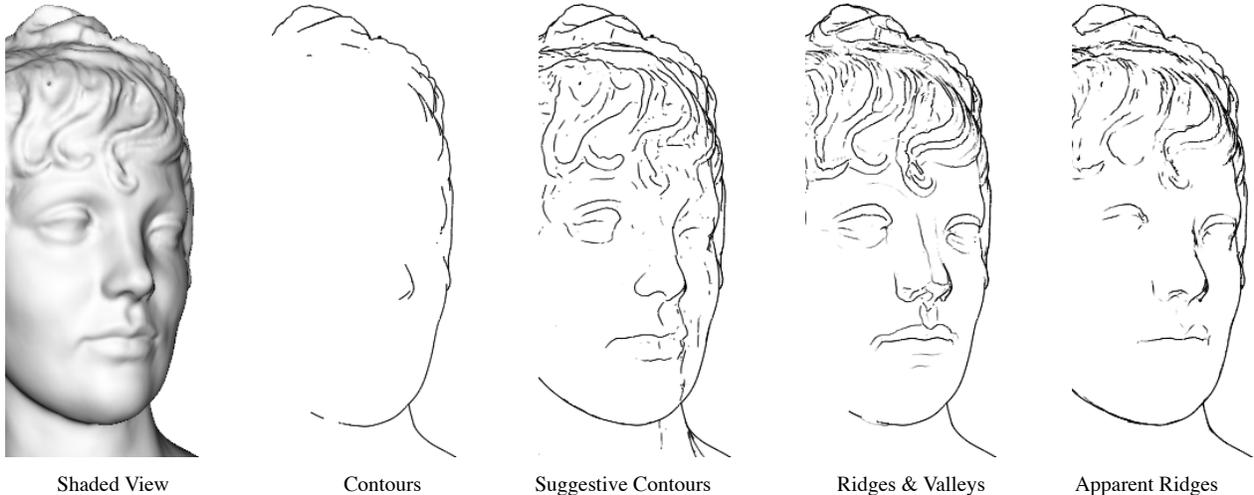


# Apparent Ridges for Line Drawing

Tilke Judd<sup>1</sup> Frédo Durand<sup>1</sup> Edward Adelson<sup>1,2</sup>

<sup>1</sup>MIT Computer Science and Artificial Intelligence Laboratory

<sup>2</sup>MIT Dept. of Brain and Cognitive Sciences



**Figure 1:** The Bust model rendered with several different feature lines. We introduce apparent ridges on the right. They correspond to the maxima of the normal variation with respect to the viewing plane. Note in particular the left side of the face (to the right) in the suggestive contour drawing and the nose drawn with ridges and valleys.

## Abstract

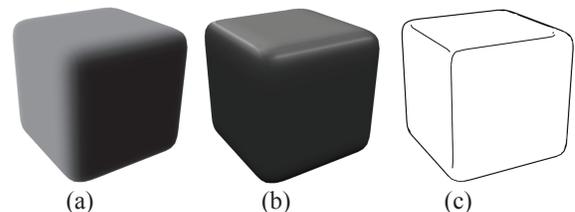
Three-dimensional shape can be drawn using a variety of feature lines, but none of the current definitions alone seem to capture all visually-relevant lines. We introduce a new definition of feature lines based on two perceptual observations. First, human perception is sensitive to the variation of shading, and since shape perception is little affected by lighting and reflectance modification, we should focus on normal variation. Second, view-dependent lines better convey smooth surfaces. From this we define *view-dependent curvature* as the variation of the surface normal with respect to a viewing screen plane, and *apparent ridges* as the loci of points that maximize a view-dependent curvature. We present a formal definition of apparent ridges and an algorithm to render line drawings of 3D meshes. We show that our apparent ridges encompass or enhance aspects of several other feature lines.

**Keywords:** Line Drawing, NPR, ridges, valleys, apparent ridges

## 1 Introduction

Suppose we wish to depict the shape of a rounded cube, which is viewed from a certain direction. One approach is photorealistic rendering. We choose a BRDF and a lighting distribution, and calculate the luminance value that each pixel would have if the real physical object were viewed under those conditions. Figures 2(a) and (b) show two examples, with different choices of BRDF and lighting. Depending on the choices, which are rather arbitrary, the shading and highlights will change. With photorealism there is no way to avoid these arbitrary choices.

An artist will often use a line drawing to convey an object's shape in a manner that is independent of BRDF and lighting. Figure 2(c) shows an example. This image is not photorealistic, since there are no conditions under which the cube's image would actually have these pixel values. Nonetheless the line drawing captures, in a compact and abstract manner, the essential visual properties that are shared by the photorealistic renderings. This image was generated by the method we will describe.



**Figure 2:** Depiction of a cube with traditional computer graphics shading and with line drawing (using our apparent ridges).

Line drawing is a popular topic in non-photorealistic rendering (NPR). The key question is: where do you put the lines? Various rules have been proposed. The bounding contour, or silhouette, is obviously important, as are occluding contours that occur within the object. Discontinuities of surface normal, such as those of a cube, produce a ridge line that should certainly be drawn. However, most natural objects contain bumps, dips, and undulations of varying geometry, and it is not clear how these features should be depicted with lines.

A number of approaches have been described. In image-based approaches, the object is rendered to form an image or geometric buffer, and then image processing methods such as edge detection yield an output drawing [Saito and Takahashi 1990; Decaudin 1996; Hertzmann 1999; Pearson and Robinson 1985; Lee et al. 2007]. The result is often visually pleasing, but the pixel-based representation suffers from low precision due to the loss of 3D scene information during rendering, and is unsuitable for additional processing. For example, lines can't be rendered with natural brush strokes unless curves are extracted from the edge map.

In object-based approaches, one finds curves that have special properties in terms of the differential geometry of the surface. Several researchers [Koenderink 1990; Ohtake et al. 2004; Interrante et al. 1995] have characterized various such curves, including parabolic lines, which divide hyperbolic from elliptic regions, as well as ridges and valleys, which occur at points of extremal curvature. (Note: in the field of topography, the terms “ridge” and “valley” have a different definition). These curves capture important object properties but do not make natural looking line drawings. They are locked to the object surface, and do not slide along it when the viewpoint changes. As a result, the objects portrayed by ridges and valleys tend to look overly sharp. Given the widely-accepted limitations of object-space ridges and valleys, we believe a major contribution of our work is defining the first successful ridge-like features for NPR.

It is also possible to define curves that have view dependence. The silhouette is one example: it is the loci of points that is only defined with respect to a viewing screen. An important type of view dependent curves, called suggestive contours, has recently been described by DeCarlo et al. [2003; 2004]. Suggestive contours are extensions of occluding contours. They are curves along which the radial curvature is zero. They are locations which are *almost* contours, and they correspond to true contours in nearby viewpoints.

Suggestive contours capture some, but not all, of the desirable curves that comprise a line drawing. For example, they ignore the dihedrals that define a cube, and will entirely miss the ridge-like features that define the noses in the drawings of Fig. 3. There is presently no single rule that will generate the full set of desirable lines, so the usual practice is to combine multiple rules.

Our new method gets a rich and comprehensive set of lines with a single rule. This rule is motivated by perceptual considerations. Human observers are highly sensitive to line-like and edge-like features, i.e., points of high luminance variation. A given object, when viewed from a given position, will tend to have such features in a limited set of locations, and these locations tend to be stable across different choices of BRDF and illumination. However, unlike the ridges and valleys of differential geometry, our features are view dependent, which we find is necessary in order to give the line drawing a natural appearance to a human viewer. The lines are drawn at the same places that “line detectors” and “edge detectors” are likely to fire when presented with a rendered image of the same object in the same pose.

Our rule can be succinctly stated as follows: *Draw a line when the surface normal is changing at a locally maximal rate with respect to image position.* We refer to these lines as “apparent ridges”. They depend on the curvature of the object surface as well as the foreshortening that occurs in projection to the viewing plane.



**Figure 3:** Lines drawings by Matisse and Juan Gris. Notice the depiction of the nose which exhibits ridge-like features. Left to right, Etched illustration for the poem “La chevelure d’une flamme” by Matisse <sup>1</sup> ©2007 Succession H. Matisse, Paris / Artist Rights Society (ARS), New York, and Portrait of Max Jacob by Juan Gris, ©2007 Artists Rights Society (ARS), New York / ADAGP, Paris.

**Motivation and overview** If a line drawing is to be independent of the rendering parameters of BRDF and illumination, then the only remaining local property is the surface normal. This led Felix Klein and after him Hilbert and Cohn-Vossen [1952] and Koenderink [1980] to hypothesize that parabolic lines were of special significance in artist’s drawings. However, attempts to leverage this idea have largely failed: parabolic lines do not yield compelling drawings. Likewise, line drawings based on ridges and valleys tend to look “blocky.” These drawings look particularly awkward when they are used in animations, as the lines appear to be stuck to the surface, and do not slide naturally over the surface. We believe that the use of strict view independence leads inevitably to this sort of problem.

An alternative approach is to draw lines at locations that are view dependent but are stable across the other rendering conditions. The idea that this might be possible is motivated by the observations of Fleming et al [2004], who showed that the local orientation structure of rendered objects was similar across multiple choices of BRDF and environment map, and therefore could be used as an invariant description the object’s appearance (as seen from a given viewpoint). They hypothesized that the human visual system extracts these invariant properties and uses them to infer the 3-D shape.

Rapid luminance changes occur at points where the angle of the surface normal is changing rapidly. Points in the image with maximal view-dependent curvature (i.e. our apparent ridges) will usually contain maximal luminance gradients. Similarly, Durand et al. [2005] showed that for realistic rendering, such points have highest local frequency content in an image. In computer vision, Yuille [1989] has derived equations in search of photometric invariants that are similar to our lines.

We first describe necessary background differential geometry and introduce the equations for view-dependent curvature and apparent ridges in Section 2. Given the mathematical definition of apparent ridges, we present algorithms to extract them on discrete meshes in Section 3. We show results of apparent ridge line drawings in Section 4. We demonstrate that apparent ridges are useful for conveying shape information and show how they are related to other feature lines.

<sup>1</sup>Reproduction, including downloading of Matisse works is prohibited by copyright laws and international conventions without the express written permission of Artists Rights Society (ARS), New York.

## 2 Apparent Ridges and Valleys

We develop the equations for view-dependent curvature and apparent ridges. First we review traditional concepts of differential geometry—curvature, ridges, valleys—that are defined from the first and second derivatives of the normal. Then, to define view-dependent curvature and apparent ridges, we calculate the same derivatives with one key change: we derive with respect to a screen plane instead of the surface.

### 2.1 Ridges and Valleys on Surfaces

Given a smooth closed surface,  $\mathbf{n}(m)$  is the outward facing unit normal to the surface at a point  $m$ . The tangent plane at  $m$  is perpendicular to the normal. Intuitively, curvature of a surface represents how a surface bends, or how the normal changes from point to point on a surface. We define the curvature operator  $S$  at point  $m$  as

$$S(\mathbf{r}) = D_{\mathbf{r}}\mathbf{n}$$

where  $D_{\mathbf{r}}\mathbf{n}$  is the directional derivative of the normal along vector  $\mathbf{r}$  in the tangent plane.  $S$ , which is known as the *Weingarten map* or the *shape operator*, is a linear map from the tangent plane at  $m$  to a tangent of the Gauss sphere parallel to the tangent plane. Given a choice of basis,  $S$  can be represented as a symmetric  $2 \times 2$  matrix.

For every point on the surface, the maximum and minimum principal curvatures  $k_1$  and  $k_2$  are the eigenvalues of  $S$  where  $|k_1| \geq |k_2|$ . The associated eigenvectors  $\mathbf{e}_1$  and  $\mathbf{e}_2$  correspond to the maximum and minimum principal curvature directions.

Ridges and valleys are the loci of points at which the principal curvature assumes an extremum in the principal direction. They are found where  $D_{\mathbf{e}_1}k_1 = 0$ , where ridges have  $k_1 > 0$  and valleys have  $k_1 < 0$ . In addition, we use the higher-order derivative to ensure we get the correct flavor of extrema: the higher order derivative must be negative (positive) to ensure a ridge (valley) is a maximum (minimum) of curvature.

### 2.2 View-dependent Curvature

By including viewing projection with curvature, we define view-dependent curvature, which will lead to apparent ridges.

**Projection** We consider a viewing screen plane  $V$  on which we draw a line drawing of an object  $M \subset \mathbb{R}^3$ . We define a parallel projection  $P$  that maps points  $m \in M$  onto points  $m' \in V$ . If  $m$  is not a contour point, there exists a neighborhood where  $P$  has an inverse,  $P^{-1}$  (essentially a ray-casting function). We can locally express the normal as a function of screen location  $\mathbf{n}'(m') = \mathbf{n}(P^{-1}(m'))$ .

At a point  $m$ , the Jacobian of  $P$ , which we call  $\bar{P}$ , takes tangent vectors at  $m$  to vectors in  $V$ . Given a choice of basis  $(\mathbf{r}_1, \mathbf{r}_2)$  for the tangent plane and  $(\mathbf{s}_1, \mathbf{s}_2)$  for the screen plane,  $\bar{P}$  can be expressed as a  $2 \times 2$  matrix

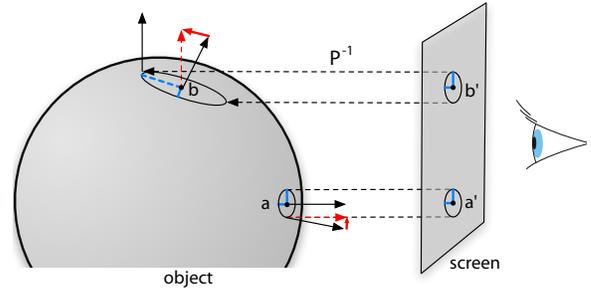
$$\bar{P} = \begin{pmatrix} \mathbf{r}_1 \cdot \mathbf{s}_1 & \mathbf{r}_2 \cdot \mathbf{s}_1 \\ \mathbf{r}_1 \cdot \mathbf{s}_2 & \mathbf{r}_2 \cdot \mathbf{s}_2 \end{pmatrix}.$$

Note that  $\bar{P}$  is different at every point on the object and is invertible everywhere except at the contour points.

**View-dependent Curvature** Intuitively, view-dependent curvature is how much the surface is seen to bend from the viewpoint. It takes into consideration both the curvature of the object and the foreshortening due to surface orientation. Formally, we define the view-dependent curvature operator  $Q$  at a point  $m'$  on the screen as

$$Q(\mathbf{s}) = D_{\mathbf{s}}\mathbf{n}'$$

where  $D_{\mathbf{s}}\mathbf{n}'$  is the directional derivative of the object normal along vector  $\mathbf{s}$  in the screen plane. The view-dependent curvature operator



**Figure 4:** The maximum view-dependent curvature at  $b'$  is much larger than at  $a'$  uniquely because of projection.

$Q$  at a point  $m'$  is a linear map from the screen plane into the tangent of the Gauss sphere. Note that this derivative is taken with respect to the screen space, but the change in normal is in the object space. We do *not* project the normal on to the screen space because we are motivated by the shading which is based on the object space normal. Projecting the normal into the screen space would ignore potential lighting components along the view direction.

We apply the chain rule to obtain  $Q$  in terms of surface curvature,

$$Q = S\bar{P}^{-1}. \quad (1)$$

where the basis of the tangent plane chosen for expressing  $S$  and  $\bar{P}$  are the same.  $Q(\mathbf{s})$  is a vector in the tangent plane that describes how the surface normal changes when one moves along vector  $\mathbf{s}$  on the screen.

We are interested in the extrema of the *magnitude* of the change, so we define the maximum view-dependent curvature as

$$q_1 = \max_{\|\mathbf{s}\|=1} \|Q(\mathbf{s})\|. \quad (2)$$

That is, the maximum view-dependent curvature  $q_1$  is the maximum singular value of  $Q$  (the square root of the maximum eigenvalue of  $Q^T Q$ ). It is achieved when  $\mathbf{s}$  is the maximum singular vector of  $Q$ . This singular vector, denoted by  $\mathbf{t}_1$ , is the maximum view-dependent principal direction. Note that unlike curvature, which is defined from eigenvalues of  $S$ , view-dependent curvature is defined from singular values of  $Q$  because the input and output spaces of  $Q$  are different.

Equivalently, we can define the maximum view-dependent curvature as

$$q_1 = \max_{\|P(\mathbf{r})\|=1} \|S(\mathbf{r})\|$$

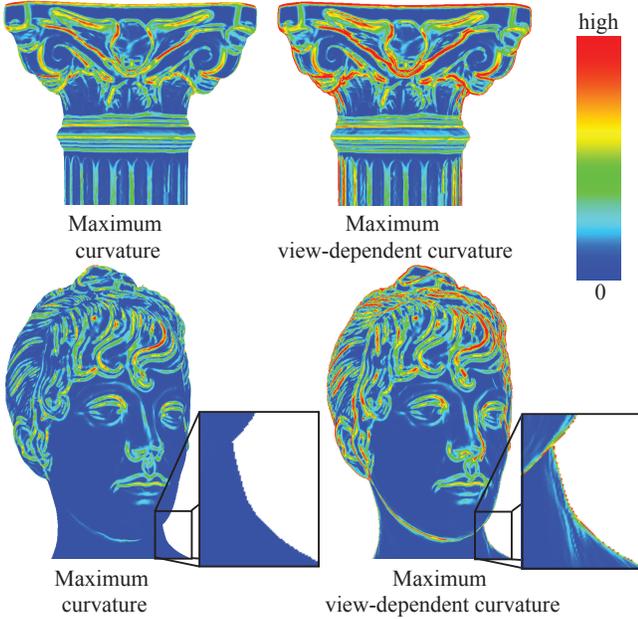
where  $\mathbf{r}$  is a vector in the tangent plane. This is the maximum of the norm of the object curvature over an ellipsoid elongated in the view direction (Fig. 4).

View-dependent curvature adds view dependency to the traditional definition of curvature. On front facing parts of an object where the object normal points towards the screen, curvature and the view-dependent curvature are the same (Fig. 5). Where the object turns away from the screen plane, the view-dependent curvature becomes much larger and the view-dependent principal direction is shifted towards the view vector.

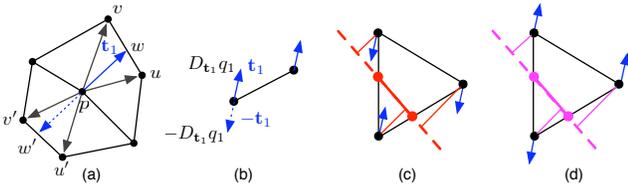
### 2.3 Apparent Ridges

Following the definition of ridges, we define apparent ridges as the loci of points at which the maximum view-dependent curvature  $q_1$  assumes a local maximum in the principal view-dependent curvature direction  $\mathbf{t}_1$ . This occurs where

$$D_{\mathbf{t}_1}q_1 = 0. \quad (3)$$



**Figure 5:** Comparison of curvature and view-dependent curvature. At front facing parts of the object, the values are similar. As the object normal turns away from the viewer, view-dependent curvature becomes much larger due to projection. View-dependent curvature approaches a maximum of infinity at the contours and so contours are extracted as apparent ridges.



**Figure 6:** We find apparent ridges on meshes with the following steps: For each edge, we flip  $\mathbf{t}_1$  so that it points towards increasing view-dependent curvature. If both  $\mathbf{t}_1$  along the edge point in opposite directions, there is a zero crossing. To test for maxima, we drop a perpendicular from each vertex to the zero crossing line. If  $\mathbf{t}_1$  at both vertices of an edge make an acute angle with their perpendicular, the zero crossing is a maximum (as in c). If not, the zero crossing is eliminated as a minimum (as in d).

We keep only the maximum points by selecting those whose higher-order derivative is negative. Note that view-dependent curvature is always positive, yet our line definition captures ridge-like and valley-like features. For example, the line for the profile of the nose and the line between the lips on the bust model (Fig. 1) are ridge-like and valley-like features respectively. We can distinguish between features by the sign of the *object space* curvature: ridge-like features have  $k_1 > 0$  and valley-like features have  $k_1 < 0$ .

### 3 Apparent Ridges on Meshes

We adapt view-dependent curvature computations to discrete triangular meshes.

**Estimating view-dependent curvature** We leverage standard techniques to estimate the curvature  $S$  at each point on the mesh [Rusinkiewicz 2004]. We then multiply by the inverse projec-

tion to obtain  $Q = SP^{-1}$ . Although our derivation relies on parallel projection, in practice we approximate a perspective camera using a local parallel projection for  $P$  in the neighborhood of each vertex. More specifically, we consider the projection line for each vertex to be the line between the viewpoint and the vertex of the object. From  $Q$ , we compute our maximum view-dependent curvature  $q_1$  and a maximum view-dependent curvature direction  $\mathbf{t}_1$  at each vertex.

**Estimating the view-dependent curvature derivative** We estimate  $D_{\mathbf{t}_1} q_1$  using finite differences (Fig. 6(a)). To compute the derivative at mesh vertex  $p$ , we compute view-dependent curvature at two points  $w$  and  $w'$  on edges of triangles adjacent to  $p$  in the direction  $\mathbf{t}$  and average the finite difference between  $p$  and the two  $w$  points. View-dependent curvature at point  $w$  on an edge is obtained by linear interpolation between the two vertices  $v$  and  $u$ .

**Finding a consistent  $\mathbf{t}_1$  field** Note that  $\mathbf{t}_1$  is a vector which runs along a line from the vertex in one of two opposite directions (Fig. 6(b)). To make this field consistent across the mesh, we flip  $\mathbf{t}_1$  to point in the direction of the positive derivative, where view-dependent curvature is increasing.

**Locating zero crossings** We use a method inspired by Ohtake et al. [2004] to locate the zero crossings of the view-dependent curvature derivative on a mesh given the consistent  $\mathbf{t}_1$  field (Fig. 6(c) and (d)). If the  $\mathbf{t}_1$  of both vertices along an edge point in the same direction, then there is no zero crossing. If they point in different directions (defined as  $> 90$  degrees) then there is a zero crossing. We interpolate the location of the zero crossing using the values of the derivatives at each vertex [Hertzmann and Zorin 2000].

**Trimming** Zero crossings locate both minima and maxima of normal variation, but we only want to draw lines at maxima. For this, we drop a perpendicular from each vertex to the zero crossing line (Fig. 6(c) and (d)). If the positive  $\mathbf{t}_1$  at each vertex makes an acute angle with the perpendicular, then the zero crossing is a maximum. If not, the zero crossing is eliminated. This test is more robust than that suggested by Ohtake et al. [2004] who approximate the perpendicular by the edge direction.

**Thresholding** After minima are trimmed, many lines remain. This is because our method finds all local extrema independent of whether the view-dependent curvature is high or low. Informally, not only do we want points where the magnitude of the view-dependent curvature is locally a maximum, but we also want this maximum to be high. Therefore we eliminate lines based on a threshold of the view-dependent curvature. The threshold is scaled by the feature size of the mesh (average edge length) to make it dimensionless. All line definitions require a similar thresholding.

## 4 Results and Analysis

**Performance evaluation** Ridges and valleys and suggestive contours are quick to compute since the curvature is stable across viewpoints and can be precomputed for an object. Apparent ridges rely on view-dependent curvature and its derivative which must be recomputed for each viewpoint. With our unoptimized code on a 2.33 GHz Intel Core 2 Duo Macintosh, apparent ridges are computed in real time for small meshes,  $\sim 1.5$  seconds for 50,000 polygon meshes, and  $\sim 9$  second for 250,000 polygon meshes.

In addition to performance, a limitation of apparent ridges is that they involve higher-order derivatives, which makes them prone to numerical noise in digital meshes.

**Qualitative evaluation** We now discuss how apparent ridges compare to the other major feature lines used for line drawing. For fair comparison, we used constant stroke width and sharp cutoffs for line ends, and thresholded each image to match the number of gray pixels per image.

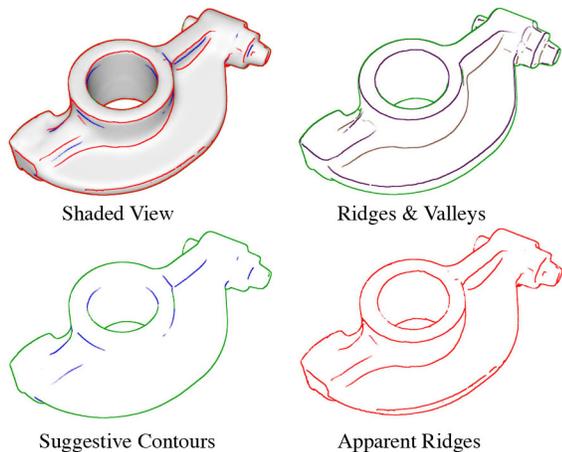
**Silhouettes and Contours** Contours are located where the normal is perpendicular to the view direction. As we move toward the contour, the view-dependent curvature approaches a maximum of infinity because of projection (Fig. 5). Though view-dependent curvature has a singularity at the exact contour, our apparent ridge computation technique on meshes does extract contours. Apparent ridge images in this paper are obtained without traditional contour extraction. This is important because it allows apparent ridges to stand alone, whereas other lines must combine with contours.

**Ridges and Valleys** Apparent ridges are closely related to traditional ridges: they share the same definition modified by a projection. By taking this projection into account however, apparent ridges are perceptually more pertinent than ridges and valleys. When the effect of projection is small, at front facing parts of the model, ridges and apparent ridges are similar. On parts of the object that turn away from the viewer, apparent ridges and ridges differ.

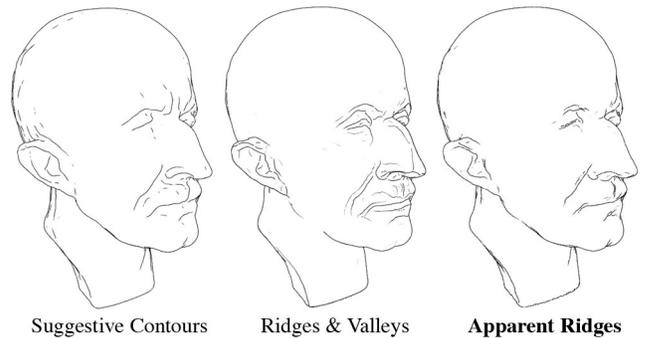
In cases where ridges and valleys do well, as on rigid objects like the rocker arm (Fig. 7) and the base of the column (Fig. 12), apparent ridges successfully mimic the same detail. In cases where ridges do less well, apparent ridges adjust ridges to be more perceptually pleasing. As seen in the tablecloth (Fig. 8), ridges capture the important rim of the table, while apparent ridges suggest a smooth feature by depicting the rim line with the ends disappearing. The same is true for the smoothed cube and dodecahedron (Fig. 10). Ridges capture the arbitrary maximal folds of the cloth which have little perceptual importance, while apparent ridges merge these ridges into more important nearby contours. Because ridges and valleys are fixed on a object, they can appear as artificial surface markings and produce boxy looking nose and mouth renderings of the Bust (Fig. 1) and Max Planck (Fig. 9) models. Apparent ridges create more appealing drawings on these human forms.

Apparent ridges are also defined in cases where ridges are ill-defined. On a symmetric Gaussian bump (Fig. 13), the apparent ridge lines extend the contour, but ridge lines don't exist because there is no maximum of curvature on this symmetric object.

**Suggestive Contours** Apparent ridges and suggestive contours are fundamentally different lines. Intuitively, suggestive con-



**Figure 7: Rocker-arm model.** Suggestive contours miss many important rim features which apparent ridges include. As seen on the shaded view image, suggestive contours do not line up with edges but rather exist on the faces of the model. Apparent ridges extend contours along the edges of the object, while suggestive contours extend contours onto the face of the object.

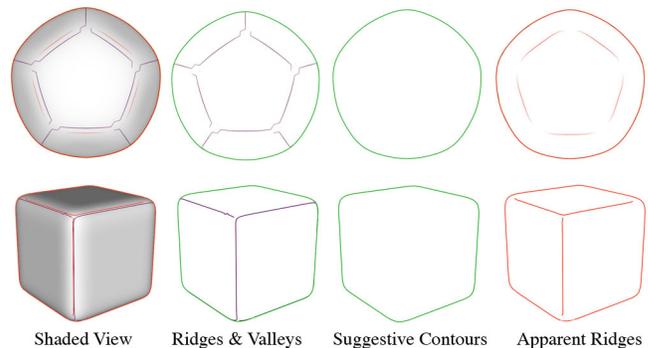


**Figure 9: Planck model.** Note that the nose of the ridge and valley drawing is unnaturally sharp. Suggestive contours seem to display excessive lines at the back of the head and don't look as natural as apparent ridges especially in the mouth and eye region.

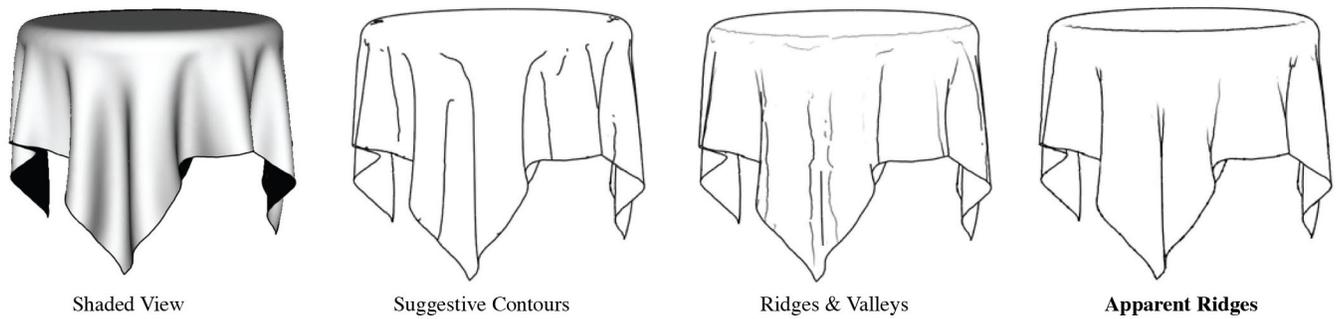
tours look at an extremum of the *normal*, where we look at the extremum of the *normal variation*. Suggestive contours look at curvature in the direction of the *view vector*, where we look at curvature in the direction of  $\mathbf{t}_1$ , the direction of *maximal normal variation as seen from the screen plane*. These directions are defined differently, but sometimes they align: foreshortening happens along the view direction, thus inflating the view-dependent curvature in that direction and making it more likely to be the direction of  $\mathbf{t}_1$ . However, even when these directions are the same, suggestive contours find where the curvature in that direction is *zero*, while we find where the curvature is *maximum*.

Given that suggestive contours and apparent ridges are so different, it is hard to say that one is clearly better than the other. Rather, both provide an interesting choice of lines with different strengths and weaknesses.

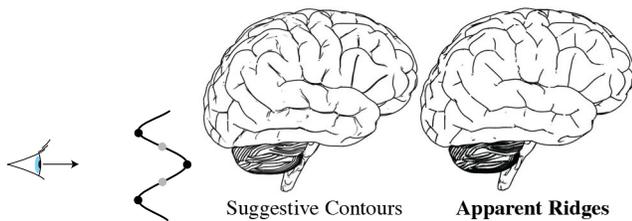
Suggestive contours and apparent ridges trade off situations where they draw single or double lines. In the brain model (Fig. 11), suggestive contours create double lines at the crevices, while we provide cleaner single lines. We locate the valley of the crevice, while suggestive contours locate two inflection points surrounding the valley. However, in the column model, (Fig. 12), we provide two apparent ridge lines for the beveled wing surface (at the ridge and valley), whereas suggestive contours provide a simpler one (at the inflection). This is also seen on the middle of the column. Drawing single inflection lines can be good for complex models, but tends to over simplify simple models. In the shaded view of the



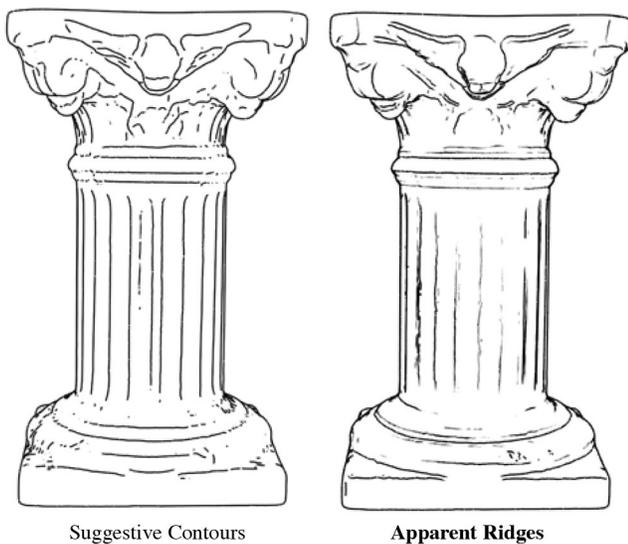
**Figure 10: Rounded dodecahedron and cube.** Compare lines of contours (green), ridges and valleys (purple and brown), suggestive contours (blue), and apparent ridges (red). Note how suggestive contours consistently miss edges of convex objects and lead to an impoverished sense of shape.



**Figure 8:** Tablecloth model. Notice how the apparent ridges convey both the smoothness of the rim of the table and the drapery of the tablecloth while suggestive contours completely miss the rim. Suggestive contours are drawn at dark regions of the shaded view, while initial experiments show that apparent ridges are drawn at important locations independent of a specific lighting situation.



**Figure 11:** Brain model. Suggestive contours are drawn at inflection points of a surface (gray dots) while apparent ridges are drawn at ridge and valley-like parts of the surface (black dots). On this model, suggestive contours draw double lines surrounding surface valleys while apparent ridges draw one in the crevice. Both produce pleasing drawings.



**Figure 12:** Column model. While the middle of the drawing is superior with suggestive contours, the bottom and top parts are superior using apparent ridges. The top angel is more clearly defined. At the bottom, the boundary between the circular column and the square base is clearly drawn with apparent ridges, while suggestive contours use a series of broken lines in the opposite direction (vertical). This is because suggestive contour lines are drawn in directions biased to be parallel to the view plane.



**Figure 13:** Gaussian bump from left to right: shaded view, occluding contour, suggestive contours and apparent ridges. Note that both the suggestive contours and apparent ridges extend contours.

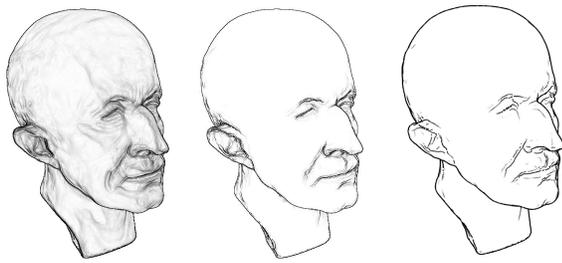
rocker-arm model (Fig. 7), suggestive contours appear on the face of surfaces between the apparent ridge lines and lead to an impoverished sense of shape as compared to apparent ridges.

One of the strong attributes of suggestive contours is that they extend contour lines. Apparent ridges seem to as well (Fig. 13). Apparent ridges lengthen the contour because nearby places of “almost contour” are locations where the perspective foreshortening effect is high, leading to high view-dependent curvature values. Note that for the rocker model (Fig. 7), while apparent ridges extend contour lines along edge-like structures, suggestive contours extend contour lines onto the face of surfaces.

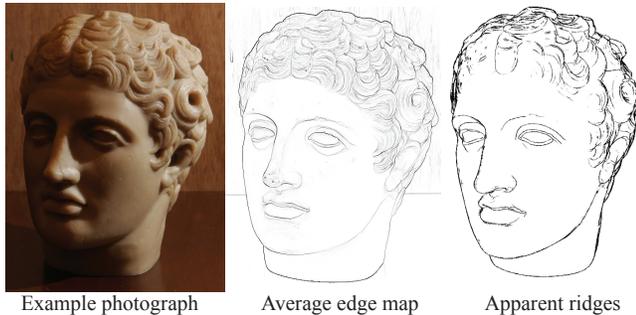
Some important features in convex regions of an object are not conveyed by suggestive contours. For example, no lines are drawn on the rim of the table (Fig. 8) or inside the rounded cube or dodecahedron (Fig. 10), whereas apparent ridges successfully locate important features on these models.

**Comparison to edge detection** Given head-on illumination with a single light source at the viewpoint and lambertian shading, suggestive contours are drawn in the shaded areas of an object. As seen in the tablecloth example (Fig. 8), dark shaded areas correspond directly to the suggestive contour lines. These lines only make sense given a certain shading setup; if the light is moved, the suggestive contour lines would seem arbitrary. On the other hand, our initial experiments show that apparent ridge lines are drawn where shading edges are detected as an average over many light sources, suggesting that apparent ridges are drawn at locations that are important independent of the light direction.

In Fig. 14, we show a Monte-Carlo experiment where a diffuse surface is rendered from a given viewpoint with thousands of random lighting configurations. The average output of a Canny edge detector [Canny 1987] on those thousands of images matches remarkably well the lines extracted with our technique. We have performed similar experiments with real photographs with flash illumination and compared Canny edge detection to an apparent ridge drawing performed on a 3D scan of the object (Fig. 15). While the



**Figure 14:** Experiments with Canny edge detection. (a) Average edge image for 1 light source and Lambertian shading over 10,000 light situations. (b) Average edge image for 10 light sources and Lambertian shading over 10,000 images. (c) Apparent ridges.



**Figure 15:** Comparison between the average Canny edge detection on 120 photographs with different lighting and our apparent ridges.

viewpoints are not matched perfectly, we can see that the two extraction approaches agree, which confirms that our lines are related to edges in shaded views. This also draws interesting relation to Raskar et al.'s [2004] NPR camera.

## 5 Conclusion

We have introduced *apparent ridges* for non-photorealistic line drawings. Apparent ridges produce visually pleasing line drawings, and capture important information about an object's shape. A variety of line types, such as contours and the sharp ridges of a polyhedron, are special cases of our definition. Where ridges and valleys do well, apparent ridges appear in similar locations. Where ridges and valleys are rigid and boxy, apparent ridges modify them to be more perceptually pertinent. Apparent ridges are related to, but distinct from, suggestive contours. Both are based on view dependent and view independent factors. Both produce pleasing images, but in many cases we find that apparent ridge images are more appealing in appearance. They produce more natural images for facial features, and more informative images for convex surfaces. Initial edge detection experiments also suggest that apparent ridges are important lines independent of a specific lighting situation.

**Acknowledgments** This work was supported by a NSF CAREER award 0447561 and Grant No. 0429739. Frédo Durand acknowledges a Microsoft Research New Faculty Fellowship and a Sloan Fellowship. Tilke Judd was supported by an NSF graduate fellowship. We thank Rusinkiewicz and DeCarlo for their `rtsc` software. We thank Yann LeTallec, MIT pre-reviewers and especially the SIGGRAPH reviewers for their extensive and insightful feedback on this work.

## References

- CANNY, J. 1987. A computational approach to edge detection. In *RCV87*, 184–203.
- DECARLO, D., FINKELSTEIN, A., RUSINKIEWICZ, S., AND SANTELLA, A. 2003. Suggestive contours for conveying shape. *ACM Transactions on Graphics* 22, 3 (July), 848–855.
- DECARLO, D., FINKELSTEIN, A., AND RUSINKIEWICZ, S. 2004. Interactive rendering of suggestive contours with temporal coherence. In *NPAR 2004*, 15–24.
- DECAUDIN, P. 1996. Cartoon looking rendering of 3D scenes. Research Report 2919, INRIA, June.
- DURAND, F., HOLZSCHUCH, N., SOLER, C., CHAN, E., AND SILLION, F. X. 2005. A frequency analysis of light transport. *ACM Transactions on Graphics* 24, 3 (Aug.), 1115–1126.
- FLEMING, R. W., TORRALBA, A., AND ADELSON, E. H. 2004. Specular reflections and the perception of shape. *Journal of Vision* 4, 9, 798–820.
- HERTZMANN, A., AND ZORIN, D. 2000. Illustrating smooth surfaces. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, 517–526.
- HERTZMANN, A. 1999. Introduction to 3d non-photorealistic rendering: Silhouettes and outlines. In *Non-Photorealistic Rendering. SIGGRAPH 99 Course Notes*, S. Green, Ed. ACM.
- HILBERT, D., AND COHN-VOSSEN, S. 1952. *Geometry and the Imagination*. Chelsea, New York.
- INTERRANTE, V., FUCHS, H., AND PIZER, S. 1995. Enhancing transparent skin surfaces with ridge and valley lines. In *VIS '95: Proceedings of the 6th conference on Visualization '95*, IEEE Computer Society, Washington, DC, USA, 52.
- KOENDERINK, J. J., AND DOORN, A. J. V. 1980. Photometric invariants related to solid shape. *Optica Acta* 27, 7 (July), 981.
- KOENDERINK, J. J. 1990. *Solid shape*. MIT Press, Cambridge, MA, USA.
- LEE, Y., MARKOSIAN, L., LEE, S., AND HUGHES, J. F. 2007. Line drawings via abstracted shading. *ACM Transactions on Graphics* 26, 3 (July).
- OHTAKE, Y., BELYAEV, A., AND SEIDEL, H.-P. 2004. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph.* 23, 3, 609–612.
- PEARSON, D. E., AND ROBINSON, J. A. 1985. Visual communication at very low data rates. *Proceedings of IEEE* 73, 795–812.
- RASKAR, R., TAN, K.-H., FERIS, R., YU, J., AND TURK, M. 2004. Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. *ACM Transactions on Graphics* 23, 3 (Aug.), 679–688.
- RUSINKIEWICZ, S. 2004. Estimating curvatures and their derivatives on triangle meshes. In *Symposium on 3D Data Processing, Visualization, and Transmission*.
- SAITO, T., AND TAKAHASHI, T. 1990. Comprehensible rendering of 3-d shapes. In *Computer Graphics (Proceedings of SIGGRAPH 90)*, vol. 24, 197–206.
- YUILLE, A. L. 1989. Zero crossings on lines of curvature. *Comput. Vision Graph. Image Process.* 45, 1, 68–87.