StyLit: Illumination-Guided Example-Based Stylization of 3D Renderings

Jakub Fišer[1]* Ondřej Jamriška[1] Michal Lukáč[1] Eli Shechtman[2] Paul Asente[2] Jingwan Lu[2] Daniel Sýkora[1]
[1]Czech Technical University in Prague, Faculty of Electrical Engineering [2]Adobe Research

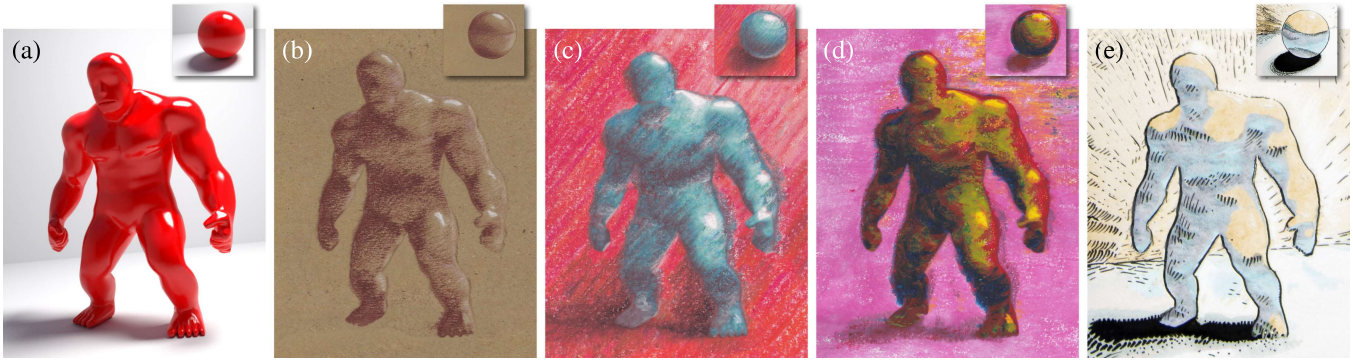

Figure 1: *Stylization of a 3D rendering (a) produced by our method using various style exemplars provided by the artist (top inset): (b) tonal drawing, (c) colored pencils, (d) oil pastel, and (e) comic drawing. Note how the specific stylization of individual lighting effects on the exemplar sphere is transferred to a similarly illuminated location in the target 3D rendering. Exemplar images:* © Daichi Ito *(b),* Pavla Sýkorová *(c, d), and* Lukáš Vlček *(e).*

Abstract

We present an approach to example-based stylization of 3D renderings that better preserves the rich expressiveness of hand-created artwork. Unlike previous techniques, which are mainly guided by colors and normals, our approach is based on light propagation in the scene. This novel type of guidance can distinguish among context-dependent illumination effects, for which artists typically use different stylization techniques, and delivers a look closer to realistic artwork. In addition, we demonstrate that the current state of the art in guided texture synthesis produces artifacts that can significantly decrease the fidelity of the synthesized imagery, and propose an improved algorithm that alleviates them. Finally, we demonstrate our method's effectiveness on a variety of scenes and styles, in applications like interactive shading study or autocompletion.

Keywords: example-based, texture synthesis, global illumination, light path expressions

Concepts: •**Computing methodologies** → **Non-photorealistic rendering;**

1 Introduction

Stylizing synthetic renderings of 3D models to give them a hand-crafted artistic appearance has wide applications in design, advertising, games, and movies. Previous example-based approaches [Sloan et al. 2001; Hertzmann et al. 2001; Bénard et al. 2013; Fišer et al. 2014; Barnes et al. 2015] have made significant progress, but the synthesized results still have a distinctively synthetic look when compared to real artwork. In this paper we identify two main factors that cause this problem and propose a solution to alleviate them.

The first limiting factor is that the state-of-the-art techniques rely mainly on color information to determine the stylized appearance. This leaves them unable to distinguish among different regions that have similar colors (see Fig. 2, top). Actual artists pay as much attention to lighting effects as they do to color when painting a scene. They often use different types of brush strokes and even different colors to depict differently lighted regions, even if they have the same color—for example, a gray diffuse region and a similarly-colored shadow might be painted differently (see Fig. 3). Artists purposefully deviate from the true colors in the scene to emphasize the lighting effects. To better emulate the look of real paintings, synthesis must take the illumination conditions of the synthetic scene into account. Although normals can partially alleviate this limitation [Sloan et al. 2001], they are useful only for a simple shading scenario where the light source is sufficiently far away that the normals correctly determine the locations of the lighting effects. When this assumption is not satisfied, or when there are other more advanced illumination effects like shadows or glossy reflections, normals become insufficient (see Fig. 2, bottom).

To allow illumination-dependent stylization, we propose a novel approach in which we compute light propagation in a simple scene (inset in Fig. 1a), and let the artist provide a corresponding painting (insets in Fig. 1b–e) that depicts various global illumination effects in arbitrary styles. (For brevity, we refer to the artist's creation as a painting, but it can use any technique like pen-and-ink, pastel, or colored pencil.) Then, for a more complex target scene (Fig. 1a) with a similar lighting environment, we compute the light propagation and use it to guide the synthesis—to find appropriate regions in the exemplar painting to transfer appearance from. The shadow, highlight and diffuse regions in the synthesis result exhibit similar visual style to the corresponding regions in the exemplar.

*e-mail:fiserja9@fel.cvut.cz

SIGGRAPH '16 Technical Paper, July 24–28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4279-7/16/07
DOI: http://dx.doi.org/10.1145/2897824.2925948

The second limiting factor of previous example-based stylization techniques is that they tend to distort high-level textural features like individual brush strokes [Sloan et al. 2001; Hertzmann et al. 2001; Bénard et al. 2013] (see Fig. 10d) or excessively reuse a small subset of source patches [Fišer et al. 2014; Barnes et al. 2015], producing a distinct wash-out effect manifesting as artificial repetitions or homogeneous areas that do not exist in the original style exemplar [Kaspar et al. 2015; Jamriška et al. 2015] (see Fig. 10i). Both types of artifacts give a distinctively synthetic look that decreases the fidelity of the synthesized image. Our solution encourages uniform usage of source patches while controlling the overall error budget to avoid enforcing the use of patches that can cause disturbing visual artifacts. This improvement allows us to generate compelling results in cases where previous approaches fail.

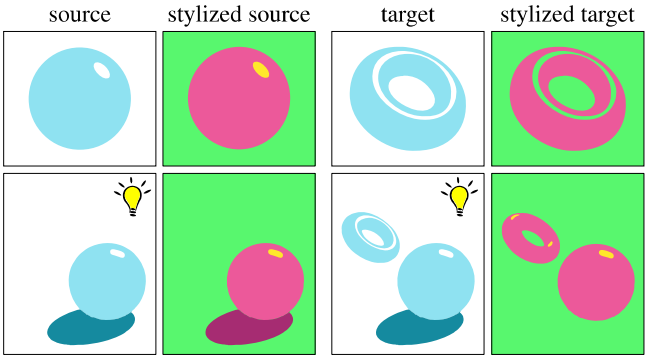

Figure 2: *Colors and normals are insufficient to guide the synthesis of stylized artwork. (Top) Color-based stylization fails when an artist wishes to apply different styles to parts of the source that have similar colors. (Bottom) Normal-based stylization cannot capture illumination changes that result when the light source is relatively close to the object. It also fails to transfer advanced illuminaiton effects such as shadows.*

Figure 3: *Examples of stylized still life paintings. Note how unique stylization was used to depict different illumination effects, e.g., pink shading on the bottom of the apple (left) or blue tint in the shadow region of the red pepper (right). Image courtesy* © Kerry Daley *(left) and* Gail Sibley *via howtopastel.com (right).*

2 Related Work

The goal of computer-assisted stylization [Kyprianidis et al. 2013] as pioneered by Paul Haeberli [1990] is to convert a photo or computer generated image into a digital painting. Numerous techniques were developed to achieve this goal, using physical simulation [Curtis et al. 1997; Haevre et al. 2007], procedural techniques [Bousseau et al. 2006; Bénard et al. 2010], advanced image filtering [Winnemöller et al. 2012; Lu et al. 2012], or an algorithmic composition of exemplar strokes [Salisbury et al. 1997; Zhao and Zhu 2011]. Zeng et al. [2009] decomposed the stylized image into a set of meaningful parts for which semantic interpretation is available [Tu et al. 2005], and then modified the stroke placement process to better convey the semantics of individual regions. All those techniques can produce impressive stylization results in certain cases, but they are limited to a specific appearance determined by the algorithm or by the library of used strokes.

Sloan et al. [2001] introduced The Lit Sphere—a generic example-based technique that uses a shaded sphere painted by an artist as the style exemplar. Pixels from this spherical exemplar are then transferred to the target 3D model using environment mapping [Blinn and Newell 1976], i.e., the color for a target pixel is transferred from the location in the source with the same normal. This leads to disturbing stretched-texture artifacts (see Fig. 10c, in which the orange shading lines are very stretched on the biggest torus). Moreover, since the technique requires one-to-one mapping between normals and lighting effects it can be used only for a simple shading scenario where the target has the same lighting environment as the source, the light is very far away, and there are no advanced illumination effects like shadows or glossy reflections.

Hertzmann et al. [2001] proposed a concept of image analogies where a pair of images (unfiltered and filtered) serves as an exemplar. For each pixel in the target, the algorithm finds the best corresponding location in the unfiltered source and transfers the look from the filtered counterpart. Due to its greedy nature it tends to resolve the balance between maintaining the texture coherency and following the guidance by introducing visible seams, which degrades the overall fidelity of the synthesized image (see Fig. 10d). This approach was later extended to handle animations [Hashimoto et al. 2003; Bénard et al. 2013] and to control the spatial location as well as local orientation of the source textural features in the synthesized image [Wang et al. 2004; Lee et al. 2010]. However, the guidance is still mainly based on color, making these approaches unable to handle illumination-dependent stylization.

In recent work [Fišer et al. 2014; Barnes et al. 2015] the original Hertzmann et al. synthesis algorithm has been replaced by a texture optimization technique [Kwatra et al. 2005; Wexler et al. 2007]. However, this approach suffers from a wash-out effect (see Fig. 10i) caused by excessively reusing patches with low-frequency content [Newson et al. 2014]. Numerous strategies have been developed to mitigate this phenomena. Those include a discrete solver [Han et al. 2006], feature masks [Lefebvre and Hoppe 2006], color histogram matching [Kopf et al. 2007], and bidirectional similarity [Simakov et al. 2008; Wei et al. 2008]. However, as recently demonstrated by Kaspar at al. [2015] and Jamriška et al. [2015], those techniques only work in some particular cases, for example when the source is mostly stationary and does not contain many nearly-homogeneous patches. Those conditions are usually violated for realistic style exemplars. Instead, Kaspar at al. and Jamriška et al. proposed more viable content-independent solutions that encourage uniform patch usage. A similar technique was previously used in [Chen and Wang 2010] and [Bénard et al. 2013]. Unfortunately, this uniform usage constraint does not apply in our scenario since some patches need to be used more often than others—for example when patches from one source highlight need to be reused for multiple highlights in the target.

Recently, an alternative approach to computer assisted stylization was proposed by Gatys et al. [2015]. It uses a deep neural network trained on object recognition [Simonyan and Zisserman 2014] to establish a mapping between features in the style and target image. Although this technique produces impressive results, it learns common visual features from a generic set of natural images and thus

does not fit the task of style transfer for rendered scenes. More importantly, the transfer is based purely on statistics of color patterns and provides no intuitive way to control the transfer process, so the result of the style transfer is mostly unpredictable.

In a related domain, Diamanti et al. [2015] showed how complex material appearance can be synthesized from limited examples using additional annotations, including normals and a simple shading descriptor. Similar to our method, they use rendered 3D models as targets. However, they focus on realistic material synthesis, and their annotations cannot capture the complex lighting phenomena expressed in our exemplars. In addition, for the synthesis, they use image melding [Darabi et al. 2012], which is unfortunately also prone to the wash-out problem described above.

3 Our Approach

We created a generic example-based stylization algorithm that supports arbitrary natural media and styles. The input is an example painting that is coarsely aligned to a simple 3D scene (Section 3.1). Our framework can then synthesize renderings of complex new scenes that imitate the visual style of the exemplar painting.

Previous stylization approaches based solely on color or normals cannot reproduce the richness of hand-painted images—higher level information is needed. One source of information can come from semantically parsing the objects [Zeng et al. 2009], but we believe that it is more important to analyze the light propagation in the scene. Because artists typically paint according to illumination effects, style-specific variations are driven more by illumination changes rather than by object identities (see Fig. 3). Since the 3D geometry is known in a computer-generated scene, the light propagation can be computed using established rendering algorithms [Kajiya 1986]. This allows us to distinguish differently illuminated regions (Section 3.2). We use the illumination information to guide the synthesis algorithm to achieve context-dependent stylization (Section 3.3).

3.1 Workflow Overview

In our workflow, an artist first prepares a stylized exemplar. We begin by creating a simple 3D scene that contains all important illumination effects that may subsequently occur in target scenes. A typical example of this scene is "a sphere on a table" (see inset in Fig. 1a), which bears resemblance to the Lit Sphere [Sloan et al. 2001]. The key difference is that placing the sphere on the table allows us to extend the the highlights and shading captured by Lit Sphere to additional illumination effects like soft shadows, color bleeding, and glossy reflections. We render the scene using a global illumination algorithm [Kajiya 1986] and print it on a paper in low contrast with special alignment marks. The artist then paints on this paper using any preferred media so that the final painting roughly aligns with the dimmed scene preview. We align a photo or scan of the painting with the original rendered image using the alignment marks.

Once the exemplar is ready, a more complex target scene can be created and rendered (see Fig. 1a). Our algorithm then transfers the hand-painted look from the exemplar to the target scene, preserving the stylization of individual illumination effects.

Implementing this workflow requires two components: (1) a mechanism to calculate light propagation in the scene and (2) an example-based algorithm that uses this light propagation information as guidance to synthesize a target image, preserving the stylization of individual illumination effects in the source exemplar. We describe these components in the following sections.

3.2 Light Path Expressions

In light transport, light-surface interactions can generally be classified as either *diffuse* or *specular*. Examining the interactions that happen on a path from a light source to the sensor lets us distinguish most important global illumination effects. This technique, known as *Light Path Expressions* (LPEs), has long been used in rendering [Heckbert 1990]. One of its uses is to separate the various illumination effects into distinct buffers for the purposes of filtering or in order to use a different rendering algorithm for each.

In our method, we use the same technique to gain insight into the light-scene interactions as seen in image space. By isolating the prominent global illumination effects, we gain additional information about how a real-life scene would appear to an artist, and we can take that information into account in the matching phase of the synthesis. This helps us make sure that effects like specular highlights, shadows, or diffuse interreflections—all of which are important shape and spatial relationship cues—are stylized consistently with the artist's intent.

We take advantage of existing renderers' ability to render these LPE buffers with little computational overhead (we use NVIDIA Iray SDK). We then create multi-channel images of the exemplar rendering and the target rendering, with channels containing the traditional RGB representation and the LPE buffers. All matching operations are performed on these multi-channel pixels.

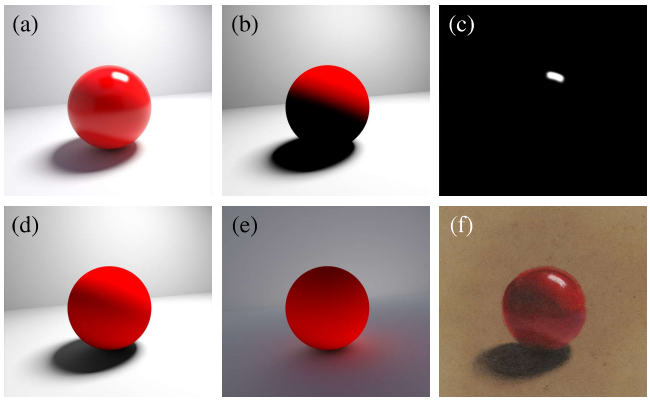

Figure 4: *An example of a style exemplar with Light Path Expression images: (a) full global illumination render, (b) direct diffuse (*`LDE`*), (c) direct specular (*`LSE`*), (d) first two diffuse bounces (*`LD{1,2}E`*), (e) diffuse interreflection (*`L.*DDE`*), (f) hand-drawn style image. Exemplar image* © Daichi Ito.

The light paths used in our examples (see Fig. 4) include direct diffuse and specular components (`LDE` and `LSE`), as well as the diffuse interreflection component (`L.*DDE`) and the first two diffuse bounces (`LD{1,2}E`). Additional channels can be added as necessitated by the given scene (e.g., the caustics channel `LS*DE`); our synthesis algorithm does not expect any particular number or set of channels and does not require that the path sets captured in different buffers be disjoint.

3.3 Synthesis Algorithm

Similarly to image analogies [Hertzmann et al. 2001] our task (see Fig. 5) is to take three multi-channel images A (exemplar scene rendered with LPE channels), A' (stylized exemplar aligned to the exemplar scene), and B (target scene rendered with LPE channels), and synthesize a new target image B' such that $A : A' :: B : B'$,

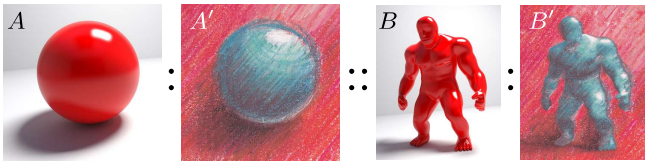

Figure 5: *The concept of image analogies. Exemplar image* © Pavla Sýkorová.

i.e., that style from A' is transferred to B' according to the similarity between A and B.

To solve this problem Hertzmann et al. originally proposed a simple multi-scale algorithm. For each resolution level and each pixel $q \in B'$ in scan-line order, a best matching pixel p is found in the source A' such that

$$E(\mathbf{A}, \mathbf{B}, p, q, \mu) = ||A'(p) - B'(q)||^2 + \mu||A(p) - B(q)||^2 \quad (1)$$

is minimized. Here $\mathbf{A} = \{A, A'\}$, $\mathbf{B} = \{B, B'\}$, and μ is a weight that controls the influence of guidance. For any image I, we use $I(p)$ to denote a feature vector at a pixel p. The vector $I(p)$ is a concatenation of all pixels in a small square patch of width w centered at the pixel p, where each pixel can have multiple feature channels. For features, Hertzmann et al. use the intensity value and an output from a steerable filter, whereas Bénard et al. [2013] augment the RGB colors with several additional guidance channels, including temporal coherence and a distance transform. In our case the feature vector contains colors of the full rendered image (RGB) and four LPE channels (each stored as a RGB image):

$$\{A, B\} = (\texttt{FULL}, \texttt{LDE}, \texttt{LSE}, \texttt{L.*DDE}, \texttt{LD\{1,2\}E}). \quad (2)$$

More LPE channels could be added to increase the discriminative power of the feature vector.

Although the original Hertzmann et al. algorithm produces impressive results, it suffers from its greedy nature and can fail to preserve high-level structures (see Fig. 10f). Subsequent work [Fišer et al. 2014; Barnes et al. 2015] showed that better results can be obtained using the optimization scheme described by Kwatra et al. [2005] and Wexler et al. [2007], which minimizes the following energy:

$$\sum_{q \in \mathbf{B}} \min_{p \in \mathbf{A}} E(\mathbf{A}, \mathbf{B}, p, q, \mu) \quad (3)$$

using EM-like iteration executed multiple times from coarse to fine resolution:

input : multi-channel images $\mathbf{A} = \{A, A'\}$ and $\mathbf{B}_k = \{B, B'_k\}$
output: synthesized target image B'_{k+1}

for *each pixel* $q \in \boldsymbol{B}_k$ **do**
  $NNF(q) = \underset{p \in \mathbf{A}}{\operatorname{argmin}} E(\mathbf{A}, \mathbf{B}_k, p, q, \mu)$

for *each pixel* $q \in \boldsymbol{B}_k$ **do**
  $B'_{k+1}(q) = Average(\mathbf{A}, NNF, q)$

Algorithm 1: EM-like iteration used to minimize energy (3).

Here B'_k denotes the current pixel values stored in B' and B'_{k+1} the updated values. NNF is the nearest neighbour field that assigns source patch to each target patch and function $Average$ computes the average color of colocated pixels in neighbour patches.

This approach produces notably better results, but as described in Section 2, it frequently leads to the wash-out effect (see Fig. 10i). Kaspar et al. [2015] and Jamriška et al. [2015] mitigate this problem by encouraging uniform source patch usage. However, this restriction is suitable only when each randomly picked sub-region of the source texture is perceived similarly. Enforcing uniform patch usage in our scenario would create artifacts; see Fig. 7 for a simplified illustration of this problem. For example, if the target has comparatively more highlight region than the source, uniform patch usage would force the highlight in the target to be filled with non-highlight source patches (see, e.g., Fig. 10m, n). In our scenario we need a different scheme that avoids the excessive use of certain patches while still permitting non-uniform utilization.

A possible solution would be for each source patch to estimate its optimal usage and then augment the original uniformity-preserving approaches to handle this non-uniform case. However, it is difficult to estimate optimal patch utilization in advance unless we run the actual synthesis. To overcome this chicken-and-egg problem we propose a different approach that inspects the actual matching errors during the synthesis and detects cases when the algorithm starts to force patches into inappropriate locations.

Our solution is based on the idea of reversed NNF retrieval [Rosenberger et al. 2009; Jamriška et al. 2015], in which a best matching target patch is retrieved for each source patch. The advantage of this approach is that it can enforce uniform usage of source patches. However, we must avoid forcing source patches to inappropriate locations in the target. We observe that, in practice, when we sort all matching error values and plot them with normalized axes, the resulting graph has a distinct, hyperbolic shape (see Fig. 6). It starts with small error values, corresponding to feasible assignments ($\mathbf{A}^*$ in Fig. 6). There is a knee point k where the error starts to increase rapidly. We estimate k by fitting a hyperbolic function $f(x) = (a - bx)^{-1}$ to the data and retrieving the point where $f'(x) = 1$, i.e., $k = \sqrt{1/b} + a/b$. Patches with indices above k are probably erroneous assignments ($\mathbf{A}^\times$ in Fig. 6) that need to be avoided. We thus set a feasible error budget T that is an integral of all patch errors with indices below k and modify the

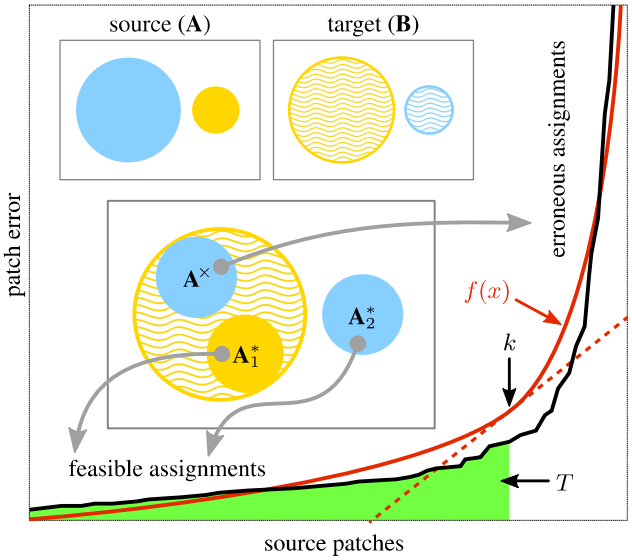

Figure 6: *Estimation of the error budget T: the sorted matching errors of all potential source-to-target patch assignments can be approximated by a hyperbolic curve $f(x)$ on which a knee point k is detected and used to distinguish between feasible $\mathbf{A}^*$ and erroneous $\mathbf{A}^\times$ assignments. The integral of the errors in $\mathbf{A}^*$ gives an estimate of the error budget T.*

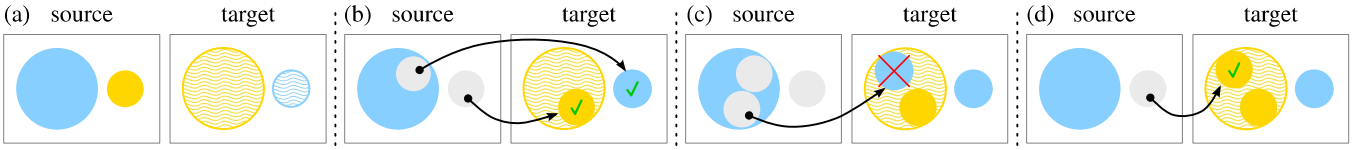

Figure 7: *Why enforcing uniform source patch usage is inappropriate in our scenario. (a) We often have the case when different types of patches have different distributions in the source and the target; here the source has much more blue than yellow, but the target requires much more yellow than blue. (b) The uniformity-preserving algorithm initially transfers source patches (marked with gray color) to proper locations in the target. (c) Eventually all suitable target locations can become occupied, leading the algorithm to force remaining source patches (not gray in b) into target positions with high matching error. (d) Our approach detects this erroneous case and restarts the retrieval so that appropriate source patches can be reused to fill remaining suitable positions in the target.*

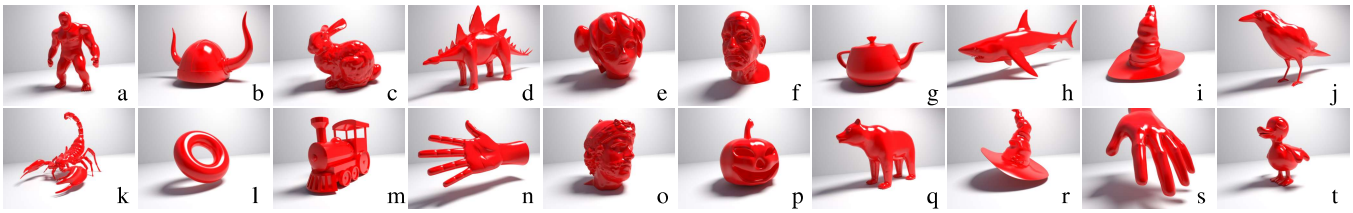

Figure 8: *A collection of 20 different models used to produce the results in this paper and supplementary material. The supplementary material shows the computed LPE channels. Source meshes via TurboSquid:* Veleran *(a),* Gerzi 3D ART *(b),* shoiko *(e),* cvbtruong *(f),* cartoomotion *(h),* luxxeon *(i, r),* Fernando Luceri *(k),* sylky *(m),* Giimann *(n),* oliverlaric *(o),* Nagashi *(p),* WindTrees *(q),* shiyamon *(s).*

original source-to-target assignment in a way that maximizes the number of used source patches $|\mathbf{A}^*|$ while satisfying an additional *feasibility* constraint:

$$\sum_{p \in \mathbf{A}^*} \min_{q \in \mathbf{B}} E(\mathbf{A}^*, \mathbf{B}, p, q, \mu) < T \tag{4}$$

Such a constrained assignment leads to the desired situation with some target patches remaining unassigned because assigning them would introduce artifacts (c.f. Fig. 7c). We can repeat the retrieval (c.f. Fig. 7d) and reuse good source patches to cover remaining positions in the target.

This iterative scheme stops when all target patches are covered. In practice it is feasible to stop even earlier (e.g., when 95% are covered) and use standard target-to-source nearest-neighbour retrieval to speed up the process. The number of iterations depends on the structure of the target scene and the complexity of the exemplar. Typically after 10 iterations more than 80% of target patches have been covered. In the general case, the number of iterations is roughly proportional to the ratio of the areas of corresponding illumination effects in the source and target images. For example, if the source contains one highlight and the target has four of similar size, then at least four iterations will be necessary to cover them. In practice, the number of iterations is typically slightly higher due to different structures of individual effects.

To complete the algorithm we plug our modified patch assignment process into the original EM iteration (Algorithm 1) by replacing the step where the nearest neighbour field *NNF* is created. We then run the standard coarse-to-fine synthesis.

3.4 Implementation details

We implemented our technique in C++ and CUDA. To accelerate the retrieval of nearest neighbours, we use PatchMatch with integrated support for masking [Barnes et al. 2009]. To further accelerate the processing we exploit multicore processing using parallel tiling on the CPU [Barnes et al. 2010] and jump flooding on the GPU [Rong and Tan 2006]. We use fixed patch size $w = 5$ and guidance influence $\mu = 2$. Our pyramid uses 2 for the downsampling ratio, and for a one-megapixel image, we run the synthesis on 6 levels with 6 optimization iterations on each level. The NNF retrieval is accelerated with 6 PatchMatch iterations.

Synthesizing a one-megapixel image takes about 15 minutes on a 3GHz CPU with 4 cores or 3 minutes on the GPU (GeForce GTX Titan Black). In addition to high quality synthesis, we also implemented a preview mode of the algorithm that uses half resolution, compresses LPE channels using PCA, and stores all textures as integers instead of the floating-point numbers used by the high-quality version. This achieves interactive response within 3–6 seconds on the GPU, enabling the applications discussed in Section 4.2.

4 Results

To validate our method we created a simple "sphere on the table" exemplar scene (see Fig. 4) and had five trained artists paint it in various styles using different kinds of media, including colored pencils, pastels, acrylic paint, watercolor, pen-and-ink, and markers (see Fig. 9, top). The artists had the option of including or not including certain effects; for example, they were not required to paint shadow or background if they did not want us to synthesize them.

In practice, a different exemplar scene could have been created to contain specific illumination effects (see, e.g., Figures 13 and 15), but we found that in most cases very good results could be obtained using a generic "sphere on the table" scene. This proves that our technique generalizes well despite the complexity of the target scene. Once the exemplar was painted we selected various meshes and rendered them under similar lighting conditions (see Fig. 8), i.e., light positions and materials were similar to the exemplar scene (Figures 16 & 17 and supplementary videos demonstrate how changing lighting conditions affect the final stylization).

We used different colors to distinguish the object from the background. This helped to avoid patch transfer between different materials and also allowed discrimination among different light interaction effects when one object reflected other, differently colored objects on its surface. The resulting color mixture guided the algo-

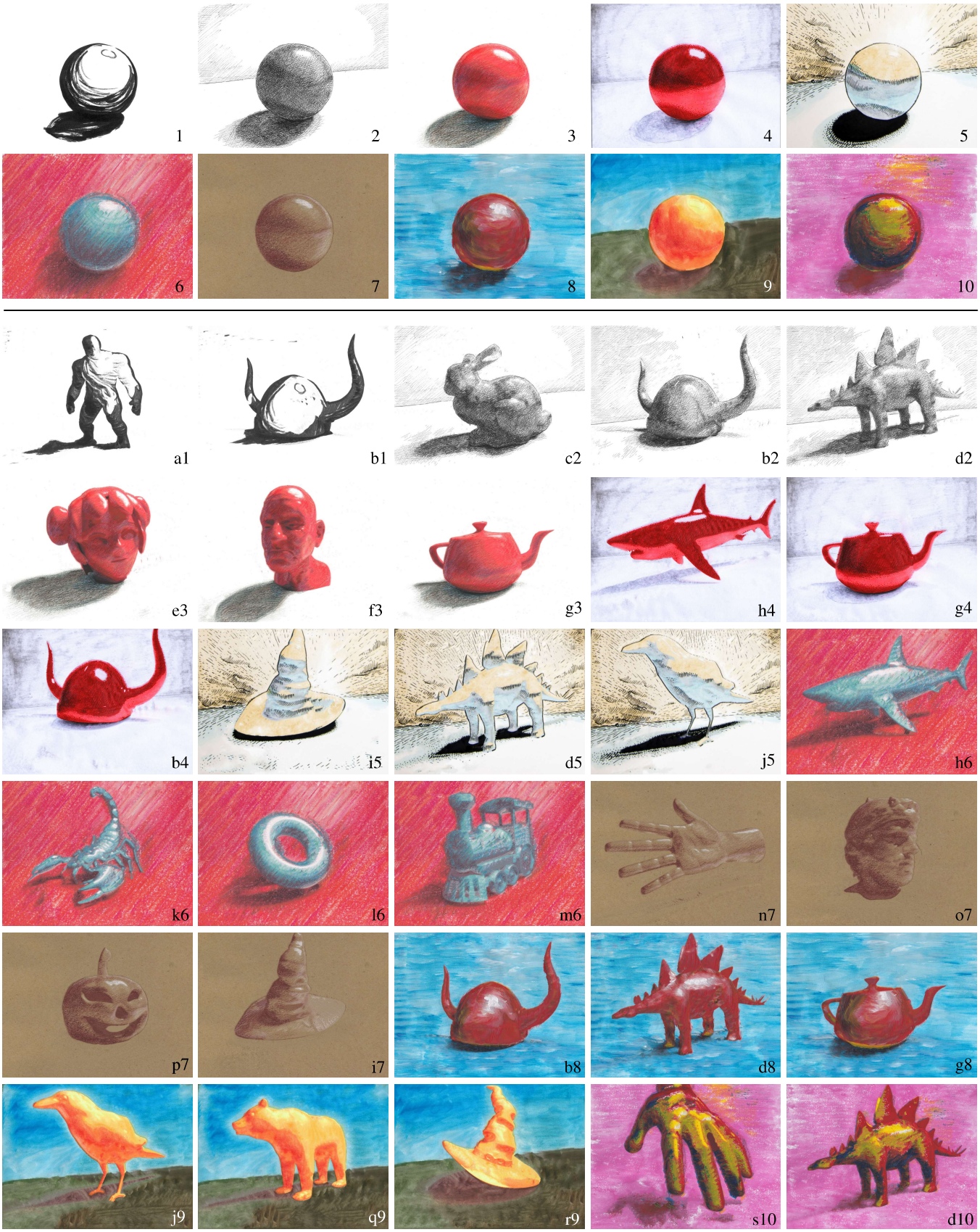

Figure 9: *Results—style examplars created by two trained artists in different kinds of media (top, denoted by numbers) were applied to models presented in Fig. 8 (denoted by letters) using our algorithm. Note how the resulting synthesized images (bottom) convey the stylization of lighting effects and preserve the textural richness of the source exemplar. Exemplar images* © Daichi Ito *(1, 2, 3, 7),* Karel Seidl *(4),* Lukáš Vlček *(5),* Lucie Svobodová *(9), and* Pavla Sýkorová *(6, 8, 10).*

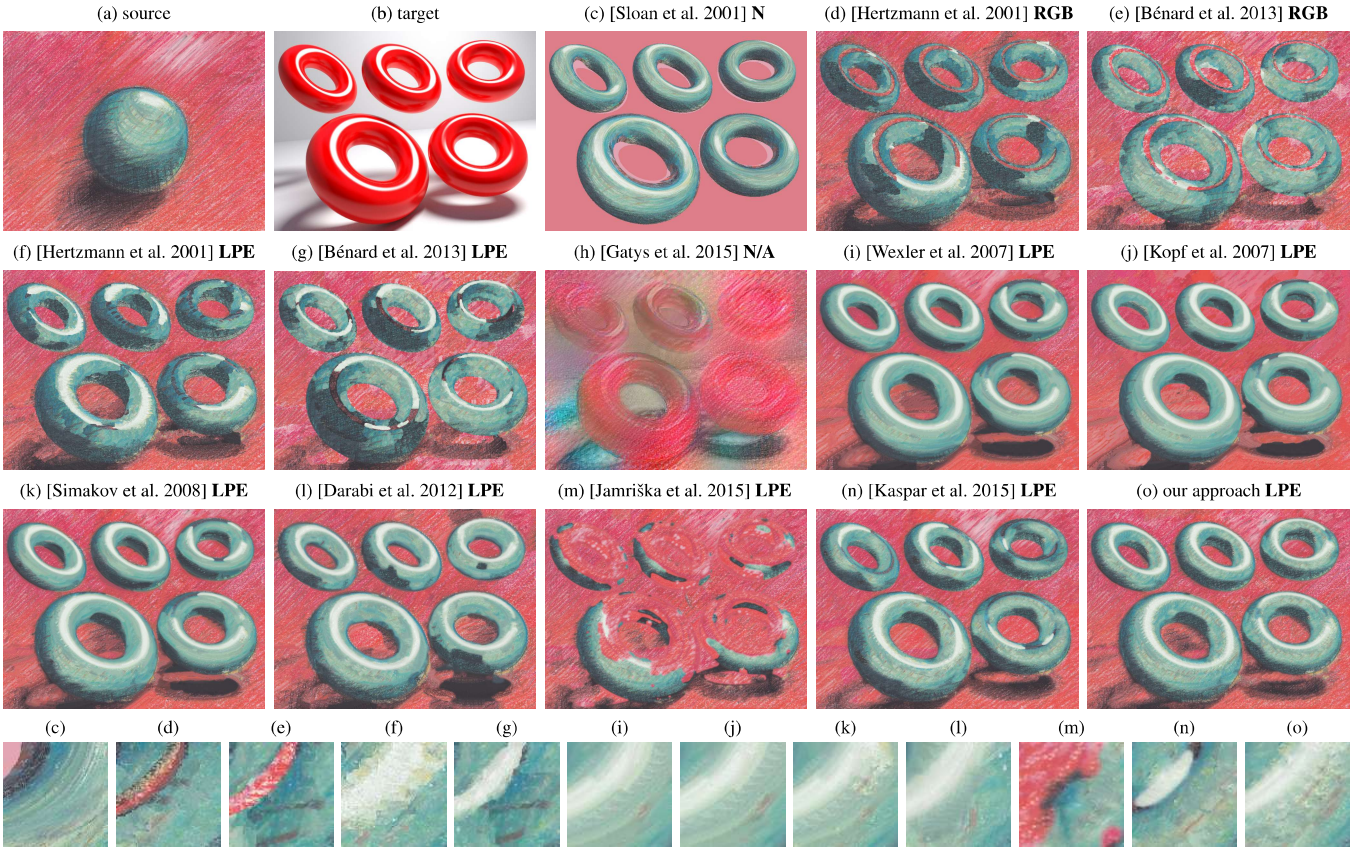

Figure 10: *Comparison with previous work—an expressive style exemplar (a) has been applied to a scene with five toruses (b) using previous approaches to example-based stylization (c, d, e, h, i, j) and techniques for general patch-based texture synthesis (f, g, k, l, m, n). Different approaches use different guidance channels: normals (**N**), colors (**RBE**), LPEs (**LPE**), and no guidance (**N/A**). Note how our approach (o) better preserves the visual structure of the target and the textural details of the source (see insets below). Exemplar image* © Pavla Sýkorová.

rithm to use samples that correspond to a similar interaction area in the source exemplar. In our results only two colors (white and red) were used, however, this color-coded guidance could be generalized to handle more objects with different materials.

The resulting synthesized images are presented in Figures 1 and 9 with additional results in the supplementary material. Note how accurately they follow the exemplar style. Strokes that correspond to highlights and shadows in the source are consistently transferred to proper locations in the target. Although there is no special treatment for object boundaries, the algorithm synthesizes them convincingly even in the presence of overdrawn strokes (see Fig. 1c, d). This happens because the discontinuities in LPE channels guide the synthesis to place boundary patches from the source at the boundaries in the target. Furthermore, the mechanism for eliminating excessive use of smooth patches encourages the algorithm to use high-frequency patches that typically contain overdrawn strokes.

The feedback from the trained artists who created the exemplars was very positive, with some commenting that many results looked exactly as if they had painted them by hand.

4.1 Comparison

We compared our technique with previous example-based stylization algorithms. For this we prepared a source exemplar and a rendering of a target scene where all previously mentioned issues occur; see Fig. 10. The exemplar (Fig. 10a) contains distinct stylization of each individual illumination effect and has rich texture details. In the target (Fig. 10b) the colors of the background and highlights match. The area light is close to the object, so that the assumption of the light being far away is violated, and the distribution of areas representing different lighting conditions is notably different from the source.

The Lit Sphere [Sloan et al. 2001] (Fig. 10c) uses normals to guide the rendering, causing flaws in the stylization and position of the highlights in the rendered scene. Because it just uses texture mapping to transfer the appearance in a pixel-wise fashion, there is no patch-wise consistency of the rendered output. The result is that the texture details of the exemplar are not reproduced correctly.

Image analogies [Hertzmann et al. 2001] (Fig. 10d) and extension [Bénard et al. 2013] (Fig. 10e) use color for guidance. They fail to retrieve patches from appropriate locations in the exemplar, most visibly by mistakenly using patches from the background for the highlights. The greedy nature of the algorithm leads to texture details being corrupted by artificial seams that break the fidelity of the synthesized image. We extended both approaches using our additional LPE channels (Fig. 10f, g), but the appearance of texture in the output is still far from the exemplar. Even with the LPE channels, Bénard et al. fails to reproduce highlights because of an additional histogram term [Chen and Wang 2010] that tries to encourage uniform patch usage.

The deep neural network approach of Gatys et al. [2015] strongly depends on visual patterns used during the training phase. In our

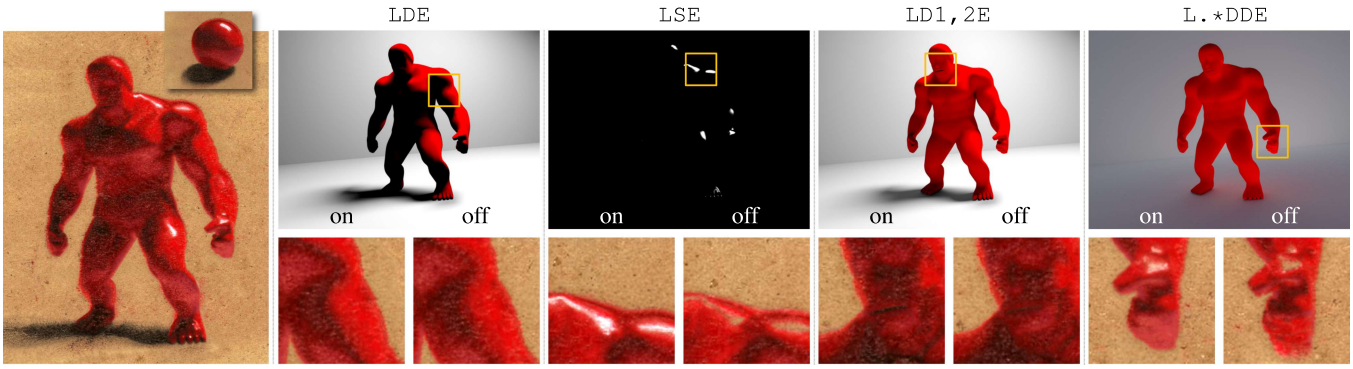

Figure 11: *The effect of adding individual LPE channels: diffuse (`LDE`) emphasizes contrast between lighted areas and areas in shadow, specular component (`LSE`) provides proper stylization of highlights, first and second diffuse bounce (`LD{1,2}E`) emphasizes details in shadows, and diffuse interreflections (`L.*DDE`) transfer the stylization of indirect illumination. Exemplar image* © Daichi Ito.

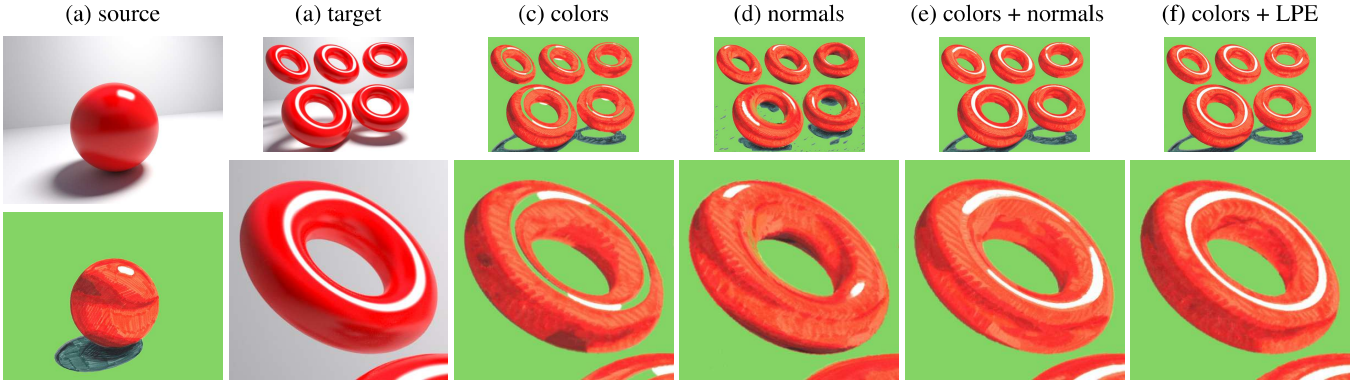

Figure 12: *To show the need for guidance based on LPE channels, we show the result of our improved synthesis method when based upon different sets of input channels: (c) colors only (d) normals only (e) colors with normals (f) colors with LPE. Only colors with LPE correctly capture the highlights, shadows, and shading effects. Exemplar image* © Daichi Ito.

scenario this approach completely fails (Fig. 10h); without proper guidance it is impossible to recover a meaningful assignment between the source and target features.

Fig. 10i shows how image analogies guided with LPE channels would look when computed using the texture optimization scheme [Wexler et al. 2007] with the original EM-like iteration (Algorithm 1). The wash-out effect is clearly visible because patches with low-frequency content [Newson et al. 2014] have been heavily overused. The variation by Kopf et al. [2007] (Fig. 10j) fails to take into consideration that the color histogram of the target must be different from the source, so its matching cannot improve the result in our scenario. Using bidirectional similarity [Simakov et al. 2008; Wei et al. 2008] (Fig. 10k) improves the results only a bit.

The same issue occurs also in the Image Melding method [Darabi et al. 2012] where patches are allowed to change color, rotate and scale and where image gradient channels are used in addition to our LPEs to guide the synthesis (Fig. 10l).

Enforcing uniform patch usage according to [Jamriška et al. 2015] gives good texture quality, but completely breaks the overall structure (Fig. 10m). Kaspar et al. [2015] give more flexibility by providing a parameter λ that controls the strength of the uniformity enforcement. However, we have found that λ must be manually tuned per scene to produce compelling results. Setting λ too low does not eliminate the wash-out effect, and setting it too high breaks the structure by enforcing overly uniform patch usage. To get the best result for the target in Fig. 10b we experimentally set $\lambda = 0.1$. This setting reduces the wash-out effect at the expense of distorting some highlights (Fig. 10n). Other settings of λ produce worse results—see supplementary material for comparison. Finally, our approach preserves both the textural richness and the overall structure of the target scene without tedious parameter tuning (Fig. 10o).

In Fig. 11 we show the influence of individual LPE channels on the resulting synthesis. Despite some changes being rather subtle, they significantly improve the fidelity of the resulting image and makes the style transfer more visually compelling. We also show in Fig. 12 how our improved synthesis algorithm behaves when used with only colors and normals as a guide. Just using colors (Fig. 12c) cannot distinguish between the highlights and the background. Just using normals (Fig. 12d) fails to place the highlights in the correct places, because the relatively close light source places highlights in areas with different surface orientations from the source. Using colors and normals together (Fig. 12e) works better, but still fails to place all highlights correctly and fails to reproduce some of the shading effects. Only using colors combined with LPE (Fig. 12f) correctly reproduces all the lighting effects of the target.

4.2 Applications

Our approach has numerous potential applications. It can be used to preview a target rendering style on various different geometries (see Fig. 9) or to test out multiple rendering styles on the same geometry (see Fig. 1). Alternatively, exemplars made by experienced artists can be used by others to produce stylized digital content.

It can also be used in animation (see supplementary video) or for autocompletion [Xing et al. 2014; Xing et al. 2015]. The user can stylize a single frame or only a portion of the model (see Fig. 13) and then use our technique to transfer the hand-drawn look to the rest of the model or sequence.

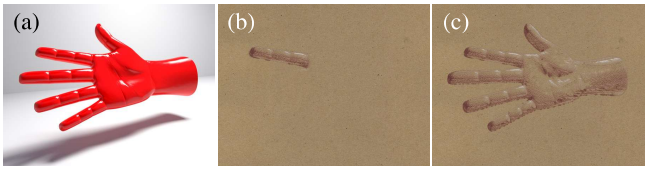

Figure 13: *Autocomplete shading—an artist draws shading for one finger (b) and our method synthesizes (c) the rest of the hand (a). Exemplar image* © Daichi Ito.

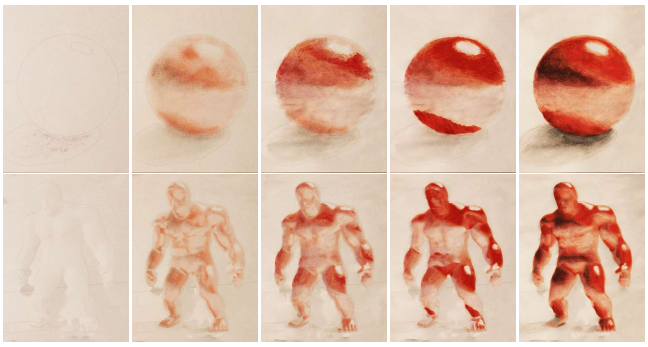

Figure 14: *On-the-fly shading study—the artist paints a simplified "sphere on the table" scene (top row) and watches as the stylization is transferred to the target model (bottom row). Such a continuous feedback helps to gradually improve the final stylization. Exemplar image* © Karel Seidl.

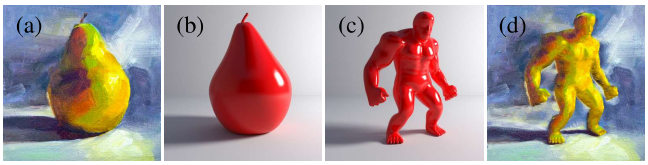

Figure 15: *Transferring style when a reference 3D model does not exist (a). We create an approximate 3D reconstruction, and light it to roughly capture illumination conditions in the original painting (b). We can then render a target model (c) and use our algorithm to transfer the style (d). Exemplar image* © Tina Wassel Keck *via Etsy.*

Artists and designers whose skills are insufficient to create a convincing painting of a detailed scene can instead create a "shading study" (as in Sloan et al. [2001]), and then let our method transfer the appearance to the target model automatically. Thanks to our GPU implementation this process can be performed on the fly (see Fig. 14) using a camera mounted above the canvas. The artist can improve the study based on immediate visual feedback. In Fig. 14, after seeing the rendering preview of the target model (bottom row), our artist decided to increase the color contrast and make the shadows darker. This approach can also be useful when an artist needs to paint an entire model by hand but would appreciate some reference visualizing how the actual stylization would look like on a more complex surface. Our technique can also be used to transfer style from an existing painting when a reference scene does not exist. We exploit existing techniques for 3D reconstruction from a single image [Zeng et al. 2015] and manually set up lights to mimic the illumination conditions of the original painting. Then we render corresponding LPE channels and feed them into our stylization pipeline to obtain a roughly corresponding stylized target scene (see Fig. 15).

5 Limitations and Future Work

Although our approach works quite well for many hand-painted exemplars and many target 3D scenes, there are some limitations that must be taken into account.

In Fig. 9 we demonstrate that our technique does a good job in generalizing illumination effects from a simple exemplar to relatively complex scenes. However, roughness in the style exemplar can sometimes suppress fine geometric details (see Fig. 16b, k, g). This effect can be alleviated with an additional edge map channel that emphasizes visually important features (Fig. 16c, d, m).

Our method produces best results when the lighting environment is similar to that in the exemplar scene. While some illumination changes can be handled (see adding a light source in Fig. 16f, h) we assume that all important illumination effects present in the target are present in the exemplar scene. For example, if the target scene has a dark shadow that is not present in the exemplar, our method cannot find proper patches and fills that area with inappropriate content, producing artifacts (see Fig. 17). We can sometimes alleviate this by matching the appearance of the exemplar's rendering with the rendering of the target scene (Fig. 16i, j, q).

When the rendering of the target scene is even more complicated, like having multiple objects with interreflections or challenging lighting effects like caustic or subsurface scattering, a different exemplar must be prepared and new LPE channels must be added. Our method is also not suitable for highly exaggerated stylization that does not closely fit the exemplar 3D scene—for example, when a stylized highlight is drawn where there is no rendered highlight.

In the future we would like to extend our technique to better handle animations. We plan to incorporate temporal coherence between stylized frames and give artistic control over the perceived temporal noise in the spirit of Fišer et al. [2014]. We would also like to explore how our synthesis algorithm with the error budget can address more general texture synthesis problems.

6 Conclusion

We have presented an approach to example-based stylization of 3D renderings that takes into account illumination effects. It includes an extended synthesis algorithm that better preserves the visual richness of hand-created style exemplars. Both the general approach and the synthesis algorithm dramatically improve the fidelity of the stylized images. We have demonstrated that our technique can handle a great variety of different stylizations. Our approach confirms the great potential of example-based techniques, and we hope it will inspire others to explore further their applicability in the field of non-photorealistic rendering.

Acknowledgements

We would like to thank artists D. Ito, P. Sýkorová, K. Seidl, L. Svobodová, L. Vlček, and J. Javora for creating examplars and recording interactive sessions. We are grateful also to J. Hendrich, T. Krupka, R. Smetana, D. Sedláček, M. Dvorožňák, J. Bittner, and P. Bénard for helping us with preparation of this paper and to all anonymous reviewers for insightful comments and suggestions.

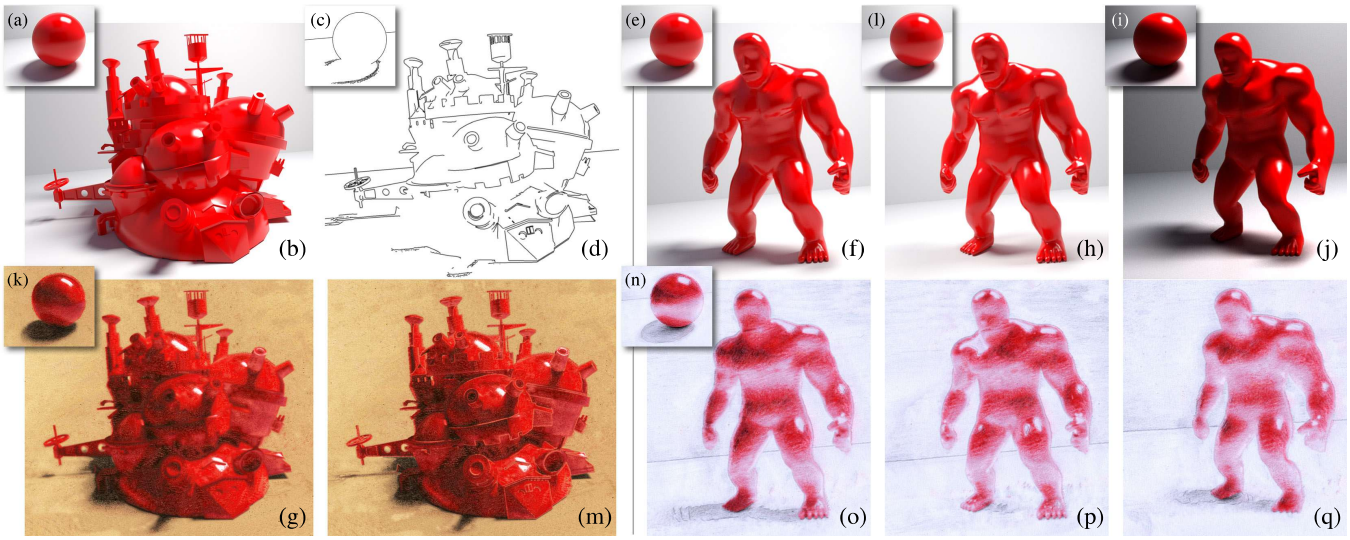

Figure 16: *Stylization of renderings with fine geometric details (b) and different lighting conditions (f, h, i). (a, e, l, i) are the source renderings and (k, n) are the style exemplars used to produce stylized targets (g, m) and (o, p, q) respectively. Lack of geometric details in the stylized rendering (g) can be alleviated (m) with an additional edge map channel (c, d). (o) shows synthesis of rendering with single and (p) with multiple light sources. (q) shows synthesis with the source rendering (i) matched to the target rendering (j). Exemplar images* © Daichi Ito *(k) and* Karel Seidl *(n). Source mesh via CGTrader:* luisma_l *(b).*

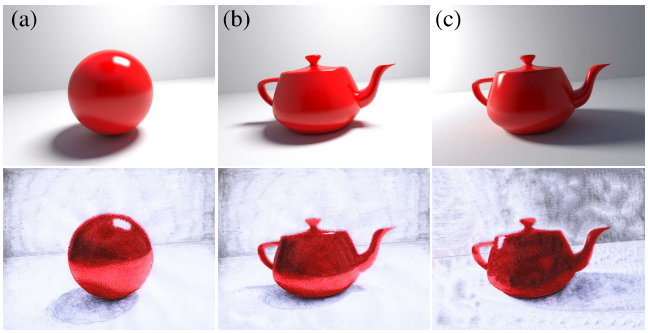

Figure 17: *Limitation—when the lighting environment is similar to that in the original exemplar scene (a) our approach produces best results (b). It can fail (c) when some illumination effects are missing in the source exemplar—here, highly shaded areas with little indirect illumination. Exemplar image* © Karel Seidl.

This research was funded by Adobe and has been supported by the Technology Agency of the Czech Republic under research program TE01020415 (V3C – Visual Computing Competence Center) and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/237/OHK3/3T/13 (Research of Modern Computer Graphics Methods). Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum provided under the programme "Projects of Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated.

References

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A., AND GOLDMAN, D. B. 2009. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics 28*, 3, 24.

BARNES, C., SHECHTMAN, E., GOLDMAN, D. B., AND FINKELSTEIN, A. 2010. The generalized PatchMatch correspondence algorithm. In *Proceedings of European Conference on Computer Vision*, 29–43.

BARNES, C., ZHANG, F.-L., LOU, L., WU, X., AND HU, S.-M. 2015. PatchTable: Efficient patch queries for large datasets and applications. *ACM Transactions on Graphics 34*, 4, 97.

BÉNARD, P., LAGAE, A., VANGORP, P., LEFEBVRE, S., DRETTAKIS, G., AND THOLLOT, J. 2010. A dynamic noise primitive for coherent stylization. *Computer Graphics Forum 29*, 4, 1497–1506.

BÉNARD, P., COLE, F., KASS, M., MORDATCH, I., HEGARTY, J., SENN, M. S., FLEISCHER, K., PESARE, D., AND BREEDEN, K. 2013. Stylizing animation by example. *ACM Transactions on Graphics 32*, 4, 119.

BLINN, J. F., AND NEWELL, M. E. 1976. Texture and reflection in computer generated images. *Communications of the ACM 19*, 10, 542–547.

BOUSSEAU, A., KAPLAN, M., THOLLOT, J., AND SILLION, F. 2006. Interactive watercolor rendering with temporal coherence and abstraction. In *International Symposium on Non-Photorealistic Animation and Rendering*, 141–149.

CHEN, J., AND WANG, B. 2010. High quality solid texture synthesis using position and index histogram matching. *The Visual Computer 26*, 4, 253–262.

CURTIS, C. J., ANDERSON, S. E., SEIMS, J. E., FLEISCHER, K. W., AND SALESIN, D. H. 1997. Computer-generated watercolor. In *SIGGRAPH Conference Proceedings*, 421–430.

DARABI, S., SHECHTMAN, E., BARNES, C., GOLDMAN, D. B., AND SEN, P. 2012. Image Melding: Combining inconsistent images using patch-based synthesis. *ACM Transactions on Graphics 31*, 4, 82.

DIAMANTI, O., BARNES, C., PARIS, S., SHECHTMAN, E., AND SORKINE-HORNUNG, O. 2015. Synthesis of complex image appearance from limited exemplars. *ACM Transactions on Graphics 34*, 2, 22.

FIŠER, J., LUKÁČ, M., JAMRIŠKA, O., ČADÍK, M., GINGOLD, Y., ASENTE, P., AND SÝKORA, D. 2014. Color Me Noisy: Example-based rendering of hand-colored animations with temporal noise control. *Computer Graphics Forum 33*, 4, 1–10.

GATYS, L. A., ECKER, A. S., AND BETHGE, M. 2015. A neural algorithm of artistic style. *CoRR abs/1508.06576*.

HAEBERLI, P. 1990. Paint by numbers: Abstract image representations. *SIGGRAPH Computer Graphics 24*, 4, 207–214.

HAEVRE, W. V., LAERHOVEN, T. V., FIORE, F. D., AND REETH, F. V. 2007. From Dust Till Drawn: A real-time bidirectional pastel simulation. *The Visual Computer 23*, 9-11, 925–934.

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

HASHIMOTO, R., JOHAN, H., AND NISHITA, T. 2003. Creating various styles of animations using example-based filtering. In *Proceedings of Computer Graphics International*, 312–317.

HECKBERT, P. S. 1990. Adaptive radiosity textures for bidirectional ray tracing. *SIGGRAPH Computer Graphics 24*, 4, 145–154.

HERTZMANN, A., JACOBS, C. E., OLIVER, N., CURLESS, B., AND SALESIN, D. H. 2001. Image analogies. In *SIGGRAPH Conference Proceedings*, 327–340.

JAMRIŠKA, O., FIŠER, J., ASENTE, P., LU, J., SHECHTMAN, E., AND SÝKORA, D. 2015. LazyFluids: Appearance transfer for fluid animations. *ACM Transactions on Graphics 34*, 4, 92.

KAJIYA, J. T. 1986. The rendering equation. *SIGGRAPH Computer Graphics 20*, 4, 143–150.

KASPAR, A., NEUBERT, B., LISCHINSKI, D., PAULY, M., AND KOPF, J. 2015. Self tuning texture optimization. *Computer Graphics Forum 34*, 2, 349–360.

KOPF, J., FU, C.-W., COHEN-OR, D., DEUSSEN, O., LISCHINSKI, D., AND WONG, T.-T. 2007. Solid texture synthesis from 2D exemplars. *ACM Transactions on Graphics 26*, 3, 2.

KWATRA, V., ESSA, I. A., BOBICK, A. F., AND KWATRA, N. 2005. Texture optimization for example-based synthesis. *ACM Transactions on Graphics 24*, 3, 795–802.

KYPRIANIDIS, J. E., COLLOMOSSE, J., WANG, T., AND ISENBERG, T. 2013. State of the "art": A taxonomy of artistic stylization techniques for images and video. *IEEE Transactions on Visualization and Computer Graphics 19*, 5, 866–885.

LEE, H., SEO, S., RYOO, S., AND YOON, K. 2010. Directional texture transfer. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, 43–48.

LEFEBVRE, S., AND HOPPE, H. 2006. Appearance-space texture synthesis. *ACM Transactions on Graphics 25*, 3, 541–548.

LU, C., XU, L., AND JIA, J. 2012. Combining sketch and tone for pencil drawing production. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, 65–73.

NEWSON, A., ALMANSA, A., FRADET, M., GOUSSEAU, Y., AND PÉREZ, P. 2014. Video inpainting of complex scenes. *SIAM Journal of Imaging Science 7*, 4, 1993–2019.

RONG, G., AND TAN, T.-S. 2006. Jump flooding in GPU with applications to Voronoi diagram and distance transform. In *Proceedings of Symposium on Interactive 3D Graphics and Games*, 109–116.

ROSENBERGER, A., COHEN-OR, D., AND LISCHINSKI, D. 2009. Layered Shape Synthesis: Automatic generation of control maps for non-stationary textures. *ACM Transactions on Graphics 28*, 5, 107.

SALISBURY, M. P., WONG, M. T., HUGHES, J. F., AND SALESIN, D. H. 1997. Orientable textures for image-based pen-and-ink illustration. In *SIGGRAPH Conference Proceedings*, 401–406.

SIMAKOV, D., CASPI, Y., SHECHTMAN, E., AND IRANI, M. 2008. Summarizing visual data using bidirectional similarity. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

SIMONYAN, K., AND ZISSERMAN, A. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR abs/1409.1556*.

SLOAN, P.-P. J., MARTIN, W., GOOCH, A., AND GOOCH, B. 2001. The Lit Sphere: A model for capturing NPR shading from art. In *Proceedings of Graphics Interface*, 143–150.

TU, Z., CHEN, X., YUILLE, A. L., AND ZHU, S.-C. 2005. Image Parsing: Unifying segmentation, detection, and recognition. *International Journal of Computer Vision 63*, 2, 113–140.

WANG, B., WANG, W., YANG, H., AND SUN, J. 2004. Efficient example-based painting and synthesis of 2D directional texture. *IEEE Transactions on Visualization and Computer Graphics 10*, 3, 266–277.

WEI, L.-Y., HAN, J., ZHOU, K., BAO, H., GUO, B., AND SHUM, H.-Y. 2008. Inverse texture synthesis. *ACM Transactions on Graphics 27*, 3.

WEXLER, Y., SHECHTMAN, E., AND IRANI, M. 2007. Space-time completion of video. *IEEE Transactions on Pattern Analysis and Machine Intelligence 29*, 3, 463–476.

WINNEMÖLLER, H., KYPRIANIDIS, J. E., AND OLSEN, S. C. 2012. XDoG: An extended difference-of-gaussians compendium including advanced image stylization. *Computers & Graphics 36*, 6, 740–753.

XING, J., CHEN, H.-T., AND WEI, L.-Y. 2014. Autocomplete painting repetitions. *ACM Transactions on Graphics 33*, 6, 172.

XING, J., WEI, L.-Y., SHIRATORI, T., AND YATANI, K. 2015. Autocomplete hand-drawn animations. *ACM Transactions on Graphics 34*, 6, 169.

ZENG, K., ZHAO, M., XIONG, C., AND ZHU, S.-C. 2009. From image parsing to painterly rendering. *ACM Transactions on Graphics 29*, 1, 2.

ZENG, Q., CHEN, W., WANG, H., TU, C., COHEN-OR, D., LISCHINSKI, D., AND CHEN, B. 2015. Hallucinating stereoscopy from a single image. *Computer Graphics Forum 34*, 2, 1–12.

ZHAO, M., AND ZHU, S.-C. 2011. Portrait painting using active templates. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, 117–124.