Computers & Graphics (2021)

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Spatio-temporal BRDF: modeling and synthesis

Daniel Meister[a,b,1,*], Adam Pospíšil[a], Imari Sato[c], Jiří Bittner[a]

[a] *Czech Technical University in Prague, Czech Republic*
[b] *The University of Tokyo, Japan*
[c] *National Institute of Informatics, Japan*

ARTICLE INFO

Article history:
Received April 8, 2021

Keywords: Texture Synthesis, Reflectance Models, Natural Phenomena Animation

ABSTRACT

We propose a generalization of example-based texture synthesis to spatio-temporal BRDFs. A key component of our method is a novel representation of time-varying materials using polynomials describing time-varying BRDF parameters. Our representation allows efficient fitting of measured data into a compact spatio-temporal BRDF representation, and it allows an efficient analytical evaluation of distances between spatio-temporal BRDF parameters. We show that even polynomials of low degree are sufficient to represent various time-varying phenomena and provide more accurate results than the previously proposed representation. We are the first who applied the example-based texture synthesis on abstract structures such as polynomial functions. We present two applications of synthesizing spatio-temporal BRDFs using our method: material enlargement and transfer of time-varying phenomenon from an example to a given static material. We evaluated the synthesized BRDFs in the context of realistic rendering and real-time rendering.

1. Introduction

Despite enormous effort researchers devoted to studying optical properties of materials, it still remains a hot topic. However, significantly less attention has been paid to time-varying materials. The properties of a material might change drastically in time (e.g., wet vs. dry road) leading to very different appearance. Representing static spatially varying materials is a relatively complex task, and accounting for temporal effects adds another dimension to the representation, which makes the problem significantly more difficult. Despite this fact, there are methods that successfully handle time-varying materials for certain use cases. There are powerful techniques that simulate appearance changes such as weathering or aging [1, 2, 3, 4]; however, these methods can be hard to set up to match a particular real-world material. Capturing the temporal phenomena solves this problem [5, 6, 7]. On the other hand, the data acquisition is tedious and efficient processing of the captured data is still an open problem. For example, synthesis of larger material patches from measured data required enforcing continuity in both spatial and temporal domains, which is a non-trivial task addressed only with a partial success so far [5, 8].

In this paper, we extend example-based texture synthesis [9, 10, 11] for the spatio-temporal BRDF. This extension allows us to exploit the powerful example-based synthesis algorithms for tasks such as enlargement of spatio-temporal BRDFs, synthesis of spatio-temporal BRDFs guided by object geometry, or transfer of temporal phenomena from a spatio-temporal BRDF to a static BRDF. To achieve this, we propose two key components: a suitable representation of time-varying channels and a distance function for measuring the distance between time-varying channels. We represent time-varying channels as polynomial functions showing that even polynomials of low degree are sufficient to represent most of the materials. We show

*Corresponding author:
e-mail: meister.daniel@mail.u-tokyo.ac.jp (D. Meister)
[1] JSPS International Research Fellow

that this representation, unlike other approaches, is compatible with contemporary texture synthesis algorithms. Using this representation, we propose a distance function defined as integration over the temporal domain taking into account the temporal variation of a particular phenomenon. According to our best knowledge, we are the first who applied example-based texture synthesis on abstract structures such as polynomial functions, where the distance function is induced by the structure itself instead of scalar values such as RGB channels.

We present two applications of our approach. First, we synthesize larger material from a small example without visible seams, which is a basic task of the example-based texture synthesis. Second, we transfer a time-varying phenomenon from an exemplar material to a given static material (see Figure 1). To show the potential of our method, we integrate it with a global illumination renderer as well as real-time game engine.

In summary, our paper provides the following contributions:

- compact polynomial representation of spatio-temporal BRDFs;
- closed-form evaluation of distance between time-varying BRDFs;
- patch-based synthesis of spatio-temporal BRDFs;
- two applications of the proposed method: spatio-temporal BRDFs synthesis and transfer.

The paper is further structured as follows. Section 2 summarizes related work on time-varying materials and texture synthesis. Section 3 recalls the space-time appearance factorization model with related database of spatio-temporal BRDFs, and describes the concept of example-based texture synthesis. Section 4 presents our model of spatio-temporal BRDFs and the data used for our experiments. Section 5 explains details of the extension of texture synthesis for spatio-temporal BRDFs. Section 6 presents two application of this extension. Section 7 discusses the results and shows applications of the proposed method. Section 8 concludes the paper.

2. Related Work

Time-varying materials: Traditionally, various phenomena have been addressed individually using models based on physically inspired simulations: metallic patinas [12], flow of water [13], stone weathering [1, 14], lichen growth [15], peeling and cracking [2], dust accumulation [16], scratches [3], stains [4], and wet surfaces [17, 18]. Elton and Legrix [19] studied drying paints using polarized light reflectometry. Recently, Bosch and Patow [20] proposed a method that allows the transfer of flow phenomena between photographs. All these methods provide impressive results; nonetheless, they are restricted to particular phenomena. In our paper, we focus on a data-driven approach that is able to implicitly handle a large class of measured or simulated time-varying materials.

Chen et al. [21] proposed a general simulation based on tracing weathering particles conceptually similar to photon mapping. The density of these particles corresponds to a weathering degree which serves as blending parameter between an object texture and a weathered texture (e.g., wall and mold).

In the last decade, data-driven models have become popular. Lu et al. [6] proposed a data-driven parametric model of the drying effect. Gu et a.l [5] proposed a general non-linear factorization of spatial and temporal components of a general phenomenon. Similarly, Sun et al. [22] studied more complex time-varying phenomena such as drying of paints and dust accumulation. However, this approach is not spatially varying. Lu et al. [23] measured several time-varying phenomena on different objects and proposed a technique for transferring these phenomena to other objects based on the geometric context. Langenbucher et al. [7] captured time-varying BTF data of metal rusting and car paint. Ahmed et al. [24] studied spatio-temporal reflectance sharing for combining dynamic reflectance samples from multi-view video streams recorded under calibrated illumination. Zaidi [25] provided a survey on how to acquire and model time-varying materials.

Wang et al. [8], Xue et al. [26], Bandeira and Walter [27], Bellini et al. [28], and Iizuka et al. [29] proposed methods based on capturing data at a single moment from which they extract the effect using non-linear dimensionality reduction tools such as manifold learning. Recently, Guingo [30] proposed a content-aware texture deformation using the same principle. Although these approaches provide impressive results, they rely on user annotation and manually created weathering maps. Furthermore, data captured at a single moment may not be sufficient for more complex phenomena.

Kurt et al. [31] proposed an anisotropic analytical BRDF model with accurate data fitting, efficient importance sampling, and effective real-time rendering. Similarly, Ozturk et al. [32] studied linear approximation of BRDFs motivated by the fact that fitting the linear models is much easier than non-linear ones. Recently, Tongbuasirilai et al. [33] investigated separation of 2D and 3D BRDF representations into 1D factors introducing only a small fitting error. All these works concentrate on accurate fitting of the proposed models. In our work, we study a different problem. We fit polynomial functions through already fitted BRDF parameters in discrete time steps. Note that these models are neither spatially-varying nor temporally-varying.

Texture synthesis: Efros et al. [34] proposed texture synthesis based on region growing when a synthesized texture is composed of small quilts. Kwatra et al. [9] formulated texture synthesis problem as a global optimization iteratively improving the synthesized texture. Contemporary texture synthesis algorithms are based on this optimization. Barnes et al. [35] proposed an efficient randomized algorithm for searching nearest neighbors between patches which is a common problem in texture synthesis. Hertzmann et al. [36] proposed to use additional guiding channels to control the texture synthesis. The texture synthesis is not limited to RGB images. Fišer et al. [11] proposed to use the guiding channels corresponding to global illumination to stylize 3D renderings. Tong et al. [37] extended texture synthesis for BTF, which represents a material directly by measured data consisting of thousands of measurements from different views and light directions. Synthesizing temporal BTF data might be interesting. However, the amount

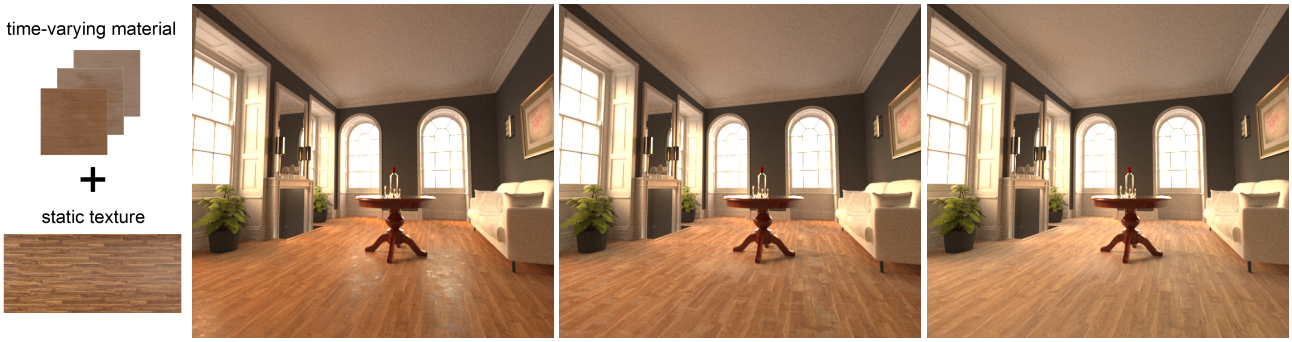


Fig. 1. Rendered images of the living room scene with transferred wood drying to the static floor texture using our method. The images are rendered in different stages from completely wet (left) to completely dry (right). Notice how specularity changes depending on the time-varying phenomenon.

of data would be enormous. Lefebvre and Hoppe [38] replaced point-wise colors by appearance vectors that incorporate non-local information such as feature and radiance-transfer data. Han et al. [39] proposed a method for synthesis animated textures on 3D mesh surfaces. Enrique et al. [40] suggested how to measure the distance between time-varying parameters, which are modeled as general functions. Recently, texture synthesis based on deep neural networks achieve remarkable results [41]. Zhou et al. [42] leveraged neural networks to synthesize non-stationary textures. Mazlov et al. [43] proposed a neural synthesis and transfer of spatially varying BRDFs. However, these methods currently cannot handle time-varying material models.

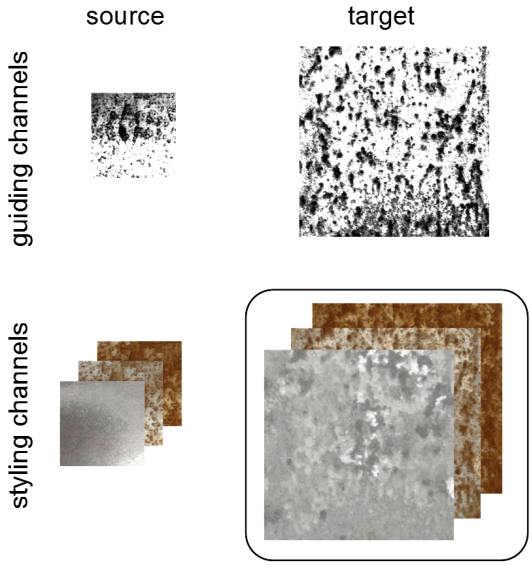


Fig. 2. The concept of styling and guiding channels. We transfer a time-varying phenomenon from a given example time-varying material, i.e., source styling channels, to a static texture. We extract target guiding channel from the static texture and source guiding channel from the exemplar time-varying material. The resulting material, i.e., target styling channels, is emphasized in the black border.

3. Background

Physically inspired simulations excel in simulating one particular phenomenon. However, we are interested in a general algorithmic approach. The most relevant is work by Gu et al. [5] who model and measured general time-varying phenomena. Regarding texture synthesis, the most relevant is the optimization by Kwatra et al. [9] as it is the core of contemporary texture synthesis algorithms. Therefore, in this section, we describe these two works in more detail.

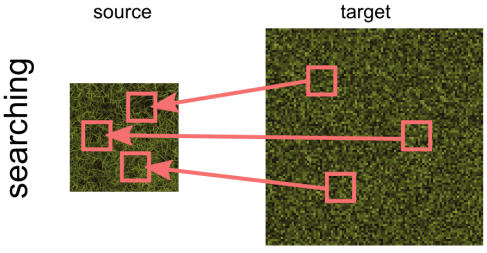


Fig. 3. An illustration of the searching step, in which we search for nearest neighbor patches in the source.

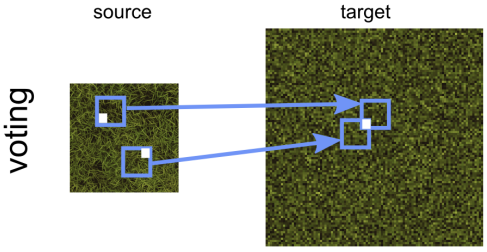


Fig. 4. An illustration of the voting step, in which we adjust values of texels in the target according to nearest neighbors of overlapping patches.

3.1. Space-Time Appearance Factorization

We recall the space-time appearance factorization (STAF) proposed by Gu et al. [5] since we use it as a basis of our work.

The measured data are fit into the combination of the Lambertian (diffuse) and simplified Torrance-Sparrow (specular) [44] BRDF model:

$$\rho(x, y, \vec{\omega_i}, \vec{\omega_o}, \lambda, t) = \frac{K_d(x, y, \lambda, t)}{\pi} + K_s(x, y, t)\rho_s(x, y, \vec{\omega_i}, \vec{\omega_o}, t), \tag{1}$$

$$\rho_s(x, y, \vec{\omega_i}, \vec{\omega_o}, t) = \frac{1}{4(\vec{\omega_i} \cdot \vec{n})(\vec{\omega_o} \cdot \vec{n})} \exp\left[-\left(\frac{\vec{\omega_h} \cdot \vec{n}}{\sigma(x, y, t)}\right)^2\right], \tag{2}$$

where (x, y) are texel spatial coordinates, $\vec{n}$ is the surface normal, $\vec{\omega_i}$ and $\vec{\omega_o}$ are incoming and outgoing directions, $\vec{\omega_h}$ is the half-angle vector such that $\omega_h = \frac{\omega_i + \omega_o}{\|\omega_i + \omega_o\|}$, λ is the wavelength, and t is time. The BRDF model consists of five parameters in total: the diffuse reflectivity (albedo) K_d (RGB color), the specular reflectivity K_s, and the surface roughness σ. Each of the BRDF parameters p is factorized into four factors:

$$p(x, y, t) = A(x, y)\phi(t') + D(x, y), \tag{3}$$

$$t' = R(x, y)t - O(x, y), \tag{4}$$

where $\phi(t')$ is the *temporal characteristic curve*, the *rate* $R(x, y)$ and the *offset* $O(x, y)$, and the time-invariant $A(x, y)$ and $D(x, y)$. Intuitively, the temporal characteristic curve is common for all texels corresponding to the particular phenomenon assuming that the parameters of all texels are similar in some sense. However, different texels differ in both time and spatial domains. The factors $R(x, y)$ and $O(x, y)$ describe how texels evolve in different locations in time. The factors $A(x, y)$ and $D(x, y)$ describe time-invariant features of the material. For example, we can change the underlying texture by modifying the $A(x, y)$ and $D(x, y)$. The temporal characteristic curve is a polynomial function:

$$\phi(t) = \sum_{k=0}^{d} a_k t^k, \tag{5}$$

where d is the degree of the polynomial function. The authors provided the publicly available database containing 26 samples of various phenomena (burning, drying, decay, and corrosion). The database contains both factorized data and fitted BRDF parameters in discrete time steps as they were measured. Raw data are not available. Most of the samples use polynomials of degree 6. The notable problem is that some of the samples have not been factorized successfully (e.g., rock drying), and hence the STAF parameters are missing. The factorization fails for most of the specular materials (e.g., copper patina) providing incorrectly a constant temporal characteristic curve.

3.2. Texture Synthesis

We briefly describe the concept and terminology of example-based texture synthesis, more details can be found in a survey by Wei et al. [45] or by Barnes and Zhang [10]. The goal is to compose resulting texture (target) from quilts (i.e., small parts) from example texture (source) without visible seams. There are two types of channels: *styling channels* and *guiding channels*. The styling channels is what we want to compute, i.e., channels of the resulting texture changing during the synthesis. The texture synthesis can be guided by additional guiding channels [11]. Unlike styling channels, guiding channels are known a priori, and they do not change during the synthesis. Both styling and guiding channels are included in the distance (see Figure 2). We will describe the texture synthesis formulated as iterative global optimization proposed by Kwatra et el. [9], which is very similar to the well-known Expectation-Maximization algorithm. Each iteration consists of two steps: *searching* (expectation) and *voting* (maximization).

Searching: The texture synthesis is based on searching for similar square patches of fixed width in the source and target, where each patch is centered around a given texel and the width of the patch is a parameter of the synthesis. The similarity between patches is defined as the sum of distances of individual texels, where each texel may have multiple channels, and the distance between texels is the sum of distances of individual channels. The distance between individual channels is simply their squared difference. In this step, for each patch in the target, we search for the nearest neighbor patch in the source, where the target patches are given by texels in the target (see Figure 3). To efficiently find the nearest neighbors, the PatchMatch algorithm [35] is commonly used.

Voting: In the previous step, we found for each texel in the target its nearest neighbor patch in the source. Each target texel overlaps with many patches of neighboring texels. In this step, we modify target texel values according to nearest neighbors of the overlapping patches. For each such patch, we look at its nearest neighbor patch and check a texel at the position corresponding the position of the target texel (see Figure 4). The value of this texel is a value that minimizes the distance for one particular patch, but there are many overlapping patches. To minimize the distance with respect to all overlapping patches, the voting step uses the mean of these values since we minimize the sum of squared differences.

4. Spatio-Temporal BRDF

In this section, we describe our proposed model of spatio-temporal BRDFs and the data that we used for our experiments.

4.1. Modeling

For representing the static BRDF of the input material, we use a combination of the Lambertian (diffuse) and Torrance-Sparrow (specular) [44] BRDF models in a similar manner as Gu et al. [5]:

$$\rho^\star(x, y, \vec{\omega_i}, \vec{\omega_o}, \lambda, t) = \frac{K_d^\star(x, y, \lambda, t)}{\pi} + K_s^\star(x, y, \lambda, t)\rho_s^\star(x, y, \vec{\omega_i}, \vec{\omega_o}, \lambda, t), \tag{6}$$

$$\rho_s^\star(x, y, \vec{\omega_i}, \vec{\omega_o}, \lambda, t) = \frac{F(\vec{\omega_i}, \vec{\omega_h}, \lambda) G_{\sigma^\star(x,y,t)}(\vec{\omega_i}, \vec{\omega_o}) D_{\sigma^\star(x,y,t)}(\vec{\omega_h})}{4(\vec{\omega_i} \cdot \vec{n})(\vec{\omega_o} \cdot \vec{n})}, \tag{7}$$

where F is the Fresnel term, $G_{\sigma^\star(x,y,t)}$ is a shadowing-masking function, and $D_{\sigma^\star(x,y,t)}$ is a normal distribution. We use seven BRDF parameters in total: the diffuse reflectivity $K_d^\star$ (RGB color), the specular reflectivity $K_s^\star$ (RGB color), and the surface roughness $\sigma^\star$. In particular, we used the Smith shadowing-masking function [46, 47] and the GGX normal distribution [48] both induced by the surface roughness $\sigma^\star$.

Table 1. The BRDF model is represented by seven parameters: $K_d^\star$ (RGB), $K_s^\star$ (RGB), and $\sigma^\star$. Each BRDF parameter is represented by a polynomial with $d+1$ coefficients. In total, there are $7\times(d+1)$ coefficients. In the case of scalar representation of K_S^* (as in the STAF database), there are only $5\times(d+1)$ coefficients.

$K_d^\star$			$K_s^\star$			$\sigma^\star$
R	G	B	R	G	B	-
a_0	a_0	a_0	a_0	a_0	a_0	a_0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
a_d	a_d	a_d	a_d	a_d	a_d	a_d

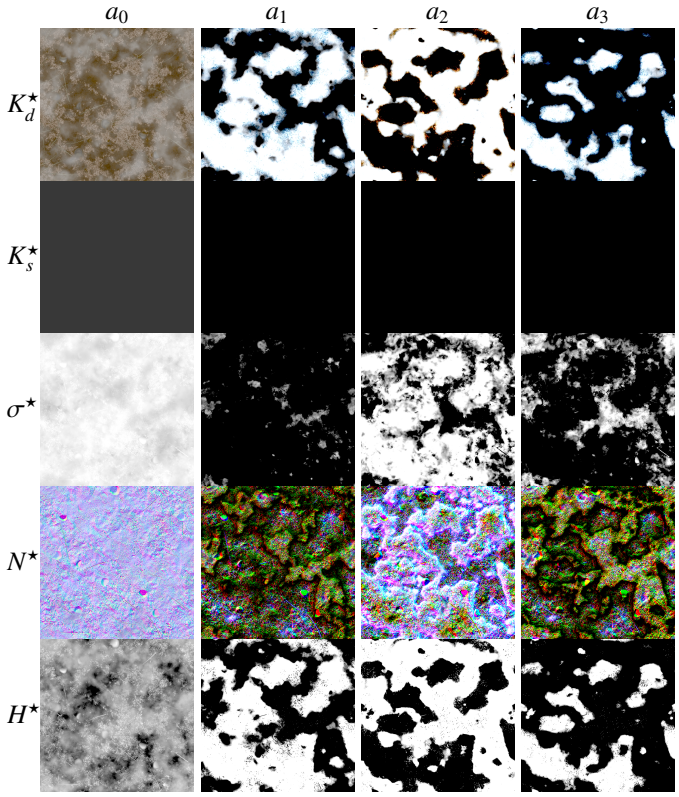


Fig. 5. Data representation of snowy ground with $d=3$: diffuse reflectivity (albedo) $K_d^\star$, specular reflectivity $K_s^\star$, surface roughness $\sigma^\star$, surface normal $N^\star$, and height $H^\star$. Left pictures represent the absolute coefficients which correspond to the initial state in $t=0$. Notice that the specular reflectivity is constant in time as all coefficients except a_0 are zero.

Gu et al.[5] proposed a factorization of each time-varying BRDF parameter into four factors for each texel and the common temporal characteristic curve. We can easily show that transformations in Equations 3 and 4 of the temporal characteristic curve result in another polynomial function with the same degree as the temporal characteristic curve. Unlike Gu et al.[5], we use a more straightforward approach. We fit the data of each texel directly into polynomial functions:

$$p^\star(x,y,t) = \sum_{k=0}^{d} a_k(x,y)t^k, \tag{8}$$

where $a_k(x,y)$ are coefficients of the polynomial function and d is the degree of the polynomial function. Thus, we use $d+1$ coefficients to represent each of the seven BRDF parameters (see Table 1). Albeit simpler, our representation has a crucial advantage over the STAF representation for the voting step of example-based texture synthesis as we discuss in Section 5.2.

4.2. Data

The STAF database [5] is the only publicly available database of the spatio-temporal BRDF. We use the available BRDF parameters in discrete time steps for fitting into the polynomials. We require that values of the BRDF parameters are in range $[0,1]$, which is true for the diffuse reflectivity and surface roughness. However, this does not hold for the specular reflectivity as it is coupled with the Gaussian distribution of normals. Normalization of the BRDF parameter is crucial for texture synthesis. If the BRDF parameters had different ranges, then the synthesis would be driven by parameters with higher values and others would be omitted.

Using the equations 1 and 2, we can compute the decoupled specular reflectivity $K_s^{i\star}(x,y)$ in time step i as:

$$\begin{aligned} K_s^{i\star}(x,y) &= K_s^i(x,y)\int_{\vec{\omega_h}\in\Omega} \exp\left[-\left(\frac{\vec{\omega_h}\cdot\vec{n}}{\sigma^i(x,y)}\right)^2\right](\vec{\omega_h}\cdot\vec{n})d\vec{\omega_h}, \\ &= K_s^i(x,y)(\sigma^i(x,y))^2\pi(1-\exp(-1/(\sigma^i(x,y))^2)), \end{aligned} \tag{9}$$

where $\sigma^i(x,y)$ is the surface roughness in time step i and $K_s^i(x,y)$ is the original specular reflectivity in time step i. We used the fact that the integration over the hemisphere of the normal distribution must be equal to one.

The STAF database does not include materials with wavelength dependent specular reflection. Additionally, the database covers only a limited number of measured materials. To extend the range of materials for our experiments, we also used procedurally generated materials by Substance Designer [49]. We changed one of the procedural parameters (such as *wetness*) continuously influencing most of the BRDF parameters in a non-trivial manner. For each such material, we exported 50 samples uniformly covering the temporal domain. While converting a procedural material model to our representation loses the expression possibilities of the original model, it allows us to use a common framework particularly for the transfer of time-varying phenomena.

4.3. Additional Data Channels

The visual quality of results generated by texture synthesis can be improved by explicitly using geometric information [21, 23]. In our case, we can use additional data channels such as normal maps, height maps, or ambient occlusion to account for geometric properties. Unfortunately, the STAF database does not contain such additional information. Nonetheless, some materials generated by the Substance Designer [49] contain both normal maps and height maps. We simply use these data as additional channels for texture synthesis in the same manner as we use, for example, the surface roughness. We assume that the additional data channels are also normalized to range $[0, 1]$. Figure 5 shows an example of channels that can be used for texture synthesis.

5. Synthesis of Spatio-Temporal BRDFs

Two crucial components of the texture synthesis process are the searching and voting steps outlined in Section 3.2. In the static case, one constant value corresponds to one channel. In our case, $d + 1$ coefficients correspond to one channel, which is a polynomial function of time. We have to carefully modify both steps of the texture synthesis using our representation.

5.1. Searching Step

In the searching step, we search for the nearest neighbor patches, and we have to measure the distance between channels. In the standard case of synthesizing static textures, the distance between channels is computed as squared differences of channel values. In our case, the channels correspond to BRDF parameters, which are functions of time instead of constant values.

Evaluating the distance between general BRDFs is an open problem. Since all our materials are expressed using a common BRDF model, we use a simplified BRDF distance evaluation by computing distances of normalized BRDF parameters. For the sake of clarity, we omit spatial coordinates of texels since they are not relevant for computing distance, and we refer to individual texels by a single index:

$$p_i(t) = \sum_{k=0}^{d} a_{i,k} t^k. \tag{10}$$

To measure the distance between functions, we use the L^2-norm, i.e., integration of squared differences over the temporal domain:

$$d\left(p_1(t), p_2(t)\right) = \int_0^1 \left[p_1(t) - p_2(t)\right]^2 dt. \tag{11}$$

Since the subtracting and squaring of polynomials results in a polynomial function, the integrand is a polynomial function, which can be analytically integrated. Subtracting two polynomials is trivial as it is simply subtracting the coefficients:

$$\begin{aligned} q(t) &= p_1(t) - p_2(t), \\ &= \sum_{k=0}^{d} a_{1,k} t^k - \sum_{k=0}^{d} a_{2,k} t^k, \\ &= \sum_{k=0}^{d} (a_{1,k} - a_{2,k}) t^k, \\ &= \sum_{k=0}^{d} b_k t^k. \end{aligned} \tag{12}$$

We express coefficients of the squared polynomials:

$$\begin{aligned} \left[q(t)\right]^2 &= \left[\sum_{k=0}^{d} b_k t^k\right]^2, \\ &= \sum_{k=0}^{2d} \left[\sum_{j=\max\{0,k-d\}}^{\min\{k,d\}} b_j b_{k-j}\right] t^k, \\ &= \sum_{k=0}^{2d} c_k t^k. \end{aligned} \tag{13}$$

Finally, we can evaluate the distance function since the integrand is a polynomial function using basic calculus:

$$\begin{aligned} d\left(p_1(t), p_2(t)\right) &= \int_0^1 \left[p_1(t) - p_2(t)\right]^2 dt, \\ &= \int_0^1 \left[q(t)\right]^2 dt, \\ &= \int_0^1 \sum_{k=0}^{2d} c_k t^k dt, \\ &= \sum_{k=0}^{2d} \frac{c_k}{k+1}. \end{aligned} \tag{14}$$

Since we integrate over the interval $[0, 1]$, the distance function also expresses the average squared difference. Enrique et al. [40] considered general functions and proposed a distance function defined as a sum of squared differences in different time steps:

$$d\left(p_1(t), p_2(t)\right) = \sum_{i=1}^{N} \left[p_1(t_i) - p_2(t_i)\right]^2, \tag{15}$$

which can be seen as a numerical solution of the integral in Equation 11. In our context, the analytical approach leads to a precise and faster solution.

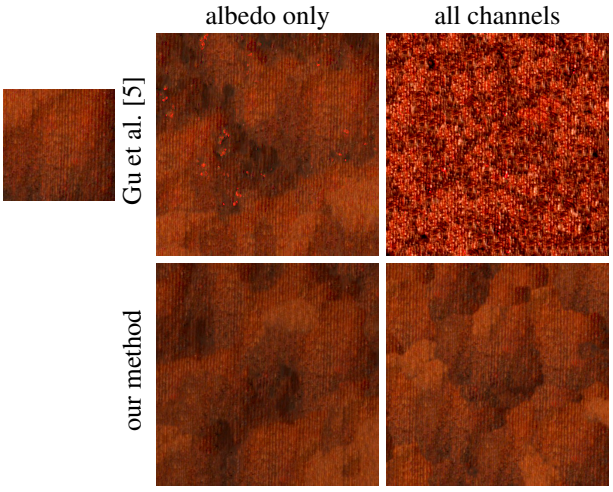


Fig. 6. Comparison of enlargement of wood drying using the STAF representation proposed by Gu et al. [5] (top) and our polynomial representation (bottom). The left column shows setup where all channel weights except albedo are set to zero. Both methods provide smooth albedo channel texture, but the STAF representation leads to noticable bright red spots caused by incorrect averaging of factors during the voting step. The right column shows setup with unit weights for all channels. The STAF representation fails due to the unnormalized surface rougness. For our representation, we can observe only minor scaly look of the albedo that results from enforcng continuity in the other channels.

Our BRDF parametrization is designed for a uniform range of all parameters. This reduces the need for weighting the channels. In fact, we used unit weights for all our results (unless stated otherwise), which works reasonably well for most of the cases. Weighting the channels during the search can be used to tweak the behavior of the method for some specific cases as we discuss in section 7.4.

5.2. Voting Step

A crucial advantage of our representation over the STAF representation proposed by Gu et al. [5] is its compatibility with the voting step of texture synthesis. In the voting step, we use mean values of corresponding polynomial coefficients. These values correctly represent mean polynomials, which does not hold for the STAF representation. For the STAF representation, the voting step can produce noticeable errors since the mean values of factors ($R(x, y)$ and $O(x, y)$) do not define the mean of the corresponding polynomials (see Figure 6).

Theoretically, we could reconstruct polynomials from the STAF factors, and then take a mean of these polynomials. However, it is difficult to recover the STAF factors of the resulting polynomial, which leads to the system of polynomial equations with no analytic solution in general. In fact, this requires repeated per voting step factorization with a given temporal characteristic curve.

6. Applications

We describe two applications of the synthesis using the proposed spatio-temporal BRDF representation.

6.1. Enlargement

The basic task of the example-based texture synthesis is enlargement of the texture, i.e., synthesize a larger texture from small example. Gu et al. [5] proposed a simple technique for enlargement based on synthesizing texture in the initial temporal frame. Using the distance function based on the integration over the temporal domain, we can naturally employ algorithms of the traditional texture synthesis to enlarge a given sample. We use $7 \times (d + 1)$ styling channels corresponding to the seven channels, and each channel is a polynomial function defined by $d + 1$ coefficients.

6.2. Transfer

We propose a novel technique to transfer a time-varying phenomenon using guided texture synthesis. Similarly to the enlargement, we use the distance function based on the integration over the temporal domain and the same styling channels. We use an additional guiding channel computed from diffuse reflectivity. The source guide corresponds to the luminance of a selected time sample of the input time-varying material; the target guide corresponds to the luminance of the static target texture. The time sample is selected in a way that it semantically corresponds to the content of the target texture. For example, for transferring floor drying to a texture of another dry floor, we select a sample from the later stage of the time-varying phenomenon. The values of the guide images might be of different scales. Thus, we use histogram equalization to make the synthesis more robust. While we could theoretically apply the transfer to any material, the results will only be meaningful for semantically similar materials.

As the last step, we have to adjust the polynomial coefficients to enforce the equality of the selected reference time samples. We scale coefficients of our polynomials:

$$a^{out}_{i,k} \leftarrow a^{in}_{i,k} \frac{p^{tex}_i}{p^{in}_i(t_0)}, \tag{16}$$

where $a^{in}_{i,k}$ are unmodified coefficients defining the original value of channel $p^{in}_i(t)$, $a^{out}_{i,k}$ are new modified coefficients, p^{tex}_i is a static texture and t_0 is a moment in which the modified time-varying channel using $a^{out}_{i,k}$ will be equal to the static texture (see Figure 7).

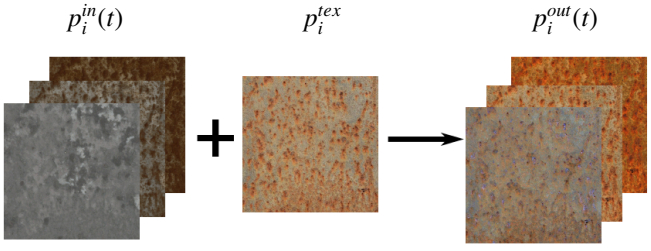


Fig. 7. An example of changing an underlying texture of time-varying material. We produce new material $p^{out}_i(t)$ from a given time-varying material $p^{in}_i(t)$ and a static texture p^{tex}_i.

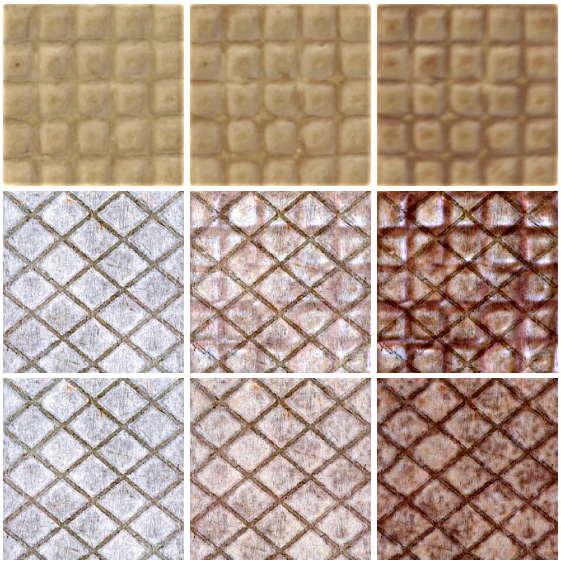


Fig. 8. Comparison of transferring of waffled toasting (top) as proposed by Gu et al. [5] (middle) and our transfer employing texture synthesis (bottom). Notice that our method respects the input texture pattern through time=, while the method of Gu et al. [5] overlays the pattern given by the transferred temporal phenomenon in later stages.

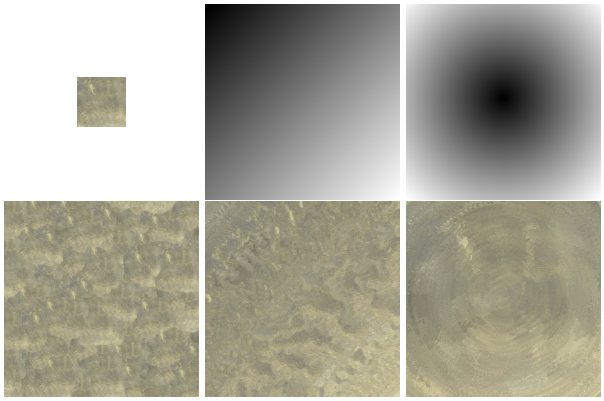


Fig. 9. Comparison of enlargement of copper patina (top-left) for different guiding channels: no guiding channels (bottom-left), linear gradient (middle), and radial gradient (right). Notice how using guiding channels suppresses scaly artifacts.

The scaling corresponds to the method used by Gu et al. [5], who used scaling of $A(x, y)$ and $D(x, y)$ factors. Note that there are other ways how to modify the coefficients to ensure equality. We can, for example, modify only the constant coefficient; however, we might get out of the interval $[0, 1]$, which is not desirable. Scaling the coefficients is simple enough and provides reasonable results.

Our method provides better results than simple scaling proposed by Gu et al. [5] as it is not prone to propagating a structure of the exemplar time-varying material to the synthesized result (see Figure 8). The core of the improvement in our method lies in the simultaneous use of the guiding channel representing the transferred phenomenon together with all other channels in a common optimization problem. While we could use the guiding channel also for the representation of Gu et al.[5], the artifacts caused by the voting step of the synthesis discussed in Section 5.2 still appear and make that approach hardly usable.

7. Results

We evaluated our representation of spatio-temporal BRDFs and the associated applications using 20 materials from the STAF database [5] and 10 materials generated by the Substance Designer [49]. For the polynomial fitting, we used MATLAB. For data manipulation, we used the OpenCV library. For texture synthesis, we used the EbSynth library [50]. The library is GPU-based supporting the guiding channels. We modified the library to support the distance function based on the integration over the time domain. To demonstrate the usage of our method in offline rendering, we implemented a BSDF plugin for Mitsuba [51] supporting our spatio-temporal BRDF. To demonstrate the usage of our method in real-time rendering applications, we implemented a specialized shader for Unity. The results were evaluated on a PC with Intel Core i9-9900 3.6 GHz, 32 GB RAM and NVIDIA RTX 2080 Ti.

7.1. Enlargement and Transfer

We enlarged the fitted materials using our method. The resolution of enlarged materials is four times the resolution of the input materials (i.e., 16× more texels). To visually evaluate synthesized materials, we used the optimized surface model designed for BRDF evaluation [52], that allows us to better understand the surface reflectance properties from a single image. We mapped synthesized materials on this surface model and rendered them in Mitsuba [51] (see Figure 14). We also rendered synthesized materials within more complex scenes. We map an enlarged material on one of the objects in the scene (see Figure 15). The rendering times of the reported images in Mitsuba were in range 22 to 78 seconds. The time for evaluation of our polynomial representation is almost negligible.

Similarly, we transferred a time-varying phenomenon to a static material of one of the objects in the scene. Transfer depends on a transferred phenomenon, a target material, and also on t_0. The transferred phenomenon and the target material should semantically correspond, otherwise, we could get unpredictable results. An example of a successful transfer of a time-varying phenomenon to a static material are shown in Figure 16.

Times for the synthesis range from a few seconds to a couple of minutes depending on the resolution of the source and the target, and the polynomial degree. For example, enlarging the 512×512 material to 2048×2048 takes 43 seconds with $d = 3$, 70 seconds with $d = 4$, and 101 seconds with $d = 5$. The standard texture synthesis using the same implementation (EbSynth) takes about 1 or 2 seconds while using only three channels (RGB) with one byte per channel. On the contrary, we use 28 to 42 parameters in the floating-point representation. The increase in running time is thus roughly proportional to the increase in the data requirements (taking into account the floating-point representation, the number of BRDF parameters, and the number of polynomial coefficients).

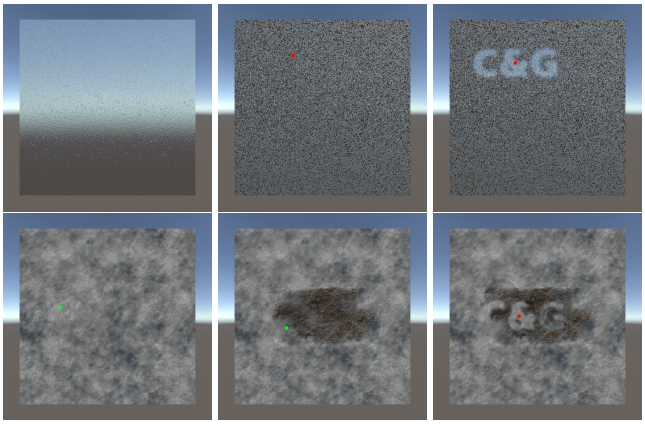


Fig. 10. A real-time application using our polynomial representation implemented in Unity. We use a brush tool and the C&G stamp to interactively modify the local time of the phenomena: aged chrome (top) and snowy ground (bottom). In both cases, we use also time-varying normal and displacement maps. All information is encoded in spatio-temporal BRDFs and no texture synthesis is needed at runtime.

7.2. Real-Time Rendering

The actual evaluation of spatio-temporal BRDF in a renderer is straightforward and induces only negligible performance penalty. We demonstrate this by implementing real-time rendering using our spatio-temporal BRDFs in the Unity game engine. The implementation extends Unity's standard PBR shaders. Each coefficient of every channel is stored as a separate texture. These textures are passed to the fragment shader where polynomials are reconstructed producing the final output. Thanks to our representations, we can also easily use mipmaps as an interpolation of polynomial coefficients provides a correct interpolation of corresponding polynomial functions.

We are also able to make local changes to the result via brush and stamp tools. This is implemented using a simple dynamically modified texture of scalar values. This texture is used to locally modify the time for retrieving the material parameters (see Figure 10).

7.3. Comparison with Other Methods

Context-aware textures: Lu et al. [23] proposed a technique for synthesizing time-varying phenomena based on the geometric context. First, they measured time-varying phenomenon on an object with known geometric properties. Then, the measured data are transferred to a new synthetic object using geometric properties as texture context in a similar way as we use the guiding channels. Lu et al. used a modified version of the texture synthesis algorithms proposed by Turk [53] and Wei and Levoy [54], while their context-aware transfer uses the approach of Hertzmann et al. [36] and Zhang et al. [55]. The paper of Lu et al. concentrates on the thorough analysis of the correlation of the measured phenomenon and geometric context including measurements and validation of the results. In contrast, our work presents a more general framework for representing and synthesizing time-varying materials. We use a polynomial representation of parametric spatio-temporal BRDFs, unlike context-aware textures, which use only albedo in different time steps. Note that also the context-aware transfer is different transfer than ours. We transfer time-varying phenomena to a static material guided by optical properties of the materials; unlike the context-aware transfer, which is guided by geometric properties. Theoretically, we could use our method in the same manner as context-aware textures if geometric properties were present in the input data. Nonetheless, the materials in the STAF database, that we have access to, were only measured on a flat surface, and the geometric properties are missing. We conducted an experiment to demonstrate this possibility. We extracted the context information either from the diffuse texture (see Figure 12) or from ambient occlusion (see Figure 13), that we further use to guide the synthesis process.

Appearance Manifold: Wang et al. [8] proposed a technique for synthesizing time-varying BRDFs from a single static measurement, exploiting the fact that the data exhibit both spatial and temporal variation of the captured phenomenon (e.g., non-uniform rusting). The core idea is to construct an appearance manifold in high-dimensional space of the BRDF parameters, and then perform the synthesis by interpolation on this appearance manifold. In comparison with our work, this approach uses only a single static measurement of BRDF. A crucial advantage is that capturing a single BRDF measurement is much simpler than capturing time-varying BRDF, which is a tedious process. However, there are also several disadvantages. First, the synthesis is done per-frame, while our method synthesizes the resulting material in a single pass thanks to the polynomial representation. Furthermore, it is difficult to capture all temporal variations of more complex time-varying phenomena in a single measurement, especially for rapid variations spanning over a longer time interval (e.g., waffle toasting). Although the synthesized sequences are temporally coherent, the BRDFs parameters are interpolated from texels in different spatial locations, which is generally incorrect. Finally, the manifold requires a user annotation of the most and least weathered part of the manifold.

Other methods: There are several other methods related to our work that do not explicitly address synthesizing spatio-temporal BRDFs. Mazlov et al. [43] used neural networks for texture synthesis and transfer of static spatially varying BRDFs. This approach shares both pros and cons of neural texture synthesis. It is not obvious how to extend this method for spatio-temporal BRDFs. It would be interesting to combine a neural synthesis proposed by Zhou et al. [42] with our polynomial representation to synthesize non-stationary materials. Sun et al. [22] measured and modeled more complex time-varying phenomena such as dust accumulation or drying paints. As these BRDFs are not spatially varying, there is no need for the synthesis. Langenbucher et al. [7] studied time-varying BTF, such as metal rusting and car paint, focusing on direct interpolation between measurements in discreet time steps, not taking synthesis into account.

7.4. Discussion and Limitations

Polynomial fitting: We fitted the data into polynomials with $d \in \{3, 4, 5\}$. As a reference method, we use STAF with one common polynomial characteristic curve of degree 6 and four

factors. Comparison of our representation and STAF in terms of the RMS error is summarized in Table 2 and Figure 17.

Generally, the polynomial representation excels if the spatial variation is dominating as each texel is fitted independently, while STAF fits better cases with higher temporal variation thanks to the higher polynomial degree of the characteristic curve. In such cases, we can also increase the polynomial degree to improve the fitting at a cost higher memory storage. Therefore, our method is scalable unlike STAF, since there is no way how to improve the accuracy of STAF if there is significant spatial variation. Generally, the RMS error decreases with a higher polynomial degree (the average overall RMS error 0.03 for $d = 3$, 0.023 for $d = 4$, and 0.017 for $d = 5$). On the other hand, polynomials with a higher degree are prone to be overfit. The major issue of STAF is its robustness. The factorization relies on non-linear optimization that fails in many cases especially for $K_s^\star$ (the average overall RMS error 0.052).

When using $d = 3$, we represent each channel by four coefficients which is the same number as the number of factors. In such a case, STAF fits slightly better $K_d^\star$ and surprisingly much better $\sigma^\star$; however, it significantly worse for $K_s^\star$ because for the robustness issue (e.g., Wood Drying). Nonetheless, polynomials with $d = 3$ are sufficient to represent all materials from the STAF database without visible artifacts, although in some cases, polynomials do not fit sufficiently the input, which may result in visible difference from the ground truth (see Figure 11).

Figure 17 reveals that some of the phenomena exhibit noise in the input data of certain channels (banana decay $\sigma^\star$, waffle toasting $K_s^\star, \sigma^\star$). We suspect that this is due to the surface deformation during the measurement of the phenomenon. If these deformations would be accounted for during the acquisition, they could be considered in the fitting phase by using image warping. The information on deformation could also be easily used in the synthesis process. Since we do not have the geometry deformation data at our disposal, we leave the verification of this hypothesis for future work.

Efficiency of evaluation and fitting: Regarding the efficiency of the BRDF parameters evaluation, our method uses only d additions and multiplications ($d \in \{3, 4, 5\}$) using the Horner algorithm to evaluate the polynomial function, while STAF uses $d + 2$ additions and multiplications ($d = 6$), which makes our approach more efficient. The fitting times range from 24 to 362 seconds depending on the spatial and temporal resolution, and the polynomial degree; STAF takes a few minutes according to the authors not showing any particular numbers. However, we believe that fitting of polynomials is much simpler than the iterative non-linear factorization, and thus it is computationally cheaper.

Memory consumption: The memory requirements for different representations depend on the texture resolution and the polynomial degree (see Table 1). In particular the texture of 156×156 (the lowest resolution texture in the STAF database) texels requires 1.9 MB for $d = 3$, 2.4 MB for $d = 4$, 2.9 MB for $d = 5$, and 1.9 MB for the STAF representation. The texture of 512×512 texels (the highest resolution texture in STAF database) requires 20.9 MB for $d = 3$, 26.2 MB for $d = 4$, 31.4 MB for $d = 5$, and 20.9 MB for the STAF representation. The reported numbers assume four-byte floating point representation and no texture compression involved.

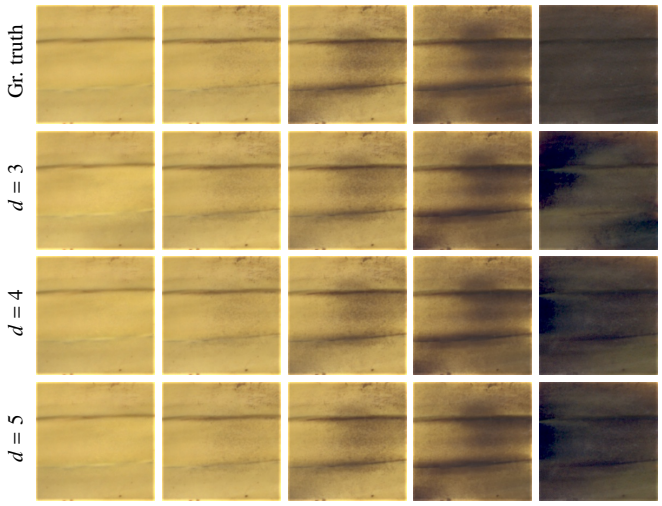


Fig. 11. Comparison of fitting data of banana decay with various number of polynomial degree.

Other basis functions: Our approach uses polynomial basis with configurable degree polynomials that mostly yields good visual fidelity. Other basis functions are possible though, such as wavelet bases or piece-wise polynomial representations using splines. These approaches might better capture sharp changes in the material parameters, but in general they would make the synthesis more complicated as they would typically require more parameters and more complex distance function evaluation. Using other bases is, however, an interesting avenue for future work.

Synthesis using STAF: Example-based synthesis using the STAF representation generally produces artifacts as we discussed in Section 5.2. Moreover, reconstructing polynomial functions from STAF needed during the search step slows down the synthesis significantly (about 20× slowdown).

Synthesis using raw data: Another option would be to use raw temporal data for both representations of spatio-temporal BRDFs and their synthesis. Computational times grow significantly with an increasing number of parameters, even when using state-of-the-art GPU-based texture synthesis implementations. Therefore using raw data would not only cause memory overhead, but it would slow down the synthesis beyond any practical use.

Scaly artifacts: Some of the synthesized materials exhibit *scaly* artifacts. Synthesis algorithms based on the optimization proposed by Kwatra et al. [9] cope well with stationary textures. However, if the exemplar is non-stationary, then these artifacts may occur. This happens when the optimization arrives at local minimum that consists of highly coherent texture regions with visible boundaries between them. This is a general issue of example-based texture synthesis that is rather orthogonal to our method [42]. A possible solution to reduce these artifacts is using additional manually-created guiding channels to preserve changes across the spatial domain as shown in Figure 9. In this

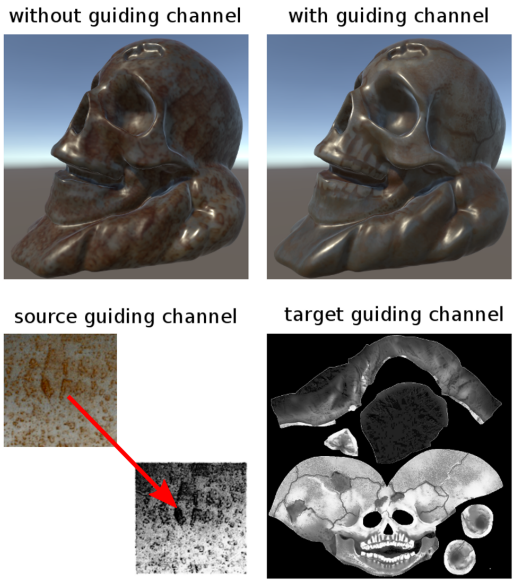


Fig. 12. Comparison of spatio-temporal BRDF transfer without guiding channels (top-left) and context-aware enlargement guided by an additional channel (top-right). Source guiding channnel extracted from the steel rusting sequence at $t_0 = 0.5$ (bottom-left). Target guiding channel extracted from the skull texture (bottom-right). Notice how the rusting mimics teeth and the seams on the skull. Technically, this is implemented as the transfer to the Skull texture without application of Equation 16.

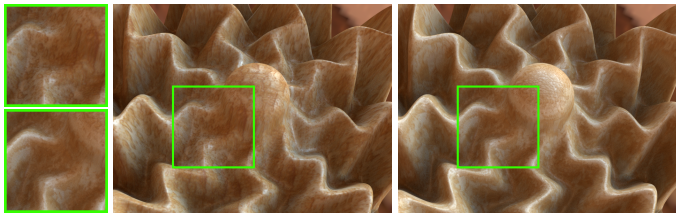


Fig. 13. The surface with steel rusting without guiding (left, top) and with guiding by ambient occlusion (right, bottom). Notice how the corrosion more naturally respects the geometric properties of the surface.

case, the target guiding channel is just a rescaled version of the source guiding channel.

8. Conclusion and Future Work

We studied the synthesis of spatio-temporal BRDFs employing example-based texture synthesis. We proposed a suitable representation of time-varying channels, which is compatible with contemporary example-based texture synthesis. To measure the distance between time-varying channels, we proposed a distance function defined as integration over the temporal domain taking into account time variation. We presented two applications of our extension for synthesizing spatio-temporal BRDFs. First, we synthesized large materials from smaller examples. Second, we transferred example time-varying phenomena to given static materials. We evaluated synthesized materials in the context of realistic rendering using Mitsuba renderer and in the context of real-time rendering using Unity. The results indicate great flexibility of the proposed approach and high accuracy of the temporal phenomena representation even using a low degree polynomial.

In the future, we would like to further investigate the distance evaluation using a non-linear perceptually motivated transformation of BRDF parameters. Another topic worth investigating is the compression of the proposed representation, particularly for the context of real-time rendering applications. The available measured spatio-temporal BRDF data are very rare. Therefore, we would like to measure and publish data with a greater variety of materials.

Acknowledgements

This research was supported by the Czech Science Foundation under project GA18-20374S, by the Research Center for Informatics No. CZ.02.1.01/0.0/0.0/16_019/0000765, Toyota Motor Europe, and by NII International Internship Program. We thank Daniel Sýkora, Ondřej Jamriška, and David Sedláček for fruitful discussions and valuable comments. We would like to also thank Benedict Bitterli for providing the test scenes [56].

References

[1] Dorsey, J, Edelman, A, Jensen, HW, Legakis, J, Pedersen, HK. Modeling and Rendering of Weathered Stone. In: Proceedings of Computer Graphics and Interactive Techniques. 1999, p. 225–234.

[2] Paquette, E, Poulin, P, Drettakis, G. The Simulation of Paint Cracking and Peeling. In: Proceedings of Graphics Interface. 2002, p. 59–68.

[3] Bosch, C, Pueyo, X, Mérillou, S, Ghazanfarpour, D. A Physically-Based Model for Rendering Realistic Scratches. Computer Graphics Forum 2004;23(3):361–370.

[4] Bosch, C, Laffont, PY, Rushmeier, H, Dorsey, J, Drettakis, G. Image-Guided Weathering: A New Approach Applied to Flow Phenomena. ACM Transactions on Graphics 2011;30(3).

[5] Gu, J, Tu, CI, Ramamoorthi, R, Belhumeur, P, Matusik, W, Nayar, S. Time-varying Surface Appearance: Acquisition, Modeling and Rendering. ACM Transactions on Graphics 2006;25(3):762–771.

[6] Lu, J, Georghiades, AS, Rushmeier, H, Dorsey, J, Xu, C. Synthesis of Material Drying History: Phenomenon Modeling, Transferring and Rendering. In: Eurographics Workshop on Natural Phenomena. 2005,.

[7] Langenbucher, T, Merzbach, S, Möller, D, Ochmann, S, Vock, R, Warnecke, W, et al. Time-Varying BTFs. In: Central European Seminar on Computer Graphics for Students. 2010,.

[8] Wang, J, Tong, X, Lin, S, Pan, M, Wang, C, Bao, H, et al. Appearance Manifolds for Modeling Time-variant Appearance of Materials. ACM Transactions on Graphics 2006;25(3):754–761.

[9] Kwatra, V, Essa, I, Bobick, A, Kwatra, N. Texture Optimization for Example-based Synthesis. ACM Transactions on Graphics 2005;24(3):795–802.

[10] Barnes, C, Zhang, FL. A Survey of the State-of-the-Art in Patch-Based Synthesis. Computational Visual Media 2016;:1–18.

[11] Fišer, J, Jamriška, O, Lukáč, M, Shechtman, E, Asente, P, Lu, J, et al. StyLit: Illumination-guided Example-based Stylization of 3D Renderings. ACM Transactions on Graphics 2016;35(4):92:1–92:11.

[12] Dorsey, J, Hanrahan, P. Modeling and Rendering of Metallic Patinas. In: Proceedings of Computer Graphics and Interactive Techniques. 1996, p. 387–396.

[13] Dorsey, J, Pedersen, HK, Hanrahan, P. Flow and Changes in Appearance. In: Proceedings of Computer Graphics and Interactive Techniques. 1996, p. 411–420.

[14] Xue, S, Dorsey, J, Rushmeier, H. Stone Weathering in a Photograph. In: Proceedings of Eurographics Symposium on Rendering. 2011, p. 1189–1196.

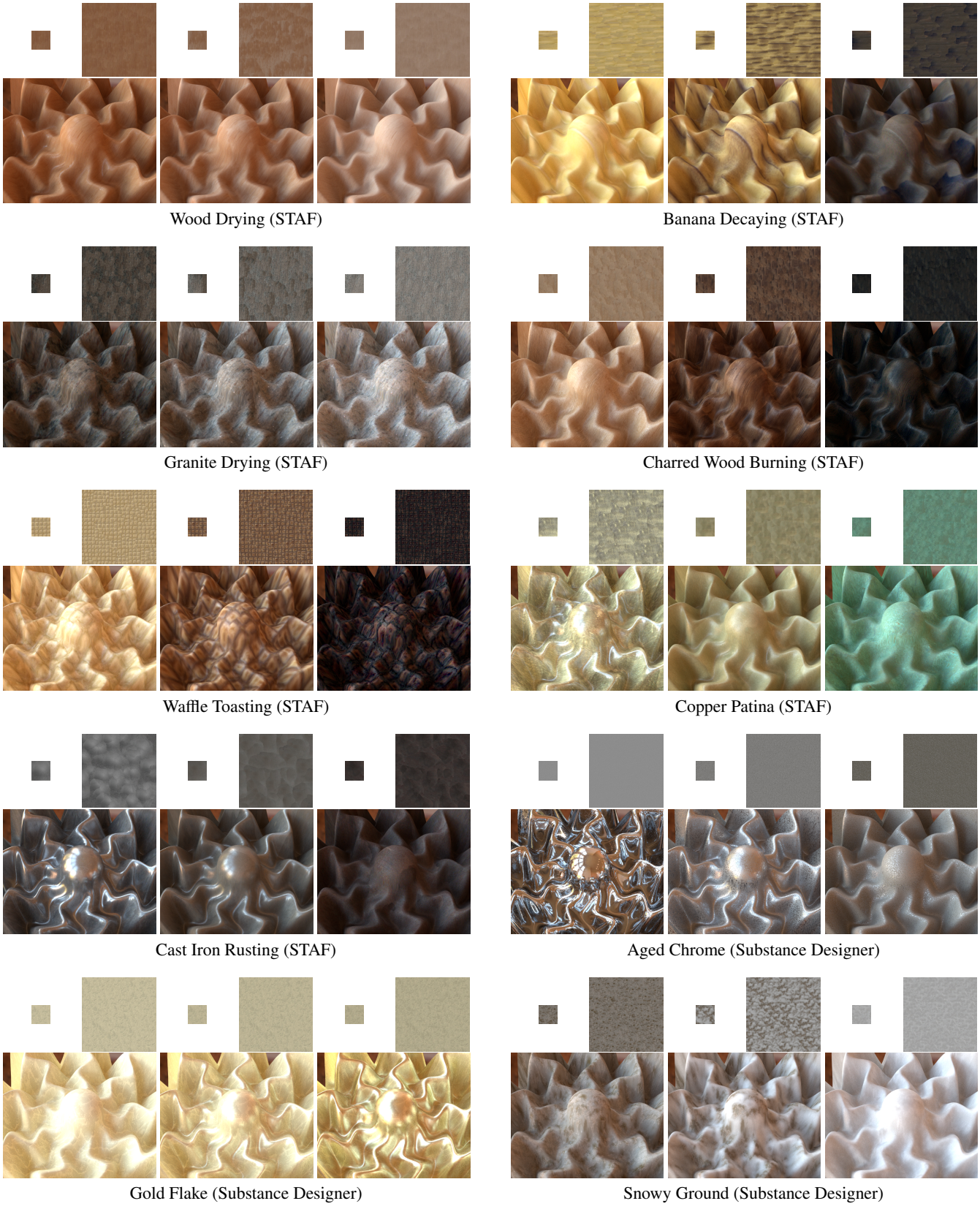


Fig. 14. Results of the enlargement of the STAF materials [5] and materials generated by Substance Designer [49]. For each frame, we show the input material (top-left), the enlarged material (top-right), and the rendering of the enlarged material (bottom).

Fig. 15. Results of the enlargement for two scenes: the bathroom scene with chrome aging mapped on the tap (top) and the kitchen scene with steel rusting mapped on the bigger pot (bottom).

Fig. 16. An example of the transfer of banana decay to a different static banana texture in the white room scene.

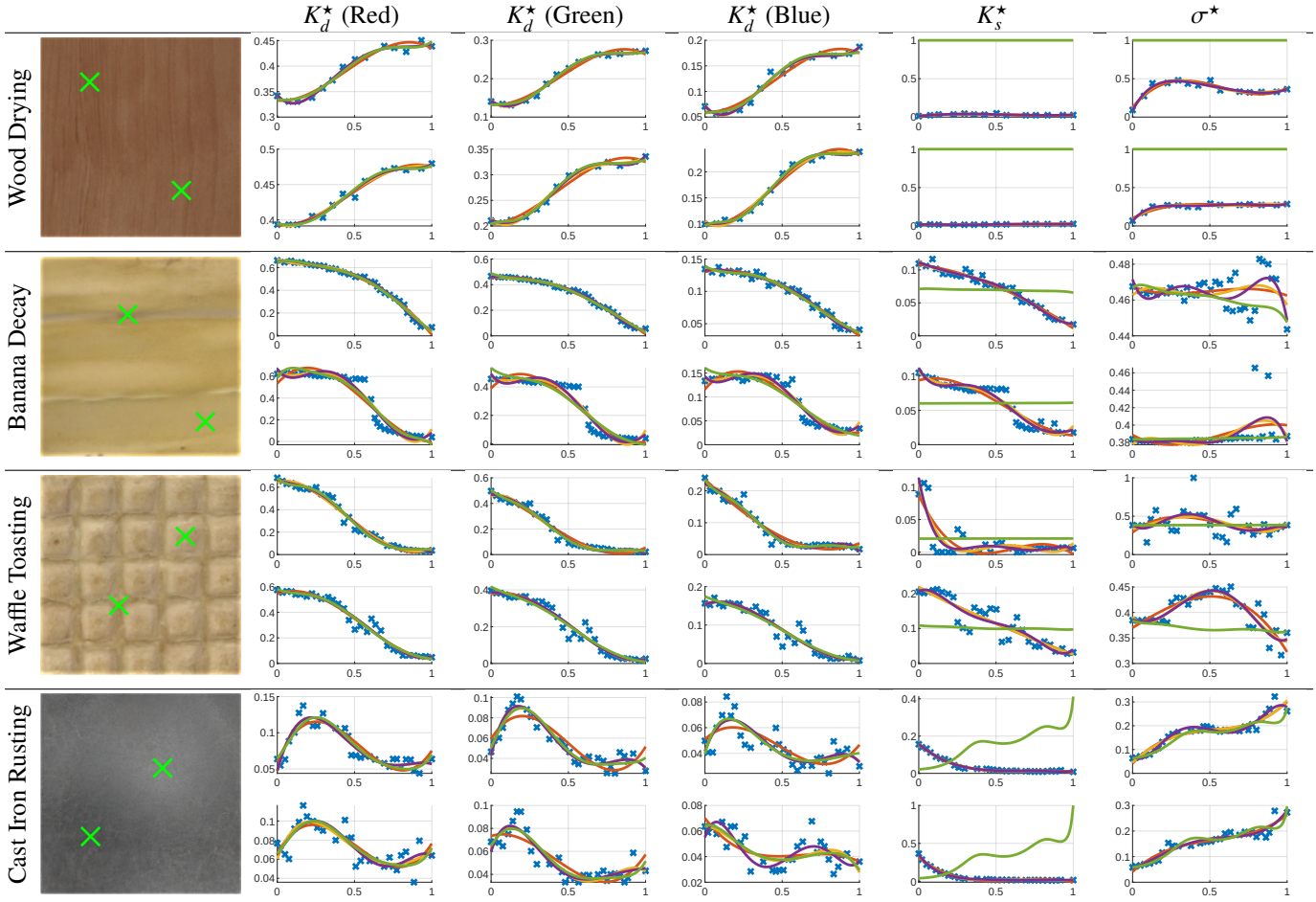


Fig. 17. Fitting BRDF parameters of two texels (green crosses) into polynomials: fitted BRDF parameters (blue crosses), polynomials with $d = 3$ (orange curves), polynomials with $d = 4$ (yellow curves), polynomials with $d = 5$ (purple curves), and STAF (green curves).

Table 2. The RMS error across all temporal frames and spatial locations (spatial and temporal resolution in the top-left cell) for our polynomial representation ($d \in \{3, 4, 5\}$) and STAF. We report times for fitting the BRDF parameters. Note that times for STAF are not available.

Wood Burning		$512^2 \times 10$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.024	0.023	0.014	0.044	0.080	-
		$d = 3$	0.029	0.024	0.015	0.026	0.052	252
		$d = 4$	0.010	0.008	0.008	0.015	0.031	339
		$d = 5$	0.005	0.004	0.005	0.008	0.018	362

Light Wood Dry.		$300^2 \times 34$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.012	0.009	0.009	0.021	0.045	-
		$d = 3$	0.014	0.016	0.015	0.016	0.068	89
		$d = 4$	0.009	0.008	0.009	0.013	0.060	121
		$d = 5$	0.008	0.007	0.008	0.011	0.051	121

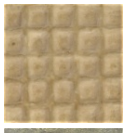
Quilted Paper Dry.		$300^2 \times 25$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.013	0.013	0.014	0.022	0.109	-
		$d = 3$	0.008	0.007	0.008	0.005	0.121	89
		$d = 4$	0.008	0.007	0.008	0.005	0.109	119
		$d = 5$	0.006	0.005	0.006	0.004	0.099	120

Wood Drying		$420^2 \times 14$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.007	0.006	0.006	0.984	0.886	-
		$d = 3$	0.006	0.007	0.008	0.005	0.050	171
		$d = 4$	0.004	0.004	0.005	0.004	0.027	229
		$d = 5$	0.002	0.002	0.002	0.002	0.016	253

Banana Decay.		$220^2 \times 33$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.035	0.035	0.010	0.072	0.010	-
		$d = 3$	0.051	0.041	0.012	0.013	0.032	47
		$d = 4$	0.025	0.020	0.006	0.009	0.026	63
		$d = 5$	0.020	0.016	0.005	0.006	0.019	64

Apple Slice Decay.		$156^2 \times 35$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.021	0.038	0.026	0.019	0.060	-
		$d = 3$	0.030	0.066	0.054	0.011	0.065	24
		$d = 4$	0.020	0.044	0.037	0.008	0.058	32
		$d = 5$	0.013	0.028	0.024	0.007	0.051	33

Potato Decaying		$200^2 \times 36$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.011	0.010	0.012	0.037	0.013	-
		$d = 3$	0.009	0.008	0.010	0.014	0.076	41
		$d = 4$	0.008	0.007	0.009	0.011	0.069	54
		$d = 5$	0.007	0.006	0.008	0.009	0.063	54

Waffle Toasting		$220^2 \times 30$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.027	0.028	0.020	0.090	0.014	-
		$d = 3$	0.037	0.032	0.023	0.030	0.071	48
		$d = 4$	0.039	0.036	0.023	0.024	0.059	65
		$d = 5$	0.014	0.016	0.013	0.016	0.049	67

Copper Patina		$460^2 \times 34$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.044	0.066	0.049	0.244	4e-04	-
		$d = 3$	0.057	0.062	0.043	0.075	0.029	209
		$d = 4$	0.043	0.053	0.040	0.050	0.018	286
		$d = 5$	0.035	0.046	0.035	0.040	0.018	287

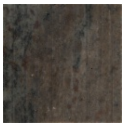
Orange Cloth Dry.		$512^2 \times 33$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.011	0.006	0.007	0.010	0.047	-
		$d = 3$	0.011	0.004	0.006	0.004	0.079	258
		$d = 4$	0.009	0.004	0.005	0.004	0.072	366
		$d = 5$	0.008	0.003	0.005	0.003	0.066	355

White Felt Dry.		$300^2 \times 28$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.014	0.012	0.013	0.009	0.019	-
		$d = 3$	0.011	0.009	0.011	0.004	0.043	89
		$d = 4$	0.008	0.007	0.008	0.003	0.037	121
		$d = 5$	0.007	0.005	0.006	0.003	0.031	123

Wet Brick Dry.		$260^2 \times 32$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.015	0.013	0.016	0.020	0.020	-
		$d = 3$	0.013	0.010	0.014	0.009	0.114	66
		$d = 4$	0.012	0.009	0.012	0.006	0.102	90
		$d = 5$	0.009	0.007	0.010	0.005	0.092	90

Green Cloth Dry.		$420^2 \times 30$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.005	0.004	0.006	0.008	0.090	-
		$d = 3$	0.006	0.006	0.005	0.004	0.081	176
		$d = 4$	0.005	0.004	0.004	0.004	0.077	232
		$d = 5$	0.004	0.003	0.004	0.004	0.071	239

Steel Rusting		$300^2 \times 22$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.013	0.011	0.014	0.013	2e-04	-
		$d = 3$	0.012	0.015	0.027	0.012	0.012	86
		$d = 4$	0.008	0.009	0.012	0.006	0.005	117
		$d = 5$	0.005	0.005	0.008	0.004	0.003	119

Granite Drying		$340^2 \times 27$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.023	0.022	0.025	0.019	0.325	-
		$d = 3$	0.040	0.040	0.040	0.012	0.119	117
		$d = 4$	0.026	0.026	0.025	0.008	0.101	157
		$d = 5$	0.013	0.012	0.014	0.006	0.086	158

Char. Wood Burn.		$480^2 \times 31$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.023	0.014	0.010	0.029	0.022	-
		$d = 3$	0.051	0.040	0.023	0.045	0.068	228
		$d = 4$	0.024	0.021	0.016	0.036	0.055	306
		$d = 5$	0.018	0.013	0.010	0.028	0.043	308

Bread Toasting		$260^2 \times 30$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.048	0.061	0.042	0.068	0.051	-
		$d = 3$	0.042	0.050	0.045	0.031	0.078	68
		$d = 4$	0.043	0.037	0.031	0.023	0.062	92
		$d = 5$	0.020	0.024	0.022	0.016	0.048	91

Cast Iron Cast.		$460^2 \times 30$	$K_d^\star$ (R)	$K_d^\star$ (G)	$K_d^\star$ (B)	$K_s^\star$	$\sigma^\star$	time [s]
		STAF	0.011	0.007	0.007	0.202	3e-04	-
		$d = 3$	0.011	0.008	0.010	0.028	0.018	209
		$d = 4$	0.012	0.007	0.006	0.016	0.011	281
		$d = 5$	0.009	0.008	0.007	0.011	0.008	286

[15] Desbenoit, B, Galin, E, Akkouche, S. Simulating and Modeling Lichen Growth. Computer Graphics Forum 2004;23:341–350.
[16] Hsu, Sc, Wong, Tt. Simulating Dust Accumulation. IEEE Computer Graphics and Applications 1995;15(1):18–22.
[17] Nakamae, E, Kaneda, K, Okamoto, T, Nishita, T. A Lighting Model Aiming at Drive Simulators. SIGGRAPH Computer Graphics 1990;24(4):395–404.
[18] Jensen, HW, Legakis, J, Dorsey, J. Rendering of Wet Materials. In: Proceedings of Eurographics Symposium on Rendering. 1999, p. 273–282.
[19] Elton, N, Legrix, A. Reflectometry of Drying Latex Paint. Journal of Coatings Technology and Research 2014;11.
[20] Bosch, C, Patow, G. Controllable Image-Based Transfer of Flow Phenomena. Computer Graphics Forum 2019;38(1):274–285.
[21] Chen, Y, Xia, L, Wong, TT, Tong, X, Bao, H, Guo, B, et al. Visual Simulation of Weathering by Gammaton Tracing. ACM Transactions on Graphics 2005;24(3):1127–1133.
[22] Sun, B, Sunkavalli, K, Ramamoorthi, R, Belhumeur, PN, Nayar, SK. Time-Varying BRDFs. IEEE Transactions on Visualization and Computer Graphics 2007;13(3):595–609.
[23] Lu, J, Georghiades, AS, Glaser, A, Wu, H, Wei, LY, Guo, B, et al. Context-Aware Textures. ACM Transactions on Graphics 2007;26(1).
[24] Ahmed, N, Theobalt, C, Seidel, HP. Spatio-Temporal Reflectance Sharing for Relightable 3D Video. In: Computer Vision/Computer Graphics Collaboration Techniques. 2007, p. 47–58.
[25] Zaidi, Q. Visual Inferences of Material Changes: Color as Clue and Distraction. WIREs Cognitive Science 2011;2(6):686–700.
[26] Xue, S, Wang, J, Tong, X, Dai, Q, Guo, B. Image-based Material Weathering. Computer Graphics Forum 2008;27(2):617–626.
[27] Bandeira, D, Walter, M. Synthesis and Transfer of Time-Variant Material Appearance on Images. In: Proceedings of Brazilian Symposium on Computer Graphics and Image Processing. 2009, p. 32–39.
[28] Bellini, R, Kleiman, Y, Cohen-Or, D. Time-varying Weathering in Texture Space. ACM Transactions on Graphics 2016;35(4):141:1–141:11.
[29] Iizuka, S, Endo, Y, Kanamori, Y, Mitani, J. Single Image Weathering via Exemplar Propagation. Computer Graphics Forum (Proceedings of Eurographics 2016) 2016;35(2):501–509.
[30] Guingo, G, Larue, F, Sauvage, B, Lutz, N, Dischler, JM, Cani, MP. Content-Aware Texture Deformation with Dynamic Control. Computers & Graphics 2020;91:95–107.
[31] Kurt, M, Szirmay-Kalos, L, Křivánek, J. An Anisotropic BRDF Model for Fitting and Monte Carlo Rendering. SIGGRAPH Computer Graphics 2010;44(1):1–15.
[32] Ozturk, A, Kurt, M, Bilgili, A, Gungor, C. Linear approximation of Bidirectional Reflectance Distribution Functions. Computers & Graphics 2008;32(2):149–158.
[33] Tongbuasirilai, T, Unger, J, Kronander, J, Kurt, M. Compact and intuitive data-driven BRDF models. The Visual Computer 2020;36(4):855–872.
[34] Efros, AA, Freeman, WT. Image Quilting for Texture Synthesis and Transfer. In: Proceedings of Computer Graphics and Interactive Techniques. 2001, p. 341–346.
[35] Barnes, C, Shechtman, E, Finkelstein, A, Goldman, DB. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. ACM Transactions on Graphics 2009;28(3):24:1–24:11.
[36] Hertzmann, A, Jacobs, CE, Oliver, N, Curless, B, Salesin, DH. Image Analogies. In: Proceedings of Computer Graphics and Interactive Techniques. 2001, p. 327–340.
[37] Tong, X, Zhang, J, Liu, L, Wang, X, Guo, B, Shum, HY. Synthesis of Bidirectional Texture Functions on Arbitrary Surfaces. ACM Transactions on Graphics 2002;21(3):665–672.
[38] Lefebvre, S, Hoppe, H. Appearance-Space Texture Synthesis. ACM Transactions on Graphics 2006;25(3):541–548.
[39] Han, J, Zhou, K, Wei, LY, Gong, M, Bao, H, Zhang, X, et al. Fast Example-based Surface Texture Synthesis via Discrete Optimization. The Visual Computer 2006;22(9):918–925.
[40] Enrique, S, Koudelka, M, Belhumeur, P, Dorsey, J, Nayar, S, Ramamoorthi, R. Time-Varying Textures: Definition, Acquisition, and Synthesis. In: ACM SIGGRAPH 2005 Sketches. 2005, p. 130–es.
[41] Gatys, L, Ecker, AS, Bethge, M. Texture Synthesis Using Convolutional Neural Networks. In: Proceedings of International Conference on Neural Information Processing Systems. 2015, p. 262–270.
[42] Zhou, Y, Shi, H, Lischinski, D, Gong, M, Kopf, J, Huang, H. Analysis and Controlled Synthesis of Inhomogeneous Textures. Computer Graphics Forum 2017;36(2):199–212.
[43] Mazlov, I, Merzbach, S, Trunz, E, Klein, R. Neural Appearance Synthesis and Transfer. In: Proceedings of Workshop on Material Appearance Modeling. 2019,.
[44] Torrance, K, Sparrow, E. Theory for Off-Specular Reflection from Roughened Surfaces. Journal of the Optical Society of America (JOSA) 1967;57(9):1105–1114.
[45] Wie, LY, Lefebvre, S, Kwatra, V, Turk, G. State of the Art in Example-based Texture Synthesis. In: Proceedings of Eurographics (State of the Art Reports). 2009,.
[46] Smith, B. Geometrical Shadowing of a Random Rough Surface. IEEE Transactions on Antennas and Propagation 1967;15(5):668–671.
[47] Heitz, E. Understanding the Masking-Shadowing Function in Microfacet-Based BRDFs. Journal of Computer Graphics Techniques 2014;3(2):48–107.
[48] Walter, B, Marschner, SR, Li, H, Torrance, KE. Microfacet Models for Refraction Through Rough Surfaces. In: Proceedings of Eurographics Symposium on Rendering. 2007, p. 195–206.
[49] Allegorithmic, . Substance Designer. 2019. Https://www.allegorithmic.com/products/substance-designer.
[50] Jamriska, O. Ebsynth: Fast Example-based Image Synthesis and Style Transfer. `https://github.com/jamriska/ebsynth`; 2018.
[51] Jakob, W. Mitsuba Renderer. 2010. Http://www.mitsuba-renderer.org.
[52] Havran, V, Filip, J, Myszkowski, K. Perceptually Motivated BRDF Comparison Using Single Image. Computer Graphics Forum 2016;35(4):1–12.
[53] Turk, G. Texture Synthesis on Surfaces. In: Proceedings of Computer Graphics and Interactive Techniques. 2001, p. 347–354.
[54] Wei, LY, Levoy, M. Texture Synthesis over Arbitrary Manifold Surfaces. In: Proceedings of Computer Graphics and Interactive Techniques. 2001, p. 355–360.
[55] Zhang, J, Zhou, K, Velho, L, Guo, B, Shum, HY, Shum, HY, et al. Synthesis of Progressively-variant Textures on Arbitrary Surfaces. ACM Transactions on Graphics 2003;22(3):295–302.
[56] Bitterli, B. Rendering resources. 2016. Https://benedikt-bitterli.me/resources/.