Using Isophotes and Shadows to Interactively Model Normal and Height Fields

Qiuying Xu[a], Songrun Liu[b], Yotam Gingold[b], Karan Singh[a]

[a] *University of Toronto*
[b] *George Mason University*

Abstract

We introduce an interactive modeling tool for designing (a) a smooth 3D normal field from the isophotes of a discretely shaded 2D image and (b) lifting the normal field into a smooth height field given a cast shadow. Block or cartoon shading is a visual style in which artists depict a smoothly shaded 3D object using a small number of discrete brightness values, manifested as regions or bands of constant color. In our approach, artists trace isophotes, or curves of constant brightness, along the boundaries between constant color bands. Our algorithm first estimates light directions and computes 3D normals along the object silhouette and at intersections between isophotes from different light sources. We then propagate these 3D normals smoothly along isophotes, and subsequently throughout the interior of the shape. We describe our user interface for editing isophotes and correcting unintended normals produced by our algorithm. We also describe a technique for lifting the generated normal field into a height field given the boundary of the shadow cast by the object. We validate our approach with a perceptual experiment and comparisons to ground truth data. Finally, we present a set of 3D renderings created using our interface.

Keywords: sketching, non-photorealistic rendering, isophotes
2010 MSC: 65D18

1. Introduction

Cartoon or toon shading is a visual style for computer-generated artwork that portrays the brightness of a shaded surface by binning or discretizing the brightness values into a small number of bins (Figure 1). The smooth brightness value at a point on a Lambertian surface with normal n and light direction l can be computed as $k = \max(n \cdot l, 0)$. In toon shading, k is rounded to one of b bins: $\lfloor bk + \frac{1}{2} \rfloor$. Toon shading is so-called because it resembles hand-drawn cartoons. 2D artists employ toon-style shading, or other forms of discrete tone shading, such as hatching at discrete tone levels [1] or blocking large areas of light and shade [2], as an effective way to quickly convey 3D relief information (Figure 1). While discretely shaded line drawings are compelling in their own right, artists often refine them into smooth-shaded realistic imagery called *presentation renderings*, when communicating their designs [3, 4]. Creating these smooth 3D renderings, often using a variety of colors, materials and lighting is tedious, labor intensive and error prone. Inspired by CrossShade [5], which addresses the presentation rendering problem for cross-sectional line drawings, we present an interactive modeling tool to assist artists with *inverse toon shading*.

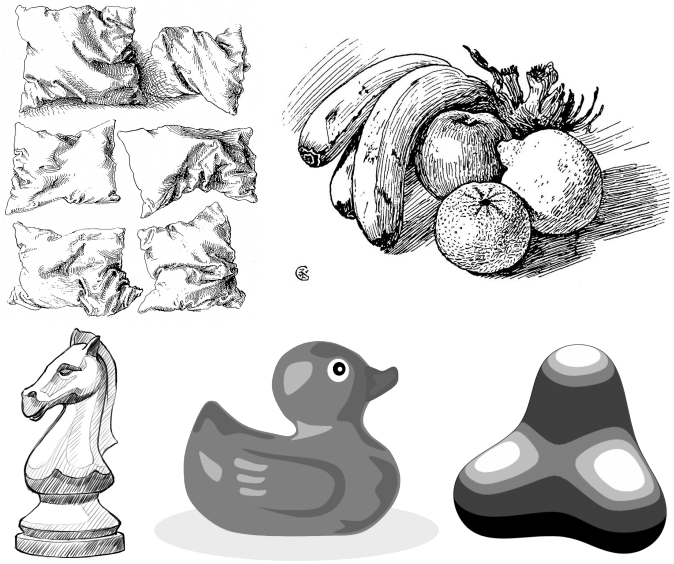
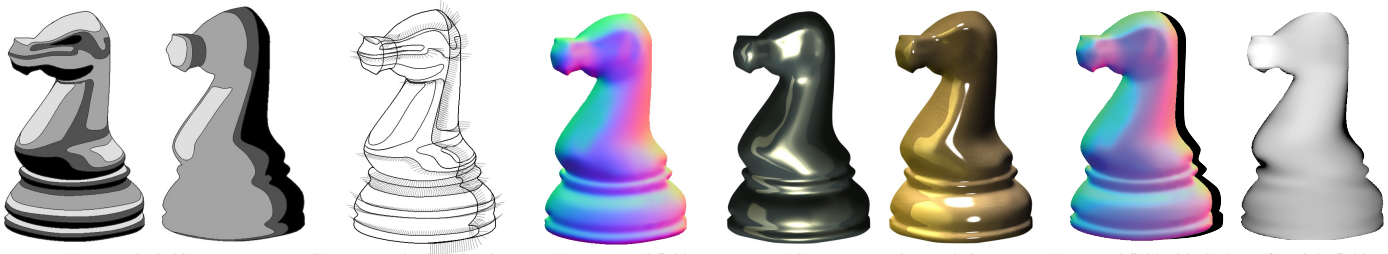


Figure 1: Motivational imagery includes drawings with discrete hatching, cartoon shaded and posterized art.

Inverse toon shading is the transformation of a discrete- or toon-shaded image into a smooth 3D normal or height field that matches the discrete shading and is consistent with viewer expectations (Figure 2). As this is a severely underconstrained problem, our aim is to create plausible normal fields with minimal but necessary user input. Normal fields allow artists to model shapes that do not correspond to physical 3D objects. Modeling in 3D is time intensive, and the drawn object may not yet be realizable in 3D. In the early stage of a design process, whole-object 3D consistency can be a distraction. Extending our recent work [6], we aim to lift a normal field into a height field given a cast shadow. We thus take as input the 2D silhouette of a smooth object along with one or more toon-shadings of the interior, each illuminated by a different directional light source (Figure 2a), and an optional cast shadow. Each toon shading is interpreted as a set of *isophote* curves (the iso-luminant boundary curves between toon-shaded bins) sharing the same light



(a) Toon shaded input (b) 3D normals along isophotes (c) 3D normal field (d) 3D presentation renderings (e) Normal field with shadow (f) Height field

Figure 2: Quick line drawings with hatching or vector art frequently use a few discrete levels of tone or shading to convey smooth 3D shape (a). Artists use our tool to trace isophotes that separate discrete levels; our algorithm estimates smooth surface normals across the entire drawn objects. The normals along isophotes are interpolated to exhibit minimal variation and match viewer expectation (b). The resulting 3D normal fields (c) allow users to create 3D presentation renderings of objects using a variety of shading styles and lighting (d). With the addition of a cast shadow (e), we lift the normal field to obtain a 2.5D height field (f).

direction (Figure 2b). Artists pick light directions and draw isophote shapes, where the 2D isophote is indicative of its 3D shape and the 3D surface normals change smoothly along the isophotes with minimal variation [4]. The boundary of the cast shadow is used to lift the normal field into a height field.

Our 2D drawing interface (Section 4) provides artists with functionality for creating and editing silhouettes, isophotes, shading values, and light directions. Light directions that best fit the shading values and 3D normals at silhouette/isophote intersection points can be automatically computed. 3D rendered results based on the surface normals computed by our inverse toon shading algorithm can be evaluated by the artist and undesirable normals interactively corrected.

Our algorithm (Section 5) begins by computing well-defined 3D normals along the silhouette and at intersections between isophotes from different lights. (Isophotes from the same light direction do not typically intersect.) We then smoothly propagate these 3D normals along the 2D isophotes, and further refine the normals along iteratively estimated 3D isophotes. Finally, we diffuse the normals throughout the interior of the shape. If a cast shadow and accompanying light direction are provided, we integrate the normal field with the knowledge that the shadow boundary is the object's silhouette from the light direction.

We perform two perceptual studies (Section 7) to measure the following: (I) consistency and accuracy of viewers' perceived surface normals along isophotes taken from ground truth 3D data, and (II) artists' ability to draw isophotes. Our algorithm produces results comparable to viewers of the above study, validating our approach. We evaluate our interface with user testers and by creating a set of plausible 3D renderings created using our interface (Section 8). We conclude with a summary of our technique's limitations (Section 9).

Our chief contribution is formulating the novel problem of inverse toon shading, to which we present a plausible, interactive modeling solution: specifically, we develop an interactive tool for drawing and shading silhouettes and isophotes, that guides artists towards geometrically valid input; an algorithm that computes a smooth 3D normal field to match the toon shaded input; an algorithm that lifts a normal field with a shadow into a height field; and perceptual studies that measures humans' ability to perceive smooth surface normals along isophotes and draw accurate isophotes.

2. Related Work

Presentation rendering of sketches. Concept or gesture sketches are quick line drawings with discrete shading cues, widely used to rapidly explore ideas, that are then refined and shaded for presentation to the client [4, 3]. While a finalized concept will indeed be turned into a 3D model for further processing, working directly with sketches facilitates rapid design iteration. Designers commonly use painting tools such as Adobe Photoshop to shade their sketches. Fast colorization and shading tools propagate scribbles [7] or diffuse color and shading using a few vector primitives [8, 9]. Despite these advances, painting convincing 3D-like shading requires extraordinary skill and time. Instead, like Lumo [10], CrossShade [5], we go a step further: computing a 3D normal field from a 2D silhouette and a few other curves and constraints. Lumo and its extensions [11, 12], based on silhouettes and internal contours, are well suited to amorphous organic shapes. CrossShade, in contrast, is based on shape cross-sections, ideal for structured man-made objects. Our work is complementary to these systems in spirit but based on isophotes and shading, which can be used to depict a mix of organic and man-made forms (Figures 2, 14, 17). Several interfaces for specifying normal fields [13, 14, 15] have also been proposed. In our tool, users specify the brightness of each isophote, which is a single value rather than a smoothly changing 3D direction. Like CrossShade, however, our method builds upon the traditional sketching workflow by using discrete tone shading commonly employed by artists.

Shape from Shading and Sketching 3D models. A well-studied and challenging Computer Vision problem concerns the recovery of a height field from a 2D image of an object (see [16] for a survey). Shape from shading approaches make this problem tractable by assuming that the image contains only luminance information from a single light source, and that the object is Lambertian (perfectly diffuse). The related problem, when images from multiple illumination directions are provided, is called Photometric Stereo [17]. While our problem input also comprises shading information from one or more light directions, shape from shading approaches are largely inapplicable as our input is very sparse (luminance along a few isophotes) and contains inevitable sketching inaccuracy. The goal of 3D rendering a sketch is also disparate from height field or 3D model reconstruction.

In the former, the 2D sketched curves are preserved and any geometric inaccuracies are absorbed into the 3D normal field, whereas in the latter the 3D output must typically deviate from the 2D sketch to be geometrically precise [18]. Our problem shares the general goal of inflating a flat 2D curve with sketch-, image-, projection-, or silhouette-based approaches to creating 3D models [13, 19, 20, 21, 22, 23, 24, 25]. 3D sketch modeling research has also noted the use of shading as a handle for editing a 3D shape by editing its smooth shading [26]; their approach requires a pre-existing 3D shape and is not designed to create one purely from shading.

Isophotes and Reflection Lines. Fair surface design often employs isophotes and reflection lines [27] to directly visualize the continuity of the surface gradient [28]. Deslandes and Bonner [29] describe an approach for exactly specifying reflection lines by modifying an existing 3D shape. Loos et al. [30] create a surface (height field) from desired illumination gradients defined everywhere on the surface (equivalent to specifying isophote spacing). Tosun et al. [31] present a real-time solution to a variant of this problem, in which an initial surface is modified. These illumination-gradient approaches allow control over the spacing and direction of isophotes, whereas our setting takes as input a sparse set of 2D projections of isophotes, without knowledge of the illumination gradient.

Toon shading and NPR. There is a large body of work in non-photorealistic rendering [32] that addresses the inverse of our problem: producing discrete or toon shaded imagery in a variety of artistic styles, even using isophote information [33], from a 3D scene or realistically shaded image.

3. Definitions and Model Assumptions

In toon shading or discrete tone shading, the surface illumination value is rounded to a discrete set of values. The boundary curves between bands of similar shading are isophotes. We pick the isophote value to be the average tone of its two adjacent bands. Assuming diffuse (Lambertian) illumination from a light direction l, every isophote is a curve along which the surface normal n satisfies the equation $n \cdot l = k$, where k is the constant tonal value. For a fixed l and k, the set of satisfying normal vectors n sweep out a cone around l, which we call the *normal-cone* (Figure 3). All 3D surface normals along the isophote lie on the normal-cone and can by parameterized by an angle value $t \in [0, 2\pi)$ that sweeps a circle around the cone.

We use x, y to denote the image plane with positive z pointing out of the plane. We also make several simplifying assumptions:

- The 3D shapes are smooth (at least tangent continuous).
- The light sources are directional and the shapes front-lit: $l_z \geq 0$.
- Shading is due to Lambertian reflection. In other words, there are no specular highlights. A specular hot-spot in 2D precisely defines a surface normal for a given view and light, and is easily incorporated into our approach. There are also no cast shadows present in the toon-shaded images; the shading only defines form shadows [2]. We ignore highlights and cast shadows to avoid the complexity of separating the Lambertian component from the overall surface illumination.
- There are no occluding contours. All surface normals in the interior of the silhouette have strictly positive z components. Technically, surface normals on the occluding side of an internal contour are well-defined and normals on the occluded side can be handled by free-boundary diffusion. Internal contours, however, often cast form shadows locally on the shape, violating our previous assumption.
- Sketched 2D isophote shapes are descriptive of their 3D geometry: they are not drawn from a degenerate view; in 3D, each isophote is as close to a planar curve as possible; and surface normals vary smoothly with fairness along the isophotes.

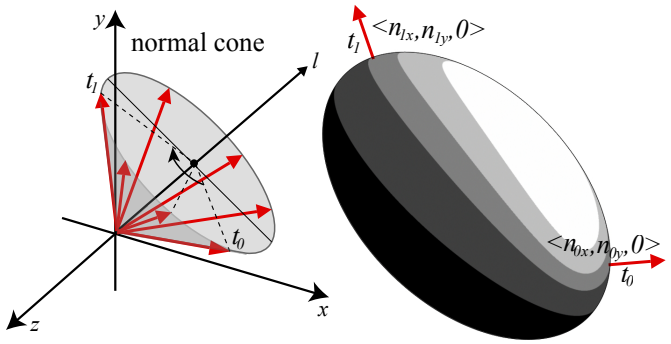


Figure 3: A normal cone is formed by iso-luminant surface normals at a constant angle to the light.

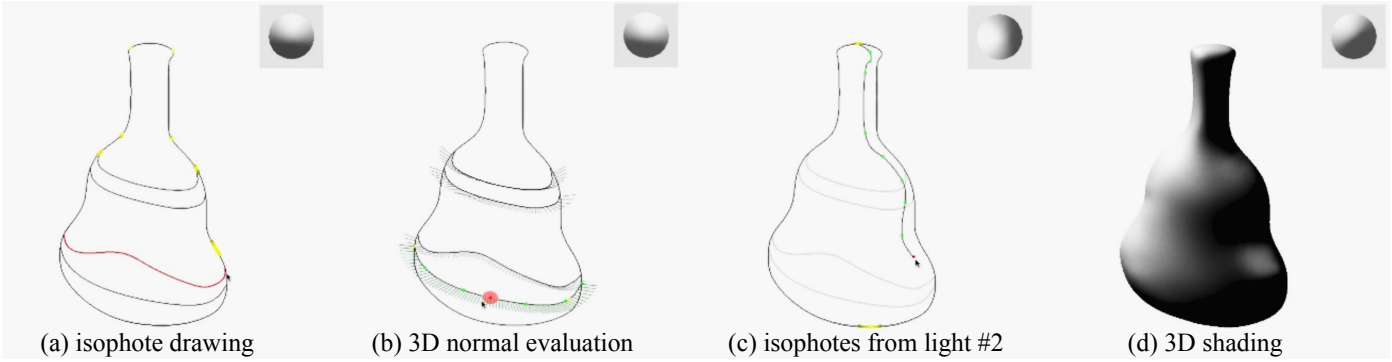
4. Artist Workflow

Our interface (see accompanying video) is essentially a 2D curve drawing and editing program. Artists begin by drawing a 2D silhouette. They may then manually adjust a light source using a light-sphere widget (top-right in Figure 4a–d) or simply begin drawing isophotes (and an optional cast shadow).

Estimating the light direction: A single isophote intersecting the silhouette is sufficient to define the direction of the light. Surface normals at the isophote/silhouette intersections are well-defined 2D silhouette normals, $n_0 = \langle n_{0_x}, n_{0_y}, 0\rangle$, $n_1 = \langle n_{1_x}, n_{1_y}, 0\rangle$ (Figure 3, right). For distinct unit normals with the same illumination value k, the constraint $n_0 \cdot l = n_1 \cdot l = k$ implies that the xy components of the light direction l must be $\pm(n_0 + n_1)$. In 3D, $l = \langle n_{0_x} + n_{1_x}, n_{0_y} + n_{1_y}, z\rangle$, normalized. For a given non-zero isophote value k, the constraint $n \cdot l = k$ results in a quadratic equation for z:

$$z = \pm\sqrt{\left(\frac{1 + n_0 \cdot n_1}{k}\right)^2 - 2(1 + n_0 \cdot n_1)}$$

An average light direction can similarly be computed given multiple isophotes that intersect the silhouette. For isophotes

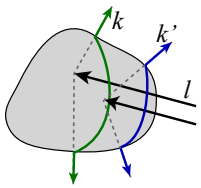


(a) isophote drawing (b) 3D normal evaluation (c) isophotes from light #2 (d) 3D shading

Figure 4: Workflow snapshots: isophote drawing is guided by yellow regions where isophotes with a desired luminance value should intersect the silhouette (a); 3D normals along the isophote can be interactively viewed and corrected using the red gauge (b); isophotes from new light directions can be defined (c); and the normal field evaluated by shading (d).

with illumination value $k = 0$, $n_0 = -n_1$ so only the 2D direction of the light is fixed. At least one isophotes with non-zero k value is needed to define a 3D light direction.

Interaction: Once a light direction l has been estimated or manually defined and an isophote value k specified, our interface highlights points along the silhouette (in yellow) where the desired isophote value is achieved ($n \cdot l = k$) to guide the drawing process towards consistent input (Figure 4a,c). Isophotes are input as a sequence of points beginning and ending on the silhouette; the points are interpolated with a natural cubic spline. If the silhouette, light direction, or isophote value is subsequently edited, the isophote is affinely transformed on the 2D drawing, so that the two old isophote-silhouette intersection points map to new isophote-silhouette intersections (as shown inset where the isophote value is changed from k to k'). The 3D normals along drawn isophotes can be viewed interactively and corrected smoothly at any point using a red gauge figure [34], constrained to rotate around the normal-cone (Figure 4b). Multiple toon images drawn from different light directions further help constrain our problem (Figure 4c). Where isophotes from different light directions intersect, the surface normal is one of two intersection directions of the two normal-cones (if they intersect). Intersecting isophotes with non-intersecting normal-cones are visualized as inconsistent. The 3D normal field from the current toon shaded input can be computed at any time and evaluated using 3D shading under interactive lighting (Figure 4d). Additional isophotes from the current 3D normal field can also be extracted and subsequently edited. The cast shadow, for lifting normal fields into height fields, is drawn as either a 2D boundary curve or as a 2D region whose boundary is automatically extracted.



5. 3D Normal Field Algorithm

Our algorithm proceeds in four stages as shown by the green path in Figure 5. The alternate paths suggest that intermediate steps could be considered optional. However, we find empirically that this sequence of all four stages provides the best results

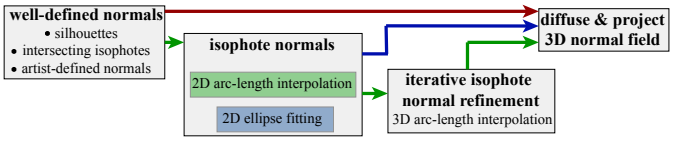


Figure 5: Stages of the 3D Normal Field Algorithm.

overall. Our proposed algorithm is thus: compute well-defined normals, interpolate normals along isophotes based on 2D arc-length, iteratively re-interpolate normals based on 3D isophote arc-length, and iteratively diffuse normals into the interior and project normals along isophotes back to the normal-cones.

5.1. Well-Defined Normals

The 3D surface normals along the silhouette of a smooth shape as shown in (Figure 3, right) are $n = \langle n_x, n_y, 0 \rangle$, where $\langle n_x, n_y \rangle$ is the 2D silhouette normal. Where isophotes from different light directions l_1, l_2 intersect, the surface normal n is one of two intersection directions of the two normal-cones obtained as the solution to a quadratic equation from $n \cdot l_1 = k_1$, $n \cdot l_2 = k_2$, and $\|n\| = 1$. Assuming the isophotes are consistent and a solution exists, our desired normal n must have $n_z > 0$. If only one of the two normals has $n_z > 0$ (both n_z cannot be negative since we assume front lighting), we select it as a normal constraint. If there are two normals with $n_z > 0$, we select the normal that is closer (the one with larger dot product) to the currently estimated surface normal, and re-run the algorithm with the normal constraint. If the normal cones do not intersect, we ask the user for input adjustment.

5.2. Estimating Normals Along Isophotes

With the aim of producing smoothly varying normals with global fairness along isophote curves, we consider two alternatives. The first is a linear interpolation of the cone-angle along an isophote between two known normals, which ensures minimally varying normals along the isophote. The second is a reconstruction of isophotes in 3D as piecewise line segments and circular arcs, producing fair isophotes of minimally varying curvature.

2D arc-length based cone-angle interpolation. As shown in Figure 3, surface normals along an isophote between two known normals n_0 and n_1, are minimally interpolated by linearly interpolating the cone angle t, between t_0 and t_1. We divide each isophote into segments between normals computed in Section 5.1, and perform piecewise linear interpolation. Interpolation of t is circular and can take place around one of two directions between t_0 and t_1. Note that while n_{0z} and n_{1z} for any two known normals are non-negative, there may be a segment of t values where the cone-angle normals n are back-facing ($n_z < 0$). If such a segment exists, we pick the interpolation direction where the cone-angle remains positive, or else we pick the direction with the smaller interpolation angle. While this approach works well for simple isophote shapes, it is unacceptable for complex isophotes as shown in Figure 6. As the 2D isophote is indicative of its 3D shape, it is reasonable to expect that, relative to convex segments, the surface normal in flat and concave regions of the 2D isophote, will stay constant and reverse interpolation direction, respectively.

We thus further segment the 2D isophote segment into convex, flat and concave regions according to McCrae et al. [35] (Figure 6). Let there be m such segments $S_0, \ldots, S_m$. Let the 2D turning angle between the 2D tangent at the start and end of the i-th segment be ϕ_i. If Δt is the overall change in cone-angle, we define the cone-angle change for the i-th segment δt_i as

$$\delta t_i = \frac{\phi_i \Delta t}{\sum_0^m \phi_i}.$$

By definition of ϕ_i, δt_i will now be zero for flat segments and of opposite sign for convex and concave segments. Within each segment S_i, δt_i is still linearly interpolated by 2D arc-length, as the relationship between 2D and 3D curvature is not necessarily proportional due to foreshortening.

There are still two issues with the above formulation that need to be resolved. First, the above formulation is unstable when the overall start and end tangent for the isophote segment are nearly equal $\sum_0^m \phi_i \approx 0$. Barring straight-line isophotes with constant surface normal, we rarely encounter this scenario in practice and handle it by asking the user to specify a normal within the segment, breaking it into two well-defined segments. Second, the cone-angle interpolation is no longer guaranteed to lie within Δt (due to direction reversal in concave regions) and can result in backfacing normals. Let the isophote attain its most negative normal after the i-th segment S_i at a t value $t_{\min}$, i.e. $t_{\min} = t_0 + \sum_0^i \delta t_i$. Let its adjacent t value where the normal is in the viewplane $n_z = 0$ be t_{sil}. We can compute new scaled interpolation angles for segment from $0 \ldots i$ to be

$$\delta t_i \frac{t_{\text{sil}} - t_0}{t_{\min} - t_0}$$

and for segments from $i+1 \ldots m$ to be

$$\delta t_i \frac{t_1 - t_{\text{sil}}}{t_1 - t_{\min}}.$$

It is straightforward to verify that the cone-angle now interpolates from t_0 to t_{sil} at the end of segment S_i, and from t_{sil} to t_1 and the end of segment S_m, with strictly front-facing normals.

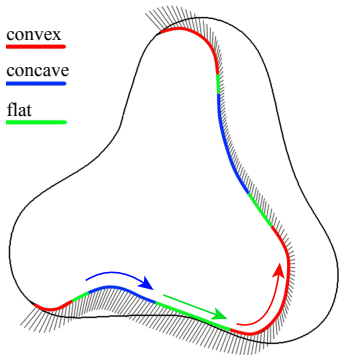


Figure 6: Cone-angle interpolation stops in flat green segments of an isophote and reverses direction in concave blue regions.

For closed interior isophotes, we require that they be intersected by at least one isophote from a different light, so that it may be segmented by at least two well-defined normals.

Elliptical arc fitting. We are motivated by the desire to fit simple 3D surface primitives whose isophotes would match the drawn isophotes. Unfortunately even the isophotes of simple 3D ellipsoids are generally non-planar curves called spherical cyclic curves. We thus pick a line and a circle, the simplest 3D isophotes that we can reconstruct from their drawn 2D projections. If we assume a 2D straight line is a straight line in 3D [18], the isophote segment lies on a ruled surface with constant surface normal.

Let us model a circular arc in object space as $(0, \cos(a), \sin(a))$ for some range $[a_0, a_1]$. (In object space, the circle lies on a plane perpendicular to the X axis.) The surface normal for a ribbon around this arc is

$$\frac{\langle x, \cos(a), \sin(a) \rangle}{\sqrt{1 + x^2}}$$

for some x. The above circular arc becomes an ellipse in the xy image plane as a result of rotation $R_y(u)$ by angle u about Y and $R_z(v)$ by v about Z. The aspect ratio of this ellipse is now $\sin(u)$, and the angle of its minor axis to the X-axis is v. Given any 2D elliptic arc, we can determine its transformation $M = (R_z(v)R_y(u))^{-1}$ from image to the object space of the circular arc from u and v. We can thus transform the light l from image to object space $l' = Ml$. In object space the isophote equation is

$$x l'_x + \cos(a) l'_y + \sin(a) l'_z = k\sqrt{1 + x^2}$$

where k is the isophote value, and x can be solved as a quadratic equation whose roots are functions of the circular-arc parameter a. If there are no solutions, we snap to the solution for the closest k value, and project this solution normal back to the desired normal-cone. As a sanity check, when the light direction is perpendicular to the circular arc, x is a constant function of k and the ribbon around the circular isophote is locally cone shaped.

For each convex and concave isophote segment, we fit a minimal number of 2D ellipses given an error tolerance (3 pixels in our implementation) [36]. As pointed out by the recent approach of Company et al. [37], the approach we use is slower

than needed for some data, not as robust as needed for other data, and our error tolerance is not perceptually motivated. As an alternative, we could use the approach of Company et al. [37], which chooses between three different ellipse fitting approaches based on perceptual accuracy and running time and returns a perceptual goodness-of-fit. For each ellipse, which we assume to be the projection of a circle, there are 4 possible circle plane normals (2 tilt directions for angle v, and convex or concave normals). Segments that intersect the silhouette are convex by virtue of a smooth 3D shape and have two choices. Additional criteria are: the normal at the shared point between adjacent segments should match; the normals within each segment should vary smoothly; the normals have positive z components. We define a fitness metric for any combination to be the overall sum of absolute differences between cone-angle t at the common point of adjacent segments, sum of the standard deviation of cone-angle t within each segment, and the percent of normals with negative z components. We permute all $2 \cdot 2 \cdot 4^{\#\text{segments}-2}$ possible combinations and pick the fittest.

5.3. Iterative 3D arc-length based interpolation

The 2D arc-length based interpolation of Section 5.2 ignores foreshortening effects. As a result, normals along the 3D circular isophote of a sphere, would turn quickly near the silhouette and appear flat in the interior, as dictated by the arc length of a 2D ellipse. Alternatively, the ellipse fitting algorithm has discontinuities at segment junctions. We improve these estimated normals by iteratively estimating a 3D isophote. An estimated surface normal n along an isophote is perpendicular to its 3D isophote tangent $\langle t_x, t_y, t_z \rangle$, as

$$t_z = -\frac{n_x t_x + n_y t_y}{n_z}$$

where $\langle t_x, t_y \rangle$ is the known 2D isophote tangent. Given surface normals along an isophote, we compute 3D tangents as just described. Near silhouettes and where $n_z < \epsilon$ (we use $\epsilon = 0.004$), we compute t_z using l'Hospital's formula and finite differences, i.e.

$$t_z = -\frac{\Delta(n_x t_x + n_y t_y)}{\Delta n_z}.$$

We then smooth t_z minimally by neighbor averaging (strength 0.3), recompute normals using the equations $n \cdot t = 0$, $n \cdot l = k$, $\|n\| = 1$, and repeat the process for a fixed number of iterations or until the normals converge.

5.4. Diffuse and Project across the Shape

We treat the interior of the shape as a 3D normal field $\mathbb{R}^2 \rightarrow \mathbb{R}^3$. The known $\langle n_x, n_y, n_z \rangle$ values for normals on the silhouette and at isophote intersections are boundary constraints for a diffusion process. Estimated normals along isophotes, and user provided additional normals, are included as soft constraints. After diffusion, the computed normals along isophotes may have deviated from the normal cone. We project them onto the normal cone and update the soft constraints to the new normals. We repeat these diffuse and project steps until convergence. Unlike our previous steps, the diffuse-and-project method does not rely on specific shape properties of isophotes.

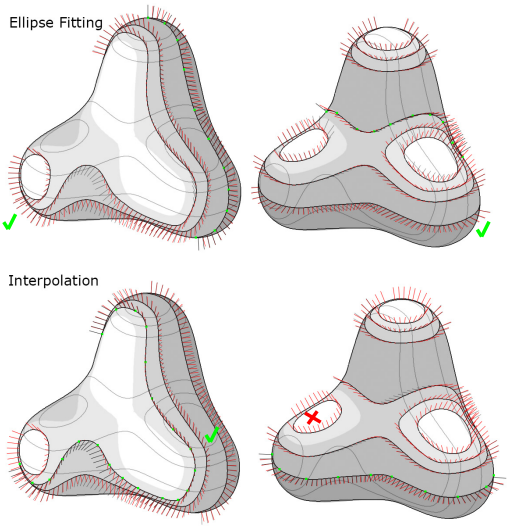


Figure 7: Ellipse fitting and 2D arc-length based cone-angle interpolation produce complementary results.

We presented two approaches in Section 5.2 for initial surface normal estimation along isophotes. While we generally promote 2D arc-length based interpolation for its robust simplicity, ellipse fitting has the advantage that it does not require any known normals along the isophote. It is thus applicable to closed and isolated isophotes for which there are no normals to be interpolated. It also complements 2D arc-length based interpolation by implicitly handling view foreshortening. Figure 7 marks regions where one approach tends to perform better than the other.

As Figure 5 suggests, one could skip normal estimation along isophotes (Sections 5.2 and 5.3) altogether and simply diffuse well-defined normals throughout the shape, projecting the diffused isophote normals onto their normal-cone. Theoretically, we expect poorer results as this process will result in smooth normals along isophotes, but ones that might lack fairness, oscillating back and forth arbitrarily around the normal-cone. We do indeed observe this behaviour in practice. This is discussed in Section 7 and visualized in Figure 9. Both isophote normal estimation stages (Sections 5.2 and 5.3) have a positive impact on the 3D normal field construction.

6. Height Field Algorithm

Our algorithm for producing a height field $H(x, y)$ takes as input a normal field $N(x, y)$, a light direction l, and a hard cast shadow $\mathbf{S}$. Form shadows, in which the shadow is cast onto the shape itself, are left as future work. We assume that the surface is C^1 continuous and closed. We additionally assume that the shadow is cast onto the $z = 0$ plane, although we relax this constraint somewhat in a later stage.

Our algorithm begins by rotating the input around the z axis such that the light direction lies in the xz plane ($l_y = 0$) with $l_x < 0$. (If the light is directly overhead, $l_x = l_y = 0$, and there

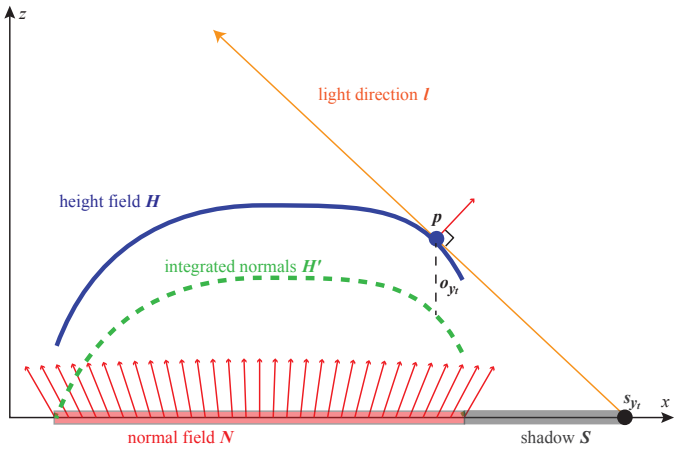


Figure 8: An illustration of the height field algorithm. An xz-plane slice of the normal field N is integrated to obtain a relative height field H'. The absolute height field H is obtained by offsetting H' to graze the light direction l.

would be no cast shadow). Our algorithm processes each horizontal slice (row of pixels) $H(x, y_t)$ for each y_t, independently (Figure 8). For each slice, we collect the shadow and surface extents (1D connected components). For each surface extent, we find the boundary of the corresponding shadow extent $\mathbf{S}_{y_t,i}$ in the $+x$ direction:

$$s_{y_t} = \max_x \text{ such that } x \in \mathbf{S}_{y_t,i}.$$

(If the correct s_{y_t} is occluded and the *visible* shadow boundary is used instead, our algorithm will produce erroneous results (Section 8).) The light that reaches s_{y_t} grazes the surface at some point $p = (x, y_t)$. Because the surface is C^1 continuous, $l \cdot N(p) = 0$. These two criteria inform our algorithm. By integrating the normals along the slice, we obtain a 1D *relative* height field H'. (We integrate by treating pixels as piecewise planar patches with the given normals, and cumulatively sum the z values of the patch gradients.) We find the absolute height field slice $H(x, y_t) = H'(x, y_t) + o_{y_t}$ in two stages. Let $z(x) = mx + b$ be the equation for the line parallel to the light direction and passing through s_{y_t}. First, we find x' such that the offset $o'_{y_t} = z(x') - H'(x_0, y_t)$ is minimized. This is the offset such that the light grazes $H'(x, y_t) + o'_{y_t}$. Second, we incorporate the normal condition. We find the x'' in the range $x' \pm 10$ whose normal $N(x, y_t)$ angle with the light direction is most perpendicular (closest to 90°). The final offset o_{y_t} for extent is determined from the offset at x'': $o_{y_t} = z(x'') - H'(x'', y_t)$.

Denoising. To relax the assumption that the shadow is cast onto the $z = 0$ plane, or to correct inaccuracies in user-drawn shadows, we optionally offset adjacent slices in z to minimize the deviation

$$\min_c \int_a^b |H(x, y_t) - H(x, y_t + 1) + c|^p \, dx.$$

For the ℓ^2 norm ($p = 2$), the minimizer c is the average difference $H(x, y_t) - H(x, y_t + 1)$. For the ℓ^1 norm ($p = 1$), the minimizer c is the median difference $H(x, y_t) - H(x, y_t + 1)$. We use the ℓ^1 norm, as it is more robust to outliers.

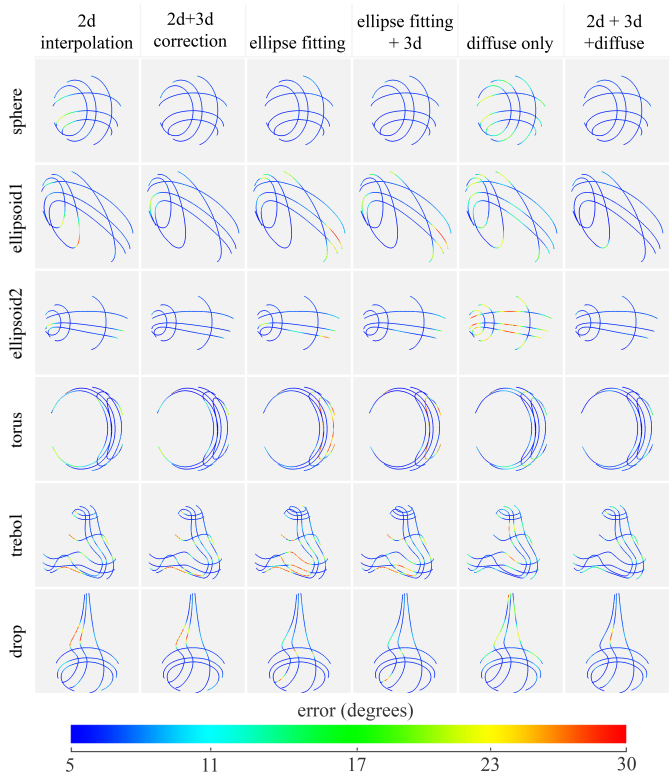


Figure 9: Impact of algorithm stages on shapes in Figure 13. We measure angular error along isophotes with respect to ground truth normals. A numerical summary is shown in Table 1.

7. Validation

Algorithm Evaluation. Figure 9 shows the impact of each stage of the algorithm in Section 5 on the reconstruction error of ground truth 3D data. Ground truth data in all of our experiments was obtained from Autodesk Maya. Along each isophote, we measured the angular error with respect to ground truth (Table 1). We evaluated both isophote normal estimation algorithms (Section 5.2) by themselves, and combined with our iterative refinement stage (Section 5.3). In isolation, 2D arc-length based interpolation performed comparably to ellipse fitting. Iterative 3D refinement decreased the median error for both approaches, but more significantly for 2D interpolation (correcting foreshortening errors). Our diffuse-and-project algorithm for propagating normals through the shape produces poor results in isolation, typically worse than our isophote normal estimation algorithms *without* 3D refinement. Combining the stages of 2D arc-length interpolations, 3D refinement and diffuse-and-project produced the lowest median error of $\approx 5.5°$.

Figure 10 shows our algorithm to be robust to the incremental addition and removal of isophotes, with predictable changes in the resulting 3D normal field.

Perceptual Study I. We carried out the first perceptual study to answer the following three questions about toon shaded images:

Q1 Do humans perceive isophote normals consistently?

Q2 Are humans accurate (consistent with ground truth)?

	2d		2d + 3d		ellipse fitting		ellipse + 3d		diffuse only		2d + 3d + diffuse	
	med.	std.	med.	std.	med.	std.	med.	std.	med.	std.	med.	std.
sphere	6.51	4.55	4.94	3.35	4.99	3.4	4.94	3.35	8.85	5.44	4.9	3.24
ellipsoid1	6.33	5.41	5.76	4.11	6.71	6.34	6.6	6.35	8.52	5.5	5.3	3.45
ellipsoid2	6.29	4.05	5.51	3.38	6.28	5.19	5.84	3.81	14.11	14.52	5.46	3.3
torus	7	5.3	6.06	5.09	7.9	12.97	6.27	12.31	7.6	5.05	5.97	4.89
trebol	7.21	9.21	7.04	9.57	8.15	11.63	7.39	9.69	7.7	5.73	6.4	4.63
drop	7.09	7.48	6.1	6.95	6.27	4.92	5.93	4.56	7.27	5.7	5.86	5.1

Table 1: Summary of our experiments comparing each step of our surface normal estimation algorithm to ground truth. The median (med.) and standard deviation (std.) are shown. A visual summary can be found in Figure 9.

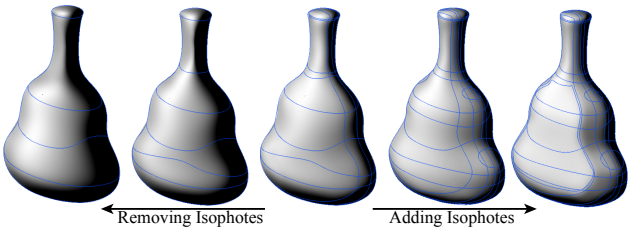


Figure 10: Impact of isophote addition and removal on the 3D shaded result.

Q3 Is our algorithm's accuracy comparable to humans?

Our subject pool consisted of 11 artists and 9 non-artists/non-designers, all with some CG background. Participants were shown a description of toon shading and then completed a series of 52 gauge figure orientation tasks [34] over toon shaded objects. Gauge figures were always placed on the isophote between two discrete tones. The first 26 gauge figures were shown in isolation with only two tonal bands adjacent to the isophote. The entire object was shown for the last 26 gauge figures. The first and second half of each group of 26 gauge figures were either (randomly) freely orientable or constrained to lie on the normal-cone. The constrained scenario, giving participants one degree of freedom, corresponds to our algorithm's constrained solution space. The unconstrained scenario allows us to compare our study's results to previous gauge figure experiments. Table 2 summarizes our results. Our automatic approach for estimating normals along isophotes substantially outperformed humans (Table 1 and 2).

Findings:

- Human perception of isophote normals are more consistent with each other than accurate to ground truth. Humans have a tendency to orient the surface normal perpendicular to the isophote curve in 2D. In many cases the spread of gauges is not centered around the ground truth normal, but biased towards the 2D curve's normal direction (Figure 11a–c).

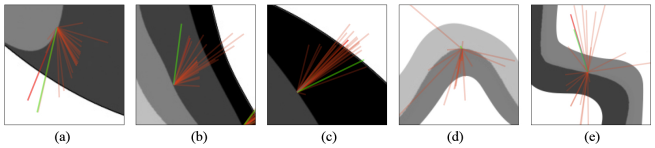


Figure 11: First perceptual study results. Red: user placed normals. Green: ground truth normals.

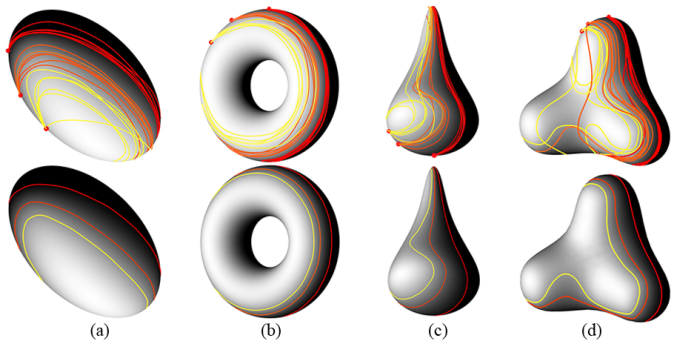


Figure 12: Second perceptual study results. Top row: artist-drawn isophotes. Bottom row: ground truth isophotes. Isophotes of value $k = 0, 0.5$, and 0.7 are shown in red, orange, and yellow, respectively.

- Human accuracy at the crest shadow (tone value $k = 0$) is notably higher than along isophotes with lighter values. The average median error of all k = 0 test cases is 14°, and 20.3° for other values. For non-extreme light directions, the crest shadow's shape is closest to the silhouette, where the 2D shape is indeed representative of the 3D normal. Human consistency with our algorithmic result is higher than with ground truth. While perhaps surprising, we designed our algorithm around the assumption that 2D isophote shape is indicative of local 3D geometry, which is inspired by human perception [4].

- Isophotes are ambiguous when the complete shape is not shown (Figure 11d,e).

- Human consistency and accuracy is better when the gauge is constrained to lie on the normal-cone. The artist pool reported $\approx 2°$ better consistency and accuracy than non-artists.

Perceptual Study II. We carried out a second study to assess artists' ability to draw isophotes consistently and accurately. Our subject pool consisted of 5 artists. Participants were shown 4 Lambertian shaded shapes and then asked to draw 3 isophotes ($k = 0$, 0.5, and 0.7) for each shape. The starting point of each isophote was marked as a red dot on the silhouette. The results of this study are summarized in Figure 12.

Findings:

	Pers. intra user	Cons. inter user	Acc. user/GT	Acc. user/GT - outliers	Cons. user/algo	Acc. algo/GT
Complete (constrained movement)						
median	7.1	8.2	16.4	15	14.5	5.6
mean	10.1	14.1	20.2	17.9	15.6	5.8
std. dev.	16.3	17.1	16.8	13.5	13.4	4.2
samples	98	1616	314	299	314	65
Complete (free movement)						
median	9.3	16	19.5	18.8	19.1	
mean	10.5	19.7	22.3	20.1	21.2	
std. dev.	15.3	16.1	16.2	12.9	16.1	
samples	186	6260	628	597	628	
Partial (constrained movement)						
median	7.2	8.2	17.3	16	15.9	
mean	11.4	15.4	23.5	20.5	20.2	
std. dev.	17.5	21.5	20.7	15.8	19.8	
samples	103	1681	333	317	333	
Partial (free movement)						
median	7.6	18.3	23.5	22.3	21.2	
mean	11.9	24.1	27.1	24.4	24.3	
std. dev.	18.3	20.8	19.6	15.6	15.1	
samples	205	6580	665	632	665	

Table 2: Summary of our validation data. (Left to right) persistence same user, consistency between users, accuracy with respect to ground truth (GT), accuracy with worst 5% of errors removed, consistency with respect to our solution, accuracy of our solution with respect to ground truth.

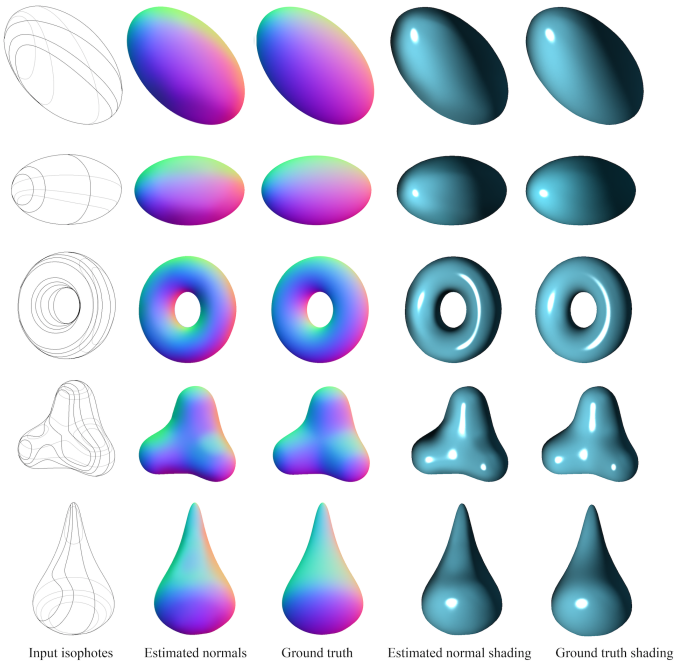


Figure 13: Smooth shading for simple shapes computed with our approach.

- Artists have a good understanding of isophote properties when presented with toon-shaded examples. All user-drawn isophotes from a single light source are continuous and non-intersecting.

- Artist-drawn isophotes are highly consistent at the crest shadow ($k = 0$) and less consistent at other k values. This observation matches the result from **Perceptual Study I**. For more complicated shapes (Figure 12d), artist consistency is notably lower.

- Artist-drawn isophotes are distributed closely to the ground truth. However, most drawn isophotes match a k value higher than the assigned value. This is in line with Perceptual Study I, which found that humans overestimate k.

- When an isophote approaches the silhouette, its tangent starts to follow the tangent of the silhouette. The more perpendicular the light direction is to the image plane, the longer the segment of isophote drawn this way.

8. Results

Implementation Details. Our prototype was implemented on a Intel Core i7 2.3 GHz processor with 8 GB RAM. The interface canvas resolution is 700 by 700 pixels. Typical time for normal estimation by 2D interpolation is negligible. Ellipse fitting takes on the order of 6 seconds. 3D refinement takes 1 sec (75 iterations) and diffusion takes approximately 2 minutes (5 iterations).

User testers. We created a variety of smooth shaded objects with our interface, from simple (Figure 13) to complex (Figures 2, 14, and 17). The results were produced by an author and two experienced artists (non-authors) (Phone, Purse, Head and Horse). Results took between 10 minutes (Pear) and 2 hours (Horse) to create (typically 20 minutes), including time spent learning the interface. Users reported being somewhat uncertain about which luminance value to choose for isophotes and suggested that our interface's "sample illumination sphere" (Figure 4) be shown with toon shading. Notably, users reported being more confident closer to the 0 luminance value, consistent with our perceptual study.

Importantly, our interface is *predictable*. Adding (or removing) a single isophote is a stable operation that incrementally affects the resulting normal field and generally produces the desired result (Figure 10). Simple diffusion from silhouettes as in Lumo [10] does not provide designers with sufficient control to model complex and structured objects (Figure 17d). For even finer control, designers can manually specify normals, such as on the Bunny's belly, or the Head's eye socket (Figure 17).

User testers provided several suggestions to facilitate more accurate user input. Our interface could provide a better visualization of discrete tone-shading by displaying the light-sphere widget with toon shading instead of Lambertian shading; and, when drawing isophotes, our interface could show an isophote on the light-sphere widget with the same brightness as the currently selected one. To allow users to rapidly iterate on their corrections, our interface could display a low-resolution normal field quickly, and solve for a full resolution normal field less frequently. Finally, we could give users control over weights in the diffusion step of our algorithm, via a marking tool to highlight important portions of isophotes.

We evaluate our approach for creating height fields from nor-

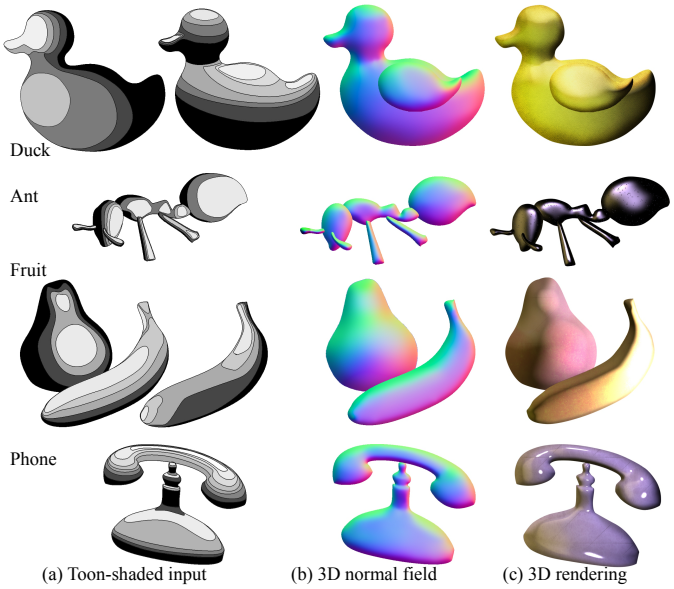


Figure 14: Artist creations using our interface.

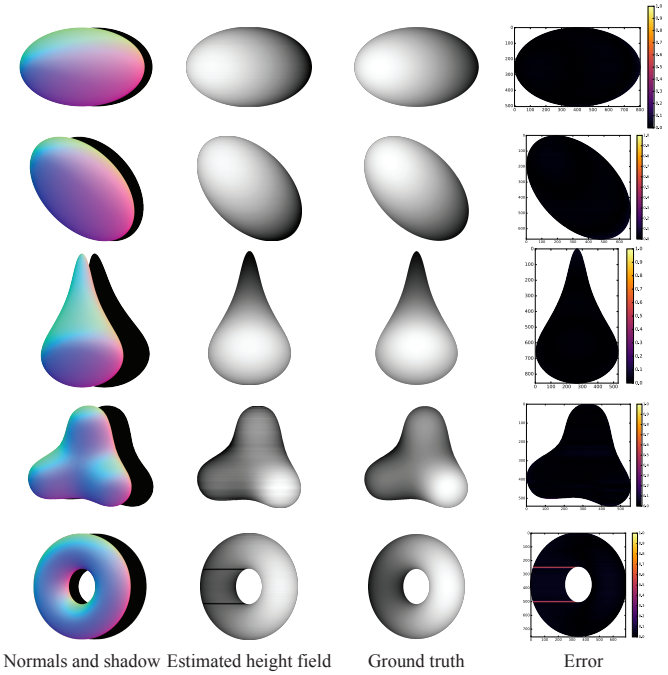


Figure 15: Height fields lifted from ground truth normal maps and shadows. Ground truth height fields are normalized within the range [0,1]. Estimated height fields are scaled and offset in depth to align with ground truth height fields (minimizing the sum of squared differences). The error plots the absolute difference.

mal fields and cast shadows (Section 6) on ground truth data and artist drawings. Figure 15 displays normal fields, ground truth shadows cast onto a $z = 0$ plane, and height fields for known 3D shapes, as well as height fields resulting from our lifting algorithm. These height maps do not employ our denoising algorithm. Our height fields are qualitatively and quantitatively quite similar to the ground truth height fields. Note that the ground truth height fields obtained from Maya have been normalized to lie between 0 and 1. We align height fields generated by our lifting algorithm with ground truth by scaling and offsetting in depth to minimize the sum of squared differences. Note that due to the bas-relief ambiguity [38], height fields are ambiguous up to a uniform scale in the z (depth) direction. The torus illustrates a limitation of our approach. Our algorithm requires that shadow boundaries are visible. However, the shadow boundaries for the slices of the left side of the torus near the top and bottom of the hole are occluded, so an erroneous shadow boundary is detected. This results in incorrect, discontinuous depth.

Figure 16 displays height fields obtained from artist-created normal fields in Figures 2 and 17 augmented with drawn cast shadows. In this setting, our denoising algorithm produces a significant increase in height field quality.

9. Conclusion

For a variety of shapes, smooth 3D normal fields can be quickly created from a toon shaded, or discrete tone image, or even de novo. With the addition of a cast shadow, smooth height fields can be quickly obtained. Drawing sparse isophotes is far easier than meticulously drawing a smoothly shaded image or "drawing" a smooth, dense 3D normal field. We thus propose the novel problem of inverse toon shading and present a plausible algorithm. Every stage of our algorithm is important: diffuse-and-project alone produces smooth but high frequency artifacts along isophotes. 2D arc-length-based interpolation is in accordance with the principle of minimum variation, but does not account for foreshortening, that is handled by iterative 3D refinement, providing a reliable set of surface normals to diffuse-and-project. Our height field algorithm resolves the depth placement resulting from naive integration. The 3D normal fields we generate allows designers to layer shapes like the Head in Figure 17, and experiment with color, materials and lighting when creating presentation renderings.

Our work has several **limitations** that we hope to address in future work. In addition to the various simplifying assumptions made in Section 3, our proposed 2D arc-length-based interpolation along isophotes requires that isophotes intersect the silhouette or other isophotes. Certain single-light isophotes, such as on a severely tilted ellipse (Figure 9, second row), are poorly approximated. Our isophote normal estimation algorithms sometimes require manual correction (red gauge in Figure 4). Our algorithms assume that visible surfaces are C^1 continuous. There can be no C^0 edges or occluding contours. For example, the Horse created using our approach (Figure 2) could not be created from the discrete hatched sketch in Figure 1, because the chin forms an occluding contour. We find that some people do not see isophotes well, even though sketching isophotes is important to drawing instruction [2]. Our height field algorithm assumes that shadow boundaries are visible, yet this assumption may be violated by holes or concave regions in the shape.

Despite these limitations, we believe that inverse toon shading, like cross-section drawing, can be a powerful technique to bridge the gap between rapid 2D sketches and 3D presentation renderings.

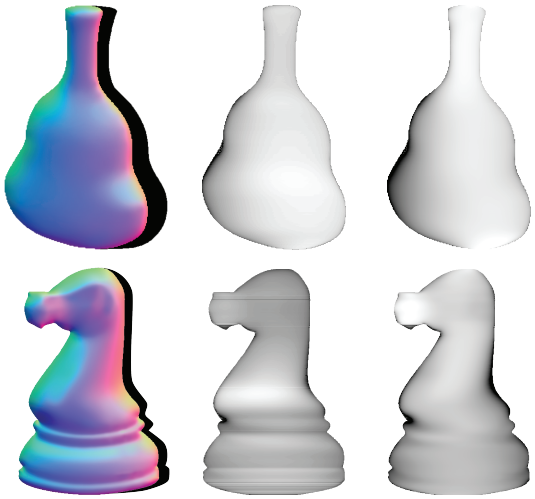


Figure 16: Height fields lifted from normal maps and user-drawn shadows.

Acknowledgements

We are grateful to Adrien Bousseau and the anonymous reviewers for their constructive comments. This work was supported in part by the United States National Science Foundation (IIS-1451198 and IIS-1453018) and a Google research award.

[1] A. L. Guptill, S. E. Meyer, Rendering in Pen and Ink: The Classic Book on Pen-And-Ink Techniques for Artists, Illustrators, Architects, and Designers, Watson-Guptill, 1997.

[2] B. Edwards, The New Drawing on the Right Side of the Brain, Jeremy P. Tarcher/Putnam, ISBN 9780874774245, URL http://books.google.ca/books?id=MKD7NG267mIC, 1999.

[3] K. Eissen, R. Steur, Sketching: Drawing Techniques for Product Designers, Bis Publishers, 2008.

[4] D. Powell, Presentation techniques: a guide to drawing and presenting design ideas, Little, Brown, ISBN 9780316912433, URL http://books.google.co.in/books?id=CAiAQgAACAAJ, 1994.

[5] C. Shao, A. Bousseau, A. Sheffer, K. Singh, CrossShade: Shading Concept Sketches Using Cross-Section Curves, ACM Transactions on Graphics (Proc. SIGGRAPH) 31 (4).

[6] Q. Xu, Y. Gingold, K. Singh, Inverse Toon Shading: Interactive Normal Field Modeling with Isophotes, in: Proceedings of Sketch-Based Interfaces and Modeling (SBIM), 2015.

[7] D. Sýkora, J. Dingliana, S. Collins, LazyBrush: Flexible Painting Tool for Hand-drawn Cartoons, Computer Graphics Forum (Proc. EUROGRAPHICS) 28 (2).

[8] A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, D. Salesin, Diffusion Curves: A Vector Representation for Smooth-Shaded Images, ACM Trans. on Graph. (Proc. SIGGRAPH) 27.

[9] M. Finch, J. Snyder, H. Hoppe, Freeform vector graphics with controlled thin-plate splines, ACM Trans. on Graph. (Proc. SIGGRAPH Asia) 30.

[10] S. F. Johnston, Lumo: illumination for cel animation, in: Proc. Symp. on Non-Photorealistic Animation and Rendering, 2002.

[11] H. Winnemöller, A. Orzan, L. Boissieux, J. Thollot, Texture Design and Draping in 2D Images, Computer Graphics Forum (Proc. Symp. on Rendering) 28 (4).

[12] D. Sýkora, L. Kavan, M. Čadík, O. Jamriška, A. Jacobson, B. Whited, M. Simmons, O. Sorkine-Hornung, Ink-and-ray: Bas-relief Meshes for Adding Global Illumination Effects to Hand-drawn Characters, ACM Trans. Graph. 33 (2) (2014) 16:1–16:15, ISSN 0730-0301, doi:10.1145/2591011, URL http://doi.acm.org/10.1145/2591011.

[13] L. Zhang, G. Dugas-Phocion, J.-S. Samson, S. M. Seitz, Single View Modeling of Free-Form Scenes, in: IEEE Computer Vision and Pattern Recognition, 990–997, 2002.

[14] M. Okabe, G. Zeng, Y. Matsushita, T. Igarashi, L. Quan, H. yeung Shum, Single-view relighting with normal map painting, in: Proc. Pacific Graphics, 27–34, 2006.

[15] T.-P. Wu, C.-K. Tang, M. S. Brown, H.-Y. Shum, ShapePalettes: interactive normal transfer via sketching, ACM Trans. on Graph. (Proc. of SIGGRAPH) 26, ISSN 0730-0301.

[16] J.-D. Durou, M. Falcone, M. Sagona, Numerical methods for shape-from-shading: A new survey with benchmarks, Computer Vision and Image Understanding 109 (2008) 22–43, ISSN 1077-3142, doi:10.1016/j.cviu.2007.09.003, URL http://portal.acm.org/citation.cfm?id=1326363.1326482.

[17] R. J. Woodham, Photometric method for determining surface orientation from multiple images, Optical Engineering 19 (1) (1980) 191139–191139, doi:10.1117/12.7972479, URL http://dx.doi.org/10.1117/12.7972479.

[18] B. Xu, W. Chang, A. Sheffer, A. Bousseau, J. McCrae, K. Singh, True2Form: 3D Curve Networks from 2D Sketches via Selective Regularization, Transactions on Graphics (Proc. SIGGRAPH) 33 (4), doi:2601097.2601128.

[19] P. Joshi, N. A. Carr, Repoussé: Automatic Inflation of 2D Artwork, in: Proc. of Sketch Based Interfaces and Modeling, 2008.

[20] D. Cremers, Fast and Globally Optimal Single View Reconstruction of Curved Objects, in: Computer Vision and Pattern Recognition, CVPR, ISBN 978-1-4673-1226-4, 534–541, URL http://dl.acm.org/citation.cfm?id=2354409.2354928, 2012.

[21] T. Igarashi, S. Matsuoka, H. Tanaka, Teddy: a sketching interface for 3D freeform design, SIGGRAPH .

[22] A. Ecker, K. Kutulakos, A. Jepson, Shape from Planar Curves: A Linear Escape from Flatland, in: IEEE Computer Vision and Pattern Recognition, doi:10.1109/CVPR.2007.383020, 2007.

[23] P. Tan, T. Zickler, A projective framework for radiometric image analysis, in: Computer Vision and Pattern Recognition, CVPR, IEEE, 2977–2984, 2009.

[24] M. Kaplan, E. Cohen, Producing Models from Drawings of Curved Surfaces, in: Sketch-Based Interfaces and Modeling, SBM, ISBN 3-905673-39-8, 51–59, doi:10.2312/SBM/SBM06/051-058, URL http://dx.doi.org/10.2312/SBM/SBM06/051-058, 2006.

[25] Y. Gingold, T. Igarashi, D. Zorin, Structured Annotations for 2D-to-3D Modeling, ACM Trans. on Graph. (Proc. SIGGRAPH Asia) 28 (5), ISSN 0730-0301.

[26] Y. Gingold, D. Zorin, Shading-based surface editing, Transactions on Graphics (Proc. SIGGRAPH) 27 (3) (2008) 95:1–95:9, ISSN 0730-0301.

[27] H. Theisel, Are Isophotes and Reflection Lines the Same?, Comput. Aided Geom. Des. 18 (7) (2001) 711–722, ISSN 0167-8396, doi:10.1016/S0167-8396(01)00063-2, URL http://dx.doi.org/10.1016/S0167-8396(01)00063-2.

[28] T. Poeschl, Detecting Surface Irregularities Using Isophotes, Comput. Aided Geom. Des. 1 (2) (1984) 163–168, ISSN 0167-8396, doi:10.1016/0167-8396(84)90028-1, URL http://dx.doi.org/10.1016/0167-8396(84)90028-1.

[29] A. Deslandes, D. L. Bonner, Reflection line control, uS Patent 6,717,579, 2004.

[30] J. Loos, G. Greiner, H.-P. Seidel, Modeling of Surfaces with Fair Reflection Line Pattern, in: Shape Modeling and Applications, SMI, ISBN 0-7695-0065-X, 256, URL http://dl.acm.org/citation.cfm?id=829509.830285, 1999.

[31] E. Tosun, Y. Gingold, J. Reisman, D. Zorin, Shape optimization using reflection lines, in: Eurographics Symposium on Geometry Processing, SGP, ISBN 978-3-905673-46-3, 193–202, 2007.

[32] B. Gooch, A. Gooch, Non-Photorealistic Rendering, AK Peters Ltd, iSBN: 1-56881-133-0, 2001.

[33] T. Goodwin, I. Vollick, A. Hertzmann, Isophote Distance: A Shading Approach to Artistic Stroke Thickness, in: Proceedings of Non-photorealistic Animation and Rendering, NPAR, ISBN 978-1-59593-624-0, 53–62, doi:10.1145/1274871.1274880, URL http://doi.acm.org/10.1145/1274871.1274880, 2007.

[34] J. J. Koenderink, A. J. V. Doorn, A. M. L. Kappers, Surface perception in pictures, Perception & Psychophysics (1992) 487–496.

[35] J. McCrae, K. Singh, Sketching piecewise clothoid curves, Computers & Graphics 33 (4) (2009) 452–461.

[36] R. Halíř, J. Flusser, Numerically stable direct least squares fitting of el-

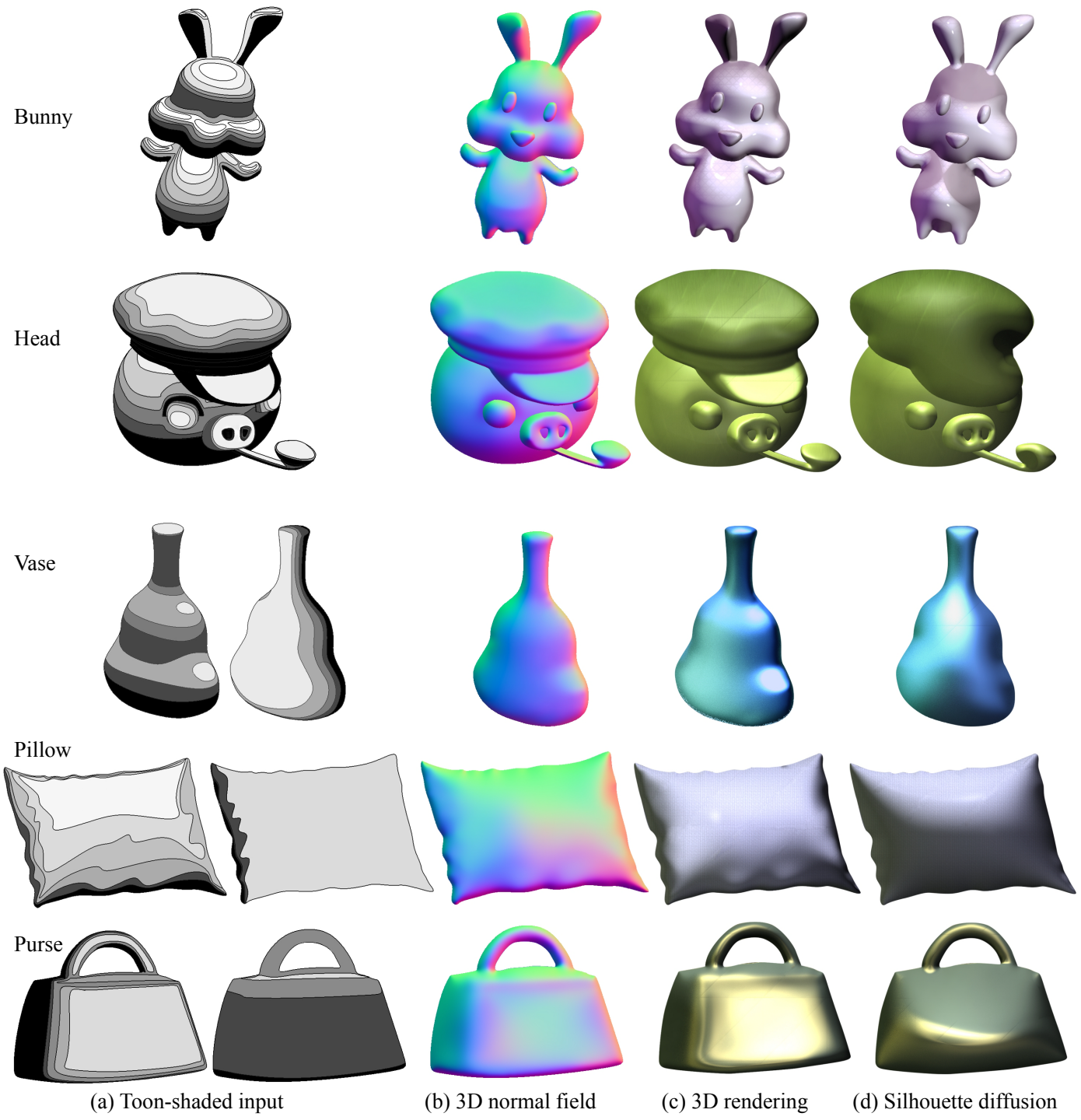


Figure 17: Artist creations using our interface. Artists created toon-shaded inputs from one or two light directions (a). The isophotes between discrete shading levels were input into our system to obtain a 3D normal field (b). 3D renderings of our results (c) show the control and detail of isophotes over simple silhouette diffusion from (d).

lipses, in: 6th International Conference in Central Europe on Computer Graphics and Visualization (WSCG), vol. 98, 125–132, 1998.

[37] P. Company, R. Plumed, P. A. Varley, A fast approach for perceptually-based fitting strokes into elliptical arcs, The Visual Computer 31 (6) (2015) 775–785, ISSN 0178-2789, doi:10.1007/s00371-015-1099-6, URL http://dx.doi.org/10.1007/s00371-015-1099-6.

[38] P. N. Belhumeur, D. J. Kriegman, A. L. Yuille, The Bas-Relief Ambiguity, in: Proceedings of IEEE CVPR, 1060–1066, 1997.