On Edge Detection on Surfaces

Michael Kolomenkin
Technion
michkol@tx.technion.ac.il

Ilan Shimshoni
University of Haifa
ishimshoni@mis.haifa.ac.il

Ayellet Tal
Technion
ayellet@ee.technion.ac.il

Abstract

Edge detection in images has been a fundamental problem in computer vision from its early days. Edge detection on surfaces, on the other hand, has received much less attention. The most common edges on surfaces are ridges and valleys, used for processing range images in computer vision, as well as for non-photorealistic rendering in computer graphics. We propose a new type of edges on surfaces, termed relief edges. *Intuitively, the surface can be considered as an unknown smooth manifold, on top of which a local height image is placed. Relief edges are the edges of this local image. We show how to compute these edges from the local differential geometric surface properties, by fitting a local edge model to the surface. We also show how the underlying manifold and the local images can be roughly approximated and exploited in the edge detection process. Last but not least, we demonstrate the application of relief edges to artifact illustration in archaeology.*

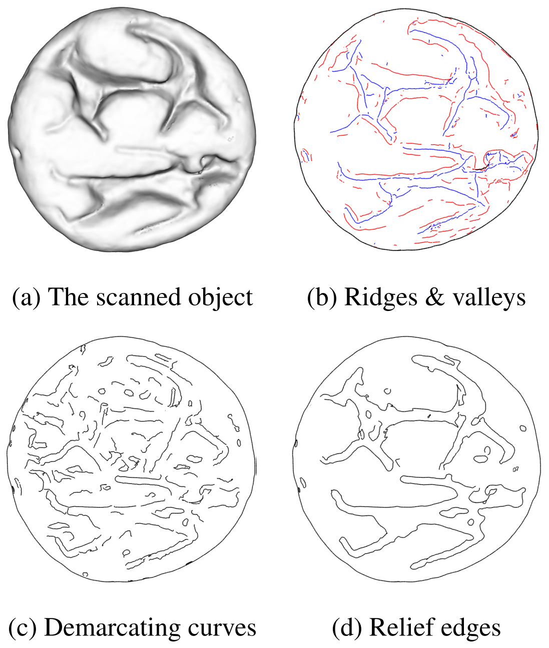

(a) The scanned object (b) Ridges & valleys
(c) Demarcating curves (d) Relief edges

Figure 1. A seal from the early Iron Age, 11th century BCE

1. Introduction

Edges in images provide low-level cues, which can be utilized in higher level processes, such as object detection, recognition, and classification, as well as motion detection, image matching, and tracking [3, 18]. They are more resilient to image formation parameters than the image intensity values, while containing less information than the whole image.

Edges on surfaces can be used in a similar way [2, 9]. While edges in images can have a variety of causes, such as depth discontinuities, textures, shadows, and other lighting effects that might hinder their use for higher level processes, edges on surfaces are the outcome of the surface geometry only (see Figure 1). This paper focuses on the problem of accurately detecting edges on surfaces.

Many of the existing algorithms detect *ridges and valleys*, which are the extrema of principal curvatures [12, 19, 22]. Other types of curves are *parabolic curves*, which partition the surface into hyperbolic and elliptic regions, and *zero-mean curvature curves*, which classify sub-surfaces into concave and convex shapes [13]. They correspond to the zeros of the Gaussian and mean curvature, respectively. Finally, *demarcating curves* are the zero-crossings of the curvature in the curvature gradient direction [14]. While portraying important object properties, the aforementioned curves sometimes fail to capture relevant features, such as weak edges, highly curved edges, and noisy surfaces. As shown in [5], no specific curve fits all applications.

This paper proposes a novel type of surface edges, termed *relief edges*, which addresses these limitations. Consider a surface as an unknown smooth manifold (*base*), on top of which a local height function is defined (e.g., a relief). The function can be considered locally as a standard image defined on the tangent plane of the base. Relief edges are the edges of this local image, i.e., a surface point $\mathbf{p}$ is a relief edge point if it is an edge point of this image.

We demonstrate that relief edges are smoother and more accurate than the other types of curves. They are better suited for certain surfaces, such as reliefs prevalent in archaeological artifacts.

The main contributions of the paper is thus threefold. First, we extend the definition of edges from functions on a plane to functions on an unknown manifold. Second, we describe an algorithm that extracts these edges. Finally, we demonstrate the utility of these edges in archaeological artifact illustration.

Algorithm overview: Relief edges are defined as the zero crossings of the normal curvature in the direction perpendicular to the edge. Initially, the edge direction is estimated for every point by fitting a step edge model to the surface. Given the edge directions, the precise edge localization is obtained (Section 3).

The quality of the estimation of the edge directions is further improved (Section 4). First, a rough estimation of the base normal is employed to limit the range of possible edge directions. Second, the edge directions are smoothed, while maintaining the properties of relief edges.

2. Related work

The paper proposes an extension of edge detection in images to arbitrary 2D surfaces. Hence, this section presents related work both on images and on surfaces. It does not describe volumetric edges [26] that are mere extensions of 2D edges to a higher dimension.

Edge detection in images: Edge detection has been extensively investigated [8]. Our work is most closely related to gradient-based edge detection, which can be generally classified into two classes.

The first class defines edges as the maximum of a smoothed first derivative or zero crossings of a smoothed second derivative. These methods differ in the manner in which they smooth and the way the derivatives are calculated. Examples include the maximum of the derivative of the Gaussian filter [4], the zero crossings of the Laplacian of the Gaussian [16], and the cubic spline filter [24].

Other methods attempt to implicitly fit the data to an edge model, such as a parametric-feature model [1] or a 1D polynomial [20]. The fitting determines both the orientation and the strength of the edge. These algorithms strongly rely on the edge model and thus might fail when the underlying assumption of the edge is unsuitable.

Our approach most resembles [17], which combines both types of edge detection algorithms. Canny edge detection is used for the initial edge estimation, followed by verification that is based on the correlation of the data with an edge template. This significantly increases the ability of the detector to eliminate spurious edges and deal with weak edges.

Edge detection on surfaces: There are two classes of edges on surfaces. The first includes ridges and valleys [12, 19, 22], which are the loci of points at which the curvature obtains extrema along the principal direction. They occur at surface normal discontinuities. Ridges and valleys portray important object properties. However, illustrating the object only by valleys (or ridges) is often insufficient, since they do not always convey its structure. Drawing both will overload the image with too many lines. It should be noted that relief edges do not compete with ridges and valleys, but rather complement them, since they portray locations with different geometric properties.

The second class includes curves that are defined as the zero crossings of some function of curvature. Examples include parabolic lines (zeros of Gaussian curvature) [13], curves of zero-mean curvature [13], and demarcating curves (zeros of the normal curvature in the curvature gradient direction) [14]. Parabolic curves are demonstrated to be noisy and unreliable [6, 14]. The curves of zero-mean curvature depend on the curvatures both along the edge and in the direction perpendicular to it; hence their error is high when the curvature in the edge direction is large. Both types of curves are isotropic operators and suffer from similar flaws as isotropic edges in images (e.g., Laplacian), such as poor behavior at corners and inexact edge localization [8]. Demarcating curves might be noisy when the curvature along the edge varies. The relief edges proposed in this paper belong to this class and address these problems.

Other kinds of curves are view dependent, i.e., they change when the viewpoint changes [6, 11, 25]. These curves are often aesthetically pleasing and thus are applicable for non-photorealistic rendering in computer graphics.

3. Relief edges

Given a surface $S(u,v) : \mathbb{R}^2 \rightarrow \mathbb{R}^3$, we assume that it consists of a smooth base surface $B(u,v) : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ and a function (local image) $I(u,v) : \mathbb{R}^2 \rightarrow \mathbb{R}$ defined on B:

$$S(u,v) = B(u,v) + \bar{\mathbf{n}}(u,v) I(u,v), \qquad (1)$$

where u and v are the coordinates of a planar parametrization and $\bar{\mathbf{n}}(u,v) : \mathbb{R}^2 \rightarrow \mathbb{S}^2$ is the normal of B ($\mathbb{S}^2$ is the unit sphere). We assume that B is locally a manifold and that its curvature has a smaller value than the curvature of I (Figure 2). The decoupling of S into B and I is unknown. Note that in the special case of an image, B is the image plane, $\bar{\mathbf{n}}(u,v)$ is constant, and I is the image intensity.

The goal is to detect edges on S that correspond to edges on the local images I. We consider the common definition of edges in images, as points at which the derivative obtains a maximum in the gradient direction. We will show that the edges can be detected without accurately estimating B or its normal $\bar{\mathbf{n}}$ – a rough estimate suffices.

In the following we first provide the necessary mathematical background and then describe the computation of the relief edges.

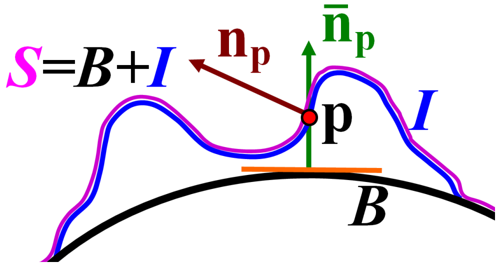


Figure 2. The surface S (magenta) is composed of a smooth base B (black) and a function I (blue). Function I at point $\mathbf{p}$ can be locally viewed as an image defined on the tangent plane (orange) of the base. Point $\mathbf{p}$ is a relief edge point if it is an edge point of this image. The normal $\mathbf{n_p}$ (brown) is the normal of S and $\bar{\mathbf{n}}_\mathbf{p}$ (green) is the normal of B corresponding to $\mathbf{p}$.

3.1. Background

Before defining the curves, we review some definitions in differential geometry [7]. The *normal section* of a surface at a point $\mathbf{p}$ in a tangent direction $\mathbf{v}$ is the intersection of the surface with the plane defined by $\mathbf{v}$ and the normal to the surface at $\mathbf{p}$. The *normal curvature* at point $\mathbf{p}$ in direction $\mathbf{v}$ is the curvature of the normal section at $\mathbf{p}$.

For a smooth surface, the normal curvature in direction $\mathbf{v}$ is

$$\kappa(\mathbf{v}) = \mathbf{v}^T \mathbf{II} \mathbf{v}. \tag{2}$$

The symmetric matrix $\mathbf{II}$ is the second fundamental form:

$$\mathbf{II} = \begin{bmatrix} \kappa_1 & 0 \\ 0 & \kappa_2 \end{bmatrix}, \tag{3}$$

where κ_1 and κ_2 are the principal curvatures.

The derivatives of the curvature are defined by a $2 \times 2 \times 2$ tensor with four unique numbers:

$$\mathbf{C} = (\partial_u \mathbf{II}; \partial_v \mathbf{II}) = \left[\begin{pmatrix} a_1 & a_2 \\ a_2 & a_3 \end{pmatrix}; \begin{pmatrix} a_2 & a_3 \\ a_3 & a_4 \end{pmatrix} \right], \tag{4}$$

where ∂_u and ∂_v are the derivatives along the principal directions. Multiplying C from its three sides by a direction vector $\mathbf{v}$, $C_{ijk}\mathbf{v}_i\mathbf{v}_j\mathbf{v}_k$ gives a scalar, which is the derivative in direction $\mathbf{v}$ of the curvature in this direction.

The *Monge form* is a polynomial approximation of a surface S on the tangent plane at a given point, expressed as:

$$S(\mathbf{v}) = \frac{1}{2}\mathbf{v}^T \mathbf{II} \mathbf{v} + \frac{1}{2} C_{ijk}\mathbf{v}_i\mathbf{v}_j\mathbf{v}_k = \tag{5}$$
$$\tfrac{1}{2}(\kappa_1 u^2 + \kappa_2 v^2) + \tfrac{1}{6}(a_1 u^3 + 3a_2 u^2 v + 3a_3 u v^2 + a_4 v^3),$$

where u and v are the coordinates of $\mathbf{v}$ in the principal directions. We will be using the Monge form to locally estimate the surface, utilizing the state-of-the-art techniques developed for estimating the curvature and its derivative [10, 15, 23].

3.2. Computing relief edges

Relief edges are computed in two steps: estimating the edge direction at every point and determining the relief edge points using this estimation. We elaborate on these steps below.

Estimating the edge direction: The edge direction is estimated by fitting an edge model that best approximates the surface locally. Below we first describe our edge model and then the process of fitting it to the surface.

We utilize the commonly-used smoothed step edge to model relief edges. Since B is unknown, all our computations are performed with respect to the local tangent plane of S at $\mathbf{p}$, where the principal directions define the coordinate system on this plane. To locally approximate surface S, we use the Monge form polynomial (Equation 5). A smoothed step edge E passing through point $\mathbf{p}$ in direction $(-\sin(\theta), \cos(\theta)) \equiv (-s, c)$ can be approximated by a cubic polynomial as:

$$E(\theta, \alpha, u, v) = \frac{1}{6}\alpha(cu + sv)^3 = \tag{6}$$
$$= \frac{1}{6}\alpha(c^3u^3 + 3c^2su^2v + 3cs^2uv^2 + s^3v^3),$$

where α is the edge intensity, and u and v are the local coordinates. (Note that the other coefficients of the polynomial are zero because the step edge is constant along its direction and antisymmetric in the perpendicular direction.)

Using polar coordinates: $(u, v) = (\rho\cos(\phi), \rho\sin(\phi)) \equiv (\rho\tilde{c}, \rho\tilde{s})$, Equations 5 and 6 can be rewritten as:

$$S = \frac{\rho^2}{2}(\kappa_1\tilde{c}^2 + \kappa_2\tilde{s}^2) + \frac{\rho^3}{6}(a_1\tilde{c}^3 + 3a_2\tilde{c}^2\tilde{s} + 3a_3\tilde{c}\tilde{s}^2 + a_4\tilde{s}^3),$$
$$E(\theta, \alpha) \equiv \alpha\tilde{E}(\theta) = \alpha\frac{\rho^3}{6}(c^3\tilde{c}^3 + 3c^2s\tilde{c}^2\tilde{s} + 3cs^2\tilde{c}\tilde{s}^2 + s^3\tilde{s}^3).$$

We define the orientation of the edge as the direction that best fits the edge model. In other words, we seek $(\hat{\theta}, \hat{\alpha})$ that minimize the difference between $E(\hat{\theta}, \hat{\alpha})$ and S. We define the approximation error as:

$$\text{Err}(\theta, \alpha) = \int \|E(\theta, \alpha) - S\|^2 \rho \, d\rho \, d\phi, \tag{7}$$

where the integral is defined over a neighborhood of $\mathbf{p}$ and ρ is the Jacobian of the polar coordinates substitution. The optimal edge is determined by $(\hat{\theta}, \hat{\alpha}) = \arg\min \text{Err}(\theta, \alpha)$.

We reformulate Equation 7 in terms of vectors in the polynomial space of cos and sin. This formulation allows us to represent our optimization problem as the problem of finding the roots of a third-order polynomial of $\sin^2(\theta)$, as explained below.

Let the basis vectors and their inner product be:

$$\mathbf{x}_1 = \tilde{c}^3,\ \mathbf{x}_2 = \tilde{c}^2\tilde{s},\ \mathbf{x}_3 = \tilde{c}\tilde{s}^2,\ \mathbf{x}_4 = \tilde{s}^3,\ \mathbf{x}_5 = \tilde{c}^2,\ \mathbf{x}_6 = \tilde{s}^2,$$

$$\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \int_{\phi=0}^{2\pi} \mathbf{x}_i \mathbf{x}_j d\phi. \tag{8}$$

Surface S, the step edge E, and the error $\text{Err}(\theta, \alpha)$ can be rewritten in terms of the basis vectors $\mathbf{x}_i$ as:

$$S = \frac{\rho^3}{6}(a_1\mathbf{x}_1 + 3a_2\mathbf{x}_2 + 3a_3\mathbf{x}_3 + a_4\mathbf{x}_4) + \frac{\rho^2}{2}(\kappa_1\mathbf{x}_5 + \kappa_2\mathbf{x}_6)$$
$$= \frac{\rho^3}{6}S_1 + \frac{\rho^2}{2}S_2,$$
$$\tilde{E}(\theta) = \frac{\rho^3}{6}(c^3\mathbf{x}_1 + 3c^2s\mathbf{x}_2 + 3cs^2\mathbf{x}_3 + s^3\mathbf{x}_4) = \frac{\rho^3}{6}E_1(\theta),$$
$$\text{Err}(\theta, \alpha) = \int_\rho \|\alpha\tilde{E}(\theta) - S\|^2 \rho d\rho, \tag{9}$$

where the norm is calculated according to the inner product in Equation 8.

$$\text{Err}(\theta, \alpha) = \tag{10}$$
$$\alpha^2 \int \|\tilde{E}\|^2(\theta) d\rho + \int \|S\|^2 d\rho - 2\alpha \int \langle \tilde{E}(\theta), S \rangle d\rho =$$
$$= \alpha^2 \|E_1\|^2 \int \rho \frac{\rho^6}{36} d\rho + \int \|S\|^2 d\rho -$$
$$-2\alpha \langle E_1, S_1 \rangle \int \rho \frac{\rho^6}{36} d\rho - 2\alpha \langle E_1, S_2 \rangle \int \rho \frac{\rho^4}{4} d\rho.$$

Appendix A proves that $\langle E_1, S_2 \rangle = 0$. The value of $\|S\|^2$ is independent on θ and α and can be removed. Therefore, the optimal parameters need to minimize:

$$(\hat{\theta}, \hat{\alpha}) = \arg\min \ (\alpha^2 \|E_1\|^2 - 2\alpha \langle E_1, S_1 \rangle) \int \rho \frac{\rho^6}{36} d\rho. \tag{11}$$

It is interesting to note that by Equation 11, $\hat{\theta}$ and $\hat{\alpha}$ are independent on the size of the region on which the integral is computed. Since the magnitude $\|E_1\|$ of the edge is independent of its direction θ, $\hat{\theta}$ should maximize the edge–surface correlation $\langle E_1(\theta), S_1 \rangle$:

$$\hat{\theta} = \arg\max \langle E_1(\theta), S_1 \rangle. \tag{12}$$

In Appendix A we show that:

$$\hat{\theta} = \arg\max(c^3C_1 + c^2sC_2 + cs^2C_3 + s^3C_4), \tag{13}$$

where the C_is are scalars depending on the parameters of the curvature derivative tensor.

After $\langle E_1(\hat{\theta}), S_1 \rangle$ has been computed, $\hat{\alpha}$ is:

$$\hat{\alpha} = \frac{\langle E_1(\hat{\theta}), S_1 \rangle}{\|E_1(\hat{\theta})\|^2}. \tag{14}$$

Equations 13 and 14 determine the orientation and the intensity of the best fitting edge. In [14], it is shown that the maxima of an equation of the type of Equation 13 correspond to the roots of a cubic polynomial in $\sin^2(\theta)$, and thus the polynomial may have up to three maxima. Multiple maxima appear when there are several step edges that can locally fit the surface. Section 4 describes our method for choosing the appropriate one.

Determining the relief edges: The previous step computed $\hat{\theta}$ and $\hat{\alpha}$ for every point on the surface. Our goal is to find the edge points, which are the loci of points where the gradient obtains maximum in the gradient direction.

In [24] it is shown that the maximum of the gradient in the gradient direction corresponds to the zeros of the normal curvature in this direction. For a smoothed step edge, the gradient direction is perpendicular to the edge direction. Therefore, the loci of the relief edges are the zero crossings of the curvature in the direction perpendicular to the edge direction $\hat{\theta}$.

We can now formally define a relief edge point. Let $\mathbf{g_p} = [\cos(\hat{\theta}), \sin(\hat{\theta})]$ be a vector perpendicular to the edge direction at point $\mathbf{p}$, and let $\mathbf{G_p} \equiv \mathbf{g_p}^T \mathbf{II} \mathbf{g_p}$ be the value of the normal curvature in the gradient direction at $\mathbf{p}$.

Definition 3.1. *Point* $\mathbf{p}$ *is a relief edge point iff* $\mathbf{G_p} = 0$.

The algorithm is applied to meshes. To achieve sub-vertex accuracy, points on the mesh edges satisfying the constraint are found. We use the method in [14, 22], which accurately estimates the zero curves of a function on a mesh, given the function values on the vertices.

In the implementation, we threshold the error defined in Equation 7, normalized by $||S||^2$, which reflects the dissimilarity of the surface to the edge model. This removes points that satisfy Definition 3.1, but do not resemble step edges.

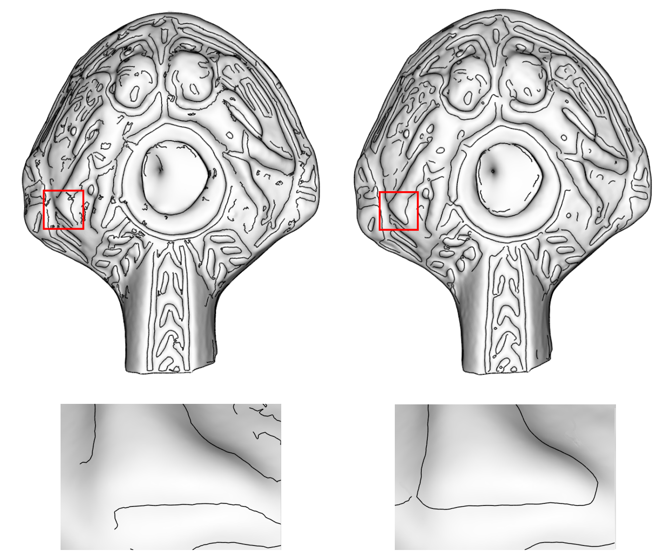
4. Enhancing relief edges

The algorithm proposed in the previous section usually produces high quality edges, as can be seen in Figure 1. However, when the surface is very noisy or deviates from the step edge model, the resulting curves might be noisy or incorrect, as illustrated in Figure 3. This section describes how to handle these cases, by utilizing the relief surface model illustrated in Figure 2.

Choosing edge orientations: The orientation obtained by Equation 13 is optimal for edges well-approximated by the step model. Deviations from the model, such as when several maxima exist in Equation 13, might lead to erroneous orientations. Though these deviations are rare, we present a method that aids in choosing the correct orientation.

The method uses a rough estimation of B's normal $\bar{n}_p$ to determine the possible orientations. Obviously, if $\bar{n}_p$ were known, the edge orientation could be computed precisely, by using an edge detector on the local image (the plane perpendicular to the normal). Though our normal's rough estimation is insufficient for a precise calculation of the edge, it suffices to limit the range of possible orientations.

To do that, we modify the function maximizing Equation 13. Let $f_{\text{corr}}(\theta) = c^3C_1 + c^2sC_2 + cs^2C_3 + s^3C_4$ be the original correlation of the edge and the surface and let $f_{\text{base}}(\theta)$ be a function that limits the range of orientations,



(a) Relief edges (b) Enhanced relief edges

Figure 3. A late Hellenistic lamp (150-50 BCE): top, full object; bottom, zoom in. Note the closing of the outline of Cupid's foot due to correcting the edge orientation and the smooth edges resulting from the smoothing procedure.

using the estimated base normal (defined below). We define the modified function as:

$$f_{\text{mod}}(\theta) = f_{\text{corr}}(\theta) \cdot f_{\text{base}}(\theta). \qquad (15)$$

If $\bar{n}_p$ were known, the edge direction θ_{base} could be calculated as the projection of $\bar{n}_p$ on the local tangent plane. Then, $f_{\text{base}}(\theta) = \delta(\theta - \theta_{\text{base}})$, where δ is the Kronecker delta function. Since $\bar{n}_p$ is known only approximately, we use:

$$f_{\text{base}}(\theta) = r_w(\theta - \theta_{\text{base}}), \qquad (16)$$

where

$$r_w(x) = \begin{cases} 1 & \|x\| \leq w \\ 0 & \|x\| > w. \end{cases} \qquad (17)$$

Below we describe how to calculate the rough estimation of $\bar{n}_p$ and the width w.

To calculate $\bar{n}_p$, the surface (S) normals are smoothed at the neighborhood of the point. This neighborhood should be sufficiently large, so as to reduce the influence of the local features on the estimated normal. Our approach utilizes an adaptive Gaussian filter, similarly to [21]. However, since the σ of the smoothing Gaussian in [21] estimates the size of the local feature, it is unsuitable for estimating the base surface. We therefore use a three times larger σ . This enables us to average the normals of several features and thus achieve a better approximation.

To calculate width w, we first compute the directions θ_{base} and $\hat{\theta}$ (Equation 13) for all the vertices. Then, w is set to the standard deviation of the histogram of the error $\|\hat{\theta} - \theta_{\text{base}}\|$. Assuming that most of the values of $\hat{\theta}$ are correct, w is statistically meaningful. When the edge is weak, its gradient estimation is unreliable, and thus it is removed, by setting $f_{\text{base}}(\theta) \equiv 1$. In the implementation, weak edges are characterized by a small angle between the base normal ($\angle(n_p, \bar{n}_p) \leq 11°$). The value $1/(n_p \cdot \bar{n}_p)$ is proportional to the magnitude of the local image gradient. This measure is most commonly used to threshold edges.

Since we are utilizing two thresholds – one that measures the similarity to the edge model (Section 3.2) and one that measures the edge strength, we can combine them to produce better results using a two-dimensional hysteresis.

Edge smoothing: While the method described above captures the features correctly, scanning noise and edge direction estimation errors may cause the edges to become jagged. In this case, smoothing should be applied. While smoothing could be applied to the edges themselves, this correction would not relate to the geometry of the surface. Therefore, a smoothing scheme which indirectly smoothes the edges is proposed.

This is done by first smoothing the function $\mathbf{G_p}$, which is defined at every vertex, yielding $\hat{\mathbf{G}}_\mathbf{p}$. Then, we compute the updated edge directions $\hat{\mathbf{g}}_\mathbf{p}$ that satisfy:

$$\hat{\mathbf{G}}_\mathbf{p} = \hat{\mathbf{g}}_\mathbf{p}^T \mathbf{II} \hat{\mathbf{g}}_\mathbf{p}. \qquad (18)$$

When such a direction does not exist (e.g., when $\hat{\mathbf{G}}_\mathbf{p}$ is required to have a negative value at a point with two positive principal curvatures), the direction that minimizes the error $\|\mathbf{G_p} - \hat{\mathbf{G}}_\mathbf{p}\|$ is chosen. Finally, $\mathbf{G_p}$ is recalculated according to $\hat{\mathbf{g}}_\mathbf{p}$.

We observed that good results are achieved when simple Gaussian smoothing is used to smooth $\mathbf{G_p}$. The smoothing parameter can be controlled by the user. In all our experiments, the σ of the Gaussian is equal to 0.8 of the median edge length of the mesh.

5. Results

This section shows results of relief edges and compares them to other major edge families. While relief edges can be used on any object, as shown in Figure 4, we focus on the challenging archaeological artifacts, which are noisy and contain edges which are difficult to detect.

Analysis of archaeological artifacts such as ceramic vessels, stone tools, coins, seals and figurines is a major source of our knowledge about the past. Traditionally, archaeological artifacts are drawn by hand and printed in the reports of archaeological excavations. These are produced manually by artists, in an extremely time-consuming and expensive procedure, prone to inaccuracies and biases. The main purpose of these drawings is to depict the features of the 3D object so that the archaeologist can visualize and compare

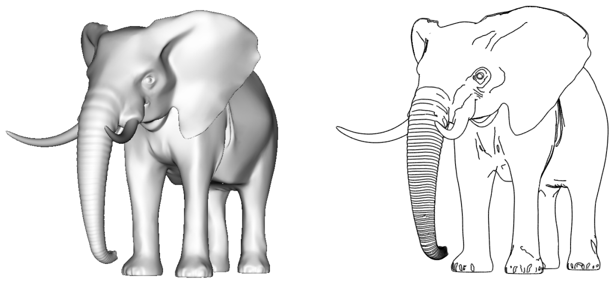


Figure 4. Elephant. Note that the relief edges are shown together with the surface contours.

artifacts. Thus, all the major features (edges) have to be detected. When, in the near future, digitization of the findings by high resolution scanners will replace the 2D representations, accurate, automatic curve drawing will be needed.

Figures 5–8 show some results. Figures 5–6 demonstrate the importance of choosing the correct orientation of the edges. Since the edges pass on almost flat surfaces, the base normal can be calculated accurately and aid in estimating the edge direction. Figure 6 is a difficult object, due to the high level of noise. Locally true edges and noisy surfaces look similar and therefore demarcating curves fail to differentiate between them. Relief edges on the other hand exploit the approximated base for estimating the local image gradient. In addition, relief edges perform better at places where the curve curvature is not constant.

Figures 7–8 demonstrate models having non-planar bases. In particular, the base surface of Figure 8 is quite complex. As can be seen, relief edges outperform the other types of edges.

The algorithm was implemented in C++ using the trimesh2 library by S. Rusinkiewicz. On a 2.66 GHz Intel Core 2 Duo PC all the steps of the algorithm run in real time except the estimation of the base normal which currently takes 16 seconds for a surface of 50K vertices and 55 seconds for a surface of 140K vertices.

6. Conclusion

This paper has extended the definition of edges from images to surfaces, for which image edges are a special case. These edges, termed relief curves, use local intrinsic surface properties together with a rough approximation of the base surface to produce superior results.

The results show that relief edges manage to capture the 3D features. They have been utilized to draw edges on scanned objects for artifact illustration in archaeology. In the future we intend to utilize these edges for shape-matching applications, which is an important challenge in archaeology, as well as in computer vision in general.

Acknowledgements: This research was supported in part by the Israel Science Foundation (ISF) 628/08, the Ollendorff foundation, and the Joint Technion University of Haifa Research Foundation. We thank Dr. A. Gilboa and the Zinman Institute of Archaeology at the University of Haifa.

References

[1] S. Baker, S. Nayar, and H. Murase. Parametric feature detection. *Int. J. of Comp. Vis.*, 27(1):27–50, 1998.

[2] A. Bartoli and P. Sturm. The 3D line motion matrix and alignment of line reconstructions. *Int. J. of Comp. Vis.*, 57(3):159–178, 2004.

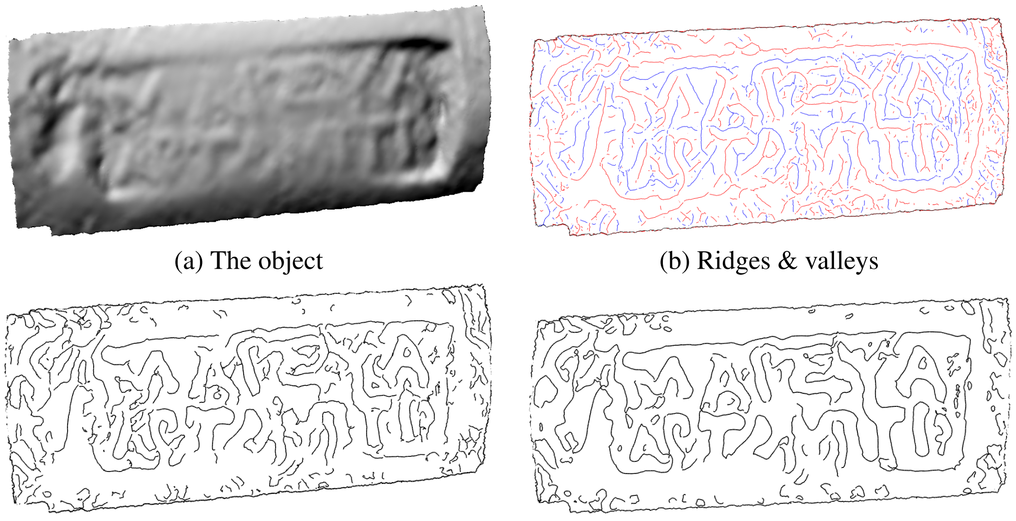
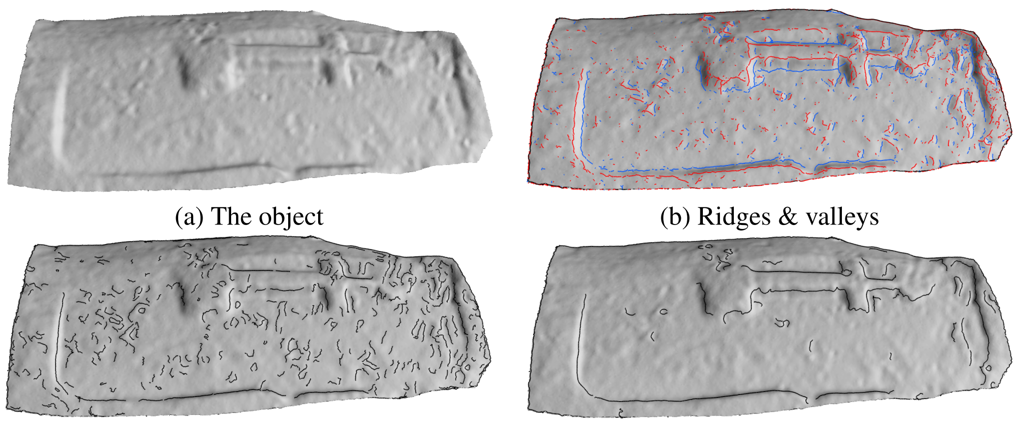
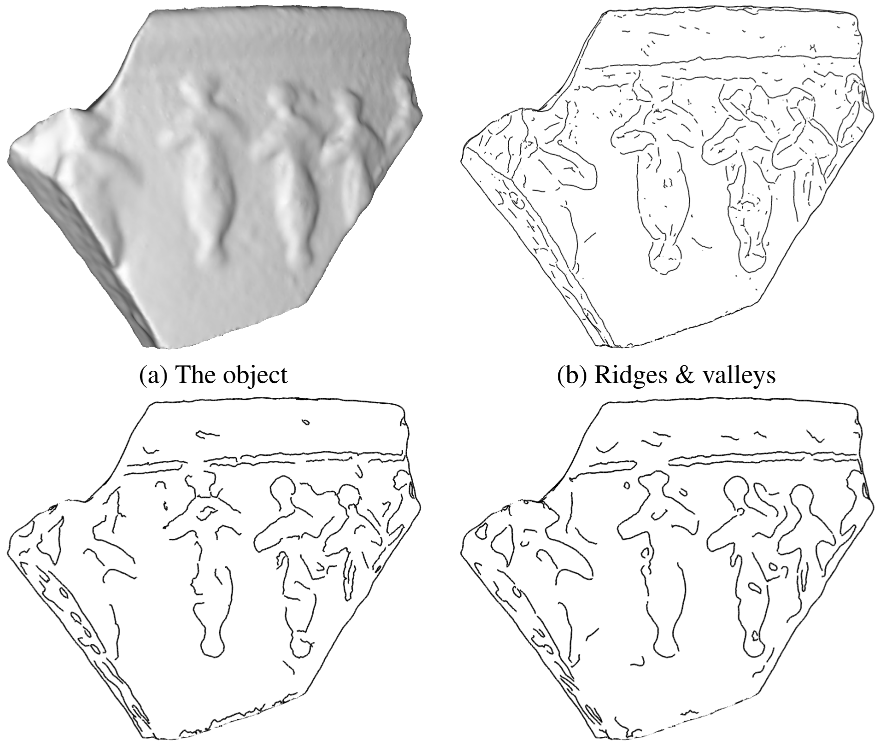


Figure 5. Hellenistic stamped amphora handle from the first century BCE. While the text is hardly legible in the 3D object, relief edges make most of the letters visible and improve on the alternatives. The text reads ΜΑΡΣΥΑ ΑΡΤΑΜΙΤΙο.


(a) The object (b) Ridges & valleys

(c) Demarcating curves (d) Relief edges

Figure 6. Hellenistic stamped amphora handle from the first century BCE. This is an example of a noisy surface. Only relief edges manage distinguish between the edges and the noise utilizing the approximated base surface.


(a) The object (b) Ridges & valleys

(c) Demarcating curves (d) Relief edges

Figure 7. Hellenistic vase. The figures are well-depicted with long meaningful edges. Note especially the quality of the recovered arms where the curvature of the edges change considerably .

[3] H. Bay, V. Ferraris, and L. Van Gool. Wide-baseline stereo matching with line segments. *IEEE Conf. on Comp. Vis. and Patt. Rec.*, 1:329 – 336, 2005.

[4] J. Canny. A computational approach to edge detection. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 8(6):679–698, 1986.

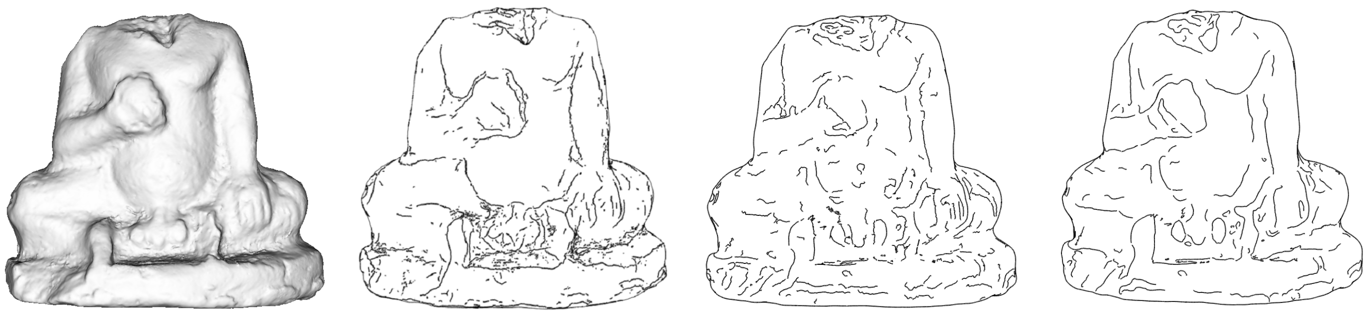
[5] F. Cole, A. Golovinskiy, A. Limpaecher, H. S. Barros, A. Finkelstein, T. Funkhouser, and S. Rusinkiewicz. Where do people draw lines? *ACM Trans. on Graph.*, 27(3):1–11, 2008.

[6] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. *ACM Trans. on Graph.*, 22(3):848–855, 2003.

[7] M. P. Do Carmo. *Differential geometry of curves and surfaces*. Prentice-Hall, 1976.

[8] D. A. Forsyth and J. Ponce. *Computer Vision – A Modern Approach*. Prentice-Hall, 2002.

[9] A. Guéziec and N. Ayache. Smoothing and matching of 3D space curves. *Int. J. of Comp. Vis.*, 12(1):79–104, 1994.



(a) The object (b) Ridges & valleys (c) Demarcating curves (d) Relief edges

Figure 8. Figurine from the Persian period (4th c. BCE). The relief edges are continuous and smoother than the alternatives. Noisy edges have been successfully removed.

[10] E. Hameiri and I. Shimshoni. Estimating the principal curvatures and the Darboux frame from real 3D range data. *IEEE SMC B*, 33(4):626–637, August 2003.

[11] T. Judd, F. Durand, and E. Adelson. Apparent ridges for line drawing. *ACM Trans. on Graph.*, 22(3):19:1 – 19:7, 2007.

[12] D. Katsoulas and A. Werber. Edge detection in range images of piled box-like objects. *ICPR*, 2:80–84, 2004.

[13] J. J. Koenderink. *Solid Shape*. MIT Press, 1990.

[14] M. Kolomenkin, I. Shimshoni, and A. Tal. Demarcating curves for shape illustration. *ACM Trans. on Graph., SIGGRAPH Asia*, 27(4), 2008.

[15] T. Langer, A. Belyaev, and H. Seidel. Exact and interpolatory quadratures for curvature tensor estimation. *Comp. Aided Geometric Design*, 24(8-9):443–463, 2007.

[16] D. Marr and E. C. Hildreth. Theory of edge detection. *Proc. of the Royal Society of London*, B(207):187–217, 1980.

[17] P. Meer and B. Georgescu. Edge detection with embedded confidence. *IEEE PAMI*, 23(12):1351–1365, 2001.

[18] K. Mikolajczyk, A. Zisserman, and C. Schmid. Shape recognition with edge-based features. *British Mach. Vis. Conf.*, 2:779 – 788, 2003.

[19] O. Monga, R. Deriche, G. Malandain, and J. P. Cocquerez. Recursive filtering and edge tracking: two primary tools for 3D edge detection. *IVC*, 9(4):203–214, 1991.

[20] V. S. Nalwa and T. O. Binford. On detecting edges. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 8(6):699–714, 1986.

[21] Y. Ohtake, A. Belyaev, and H. Seidel. Mesh smoothing by adaptive and anisotropic gaussian filter applied to mesh normals. *Vis., Model., and Visual.*, pages 203–210, 2002.

[22] Y. Ohtake, A. Belyaev, and H. Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. on Graph.*, 23(3):609–612, 2004.

[23] S. Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *3D Data Processing, Visualization and Transmission*, pages 486–493, 2004.

[24] V. Torre and T. Poggio. On edge detection. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 8:147–163, 1986.

[25] X. Xuexiang, H. Ying, T. Feng, and S. Hock-Soon. An effective illustrative visualization framework based on photic extremum lines (PELs). *IEEE Trans. on Vis. and Comp. Graph.*, 13(6):1328–1335, 2007.

[26] S. Zucker and R. Hummel. A three-dimensional edge operator. *IEEE PAMI*, 3(3):324–331, 1981.

Appendix A: Inner products of polynomials

In Equation 8, the inner products of the basis functions need to be computed. When the exponent of $\sin(\phi)$ or $\cos(\phi)$ is odd, the inner product is zero. Otherwise, the inner product is defined by the Euler beta function:

$$\mathfrak{B}(x,y) = 2\int_{\phi=0}^{\pi/2} (\sin(\phi))^{2x-1}(\cos(\phi))^{2y-1}d\phi.$$

Thus, $\langle \mathbf{x}_1, \mathbf{x}_1\rangle = \langle \mathbf{x}_4, \mathbf{x}_4\rangle = \frac{5}{8}\pi \equiv A$,

$$\langle \mathbf{x}_2, \mathbf{x}_2\rangle = \langle \mathbf{x}_3, \mathbf{x}_3\rangle = \langle \mathbf{x}_1, \mathbf{x}_3\rangle = \langle \mathbf{x}_2, \mathbf{x}_4\rangle = \frac{\pi}{8} \equiv B. \quad (19)$$

We can now calculate the inner products in Section 3.2:

$$\langle E_1, S_2\rangle = \quad (20)$$
$$\langle c^3\mathbf{x}_1 + 3c^2s\mathbf{x}_2 + 3cs^2\mathbf{x}_3 + s^3\mathbf{x}_4, k_1\mathbf{x}_5 + k_2\mathbf{x}_6\rangle = 0,$$

$$\begin{aligned}\|E_1\|^2 &= \|c^3\mathbf{x}_1 + 3c^2s\mathbf{x}_2 + 3cs^2\mathbf{x}_3 + s^3\mathbf{x}_4\|^2 = \\ &= c^6\langle \mathbf{x}_1, \mathbf{x}_1\rangle + 9c^4s^2\langle \mathbf{x}_2, \mathbf{x}_2\rangle + 9c^2s^4\langle \mathbf{x}_3, \mathbf{x}_3\rangle \\ &\quad + s^6\langle \mathbf{x}_4, \mathbf{x}_4\rangle + 6c^4s^2\langle \mathbf{x}_1, \mathbf{x}_3\rangle + 6c^2s^4\langle \mathbf{x}_2, \mathbf{x}_4\rangle \\ &= \cdots = ((1 - 3c^2 + 3c^4))A + 3A(c^2 - c^4) = A,\end{aligned}$$

$$\begin{aligned}\langle E_1, S_1\rangle &= c^3a_1\langle \mathbf{x}_1, \mathbf{x}_1\rangle + 9c^2sa_2\langle \mathbf{x}_2, \mathbf{x}_2\rangle + \\ &\quad + 9cs^2a_3\langle \mathbf{x}_3, \mathbf{x}_3\rangle + s^3a_4\langle \mathbf{x}_4, \mathbf{x}_4\rangle + \\ &\quad + 3c^3a_3\langle \mathbf{x}_1, \mathbf{x}_3\rangle + 3cs^2a_1\langle \mathbf{x}_1, \mathbf{x}_3\rangle \\ &\quad + 3s^3a_2\langle \mathbf{x}_2, \mathbf{x}_4\rangle + 3c^2sa_4\langle \mathbf{x}_2, \mathbf{x}_4\rangle = \\ &= c^3(a_1A + 3a_3B) + 3c^2s(3a_2B + a_4B) \\ &\quad + 3cs^2(3a_3B + a_1B) + s^3(a_4A + 3a_2B) = \\ &= c^3C_1 + 3c^2sC_2 + 3cs^2C_3 + s^3C_4,\end{aligned}$$

where the C_is are scalars depending on the parameters of the curvature derivative tensor (Equation 13).