Surface Flows for Image-based Shading Design

Romain Vergne, Pascal Barla, Roland Fleming, Xavier Granier

HAL Id: hal-00702280
https://hal.inria.fr/hal-00702280v2

Submitted on 4 Sep 2012

Surface Flows for Image-based Shading Design

Romain Vergne[1]*   Pascal Barla[2]   Roland W. Fleming[1]   Xavier Granier[2]

[1]Justus-Liebig-Universität Gießen   [2]Inria - U. Bordeaux - IOGS - CNRS

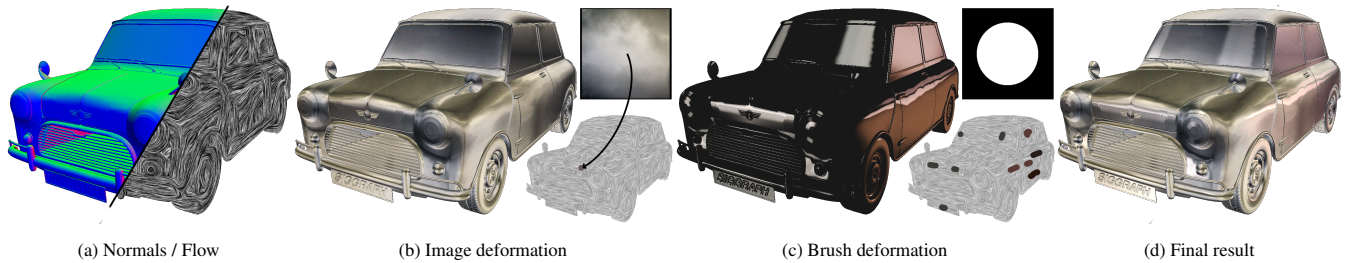| (a) Normals / Flow | (b) Image deformation | (c) Brush deformation | (d) Final result |

Figure 1: *Our system produces convincing shading results by working purely from image deformations. Here, a surface flow is extracted from a normal image (a). It is first used to deform an entire image that serves as a reflective base coat (b). Then it is employed to deform brush strokes that represent highlights (c). The two components are added together to convey a stylized shading appearance (d).*

Abstract

We present a novel method for producing convincing pictures of shaded objects based entirely on 2D image operations. This approach, which we call *image-based shading design*, offers direct artistic control in the picture plane by deforming image primitives so that they appear to conform to specific 3D shapes. Using a differential analysis of reflected radiance, we identify the two types of *surface flows* involved in the depiction of shaded objects, which are consistent with recent perceptual studies. We then introduce two novel *deformation operators* that closely mimic surface flows while providing direct artistic controls in real-time.

CR Categories: I.3.3 [Computing Methodologies]: Computer Graphics—Picture/Image Generation

Keywords: Shading, differential analysis, image deformation.

Links: ◈DL 📄PDF ◉WEB ◉VIDEO ⬇CODE

1 Introduction

It has long been known that texture gradients and shading patterns are critically important monocular cues to 3D shape (*e.g.*, [Gibson 1950; Koenderink and Van Doorn 1980; Todd and Mingolla 1983; Todd and Akerstrom 1987]). However, how the brain extracts shape from these cues is still poorly understood. Recently, some authors have suggested that the visual system extracts certain shape and material properties directly from the spatial patterns of intensity gradients in the image (*e.g.*, [Breton and Zucker 1996; Li and Zaidi 2000; Huggins and Zucker 2001]). In particular, Fleming *et al.* [2004] found that for typical surfaces, the local shading orientation signals are organized into smoothly-varying *distortion patterns*, which are systematically related to 3D surface shape. Since then, computer vision researchers have shown such

---

*e-mail: romain.vergne@gmail.com

flows can be used to identify specular surfaces and reconstruct their shape [DelPozo and Savarese 2007; Vasilyev et al. 2011; Tappen 2011]. More recently, Fleming *et al.* [2011a] found that similar measurements also play a key role in perceiving shape from texture patterns. However, for textures, the image distortion measurements indicate surface orientation (i.e., first-order properties) rather than curvatures (i.e., second-order properties) as they do for shading and specular reflections. A number of researchers have also suggested that different distortion patterns—and the intensity gradients within them—strongly affect not only perceived surface shape, but also perceived surface material (*e.g.*, [Motoyoshi et al. 2005; Anderson and Kim 2009; Kim and Anderson 2010; Fleming et al. 2011b]).

The main objective of this paper is to take these observations one step further: by deforming images in 2D to produce appropriate distortion patterns, it should be possible to reliably create the illusion of 3D shapes and materials, without simulating the physics of light transport. In spirit, this approach is similar to Image Based Material Editing [Khan et al. 2006], except that instead of altering the appearance of objects in existing images, we aim to allow the user also to create novel images. Like [Ramanarayanan et al. 2007] we exploit the fact that the human visual system is highly sensitive to certain types of distortion—those that indicate object properties—while being almost completely oblivious to large deviations from physical accuracy in other respects. This approach, which we call *image-based shading design*, raises two intricately linked issues. First, we must identify what kind of deformation operators lead to perceptually plausible shape and material depiction. For this purpose we assume that normal and depth images are given as input and relied upon to produce appropriate deformations. This is a reasonable assumption since a variety of solutions exist for creating normal and depth maps from 3D scenes, color images and even from scratch (*e.g.*, [Johnston 2002; Toler-Franklin et al. 2007; Wu et al. 2007; Quixel 2011]). Second, we must give artists control over the form and magnitude of the deformation, and the content of the images being deformed, to alter the resulting shading patterns.

We address these issues by making the following contributions:

- Using a differential analysis of reflected radiance in the picture, we identify the two different types of *surface flows* at the origin of the distortion patterns observed in shaded images.

- We introduce image-based shading tools that faithfully convey shape and material properties while providing accurate artistic control directly in the picture plane. This is made possible thanks to novel *deformation operators* that mimic information found in surface flows.

2 Related work

Drawing is the technique that gives artists the most immediate control for creating and manipulating appearance directly in pictorial space. However, the creation of complex (e.g., photoreal) pictures demands exceptional drawing skills and a huge amount of time. To address this issue, a number of computer-assisted drawing techniques have been introduced. For instance, when working with vector graphics, the production of smooth-shaded images is greatly helped by tools such as Gradient Meshes (*e.g.*, [Sun et al. 2007]) or Diffusion Curves [Orzan et al. 2008], that provide options for using photographs as references. Although these have been primarily used for the creation of color gradients, they might be extended to the production of texture patterns as exemplified in the work of Jeschke *et al.* [2011]. Similar approaches have been proposed for raster graphics drawing applications, such as Gradient-domain Painting [McCann and Pollard 2008]. Although these methods considerably speed up the image creation process, they still require advanced skills and become time-consuming with increasing image complexity. This may be made easier by using a photograph as a reference for shading and textures, but then the choice of materials and illumination is tied to the ones found in the input photograph.

Rendering is the natural alternative when one wants to freely vary material and illumination parameters. However, controlling appearance directly in the picture plane requires specific solutions. Inverse rendering techniques [Okabe et al. 2007; Pellacini et al. 2007; Pellacini 2010] rely on known material characteristics and user scribbles to infer plausible illumination environments through a constrained optimization process. Conversely, shaded appearance may be designed through a painting interface by inferring material characteristics under known illumination conditions, using optimization as well. It has been demonstrated for both realistic [Colbert et al. 2006] and stylized shading [Anjyo et al. 2006; Pacanowski et al. 2008]. Although they considerably reduce the amount of trial and error required to reach a satisfying result, both kinds of approaches still offer a limited control on final appearance and are restricted either to physically-realistic renderings or specific stylized appearances. Ritschel *et al.* [2009; 2010] provide solutions to depart from the laws of physics by deforming shading effects such as reflections, shadows or textures with user-controlled handles. However, their methods are still bound to the specification of a 3D scene (objects, lights and materials). Therefore, shading effects cannot be created as with drawing applications, but only modified.

Image-based techniques represent intermediate solutions in terms of artistic control and flexibility. Although they may start from 3D-like information such as normal or depth images, they do not require a full simulation of light transport in a 3D scene. For instance, images of textured objects may be created either by segmenting a normal image into patches onto which the texture is synthesized and aligned [Fang and Hart 2004], or by applying parallax mapping to a depth image integrated from normals [Winnemöller et al. 2009]. Many image-based techniques focus on editing existing images to create new content. For example, [Oh et al. 2001] provided a set of intuitive tools for segmenting, reconstructing and editing depth and color maps from single uncalibrated images enabling a wide range of radical changes, including relighting, removal and copying of content in 3D and viewing from new viewpoints. Image-based material editing [Khan et al. 2006] is another example: depth and normal images are first estimated from a color image of an object; then its material is modified by means of image processing techniques. The approach provides compelling results but remains limited in the kinds of manipulations it affords: reflections and refractions cannot be positioned freely on the object and one has to work with information found in the image (mainly the background). The method of Yeung *et al.* [2011] provides a more flexible artistic control whereby users can sketch distortion patterns due to refraction, but it is restricted to the matting of transparent objects. The Lit Sphere technique [Sloan et al. 2001] may also be considered as an image-based shading design technique, although it is strongly limited in terms of artistic control, since it relies on a fixed global mapping from screen-space normal coordinates to the image of a lit sphere.

3 Overview

Our approach belongs to the image-based category, and like most existing methods it takes normal and depth images as input. Contrary to previous work though, it is not limited to a single shading effect and provides direct artistic control in the picture.

To reach this goal, we first demonstrate in Section 4 that two kinds of surface flows are at the origin of distortion patterns found in rendered images. This is done by conducting a differential analysis of the reflected radiance equation in the vein of [Ramamoorthi et al. 2007], with the important difference that it is performed directly in the picture. This is crucial to our approach, since it explains distortion patterns observed in images of shaded objects in terms of depth and normal variations. We then describe in Section 5 how the gained insights are used in deformation operators to trade-off artistic control versus perceptual plausibility, and we illustrate their usefulness through three types of shading tools. Combining these tools, we produce a variety of shading appearances as shown in Section 6. We discuss limitations and potential extensions to our approach in Section 7.

4 Surface flows

The main claim of this paper is that it is possible to elicit a powerful illusion of complex shapes or materials without accurately simulating the physics of light transport; one only needs to convey the proper visual cues to the human visual system. But which cues exactly? As mentioned in the introduction, a growing body of evidence suggests that the visual system is particularly sensitive to distortion patterns in the image, which are mostly measurable through *spatial variations* of luminance. This naturally leads us to perform a differential analysis of the rendering process, with the goal in mind to identify how surface shape participates in the pictorial appearance of a 3D object.

4.1 Definitions

We focus on the analysis of reflected radiance L_r, defined at each point (x, y) of the picture plane by:

$$L_r(x, y) = \int_{\Omega_{\mathbf{n}}^+} \rho_{\mathbf{n}}(\mathbf{p}, \mathbf{v}, \omega)\, L_i(\mathbf{p}, \omega)\, (\mathbf{n} \cdot \omega)\, d\omega, \quad (1)$$

where ω is an incoming light direction, $\mathbf{n}(x, y)$ is the outward surface normal at the point $\mathbf{p}(x, y)$ on the surface, $\mathbf{v}(x, y)$ is the view direction and $\Omega_{\mathbf{n}}^+$ is the hemisphere oriented around $\mathbf{n}$. Material characteristics are provided through $\rho_{\mathbf{n}}$, the (potentially spatially-varying) Bidirectional Reflectance Distribution Function [Nicodemus et al. 1977], or BRDF for short. Direct and indirect illumination are encapsulated inside the incoming radiance L_i.

Unfortunately, Equation 1 is not directly amenable to differentiation since the domain of integration $\Omega_{\mathbf{n}}^+$ varies itself in the picture plane. This is fixed with a change of variable $\ell \leftarrow \mathcal{R}_{\mathbf{n}}\omega$, where $\mathcal{R}_{\mathbf{n}}$ is the smallest rotation that aligns the normal $\mathbf{n}$ with the $\mathbf{z}$ axis. In the following, we consider only isotropic materials, hence $\mathcal{R}_{\mathbf{n}}$ is well-defined. As detailed in the Appendix, Equation 1 then becomes:

$$L_r(x, y) = \int_{\Omega_{\mathbf{z}}^+} \rho_{\mathbf{z}}(\mathbf{p}, \mathcal{R}_{\mathbf{n}}\mathbf{v}, \ell)\, L_i(\mathbf{p}, \mathcal{R}_{\mathbf{n}}^{-1}\ell)\, (\mathbf{z} \cdot \ell)\, d\ell, \quad (2)$$

where $\Omega_{\mathbf{z}}^+$ is the (constant) hemisphere oriented around the $\mathbf{z}$ axis, and $\rho_{\mathbf{z}}$ is the BRDF defined around $\mathbf{z}$. Note that this formulation is still independent of the choice of material, since one only has to replace $\mathbf{n}$ by $\mathbf{z}$ in the chosen BRDF model.

4.2 Differentiation

With this new formulation, we are now able to compute derivatives of L_r as combinations of derivatives of the terms found *inside* the integral. For ease of notation, we will omit function variables and write Equation 2 as $L_r = \int_{\Omega_{\mathbf{z}}^+} \rho_{\mathbf{z}} L_i \, dF$, where $dF = (\mathbf{z} \cdot \ell)d\ell$ is independent of the position in the picture. The derivative of the reflected radiance L_r along the horizontal direction $\mathbf{x}$ in the picture plane is obtained by applying the product rule inside the integral:

$$\partial_{\mathbf{x}} L_r = \int_{\Omega_{\mathbf{z}}^+} L_i \; \partial_{\mathbf{x}} \rho_{\mathbf{z}} \, dF + \int_{\Omega_{\mathbf{z}}^+} \rho_{\mathbf{z}} \; \partial_{\mathbf{x}} L_i \, dF, \quad (3)$$

where $\partial_{\mathbf{x}}$ stands for $\partial/\partial\mathbf{x}$. Material variations $\partial_{\mathbf{x}}\rho_{\mathbf{z}}$ and illumination variations $\partial_{\mathbf{x}} L_i$ are obtained by applying the chain rule:

$$\partial_{\mathbf{x}}\rho_{\mathbf{z}} = (\partial_{\mathbf{x}}\mathbf{p} \cdot \nabla_{\mathbf{p}}\rho_{\mathbf{z}}) + (\partial_{\mathbf{x}}\mathcal{R}_{\mathbf{n}}\mathbf{v} \cdot \nabla_\nu\rho_{\mathbf{z}}), \quad (4)$$

$$\partial_{\mathbf{x}} L_i = (\partial_{\mathbf{x}}\mathbf{p} \cdot \nabla_{\mathbf{p}} L_i) + (\partial_{\mathbf{x}}\mathcal{R}_{\mathbf{n}}^{-1}\ell \cdot \nabla_1 L_i), \quad (5)$$

where $\nabla_\nu\rho_{\mathbf{z}}$ is the derivative of $\rho_{\mathbf{z}}$ in the viewing direction, and $\nabla_1 L_i$ is the derivative of L_i in the incident lighting direction. For the sake of simplicity, we have considered that the projection is locally orthographic, and hence $\partial_{\mathbf{x}}\mathbf{v} = 0$.

Variations due to surface shape appear in both material and illumination derivatives, and are of two different kinds: they represent either first-order variations ($\partial_{\mathbf{x}}\mathbf{p}$) or second-order variations ($\partial_{\mathbf{x}}\mathcal{R}_{\mathbf{n}}^{\pm1}$). Although the meaning of the former is easily grasped (and depends only on depth variations with our local orthographic approximation), the understanding of the latter is indirect since it involves derivatives of the rotation $\mathcal{R}_{\mathbf{n}}$ and its inverse. As detailed in the Appendix, we rewrite the derivatives of $\mathcal{R}_{\mathbf{n}}$ and $\mathcal{R}_{\mathbf{n}}^{-1}$ in terms of cross products that directly involve normal variations:

$$\partial_{\mathbf{x}}\mathcal{R}_{\mathbf{n}}\mathbf{v} = \mathcal{R}_{\mathbf{n}}\partial_{\mathbf{x}}\mathbf{n} \times \mathbf{n} \times \mathbf{v}, \quad (6)$$

$$\partial_{\mathbf{x}}\mathcal{R}_{\mathbf{n}}^{-1}\ell = -\partial_{\mathbf{x}}\mathbf{n} \times \mathbf{n} \times \mathcal{R}_{\mathbf{n}}^{-1}\ell, \quad (7)$$

where $\partial_{\mathbf{x}}\mathbf{n} \times \mathbf{n}$ corresponds to the axis of the minimal rotation from $\mathbf{n}$ to $\mathbf{n} + \partial_{\mathbf{x}}\mathbf{n}$, and its magnitude represents the corresponding rotation angle. Aside from a more direct interpretation, this formulation affords better segregation between first- and second-order surface structures, as explained in the next section.

4.3 Decomposition

The analysis conducted so far has identified two types of variations due to surface shape: first-order ($\partial_{\mathbf{x}}\mathbf{p}$) and second-order ($\partial_{\mathbf{x}}\mathbf{n} \times \mathbf{n}$). We now show how these terms are factored out to yield a decomposition of luminance variations in image space.

We begin by inserting Equations 6 and 7 inside Equations 4 and 5 respectively. The first-order term $\partial_{\mathbf{x}}\mathbf{p}$ is then directly taken out of the integral. After re-ordering the scalar triple products in Equations 4 and 5, the second-order term $\partial_{\mathbf{x}}\mathbf{n} \times \mathbf{n}$ is taken out of the integral as well. This leads to the following compact formulation (see supplemental material for a detailed derivation):

$$\boxed{\partial_{\mathbf{x}} L_r = (\partial_{\mathbf{x}}\mathbf{p} \cdot V_{\mathbf{p}}) + (\partial_{\mathbf{x}}\mathbf{n} \times \mathbf{n} \cdot V_{\mathbf{n}}).} \quad (8)$$

$V_{\mathbf{p}}$ and $V_{\mathbf{n}}$ give spatial and angular variations respectively:

$$V_{\mathbf{p}} = \int_{\Omega_{\mathbf{z}}^+} \underbrace{\nabla_{\mathbf{p}}\rho_{\mathbf{z}} \; L_i}_{\text{textures}} + \underbrace{\nabla_{\mathbf{p}} L_i \; \rho_{\mathbf{z}}}_{\text{spatial illum.}} \, dF, \quad (9)$$

$$V_{\mathbf{n}} = \int_{\Omega_{\mathbf{z}}^+} \underbrace{(\mathbf{v} \times \mathcal{R}_{\mathbf{n}}^{-1}\nabla_\nu\rho_{\mathbf{z}}) L_i}_{\text{reflections}} - \underbrace{(\mathcal{R}_{\mathbf{n}}^{-1}\ell \times \nabla_1 L_i)\rho_{\mathbf{z}}}_{\text{angular illum.}} \, dF \quad (10)$$

Equation 8 constitutes the main result of this section. It shows that first- and second-order surface variations have *distinct* effects on luminance variations, and that this distinction is *invariant* to the choice of materials or lighting environments. So far, we have only considered derivatives in the $\mathbf{x}$ direction, but the same analysis holds for the $\mathbf{y}$ direction. Taken together, this provides us with a pair of Jacobians $[\partial_{\mathbf{x}}\mathbf{p}, \partial_{\mathbf{y}}\mathbf{p}]^T$ and $[\partial_{\mathbf{x}}\mathbf{n} \times \mathbf{n}, \partial_{\mathbf{y}}\mathbf{n} \times \mathbf{n}]^T$, which we call *surface flows* to emphasize the distortion effects they impose on other types of 3D variations.

The first-order flow is related to 3D *spatial* variations, as made explicit by Equation 9: $\nabla_{\mathbf{p}}\rho_{\mathbf{z}}$ represents variations of BRDF properties across the object (commonly called "texture"), while $\nabla_{\mathbf{p}} L_i$ represents spatial illumination variations due to shadowing or inter-reflections. The second-order flow is related to 3D *angular* variations as made explicit by Equation 10: $\nabla_\nu\rho_{\mathbf{z}}$ represents variations of the BRDF for a varying viewpoint (hence conveying reflections), while $\nabla_1 L_i$ represents angular illumination variations due to the environment. Both $V_{\mathbf{p}}$ and $V_{\mathbf{n}}$ deserve further investigation and should be connected to perceptual findings that have recently been made thanks to experiments on shape-from-texture, shape-from-shadows and shape-from-reflections. However, in this paper, we mainly focus on surface flows since we are interested in the production of believable distortion patterns. We will thus replace $V_{\mathbf{p}}$ and $V_{\mathbf{n}}$ altogether with data coming from images and leave their studies to future work. This solution enables the creation of powerful design operators, as demonstrated in Section 5.

Before we move on to shading design, we need an intuitive way to visualize surface flows in an image. A common solution to represent a Jacobian consists in converting it to a structure tensor [Bigün and Granlund 1987], which corresponds here to a 2×2 covariance matrix, from which directions and magnitudes of maximal and minimal change in the image are extracted with an eigen decomposition. The first- and second-order structure tensors $T_{\mathbf{p}}$ and $T_{\mathbf{n}}$ are defined in terms of the directional derivatives of $\mathbf{p}$ and $\mathbf{n}$:

$$T_{\mathbf{p}} = \begin{bmatrix} \partial_{\mathbf{x}}\mathbf{p} \cdot \partial_{\mathbf{x}}\mathbf{p} & \partial_{\mathbf{x}}\mathbf{p} \cdot \partial_{\mathbf{y}}\mathbf{p} \\ \partial_{\mathbf{x}}\mathbf{p} \cdot \partial_{\mathbf{y}}\mathbf{p} & \partial_{\mathbf{y}}\mathbf{p} \cdot \partial_{\mathbf{y}}\mathbf{p} \end{bmatrix}; \; T_{\mathbf{n}} = \begin{bmatrix} \partial_{\mathbf{x}}\mathbf{n} \cdot \partial_{\mathbf{x}}\mathbf{n} & \partial_{\mathbf{x}}\mathbf{n} \cdot \partial_{\mathbf{y}}\mathbf{n} \\ \partial_{\mathbf{x}}\mathbf{n} \cdot \partial_{\mathbf{y}}\mathbf{n} & \partial_{\mathbf{y}}\mathbf{n} \cdot \partial_{\mathbf{y}}\mathbf{n} \end{bmatrix}.$$

Two important observations must be made. First, the cross product with $\mathbf{n}$ does not appear in $T_{\mathbf{n}}$, as explained in the Appendix. It suggests that this cross product has little effect on the pattern of deformations induced by normal variations. Second, $T_{\mathbf{p}}$ and $T_{\mathbf{n}}$ do *not* directly correspond to the fundamental forms of Riemannian geometry, since they characterize variations in image space as opposed to variations in tangent space. For a more detailed discussion on this topic, we refer the reader to [Fleming et al. 2009].

We visualize both types of tensors with a combination of line-integral convolution (LIC) and a color code as shown in Figure 2. Visual comparison between compression and stretching patterns in textured or shaded images on the one hand (a), and the representation of their corresponding surface flows on the other hand (b), reveals a strong correlation as expected.

5 Shading Design

Having identified the surface flows responsible for distortion patterns found in shading and texture, we are now left with a second challenge: how to convey such distortions while still providing direct artistic control? For this purpose, we introduce in Section 5.1
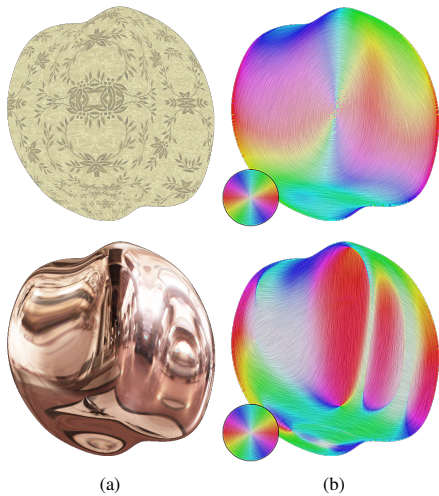
Figure 2: *3D rendering:* *conventional texture (top) and environment (bottom) mappings (a) produce variations related to first- and second-order surface flows. They are visualized in (b) where hue and saturation indicate direction and magnitude of the flow respectively.*
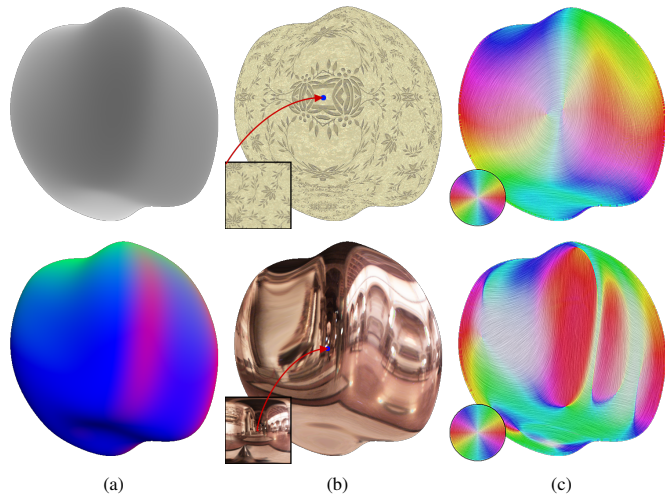
Figure 3: *Image deformation:* *our shading design system mimics texture (top) and shading (bottom) effects using image processing alone. It takes depth and normal images as input (a), and uses them to deform images (b) in ways that closely approximate surface flows (c). It provides a convincing, yet artistically controllable illusion of 3D shape conveyed through texture or shading cues.*

two novel image deformation operators, one for each surface flow. We then show in Section 5.2 how they might be incorporated into three example drawing tools, each having its own way of conveying additional texture or material properties. Note that the equations presented throughout Section 5 are heuristics inspired by Section 4 and intended to approximate surface flows rather than reproduce them perfectly.

5.1 Deformation operators

Let I be the image we wish to deform. We look for a pair of deformation operators which, once applied to I, produce distortion patterns similar to those of first- and second-order surface flows. One may think that $T_{\mathbf{p}}$ and $T_{\mathbf{n}}$ could be used for this purpose, but this is not straightforward: structure tensors only provide non-oriented distortion fields (by construction), and it is not clear how they could be adapted for artistic control since they provide only local information.

Instead, our approach consists in finding user-controllable deformation operators $\mathcal{D}_{\mathbf{p}}$ and $\mathcal{D}_{\mathbf{n}}$ whose derivatives mimic surface flows. Formally, for a point $\mathbf{q} = (x, y)$ in the image, its color after deformation is obtained through composition with one of these operators: $I_{\mathbf{p}|\mathbf{n}}(\mathbf{q}) = I(\mathcal{D}_{\mathbf{p}|\mathbf{n}}(\mathbf{q}))$, where $\mathbf{p}|\mathbf{n}$ is a shorthand notation for "$\mathbf{p}$ or $\mathbf{n}$". The derivative in the $\mathbf{x}$ direction of the image *deformation* process is then given by:

$$\partial_{\mathbf{x}} I_{\mathbf{p}|\mathbf{n}} = (\partial_{\mathbf{x}} \mathcal{D}_{\mathbf{p}|\mathbf{n}} \cdot \partial_{\mathbf{x}} I(\mathcal{D}_{\mathbf{p}|\mathbf{n}})). \quad (11)$$

The similarities between Equations 11 and 8 are important. On one side, variations due to deformations $\partial_{\mathbf{x}} \mathcal{D}_{\mathbf{p}}$ or $\partial_{\mathbf{x}} \mathcal{D}_{\mathbf{n}}$ are meant to take the role of $\partial_{\mathbf{x}} \mathbf{p}$ or $\partial_{\mathbf{x}} \mathbf{n}$ respectively. On the other side, variations due to the input image $\partial_{\mathbf{x}} I$ are designed to mimic 3D spatial ($V_{\mathbf{p}}$) or angular ($V_{\mathbf{n}}$) variations. Since our focus is on surface flows, these latter variations will be replaced by variations found in (possibly filtered) images, as shown in Section 5.2.

On top of reproducing 3D-like surface variations, we aim to provide artists with controls over deformation operators. First and foremost, we want to enable direct manipulation through a few *anchor points* $\mathbf{q}_i$, where the deformation is required to have no effect (i.e.,

$\mathcal{D}_{\mathbf{p}|\mathbf{n}}(\mathbf{q}_i) = \mathbf{q}_i$). Moreover, although the direction of deformation is entirely determined by input normal and depth maps in our implementation, its magnitude should remain controllable with a few parameters. With this trade-off in mind, we have empirically designed the following operators.

First-order deformations Considering a single anchor point $\mathbf{q}_0$ for simplicity, we want to locally deform I in a way that preserves its overall structure, in particular its orientation. Consequently, we restrict deformation to a 2D offset of $\mathbf{q}$ proportional to $(\mathbf{q} - \mathbf{q}_0)$, with an offset factor $f_{\mathbf{p}}$ that depends on the depth difference between $\mathbf{q}$ and $\mathbf{q}_0$. We use depth instead of full positions for two reasons: depth images are more easily obtained; and depth differences are signed which provides for more intuitive artistic controls. In the simple case of a single anchor point, the offset vector is given by $f_{\mathbf{p}}(z - z_0)(\mathbf{q} - \mathbf{q}_0)$, where depth values z and z_0 lie in the $[0, 1]$ range and

$$f_{\mathbf{p}}(u) = \frac{2u(e^{\gamma} - 1)}{(e^{\gamma} + 1) - u(e^{\gamma} - 1)}. \quad (12)$$

As shown in Figure 4(a), $f_{\mathbf{p}}$ is a function of relative depth controlled via a single parameter $\gamma \in \mathbb{R}^+$. This formula has been designed empirically to provide more natural distortions whenever a distant anchor point is used. Compared to a simple linear function, it prevents extensive distortions on front surfaces (i.e. when $z - z_0 < 0$) while still preserving the overall magnitude of deformation (Figure 5).

Generalizing this idea to multiple anchor points, we obtain the following formula for first-order deformations:

$$\mathcal{D}_{\mathbf{p}}(\mathbf{q}) = \mathbf{q} + \sum_i w_i(\mathbf{q}) f_{\mathbf{p}}(z - z_i)(\mathbf{q} - \mathbf{q}_i), \quad (13)$$

where the w_i are used to balance the influence of anchor points. When applied to the whole picture, $\mathcal{D}_{\mathbf{p}}$ either inflates ($f_{\mathbf{p}} < 0$) or deflates ($f_{\mathbf{p}} > 0$) the image I. It closely mimics first-order surface variations, as shown in the top rows of Figures 2 and 3, where we compare $T_{\mathbf{p}}$ to structure tensors obtained from $\partial_{\mathbf{x}} \mathcal{D}_{\mathbf{p}}$ and $\partial_{\mathbf{y}} \mathcal{D}_{\mathbf{p}}$.

Second-order deformations Considering once again a single anchor point $\mathbf{q}_0$, the second-order operator is obtained with a sim-
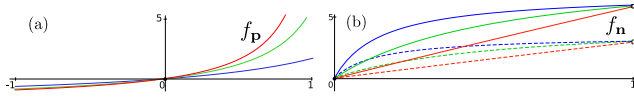
Figure 4: *Offset functions:* *(a) Depth-based offset functions with* $\gamma = \{1, 2, 3\}$. *The non-linear behavior is chosen to mimic the non-linearity of depth in 3D scenes. Note the invariant* $f_{\mathbf{p}}(0) = 0$; *(b) Normal-based offset functions with* $\beta = \{1, 2, 3\}$ *for* $\alpha = 3$ *(dashed curve) and* $\alpha = 6$ *(solid curve). Non-linearity is controlled by the* β *parameter. Note again the invariant* $f_{\mathbf{n}}(0) = 0$.

(a) Linear   (b) Non-linear   (c) Linear   (d) Non-linear
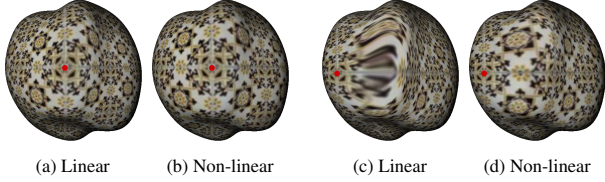
Figure 5: *Linear vs non-linear Similar results are obtained with a foreground anchor point (a-b); while a linear function over-stretched the texture with a background anchor point (c-d).*

ilar approach. It consists in offsetting a point $\mathbf{q}$ in the direction $(\bar{\mathbf{n}} - \bar{\mathbf{n}}_0)$ where $\bar{\mathbf{n}}$ and $\bar{\mathbf{n}}_0$ are the normals of $\mathbf{q}$ and $\mathbf{q}_0$ projected onto the picture plane. The offset factor $f_{\mathbf{n}}$ now depends on relative normal variations, and the reason we use their projection is to get a 2D offset direction in the image plane. Considering only $\mathbf{q}_0$, the offset vector is given by $f_{\mathbf{n}}(||\bar{\mathbf{n}} - \bar{\mathbf{n}}_0||)(\bar{\mathbf{n}} - \bar{\mathbf{n}}_0)$, with

$$f_{\mathbf{n}}(u) = \frac{\alpha u}{e^{-\beta} + u(1 - e^{-\beta})}. \tag{14}$$

As shown in Figure 4(b), $f_{\mathbf{n}}$ is a function of relative normal variations controlled via two parameters: $\alpha \in \mathbb{R}$ controls the signed magnitude of the deformation (positive and negative values mimic reflection and refraction patterns respectively), while $\beta \in \mathbb{R}^+$ controls its rate. The function behaves linearly when $\beta = 0$ and non-linearly when $\beta > 0$, hence producing more distortions around forshortened areas, which is useful for some refractive patterns (see Figure 6(e-f)).

As before, generalizing this idea to multiple anchor points, we obtain the following formula for second-order deformations:

$$\mathcal{D}_{\mathbf{n}}(\mathbf{q}) = \mathbf{q} + \sum_i w_i(\mathbf{q}) f_{\mathbf{n}}(||\bar{\mathbf{n}} - \bar{\mathbf{n}}_i||)(\bar{\mathbf{n}} - \bar{\mathbf{n}}_i). \tag{15}$$

When applied to the whole picture, $\mathcal{D}_{\mathbf{n}}$ distorts the image I in ways that closely mimic second-order surface variations, as seen in the bottom rows of Figures 2 and 3. Although our approach is heuristic, in practice the resulting distortions approximate surface flows closely, while maintaining intuitive artistic control.

Figure 6 demonstrates the effects of deformation parameters, here applied to a noise input image with a single anchor point (see also the supplemental video). When used with multiple anchor points, both operators may be seen as weighted sums of offset vectors, one for each anchor point. Provided that we use an interpolating weighting scheme ($\forall i \neq j, w_i(\mathbf{q}_i) = 1$ and $w_i(\mathbf{q}_j) = 0$), deformations will have no effect at anchor points. Indeed, $f_{\mathbf{p}|\mathbf{n}}(0) = 0$ and hence $\mathcal{D}_{\mathbf{p}|\mathbf{n}}(\mathbf{q}_i) = \mathbf{q}_i$. The choice of weighting scheme depends on selected tools, as described in the next Section.

5.2 Drawing tools

Our shading design system also relies on variations found in the input image I to convey surface material (see Equation 11). Operations affecting I, like blurring or color changes, are thus intended to approximate variations due to $V_{\mathbf{p}}$ and $V_{\mathbf{n}}$. We now introduce shading tools for the control of final image appearance.

Input noise   **First-order operator**

(a) $\alpha = \beta = \gamma = 0$ (b) $\gamma = 1$ (c) $\gamma = 5$

Second-order operator

(d) $\alpha = 1, \beta = 0$ (e) $\alpha = -0.8, \beta = 0$ (f) $\alpha = -0.8, \beta = 1.5$
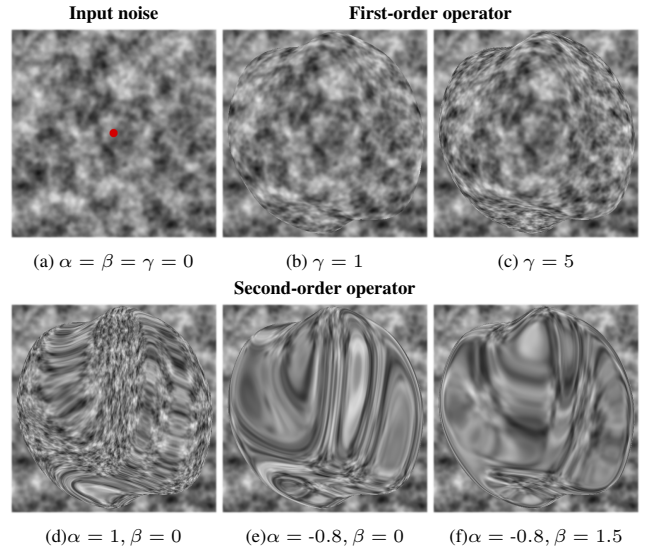
Figure 6: *Effect of deformation operators. In (a), a noise image is anchored to a single point in the picture. In (b-c), it is deformed using* $\mathcal{D}_{\mathbf{p}}$: *progressively increasing* γ *makes the noise appear as if it were textured on the object. In (d-f) the image is deformed using* $\mathcal{D}_{\mathbf{n}}$: *patterns resembling reflections appear with positive values of* α, *while negative values imitate refraction patterns; distortions around foreshortened areas are controlled via* β.

The brush tool allows users to paint shading in exactly the same way they would paint brush strokes in raster graphics software. It consists in repeatedly applying a brush footprint (I) with a single anchor point in its center; the only difference is that the footprint is deformed before being applied. When used with $\mathcal{D}_{\mathbf{p}}$, it permits the creation of decal textures as shown at the top of Figure 7 (we provide simple scaling and rotation controls of the footprint). When applied with $\mathcal{D}_{\mathbf{n}}$, it becomes easy to create plausible highlights. Their glossiness can be controlled by varying the blur applied to their footprints. As shown in Figure 1(c) and at the bottom of Figure 7, the tool is effective when multiple footprints are accumulated, hence producing non-physical, yet convincing highlights.

The gradient tool directly takes its inspiration from the classic linear color gradient, except that the gradient image (I) is deformed prior to its application. It makes use of two anchor points $\mathbf{q}_0$ and $\mathbf{q}_1$, one for each extremity of the gradient, and naturally requires two weighting functions w_0 and $w_1 = 1 - w_0$. The weight w_0 for a point $\mathbf{q}$ is obtained by projecting it perpendicularly onto the $[\mathbf{q}_0, \mathbf{q}_1]$ segment, and performing a linear interpolation (clamped at extremities). When applied with $\mathcal{D}_{\mathbf{n}}$, different material properties are mimicked depending on the content of I, as illustrated in Figure 8 where we show diffuse, cartoon or metallic appearances.

The image tool allows users to distort a full image (I) to mimic realistic textures or reflections, place highlights or cast shadows. Examples are shown in Figure 1(b) and Figure 3(b) where we have used a single anchor point. The manipulation of multiple anchor points is particularly useful for controlling the location of particular points of the input image, as shown in Figure 9. We have chosen Shepard interpolants [1968] for the weight functions, but any other interpolation scheme could have been used instead.

6 Results and comparisons

As shown in Figure 1(d), the final image is obtained by blending the results coming from multiple shading tools. Our implementation makes use of a nodal network that permits any kind of combination, in a way similar to systems found in Nuke or MentalMill for
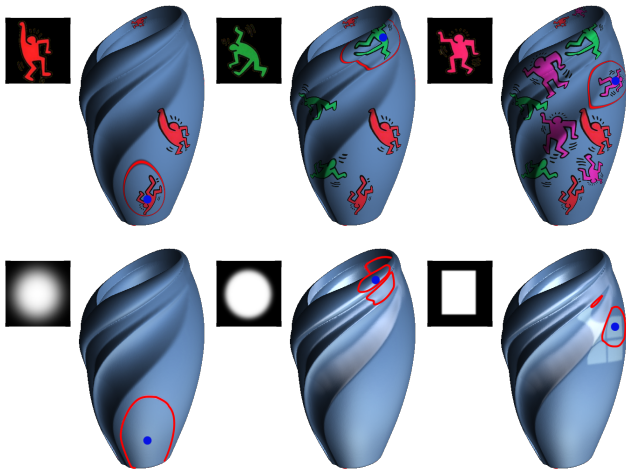
Figure 7: *Brush tool. Deformed textures (top row) or shading patterns (bottom row) are applied at arbitrary sizes (red contours) and locations (blue dots). Textures are oriented by the user to position decals. Shading patterns are blurred to convey different materials.*
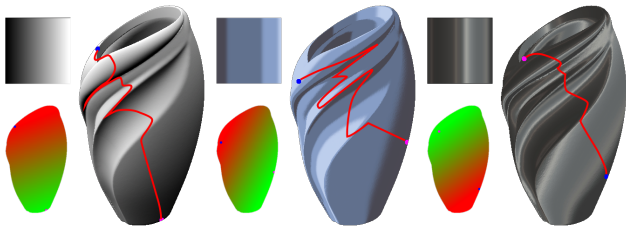

Figure 8: *Gradient tool. Smooth shading patterns are created by deforming a linear gradient (red curve). Two anchor points (blue and red dots) control its extremities. The gradient image controls material appearance: from diffuse, to cartoon, to metallic.*
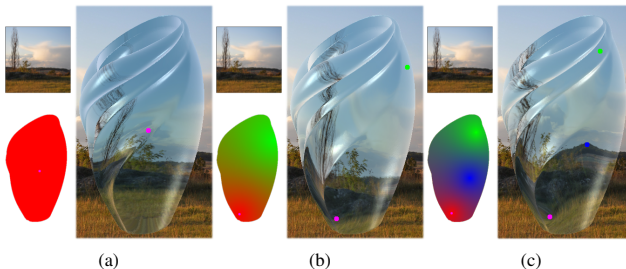

(a)   (b)   (c)

Figure 9: *Image tool. A refraction pattern is manipulated using anchor points: (a) the image is positioned (blue dot); (b) the tree refraction is moved downward (green dot); (c) the horizon is aligned with the background. Color insets visualize weight functions.*

instance. Each node is implemented as a GPU shader, which permits to observe the effect of intermediate nodes in real-time (>60 fps), as shown in the live demonstration video. In the following, we present a variety of shading combinations produced with our system, and compare them to related work. Step-by-step explanations for most of them are provided in supplemental materials.

Image-based shading   Figure 10 illustrates second-order deformations using normal images coming from different sources. When applied to captured normals, our image tool produces a compelling shading effect with a single drag-and-drop. We have also experimented with a simple enhancement functionality inspired by the Light Warping technique [Vergne et al. 2011]; we amplify the mag-

nitude of deformations proportionally to their view-centered curvature measure, which yields equivalent results. Our brush tool is the best choice when working with drawn normals: reflections are accurately positioned to create cartoon results. Our method is naturally adapted to edit material appearance in existing images as in [Khan et al. 2006], using estimated normals. The key difference is that our approach provides a more direct and versatile control over apparent shape, materials and illumination effects, for instance to place and move highlights at specific locations on the object.

Texture mapping   Figure 11 compares our first-order deformation operator with the texture draping technique of Winnemoeller *et al.* [2009]. The first important difference is that our approach works directly from a depth image as advocated by our differential analysis, whereas theirs integrates the normal image to get a height field. Using true depths produces more convincing depictions as best seen around occluding boundaries. The second difference is that our operator is controlled by anchor points located at arbitrary positions in the picture, while theirs relies on parallax mapping, and is thus controlled indirectly by the specification of a view vector.

Complex effects   are obtained by combining multiple deformations. For instance in figure 12(a), we add shadows to Figure 1(d) using the brush tool with first-order deformations. In Figure 12(b), we apply a normal texture to create an additional surface normal layer, which is then used as a base for further operations. The translucent object in Figure 12(c) is obtained by combining transparent and specular deformations, blended using Schlick's approximation [Schlick 1994] of the Fresnel term. We also perturbed the input normal map with a noise texture for deforming the structure of refractions, as suggested in [Motoyoshi 2010].

Material design   A variety of material effects is achievable using our image-based shading design approach. This is illustrated in Figure 13, where we have used simple shapes to let the reader better appreciate shading details. Stylized material appearances such as the ones found in logos are trivially obtained with either of our tools, saving a lot of time and training to artists. More photorealistic materials are produced by blurring and deforming real-world photographs. For complex fabric materials, we found it sufficient to layer brush strokes of varying intensity at different locations.

Ease of use   Our shading design system is a prototype that is only intended to illustrate our perceptually-motivated approach. Even though it is beyond the scope of the present paper, a proper user study will be useful in the future. Nevertheless, our first experiments are encouraging. Let us first point out that many of the images were created using only a single anchor point, for which the extent of artistic direction of appearance is highly limited. Figure 14(a) provides an extreme example, where we compare our approach with the photorealistic rendering of a mirror object obtained with PBRT [Pharr and Humphreys 2010]. In our result, the original angular environment map is deformed using a single anchor point located at the center of the picture. Slightly more complex examples are provided in Figure 14(b-c): the refractive object is obtained by deforming the same environment map with an opposite magnitude and positioned so that it aligns with the background; whereas the plastic object uses the same configuration as the mirror case, but blurs the environment prior to deformation.

Our second observation is that deformation operators vary appearance *within* the constraints of a given shape; hence there is a broad "sweet spot" since we exploit key image features that drive shape perception. However, as illustrated in Figure 15, even if deformations remain plausible, unnatural images are obtained when the expected congruence between shading effects is not respected [Kim et al. 2011] (e.g., placing highlights in shadowed regions).
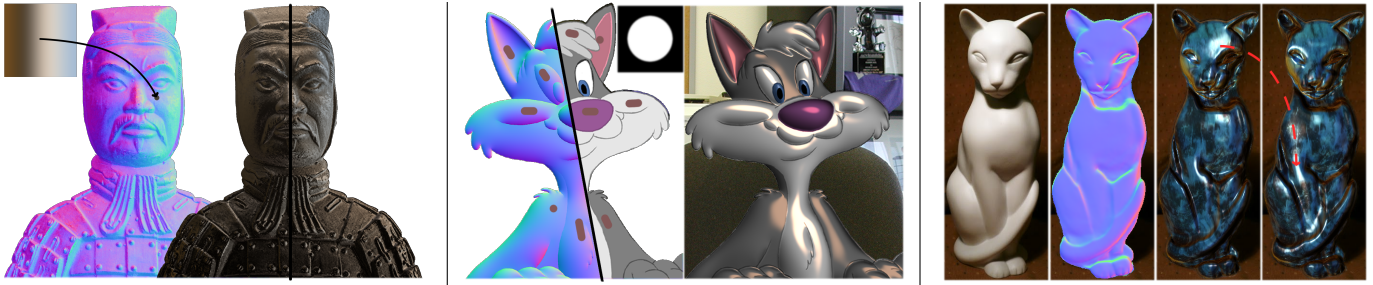
Figure 10: ***Image-based shading.*** *Left: Image tool applied to a captured normal map (RGBN image [Toler-Franklin et al. 2007]), with details enhanced. Middle: Brush tool applied to a drawn normal map. Right: Image-based material editing with our approach: highlights are directly manipulated.*

7 Discussion

Contributions The primary benefits of our approach lie in the speed and directness with which images can be built and edited. Users are free to drag shading, highlights and texture around to edit appearance directly, while preserving apparent shape. Appearance can be honed gradually by adding and tweaking layers, or changed radically in a couple of seconds by swapping between different source textures. Because the results are built out of other images, it is easy to acquire or generate novel source textures. The other advantage is that because the method explicitly relies on the flows that determine perceived shape and material properties, the appearance of these properties can be emphasized, exaggerated or modified directly in the image.

Limitations A minor problem appears when using too many anchor points: shape perception may then be affected since the resulting deformations will differ from plausible variations; on the other side, it will come closer to user desires. Another issue with manually controlled anchor points is animation: when full geometry is given (rather than just normal maps), and objects are rotated and moved, anchor points should move accordingly to yield convincing shading and texture motion. Materials are also controlled with an artistic approach, using simple image processing operators. If plausible renderings can be obtained with this technique, taking the second-order flow into account when applying these operators could lead to more accurate results when it comes to photorealistic appearances. For instance, the current deformation cannot imitate anisotropic materials. With extreme deformations, our system will also tend to produce aliasing. However, we know precisely the extent of deformations at each pixel, which is ideal for designing anti-aliasing strategies.

Future work Although we have only applied deformations to raster images, there is no a priori reason why it would not work with vector graphics. Connecting it with a system that directly outputs vector normal and depth data will likely result in a fine solution for drawing applications (e.g., [Shao et al. 2012]). Beyond practical matters, we hope the differential analysis we have conducted in the image plane will foster interest in both the vision and graphics communities. Although our deformation operators closely approximate surface flows, they still deserve further investigations not only to understand their connections with real images, but also to provide new effects, such as anisotropy. In particular, a detailed study of $V_{\mathbf{P}}$ and $V_{\mathbf{n}}$ would provide more insights into how light and material properties interact with shape. They connect variations in a 3D scene to observable pictorial variations, and hence have potential applications not only in rendering and compositing, but also for the study of the human visual system.

Figure 11: ***Texture mapping:*** *compared to [Winnemöller et al. 2009] (a), our approach (b) better conveys discontinuities at occluding contours and produces more plausible deformations.*
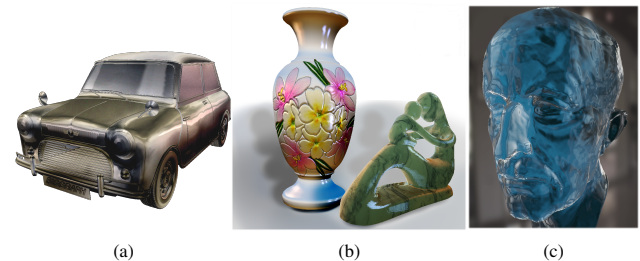

Figure 12: ***Complex results*** *obtained by combining multiple deformations: (a) adding shadows; (b) layering normals; and (c) imitating translucency (see supplemental materials).*

Acknowledgments

References

ANDERSON, B. L., AND KIM, J. 2009. Image statistics do not explain the perception of gloss and lightness. *J. Vis. 9*, 11, 1–17.

ANJYO, K., WEMLER, S., AND BAXTER, W. 2006. Tweakable light and shade for cartoon animation. In *Proc. symposium on Non-photorealistic animation and rendering*, ACM, 133–139.

BIGÜN, J., AND GRANLUND, G. 1987. Optimal orientation detection of linear symmetry. In *Proc. IEEE First International Conference on Computer Vision*, 433–438.

BRETON, P., AND ZUCKER, S. W. 1996. Shadows and shading flow fields. In *IEEE Conf. Computer Vision and Pattern Recognition*, 782–789.
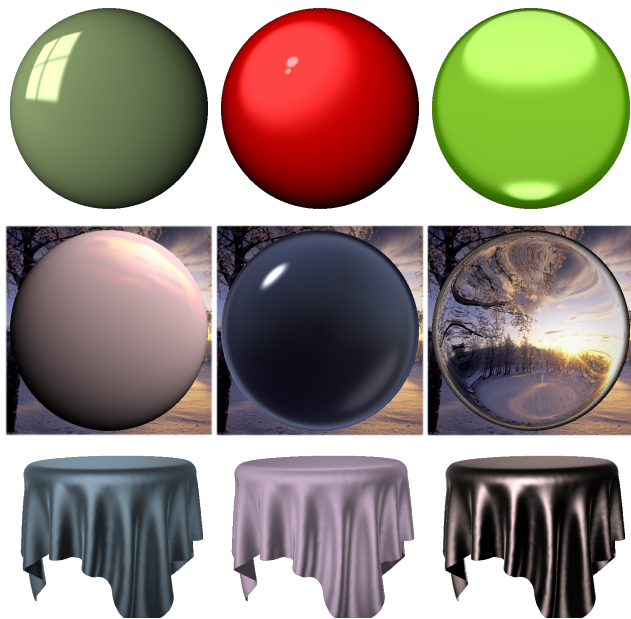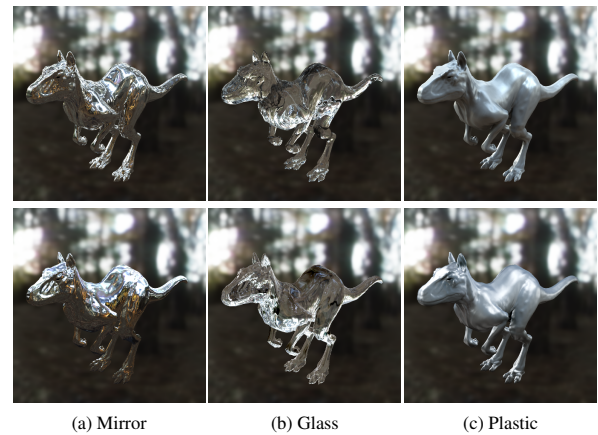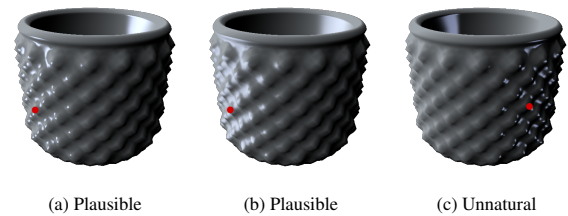
COLBERT, M., PATTANAIK, S., AND KRIVANEK, J. 2006. BRDF-

Figure 13: *Material design. Top: logo styles are created in a matter of seconds with the brush tool. Middle: photorealistic effects obtained by deforming the background image. Bottom: fabrics appearance produced with carefully positioned highlights.*

(a) Mirror   (b) Glass   (c) Plastic

Figure 14: *Comparison with PBRT. Photo-realistic renderings (top) are imitated using the image tool with a single anchor point (bottom) and a few brush strokes for three different materials.*

(a) Plausible   (b) Plausible   (c) Unnatural

Figure 15: *Failure case. Varying distortion parameters (here magnitude) often produces plausible shading (a-b), although unconventional anchor point placement leads to unnatural images (c).*

Shop: Creating Physically Correct Bidirectional Reflectance Distribution Functions. *IEEE Comput. Graph. Appl. 26*, 30–36.

DELPOZO, A., AND SAVARESE, S. 2007. Detecting specular surfaces on natural images. In *IEEE Conf. Computer Vision and Pattern Recognition*, 18–23.

FANG, H., AND HART, J. C. 2004. Textureshop: texture synthesis as a photograph editing tool. *ACM Trans. Graph. 23*, 354–359.

FLEMING, R. W., TORRALBA, A., AND ADELSON, E. H. 2004. Specular reflections and the perception of shape. *J. Vis. 4*, 9, 798–820.

FLEMING, R. W., TORRALBA, A., AND ADELSON, E. H. 2009. Shape from sheen. Tech. rep., MIT-CSAIL-TR-2009-051.

FLEMING, R. W., HOLTMANN-RICE, D., AND BÜLTHOFF, H. H. 2011. Estimation of 3D shape from image orientations. *Proc. National Academy of Sciences 108*, 51, 20438–20443.

FLEMING, R., JÄKEL, F., AND MALONEY, L. 2011. Visual perception of thick transparent materials. *Psychological Science 22*, 6, 812–820.

GIBSON, J. J. 1950. *The Perception of the Visual World*. Houghton Mifflin.

HUGGINS, P. S., AND ZUCKER, S. W. 2001. Folds and cuts: How shading flows into edges. In *Proc. IEEE Int. Conf. Computer Vision*, 153–158.

JESCHKE, S., CLINE, D., AND WONKA, P. 2011. Estimating color and texture parameters for vector graphics. *Computer Graphics Forum 30*, 2, 523–532.

JOHNSTON, S. F. 2002. Lumo: illumination for cel animation. In *Proc. symposium on Non-photorealistic animation and rendering*, ACM, 45–ff.

KHAN, E. A., REINHARD, E., FLEMING, R. W., AND BÜLTHOFF, H. H. 2006. Image-based material editing. *ACM Trans. Graph. 25*, 654–663.

KIM, J., AND ANDERSON, B. L. 2010. Image statistics and the perception of surface gloss and lightness. *J. Vis. 10*, 9, 3.

KIM, J., MARLOW, P., AND ANDERSON, B. 2011. The perception of gloss depends on highlight congruence with surface shading. *J Vis 11*, 9.

KOENDERINK, J. J., AND VAN DOORN, A. J. 1980. Photometric invariants related to solid shape. *Optica Acta 27*, 7, 981–996.

LI, A., AND ZAIDI, Q. 2000. Perception of three-dimensional shape from texture is based on patterns of oriented energy. *Vision Research 40*, 2, 217–42.

MCCANN, J., AND POLLARD, N. S. 2008. Real-time gradient-domain painting. *ACM Trans. Graph. 27*, 3, 93:1–93:7.

MOTOYOSHI, I., NISHIDA, S., AND ADELSON, E. H. 2005. Luminance re-mapping for the control of apparent material. In *Proc. Symposium on Applied Perception in Graphics and Visualization*, ACM, 165–165.

MOTOYOSHI, I. 2010. Highlight-shading relationship as a cue for the perception of translucent and transparent materials. *J Vis 10*, 9, 6.

NICODEMUS, F. E., RICHMOND, J. C., HSIA, J. J., GINSBERG, I. W., AND LIMPERIS, T. 1977. Geometric Considerations and Nomenclature for Reflectance. *National Bureau of Standards*.

OH, B. M., CHEN, M., DORSEY, J., AND DURAND, F. 2001. Image-based modeling and photo editing. In *ACM Trans. Graph.*, 433–442.

OKABE, M., MATSUSHITA, Y., SHEN, L., AND IGARASHI, T. 2007. Illumination Brush: Interactive Design of All-Frequency Lighting. In *Pacific Conference on Computer Graphics and Applications*, 171–180.

ORZAN, A., BOUSSEAU, A., WINNEMÖLLER, H., BARLA, P., THOLLOT, J., AND SALESIN, D. 2008. Diffusion curves: a vector representation for smooth-shaded images. *ACM Trans. Graph. 27*, 3, 92:1–92:8.

PACANOWSKI, R., GRANIER, X., SCHLICK, C., AND POULIN, P. 2008. Sketch and Paint-based Interface for Highlight Modeling. In *Eurographics workshop on Sketch-Based Interfaces and Modeling*, 17–23.

PELLACINI, F., BATTAGLIA, F., MORLEY, R. K., AND FINKELSTEIN, A. 2007. Lighting with paint. *ACM Trans. Graph. 26*, 2.

PELLACINI, F. 2010. envylight: an interface for editing natural illumination. *ACM Trans. Graph. 29*, 34:1–34:8.

PHARR, M., AND HUMPHREYS, G. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation*, 2nd ed. Morgan Kaufmann Publishers Inc.

QUIXEL, 2011. ndo2. http://quixel.se/.

RAMAMOORTHI, R., MAHAJAN, D., AND BELHUMEUR, P. 2007. A first-order analysis of lighting, shading, and shadows. *ACM Trans. Graph. 26*, 1.

RAMANARAYANAN, G., FERWERDA, J., WALTER, B., AND BALA, K. 2007. Visual equivalence: towards a new standard for image fidelity. *ACM Trans. Graph. 26*, 3.

RITSCHEL, T., OKABE, M., THORMÄHLEN, T., AND SEIDEL, H.-P. 2009. Interactive reflection editing. *ACM Trans. Graph. 28*, 5, 129:1–129:7.

RITSCHEL, T., THORMÄHLEN, T., DACHSBACHER, C., KAUTZ, J., AND SEIDEL, H.-P. 2010. Interactive on-surface signal deformation. *ACM Trans. Graph. 29*, 4.

SCHLICK, C. 1994. *Graphics Gems V*. Morgan Kauffman, ch. A Fast Alternative to Phong's Specular Model, 385–384.

SHAO, C., BOUSSEAU, A., SHEFFER, A., AND SINGH, K. 2012. Crossshade: Shading concept sketching using cross-section curves. *ACM Transactions on Graphics (Proc. SIGGRAPH) 31*, 4 (Aug.).

SHEPARD, D. 1968. A two-dimensional interpolation function for irregularly-spaced data. In *Proc. 1968 23rd ACM national conference*, 517–524.

SLOAN, P.-P. J., MARTIN, W., GOOCH, A., AND GOOCH, B. 2001. The lit sphere: a model for capturing npr shading from art. In *Graphics interface*, 143–150.

SUN, J., LIANG, L., WEN, F., AND SHUM, H.-Y. 2007. Image vectorization using optimized gradient meshes. *ACM Trans. Graph. 26*, 3, 11.

TAPPEN, M. F. 2011. Recovering shape from a single image of a mirrored surface from curvature constraints. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2545–2552.

TODD, J. T., AND AKERSTROM, R. A. 1987. Perception of three-dimensional form from patterns of optical texture. *J. experimental psychology. Human perception and performance 13*, 2, 242–255.

TODD, J. T., AND MINGOLLA, E. 1983. Perception of Surface Curvature and Direction of Illuminant from Patterns of Shading. *J. Experimental Psychology: Human Perception and Performance 9*, 583–595.

TOLER-FRANKLIN, C., FINKELSTEIN, A., AND RUSINKIEWICZ, S. 2007. Illustration of complex real-world objects using images with normals. In *Proc. symposium on Non-photorealistic animation and rendering*, ACM, 111–119.

VASILYEV, Y., ZICKLER, T., GORTLER, S. J., AND BEN-

SHAHAR, O. 2011. Shape from specular flow: Is one flow enough? In *IEEE Conf. Computer Vision and Pattern Recognition*, 2561–2568.

VERGNE, R., PACANOWSKI, R., BARLA, P., GRANIER, X., AND SCHLICK, C. 2011. Improving Shape Depiction under Arbitrary Rendering. *IEEE Trans. Visualization and Computer Graphics 17*, 8, 1071 – 1081.

WINNEMÖLLER, H., ORZAN, A., BOISSIEUX, L., AND THOLLOT, J. 2009. Texture Design and Draping in 2D Images. *Computer Graphics Forum 28*, 4, 1091–1099.

WU, T.-P., TANG, C.-K., BROWN, M. S., AND SHUM, H.-Y. 2007. ShapePalettes: interactive normal transfer via sketching. *ACM Trans. Graph. 26*, 3, 44.

YEUNG, S. K., TANG, C.-K., BROWN, M. S., AND KANG, S. B. 2011. Matting and compositing of transparent and refractive objects. *ACM Trans. Graph. 30*, 1, 2:1–2:13.

Appendix

Change of variable The BRDF as expressed in Equation 1 is dependent on the local normal $\mathbf{n}$. Let's write this dependence more explicitly using the rotation $\mathcal{R}_\mathbf{n}$: $\rho_\mathbf{n}(\mathbf{p}, \mathbf{v}, \omega) = \rho_\mathbf{z}(\mathbf{p}, \mathcal{R}_\mathbf{n}\mathbf{v}, \mathcal{R}_\mathbf{n}\omega)$.

Applying the change of variable $\ell \leftarrow \mathcal{R}_\mathbf{n}\omega$ to Eq. 1, we obtain:

$$L_r(x,y) = \int_{\mathcal{R}_\mathbf{n}\Omega_\mathbf{n}^+} \rho_\mathbf{z}(\mathbf{p}, \mathcal{R}_\mathbf{n}\mathbf{v}, \ell)\, L_i(\mathbf{p}, \mathcal{R}_\mathbf{n}^{-1}\ell)\, (\mathbf{n} \cdot \mathcal{R}_\mathbf{n}^{-1}\ell)\, |\mathcal{R}_\mathbf{n}| d\ell.$$

Noting that $\Omega_\mathbf{z}^+ = \mathcal{R}_\mathbf{n}\Omega_\mathbf{n}^+$, $|\mathcal{R}_\mathbf{n}| = 1$ since $\mathcal{R}_\mathbf{n}$ is a rotation, and $(\mathbf{n} \cdot \mathcal{R}_\mathbf{n}^{-1}\omega) = (\mathcal{R}_\mathbf{n}\mathbf{n} \cdot \ell) = (\mathbf{z} \cdot \ell)$, we obtain Equation 2.

Derivative of a rotation Let's write the rotation that brings the normal $\mathbf{n}$ into alignment with $\mathbf{z}$ more explicitly as $\mathcal{R}_{\mathbf{n}\to\mathbf{z}}$. With the same notation, the first-order approximation of the rotation operator for a neighbor unit normal $\mathbf{n}' = \mathbf{n} + \partial_\mathbf{x}\mathbf{n}$ is given by $\mathcal{R}_{\mathbf{n}'\to\mathbf{z}} = \mathcal{R}_{\mathbf{n}\to\mathbf{z}}\mathcal{R}_{\mathbf{n}'\to\mathbf{n}} \approx \mathcal{R}_{\mathbf{n}\to\mathbf{z}} + \partial_\mathbf{x}\mathcal{R}_{\mathbf{n}\to\mathbf{z}}$, which yields for an arbitrary vector ν: $\partial_\mathbf{x}\mathcal{R}_{\mathbf{n}\to\mathbf{z}}\nu = \mathcal{R}_{\mathbf{n}\to\mathbf{z}}(\mathcal{R}_{\mathbf{n}'\to\mathbf{n}} - I)\nu$.

Let's represent $\mathcal{R}_{\mathbf{n}'\to\mathbf{n}}$ as a rotation of axis $\bar{\mathbf{u}}$ and angle θ. To a first-order approximation, we have $\cos\theta = 1$, $\sin\theta = ||\mathbf{u}||$, and $\bar{\mathbf{u}} = \mathbf{u}/||\mathbf{u}||$, with $\mathbf{u} = \mathbf{n}' \times \mathbf{n} = \partial_\mathbf{x}\mathbf{n} \times \mathbf{n}$. Using Rodrigues' formula, the rotation applied to an arbitrary vector $\mathbf{v}$ is given by:

$$\mathcal{R}_{\mathbf{n}\to\mathbf{n}'}\mathbf{v} = \mathbf{v}\cos\theta + (\bar{\mathbf{u}}\times\mathbf{v})\sin\theta + \bar{\mathbf{u}}(\bar{\mathbf{u}}\cdot\mathbf{v})(1-\cos\theta) = \mathbf{v} + \mathbf{u}\times\mathbf{v}.$$

Setting $\mathbf{v} = \mathcal{R}_{\mathbf{n}\to\mathbf{z}}\nu$ and plugging this back inside the derivative formula, we obtain $\partial_\mathbf{x}\mathcal{R}_{\mathbf{n}\to\mathbf{z}}\nu = \mathcal{R}_{\mathbf{n}\to\mathbf{z}}\partial_\mathbf{x}\mathbf{n} \times \mathbf{n} \times \nu$. The derivative of $\mathcal{R}_{\mathbf{n}\to\mathbf{z}}^{-1}$ is obtained by taking a rotation axis $\bar{\mathbf{u}}$ that points in the opposite direction, yielding $\partial_\mathbf{x}\mathcal{R}_{\mathbf{n}\to\mathbf{z}}^{-1}\nu = -\partial_\mathbf{x}\mathbf{n} \times \mathbf{n} \times \mathcal{R}_{\mathbf{n}\to\mathbf{z}}^{-1}\nu$.

Second-order structure tensor The structure tensor is a 2 by 2 matrix where each coefficient is obtained as a dot product involving partial derivatives. For the second-order shape term, it also involves cross products with the surface normal $\mathbf{n}$. Lagrange's identity permits expressing each coefficient in terms of dot products only, as exemplified below with off-diagonal coefficients:

$$(\partial_\mathbf{x}\mathbf{n} \times \mathbf{n}) \cdot (\partial_\mathbf{y}\mathbf{n} \times \mathbf{n}) = (\partial_\mathbf{x}\mathbf{n} \cdot \partial_\mathbf{y}\mathbf{n})(\mathbf{n} \cdot \mathbf{n}) - (\partial_\mathbf{x}\mathbf{n} \cdot \mathbf{n})(\partial_\mathbf{y}\mathbf{n} \cdot \mathbf{n})$$

This expression is further simplified by noting that the normal is a unit vector ($\mathbf{n} \cdot \mathbf{n} = 1$), and thus is always orthogonal to its derivatives ($\partial_\mathbf{t}\mathbf{n} \cdot \mathbf{n} = 0$). This yields $(\partial_\mathbf{x}\mathbf{n} \times \mathbf{n}) \cdot (\partial_\mathbf{y}\mathbf{n} \times \mathbf{n}) = \partial_\mathbf{x}\mathbf{n} \cdot \partial_\mathbf{y}\mathbf{n}$. The derivation of diagonal terms follows the same reasoning.