An Effective Illustrative Visualization Framework Based on Photic Extremum Lines (PELs)

Xuexiang Xie, Ying He, Feng Tian, Hock-Soon Seah, Xianfeng Gu, *Member, IEEE*, and Hong Qin, *Member, IEEE*

Abstract—Conveying shape using feature lines is an important visualization tool in visual computing. The existing feature lines (e.g., ridges, valleys, silhouettes, suggestive contours, etc.) are solely determined by local geometry properties (e.g., normals and curvatures) as well as the view position. This paper is strongly inspired by the observation in human vision and perception that a sudden change in the luminance plays a critical role to faithfully represent and recover the 3D information. In particular, we adopt the edge detection techniques in image processing for 3D shape visualization and present *Photic Extremum Lines* (PELs) which emphasize significant variations of illumination over 3D surfaces. Comparing with the existing feature lines, PELs are more flexible and offer users more freedom to achieve desirable visualization effects. In addition, the user can easily control the shape visualization by changing the light position, the number of light sources, and choosing various light models. We compare PELs with the existing approaches and demonstrate that PEL is a flexible and effective tool to illustrate 3D surface and volume for visual computing.

Index Terms—Surface and volume illustration, illumination, photic extremum lines (PELs), silhouettes, suggestive contours, ridges and valleys, digital geometry processing.

1 Introduction

Throughout history and technological evolution, scientific illustration has been widely used to depict the most salient features for scientific data in an accurate and intuitive way. Knowledge of human cognition shows that, so frequently an artistic drawing or painting of a same scene is proved to be both more effective in communication and more pleasing in visual experience than a photograph [13] [36] [33] [10]. This is mainly because of the abstraction nature of various artistic styles. Among various techniques used in scientific illustration, line drawings have been proven to be an effective and commonly-used way to achieve this goal because they have the capability to display information more efficiently by ignoring less important details. By taking advantage of human visual acuity, line drawings can represent a large amount of information in a relatively succinct manner, which enables them to be more expressive than photographs [30].

In graphics and visualization, extensive research has been carried out in computer-generated illustration with different types of feature lines. So far, the commonly-used feature lines on 3D shapes, e.g., contours (the external and internal silhouettes), ridge-valley lines [26], suggestive contours [6], are solely defined by local geometrical properties, such as surface normal and curvatures, and/or the view position. However, an important perceptual observation shows that human perception is highly sensitive to edge-like features, i.e., points of high *luminance variation*. It is well-known in psychology that the human perceptual system interprets natural scenes with a wealth of visual cues including luminance, color opponency, and orientation which are proven to play a vital role in human perception and a sudden spatial change in any of these factors gives rise to features [27]. This has led to great success in the research of edge detection in image processing. In 2D images, edges, characterizing the sudden change in the intensity, are one of the most important properties of objects since they correspond to the changes in geometric or photometric properties of modeled scenes.

- *X. Xie, Y. He, F. Tian and H.S. Seah are with the School of Computer Engineering, Nanyang Technological University, Singapore. Emails: {xi0002ng|yhe|asftian|ashsseah}@ntu.edu.sg*
- *X. Gu and H. Qin are with the Department of Computer Science, Stony Brook University, New York, 11794-4400. Emails: {gu|qin}@cs.sunysb.edu*

Manuscript received 31 March 2007; accepted 1 August 2007; posted online 27 October 2007. Published 14 September 2007.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

Strongly inspired by the edge detection techniques for 2D images, this paper presents the Photic Extremum Lines (PELs), a new type of feature lines which characterizes the sudden change of illumination on 3D shapes. Since the illumination depends on the view position, light models, material, texture, and geometric properties, PELs are more general than the existing feature lines. More importantly, PEL provides the user more freedom to control the desired illustration by manipulating the above parameters. As shown in Fig. 1, PEL illustrations are visually pleasing and convey the 3D surface and volume effectively. Although originated from image processing, PEL is different from image space edge detection in that PEL is computed in object space. The pixel-based representation of feature lines in image space will suffer from low precision due to the loss of 3D scene information during rendering, and is not suitable for further processing, such as stylization. In contrast, PEL does not have these constraints since PEL is defined in object space, thus, the user can totally control the illumination by manipulating the light which in turn helps the user to achieve the desired illustration.

The main contributions of this paper are:

1. We propose the PEL on 3D surfaces, a new type of feature lines which characterizes the sudden change of the luminance. We give the mathematical definition of PEL and develop an efficient algorithm to compute PEL on polygonal meshes.
2. We show that PEL is view and light dependent and develop the techniques to determine the optimal setting of light which can help the user to achieve the most desirable illustration.
3. We compare PEL with the existing feature lines thoroughly and demonstrate that PEL is a flexible and powerful tool to illustrate 3D shapes for better visualization.

2 Related Work

NPR techniques for scientific illustration. Inspired by the effectiveness in communication and the beauty of perceptual experience of traditional line drawings, researchers of scientific visualization are working towards improving the expressiveness of visualizations with computer generated line drawing techniques. Appel [1] proposed methods for silhouettes and sharp features extraction in object space and rendering in image space. Gooch *et al.* introduced an interactive technical illustration system in [10]. Hertzmann and Zorin [12] proposed algorithm to extract silhouettes on smooth surfaces and generate line drawings automatically. To convey the structure and complexity of the interior of 3D shapes, Kalnins *et al.* [16] proposed a method to draw creases in their WYSIWYG NPR drawing system, and Interrante *et al.* [14] used ridge and valley lines to enhance transparent skin surfaces. DeCarlo *et al.* [6] introduced the suggestive contours,

which extends the silhouettes and conveys the shape elegantly. Wilson and Ma [37] described a method of representing geometric complexity with line drawing in a pen-and-ink style. Ni *et al.* [25] focused on multi-scale line drawings from 3D meshes and presented a method to view-dependently control the size of shape features depicted in computer-generated line drawings.

Feature lines on surfaces. 3D shapes can be effectively illustrated with feature lines in both image space and object space. However, image space methods suffer from the low precision problem and the loss of 3D scene information during rendering. Saito and Takahashi [31] produced 3D images comprehensibly with 2D image processing operations. In [30], a classification of feature lines based on the surface derivative order and view dependency is given. It considers the surface normal as a first order differential quantity of surfaces. According to this classification, isophotes lines which connect points with constant brightness, are first order, view independent feature lines. They are widely used as the shading boundaries in toon shading [19]. Silhouettes are first order, view dependent feature lines [10] [12]. An excellent overview and classification of silhouettes generation algorithms for polygonal models are provided in [15]. Suggestive contours [6] are second order, view dependent feature lines. Ridge-valley lines [14] are third order, view independent feature lines. In contrast to the aforementioned feature lines, the proposed PELs are third order, lighting and view dependent. Light dependency property provides the user more freedom to design the desirable illustration for better visualization.

Computing the differential properties of surfaces. The computation of differential properties of surfaces is very important to any feature line extraction algorithm. The current methods fall into two categories: patch fitting and discrete differential geometry. Patch fitting methods fit an analytic surface to points either locally [34] [5] or globally [17] [26], then compute the derivatives of the fitted surface analytically. Discrete differential geometry [29] [22] [7] computes the differential properties on polygonal meshes directly, thus more efficient than patch fitting methods.

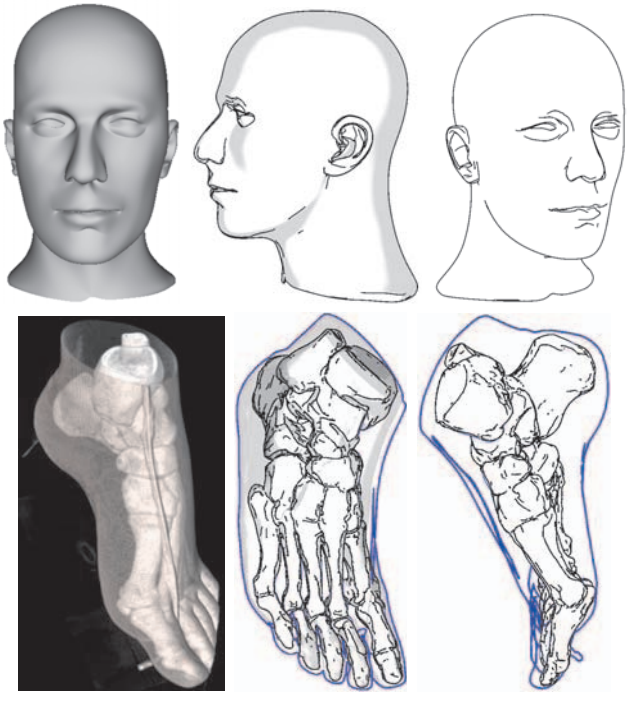


Fig. 1. Photic Extremum Lines (PELs) convey surfaces and volumes more effectively in a perceptually correct way. Two isosurfaces are extracted from volumetric datasets. The contours of the outer surface are drawn in blue. PELs of the inner surface are drawn in black. Toon shading is used in images shown in the middle.

Volume illustration. Volumetric data are widely used in scientific and medical applications. Volume illustration can be viewed as non-photorealistic rendering applied to volume visualization. The current approaches can be classified into two categories: the first approach enhances standard volume rendering algorithms with NPR techniques (e.g., [35] [28]), while the second approach applies illustrative drawing styles to volumes (e.g., [20] [3] [24] [18] [23] [32] [2]). Rendering volumetric datasets using features such as ridges, valleys, silhouettes, and suggestive contours falls into the second category. Svakhine and Ebert developed an interactive volume illustration system (IVIS) that provides a number of volume illustration enhancements [35]. Kindlmann *et al.* demonstrated that curvature-based transfer functions enhance the expressive and informative power of direct volume rendering [18]. Nagy and Klein rendered high-quality silhouettes from volumes using GPU [23]. Schein and Elber developed an adaptive algorithm to extract and visualize silhouette from volumetric data by modeling the data with *B*-spline functions [32]. Burns *et al.* presented a volumetric drawing system that directly extracts linear features, such as contours and suggestive contours, using a temporally coherent see-and-traverse framework [3]. Bruckner and Gröller [2] proposed VolumeShop, a dynamic volume illustration system which integrates many non-photorealistic rendering models to interactively alter and explore volume data.

3 Photic Extremum Lines (PELs)

3.1 Definition

Edge detection is a well-studied problem in computer vision [4] [21]. In 2D images, an edge point is defined as a point at which the gradient magnitude assumes a maximum in the gradient direction. The observation in [38] shows that for a grey-scale image of an illuminated 3D object under general illumination and reflection conditions, the zero-crossings of the second order directional derivative of the image intensity along the direction of the intensity gradient occur near the ridges and valleys of the 3D object. Inspired by the edge detection in image processing, we define Photic Extremum Lines which characterize the local variation of illumination directly on 3D surfaces.

Definition: The PEL is a set of points on the 3D surface where the variation of illumination in the direction of its gradient reaches the local maximum.

Given a smooth surface patch $S : D \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$, assume the illumination function $f : S \rightarrow \mathbb{R}$ is C^3-continuous. The gradient of f is given by [8]:

$$\nabla f = \frac{f_u G - f_v F}{EG - F^2} S_u + \frac{f_v E - f_u F}{EG - F^2} S_v \tag{1}$$

where S_u and S_v are the tangent vectors, E, F, and G are the coefficients of the first fundamental form of S. Given arbitrary point $\mathbf{p} \in S$, denote $\mathbf{w}$ the unit vector of the gradient of f, i.e., $\mathbf{w} = \frac{\nabla \mathbf{f}(\mathbf{p})}{||\nabla \mathbf{f}(\mathbf{p})||}$. Then, the PEL is defined as the set of points satisfying:

$$D_{\mathbf{w}} ||\nabla f|| = 0 \quad \text{and} \quad D_{\mathbf{w}} D_{\mathbf{w}} ||\nabla f|| < 0 \tag{2}$$

The entire solutions of $D_{\mathbf{w}} \| \bigtriangledown f \| = 0$ form closed curves or curves that terminate at the surface boundaries. Thus, the surface can be divided into two regions, i.e., $D_{\mathbf{w}} \| \bigtriangledown f \| > 0$ (colored in green in Fig. 2(d)) and $D_{\mathbf{w}} \| \bigtriangledown f \| < 0$ (colored in yellow in Fig. 2(d)). PELs (blue curves in Fig. 2(b)) correspond to part of the curves where $D_{\mathbf{w}} D_{\mathbf{w}} \| \bigtriangledown f \| < 0$, i.e., the variation of the illumination reaches its local maxima, which effectively convey useful information about the shape.

3.2 Lighting Dependency of PELs

Although originated from image processing, the key difference between PEL and edge detection is that the illumination on the 3D surface is totally user-controllable, i.e., the user can achieve the desired illustration by manipulating light freely. We will explain this in Sec. 3.3 in details.

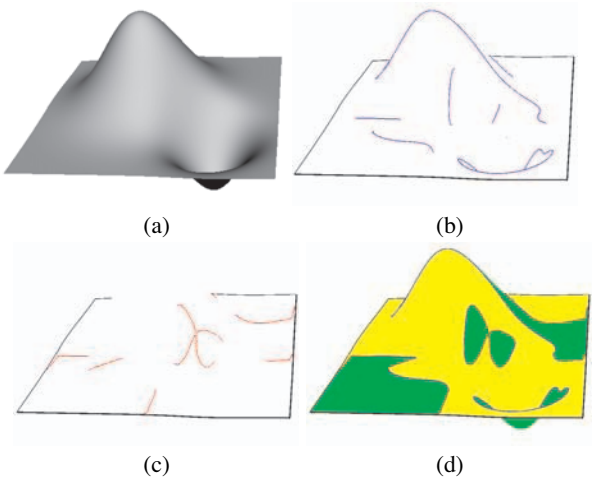


Fig. 2. The entire solutions of $D_{\mathbf{w}}\| \bigtriangledown f\| = 0$ form closed curves or curves that terminate at the surface boundaries. These curves divide the surface (a) into two regions, i.e., $D_{\mathbf{w}}\| \bigtriangledown f\| > 0$ (colored in green in (d)) and $D_{\mathbf{w}}\| \bigtriangledown f\| < 0$ (colored in yellow in (d)). PELs (blue curves in (b) and (d)) correspond to part of the curves where $D_{\mathbf{w}}D_{\mathbf{w}}\| \bigtriangledown f\| < 0$, i.e., the variation of the illumination reaches the local maxima. The red curves in (c) and (d) correspond to the points satisfying $D_{\mathbf{w}}\| \bigtriangledown f\| = 0$ and $D_{\mathbf{w}}D_{\mathbf{w}}\| \bigtriangledown f\| > 0$ which do not convey the useful information about the shape. Note that the curves on the hidden surface are removed in (b) and (c).

The illumination of 3D shapes depends on lighting conditions. Among various light models available in graphics, a commonly-used one is Phong specular-reflection model [11]:

$$I = I_{amb} + I_{diff} + I_{spec} = k_a I_a + k_d I_d(\mathbf{n}\cdot\mathbf{l}) + k_s I_s(\mathbf{v}\cdot\mathbf{r})^{n_s}, \quad (3)$$

where $\mathbf{n}$, $\mathbf{l}$, $\mathbf{v}$, $\mathbf{r}$ are the surface normal, light vector, view vector, and reflection vector, respectively. Since Eq. 3 is a function of the light and view vectors, the corresponding PELs are view and light dependent.

PELs convey shapes by emphasizing significant variations of illumination over 3D surfaces. Aiming at 3D shapes illustration with concerns of human perceptual experiences, the light model should emphasize illumination variations strongly related to shape variations, but suppress those less related or orthogonal to shape variations. In Phong shading model, the ambient light is constant which does not contribute to the variation of illumination. The diffuse light is determined by both shape and light. The variation of the diffuse light is highly affected by the variation of the surface normal. The specular light is dominated by the viewing and lighting conditions, which can be considered an orthogonal factor of the 3D surface. Thus, it may introduce feature lines which are less related to the 3D shape and distracting for 3D shape illustration. At the same time, the power factor of the specular light is more computational expensive. So we ignore the ambient light and the specular light, and consider the diffuse light only,

$$I = k_d I_d \mathbf{n}\cdot\mathbf{l}. \quad (4)$$

Note that the view vector is not involved in Eq. 4, thus, the corresponding PELs are independent of view position. However, the view dependent property is highly desirable in the real-world applications. More importantly, for real-world 3D shapes with complicated geometry and many details, the criteria to locate the feature lines are usually different in different parts and rather subjective. Thus, it is also highly desirable to provide users more freedom to control the desired illustration. To satisfy the view dependence as well as the user controllability and performance requirements, we propose the following light model for PEL:

- Main light: we set the main light source using a directional light whose light rays are parallel to the view vector. This setting is based on our perceptual experience that setting the light direction coinciding with the view direction would result in the best lighting and visual effects. Therefore, the light moves when the view point changes. As a result, the illumination and the corresponding PELs are view dependent.
- Auxiliary lights: we use *optional* spot lights to highlight the user-specified areas. The contribution of auxiliary lights to the illumination is local and independent of the view position.

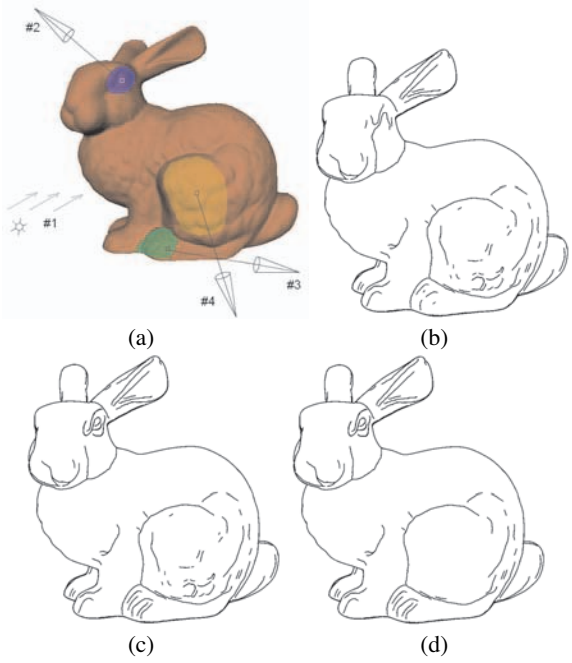


Fig. 3. Four lights (shown in (a)) are used to extract PELs from the Stanford Bunny. The main light #1 is a directional light whose direction coincides with the camera direction. By using the main light #1 alone, PELs convey the rough shape very well. However, some important lines on the eyes and feet are missing in (b). Then we use auxiliary spot lights #2 and #3 to *increase* the local contrast on eyes and feet to *add* more details (shown in (c)). Note that the Bunny body contains many short feature lines due to the small reliefs. These lines, however, are less favorable by the user. To *remove* these lines, we add another auxiliary spot light #4 to *decrease* the local contrast on the body. As a result shown in (d), most of the small feature lines are removed successfully.

The auxiliary spot lights are also defined with $\mathbf{n}\cdot\mathbf{l}$, but applied locally to user interested regions, which provide the user freedom to control PEL in different areas. Fig. 3 illustrates the proposed light model for PEL. Four lights (Fig. 3(a)) are used to illustrate the Stanford Bunny. Light #1 is the main light, whose direction coincides with the view direction. The purpose of the main light is to convey the rough geometry (Fig. 3(b)). However, some important lines, such as eyes and feet, are missing. Also, due to some small features, there exists many small PELs which cause distractions and are not helpful to convey the shape. Then, auxiliary lights are used to increase or decrease the local contrast of the specified regions, thus, PELs can be added or removed accordingly. For example, two auxiliary lights #2 and #3 are used to increase the contrast on the eye and foot (Fig. 3(c)). Another auxiliary light #4 is used to decrease the contrast on the body. As a result, we achieve better PELs shown in Fig. 3(d).

3.3 Optimal Lighting for PEL

For simple shapes, one can usually achieve the desired illustration using the main light alone. However, most of the real-world models have complicated geometry and many details, which can not be illustrated using a single light. Thus, we need to use auxiliary lights to improve the illustration. As mentioned above, the key advantage of PEL is

that users can fully control the auxiliary lights, such as the number of lights, their positions, and directions. It is usually a tedious work to manipulate these parameters manually. To minimize the effort of user interaction, we develop a technique to compute the optimal setting of the auxiliary lights automatically.

We assume that the user specifies several regions of interest to be improved. Each region is small enough that we can use just one auxiliary light. Now, the problem can be stated as follows:

Given a user-specified patch $\bar{S} \subset S$, $\mathbf{c} = \frac{\int_{\bar{S}} SdA}{\int_{\bar{S}} dA}$. We want to find a spot light l_{aux} at position $\mathbf{x}$ with light direction $\mathbf{d} = \frac{\mathbf{x}-\mathbf{c}}{\|\mathbf{x}-\mathbf{c}\|}$ such that the contrast of illumination on $\bar{S}$ is either maximized if one wants to add details or minimized if one wants to remove lines, i.e.,

$$\max \iint_{\bar{S}} ||\nabla f||^2 dA, \quad \textit{to add details}, \tag{5}$$

$$\textit{or} \qquad \min \iint_{\bar{S}} ||\nabla f||^2 dA, \quad \textit{to remove details}, \tag{6}$$

where

$$\begin{aligned} f &= I_{aux}, \\ I_{aux}(u,v) &= \frac{k_{aux} I_{aux}}{k_c + k_l d + k_q d^2} \mathbf{n}(u,v) \cdot \mathbf{l}, \\ \mathbf{l} &= \frac{\mathbf{x} - S(u,v)}{\|\mathbf{x} - S(u,v)\|}, \\ d &= \|S(u,v) - \mathbf{x}\| \end{aligned}$$

Note that the cut-off angle of the spot light gives rise to the discontinuity of the illumination which in turns would result in unnecessary feature lines. To avoid this, we require that the cut-off angle of the spot light is large enough to cover $\bar{S}$. For each point inside the spot light region (which is larger than $\bar{S}$), two illumination functions $f = I_{main}$ and $f = I_{aux}$ are defined. To compute derivatives of illumination, $f = I_{aux}$ is used for any point $\mathbf{p} \in \bar{S}$, while $f = I_{main}$ is used for any point $\mathbf{p} \in S \setminus \bar{S}$. Therefore, though the overlap of the two functions results in discontinuities, we can guarantee local continuity everywhere by switching between the two functions. Since PELs are defined with local extremum and the illumination is locally continuous for all the points, the auxiliary light will improve the PEL illustration of $\bar{S}$ without introducing unnecessary lines on the boundary.

We would point out that we set the illumination function to be $f = I_{aux}$ for the optimization of the auxiliary light and the extraction of PEL in $\bar{S}$. The optimal auxiliary lights are located in object space. Thus, the optimal setting of the auxiliary lights are not affected by the change of the main light and various viewing conditions. Though the optimization process is not real time, once computed, the auxiliary lights are fixed in the object space. Therefore, additional computational overheads are avoided. More importantly, the auxiliary lights improve the PEL illustration without causing any incoherence in animation with changing viewing conditions, as demonstrated in the accompanying video sequences.

Fig. 4 illustrates the optimal lighting for the Bunny model. Note that the discontinuity of the illumination occurs in the shaded images (Fig. 4 (b) and (c)). With the guarantee of local continuity for PEL extraction as mentioned above, user-desirable improved PEL illustration of the bunny eye is achieved without introducing meaningless feature lines on the boundaries (Fig. 4 (e) and (f)). It also demonstrates that PEL can *NOT* be extracted from image space. Using any edge detection algorithms, the aforementioned discontinuity of illuminance will definitely result in an edge, which is not acceptable. This also explains one important difference between PEL and the edge detection in image processing.

4 PEL Extraction Algorithm

This section presents the algorithm to extract PEL from 3D models. Since polygonal meshes are widely used in graphics, we develop our PEL extraction algorithm on surfaces approximated by triangle meshes.

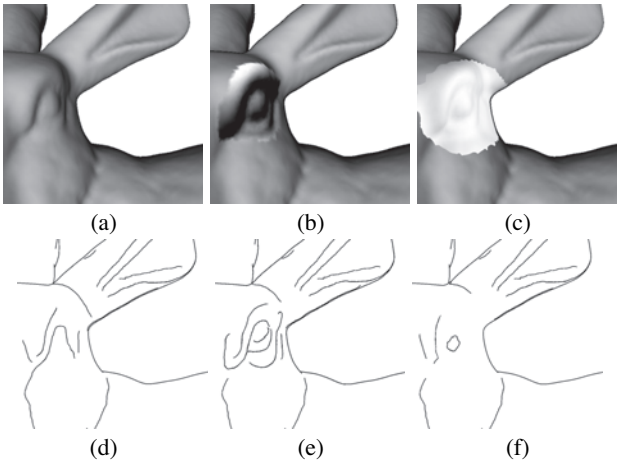


Fig. 4. An example showing improvement of the PEL illustration on the Bunny eye region using optimal lighting. Using the main light alone, the eye is not illustrated well due to the low contrast around the eye (see (a)). To improve this, we add a spot light which covers the eye and maximizes the contrast (see (b)). As a result, the features are enhanced and the corresponding PELs illustrate the eye very well (see (e)). (c) and (f) show the effect of using a spot light which *minimizes* the local contrast on the eye. As expected, the feature lines are removed accordingly.

4.1 Algorithmic Overview

Given a triangle mesh S and the illumination f on each vertex, the extraction of PEL on S is as follows:

1. (Optional preprocessing) Smooth normal $\mathbf{n}$ of surface S.
2. Compute the gradient of illumination ∇f for each vertex.
3. Compute the directional derivative of $D_{\mathbf{w}}||\nabla f||$ along the gradient direction $\mathbf{w}$ for each vertex.
4. Detect the zero-crossing and filter out the ones which are the local minima, i.e., $D_{\mathbf{w}} D_{\mathbf{w}} ||\nabla f|| > 0$.
5. Trace the zero-crossings to get the PEL.

4.2 Preprocessing

The purpose of preprocessing is to reduce the noise of illumination. In our proposed light model, both the main diffuse light and the auxiliary spot lights are functions of $\mathbf{n} \cdot \mathbf{l}$. Thus, the noise of illumination is mainly caused by the noise of geometry, i.e., the surface normal. We apply the bilateral filter [9] to process the vector field $\mathbf{n}$ on S. For each vertex $\mathbf{v}$, denote its neighborhood by $N(\mathbf{v})$. The bilateral filter is defined as:

$$\hat{\mathbf{n}}(\mathbf{v}) = \frac{\sum_{\mathbf{p} \in N(\mathbf{v})} W_c(\|\mathbf{p} - \mathbf{v}\|) W_s(\rho(\mathbf{v}, \mathbf{p})) \mathbf{n}(\mathbf{p})}{\sum_{\mathbf{p} \in N(\mathbf{v})} W_c(\|\mathbf{p} - \mathbf{v}\|) W_s(\rho(\mathbf{v}, \mathbf{p}))}$$

with

$$\rho(\mathbf{v}, \mathbf{p}) = \frac{\arccos(\|\mathbf{n}(\mathbf{v}) \cdot \mathbf{n}(\mathbf{p})\|)}{\|\mathbf{p} - \mathbf{v}\|},$$

where W_c is the closeness smoothing filter with parameter σ_c: $W_c(x) = e^{-\frac{-x^2}{2\sigma_c^2}}$, $\rho(\mathbf{v}, \mathbf{p})$ mimics the relative curvature of $\mathbf{v}$ and $\mathbf{p}$, and W_s is the feature preserving filter with parameter σ_s, $W_s(x) = e^{-\frac{-x^2}{2\sigma_s^2}}$, which penalizes large variation of the feature field. In practice, we find this step is usually helpful to improve the robustness of our algorithm for the scanned models. We follow Fleishman *et al.*'s method [9] to set the parameters σ_c and σ_s: the user selects a point on the mesh where the surface is expected to be smooth, and then a radius r of the neighborhood of the point is defined. Set $\sigma_c = r/2$ and σ_s to be the standard deviation of $\rho(\mathbf{v}, \mathbf{p})$ in the selected neighborhood.

The advantage of smoothing surface normal is that it needs to be computed only once, which avoids computational overheads and guarantees achieving PEL illustration in real time for meshes of reasonable size. Another option is to smooth the illumination f on the surface, which has to be performed in each frame once the lighting or viewing condition is changed.

4.3 Computing the Derivatives

The definition of PEL involves third order derivatives of the surface illumination, thus the key is to compute the derivatives efficiently and robustly. Similar to Rusinkiewicz's method to estimate curvatures and their derivatives [29], we compute the per-vertex derivatives by averaging adjacent per-face derivatives.

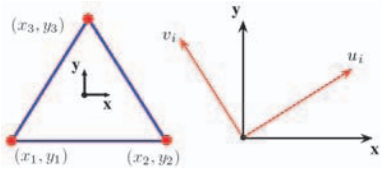


Fig. 5. The vertex and face coordinate system.

Given an arbitrary scalar function g defined on the mesh M, i.e., $g : M \to \mathbb{R}$, we consider a triangle $T \in M$ with vertices $\mathbf{v}_i \in \mathbb{R}^3$ and the associated scalar value $g_i = g(\mathbf{v}_i) \in \mathbb{R}$, $i = 1,2,3$. The per-face coordinate system is defined in the plane perpendicular to the face normal. Three vertices of T in the per-face coordinate system are denoted by $(x_i, y_i) \in \mathbb{R}^2$, $i = 1,2,3$. The per-face gradient ∇g is computed as [22]:

$$\nabla g = \begin{pmatrix} \frac{\partial g}{\partial x} \\ \frac{\partial g}{\partial y} \end{pmatrix} = \frac{1}{d_T} \begin{pmatrix} y_2 - y_3 & y_3 - y_1 & y_1 - y_2 \\ x_3 - x_2 & x_1 - x_3 & x_2 - x_1 \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \\ g_3 \end{pmatrix}$$

where $d_T = (x_1 y_2 - y_1 x_2) + (x_2 y_3 - y_2 x_3) + (x_3 y_1 - y_3 x_1)$ is twice the signed area of T.

The gradient of each vertex is considered the weighted average of the gradient of its adjacent faces. The vertex coordinate system (u, v) is defined in the plane perpendicular to the vertex normal. Note that the vertex and face coordinate systems are usually different. Thus coordinate system transformation must be applied before computing the contribution of the per-face derivatives to the per-vertex derivatives [29]. We first rotate the local vertex coordinate system to be coplanar with the local face coordinate system (see Fig. 5). Then, given a point $(u_i, v_i) = u_i \mathbf{u}_i + v_i \mathbf{v}_i$ in the vertex coordinate system, we compute the per-vertex gradient in the local face coordinate system with

$$\nabla g(u_i, v_i) = \begin{pmatrix} \mathbf{u_i} \cdot \nabla g(x,y) \\ \mathbf{v_i} \cdot \nabla g(x,y) \end{pmatrix}.$$

After the per-vertex gradients for all the adjacent faces of the vertex are computed, they will be accumulated with weighted averaging [29]. Following the method in [22], we use the Voronoi area weight w_i, which is set to be the portion of the area which lies closest to the vertex. Finally, the per-vertex derivative is computed as: $\nabla g(u,v) = \sum_i w_i \nabla g(u_i, v_i)$.

4.4 Tracing PEL

Once we compute the directional derivatives of variation of illumination at each mesh vertex $\mathbf{v}$, we are ready to check whether the mesh edge $[\mathbf{v}_1, \mathbf{v}_2]$ contains the zero-crossing. Let $h(\mathbf{v}) = D_{\mathbf{w}} \|\nabla f(\mathbf{v})\|$. If $h(\mathbf{v}_1) h(\mathbf{v}_2) < 0$, we use linear interpolation to approximate a zero-crossing on edge $[\mathbf{v}_1, \mathbf{v}_2]$

$$\mathbf{p} = \frac{|h(\mathbf{v}_2)|\mathbf{v}_1 + |h(\mathbf{v}_1)|\mathbf{v}_2}{|h(\mathbf{v}_2)| + |h(\mathbf{v}_1)|}$$

and consider $\mathbf{p}$ a PEL vertex. If two PEL vertices are detected on edges of a triangle, we connect them by a straight line segment.

Similar to [26], we also measure the strength of a PEL by the integral of $\|\nabla f\|$ along the line

$$\int \|\nabla f\| ds \approx \sum_i \frac{\|\nabla f(\mathbf{p}_i)\| + \|\nabla f(\mathbf{p}_{i+1})\|}{2} \|\mathbf{p}_i - \mathbf{p}_{i+1}\|.$$

We define a threshold T to filter out noisy PEL with strength less than T. This threshold is specified by the user.

Table 1. Comparison of feature lines

Technique	Equation	Derivatives	Dependency
Contours	$\mathbf{n} \cdot \mathbf{v} = 0$	1st order	view
Suggestive Contours	$D_{\mathbf{w}}(\mathbf{n} \cdot \mathbf{v}) = 0$	2nd order	view
Ridges& Valleys	$D_{\mathbf{t_{max}}} k_{max} = 0$ $D_{\mathbf{t_{min}}} k_{min} = 0$	3rd order	N/A
PEL	$D_{\mathbf{w}} \|\nabla f\| = 0$	3rd order	light & view

5 Comparisons of PEL with Other Feature Lines

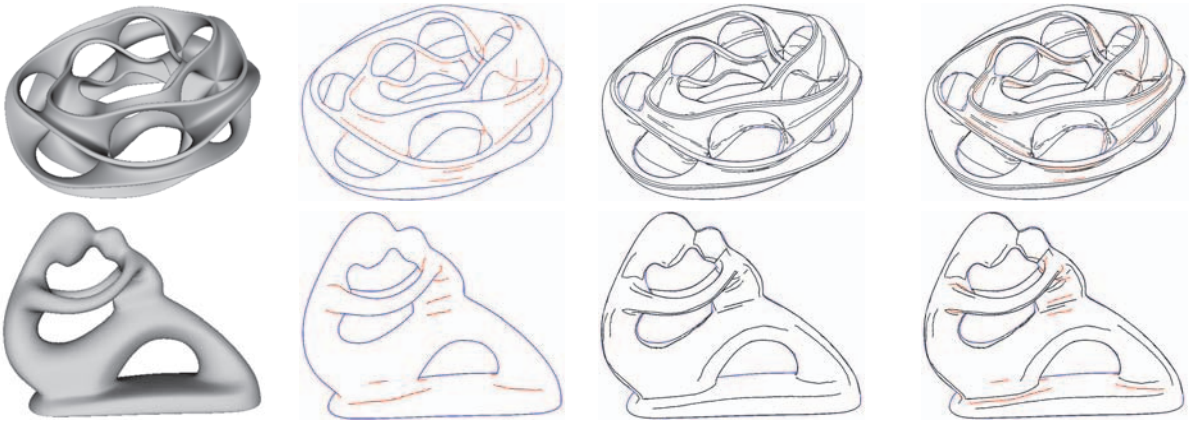
This section compares PEL with several commonly-used feature lines, contours, suggestive contours, and ridge-valley lines. These feature lines are solely determined by local geometry properties (e.g., normals and curvatures) as well as the view position. PEL, however, depends on view position and illumination (which in turns depends on local geometric properties). Table 1 summarizes the properties of contours, suggestive contours, ridges & valleys, and PEL. In [30], a classification of feature lines based on differential properties is proposed, which considers surface normal as a first order differential quantity of surfaces. In this way, suggestive contours are defined with radial curvature which is a second order derivative. Ridge-valley lines are third order derivatives of surfaces. Similarly, in our proposed light model, PELs are defined as second order derivative of $\mathbf{n} \cdot \mathbf{l}$, which turns out to be third order derivative of surfaces as well.

PEL versus Contours. Contours, one of the widely used feature lines, are lines where a surface turns away from the viewer and becomes invisible. Contours are defined as the set of points with $\mathbf{n} \cdot \mathbf{v} = 0$. Although contours show strongest cues with model-to-background distinction, contours alone are quite limited, since they can not capture the structure and complexity of the shape interior. An example is shown in Fig. 8, the contours (colored in blue) fail to show the interior geometry.

PEL versus Suggestive Contours. Suggestive contours are considered as the extension of the actual contours of surfaces. Given an arbitrary point $\mathbf{p}$ on a smooth surface S, the view vector $\mathbf{v}$ is projected onto the tangent plane of $\mathbf{p}$ to obtain $\mathbf{w}$. The normal curvature in the direction of $\mathbf{w}$ is called as radial curvature k_r. Then the suggestive contour generator is the set of points satisfying [6]:

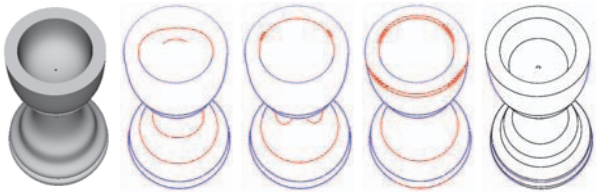
$$k_r = 0, \ where \ D_{\mathbf{w}} k_r > 0.$$

Suggestive contours extend the actual contours of surfaces in concave regions and convey the shape in a succinct and elegant way. However, they can not illustrate salient features in convex regions. Examples are shown in Fig. 6 and Fig. 8. To compensate for this, DeCarlo *et al.* [6] suggested to set different value of k_r, i.e., given a user-specified value c, compute the points satisfying $k_r = c$. By carefully choosing a positive value c, we can get suggestive contours in certain convex regions, while losing useful suggestive contours in other regions including concave regions as a tradeoff. So for a real-world complicated shape with many convex and concave regions, it is almost impossible to find a globally consistent value c for all regions. Fig. 7 illustrates this phenomenon by showing the suggestive contours with three different value $c < 0$ (Fig. 7(b)), $c = 0$ (Fig. 7(c)), and $c > 0$ (Fig. 7(d)). In contrast, the proposed PEL method works well on both convex and concave regions ((Fig. 7(e))). Simultaneously, suggestive contours may also extend contour lines onto smooth surfaces which cause visual distraction, as shown in Fig. 8.



(a) Shaded model (b) Suggestive contours (c) PEL (d) Suggestive contours and PEL

Fig. 6. Comparison of PEL with suggestive contours. Suggestive contours extend contours by adding interior information in **concave** regions, but can not convey salient features in **convex** regions. PEL is defined with the variation of the illumination, thus, can appear anywhere the illumination changes significantly. The combination of PEL and suggestive contours (d) shows that PEL locates in different positions with suggestive contours, including **convex** regions. Contours, suggestive contours and PELs are drawn in blue, red, and black, respectively.



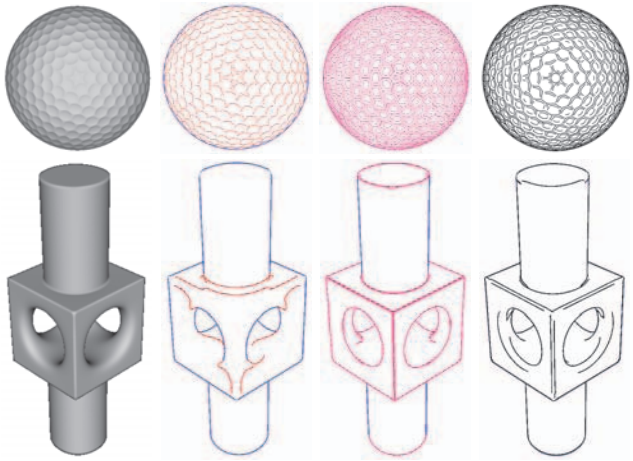
(a) Shading (b) $k_r = -1$ (c) $k_r = 0$ (d) $k_r = 1$ (e) PEL

Fig. 7. Suggestive contours (c) are the set of points on the surface at which the radial curvature $k_r = 0$. The radial curvature of the convex region is always positive, i.e., $k_r > 0$. Thus, suggestive contours occur only on the concavities. By offsetting the radial curvature $k_r = c$, suggestive contours change accordingly (see (b) to (d)). With a positive offset $k_r = c > 0$, suggestive contours appear on *SOME* of the convex parts. However, it is impossible to choose an appropriate value c such that suggestive contours appear on *ALL* the convex and concave parts simultaneously. PEL (e) does not have such restriction and can appear anywhere where the illumination changes significantly. Contours, suggestive contours, and PEL are drawn in blue, red and black, respectively.

PEL versus Ridges/Valleys. Ridge/valley lines (a.k.a. curves on a surface along which the surface bends sharply) are powerful shape descriptors. For a smooth surface S, we denote by k_{max} and k_{min} the maximum and minimum principal curvatures, and $\mathbf{t_{max}}$ and $\mathbf{t_{min}}$ the corresponding principal directions. By denoting the derivatives of the principal curvatures along their corresponding curvatures directions $e_{max} = \partial k_{max}/\partial t_{max}$ and $e_{min} = \partial k_{min}/\partial t_{min}$, ridges and valleys are the extrema of the principal curvatures along their curvature directions:

$$e_{max} = 0, \quad \partial e_{max}/\partial t_{max} < 0, \quad k_{max} > |k_{min}|, \quad (ridges)$$
$$e_{min} = 0, \quad \partial e_{min}/\partial t_{max} > 0, \quad k_{max} < |k_{min}|, \quad (valleys)$$

Ridge/valley lines convey shapes where surface bends sharply. However, visually salient features may also occur in smooth regions such as the vertical contours of the cylindrical part of the CSG model of Fig. 8 where ridge-valley lines fail. Comparing with ridge/valley lines, PEL alone can convey features in smooth regions. Another widely recognized limitation of ridge/valley lines is that they are view-independent curves. Thus, they are locked on the object without being able to slide on it when the viewing conditions change, which does not make a natural looking line drawing in animation sequences. Examples are shown in the accompanying video sequences to highlight this effect. It is also demonstrated in Fig. 9 that the illustration of ridge/valley lines appears more like surface marks rather than a natural line drawing.



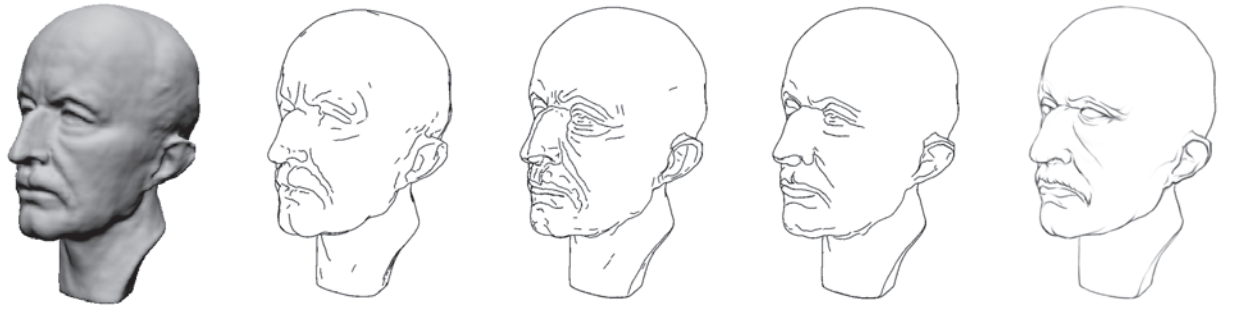
(a) Shaded model (b) Suggestive contours (c) Ridges/ valleys (d) PEL

Fig. 8. Comparison of PEL with other feature lines. (b) shows that contours can not convey the structure and complexity of the interior, suggestive contours fail to illustrate salient features in convex regions, and suggestive contours extend contours onto smooth surfaces which causes visual distraction. (c) shows that ridge-valley lines with contours convey shapes nicely, but ridge-valley lines are view independent and can not capture features in smooth regions, such as the vertical contours of the cylindrical part of the CSG model. PEL alone in (d) works well in all these aspects. The contours, suggestive contours, ridge-valley lines, and PEL are drawn in blue, red, cyan, and black, respectively.

We also compare the feature lines with hand-drawn illustration done by artists in Fig. 9. Clearly, PEL mimics the hand-drawn illustration better than all the other feature lines. We survey and discuss with 10 artists from a cartoon animation studio. We find out that our approach of defining PEL based on the significant variation of illumination is quite similar to the way of finding a line where an artist actually draws. As the result of our survey, 9 of the 10 artists agree that the PEL illustration appears better than the other feature lines shown in Fig. 9.

6 Experimental Results and Discussion

We conduct our experiments on a workstation with a 3.2GHz Xeon processor and 3GB of memory. With our un-optimized code, 0.170 to 1.684 seconds per frame can be achieved with test surfaces ranging from $44K$ to $320K$ triangles, as shown in Table 2. The optional surface



(a) Shaded model (b) Suggestive contours (c) Ridges & Valleys (d) PEL (e) Hand-drawn illustration

Fig. 9. PELs appear more consistent to the hand-drawn illustration than suggestive contours and ridge-valley lines.

Table 2. Performance of PEL illustration

Model	♯Δ	Seconds/Frame
Synthetic Surface (Fig. 2)	44K	0.170
Max-Planck (Fig. 9)	98K	0.350
Bunny (Fig. 3)	144K	0.515
Foot (Fig. 1)	150K	0.650
Sculpture (Fig. 6)	200K	0.869
Head (Fig. 12)	320K	1.684

normal smoothing with bilateral filtering is performed only once in the preprocessing step. And the optimal lighting is also computed once for each auxiliary light when it is introduced. We have also developed a user-friendly interface that allows the user to easily select the to-be-improved regions on 3D surfaces. Fig. 10 shows examples of PEL on surfaces.

Besides surfaces, we also apply PEL to volumes. For volumetric datasets, the user first extracts several isosurfaces which are of specific interest. Then our system draws PELs and/or contours on each surface. Usually, the extracted isosurfaces are not equally important. Therefore, we only draw the contours on the isosurfaces which are less important. The user can also use various NPR shadings (e.g., toon shading and Gooch shading) to highlight the important details. Fig. 12 shows examples of PEL from volumetric datasets. Two isosurfaces are extracted from each dataset. We illustrate the outer isosurface using contours and the inner isosurface using PEL. Fig. 11 demonstrates that one can achieve better illustration by combining PEL and contours than using contours alone.

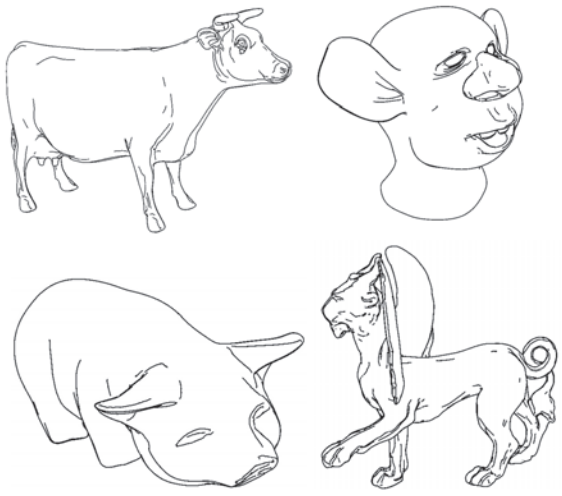


Fig. 10. Conveying 3D shapes using PELs: four examples.

(a) Contours (b) PEL & contours

Fig. 11. PEL together with contours illustrates the volumetric data much better than using contours alone.

In most of our test cases, PEL can convey the shapes similar to human perception. However, there exists several models (especially synthetic surfaces with very few or even no features) that PEL does not work well. Fig. 13 illustrates PEL on a torus. In general, torus can be nicely illustrated by contours. Without the aid of auxiliary lights, PEL is a closed, ellipse-like curve (colored in blue in Fig. 13(b)) which locates very close to the silhouettes. Clearly, PEL is not helpful to convey the shape in this case.

7 Conclusion and Future Work

In this paper, we have presented Photic Extremum Lines (PELs), a new type of feature lines for scientific illustration of 3D surfaces and volumes. In contrast to the existing feature lines which are solely defined by geometry and/or viewing positions, PEL characterizes the significant variations of illumination on 3D surfaces. PEL offers the user many degrees of freedom, such as light positions, the number of lights, and various light models, to achieve the desirable illustration. We also made comprehensive comparisons between PEL and the existing approaches. Through a wide array of experiments, we have shown that PEL enables a flexible and effective tool to illustrate 3D surfaces and volumes and reveal most important insights towards better visualization.

In our current system for volume illustration, PEL is computed on the extracted iso-surfaces of the underlying volume. Thus, it is difficult to achieve real-time interaction. Recently, Burns *et al.* [3] proposed a method to reduce the dimensionality by directly extracting feature lines that lie on iso-surfaces within a volume. One of our ongoing

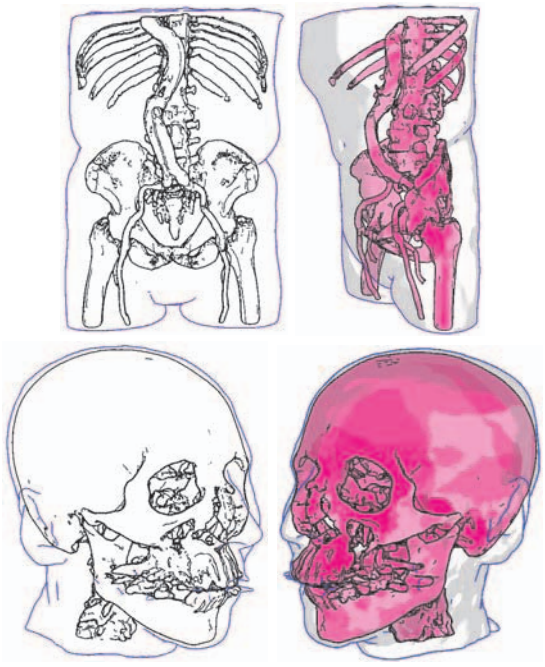
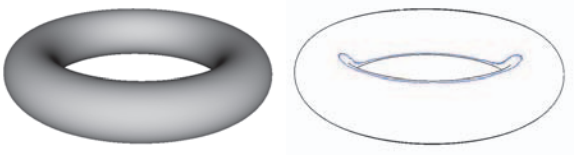


Fig. 12. PEL for volumetric datasets. Two isosurfaces are extracted from the volumetric datasets. To emphasize the inner structure, the outer isosurface is illustrated with contours only (colored in blue).



(a) Shaded model (b) PEL and contours

Fig. 13. PEL on torus has distracting lines (shown in blue) which are not helpful to convey the shape.

work is to compute PEL from volumetric datasets directly without the need of iso-surface extraction. We will also work on the hardware implementation to further speedup the PEL computation for surfaces and volumes.

ACKNOWLEDGEMENTS

The authors would like to thank Doug DeCarlo and his co-workers who make the suggestive contours software available to the public. Thanks also go to Lv Peng who draws Fig. 9 (e). The models are courtesy of Stanford University, Princeton University, Rutgers University, Hugues Hoppe, and Michael Meißner.

REFERENCES

[1] A. Appel. The notion of quantitative invisibility and the machine rendering of solids. In *Proceedings of the 1967 22nd national conference*, pages 387–393. ACM Press, 1967.

[2] S. Bruckner and M. E. Gröller. Volumeshop: An interactive system for direct volume illustration. In *IEEE Visualization 2005*, pages 671–678.

[3] M. Burns, J. Klawe, S. Rusinkiewicz, A. Finkelstein, and D. DeCarlo. Line drawings from volume data. In *Siggraph 2005*, pages 512–518.

[4] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8:679–698, 1986.

[5] F. Cazals and M. Pouget. Ridges and umbilics of a sampled smooth surface: a complete picture gearing toward topological coherence. In *Rapport de Recherche RR-5294, INRIA, September*, 2004.

[6] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. In *Siggraph 2003*, pages 848–855.

[7] M. Desbrun. Applied geometry: Discrete differential calculus for graphics. *Comput. Graph. Forum*, 23(3):269–269(1), 2004.

[8] M. P. Docarmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976.

[9] S. Fleishman, I. Drori, and D. Cohen-Or. Bilateral mesh denoising. *ACM Trans. Graph.*, 22(3):950–953, 2003.

[10] B. Gooch, P. J. Sloan, A. Gooch, P. Shirley, and R. F. Riesenfeld. Interactive technical illustration.

[11] D. Hearn and M. P. Baker. *Computer Graphics*. Prentice-Hall, 1997.

[12] A. Hertzmann and D. Zorin. Illustrating smooth surfaces. In *Siggraph 2000*, pages 517–526.

[13] E. R. S. Hodges. *The Guild Handbook of Scientific Illustration*. John Wiley and Sons, 2003.

[14] V. Interrante, H. Fuchs, and S. M. Pizer. Enhancing transparent skin surfaces with ridge and valley lines. In *IEEE Visualization*, pages 52–59, 1995.

[15] T. Isenberg, B. Freudenberg, N. Halper, S. Schlechtweg, and T. Strothotte. A developer's guide to silhouette algorithms for polygonal models. *IEEE Computer Graphics and Applications*, 23(4):28–37, 2003.

[16] R. Kalnins, L. Markosian, B. Meier, M. Kowalski, J. Lee, P. Davidson, M. Webb, J. Hughes, and A. Finkelstein. WYSIWYG NPR: Drawing strokes directly on 3d models. In *Siggraph 2002*, pages 755–762.

[17] J. Kent, K. Mardia, and J. West. Ridge curves and shape analysis. In *Monograph, Department of Statistics, University of Leeds., Leeds LS2 9JT, UK, May, 1996.*, 1996.

[18] G. L. Kindlmann, R. T. Whitaker, T. Tasdizen, and T. Möller. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *IEEE Visualization*, pages 513–520, 2003.

[19] A. Lake, C. Marshall, M. Harris, and M. Blackstein. Stylized rendering techniques for scalable real-time 3d animation. In *Proceedings of NPAR 2000*, pages 13–20, 2000.

[20] A. Lu, C. J. Morris, D. S. Ebert, P. Rheingans, and C. D. Hansen. Non-photorealistic volume rendering using stippling techniques. In *IEEE Visualization*, pages 211–218, 2002.

[21] D. Marr and E. Hildreth. Theory of edge detection. In *Proc. Royal Soc. London*, volume 207, pages 187–217, 1980.

[22] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *VisMath*, pages 35–57, 2002.

[23] Z. Nagy and R. Klein. High-quality silhouette illustration for texture-based volume rendering. In *Proc. WSCG*, pages 301–308, 2004.

[24] Z. Nagy, J. Schneider, and R. Westermann. Interactive volume illustration. In *VMV*, pages 497–504, 2002.

[25] A. Ni, K. Jeong, S. Lee, and L. Markosian. Multi-scale line drawings from 3D meshes. In *I3D 2006*, pages 133–137.

[26] Y. Ohtake, A. Belyaev, and H. Seidel. Ridge-valley lines on meshes via implicit surface fitting. In *Siggraph 2004*, pages 609–612.

[27] S. E. Palmer. *Vision Science: Photons to Phenomenology*. The MIT Press, 1999.

[28] P. Rheingans and D. S. Ebert. Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Trans. Vis. Comput. Graph.*, 7(3):253–264, 2001.

[29] S. Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *3DPVT 2004*, pages 486–493.

[30] S. Rusinkiewicz, D. DeCarlo, and A. Finkelstein. Line drawings from 3D models. In *Course Note of SIGGRAPH 2005*.

[31] T. Saito and T. Takahashi. Comprehensible rendering of 3-d shapes. *SIGGRAPH Comput. Graph.*, 24(4):197–206, 1990.

[32] S. Schein and G. Elber. Adaptive extraction and visualization of silhouette curves from volumetric datasets. *Vis. Comput.*, 20(4):243–252, 2004.

[33] M. C. Sousa, A. A. Gooch, and B. Gooch. Illustrative scientific visualization framework. In *Eurographics Workshop on Computational Aesthetics*, 2005.

[34] G. Stylianou and G. Farin. Crest lines for surface segmentation and flattening. In *IEEE Transactions on Visualization and Computer Graphics 10, 5 (September/October)*, pages 536–544, 2004.

[35] N. A. Svakhine and D. S. Ebert. Interactive volume illustration and feature halos. In *Pacific Graphics*, pages 347–354, 2003.

[36] I. Viola, M. E. Gröller, M. Hadwiger, K. Bühler, B. Preim, M. C. Sousa, D. S. Ebert, and D. Stredney. Illustrative visualization. In *IEEE Visualization*, page 124, 2005.

[37] B. Wilson and K. Ma. Representing complexity in computer-generated pen-and-ink illustrations. In *Proceedings of NPAR 2004*, pages 129–137.

[38] A. Yuille. Zero crossings on lines of curvature. *Graphical Models and Image Processing*, 45:68–87, 1989.