max, and the user-provided target guidance maps were drawn over these UV maps (in the figure the guidance map is shown texture-mapped onto the model). The scalar channel was created using a few simple gradient fills in Photoshop. The vector field for the puppy example was interpolated from a

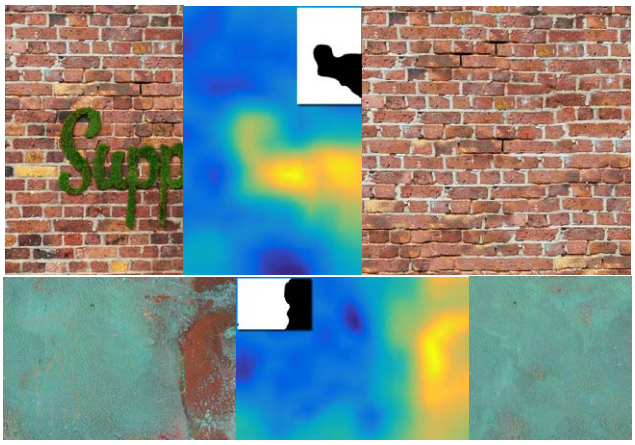


Figure 11: *Dominant texture synthesis: for textures that exhibit one dominant mode (such as the two examples on the left) we are able to synthesize a texture having only that mode (right) by applying automatic thresholding to our scalar guidance channel (middle).*

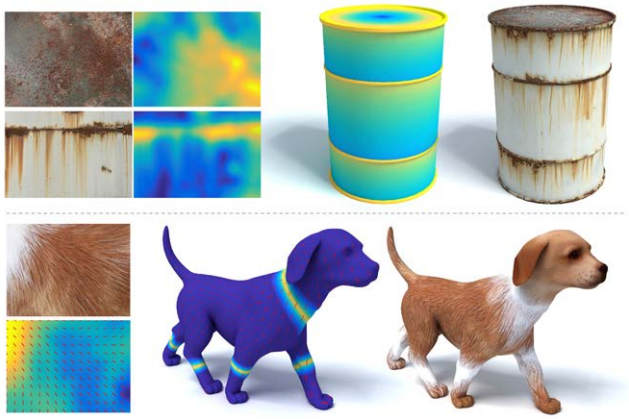
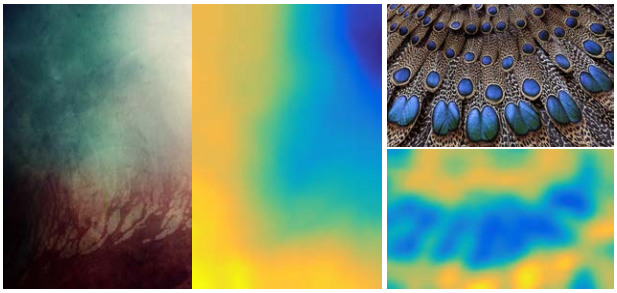


Figure 13: *Controlled texturing for 3D objects with manually specified target guidance maps. The barrel in the top row is textured using two different rust textures, one for the top and another for the side. The user provided scalar channel is created using several simple gradients. The puppy is textured using a directional texture, whose guidance map consist of a scalar channel and a vector field. The puppy's head is textured using a separately provided texture, while the fur texture over the body is synthesized using our method.*



(a) (b)

Figure 15: *Limitations: (a) The transition in hue (from red to green) is uncorrelated with the lightness change from bottom left to top right, and the resulting progression channel does not capture the transition in hue. (b) The progression channel does not adequately distinguish between the largest blue spots and the smaller ones next to them.*

few user-drawn guiding curves using harmonic interpolation, i.e., by solving Laplace equations with Dirichlet boundary conditions along the curves.

In many cases, the target guidance maps can also be automatically generated, since texture variations are often correlated with local geometric characteristics [MKC*06]. For example, when texturing a dusty object, the portions of the surface pointing upward tend to collect more dust. Hence, a target guidance map can be automatically computed based on surface normal orientation; see Figure 14(e). Similarly, when rendering a translucent object, the degree of translucency is determined by the local thickness of the object. Synthesizing the texture by using a target guidance map generated based on shape diameter function [SSCO08] yields convincing results; see Figure 14(c). Other examples in Figure 14 show that other geometric features, such as height (z-coordinates) and curvature, can also be used to achieve various desired effects. In addition, using the method of [XCOJ*09], a target vector field can also be automatically generated according to crest lines on 3D surfaces, which provides additional controls over the local orientation of the texture. Such examples are shown in Figures 1 and 14(g), where the target scalar guidance channel was computed from normal direction (inwards or outwards), and in Figure 14(h), where height map was used as the target scalar channel.

6. Conclusions and Future Work

We have described a new method for automatic analysis of inhomogeneous textures. Factoring out local orientation from spatial changes in other visual attributes, we extract two guidance channels from an input exemplar. These channels enable controlling both spatial progression and local orientation across synthesized textures. While the analysis is surprisingly simple, the guidance maps that it produces are effective for a wide variety of inhomogeneous textures, and we have demonstrated their usefulness for a number of scenarios, some of which were previously addressed by dedicated works.

Our approach has a few limitations. The most obvious limitation is the high computation times required to compute the distances between the feature points that are provided as input to the Isomap algorithm.

Our approach does not handle well cases where there are multiple uncorrelated spatial progressions occurring across the surface, since such behavior cannot be captured by a scalar valued channel (Figure 15(a)). In particular, the approach is not sensitive enough to changes in the scale of texture elements. When other types of

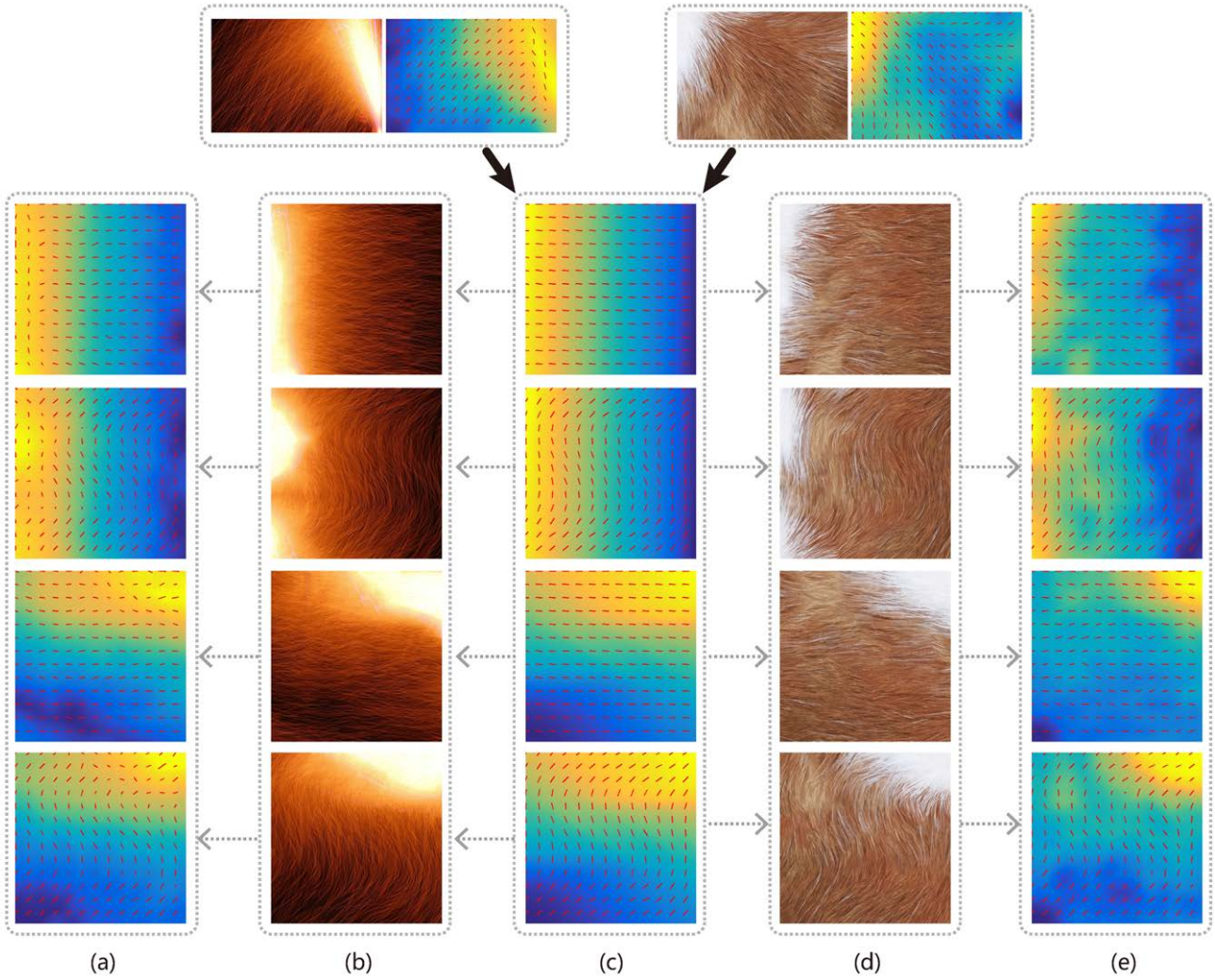


Figure 10: *Simultaneous control of progression and orientation: the top row shows two input exemplars with their corresponding extracted source guidance maps (the scalar channel is overlaid with the detected directions.) The bottom part shows four different target guidance maps (c) that control progression and direction independently. The corresponding synthesized results are shown in (b) and (d). For self-validation, we repeat our texture analysis on these results, and the recovered guidance channels are shown in (a) & (e). It may be seen that these recovered channels are indeed qualitatively close to the target guidance. Note that exact recovery should not be expected here, since the target guidance only serves as a soft constraint in our synthesis process.*

changes are simultaneously present, they seem to take precedence in the Isomap embedding (Figure 15(b)).

Finally, example-based texture synthesis in general is only as good as the input exemplar. In non-uniform texture exemplars, where the appearance is changing across the exemplar, the number of patches available to synthesize each kind of appearance is limited. Thus, when performing constrained synthesis, one encounters repetition artifacts more frequently than with unconstrained optimization. Such examples may be seen in the supplementary materials.

In future work it would be interesting to attempt addressing these limitations. In particular it would be interesting to look for multi-dimensional manifold embedding that would replace our scale progression channel with a multi-valued one. We experimented with 2D Isomap embedding, but this attempt has not been successful.

Acknowledgments

We thank the anonymous reviewers for their constructive comments, and are grateful to Su Xue and Xin Tong for providing their codes to test. This work was supported in part by NSFC (61602461, 61522213, 61402459), 973 Program (2015CB352501), Guangdong Science and Technology Program (2014TX01X033, 2015A030312015, 2016A050503036), Shenzhen Innovation Program (JCYJ20151015151249564, JCYJ20150630114942295), NSERC (293127) and the Israel Science Foundation.

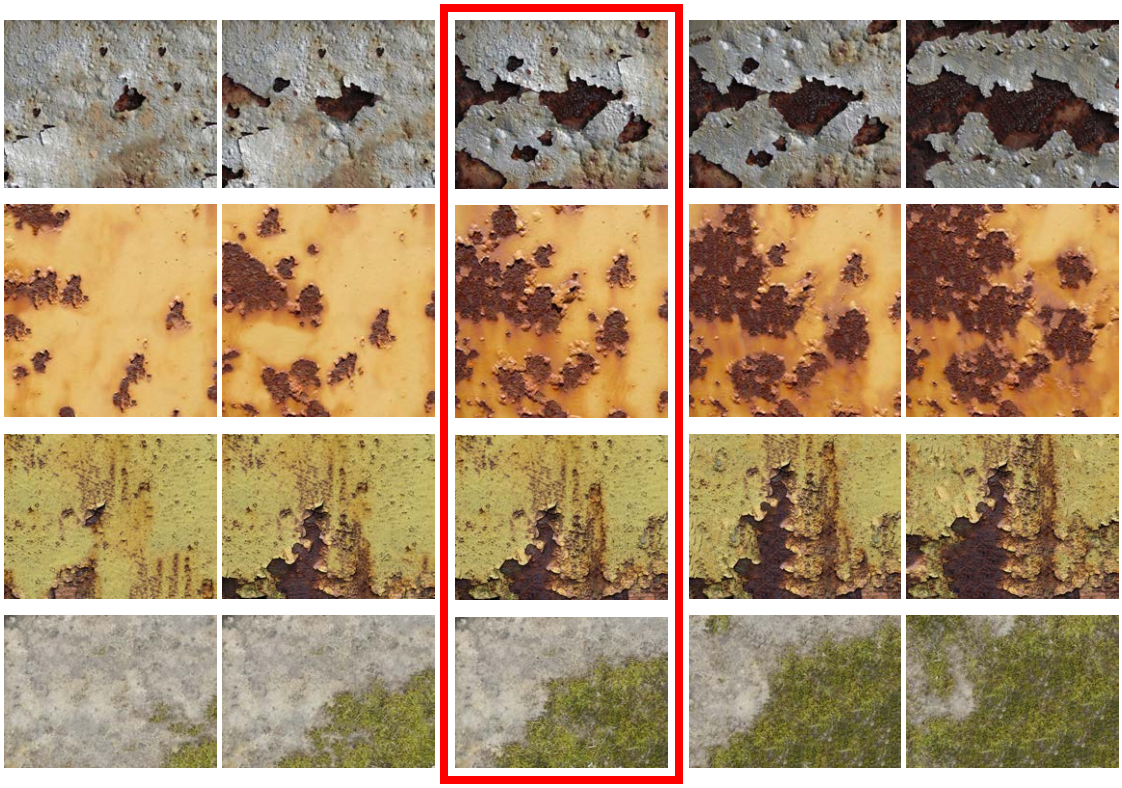


Figure 12: *Time varying weathering sequences. Our scalar progression channels can function as age maps, when extracted from textures depicting partially weathered surfaces. The progression maps can be manipulated to synthesize less or more weathered appearance, similarly to the time varying sequences of Bellini et al. [BKCO16]. The original exemplars are in the middle column. The corresponding sequences of guidance maps are included in the supplementary materials.*

References

[Ash01] ASHIKHMIN M.: Synthesizing natural textures. In *Proc. Symp. Interactive 3D Graphics* (2001), pp. 217–226. 3

[BA83] BURT P. J., ADELSON E. H.: The laplacian pyramid as a compact image code. *IEEE Transactions on Communications* (1983). 6

[BKCO16] BELLINI R., KLEIMAN Y., COHEN-OR D.: Time-varying weathering in texture space. *ACM Trans. on Graphics (Proc. of SIGGRAPH) 35*, 4 (2016), 1–11. 4, 9, 12

[BPMG04] BOSCH C., PUEYO X., MÉRILLOU S., GHAZANFARPOUR D.: A physically-based model for rendering realistic scratches. *Computer Graphics Forum 23*, 3 (Sept. 2004), 361–370. 4

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D.: PatchMatch: a randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2009) 28*, 3 (2009), Article no. 24. 3, 5

[BSGF10] BARNES C., SHECHTMAN E., GOLDMAN D. B., FINKELSTEIN A.: The generalized PatchMatch correspondence algorithm. In *Proc. ECCV* (Sept. 2010). 5

[DEJ*99] DORSEY J., EDELMAN A., JENSEN H. W., LEGAKIS J., PEDERSEN H. K.: Modeling and rendering of weathered stone. In *Proc. SIGGRAPH '99* (1999), ACM Press, pp. 225–234. 4

[DGA04] DESBENOIT B., GALIN E., AKKOUCHE S.: Simulating and modeling lichen growth. *Computer Graphics Forum 23*, 3 (Sept. 2004), 341–350. 4

[DH96] DORSEY J., HANRAHAN P.: Modeling and rendering of metallic patinas. In *Proc. SIGGRAPH '96* (Aug. 1996), Addison Wesley, pp. 387–396. 4

[DRS08] DORSEY J., RUSHMEIER H., SILLION F.: *Digital Modeling of Material Appearance*. Computer Graphics. Morgan Kaufmann / Elsevier, Dec. 2008. . 336 pages. 4

[DSB*12] DARABI S., SHECHTMAN E., BARNES C., GOLDMAN D. B., SEN P.: Image Melding: Combining inconsistent images using patch-based synthesis. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2012) 31*, 4 (2012). 3

[DT05] DALAL N., TRIGGS B.: Histograms of oriented gradients for human detection. In *Proc. CVPR 2005* (2005). 2, 5

[EF01] EFROS A. A., FREEMAN W. T.: Image quilting for texture synthesis and transfer. In *Proc. SIGGRAPH 2001* (2001), pp. 341–346. 3

[EL99] EFROS A. A., LEUNG T. K.: Texture synthesis by non-parametric sampling. *Proc. ICCV '99 2* (1999), 1033–1038. 3

[FM02] FENG X., MILANFAR P.: Multiscale principal components analysis for image local orientation estimation. In *Proceedings of the 36th Asilomar Conference on Signals, Systems and Computers* (2002), pp. 478–482. 2, 4, 5

[GTR*06] GU J., TU C.-I., RAMAMOORTHI R., BELHUMEUR P., MATUSIK W., NAYAR S.: Time-varying surface appearance: acquisition, modeling and rendering. *ACM Transactions on Graphics 25*, 3 (Proc. SIGGRAPH 2006) (2006), 762–771. 3

[HB95] HEEGER D. J., BERGEN J. R.: Pyramid-based texture analysis/synthesis. *Proc. SIGGRAPH '95* (1995), 229–238. 6

[HJO*01] HERTZMANN A., JACOBS C. E., OLIVER N., CURLESS B.,

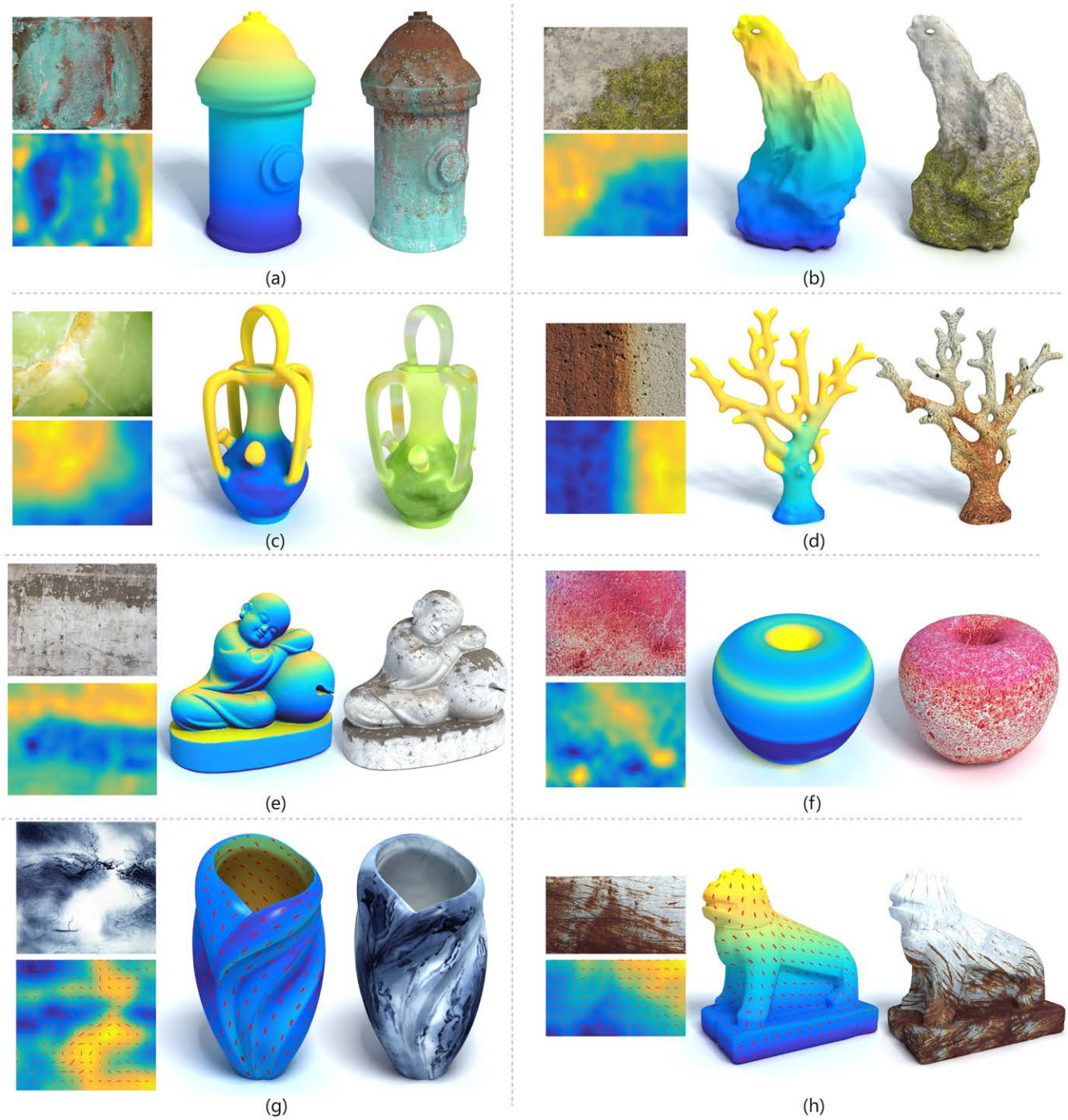


Figure 14: *Controlled texturing of 3D objects with automatically generated target guidance maps. The target scalar progression maps are automatically computed based on geometric features, where the specific features we chose include: height (a), (b) & (h), shape diameter function (c) & (d), normal orientation (e), curvature (f). Automatically generated vector fields are also used in (g) & (h) to control both non-uniform appearance and local orientation.*

SALESIN D. H.: Image analogies. *Proc. SIGGRAPH 2001* (August 2001), 327–340. ISBN 1-58113-292-1. 2, 3

[HZW*06] HAN J., ZHOU K., WEI L.-Y., GONG M., BAO H., ZHANG X., GUO B.: Fast example-based surface texture synthesis via discrete optimization. *The Visual Computer 22*, 9-11 (2006), 918–925. 3

[IEKM16] IIZUKA S., ENDO Y., KANAMORI Y., MITANI J.: Single image weathering via exemplar propagation. *Computer Graphics Forum (Proc. of Eurographics 2016)* (2016). 4

[KEBK05] KWATRA V., ESSA I., BOBICK A., KWATRA N.: Texture optimization for example-based synthesis. *ACM Transactions on Graphics 24*, 3 (Proc. SIGGRAPH 2005) (2005), 795–802. 2, 3, 5

[KNL*15] KASPAR A., NEUBERT B., LISCHINSKI D., PAULY M., KOPF J.: Self tuning texture optimization. *Computer Graphics Forum 34*, 2 (May 2015). 2, 3, 5, 6

[KSE*03] KWATRA V., SCHÖDL A., ESSA I., TURK G., BOBICK A.: Graphcut textures: image and video synthesis using graph cuts. *ACM Transactions on Graphics 22*, 3 (Proc. SIGGRAPH 2003) (2003), 277–286. 3

[LDR09] LU J., DORSEY J., RUSHMEIER H.: Dominant texture and diffusion distance manifolds. *Computer Graphics Forum 28*, 2 (2009), 667–676. 4, 9

[LFA*15] LUKÁČ M., FIŠER J., ASENTE P., LU J., SHECHTMAN E., SÝKORA D.: Brushables: Example-based Edge-aware Directional Texture Painting. *Computer Graphics Forum 34*, 7 (Oct. 2015), 257–267. 2, 4, 6, 8, 9

[LGG*07] LU J., GEORGHIADES A. S., GLASER A., WU H., WEI L.-Y., GUO B., DORSEY J., RUSHMEIER H.: Context-aware textures. *ACM Trans. Graph. 26*, 1 (2007), 3. 3

[LH06] LEFEBVRE S., HOPPE H.: Appearance-space texture synthesis. *ACM Transactions on Graphics 25*, 3 (Proc. SIGGRAPH 2006) (2006), 541–548. 3

[LL12] LASRAM A., LEFEBVRE S.: Parallel patch-based texture synthesis. *ACM SIGGRAPH / Eurographics Conference on High-Performance Graphics (HPG) 2012* (2012), 115–124. 3

[LSA*16] LOCKERMAN Y. D., SAUVAGE B., ALLÈGRE R., DISCHLER J.-M., DORSEY J., RUSHMEIER H.: Multi-scale label-map extraction for texture synthesis. *ACM Trans. on Graphics (Proc. of SIGGRAPH) 35*, 4 (July 2016), 140:1–140:12. 3, 8

[MBSL99] MALIK J., BELONGIE S., SHI J., LEUNG T.: Textons, contours and regions: Cue integration in image segmentation. In *Proc. ICCV '99* (Washington, DC, USA, 1999), vol. 2 of *ICCV '99*, IEEE Computer Society. 4

[MDG01] MERILLOU S., DISCHLER J.-M., GHAZANFARPOUR D.: Corrosion: simulating and rendering. In *Proc. Graphics Interface 2001* (2001), Canadian Information Processing Society, pp. 167–174. 4

[MH08] MAATEN L. V. D., HINTON G.: Visualizing Data using t-SNE. *Journal of Machine Learning Research 9*, Nov (2008), 2579–2605. 4

[MKC*06] MERTENS T., KAUTZ J., CHEN J., BEKAERT P., DURAND F.: Texture transfer using geometry correlation. In *Proceedings of the 17th Eurographics Conference on Rendering Techniques* (2006), EGSR '06, pp. 273–284. 10

[Ots79] OTSU N.: A threshold selection method from gray-level histograms. *Systems, Man and Cybernetics, IEEE Transactions on 9*, 1 (Jan 1979), 62–66. 9

[Per85] PERLIN K.: An image synthesizer. *Proceedings of SIGGRAPH '85* (1985), 287–296. 6

[RCOL09] ROSENBERGER A., COHEN-OR D., LISCHINSKI D.: Layered shape synthesis: Automatic generation of control maps for non-stationary textures. *ACM Trans. Graph 28*, 5 (Dec. 2009), 107:1–9. 3, 8

[RTG00] RUBNER Y., TOMASI C., GUIBAS L. J.: The earth mover's distance as a metric for image retrieval. *Int. J. Comput. Vision 40*, 2 (Nov. 2000), 99–121. 4

[SSCO08] SHAPIRA L., SHAMIR A., COHEN-OR D.: Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer 24*, 4 (2008), 249–259. 10

[TdSL00] TENENBAUM J. B., DE SILVA V., LANGFORD J. C.: A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 5500 (Dec. 2000), 2319–2323. 2, 3, 4

[VZ05] VARMA M., ZISSERMAN A.: A Statistical Approach to Texture Classification from Single Images. *Int'l J. Computer Vision 62*, 1-2 (2005), 61–81. 4

[VZ09] VARMA M., ZISSERMAN A.: A Statistical Approach to Material Classification Using Image Patch Exemplars. *IEEE Trans. Pattern Analysis and Machine Intelligence 31*, 11 (Dec. 2009), 2032–2047. 4

[WHZ*08] WEI L.-Y., HAN J., ZHOU K., BAO H., GUO B., SHUM H.-Y.: Inverse texture synthesis. *ACM Trans. Graph. 27*, 3 (2008), 1–9. 3

[WL00] WEI L.-Y., LEVOY M.: Fast texture synthesis using tree-structured vector quantization. *Proc. SIGGRAPH 2000* (2000), 479–488. 3

[WLKT09] WEI L.-Y., LEFEBVRE S., KWATRA V., TURK G.: State of the art in example-based texture synthesis. In *Eurographics 2009 State of The Art Reports* (March 2009), Eurographics. 1, 3

[WSI07] WEXLER Y., SHECHTMAN E., IRANI M.: Space-time completion of video. *Transactions on Pattern Analysis and Machine Intelligence (PAMI) 29*, 3 (2007), 463–476. 3

[WTL*06] WANG J., TONG X., LIN S., PAN M., WANG C., BAO H., GUO B., SHUM H.-Y.: Appearance manifolds for modeling time-variant appearance of materials. *ACM Transactions on Graphics 25*, 3 (Proc. SIGGRAPH 2006) (2006), 754–761. 2, 3, 4

[WWY14] WU R., WANG W., YU Y.: Optimized synthesis of art patterns and layered textures. *IEEE Transactions on Visualization and Computer Graphics 20*, 3 (Mar. 2014), 436–446. 3

[XCOJ*09] XU K., COHEN-OR D., JU T., LIU L., ZHANG H., ZHOU S., XIONG Y.: Feature-aligned shape texturing. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia) 28*, 5 (2009), 108:1–108:7. 10

[XWT*08] XUE S., WANG J., TONG X., DAI Q., GUO B.: Image-based Material Weathering. *Computer Graphics Forum 27*, 2 (Apr. 2008), 617–626. 2, 3, 4, 7, 8

[ZZV*03] ZHANG J., ZHOU K., VELHO L., GUO B., SHUM H.-Y.: Synthesis of progressively-variant textures on arbitrary surfaces. *ACM Transactions on Graphics 22*, 3 (Proc. SIGGRAPH 2003) (2003), 295–302. 3