Shading-Based Surface Editing

Yotam Gingold* and Denis Zorin
New York University

Abstract

We present a system for free-form surface modeling that allows a user to modify a shape by changing its rendered, shaded image using stroke-based drawing tools. User input is translated into a set of tangent and positional constraints on the surface. A new shape, whose rendered image closely approximates user input, is computed using an efficient and stable surface optimization procedure. We demonstrate how several types of free-form surface edits which may be difficult to cast in terms of standard deformation approaches can be easily performed using our system.

CR Categories: I.3.6 [Methodology and Techniques]:Interaction techniques I.3.5 [Computational Geometry and Object Modeling]: Geometric algorithms, languages, and systems

Keywords: sketch-based modeling, deformations, interactive modeling, image-based modeling

1 Introduction

Three-dimensional models are often created and manipulated using two-dimensional interfaces. User actions are typically translated into the three-dimensional motion of spline or subdivision surface control points, or into the three-dimensional motion of points on the surface, with the rest of the surface deforming variationally. The relationship between user actions and changes in appearance is indirect: the effect on appearance may be hard to predict, and conversely, it may be difficult to decide which deformation has to be applied to achieve a particular visual effect; e.g., make the outline smoother, remove an unwanted shadow in a view of a model, or reshape a highlight.

In this paper, we describe a sketch-based modeling technique based on changing shaded images of three-dimensional models *directly*, using free-form strokes for two-dimensional image editing. The shape is automatically adjusted to match desired changes of appearance by minimizing a quadratic functional with tangent and positional constraints deduced from user image modifications. This approach complements many other sketch-based modeling techniques. For example, an overall shape can be designed using a system similar to Teddy or FiberMesh [Igarashi et al. 1999; Nealen et al. 2007] and further refined using a combination of our system, displacement editing, and silhouette editing [Nealen et al. 2005].

Our choices of algorithms and user interface elements are guided by a general principle: *if a user makes a small change in surface appearance, the resulting shape change should be small.* For most types of modeling interfaces, this continuity of modifications

*e-mail: gingold@mrl.nyu.edu

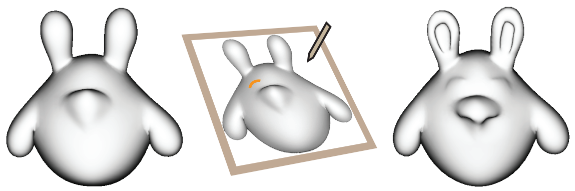

Figure 1: *In our system, users edit 3D models by drawing 2D shading strokes.*

is nearly automatic. For shading-based modeling, because a local modification of shading may require a global shape modification, enforcing this principle is inherently difficult. This can be seen for simple stroke examples in Figure 6. Furthermore, the task of inferring shape changes from image changes is closely related to the problem of recovering the shape from a single image. This is a classic problem in computer vision, which in a standard formulation (Lambertian surface, orthographic projection, directional light) is known to be ill-posed.

Compared to recovering a shape from shading or normal information, in our setting we have the advantage of access to the unmodified shape and the ability to control which surface changes are allowed to happen. At the same time, we face additional difficulties, most importantly:

- some types of small shading modifications lead to large and unintuitive model changes (see Section 3);
- one needs to preserve existing surface detail during editing;
- it is often necessary to keep the surface exactly unchanged far from the modified area and the transition between the modified area and the rest of the surface smooth;
- surface updates should happen at interactive rates.

Our technique to solve these problems has two complementary parts. First, we design user-interface tools (primarily stroke-based) which retain the intuitive feel of two-dimensional drawing and painting brushes. These tools ensure that image modifications that may lead to unexpected and discontinuous surface changes are not possible (Section 3). Second, the restricted class of image modifications permitted by the stroke-based interface allows us to update the surface by solving a quadratic optimization problem, without assuming small deformations or image changes. In contrast, typical shape-from-shading techniques solve a much more general problem, but require solving more complex nonlinear equations.

2 Related Work

We build on a broad range of work in the areas of variational surface modeling, sketch-based modeling, and shape-from-shading reconstruction (SfS). Our work is most closely related to [van Overveld 1996], [Bourguignon et al. 2004], [Kerautret et al. 2005], and [Wu et al. 2007; Ng et al. 2007].

[van Overveld 1996] is, as far as we know, the first paper to introduce the idea of shading-based modeling. This paper describes a system for editing height fields sampled on a regular grid. The user modifies the gradients of the height field directly. [Rushmeier et al. 2003] presents a technique for mesh modification and repair based on a variational shape-from-shading algorithm. [Bourguignon et al.

2004] uses equal height region propagation to reconstruct height fields from a set of initial equal height contours specified by the user and hand-drawn shading information. [Wu et al. 2007] introduces a paradigm for modeling shapes by transferring normal information from a a reference shape (shape palette) to the modeled height field. [Ng et al. 2007] describes a method for recovering a surface from dense or sparse normal information using radial basis functions in a variational context, without restricting the resulting surface to be a height field, also found to be essential in our context.

In contrast to previous work, we do not use SfS or gradient recovery methods directly. Rather, we focus on modification tools for existing surfaces that yield predictable and controllable surface changes. Most of our operations are based on simple strokes, painted by the user, considerably narrowing the scope of the problem and allowing it to be posed as a quadratic optimization problem. A related idea of painting strokes (silhouettes and suggestive contours [DeCarlo et al. 2003]) appeared in [Nealen et al. 2005].

The techniques we use for surface updates extend Laplacian and gradient-based surface editing [Sorkine et al. 2004; Yu et al. 2004] (see a recent comprehensive survey [Botsch and Sorkine 2008]), which can be thought of as data-driven variational modeling [Celniker and Gossard 1991; Moreton and Séquin 1992; Welch and Witkin 1992; Botsch and Kobbelt 2004]. Following other sketch-based modeling work (e.g., [Igarashi et al. 1999; Cheutet et al. 2004; Lawrence and Funkhouser 2004; Kara et al. 2006; Karpenko and Hughes 2006; Nealen et al. 2007]), we use free-form sketching as a user interface for surface shape changes. Silhouette-based techniques [Nealen et al. 2005; Zimmermann et al. 2007] are complementary to ours: our system allows for creating silhouettes, which can then be edited using these techniques.

Using modifiable strokes also resembles curve-based modeling tools [Singh and Fiume 1998; Schaefer et al. 2004; Nealen et al. 2007].

SfS is a well-studied problem in computer vision (the state of the art is surveyed in detail in [Prados 2004]). In its simplest setting, the scene is assumed to have a single Lambertian, directional light source and an orthographic camera. A variety of techniques for SfS were proposed, including variational (such as in [Rushmeier et al. 2003] for modeling) and various types of front propagation methods (e.g. in [Bourguignon et al. 2004]). Variational techniques typically use a smoothness penalty term, which can be thought of as a special case of differential coordinate surface deformation with the reference surface being flat.

Some recent methods use user-specified normals or gradients as additional information to solve the SfS problem [Zeng et al. 2005], or they infer the surface from sparse normal information [Zhang et al. 2001].

3 Shading Changes to Shape Changes

An ideal shading-based modeling system would allow the user to make arbitrary changes to the rendered image of an object, and the resulting modified surface would appear visually indistinguishable from the user-modified image, *while guaranteeing the stability of surface changes and satisfying boundary constraints.* (We use stability in the sense that small changes produce small effects.)

Several fundamental aspects of the relationship between the shaded image and the corresponding shape make such an ideal system impossible. Rather than attempting to match an arbitrary change of the surface image exactly, our system restricts the types of changes that can be applied, and attempts to approximate the target image modification as closely as possible, without causing unstable changes. We briefly consider several aspects of converting a change in shading into a shape change to motivate our design choices.

Shading changes along curves. Our interface is stroke-based; a typical case for which we need to solve the shape recovery problem is darkening or brightening an image along a curve of uniform thickness, while keeping it unchanged elsewhere. Consider a vertical stroke across a flat square patch, with the light and view directions coinciding (Figure 2). In this case, there are three areas of constant shading. We assume the solution to be continuous and to have well-defined normals in each area. If the left boundary is fixed, the solution can be recovered uniquely for any darkening stroke, up to a choice of one of two slopes for each area (the *slope ambiguity*, which we discuss below). We observe that the surface modification amounts to rotating the normals in the area under the stroke about the stroke direction. An additional complication arises when two sides of the patch are fixed: the target image can no longer be matched exactly. However, the solution obtained by minimizing the L_2 norm of the error in shading is also obtained by rotating the normals under the stroke about the stroke direction.

The cases of darkening and brightening strokes are asymmetrical. For certain combinations of light direction and degree of brightening, there is no continuous solution. The reason for this is that brightening to values close to maximal essentially prescribes the normal everywhere on the stroke; it has to point towards the light source. Furthermore, this discontinuous solution is not stable; for sufficiently long strokes, a small shading change (brightening) can produce a large shape change. In our model square patch example, continuity requires that normals are rotated about the stroke direction.

In our system, we use this observation to maintain stability: normals can rotate only in the planes perpendicular to the stroke (Section 5.2).

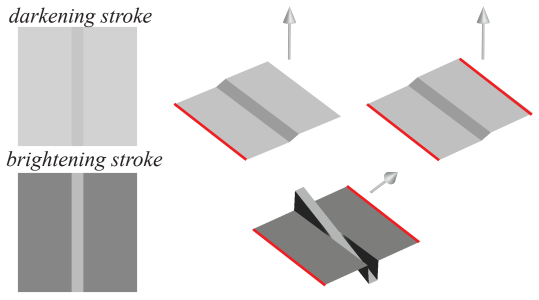

Figure 2: *A vertical stroke across a flat square patch. Red lines are fixed edges and the arrow represents the coinciding light and view directions. Above: a darkening stroke; a continuous solution for one side fixed; an approximate solution for two sides fixed. Below: a brightening stroke. For this light direction and stroke intensity, all solutions are discontinuous (with either one or two sides fixed).*

Instability near highlights. In some situations, a small change in shading may require a large change in the surface shape (Figure 3), contradicting the interface stability principle we have adopted. A common situation of this type is related to *highlight removal*. If the viewer and light positions coincide, the highlight points are also the extremal points of the distance from the surface to the image plane, requiring a large change to the surface to remove. Clearly, for a closed surface there is always a point closest to the light; we conclude that generally, *a highlight cannot be erased* by a smooth surface deformation, so decreasing highlight intensity even by a small amount requires a large change in the surface shape. To prevent this, we terminate strokes which attempt to erase highlights before the stroke reaches the highlight (Section 5.3).

Slope ambiguity. It is well known that images can be ambiguous: concave and convex objects can have exactly the same image (Fig-

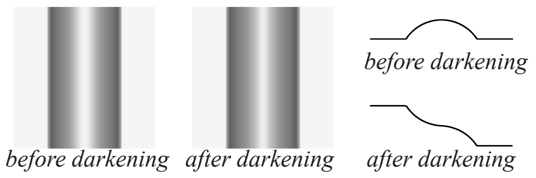

Figure 3: *Unstable change of intensity: a stroke, shown in yellow, applies very small (less than 1%) darkening to a highlight, an imperceptible change. To effect this change, a part of the exactly recovered surface has to flip.*

ure 4a). A closely related type of ambiguity, particularly relevant to stroke-based editing, is slope direction ambiguity (Figure 4b). In our system, strokes modify an existing surface, so we can resolve the ambiguity by choosing the slope which changes the surface the least.

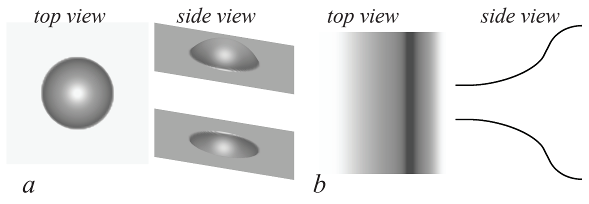

Figure 4: *(a) Convex-concave ambiguity; (b) slope ambiguity*

4 Overview of the System

The input to our system is an arbitrary manifold mesh, possibly with boundary. The type of interaction we describe is most suitable for relatively smooth meshes; otherwise, shading is not likely to provide easily understandable information concerning surface shape. The user arbitrarily positions the mesh, chooses its material properties, and positions the light source. Only one light source can be used.

Most of the interaction is done using several types of brushes: a shading modification brush, highlight motion brush, and silhouette brush. These brushes are demonstrated in Figure 5. The user can choose a brush's width, opacity, smoothness, and other attributes. An applied stroke can be modified after application (i.e., its attributes can be changed).

The shading modification brush is used to change shading, primarily away from highlights. A darkening shading stroke which crosses a point highlight is terminated, but a darkening stroke can cross a highlight line or highlight area. The user has explicit control over the surface tilt ambiguity: by default, the direction is chosen to minimize the change in slope under the stroke, but can be flipped by pressing a button after the stroke is drawn. Silhouette strokes add silhouette lines to the surface. The width of the stroke in this case controls the size of the created fold. The highlight motion tool is used to reposition highlights (including pushing them to merge with other highlights). A combination of highlight repositioning and shading modification can be used to change a highlight's shape.

Last but not least, it is possible to fix a region of interest (ROI) on the surface to ensure that no changes are made to the surface outside this area.

Additional views are provided so that the user can observe the effects of modifications from different points of view.

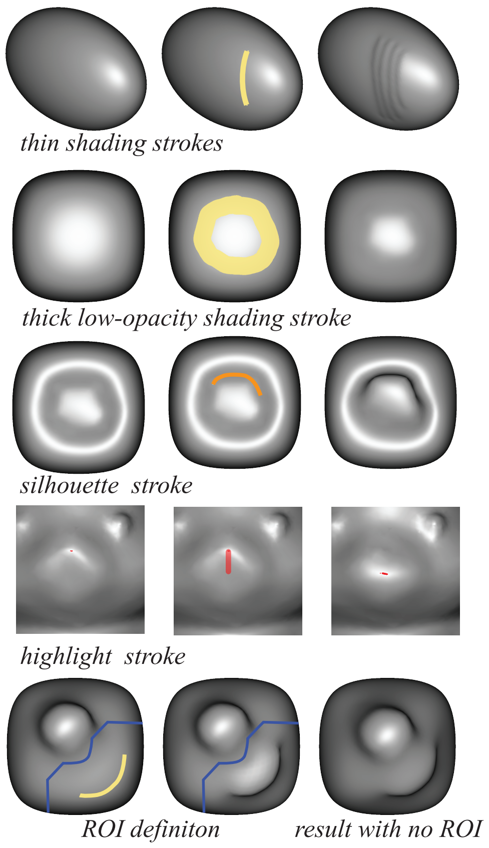

Figure 5: *Examples of different tools applied to simple surfaces.*

5 Problem Formulation

In this section, we provide a mathematical definition of our stroke parameters (Section 5.1) and demonstrate how the problem of converting shading changes specified by strokes into shape changes can be formulated as a quadratic surface optimization problem with linear constraints (Section 5.2). The problem formulation is independent of the choice of discretization, and can be applied, for example, to spline or subdivision surfaces. We consider its discretization in Section 6.1.

The goal of our technique is to modify a given surface M, with or without boundary. We assume that M is a smooth (C^1) surface, such that the normals are defined everywhere, there are no self-intersections, and it is approximated by a mesh.

A single light source is located at a point $\mathbf{p}_l$, or at infinity in direction $\mathbf{l}$. For simplicity of discussion, we use a directional light source. Similarly, the camera is located at a point $\mathbf{p}_v$, or at infinity, and the view direction (the normal to the image plane) is $\mathbf{v}$. The projection to the image plane is denoted P; e.g., for an orthographic projection to a plane passing through zero, $P = I - \mathbf{v}\mathbf{v}^T$.

We assume the material has uniform properties specified by a reflectance function $\rho(\mathbf{n})$, where $\mathbf{n}$ are surface normals. We use simple reflection functions with Lambertian and glossy reflection terms of the form $(1-\beta)\langle\mathbf{n},\mathbf{l}\rangle + \beta\langle\mathbf{n},\mathbf{h}\rangle^p$, where β is the degree of glossiness, p is the Phong exponent, and $\mathbf{h} = (\mathbf{v}+\mathbf{l})/\|\mathbf{v}+\mathbf{l}\|$ is the halfway vector. We emphasize that we use this specific type of shading as a user interface widget. We do *not* attempt to make it possible to do shading-based modeling in realistic lighting conditions. Multiple light sources, complex reflection models, and variable surface properties make it much more difficult for the user to visualize desired changes in appearance.

In the context of this work, the *image* of the surface M is a function $I(\mathbf{q}) = \rho(\mathbf{n}(\mathbf{p}))$, where $\mathbf{p}$ is the closest point of M projecting to $\mathbf{q}$. (I is defined in the area of the image plane corresponding to $P(M)$, the projection of M.)

The user modifies the image function, $I(\mathbf{q})$, to obtain a new function, $\tilde{I}$. The modification is confined to the smooth areas of I (i.e., we fix the silhouettes). Our goal is to construct a new surface, $\tilde{M}$, whose image matches $\tilde{I}$ as closely as possible, subject to a number of restrictions. Most modifications are based on *strokes*, defined by curves C in the image plane, corresponding to curves $P^{-1}(C)$ on the surface.

5.1 Stroke types

Strokes determine how the target intensity field is specified. While the attributes of strokes are similar to those found in two-dimensional drawing programs, a few aspects of these strokes need to be adapted to our application.

Strokes have an intuitive informal geometric interpretation: they correspond to variable rotations of the tangent planes about the centerline of the stroke. The attributes of the stroke determine how the planes are rotated and how the rotation propagates from the stroke. The surface change required for *simple* changes in appearance, such as uniform darkening, may be quite complex, although qualitatively we ensure that the behavior is intuitive; in particular, the rotation changes continuously along the stroke. The primary attributes of a stroke are the base curve C, width w, and value I_v.

Strokes have *softness* f, and *opacity* α, and can be applied in one of two modes, *multiply* or *replace*, which determine the interpretation of I_v. Stroke parameters, with the exception of softness and width, are used to determine target intensities for the surface. Suppose the original intensity of the surface at a point under the stroke is I_0. Then the target intensity I_{trg} is determined according to the following formulas.

- replace mode:

  $$I_{trg} = \alpha I_v + (1-\alpha)I_0$$

  In this mode, I_v is interpreted as intensity and ranges from $0\ldots1$, with blending between the old and new intensity determined by opacity. In geometric terms, for Lambertian surfaces at 100% opacity, this corresponds to twisting the surface about the stroke in a way that forces the surface normals to have a given angle with the light direction.

- multiply mode:

  $$I_{trg} = \alpha min(1, I_v I_0) + (1-\alpha)I_0$$

  In this mode, the stroke darkens or brightens the underlying surface by a percentage determined by I_v, which ranges from $0\ldots1.5$. As a surface is painted over with a darkening stroke multiple times, it gets increasingly darker, asymptotically approaching zero intensity. Geometrically, this also twists the surface at each point, by an amount proportional to the angle of the normal with the light direction (again, for Lambertian surfaces).

Softness determines the sharpness of the transition between the stroke and the rest of the surface. The mechanism used for this is described in Section 5.3. A softness of zero corresponds to a sharp transition (normal discontinuity), and the maximal softness is one. The effects of changing different stroke attributes are shown in Figure 6.

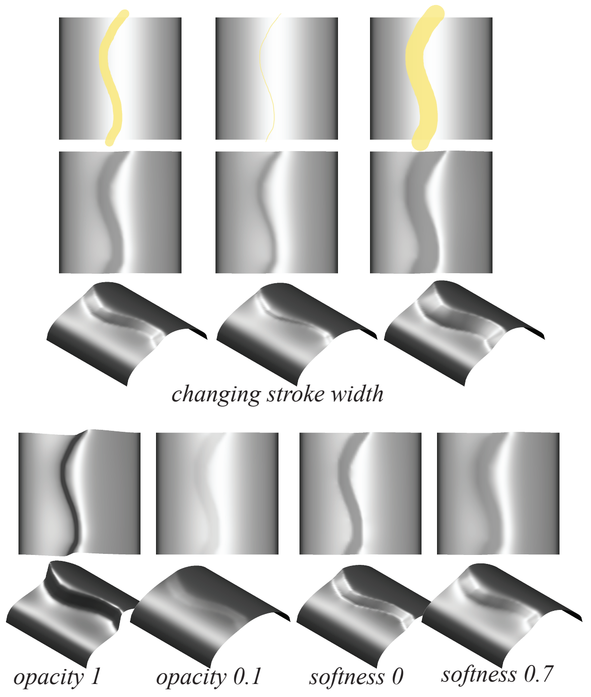

Figure 6: *Changing stroke attributes: width, smoothness, and opacity.*

5.2 Constrained Surface Optimization

In this section, we describe our surface optimization functional and basic stroke constraints.

In our system, the user modifies the surface one stroke at a time, generally changing shading in a small area of the surface. As a consequence, the goal of finding a modified surface which agrees with the updated image can be separated into two parts: recovering a (relatively small) part that matches the modified surface under the stroke, and keeping the rest of the surface as close as possible to the original.

The central idea of our approach is to treat the stroke as a special type of constraint and use a weighted detail-preserving functional to minimize the changes in the rest of the surface.

We motivate our choice for the detail-preserving functional first and then explain how stroke constraints are defined.

Preserving appearance outside strokes. There is a fundamental conflict between shading-based modifications and preserving the *surface itself* exactly unchanged outside the stroke. This can be clearly seen in Figure 6: if an area on a plane is darkened, the parts of the plane on two sides of the stroke can remain flat, but they need to be displaced. Similar to observations in [Sorkine et al. 2003] and [Alliez et al. 2003], we note that high-frequency error matters more for appearance preservation; i.e., low-frequency error is preferred.

A natural choice for functionals of this type are those based on surface gradients [Yu et al. 2004] and Laplacians [Sorkine et al. 2004], used in a variety of contexts. The vector Laplacian is the normal scaled by the mean curvature: $\Delta_M \mathbf{x} = H\mathbf{n}$, where Δ_M is the Laplace-Beltrami operator on the original surface M, and H is the mean curvature. If the surface changes remain close to isometric, the Laplacian operator does not change [Wardetzky et al. 2007], and the Laplacian difference

$$\int_M (\Delta_M \mathbf{x} - \Delta_M \mathbf{x}_0)^2 dA = \int_M (H\mathbf{n} - H_0\mathbf{n}_0)^2 dA$$

is a change in the normal orientation scaled by mean curvature. While we do not restrict our deformations to be isometric, if the triangle distortion stays small, one can view the Laplacian difference energy as a weighted normal change penalty. This closely matches what is needed for appearance preservation. A first-order Poisson approach uses the functional

$$\int_M (\nabla x - \nabla x_0)^2 dA.$$

In this case, the relationship to normal preservation can be seen from the Euler-Lagrange equation:

$$\Delta_M \mathbf{x} = \Delta_M \mathbf{x}_0.$$

The difference between the two functionals is primarily in supported boundary conditions. While the gradient-based functional allows only for positional or normal constraints on the boundary of the region of interest, the Laplacian-based functional makes it possible to join the modified patch with the surface smoothly, which is more suitable for our problem. We use a weighted Laplacian-based functional

$$\int_M g(\mathbf{x}_0)(\Delta_M \mathbf{x} - \Delta_M \mathbf{x}_0)^2 dA. \qquad (1)$$

(The weighting function g is used to implement stroke smoothness, discussed in Section 5.3.)

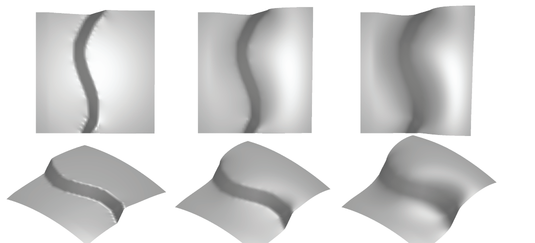

Figure 7: *Laplacian vs. Poisson penalty outside of the stroke.*

We note that detail-preserving differential coordinate techniques are often complemented by various types of rotations applied to the differential coordinates (Laplacians and gradients) (e.g., [Lipman et al. 2005; Botsch and Sorkine 2008]). In the context of large deformations, it is desirable to preserve the orientation of details with respect to some coarse reference surface, as opposed to preserving details' world-space orientations. In our context, however, these rotations are clearly undesirable; for the surface to retain unchanged appearance, we *do* want the normals to retain their spatial direction with respect to the viewing direction and the light source.

Strokes as constraints. We treat strokes as normal and positional constraints on our weighted, detail-preserving functional.

The simplest case is *hairline strokes* of zero width. We treat this type of stroke as the combination of normal constraints along the stroke curve $P^{-1}(C)$ and positional constraints on the projection of $P^{-1}(C)$ to the image plane. The target intensities for the stroke are computed as explained in Section 5.1. In principle, in the case of a hairline stroke, the normal directions can be defined arbitrarily (integrability constraints need to be satisfied only if the normals are specified on an area). We find that the most predictable behavior is obtained if the normal is rotated in the plane perpendicular to the tangent to the stroke. If the initial normal is $\mathbf{n}_0$, and the stroke unit tangent is $\mathbf{t}$, the choice of normals is restricted to

$$\mathbf{n}(\alpha) = \cos(\alpha)\mathbf{n}_0 + \sin(\alpha)\mathbf{t} \times \mathbf{n}_0. \qquad (2)$$

For each point, we find the minimal angle of rotation α such that $\rho(\mathbf{n}(\alpha)) = I_{trg}$, to obtain the target normal $\mathbf{n}_{trg}$. Figure 12 portrays these terms along with their discrete counterparts (discussed in Section 6.1). (We note that this works even if the reflectance function has multiple maxima, although we believe that simpler lighting choices are best for modeling.) If the surface is smooth, this angle changes continuously along a stroke, excluding the case of strokes passing through a highlight, as discussed in Section 5.3.

We discuss the discretization of stroke constraints in Section 6.1.

Choice of variables. Most work on shading-based surface recovery operates on *height fields*; that is, the surface is allowed to move only in the view direction. Since we regard the problem as one of maintaining the three-dimensional surface shape away from the stroke, there is no particular reason to restrict motion of the surface to this single dimension. We find that distributing the error to all three dimensions, rather than restricting it to the view direction, is preferable. As shown in Figure 13, restricting deformations to height fields leads to extreme distortion of the mesh even in the simple situation of a single darkening stroke.

5.3 Realization of Stroke Attributes

In this section, we present the implementation of stroke smoothness and thick strokes, silhouette strokes, the interaction of strokes with highlights, and highlight motion strokes.

Stroke smoothness and thick strokes. Using hairline strokes to define constraints for optimizing a detail-preserving functional would not provide much control over the width of the stroke and the sharpness of the transition between modified and unmodified parts of the surface. One of the crucial elements of our construction is adding a weighting function to the Laplacian functional, making it possible to control stroke width and softness. The base curve of a thick stroke places the same constraints as a hairline stroke; they differ only in this weighting function.

The idea of the weighting function is to "weaken" the link between the stroke area and the rest of the surface, allowing the surface to bend more flexibly at the stroke boundary or even form a sharp feature for hard strokes. At the same time, part of the surface remains tightly linked by the detail-preserving energy and rotates with the normals along the stroke base curve.

Substituting for g in Equation 1, we have our complete surface-preserving functional,

$$\int_M h(d(P(\mathbf{x}_0),C))(\Delta_M \mathbf{x} - \Delta_M \mathbf{x}_0)^2 dA, \qquad (3)$$

where $d(P(\mathbf{x}_0),C)$ is the image plane distance from the projection of a point on the surface to the stroke, and $h(r)$ is a weighting function defined by the width w and softness f of the stroke as follows.

$$h(r) = \begin{cases} 1, \text{ for } -w/2+d < r < w/2-d \text{ and } |r| > |w/2+d| \\ 1 - cB((r+w/2)/d), \text{ for } -w/2-d \le r \le -w/2+d \\ 1 - cB((r-w/2)/d), \text{ for } w/2-d \le r \le w/2+d \end{cases}$$

where $B(t)$ is a quadratic spline function satisfying $B(0) = 1$,

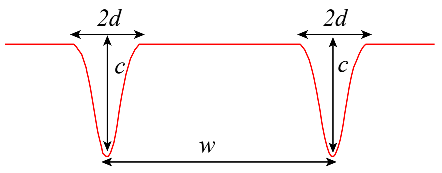

Figure 8: *Weighting function profile across the stroke.*

$B(-1) = B(1) = B'(-1) = B'(1) = 0$. As can be seen in Figure 8, the weighting function is constant, excluding two areas at the stroke boundaries, of width d, and depth c. Generally, d has a very subtle effect on appearance; for meshes, it is selected to be sufficiently wide so that there is a closed chain of mesh edges entirely within d of the stroke boundary; this ensures that when the functional is discretized, the non-unit weighting is applied to an area of the mesh surrounding the stroke. Once d is chosen, c is computed from the condition $(1-c)/d = f$ (very small values of d require c close to 1 for hard strokes).

The motivation for this choice is that the integral under the bump remains constant. Analytic computations for one-dimensional stroke cross-sections indicate, and experiments confirm, that the shape of the stroke is insensitive to the choice of d, as long as $(1-c)/d$ remains constant, within a broad range of d.

We found that this relatively simple approach works remarkably well. While it does *not* ensure uniform darkening if the stroke is applied to a surface area with lots of details, it effectively ensures average darkening while preserving the geometry of small-scale detail, as shown in Figure 9. An attempt to darken all points uniformly results in detail "smudging." While this is desirable in some cases, we believe this may be best controlled by a separate attribute.

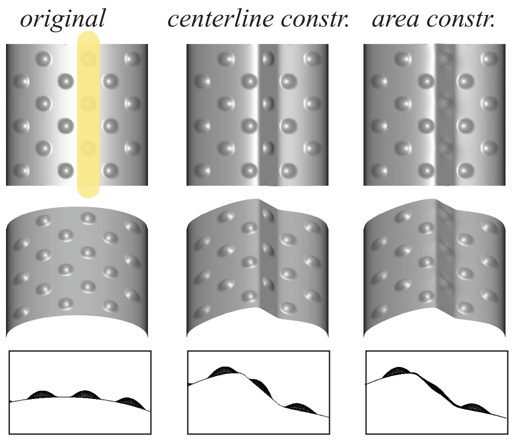

Figure 9: *Detail preservation: a thick stroke with base curve constraints only, and the same stroke with constraints on normals imposed over the whole area in the least-squares sense.*

Silhouette strokes. Silhouette strokes are implemented similarly to shading strokes, but they have no opacity α or value I_v.

Specifically, silhouette strokes are also based on constraining normal orientations along the base curve of the stroke, in this case finding $\mathbf{n}(\alpha)$ perpendicular to the view direction (see Equation 2). Silhouette strokes do have smoothness, implemented exactly in the same way as for shading strokes. Combined with shape-preservation optimization, this typically leads to some faces becoming back-facing. (Silhouette strokes would not be possible with a height field representation.)

Interaction with highlights. If a stroke passes a highlight, i.e., a local maximum of $\rho(\mathbf{n})$, then the normal rotation may experience a jump at this point: the preferred direction of rotation may change (but there are highlights for which it does not). This leads to nonintuitive behavior which we prefer to avoid. Thus, when a discontinuity of this type is detected, the stroke is terminated at the highlight (Figure 10).

Highlight motion strokes. A highlight motion stroke is designed to move a highlight. In highlight motion mode, the highlights are detected by thresholding the intensity at vertices, and the user can draw a stroke starting at a highlight.

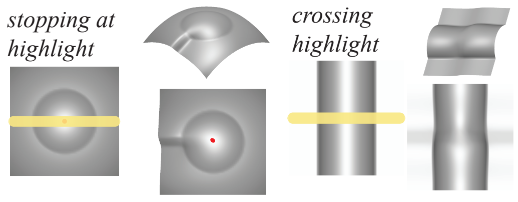

Figure 10: *Types of stroke behavior at highlights.*

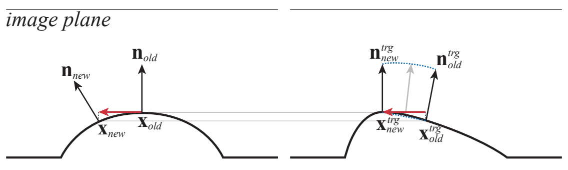

Figure 11: *The effect of a highlight motion stroke, viewed from the side. The highlight motion stroke is depicted as a red arrow parallel to the image plane. The initial configuration is shown at left, and the result is shown at right.*

Overall, the stroke operates the same way shading strokes work; however, the target normals along the stroke are defined differently. Let $\mathbf{x}_{old}$ and $\mathbf{x}_{new}$ be the old and new positions of the highlight on the surface. The constraints for a highlight motion stroke are defined as follows, and illustrated in Figure 11.

- The target position $\mathbf{x}^{trg}_{new}$ of $\mathbf{x}_{new}$ has the same position in the image plane as $\mathbf{x}_{new}$ and displacement from the image plane equal to that of $\mathbf{x}_{old}$. The normal at $\mathbf{x}_{new}$ is constrained to be the negative view direction.
- The target position $\mathbf{x}^{trg}_{old}$ of $\mathbf{x}_{old}$ has the same position in the image plane as $\mathbf{x}_{old}$ and displacement from the image plane equal to that of $\mathbf{x}_{new}$. The target normal at $\mathbf{x}_{old}$ is constrained to be perpendicular to $\mathbf{x}^{trg}_{new} - \mathbf{x}^{trg}_{old}$.
- The rotations of normals between these two points interpolate the rotations at endpoints.

6 Discrete problem

In this section, we describe the discretization we use for our surface optimization functional and for stroke constraints.

The discretization of the weighted Laplacian functional (Equation 3) is standard, using the well-known cotangent formula for Laplacians. (At the boundary of the mesh, the edges get only a single cotangent in their weight [Sorkine 2008].) We refer to the work on Laplacian editing for details.

6.1 Discretization of Stroke Constraints

The constraints we impose for strokes affect vertices of edges intersecting the stroke. The stroke curve (if necessary, smoothed by subsampling and spline interpolation) is projected to the mesh and intersected with mesh edges. The target normals along the stroke are defined using the tangent to the stroke, the original normal, and the target intensity. Furthermore, the image plane position of the stroke, C, is constrained to be fixed.

We discretize these constraints as follows. An *edge* normal, $\mathbf{n}_e$, is computed for every edge, e, intersected by the projected stroke. A target normal, $\mathbf{n}^{trg}_e$, is computed for the edge, according to the stroke type and its attributes. Figure 12 portrays these terms along with their continuous counterparts.

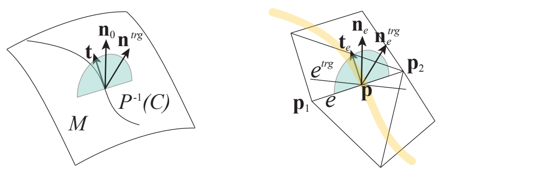

Figure 12: *Notation for stroke constraints. A continuous (left) and discrete (right) surface patch.*

The simplest linear constraint would be to require the new position of the tangent, $\mathbf{p}_2 - \mathbf{p}_1$, where $\mathbf{p}_1$ and $\mathbf{p}_2$ are the endpoints of the edge, to be orthogonal to $\mathbf{n}_e^{trg}$; i.e., $\langle \mathbf{p}_2 - \mathbf{p}_1, \mathbf{n}_e^{trg} \rangle = 0$. However, this constraint is also satisfied if the edge has zero length, and we observe that the triangles often do degenerate. Instead, we obtain a target tangent vector e^{trg} by applying the minimal rotation mapping $\mathbf{n}_e$ to $\mathbf{n}_e^{trg}$ to the edge, resulting in the constraint $\mathbf{p}_2 - \mathbf{p}_1 = e^{trg}$. The advantages of this approach are shown in Figure 13. Both types of constraints are linear in vertex positions. In addition to constraining the edge direction, we also constrain the projected position of the point $P(\mathbf{p}) = P(a\mathbf{p}_1 + (1-a)\mathbf{p}_2)$ where the stroke intersected the edge. This insures that the position of the projection of the stroke to the image plane does not change.

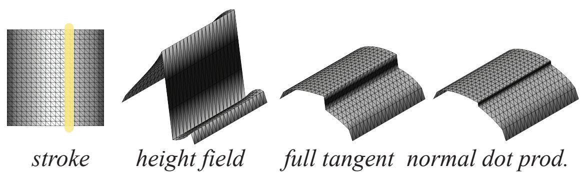

Figure 13: *Dot product with target normal vs. full tangent constraints.*

Adaptive refinement. One of the problems with image-space strokes is that it may not be possible to reproduce the stroke on the surface exactly—especially strokes with low softness or thin strokes—if the mesh is coarse. We use adaptive $\sqrt{3}$-subdivision to refine the mesh in the area of the stroke (Figure 14). The criterion for refinement is to reduce the image-space length of the edges overlapped by the stroke boundary to a user-specified threshold (in pixels). Note that if the view of the model is zoomed, and a stroke is reapplied, additional refinement will occur, as the criterion is defined in image space.

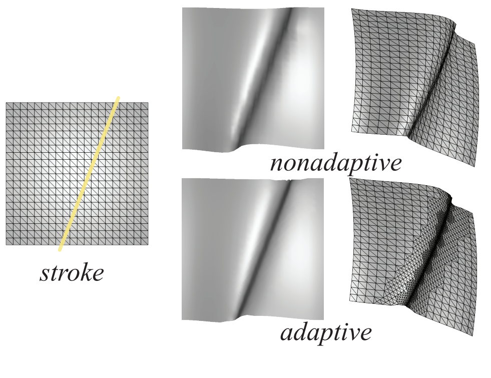

Figure 14: *Adaptive refinement. The unrefined mesh has 441 vertices, which increases to 1302 vertices in the refined mesh.*

6.2 System Assembly

Summarizing the above, we describe the construction of the linear system of equations that we solve. We minimize the following energy:

$$E = \sum_{i=0}^{N} g(\mathbf{x}_0^i) A_0^i \|\Delta \mathbf{x}^i - \Delta \mathbf{x}_0^i\|^2 + w_{lsq} E_{\text{constraint}}$$

where g is defined (by substitution) in Equation 3, $\mathbf{x}^i$ is the position of vertex i, $\mathbf{x}_0^i$ is the undeformed position of vertex i, A_0^i is the area corresponding to vertex i in the undeformed mesh, and N is the number of vertices in the ROI. $E_{\text{constraint}} = E_{\text{tangent}} + E_{\text{midpoint}}$, where

$$E_{\text{tangent}} = \sum_{(i,j)=e \in C} \|(\mathbf{x}^i - \mathbf{x}^j) - e^{trg}\|^2$$

and

$$E_{\text{midpoint}} = \sum_{(i,j)=e \in C} \|P(a_e \mathbf{x}^i + (1-a_e)\mathbf{x}^j) - P(\mathbf{p}_e)\|^2$$

and C is the set of constrained edges. e^{trg} and $\mathbf{p}$ are defined in Section 6.1. We use $w_{lsq} = 1e7$ for all examples.

Since we fix silhouettes, all vertices belonging to back-facing faces are considered to be outside the ROI. (An exception is made for small back-facing components—those whose area is less than 10% of the front-facing ROI area—to prevent small cavities such as nostrils from becoming fixed.) In addition, an automatic ROI determination is available, which sets the ROI to the part of the surface whose image plane projection lies within a user-specified distance of the stroke. This reduces the system size for large meshes.

If the positions are transformed so that z is the view direction, then the system is decoupled.

7 Results

We demonstrate how our system can be used to modify a variety of models. The two main scenarios are modifying existing detailed models or refining a simple existing model. Examples can be seen in Figures 15-17. Shading strokes are shown in yellow and silhouette strokes are shown in orange.

In Figure 15 (left), an eye is added to a simple horse model using many thin, smooth shading strokes. The initially coarse mesh is adaptively refined to allow for the addition of fine detail. In this example, all strokes are drawn from a single point of view. Muscles and sharp features are added to a simple male model in Figure 15 (right). Shading strokes are used at varying scales to create medium as well as fine geometry changes. Figure 16 (left) depicts eyes, nostrils, and ear cavities added to a model created in the FiberMesh system. The eyes are added with several thin, sharp shading strokes, creating ridges. Such features are difficult to create using displacement editing tools. The nostrils and ear cavities are each created with a single smooth silhouette stroke. In Figure 16 (right), wear and tear is added to a couch. The creases are added using with silhouette strokes. The rest of the features are added with shading strokes. In Figure 17 (left), a thick shading stroke is used to give the mannequin head puffy cheeks. A silhouette stroke creates a more pronounced lip. In Figure 17 (right), a sharp silhouette stroke is used to deepen a brow line on an elephant, and smooth shading strokes are used to create a bulging leg muscle.

All interaction sessions were recorded on a MacBook Pro with a 2 GHz Intel Core Duo processor. A single core is used for computations. A graphics tablet is used as the input device.

The time for a stroke to be applied greatly depends on the stroke size, ROI setting, mesh size, and degree of adaptive refinement, ranging from instantaneous to seconds. The total optimization time is dominated by building and solving the system of equations. In our implementation, we use the sparse direct LU solver in the PETSc package. Any direct LU solver will suffice, and faster ones are available.

Performance is summarized in the following table. Each row represents a stroke applied to a model. Models indicated appear in Figures 15-17. The ROI column indicates the number of vertices in the ROI of the stroke; the system size is three times this number, squared. The constraints column indicates the number of constraints equations induced by the stroke. The "total" column summarizes the total computation for the stroke, in seconds. The "building" and "solving" columns indicate the percentage of the total computation spent building and solving the system.

model	ROI	constraints	total (s)	building	solving
fig. 16, left	931	162	.25	37%	39%
horse	3859	237	1.38	44%	51%
elephant	9950	267	3.71	44%	52%

8 Conclusions and Future Work

We have described a general framework for controllable shading-based surface editing, providing a direct and intuitive interface for a broad range of surface modifications. Our primary tool, shading strokes, have both an intuitive meaning when viewed as two-dimensional strokes in the image plane, and a predictable geometric behavior. A set of attributes makes them sufficiently flexible to achieve fine control over surface appearance.

We plan to extend this work in a number of ways. The control of highlights our tool provides is still quite limited We found detail-preserving strokes most useful. However, in some cases it is desirable to eliminate details in a controllable way, so adding a "blur" attribute to the stroke would be useful. Our system naturally complements a number of other sketch-based approaches, so we will explore integration with other systems.

Acknowledgements. We would like to thank our NYU Computer Science colleagues for their help; the anonymous reviewers for their suggestions; and Zach Shukan for his artwork.

References

ALLIEZ, P., COHEN-STEINER, D., DEVILLERS, O., LÉVY, B., AND DESBRUN, M. 2003. Anisotropic polygonal remeshing. *ACM Transactions on Graphics (TOG) 22*, 3, 485–493.

BOTSCH, M., AND KOBBELT, L. 2004. An intuitive framework for real-time freeform modeling. *ACM Transactions on Graphics (TOG) 23*, 3, 630–634.

BOTSCH, M., AND SORKINE, O. 2008. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics 14*, 1, 213–230.

BOURGUIGNON, D., CHAINE, R., CANI, M.-P., AND DRETTAKIS, G. 2004. Relief: A modeling by drawing tool. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBM)*, 151–160.

CELNIKER, G., AND GOSSARD, D. 1991. Deformable curve and surface finite-elements for free-form shape design. *Computer Graphics (SIGGRAPH Conference Proceedings) 25*, 4, 257–266.

CHEUTET, V., CATALANO, C. E., PERNOT, J. P., FALCIDIENO, B., GIANNINI, F., AND LEON, C. 2004. 3D sketching with fully free form deformation features (δ-f4) for aesthetic design. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBM)*, 9–18.

DECARLO, D., FINKELSTEIN, A., RUSINKIEWICZ, S., AND SANTELLA, A. 2003. Suggestive contours for conveying shape. *ACM Transactions on Graphics (TOG) 22*, 3, 848–855.

IGARASHI, T., MATSUOKA, S., AND TANAKA, H. 1999. Teddy: a sketching interface for 3D freeform design. In *Computer Graphics (SIGGRAPH Conference Proceedings)*, 409–416.

KARA, L. B., D'ERAMO, C. M., AND SHIMADA, K. 2006. Pen-based styling design of 3D geometry using concept sketches and template models. In *Proceedings of the ACM Symposium on Solid and Physical Modeling (SPM)*, 149–160.

KARPENKO, O. A., AND HUGHES, J. F. 2006. SmoothSketch: 3D free-form shapes from complex sketches. *ACM Transactions on Graphics (TOG) 25*, 3, 589–598.

KERAUTRET, B., GRANIER, X., AND BRAQUELAIRE, A. 2005. Intuitive shape modeling by shading design. In *Smart Graphics: 5th International Symposium.*

LAWRENCE, J., AND FUNKHOUSER, T. 2004. A painting interface for interactive surface deformations. *Graphical Models 66*, 6, 418–438.

LIPMAN, Y., SORKINE, O., LEVIN, D., AND COHEN-OR, D. 2005. Linear rotation-invariant coordinates for meshes. *ACM Transactions on Graphics (TOG) 24*, 3, 479–487.

MORETON, H. P., AND SÉQUIN, C. H. 1992. Functional optimization for fair surface design. In *Computer Graphics (SIGGRAPH Conference Proceedings)*, 167–176.

NEALEN, A., SORKINE, O., ALEXA, M., AND COHEN-OR, D. 2005. A sketch-based interface for detail-preserving mesh editing. *ACM Transactions on Graphics (TOG) 24*, 3, 1142–1147.

NEALEN, A., IGARASHI, T., SORKINE, O., AND ALEXA, M. 2007. FiberMesh: designing freeform surfaces with 3D curves. *ACM Transactions on Graphics (TOG) 26*, 3, 41.

NG, H.-S., WU, T.-P., AND TANG, C.-K. 2007. Surface-from-gradients with incomplete data for single view modeling. In *Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV)*, 1–8.

PRADOS, E. 2004. *Application of the theory of the viscosity solutions to the Shape From Shading problem*. PhD thesis, University of Nice-Sophia Antipolis.

RUSHMEIER, H., GOMES, J., BALMELLI, L., BERNARDINI, F., AND TAUBIN, G. 2003. Image-based object editing. *Proceedings of the Fourth International Conference on 3D Digital Imaging and Modeling (3DIM)*, 20–28.

SCHAEFER, S., WARREN, J. D., AND ZORIN, D. 2004. Lofting curve networks using subdivision surfaces. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, 103–114.

SINGH, K., AND FIUME, E. 1998. Wires: a geometric deformation technique. In *Computer Graphics (SIGGRAPH Conference Proceedings)*, 405–414.

SORKINE, O., COHEN-OR, D., AND TOLEDO, S. 2003. High-pass quantization for mesh encoding. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, 42–51.

SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, 175–184.

SORKINE, O. 2008. Personal communication.

VAN OVERVELD, C. W. A. M. 1996. Painting gradients: free-

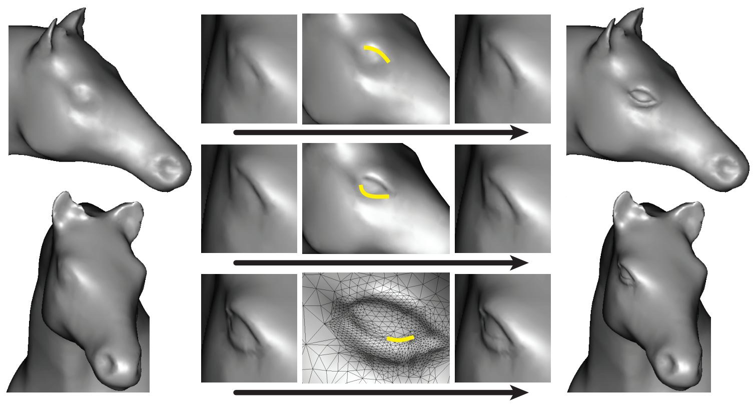
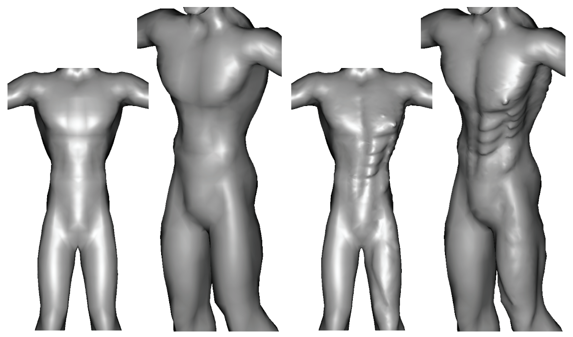
Figure 15: *Left: adding an eye to a horse model with shading strokes; the mesh is adaptively refined. The mesh begins with 19851 vertices and is refined to 21116 vertices. Right: refining a simple male model. The mesh begins with 5914 vertices and ends with 9151 vertices.*

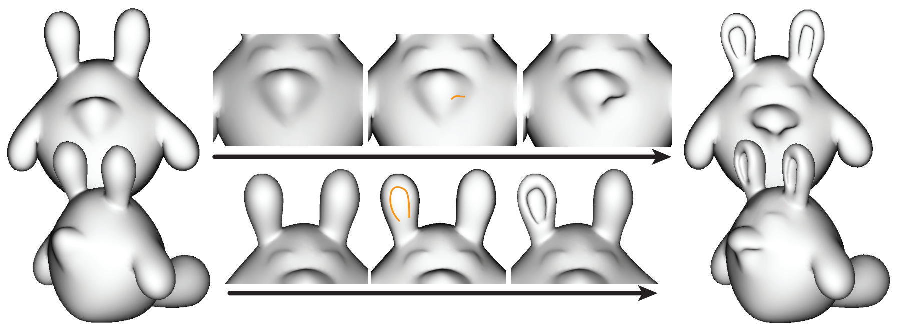
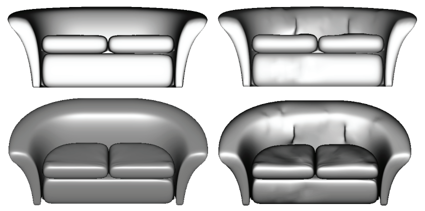
Figure 16: *Left: refining a model created in the FiberMesh system. The eyes are added using shading strokes; the nostrils and ears are added with silhouette strokes. The mesh begins with 3498 vertices and is refined to 5330 vertices. Right: adding features to a couch. The couch contains 31013 vertices.*

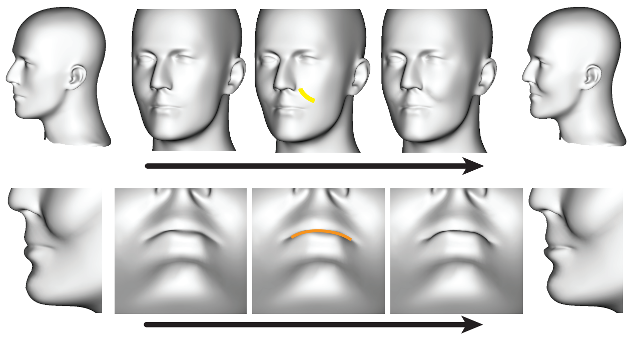
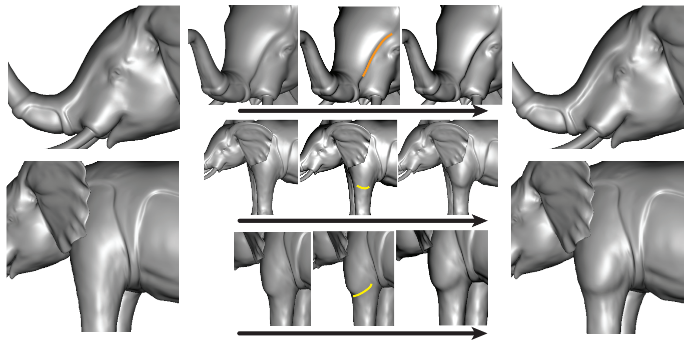
Figure 17: *Left: a shading stroke and a silhouette stroke applied to the mannequin head. The mesh begins with 10883 vertices and is refined to 12515 vertices. Right: deepening a crease and adding a muscle on an elephant model. The mesh contains 45682 vertices.*

form surface design using shading patterns. In *Proceedings of the conference on Graphics Interface (GI)*, 151–158.

WARDETZKY, M., BERGOU, M., HARMON, D., ZORIN, D., AND GRINSPUN, E. 2007. Discrete quadratic curvature energies. *Computer Aided Geometric Design 24*, 8-9, 499–518.

WELCH, W., AND WITKIN, A. 1992. Variational surface modeling. In *Computer Graphics (SIGGRAPH Conference Proceedings)*, 157–166.

WU, T.-P., TANG, C.-K., BROWN, M. S., AND SHUM, H.-Y. 2007. ShapePalettes: interactive normal transfer via sketching. *ACM Transactions on Graphics (TOG) 26*, 3, 44.

YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics (TOG) 23*, 3, 644–651.

ZENG, G., MATSUSHITA, Y., QUAN, L., AND SHUM, H.-Y. 2005. Interactive shape from shading. *CVPR 1*, 343–350.

ZHANG, L., DUGAS-PHOCION, G., SAMSON, J.-S., AND SEITZ, S. M. 2001. Single view modeling of free-form scenes. *CVPR 01*, 990.

ZIMMERMANN, J., NEALEN, A., AND ALEXA, M. 2007. SilSketch: Automated sketch-based editing of surface meshes. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBM)*.