EUROGRAPHICS 2013 / I. Navazo, P. Poulin
(Guest Editors)

Volume 32 (2013), Number 2

Material Editing in Complex Scenes by Surface Light Field Manipulation and Reflectance Optimization

Chuong H. Nguyen Daniel Scherzer Tobias Ritschel Hans-Peter Seidel

MPI Informatik

Figure 1: *An input 3D scene* (a) *is painted by view-dependent strokes manipulating its surface light field.* b): *A red stroke made on the wagon indicating a change of diffuse color.* c): *A white stroke changing highlight shapes on the plane.* d): *Painting a white highlight on the wheel. Our system finds the smallest change of shading parameters to produce a light field matching the strokes* (e).

Abstract

This work addresses the challenge of intuitive appearance editing in scenes with complex geometric layout and complex, spatially-varying indirect lighting. In contrast to previous work, that aimed to edit surface reflectance, our system allows a user to freely manipulate the surface light field. It then finds the best surface reflectance that "explains" the surface light field manipulation. Instead of classic L_2 fitting of reflectance to a combination of incoming and exitant illumination, our system infers a sparse L_0 change of shading parameters instead. Consequently, our system does not require "diffuse" or "glossiness" brushes or any such understanding of the underlying reflectance parametrization. Instead, it infers reflectance changes from scribbles made by a single simple color brush tool alone: Drawing a highlight will increase Phong specular; blurring a mirror reflection will decrease glossiness; etc. A sparse-solver framework operating on a novel point-based, pre-convolved lighting representation in combination with screen-space edit upsampling allows to perform editing interactively on a GPU.

1. Introduction

The appearance of materials is an important cue for human understanding of its surrounding, beginning with distinguishing between edible and rotten food for early humans, and reaching up to today's intuition about the price of a car by looking at the lacquer. Therefore, material appearance is highly important in many computer graphics applications, ranging from product visualization, to feature films or computer games and its proper depiction. Acquisition and manipulation can be the key to achieving a desired goal. Recent studies about material design [KP10, NRE*12] point out the difficulty of material design, even under controlled and simple illumination and without spatial variation. However, in many use-cases, e. g., feature film production, it is important how spatially-varying materials appear in combination, in complex geometrical arrangements, with occlusions, and under complex (global) illumination.

We propose a system to design materials under such settings where an artist performs interactive appearance manipulation by painting strokes onto a "3D+2D canvas" (the surface light field *(SLF)* of the scene) and the system finds the best reflectance to produce the desired appearance (Fig. 2).

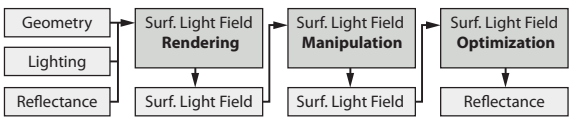

Figure 2: *Overview of our approach* (Please see text).

© 2013 The Author(s)
Computer Graphics Forum © 2013 The Eurographics Association and Blackwell Publishing Ltd. Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

Our work has two key motivations: Reflectance manipulation is too tedious to be effective, and direct SLF manipulation is too general to be intuitive. Questions like "How and where do I have to change glossiness of this ring over here to get a caustic that just is bright enough to be visible and blurry enough over there?" are avoided using our system: The user paints the bright spot and the system will find the required change. Second, direct manipulation of a SLF is too tedious, has too many degrees of freedom, in particular when the viewer moves and is not well supported by most rendering systems. Consider a user painting a white dot onto a red sphere, which could either mean a highlight, or a white, diffuse dot. If the intention was to draw a highlight, the user would need to move the camera, and draw the highlight in a new position. This would need to continue for many, if not all possible views. A prohibitively tedious process. Our system seeks to understand if the white dot is meant to be a highlight, and if yes, to change the reflectance of the red sphere in a way that generalizes to different viewpoints. Different from previous L_2-solutions in graphics and vision that match observations (i. e., a sampling of the SLF) with a reflectance model in a least-squares sense, our L_0-approach will find sparse changes of reflectance, i. e., it will prefer changes in only one parameter that reflects the users intention.

Our contributions are summarized as follows:

- A new user interface to manipulate surface light fields
- An approach to infer a sparse change of reflectance from the manipulated surface light field
- A pre-convolved, point-based representation of a family of potential surface light fields, that can be used for efficient manipulation, optimization and rendering.

2. Previous Work

Intuitive appearance editing in complex scenes requires to account for findings made in inverse rendering, material editing as well as perception.

Inverse rendering Instead of directly manipulating reflectance, light or geometry, inverse approaches allow the user to specify a desired appearance and find the matching reflectance, light or geometry. Paint with light [SDS*93] and Radio-optimization [KPC93] are classic solutions to infer light settings from user constraints. Poulin and Fournier [PF95] were first to propose a system that infers material parameters for a surface directly lit by a point light seen from a single view point. We extend this work to directional effects seen from multiple views in the presence of glossy global illumination while retaining interactivity. Pellacini proposes an interactive system to design shadows and lights [PTG02], but decoupled from material. Kerr et al. [KP10] evaluate user interfaces for material design, and identifies color editing as a main issue.

Material Editing Current BRDF editing frameworks focus on the direct manipulation of BRDFs and do not account for the light and shape context they appear in. Intuitive editing of BRDFs under complex environment lighting, limited to direct illumination is presented by Colbert [CPK06] and Ben-Artzi et al. [BAOR06]. Later work extended to editing with global illumination [BAEDR08, CPWAP08, NKLN10]. These methods are based on pre-computed radiance transfer (PRT) which takes long time (minutes to hours) for precomputation and impose a fixation of the view point [NKLN10, BAEDR08] or the number of editable BRDFs [BAEDR08]. In contrast, we exploit a fast GPU-based approach using pre-computed visibility and pre-convolved lumitexels on a point-based representation of the 3D scene, allowing to freely move the view point, arbitrary svBRDFs and editing with fast pre-computation time (below 1 second). Pellacini et al. [PL07] perform edits in appearance space, an idea we generalize by taking illuminant and geometry into account when propagating edits. More general than reflectance, SLFs [WAA*00] can be combined and displayed interactively [HC07]. Regrettably, SLFs resulting from this approach are not always physically meaningful or valid and manipulation is restricted to basic compositing. While our approach is based on the manipulation of SLFs as well, it results in physically-plausible reflectance that can be used in every rendering system.

Perception While shading models are widely used in practice, "thinking" in terms of shading parameters such as "gloss" does not map well to human perception [PFG00]. Our approach does not expose the shading model and its parameters to the user at all and uses scribbles to infer changes of reflectance. Similar, the perceptual disambiguation of light and materials under direct [LM71] or indirect [Lan99] illuminants further complicates the issue. In our system, appearance is manipulated in-situ, i. e., under a complex spatially varying illuminant such as environment maps or (indirect) shadows.

3. Background

This section introduces the required notation and states the continuous problem that has to be solved (Please also refer to the supplemental symbol table).

Notation Input to our system is a surface $\mathcal{M} \subseteq \mathbb{R}^3$ and a reflectance function $R(\mathbf{x}, \omega_\mathrm{i}, \omega_\mathrm{o}) \in \mathcal{M} \times \mathbb{S}^2 \times \mathbb{S}^2 \to \mathbb{R}^+$, i. e., a spatially varying BRDF. Here, $\mathbf{x}$ is a location on the surface $\mathcal{M}$ and ω_i as well as $\omega_\mathrm{o} \in \mathbb{S}^2$ are the incoming and outgoing directions respectively. In the following, we will ignore the dependency on wavelength and assume all operations are performed on all color channels. SLFs [WAA*00] map every location $\mathbf{x}$ and direction ω to the outgoing radiance $L_\mathrm{o}(\mathbf{x}, \omega) \in \mathcal{M} \times \mathbb{S}^2 \to \mathbb{R}^+$. Intuitively, a SLF describes how a surface looks from different viewing directions. Diffuse surfaces are invariant under different directions. Little spheres will be used in this article to visualize the SLF at a certain location (Fig. 3). The incident radiance also depends on position and orientation and will be denoted as L_i. The

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

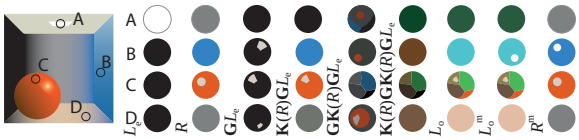

Figure 3: *Steps applied to SLFs with notation.*

geometry operator $\mathbf{G} \in \mathcal{M} \times \mathbb{S}^2 \to \mathcal{M} \times \mathbb{S}^2$ produces the field of incident radiance from a field of exitant radiance:

$$\mathbf{G}\,L_o(\mathbf{x},\omega) := L_o(v(\mathbf{x},\omega),-\omega),$$

where the *raycasting* function $v(\mathbf{x},\omega)$ returns the position that is closest to $\mathbf{x}$ along a ray from $\mathbf{x}$ in direction ω. The reflection operator $\mathbf{K} \in (\mathcal{M} \times \mathbb{S}^2 \times \mathbb{S}^2) \times (\mathcal{M} \times \mathbb{S}^2) \to \mathcal{M} \times \mathbb{S}^2$ [ATS94] in its bi-linear form [BAOR06] turns incoming radiance into outgoing radiance, depending on a BRDF

$$\mathbf{K}(R)L_i(\mathbf{x},\omega) := \int_{\mathbb{S}^2} L_i(\mathbf{x},\omega_i)R(\mathbf{x},\omega_i,\omega)\langle n(\mathbf{x}),\omega_i\rangle^+ \mathrm{d}\,\omega_i$$

were $n(\mathbf{x})$ is the surface normal at location $\mathbf{x}$. Choosing the bi-linear version will later allow to optimize over different choices of R to infer the users intention. Global illumination means to solve the *rendering equation* (RE)

$$L_o = L_e + \mathbf{K}(R)\,\mathbf{G}\,L_o,$$

i. e., to find a SLF L_o fulfilling the equation for a given lighting L_e, a given reflectance function R and a given geometry. The solution to the RE is

$$L_o = L_e + \mathbf{K}(R)\,\mathbf{G}\,L_e + \mathbf{K}(R)\,\mathbf{G}\,\mathbf{K}(R)\,\mathbf{G}\,L_e + \ldots.$$

Or shorter, using the i-bounce transport operator

$$\mathbf{T}^i(R) = \sum_{j=1}^{i} (\mathbf{K}(R)\mathbf{G})^{j-1} \quad \text{and} \quad \mathbf{T}^0(R) = \mathbf{I}.$$

Valid surface light fields Let $\mathcal{R}$ be a set of reflectance fields, e. g., all physically-plausible BRDFs $\mathcal{R}_{\text{Physical}}$, or all BRDFs a rendering system supports, e. g., Phong $\mathcal{R}_{\text{Phong}}$. We will call a SLF "valid" in respect to a surface $\mathcal{M}$, an initial emissive lighting L_e and a set of BRDFs $\mathcal{R}$ if it is the solution of the RE for *any* reflectance $R \in \mathcal{R}$. Our approach allows the user to manipulate (Sec. 4) a valid SLF L_o to become a new, potentially invalid, SLF L_o^m. Our system will seek to find a new reflectance $R^m \in \mathcal{R}$ resulting in a SLF after n_b bounces that is most similar to L_o^m. In a least squares sense, this is

$$R^m := \arg\min_{\hat{R}\in\mathcal{R}} ||\mathbf{T}^{n_b}(\hat{R})\,L_e - L_o^m||^2, \qquad (1)$$

using an SLF norm

$$||L|| := \sqrt{\int_{\mathcal{M}}\int_{\mathbb{S}^2} L(\mathbf{x},\omega)^2 \mathrm{d}\omega\mathrm{d}\mathbf{x}}.$$

4. Surface Light Field Manipulations

Manipulation is performed by selecting a tool, and painting strokes into the scene. A *tool* (Sec. 4.1) changes the SLF (spatially under the stroke; directionally from the view it is currently seen) into a new SLF. Tools can be used either direct, or indirectly (Sec. 4.2). After the system has computed a change of reflectance it gets propagated to all surface locations that are similar to the locations under the stroke (Sec. 4.3).

4.1. Tools

A *tool* is a manipulation operator $\mathbf{M} \in \mathcal{M} \times \mathbb{S}^2 \to \mathcal{M} \times \mathbb{S}^2$ that maps a SLF into an edited SLF. Common image manipulations like "replace", "clone", "brighten", "darken," "blur", "sharpen" etc. can be used here (Fig. 4). A "fixate" disallows changes to the SLF for some regions.

Figure 4: *Original SLFs* L_o (Top) *and the manipulated one* $\mathbf{M}L_o$ (Bottom). *The first three change the highlight size. The next two change its strength. The following five change the diffuse color at the same time. The last two show more complex examples, under non-point lighting.*

A manipulation *stroke* is a function $m(\mathbf{x},\omega) \in \mathcal{M} \times \mathbb{S}^2 \to \{0,1\}$, which is 1 if the surface at $\mathbf{x}$ seen from direction ω is affected by the tool and 0 otherwise. The new SLF is then given by $L_o^m := L_o + m \cdot (\mathbf{M}L_o - L_o)$.

4.2. Direct and Indirect Mode

All tools can be used either in "direct" or "indirect" mode. In direct mode, they affect the stroke and all similar regions, as defined above. In indirect mode, they do not affect the area under the stroke, but those areas, that contribute radiance to the stroke, such that $L_o^m := L_o + \mathbf{T}(R)\,(m \cdot (\mathbf{M}L_o - L_o))$ (Fig. 5).

For the example of a mirror object, $\mathbf{T}$ "copies" into every outgoing direction the value incoming in its mirrored direction, resulting in a "draw-across-the-mirror" behavior. The same in-direction works for glossy and diffuse appearance. The indirect mode, allows to keep reflectance constant at one location, but changes appearance by changing reflectance in another location of the scene. An example of indirect painting is to manipulate the color of indirect lighting. The system will change the reflectance, of the object producing the indirect light. Another example application is to paint a bright caustic-shaped stroke next to a diffuse ring and the system detects that the ring should be turned metallic to achieve a caustic. Note, that the reflectance at the location of the caustic is not changed, but at the location of the object producing the caustic (Fig. 6 and 19).

4.3. Edit propagation

After the stroke has finished, the system computes a change of appearance that is propagated to all similar locations. The similarity $s(\mathbf{x},\mathbf{y}) \in \mathcal{M}^2 \to \mathbb{R}^+$ between a pair of points $\mathbf{x}$

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

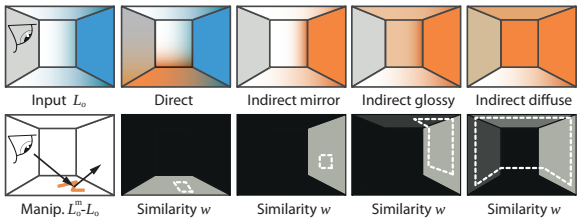

Figure 5: (From left to right)*: A user viewing from the left paints an orange stroke onto the ground* (a)*. Direct mode* (b)*: The stroke and everything similar – in this case the entire ground – changes reflectance. Indirect mode: When the ground is a mirror* (c)*, the reflected location changes its reflectance to match the stroke. A glossy* (d) *or diffuse* (e) *ground causes the change to be distributed over many locations.*

Figure 6: *Caustic example* (left to right). a)*: A user paints an indirect white stroke in the vicinity of a diffuse blue wall.* b)*: The resulting similarity.* c)*: A second stroke disambiguates the manipulation and the diffuse blue wall becomes specular.* d)*: Repeating the change makes the initially diffuse wall increasingly metallic and the caustic becomes more pronounced.*

and $\mathbf{y}$ on the manifold is the sum of the weighted distance in position, normal, or reflectance

$$s(\mathbf{x},\mathbf{y}) = \frac{\sigma_\mathrm{s}}{||\mathbf{x}-\mathbf{y}||} + \frac{\sigma_\mathrm{n}\,\langle n(\mathbf{x}), n(\mathbf{y})\rangle}{1-\langle n(\mathbf{x}), n(\mathbf{y})\rangle} + \frac{\sigma_\mathrm{r}}{\Delta_\mathrm{R}(\mathbf{x},\mathbf{y})}$$

were the $\sigma_{\{\mathrm{s,n,r}\}}$ are the weights and Δ_R is a BRDF difference function (see Figure 7).

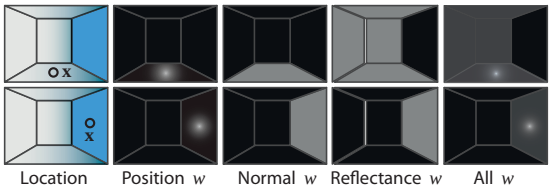

Figure 7: *Similarity of two different locations* $\mathbf{x}$ (Rows) *and other locations* $\mathbf{y}$*. Different components* (Columns, left to right)*: Position, normal, reflectance and a combination of all.*

The change of reflectance $\Delta_\mathrm{r} := R^\mathrm{m} - R \in \mathcal{M}\times\mathbb{S}^2\times\mathbb{S}^2 \to \mathbb{R}^+$ is propagated from each point to all other points, proportional to similarity, resulting in the final reflectance

$$R_\mathrm{f}(\mathbf{x},\cdot,\cdot) := R(\mathbf{x},\cdot,\cdot) + \int_\mathcal{M} s(\mathbf{x},\mathbf{y})\Delta_\mathrm{r}(\mathbf{y},\cdot,\cdot)\mathrm{d}\mathbf{y} \,/\, \int_\mathcal{M} s(\mathbf{x},\mathbf{y})\mathrm{d}\mathbf{y}.$$

5. Discretization

This section will describe the discretization of the domain used and how different forms of optimizations can be performed in practice. Surface fields will be discretized into finite spatio-directional elements stored in vectors (Sec. 5.1), and operators will take the form of matrices or matrix-valued functions (Sec. 5.2), similar to non-diffuse radiosity [ICG86]. Using these entities, the discrete reflectance that best matches the users manipulation is found in a non-linear optimization procedure (Sec. 5.3)

5.1. Discrete Domain

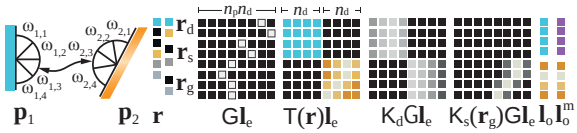

Figure 8: *Discretization example* (Left to right)*: Two elements* $\mathbf{p}_1$ *and* $\mathbf{p}_2$ *(blue diffuse; orange specular) and their directional bins* $\omega_{1,1},\ldots,\omega_{2,4}$ *. The reflectance* $\mathbf{r}$ *and its Phong components* $\mathbf{r}_\mathrm{d},\mathbf{r}_\mathrm{s},\mathbf{r}_\mathrm{g}$*. The operator* G *has one non-zero entry per row that links every element to another visible element and its bin* (arrow)*. The transport operator* $\mathsf{T}(\mathbf{r})$ *giving different weights to different incoming directions. Assuming Phong, it can be decomposed into* K_d *and* $\mathsf{K}_\mathrm{s}(\mathbf{r}_\mathrm{g})$*. The SLF* $\mathbf{l}_\mathrm{o}$ *which the user manipulates into* $\mathbf{l}_\mathrm{o}^\mathrm{m}$*.*

The problem is discretized into point elements to obtain a solution numerically (Fig. 8). The scene surface $\mathcal{M}$ is assumed to be static. It is spatially discretized into a set of n_p (typically thousands of) elements with locations $P := \{\mathbf{p}_1,\ldots,\mathbf{p}_{n_\mathrm{p}}\} \in \mathbb{R}^{3\times n_\mathrm{p}}$. Directionally, each element is discretized into n_d (typically $1024 = 32\times 32$) directional bins with directions $\omega_{i,1},\ldots,\omega_{i,n_\mathrm{d}}$. Sampling into point elements decouples the problem from the particular geometrical representation of $\mathcal{M}$. The original SLF is represented as a vector $\mathbf{l}_\mathrm{o} \in \mathbb{R}^{n_\mathrm{p}n_\mathrm{d}}$, the desired SLF as $\mathbf{l}_\mathrm{o}^\mathrm{m} \in \mathbb{R}^{n_\mathrm{p}n_\mathrm{d}}$ and the initial emissive light field as a vector $\mathbf{l}_\mathrm{e} \in \mathbb{R}^{n_\mathrm{p}n_\mathrm{d}}$. Every elements's reflectance can be parametrized by a n_s-dimensional shading model parameter vector. Over all elements this results in $\mathbf{r} \in \mathfrak{R}$, where $\mathfrak{R} \subseteq \mathbb{R}^{n_\mathrm{p}n_\mathrm{s}}$ is the set of potential BRDFs, a discrete version of $\mathcal{R}$. We will discuss the case of $n_\mathrm{s} = 3$ for Phong diffuse, specular and glossiness and write shorthand $\mathbf{r}_\mathrm{d} \in \mathbb{R}^{n_\mathrm{p}}$ for the diffuse, $\mathbf{r}_\mathrm{s} \in \mathbb{R}^{n_\mathrm{p}}$ for the specular and $\mathbf{r}_\mathrm{g} \in \mathbb{R}^{n_\mathrm{p}}$ for the glossy components of the parameter vector.

5.2. Discrete Operators

For more convenient notation, let $u_\mathrm{p} = u/n_\mathrm{d}$ the spatial index of row index u and $v_\mathrm{d} = v \bmod n_\mathrm{d}$ the directional index of column index v. Please, see Fig. 8 for the interleaving of indices.

Geometry The discrete version of the geometry operator $\mathbf{G}$ is the matrix $\mathsf{G} \in \mathbb{R}^{(n_\mathrm{p}n_\mathrm{d})\times(n_\mathrm{p}n_\mathrm{d})}$. For solid surfaces, G is zero in every column of row u and 1 in the column with the index of the element visible from $\mathbf{p}_{u_\mathrm{p}}$ in direction $\omega_{u_\mathrm{p},v_\mathrm{d}}$.

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

Reflection The discrete reflection operator is a matrix-valued function $\mathsf{K}(\mathbf{r}) \in \mathbb{R}^{(n_\mathrm{p} n_\mathrm{d}) \times (n_\mathrm{p} n_\mathrm{d})}$. Assuming a reflectance model like Phong, it can be written as

$$\mathsf{K}(\mathbf{r}) = \mathrm{diag}(\mathrm{rep}(\mathbf{r}_\mathrm{d}, n_\mathrm{d}))\mathsf{K}_\mathrm{d} + \mathrm{diag}(\mathrm{rep}(\mathbf{r}_\mathrm{s}, n_\mathrm{d}))\mathsf{K}_\mathrm{s}(\mathbf{r}_\mathrm{g}),$$

the sum of a diffuse reflection matrix, multiplied by diffuse reflection component $\mathbf{r}_\mathrm{d}$ and a specular reflection matrix, depending on glossiness $\mathbf{r}_\mathrm{g}$, multiplied by specular reflection component $\mathbf{r}_\mathrm{s}$. Both the diffuse and specular matrix are of the same size as K itself. Diffuse reflection maps incoming light into a directionally-invariant constant exitant value

$$\mathsf{K}_{\mathrm{d},u,v} := \langle n(\mathbf{p}_{u_\mathrm{p}}), \omega_{u_\mathrm{p}, v_\mathrm{d}} \rangle^+.$$

Specular reflection maps incoming light into directionally-dependent exitant values that are both mirrored and blurred proportional to glossiness

$$\mathsf{K}_{\mathrm{s},u,v}(\mathbf{r}_\mathrm{g}) := \langle \perp(\omega_{u_\mathrm{p}, u \bmod n_\mathrm{d}}, n(\mathbf{p}_{u_\mathrm{p}})), \omega_{u_\mathrm{p}, v_\mathrm{d}} \rangle^{\mathbf{r}_{\mathrm{g},u_\mathrm{p}}},$$

where $\perp(\omega, \mathbf{n}) = \omega - 2\mathbf{n}\langle \omega, \mathbf{n} \rangle$ reflects direction ω at the normal $\mathbf{n}$. Finally, the discrete i-bounce transport matrix is

$$\mathsf{T}^i(\mathbf{r}) = \sum_{j=1}^{i} (\mathsf{K}(\mathbf{r})\mathsf{G})^{j-1} \quad \text{with} \quad \mathsf{T}^0 = \mathsf{I}.$$

5.3. Discrete Minimization

The minimization seeks to find a discrete solution for the problem stated continuously in Eq. 1 by solving

$$\mathbf{r}^\mathrm{m} := \arg\min_{\hat{\mathbf{r}} \in \mathfrak{R}} ||\mathsf{W}(\mathsf{T}^{n_\mathrm{b}}(\hat{\mathbf{r}})\mathbf{l}_\mathrm{e} - \mathbf{l}_\mathrm{o}^\mathrm{m})||^2, \tag{2}$$

a reflectance, that after n_b bounces, given the initial emissive lighting $\mathbf{l}_\mathrm{e}$, produces a SLF most similar to the desired $\mathbf{l}_\mathrm{o}^\mathrm{m}$. The diagonal matrix $\mathsf{W} = \mathsf{I} + \sigma_\mathrm{m}\,\mathrm{diag}(\mathbf{m})$ is created from the stroke vector $\mathbf{m} \in \{0,1\}^{n_\mathrm{p} n_\mathrm{d}}$ (a discrete version of m) and a fixed and tool-dependent constant σ_m.

The problem is *non-linear* because the matrix T depends on the Phong glossines in $\mathbf{g}$ in a non-linear way. Furthermore, it is a *constrained* problem, as $\mathfrak{R}$ is only a subset of possible reflectance values that are valid, e. g., energy-preserving.

One-bounce weighted least-square-solution For simplicity, we will for now only consider one bounce. In this case, the reflectance for every element i can be found independently of all others. Still, the dependency of T on $\mathbf{r}_\mathrm{g}$ poses a non-linear problem, but with only one remaining non-linear degree of freedom: glossiness. Therefore, our solution iterates over n_c (typically $n_\mathrm{c} = 20$) fixed and perceptually uniform [PFG00] glossiness values $g_0, \ldots, g_{n_\mathrm{c}}$ and solves the linear problem of finding the best specular and diffuse reflectance for each (Fig. 9).

In the remainder of this section we will introduce "local" identifiers $(\mathbf{x}, \mathbf{b}, \mathsf{W}, \mathsf{A}, \hat{\mathbf{x}}, \mathfrak{X})$ with a dependency on i and j which is omitted in the notation for brevity. For element i and glossiness j, finding $\mathbf{r}_\mathrm{d}^{(i,j)}$ and $\mathbf{r}_\mathrm{s}^{(i,j)}$ is a linear problem $\mathsf{W}\mathsf{A}\mathbf{x} = \mathsf{W}\mathbf{b}$.

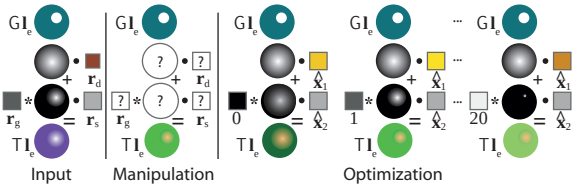

Figure 9: *Our fitting in 1D* (Top to bottom)*: Starting from a given lighting and desired SLF, the optimal reflectance is found by enumerating different glossiness levels* (horizontal)*, using a linear solution for the diffuse and specular reflectance at each.*

Here $\mathbf{x}$ is the unknown diffuse and specular reflection coefficient of the current element and the current glossiness, $\mathbf{b} = (\mathbf{l}^\mathrm{m}_{\mathrm{o},(i+0)\cdot n_d}, \ldots, \mathbf{l}^\mathrm{m}_{\mathrm{o},(i+1)\cdot n_d})^\mathsf{T}$ is the desired SLF and

$$\mathsf{A} = \begin{pmatrix} (\mathsf{K}_\mathrm{d}\mathsf{G}\mathbf{l}_\mathrm{e})_{i \cdot n_\mathrm{d}} & (\mathsf{K}_\mathrm{s}(g_j)\mathsf{G}\mathbf{l}_\mathrm{e})_{i \cdot n_\mathrm{d}} \\ \vdots & \vdots \\ (\mathsf{K}_\mathrm{d}\mathsf{G}\mathbf{l}_\mathrm{e})_{i \cdot n_\mathrm{d} + n_\mathrm{d}} & (\mathsf{K}_\mathrm{s}(g_j)\mathsf{G}\mathbf{l}_\mathrm{e})_{i \cdot n_\mathrm{d} + n_\mathrm{d}} \end{pmatrix}.$$

The first column in A is the reflected diffuse light, the second column the reflected specular light for the current glossiness.

Let $\mathsf{C} \in \mathbb{R}^{2\times 2}$ be $\mathsf{A}^\mathsf{T}\mathsf{W}\mathsf{A}$ and $\mathbf{d} \in \mathbb{R}^2$ be $\mathsf{A}^\mathsf{T}\mathsf{W}\mathbf{b}$. The closed-form solution for a least-squares fit then is

$$\hat{\mathbf{x}} = \left(\frac{\mathsf{C}_{2,2}\mathbf{d}_1 - \mathsf{C}_{1,2}\mathbf{d}_2}{\mathsf{C}_{1,1}\mathsf{C}_{2,2} - \mathsf{C}_{1,2}\mathsf{C}_{2,1}}, \frac{\mathsf{C}_{1,1}\mathbf{d}_2 - \mathsf{C}_{2,1}\mathbf{d}_1}{\mathsf{C}_{1,1}\mathsf{C}_{2,2} - \mathsf{C}_{1,2}\mathsf{C}_{2,1}} \right)^\mathsf{T} \tag{3}$$

Let $\mathfrak{X} := \{(\hat{\mathbf{x}}^{(1)}, g_1), \cdots, (\hat{\mathbf{x}}^{(n_\mathrm{c})}, g_{n_\mathrm{c}})\}$ be the set of different solutions for different glossiness values. Finally, we pick $\mathbf{r}_i := \alpha(\mathfrak{X})$, where

$$\alpha(\mathfrak{X}) := \mathfrak{X}_k \text{ with } k = \arg\min_{1 \le k \le |\mathfrak{X}|} ||\mathbf{l}_\mathrm{o}^\mathrm{m} - \mathsf{T}(\mathfrak{X}_k)\mathbf{l}_\mathrm{e}|| \tag{4}$$

is an operator to select the element from a set that produces the smallest residual magnitude.

The solution is constraint to $\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2 \le 1$ for energy-preserving BRDFs. We do not solve a constraint least-squares problem, but re-scale the solution to be energy-preserving, or to 0 if it is negative in any component. This might result in suboptimal solutions, but guarantees energy preservation.

One-bounce sparse solution While the $\mathcal{L}_2$ solution explained before changes all shading parameters to match to the desired SLF, we will now extend it to a sparse solution, which will aim to change only one or a few parameters inspired by $\mathcal{L}_0$ minimization in compressed sensing [Don06]. Let $||\mathbf{x}||_0 := \mathbb{R}^n \to \mathbb{N}$ be the count of non-zero entries in the vector $\mathbf{x}$. We are looking for a new reflectance $\mathbf{r}^\mathrm{m}$, such that $||\mathbf{r}^\mathrm{m} - \mathbf{r}||_0$ is n_p (the number of elements), or in other words, such that only one shading parameter is changed per element. At the same time, we want the solution to match the desired SLF. Please note, that we are not looking for sparse shading parameters (which is hardly meaningful), but for a sparse *change* of shading parameters. While $\mathcal{L}_0$ minimization is

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

computational intractable for large system, our problem is small enough to enumerate over the sensible solutions. To this end, the solver explained before is extended to hold all shading parameters fixed except one, and solve for the remaining parameter in the least squares sense. The closed-form least-squares fit for holding diffuse or specular fixed, is

$$\hat{\mathbf{x}}_\mathrm{d} = \left(\frac{\mathsf{C}_{1,1}(\mathbf{d}_1 - \mathsf{C}_{1,2}\mathbf{r}_\mathrm{s}) + \mathsf{C}_{2,1}(\mathbf{d}_2 - \mathsf{C}_{2,2}\mathbf{r}_\mathrm{s})}{\mathsf{C}_{1,1}^2 + \mathsf{C}_{2,1}^2}, \mathbf{r}_\mathrm{s}\right)^\mathsf{T} \quad (5)$$

and

$$\hat{\mathbf{x}}_\mathrm{s} = \left(\mathbf{r}_\mathrm{d}, \frac{\mathsf{C}_{1,2}(\mathbf{d}_1 - \mathsf{C}_{1,1}\mathbf{r}_\mathrm{d}) + \mathsf{C}_{2,2}(\mathbf{d}_2 - \mathsf{C}_{2,1}\mathbf{r}_\mathrm{d})}{\mathsf{C}_{1,2}^2 + \mathsf{C}_{2,2}^2},\right)^\mathsf{T}. \quad (6)$$

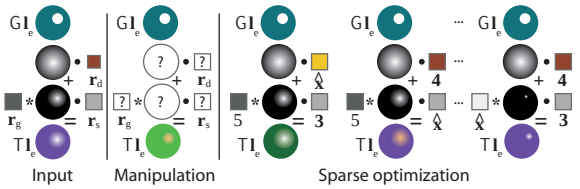

Figure 10: *Sparse fitting.*

The first solution answers, what diffuse reflectance will result in the remaining outgoing radiance, when the specular part is fixed. The second solution is of similar nature: what specular reflectance will best produce the outgoing radiance when the diffuse part is fixed? In both, glossiness is fixed. To find the solutions for varying glossiness, we simply enumerate in n_c discrete steps, resulting in $n_\mathrm{c}+2$ possible solutions: A diffuse change, a specular change and many glossy changes:

$$\mathfrak{X} := \{(\hat{\mathbf{x}}_\mathrm{d}^{(\mathbf{r}_{\mathrm{g},i})}, \mathbf{r}_{\mathrm{g},i}), (\hat{\mathbf{x}}_\mathrm{s}^{(\mathbf{r}_{\mathrm{g},i})}, \mathbf{r}_{\mathrm{g},i}), (\mathbf{r}_{\mathrm{d},i}, \mathbf{r}_{\mathrm{s},i}, g_1), \ldots, (\mathbf{r}_{\mathrm{d},i}, \mathbf{r}_{\mathrm{s},i}, g_{n_\mathrm{c}})\}$$

Again, the tentative solution with the smallest residual error is picked using α from Eq. 4

Mixed norm-solver A mixed solver computes changes in all ($\mathcal{L}_2$), in some (similar to $\mathcal{L}_0$, but with mixed degrees of freedom instead of only 1), and in only one parameter ($\mathcal{L}_0$) at the same time. We enumerate different glossiness in n_c discrete steps and current glossiness, in every glossiness iteration, we find the solution in case of fixing both, fixing diffuse only, fixing specular only or fixing none, resulting in $4(n_\mathrm{c}+1)$ different solutions:

$$\begin{aligned}\mathfrak{X} := \{\ & (\mathbf{r}_{\mathrm{d},i}, \mathbf{r}_{\mathrm{s},i}, \mathbf{r}_{\mathrm{g},i}), (\hat{\mathbf{x}}^{(\mathbf{r}_{\mathrm{g},i})}, \mathbf{r}_{\mathrm{g},i}), (\hat{\mathbf{x}}_\mathrm{d}^{(\mathbf{r}_{\mathrm{g},i})}, \mathbf{r}_{\mathrm{g},i}), (\hat{\mathbf{x}}_\mathrm{s}^{(\mathbf{r}_{\mathrm{g},i})}, \mathbf{r}_{\mathrm{g},i}) \\ & (\mathbf{r}_{\mathrm{d},i}, \mathbf{r}_{\mathrm{s},i}, g_1), (\hat{\mathbf{x}}^{(1)}, g_1), (\hat{\mathbf{x}}_\mathrm{d}^{(1)}, g_1), (\hat{\mathbf{x}}_\mathrm{s}^{(1)}, g_1) \\ & \vdots \\ & (\mathbf{r}_{\mathrm{d},i}, \mathbf{r}_{\mathrm{s},i}, g_{n_\mathrm{c}}), (\hat{\mathbf{x}}^{(n_\mathrm{c})}, g_{n_\mathrm{c}}), (\hat{\mathbf{x}}_\mathrm{d}^{(n_\mathrm{c})}, g_{n_\mathrm{c}}), (\hat{\mathbf{x}}_\mathrm{s}^{(n_\mathrm{c})}, g_{n_\mathrm{c}})\}.\end{aligned}$$

To this end, weights are given to the outcome of each solution, e. g., 1 to the sparse (fix 2 parameters out of 3), 5 to the mixed (fix 1 parameter out of 3) and 25 to the least squares solver (fix none). The best solution is again picked using α from Eq. 4.

***n*-bounce** Because change in reflectance in the presence of multiple bounces also changes the incoming light for every element, the above process is iterated. This approach smoothes out the error, but in the presence of specular transport might not converge to the global optimal reflectance. In practice we did not observe any problems with convergence towards local minima.

6. GPU Implemenation

We parallelize the solver over all elements. One thread is executed for every element and all possible solutions $\mathfrak{X}$ are enumerated and the best one returned by α. Still, computing the elements of A is computationally expensive: the first column contains exitant diffuse illumination for all directions, the second one stores exitant specular illumination for all directions. To compute exitant illumination, first the geometry operator needs to be applied, and second the convolution with one diffuse and with many specular kernels (one for each glossiness) needs to be performed. Next we will introduce pre-computed visibility (Sec. 6.1) and pre-convolved radiance (Sec. 6.2) to accelerate both steps. After this, the final algorithm is explained, including pseudocode in Sec. 6.3 and an approach to up-sample (6.5) and render (Sec. 6.4) the solution found.

6.1. Pre-computed Visibility (G)

When dealing with solid surfaces, the operator G is a matrix with only one non-zero entry per row. Let $\mathbf{h} \in \mathbb{N}^{n_\mathrm{p} n_\mathrm{d}}$ be an integer vector storing the index of this non-zero entry of row u in entry $\mathbf{h}_u$. A texture is used to store $\mathbf{h}$, allowing to apply G to a vector $\mathbf{l}_\mathrm{e}$ stored in a texture using a single indirect texture read. The vector $\mathbf{h}$ is pre-computed in two passes (Fig. 11).

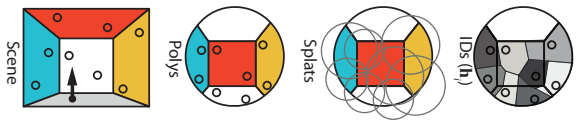

Figure 11: *Pre-computed visibility.* a): *The finite element points P* (circles) *of the surface $\mathcal{M}$* (polygons) *and one particular point element p* arrow. b): *View of the scene from p, including the other elements.* c): *The same view, but with the splat around every element.* d): *Grey-coding of the id j of the element p_j, that was closest to the surface location under q.*

Visibility from every point in every direction is resolved in a first pass. To this end, the scene's polygons are rasterized n_p times using paraboloid projection and depth-buffering from each $\mathbf{p}_i$ into a texture of size $\lceil\sqrt{n_\mathrm{d}}\rceil \times \lceil\sqrt{n_\mathrm{d}}\rceil$ that contains the nearest surface positions.

Second, the index of the element closest to each surface position in this texture is found. This is achieved by drawing every element $\mathbf{p}_j$ in P as a splat with a size that guarantees the covering of $\mathcal{M}$. For every pixel covered by this splat

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

with surface position p found in the first pass, the value j is written if, and only if, $d = ||\mathbf{p}_j - \mathbf{p}||$ is smaller than the one of all splats drawn before it. The conditional write is accomplished by using the distance d instead of the real depth value in a z-buffer. This 5D variant of the Voronoi construction proposed by Hoff et al. [HIKL*99] takes less than a second for a typical scene such as Fig. 1, and consequently allows for near-interactive editing of dynamic scenes.

Discussion The parameters n_p limits the spatial and n_d the angular resolution i. e., glossiness of rendering and editing; n_c limits the granularity of gloss control as shown in the supplemental material. Restricting G to be binary causes under-sampling, as multiple elements project into one bin and all but one are lost; a common problem for the hemi-cube in radiosity. If the emissive lighting is only from point lights, using $\mathbf{h}$ to apply G is replaced with shadow maps that use a more efficient discretization from the light's view in all our results.

6.2. Pre-convolved Radiance (K)

For a single element, applying K_s to $\mathbf{l}_\mathrm{i}$ in order to compute all A's would require as much as $n_\mathrm{c} n_\mathrm{d}^2$ operations: A loop that, for each glossiness level and each direction, visits every other direction. This cost can be reduced to constant time $4/3 n_\mathrm{d}$ by making two observations: First, applying $\mathsf{K}_\mathrm{s}(g)$ behaves as a lowpass filter with a cutoff proportional to g. Second, we need to know the result of applying all filters $\mathsf{K}_\mathrm{s}(g_0), \ldots, \mathsf{K}_\mathrm{s}(g_{n_\mathrm{c}})$ at the same time. Both can be achieved using recursive filtering [Wil83], which was used for environment maps [HS99] or radiance caching [SNRS12]. To this end, all that is required is a MIP map of a mirrored version of the incoming light $\mathbf{l}_\mathrm{i}$, denoted as $\hat{\mathbf{l}}_\mathrm{o}$. On level 0, such a map contains the reflected light only, corresponding to a mirror. On higher levels, increasingly blurred versions that correspond to lower glossiness values can be accessed in constant time. Diffuse reflection K_d is an extreme lowpass which can be stored in a small 4×4 texture.

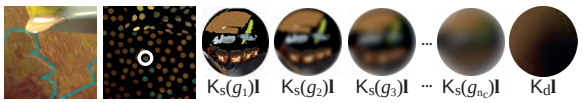

Figure 12: *Pre-convolution of one lumitexel from Fig. 1.*

Discussion Please note, that a different linear basis, such as Spherical harmonics will not allow to perform the reflection faster either [SNRS12]. Pre-convolved radiance caching will also be used to render the final result in Sec. 6.4.

6.3. Solver

Listing 1 summarizes our system in pseudocode. We implemented our approach using GLSL.

```
(P, r_r) := sample(M)
h := precalcVisibility(P, M)
r_f := r^m := r
while user interacts {
 for k := 0 to n_b { //Bounces
  for i := 0 to n_p inparallel { // Elements
   l_i := lookupVisibility(h, l_e)
   Î_o := createMIPMap(l_i)
   l_o := reflect(P, r_f,i, Î_o)
   b := tool(P, l_o)
   g := {g_1, g_2, ..., g_nc, r_g,i}
   X := {}
   for j := 0 to ||g|| inparallel { //Gloss
    A := (K_d * l_i, getMIPLevel(Î_o, g_j))
    C := A^T * W * A
    d := A^T * W * b
    X.add((sol_a(C, d), g_j)) //Eq. 3
    X.add((sol_d(C, d), g_j)) //Eq. 5
    X.add((sol_s(C, d), g_j)) //Eq. 6
    X.add((r_d,i, r_s,i, g_j))
   }
   r_i^m := alpha(X, l_o, Î_o) //Eq. 4
  }
  r_f := r^m
 }
}
```

Listing 1: *Pseudo-code of the mixed-solver approach.*

6.4. Rendering

Direct light is computed using common interactive rendering, e. g., shadow maps. To compute indirect reflected light efficiently [SNRS12], the MIP map $\hat{\mathbf{l}}_\mathrm{o}$ created in the solver is used as well. For improved quality, the correction used for environment maps by Szirmay-Kalosz et al. [SKALP05] is included. The rendering uses radiance cache splatting [GKBP08] to propagate the reflected lumitexel to a deferred framebuffer.

All radiometric units are stored as full-precission RGB float values into textures. We use a gamma tone-mapper that is applied forward when putting an image onto the screen and backwards when specifying colors. 2×2 supersampling is applied to all results.

6.5. Upsampling

While the solution for a discrete set of elements can be efficiently computed, $\mathcal{M}$ and R might contain fine details which are not represented well in P. To be able to propagate the manipulation to such details, joint bilateral upsampling in the framebuffer is used. Let Q be a frame buffer created by rasterizing the detailed scene. It typically contains millions of elements. We upsample P to Q, using radial basis function (RBF) reconstruction, with two important properties. First, we interpolate deltas in the form of differences of the original

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

reflectance and the new reflectance. This keeps fine details in Q instead of overwriting them. Second, the distance in appearance is included in the reconstruction kernel: Elements need have a similar original appearance to be changed in a similar way. Reconstruction is performed using RBF splatting

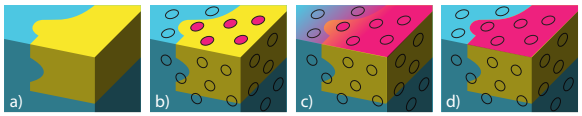

Figure 13: a): *A scene detail with appearance discontinuities.* b): *Five elements* (circles) *change to pink.* c): *Conventional upsampling according for normals and positions.* d): *Including appearance discontinuities preserves details.*

similar to the reconstruction used in rendering. At every 3D location p in P a screen-aligned quad is drawn, large enough to overlap with all similar pixels q. The RGB color of every quad is constant and encodes the change of shading parameters. The alpha value depends on the similarity between p and q. Additive blending in RGBA is used to combine multiple splats. After all splats are drawn, the RGB component in every pixel is divided by the alpha value for normalization.

7. Results

We report results in form of several use cases and a performance evaluation (please also see the supplemental video).

Usecases A basic manipulation is diffuse painting (Fig. 14).

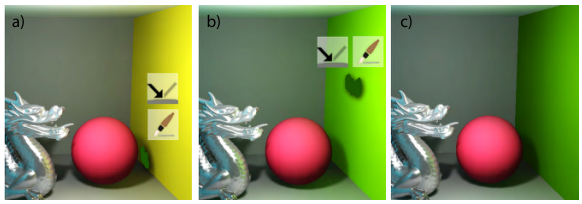

Figure 14: *Diffuse manipulation.* a): *A direct stroke in a shadow area and in the presence of GI.* b): *The reflectance is optimized resulting in a matching appearance under this complex illuminant. A new direct stroke is applied outside the shadow.* c): *The result again discounts for the illuminant.*

In the presence of multiple bounces, we find our iterative approach to converge against a minimum that is close to the desired result (cf. Fig. 15). While this optimum is likely not global, in our scenes a close local-minimum fit to the desired appearance was achieved.

A more advanced usage of our system is the design of view-dependent appearance (Fig. 16). To our knowledge, no attempts were made in previous work to "understand" the intention of such an input, which likely was to change specularity. Our mixed-$\mathcal{L}_p$ solver will detect that the best single change of material parameter is to adjust specularity.

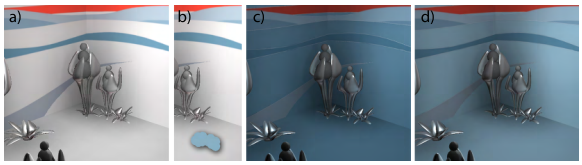

Figure 15: *Diffuse painting with multiple bounces.* a): *Input.* b): *A pale-blue stroke.* c): *After 1 iteration, the color is too dark because multiple bounces are not accounted for.* d): *Iterating the solver three times with the newly optimized reflectance, the appearance converges against the users prescription.*

Direct tuning of light color and reflectance of all surfaces that affect one surface location, can be a tedious process. Especially for diffuse surfaces, many other locations affect a single location. In Fig. 17, the desired appearance of a location subject to indirect lighting is changed and all other locations alter their reflectance to achieve the desired appearance. Indirect painting generalizes also to specular (glossy) surfaces (Fig. 18) .

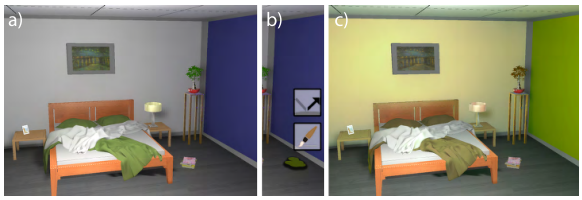

Figure 17: *Simple diffuse indirect painting.* a): *Input.* b): *An indirect green stroke.* c): *Reflectance of all other surfaces is changed to achieve the desired appearance.*

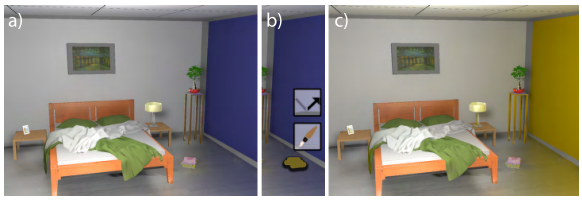

Figure 18: *Specular indirect painting.* a): *Input* b): *An indirect orange stroke is made in the input scene onto the ground.* c): *Reflectance is changed (right wall) where it contributes to the ground.*

Caustics are view-independent effects, but caused by light reflected from a specular object. Our system allows to tweak caustic appearance in an indirect way (Fig. 19). We do not assume any particular type of lighting, all that is required is a vector $\mathbf{l}_e$. This allows to design appearance under complex illumination, such as captured environment maps Fig. 20. In Fig. 21, a geometricaly detailled scene with detailled textures and bump maps is edited. Note how both material details are preserved, and manipulations are propagated to regions of similar appearance in the proximity of the stroke.

Besides sketching brushes, global stylization can be applied to

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

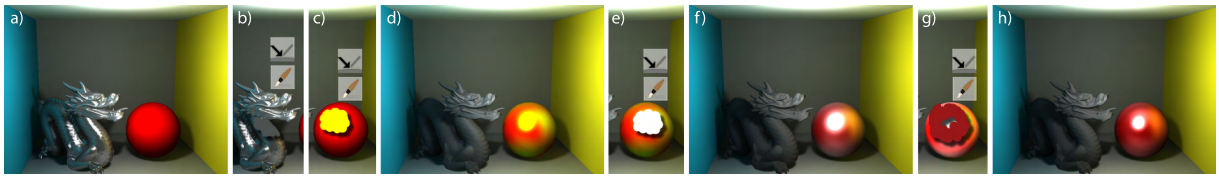

Figure 16: *Sketching specular reflectance:* a): *Input.* b): *A gray stroke over a highlight.* c): *A yellow stroke over the diffuse red ball.* d): *The dragon has turned diffuse; the ball has turned specular.* e): *A white stroke on the highlight.* f): *The ball now reflects more.* g): *A change of highlight shape.* h): *The system inferred a change of glossiness; reflections are sharp, the highlight small.*

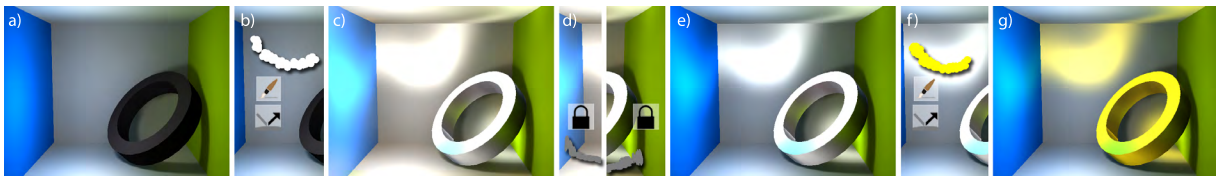

Figure 19: *Caustic design session:* a): *Input.* b): *White indirect stroke.* c): *Ambigue result. The system performs a change of reflectance on many surfaces.* d): *Two fixate strokes are made.* e): *Now, the best solution is to change the ring's material to white specular.*. f): *Yellow stroke on the caustic.* g): *The desired final appearance is achieved: a yellow caustic from a golden ring.*

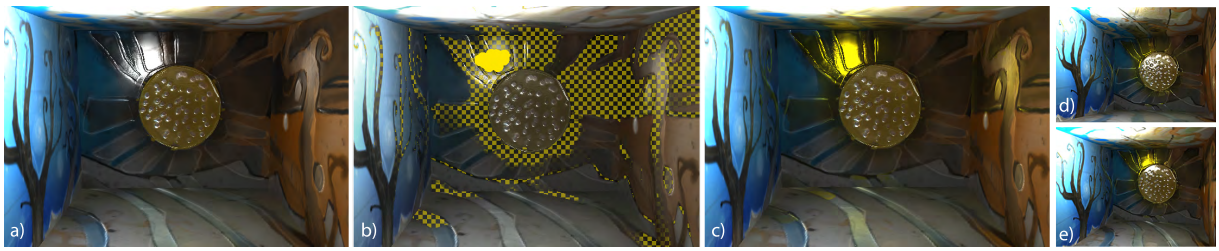

Figure 21: *Editing detailed appearance.* a): *Input.* b): *A stroke; similarity marked using 2D checkers.* c): *The change of specular.* d): *and* e): *The result from different views. Note the specular highlight appearing as yellow, including the details from the bump map.*

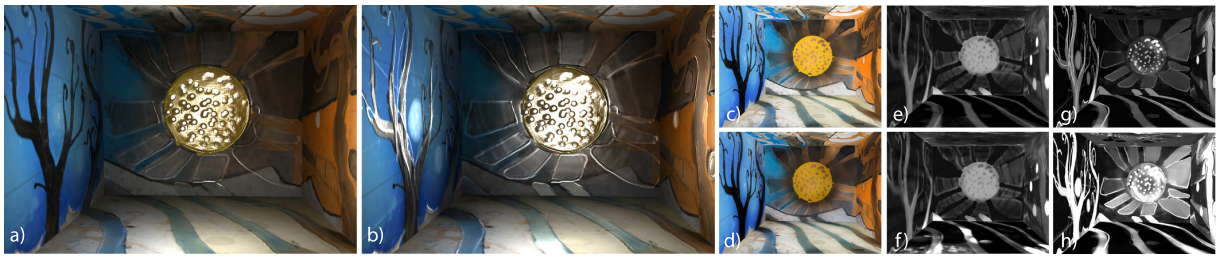

Figure 22: *Stylization of a scene* (a)*, using unsharp masking* (b) *by changing the original reflectance* (c,e,g) *into the new reflectance* (d,f,h)*, where* (c,d) *are diffuse components,* (e,f) *are specular components and* (g,h) *are the glossiness components.*

the SLF. The result is a scene appearance, a valid reflectance, but yet with an unsharp-masked look (see Fig. 22).

Performance For $n_p = 10000$ elements with a resolution of $n_d = 32 \times 32$ the solver takes around 270 ms. Reflectance upsampling from discrete points to current view take roughly 150 ms for a 900×450 resolution view port, radiance cache splatting is around 75 ms. In overall, our framework allows interactive feedback for a design session. A performance breakdown is showed in Table 1.

8. Conclusion and Future Work

We proposed a system to manipulate a SLF by painting it from different views and a solver that infers valid reflectance from this input. The result of our editing is scene-dependent, which is both a strength, but also a limitation that e. g., disallows to simply transfer appearance to a different scene. Our approach is better suited for adjusting, rather than creating something from scratch. The resulting surfaces with reflectance can be used in the following steps of a common pipeline, and are in theory even fabricable, which both is not the case when stylizing radiance alone. Transparent surfaces, would

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.

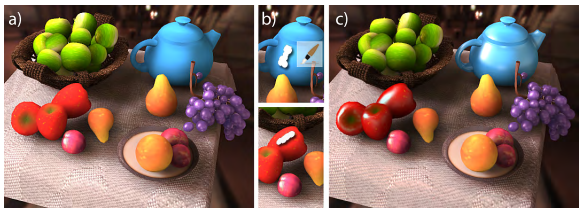

Figure 20: *Editing under enviroment map lighting.* a): *Input.* b): *White stroke onto the teapot interpreted as a highlight* (c).

Step	Time (ms)	Memory (MB)
Precomputed Vis.	870	31.64
MIP map creation	138	47.46
Tool	50	47.46
Solver	270	-
Upsampling	150	-
Radiance splatting	75	-

Table 1: *Performance breakdown for Fig. 1.*

require to exchange our pre-computed **G** with raytracing. There is a limit in rendering and editing detail which can be achieve using the proposed regular discretization. Beyond this limit, adaptive discretization would be required. Further research will be required to allow editing of shading models with multiple non-linear parameters. Finally, a user study comparing our approach with other alternatives is mandatory to assess if the proposed interface indeed is intuitive.

Our system is only one instance from a class of approaches where users loosely manipulate rendered 3D images (e. g., using strokes) and the system infers sparse and physically meaningful parameter changes. Extensions to other physically-based rendering, such as depth-of-field, motion blur, participating media or binocular stereo are exciting avenues of future research. Here, the laws of physics are considered not for the sake of accuracy, but act as a regularizer to make the users changes work together and behave in a consistent and plausible way.

References

[ATS94] ARVO J., TORRANCE K., SMITS B.: A framework for the analysis of error in global illumination algorithms. In *Proc. SIGGRAPH* (1994), pp. 75–84. 3

[BAEDR08] BEN-ARTZI A., EGAN K., DURAND F., RAMAMOORTHI R.: A precomputed polynomial representation for interactive BRDF editing with global illumination. *ACM Trans. Graph. 27*, 2 (2008), 13:1–13:13. 2

[BAOR06] BEN-ARTZI A., OVERBECK R., RAMAMOORTHI R.: Real-time BRDF editing in complex lighting. *ACM Trans. Graph. 25*, 3 (2006), 945–954. 2, 3

[CPK06] COLBERT M., PATTANAIK S., KRIVANEK J.: BRDF-Shop: Creating physically correct bidirectional reflectance distribution functions. *Comp. Graph. & App. 26*, 1 (2006), 30–36. 2

[CPWAP08] CHESLACK-POSTAVA E., WANG R., AKERLUND O., PELLACINI F.: Fast, realistic lighting and material design using nonlinear cut approximation. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 27*, 5 (2008), 128:1–128:10. 2

[Don06] DONOHO D.: Compressed sensing. *Inf. Theory, IEEE Trans. 52*, 4 (2006), 1289–1306. 5

[GKBP08] GAUTRON P., KŘIVÁNEK J., BOUATOUCH K., PATTANAIK S.: Radiance cache splatting: A gpu-friendly global illumination algorithm. In *ACM SIGGRAPH 2008 Classes* (2008), ACM, p. 78. 7

[HC07] HORN D., CHEN B.: LightShop: Interactive light field manipulation and rendering. In *Proc. I3D* (2007). 2

[HIKL*99] HOFF III K., KEYSER J., LIN M., MANOCHA D., CULVER T.: Fast computation of generalized voronoi diagrams using graphics hardware. In *Proc. SIGGRAPH* (1999), pp. 277–286. 7

[HS99] HEIDRICH W., SEIDEL H.: Realistic, hardware-accelerated shading and lighting. In *Proc. SIGGRAPH* (1999), pp. 171–178. 7

[ICG86] IMMEL D., COHEN M., GREENBERG D.: A radiosity method for non-diffuse environments. *Proc. SIGGRAPH 20*, 4 (1986), 133–142. 4

[KP10] KERR W. B., PELLACINI F.: Toward evaluating material design interface paradigms for novice users. *ACM Trans. Graph. 29* (2010), 35:1–35:10. 1, 2

[KPC93] KAWAI J. K., PAINTER J. S., COHEN M. F.: Radioptimization: goal based rendering. In *Proc. SIGGRAPH* (1993), pp. 147–154. 2

[Lan99] LANGER M. S.: When shadows become interreflections. *J Comp. Vis. 34*, 2 (1999), 193–204. 2

[LM71] LAND E. H., MCCANN J. J.: Lightness and retinex theory. *J OSA 61*, 1 (1971), 1–11. 2

[NKLN10] NGUYEN C. H., KYUNG M.-H., LEE J.-H., NAM S.-W.: A PCA decomposition for real-time BRDF editing and relighting with global illumination. *Comp. Graph. Forum (Proc. EGSR) 29*, 4 (2010), 1469–1478. 2

[NRE*12] NGUYEN C. H., RITSCHEL T., EISEMANN E., MYSZKOWSKI K., SEIDEL H.-P.: 3D material style transfer. *Comp. Graph. Forum (Proc. EG) 29*, 4 (2012), 1469–1478. 1

[PF95] POULIN P., FOURNIER A.: Painting surface characteristics. In *Proc. EGWR* (1995), pp. 160–9. 2

[PFG00] PELLACINI F., FERWERDA J. A., GREENBERG D. P.: Toward a psychophysically-based light reflection model for image synthesis. In *Proc. SIGGRAPH* (2000), pp. 55–64. 2, 5

[PL07] PELLACINI F., LAWRENCE J.: AppWand: Editing measured materials using appearance-driven optimization. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3 (2007), 54. 2

[PTG02] PELLACINI F., TOLE P., GREENBERG D. P.: A user interface for interactive cinematic shadow design. *ACM Trans. Graph. 21*, 3 (2002), 563–566. 2

[SDS*93] SCHOENEMAN C., DORSEY J., SMITS B., ARVO J., GREENBERG D.: Painting with light. In *Proc. SIGGRAPH* (1993), pp. 143–146. 2

[SKALP05] SZIRMAY-KALOS L., ASZÓDI B., LAZÁNYI I., PREMECZ M.: Approximate ray-tracing on the GPU with distance impostors. *Comp. Graph. Forum 24*, 3 (2005), 695–704. 7

[SNRS12] SCHERZER D., NGUYEN C. H., RITSCHEL T., SEIDEL H.-P.: Pre-convolved Radiance Caching. *Comp. Graph. Forum (Proc. EGSR 2012) 4*, 31 (2012). 7

[WAA*00] WOOD D., AZUMA D., ALDINGER K., CURLESS B., DUCHAMP T., SALESIN D., STUETZLE W.: Surface light fields for 3D photography. In *Proc. SIGGRAPH* (2000). 2

[Wil83] WILLIAMS L.: Pyramidal parametrics. *Proc. SIGGRAPH 17*, 3 (1983), 1–11. 7

© 2013 The Author(s)
© 2013 The Eurographics Association and Blackwell Publishing Ltd.