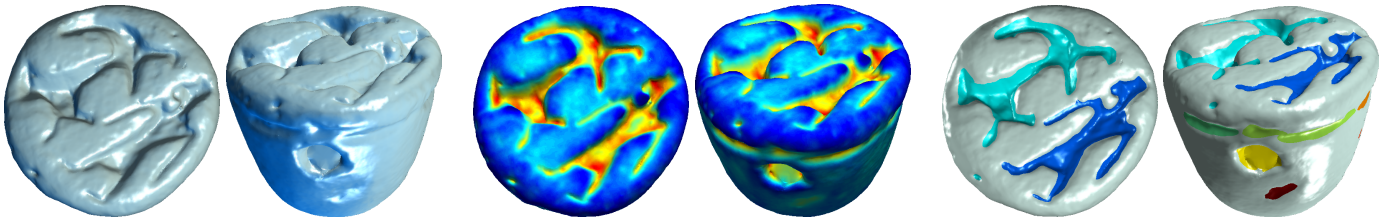
Relief Analysis and Extraction

Rony Zatzarinni* Technion
Ayellet Tal† Technion
Ariel Shamir‡ The Interdisciplinary Center



(a) Input model (b) Estimated height function (c) Relief extraction

Figure 1: *A model of a seal from the early Iron Age, 11th century BCE is composed of a base shape and reliefs (a). However, its decoupling into a base and a details is unknown. Our algorithm estimates the height function relative to the base (b), and uses it to extract the reliefs (c).*

Abstract

We present an approach for extracting reliefs and details from relief surfaces. We consider a relief surface as a surface composed of two components: a base surface and a height function which is defined over this base. However, since the base surface is unknown, the decoupling of these components is a challenge. We show how to estimate a robust height function over the base, without explicitly extracting the base surface. This height function is utilized to separate the relief from the base. Several applications benefiting from this extraction are demonstrated, including relief segmentation, detail exaggeration and dampening, copying of details from one object to another, and curve drawing on meshes.

1 Introduction

Many man-made objects are composed of a basic shape or structure and added details. For instance, a vase can be thought of as being composed of a basic tubular shape and surface reliefs; a pillar can be composed of a basic cylinder shape, groves and a decorated capital. However, the representation of 3D objects is usually a boundary surface mesh describing both the basic shape and the details with no distinction. This is true when scanning physical 3D objects or when designing them using modeling software. This means that important semantic information is missing from this type of representation. This, in turn, can result in distortion of details when applying geometric modeling operations and the destruction of the object's design. In this paper we describe a method to extract such details from a given 3D object, in effect segmenting it into its base and its details (Figure 1).

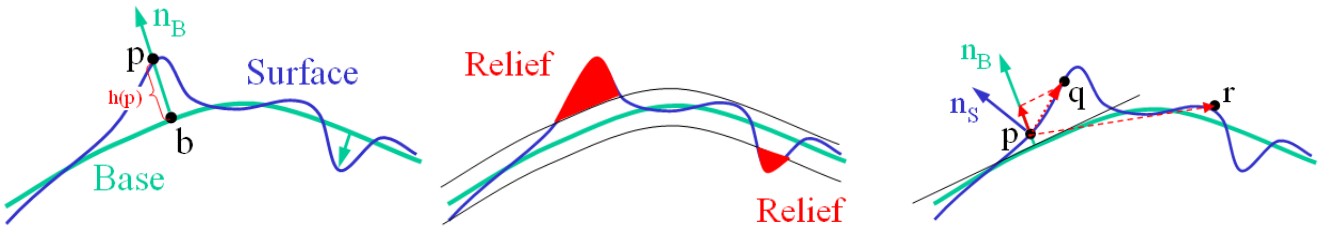
*e-mail: ronyzz@gmail.com
†e-mail:ayellet@ee.technion.ac.il
‡e-mail:arik@idc.ac.il

Object segmentation is an active research area and has numerous applications such as texture atlas generation, shape analysis, recognition, matching, shape simplification, shape modeling and shape retrieval [Attene et al. 2006; Chen et al. 2009; Shamir 2008]. The definition of a good 3D segmentation highly depends on the application itself, but two general approaches are common. Part-type segmentation divides the object into meaningful parts, such as the head and legs of a horse, while patch-type segmentation partitions the boundary of an object into surface patches.

To the best of our knowledge relief and detail segmentation has received little attention. Methods that focus on specific families of reliefs have been shown to generate good results. These families include isolated reliefs lying on a smooth background [Liu et al. 2006], reliefs lying on a textured background [Liu et al. 2007b] and periodic reliefs [Liu et al. 2007c]. The current paper aims at automatically segmenting multiple arbitrary reliefs lying on general surfaces.

A relief on a surface is a part which is above some base (or below in case of an embossed relief, see Figure 1). This means we can identify reliefs by measuring heights on a surface. However, contrary to a height function defined on a 2D planar-base domain (e.g. images), the boundary of 3D objects are curved surfaces and there is no clear base for measuring the relative height of points (hence, methods such as watershed [Mangan and Whitaker 1999] will not succeed). Towards this end, we consider the boundary surface of a 3D reliefed object as a composition of two components: a *base* surface and a *height*, which is a scalar function defined over the base in the normal direction to the base surface. Our only assumption is that the mean curvature of the base surface is smaller or equal to the curvature of the actual surface. If we could find an appropriate base surface, we would be able to measure the height of each surface point and use a threshold to define reliefs. However, given a boundary representation of an object, the decoupling of these components is unknown.

Our key observation is that there is no need to extract the real base surface to estimate details. The height function itself contains all the needed information to separate the relief from the base. Hence, we only need a good estimation of the height and not the base surface itself. Interestingly, this turns out to be easier. We show that to measure height we only need an estimation of the *normals* of the base surface, and not the surface itself. Based on the base normals, we can define relative height differences between all the points on the model. By solving a global optimization problem we eventually reach a height definition for all points. The reliefs are extracted



(a) $S = B + h \cdot n_B$ (b) Height threshold (c) Height difference

Figure 2: *(a) Defining a surface as a composition of a smooth base surface and a height function in the direction of the normal to the base. (b) A threshold on the height function is used to define the reliefs. (c) The difference in height between two points (p and q) can be found by projecting the vector connecting them on the normal to the base n_B (which is different than the normal to the surface n_S). Note that this would not be correct if the distance between the points is too large (e.g., p and r).*

by thresholding the height function, and the threshold can be computed automatically using a Gaussian mixture model (GMM) for the distribution of the height values.

The major contribution of this paper is a new algorithm for extracting reliefs. The algorithm consists of four steps: base normal estimation, height function calculation, threshold segmentation, and filtering. We present effective results on both artificial examples and challenging noisy real-life examples. We also demonstrate several applications. In particular, we show how our technique serves relief segmentation, detail exaggeration and dampening, detail copying and transfer, and curve drawing on meshes.

2 Related work

[Liu et al. 2006] define a relief as an extra material added locally to some underlying surface. The added material forms a surface with sculpted features, clearly different from the underlying surface. [Sorkine et al. 2004] refer to a relief as "coating" or high-frequency details and define it as the difference between the original surface and a low-frequency band of that surface. [Anderson and Levoy 2002; Liu et al. 2007a; Kolomenkin et al. 2009] proposed to view a relief surface as a composition of a smooth base surface and a height function defined over that base. Our paper follows the latter definition.

Relatively little work has been done in the context of relief segmentation of triangulated meshes. In [Anderson and Levoy 2002] the model of a cuneiform tablet is decomposed into a coarse B-spline, which captures the basic tablet shape, and a displacement map, which encodes the inscribed detail. Then, the B-spline surface is discarded, retaining only the displacement map.

In [Liu et al. 2006] a relief extraction problem is addressed, in which an isolated relief lies on a smooth and slowly varying background. The user needs to draw a rough closed contour on the mesh around one relief to indicate the relief that needs to be extracted. Then, a snake that starts from the user-drawn contour, evolves until it matches the boundary of the relief.

In [Liu et al. 2007a] it is shown how to use the initial segmentation of [Liu et al. 2006] to estimate the background surface lying under the relief. A B-spline surface is first fitted to the background data surrounding the relief, while ensuring this background surface smoothly continues underneath the relief. After making an initial estimate of relief offset height, a support vector machine is used to refine the segmentation.

[Liu et al. 2007b] consider the segmentation of reliefs lying on a textured background. The method is based on two approaches: classification and smoothing. In both approaches, a snake is used to move from an initial user-given contour to the desired boundary. For classification, a non-linear supervised learning method (SVM) is applied. In the smoothing approach, a simple Laplacian surface smoothing method is adopted. The snake first moves to approximately the correct location using the smoothed model; further optimization is used on the original un-smoothed model to make the snake accurately lie at the final position.

[Liu et al. 2007c] propose a method for extracting periodic reliefs. If the underlying surface and relief path vary slowly, successive repeat units approximately match each other after applying some rigid transformation. Thus, a surface registration method is used to match adjacent repeating units. The repeating units can then be identified by cutting across the relief pattern at corresponding points between adjacent repeats.

This paper attempts to extract multiple general reliefs, lying on general surfaces. These include, for instance, challenging archaeological reliefs, where both the relief and the background surfaces are very noisy and have undergone weathering processes. Moreover, our goal was to develop an automatic method, which does not rely on user input.

3 Overview

Although our primary goal is relief segmentation of 3D objects, our formulation is effective for any surface $S \subset \mathbb{R}^3$. Similar to [Kolomenkin et al. 2009], we assume S is composed of a smooth base surface $B \subset \mathbb{R}^3$ and a height function $h : S \to \mathbb{R}$. The height function represents the signed distance between the base surface B to the surface S in the direction of the base's unit normal $\vec{n}_B : B \to \mathbb{R}^3$. Hence, any point $p \in S$ can be described as the sum of two components, a point on the base $b \in B$ and a height in the normal direction at b, as follows (Figure 2(a)):

$$p = b + h(p) \cdot \vec{n}_B(b).$$

It is also assumed that the curvature of the base surface B is smaller than the curvature of S.

In practice, we usually have an orientable polygonal mesh $M = \{V_M, E_M\}$, where V_M is the set of vertices and E_M is the set of edges, and our goal is to extract the relief parts of M. We assume that M is composed of a base mesh B with the same connectivity as M, and a height function $h : M \to \mathbb{R}$, which can also be described as $h : B \to \mathbb{R}$ since there is a bijection between B and M.

Instead of trying to reconstruct the base B we extract the height function h for each mesh vertex. If we know the height of one specific point, the height of any other point can be defined relative to this point by finding their difference in height. However, the

difference in height is not easily defined globally without a base surface. Since the base surface is assumed to be smooth, one can assume that it can be approximated in a small neighborhood by a plane defined by the base-surface normal at a point. Over a plane we can measure differences in height as the projection of the vector connecting the points onto the plane's normal (Figure 2(c)). To find the differences in height between any two points, we can integrate the local differences in height over a path on the surface between the points. Consequently, to measure height we only need an estimation of the *normals* of the base surface, and not the surface itself.

By filtering the normals of the original surface S (or mesh M) with an adaptive filter we estimate the normals of the base surface B. Using these estimated normals we define the height function itself by solving a global optimization problem for the relative heights of all the points. The reliefs are defined based on thresholding of the height (Figure 2(b)). We use a Gaussian mixture model (GMM) on the distribution of the height values to find this threshold automatically.

Our basic algorithm consists of four parts. We elaborate on each of the parts in the following sections.

1. **Base normal estimation** – The normals $\vec{n}_B$ of the base mesh B are estimated using an adaptive Gaussian filtering on M.
2. **Height function calculation** – The estimated normals are used in a global minimization problem to find the height h of every vertex.
3. **Threshold segmentation** – Using thresholding of the height function, the mesh is segmented into two parts: a background part G and a relief part R, where $G \cup R = M$.
4. **Filtering** – Sharp edges are detected and discounted in the solution. Small segments are removed from the relief part R based on geometric consideration such as size and shape.

4 Base normal estimation

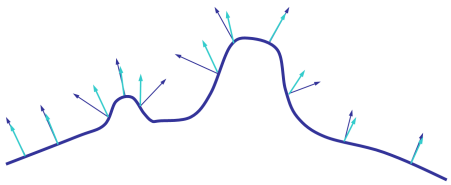


Figure 3: *Adaptive normal smoothing: blue - surface normals, cyan – estimated base normals.*

A naïve approach for finding the base B would be to apply a low pass filter on the surface of the mesh M. This approach will usually fail because smoothing deforms the base mesh as well as removes surface details. Instead, we use an estimate for the normals of the base mesh B, which turns out to be an easier task than finding the base surface itself. The base normals are found by filtering the *normals* of M with a Gaussian filter.

To find the best base normal estimation we need to smooth the normals over the whole mesh. However, the mesh can have different feature sizes in different areas. The challenge is to find the correct degree of smoothing for every normal, i.e. the standard deviation σ of the Gaussian filter. If the smoothing is too strong, the normals of the base will be deformed. If the smoothing is too weak, the normals will fit the original surface and not the base. Therefore, σ should be adapted locally to the surface. We perform the estimation in a manner similar to [Ohtake et al. 2002; Kolomenkin et al. 2009], where σ is found by the following rule:

$$\sigma_{best} = \arg\min_{\sigma} \frac{c}{\sigma^2} + \varepsilon^2(\sigma).$$

In our implementation, $c = 4.0 \times 10^{-3} l^2$, l is the arithmetic mean of the edge length of the mesh, and $\varepsilon(\sigma)$ is the local variance of the normal after smoothing it with a σ Gaussian. To find σ_{best}, we used ten uniformly-spaced values of σ between $0.6l$ to $6l$. This scheme is adaptive to the local features on the mesh: smaller values of σ are found in flat regions and larger values on curved regions. Figure 3 demonstrates our results on a one dimensional curve. It can be seen that our resulting normals coincide with the surface's normals when it is close to the base, but differ on the relief part.

5 Height function calculation

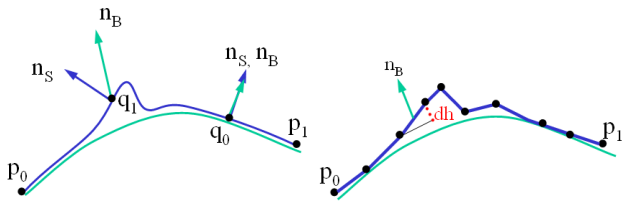


Figure 4: *Left: on the path from p_0 to p_1, when the base normal and surface normal agree (point q_0), the arc length projection on the base normal will be zero, meaning no height is gained or lost along the path. When the base normal differs from the surface normal (point q_1), the component of the arc length perpendicular to the base defines the height differences (positive or negative). Right: the discrete version of the height difference dh is the projection of the edge length between two consecutive vertices onto the base normal which is the average of the normals at these vertices.*

Let $\vec{n}_B : B \to \mathbb{R}^3$ be the estimated base normals after the adaptive Gaussian filtering. Given these normals, our goal is to find $\hat{h} : M \to \mathbb{R}$, an estimation of h.

Given a reference point $p_0 \in S$ with some known height (w.l.g. $h(p_0) = 0$), the height of every other point p_1 on the surface can be calculated as the integral of height differences along a path on the surface between p_0 and p_1. Over a flat base domain, the height difference is just the projection of the vector between the points onto the normal of the base. In general, the base surface is not flat. However, for a smooth enough surface we can assume that *locally* it is flat, and use the estimated normal of the base to find the height difference between two close enough points (see Figure 2(c)).

For points which are far away from each other, we need to integrate the height differences $dh(t)$ over the geodesic path between the two points. This is similar to the dot product of the (local) length with the (local) base normal (Figure 4, left). Therefore, given a path $\gamma(t) \subset S$, $0 \le t \le 1$ between $p_0 = \gamma(0)$ and $p_1 = \gamma(1)$, the expression for calculating the estimation of the height of p_1 is:

$$\hat{h}(p_1) = h(p_0) + \int_0^1 dh(t)dt = h(p_0) + \int_0^1 (\gamma'(t) \cdot \vec{n}_B(t))dt. \quad (1)$$

In the discrete mesh setting, we look at mesh edges and define the height difference dh between two adjacent vertices $(v_i, v_j) = e_{i,j} \in E_M$ as the projection of the vector connecting them onto the average direction of the base normals at v_i and v_j, denoted as $n_B(e_{ij})$ (Figure 4, right):

$$dh(e_{ij}) = (v_i - v_j) \cdot n_B(e_{i,j}) \quad (2)$$

To find the height of any vertex v_n given the height of some fixed vertex v_0, we can follow a path of vertices $\{v_i\}$, $0 \le i \le n$ from v_0 to v_n and use a discrete form of Equation (1):

$$\hat{h}(v_n) \quad = \quad h(v_0) + \sum_{i=1}^{n} ((v_i - v_{i-1}) \cdot n_B(e_{i,i-1})) \qquad (3)$$

This scheme suffers from two drawbacks. First, we have only an estimation of the normals of the base, calculated by smoothing. Second, we have only discrete samples of the normals at the vertices. Therefore, in practice the value of $\hat{h}$, as calculated in Equation (3), will depend on the path between vertices the v_0 and v_n, which is undesirable.

To solve these problems, we propose to use a global solution to define the height function on all vertices at once, as follows. First, we calculate $n_B(e_{ij})$ for all $e_{ij} \in E_M$. Now, we can write a system of $|E_M|$ equations for the height function estimation h, one equation for each edge e_{ij}:

$$h(v_i) - h(v_j) = dh(e_{ij}).$$

We look for $\hat{h}$ that minimizes the following functional:

$$\sum_{e_{ij} \in E_M} h(v_i) - h(v_j) - dh(e_{i,j}). \qquad (4)$$

We have $|E_M|$ equations of $|V_M|$ variables ($|E_M| > |V_M|$), which is an overly constrained system. Therefore, it is solved using the least-squares method. Furthermore, since all the equations contain only differences between the height values, adding a constant to the height $\widetilde{h} = h + const$ will yield a valid solution. Hence, in order to obtain a unique solution an additional equation needs to be added. Without loss of generality we choose an arbitrary vertex as a reference point and add the equation $h(v_0) = 0$ to the equation system (4). Once the height of the base is determined (Section 6), the height values can be shifted so as the height of the base is considered to be zero.

Now, the full equation system can be rewritten as follows:

$$\hat{h} = \arg\min_h \|S_1 h - S_2 h - dh\|, \qquad (5)$$

where S_1 and S_2 are sparse selection matrices, i.e., $S_1(k,i) = 1$ and $S_2(k,j) = 1$ if the k-th edge is $(i,j) \in E_M$ and $S_1(k,1) = 1$ if $k = |E_M| + 1$ for the soft constraint.

In order to find h, we start by manipulating Equation (5) as follows:

$$S_1 h - S_2 h = dh.$$

Let $S = S_1 - S_2$, then

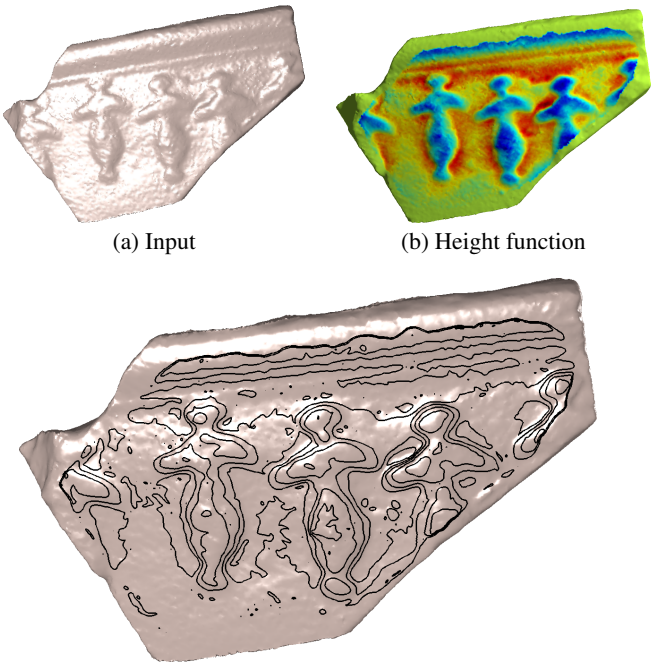
$$h = (S^T S)^{-1} S^T dh$$

and thus

$$(S^T S)h = S^T dh.$$

This equation is solved by the Conjugate Gradient method [Hestenes and Stiefel 1952], which is an iterative method that can be applied to sparse linear systems that are too large to be handled by direct methods. This solution is optimal in the mean squared error (MSE) sense.

After estimating the values of the height function on the vertices of the mesh, we use interpolation to find values between the vertices. Figure 5 illustrates the result of this step, with an additional filtering of sharp edges which will be described below in Section 7.



(a) Input (b) Height function

(c) Height contours

Figure 5: *Iso contours of the height function on a piece of a Hellenistic vase.*

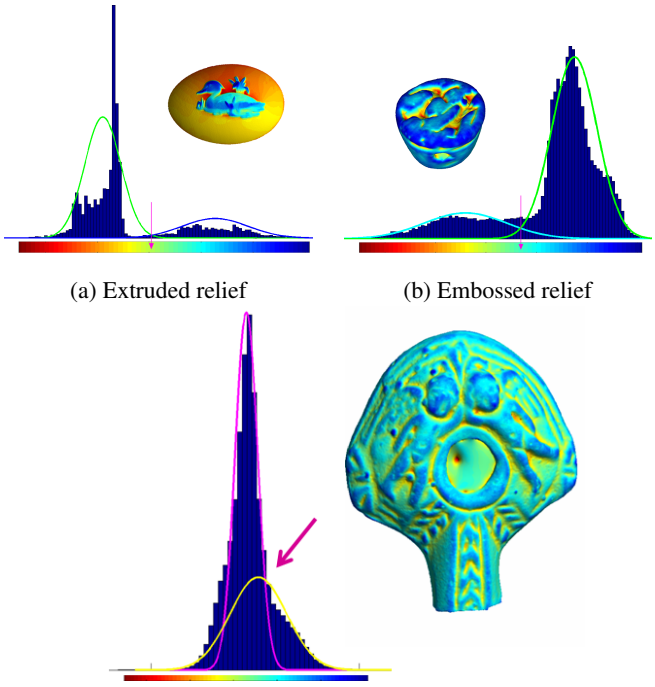
6 Threshold segmentation

Once the height of every vertex is determined, we aim to segment the mesh into two disjoint components: background $G \subset M$ and relief $R \subset M$ where $G \cup R = M$. This is performed using a threshold on the height function. We define the relief R as the union of all the vertices that are higher than a threshold θ above the base mesh: $R = \{v_i \in V_M | h(v_i) > \theta\}$ (or below for embossed reliefs). Analogously, we can write $G = \{v_i \in V_M | h(v_i) \le \theta\}$ or $G = M \setminus R$.

The threshold θ separating G and R could be found manually as an iso-contour on the mesh (Figure 5). However, we use the following heuristic for automatically determining θ. We examine the histogram of the height values and approximate it with a Gaussian mixture model (GMM) with two Gaussians $f = \sum_{i=1}^{2} \alpha_i G(\mu_i, \sigma_i)$, where μ_i and σ_i are the mean and standard deviation of the i-th component in the Gaussian mixture, α_i is its weight, and $\alpha_1 + \alpha_2 = 1$. The parameters are estimated using the Expectation Minimization (EM) method [Dempster et al. 1977].

We choose the threshold θ as the intersection of the two distributions: $\theta = \{h | G_1(h) = G_2(h)\}$. The relief part is chosen according to its type. It is the Gaussian with the larger mean value μ_i for extruded reliefs or the smaller mean value for embossed reliefs. In some examples, the histogram of values cannot be separated into two clearly distinct Gaussians, as the reliefs and base are mixed. Even in such cases, the intersection of the two distributions gives a good candidate for extracting the reliefs (Figure 6).

To increase the accuracy of the segmentation and achieve smoother boundaries between the relief and the background, the boundary curves cut through the mesh triangles. A mesh edge e_{ij} contains a curve boundary point if $h(v_i) < \theta$ and $h(v_j) > \theta$. The location of the curve point is obtained by linear interpolation of the vertices' height values on the edge. Consecutive neighboring curve points are connected on the faces of the mesh to create the curve itself.



(a) Extruded relief (b) Embossed relief

(c) Mixed reliefs

Figure 6: *Approximating the histogram of the height values using GMM to choose the relief threshold (in magenta). The relief extraction of these objects can be seen in Figures 1, 8, 11.*

7 Handling noise and sharp features

Scanned models may contain small bumps and scratches and also acquisition noise. All these may be classified as reliefs because their height deviates from the height of the base. Such phenomena is undesired in some applications. We solve this using a simple filtering scheme based on the segment's size and shape.

We decompose the obtained relief into k maximal connected components $\{c_i | i = 1..k\}$. For every segment c_i we calculate the following score:

$$\text{Score}(c_i) = \frac{N_{e-total}(c_i) - N_{e-boundary}(c_i)}{N_{e-total}(c_i)},$$

where $N_{e-total}(c_i)$ is the total number of edges in c_i and $N_{e-boundary}(c_i)$ is the number of its boundary edges. Small and skinny segments, containing many boundary edges, will receive lower scores and will be moved from the relief R to the background G. In our experiments we found that setting *score threshold* to 0.95 works for most models but the user may change the value depending on her preference.

Another possible misclassification that requires special attention occurs at sharp edges or corners of the model. First, sharp features receive high values in the height function due to our assumption that the base has a smaller curvature than the original surface. Hence, sharp edges need to be classified as such and removed from the relief R. Second, sharp edges may also cause problems in calculating the height of other vertices in the mesh, in the global solution of Equation (5). This is because vertices near the sharp edges are pulled towards higher values even if they are on a plane. To alleviate these problems, all vertices classified as situated on a sharp edges are removed from Equation (5) in effect creating a piecewise solution which is more robust.

To classify sharp edges we aim to find a coarse estimation of the base surface. The general idea is to apply a low pass filter to the mesh. While this estimation will not suffice for finding the reliefs, it is suitable for finding sharp edges on the model. This is because reliefs are located more on dome-like surfaces and tend to be more sensitive to filtering than edges of the model. After low pass filtering, areas with high mean curvature are classified as sharp features of the model. The algorithm consists of four steps, as illustrated in Figure 7.

1. Mesh simplification – We reduce the number of vertices in the model to a given threshold by applying simplification [Heckbert and Garland 1999]. This step also yields a more uniform density of the mesh. This contribute to making the filtering in the next step faster and more uniform.

2. Low pass filtering – We apply a strong low-pass filter on the mesh. The spatial frequencies of the relief are higher than the spatial frequencies of the edges. Therefore, reliefs are more sensitive to smoothing than edges. For the filtering we used 50 iterations of the improved Laplacian smoothing [Vollmer et al. 1999]. The results, however, are not very sensitive to the number of iterations.

3. Finding sharp features – After smoothing the mesh, we calculate the curvature of every vertex $k(v_i)$. Let BB be the diagonal of the model's bounding box and T_k be a threshold. Vertices with $k(v_i) \cdot BB > T_k$ are classified as sharp features of the model. A threshold of $T_k = 10$ works for all the models we experimented with.

4. Edge removal – vertices that were classified as sharp features in the previous step are removed from Equation (5). Note that this may cause a mesh to be separated into several unconnected components. However, since our technique works also for surfaces with boundary, this does not cause a problem, and we find the height function for each part separately. To synchronize the heights in the different components we constrain the height on the boundaries to be the same.

8 Applications

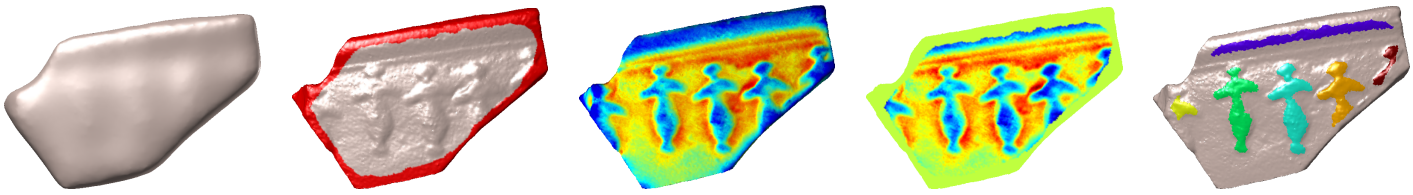
Our approach can be utilized not only for relief extraction, but also for other applications, including detail exaggeration and dampening, relief cut & paste, and curve drawing. This section presents some results.

8.1 Relief segmentation

Decorative reliefs are, and have been for thousands of years, added to real objects and CAD models to make the design more interesting. Reverse engineering of reliefs occurs in the porcelain industry, where decorative reliefs need to be extracted from old objects, in order to create new items that match the old design. Reverse engineering of reliefs is also performed in archaeology, where the reliefs serve as a "finger-print", used for comparing artifacts found in different archaeological sites. In both cases, a skilled artist needs to hand-draw the relief – a time-consuming, expensive, and biased process. Figures 8–11 illustrate some results, obtained by using our method in the different domains.

Figures 1, 7 and 8 illustrate challenging cases where our algorithm manages to extract the reliefs from noisy and weathered archaeological objects. In such examples both the base and the relief are noisy and details are lost due to aging. More examples of this type appear later.

Figures 9–10 demonstrate reverse engineering of CAD artificial



(a) Smoothed model (b) Found sharp edges (c) Height with filtering (d) Height with filtering (e) Reliefs extracted

Figure 7: *Filtering sharp features (a-b) creates a better height approximation for reliefs (c-e).*

Figure 8: *More examples of archaeological artifacts relief segmentation.*

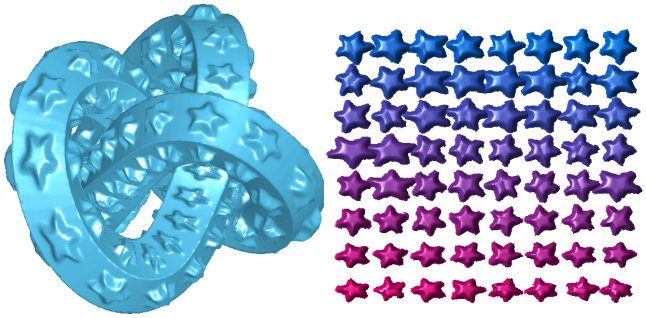


Figure 9: *Starknot – the stars are extracted from the knot and classified according to their size.*

objects, originally created by adding decorations to meshes. As can be seen, once these decorations are accurately extracted by our method, they can be classified (e.g. according to size), compared, and also removed from the original mesh.

Figure 11 shows the extraction of reliefs used in the porcelain industry. Our results are compared to those of [Liu et al. 2007a; Liu et al. 2007b] and demonstrated to be competitive. We, however, do not require the user to mark the initial position of snakes in order to extract the reliefs. Note that our method is capable of isolating non-relief portions of the surface, which are enclosed within the relief, while the method of [Liu et al. 2007b] fails to do so, due to the limitations of the snakes. This is visible, for instance in the surface between the bird's legs and the branch in Figure 11, right.

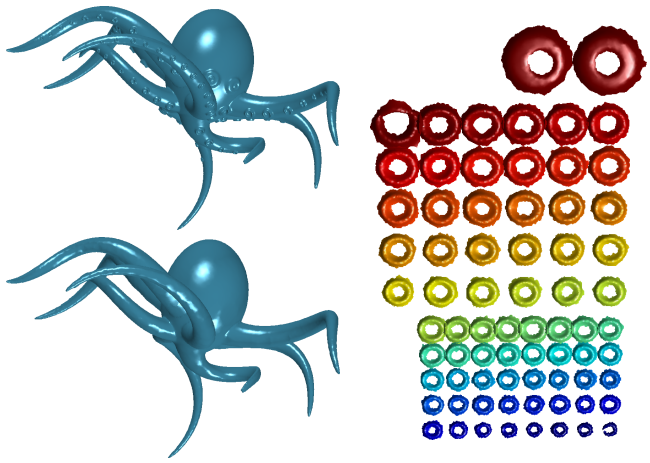


Figure 10: *Octopus – The reliefs (suction cups and eyes) are extracted and can then be removed from the Octopus's body (filling in the holes in the mesh).*

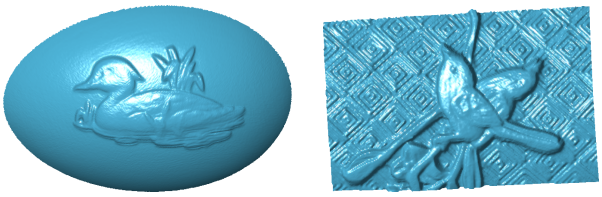
8.2 Detail exaggeration and dampening

Mesh editing has always been a fundamental problem in computer graphics [Sorkine et al. 2004; Yu et al. 2004]. Our method can be utilized to perform a specific task of shape editing – detail exaggeration and dampening, where the base remains unchanged. The key idea is simple. Once the height function is computed, one can control the details on the surface by manipulating their height through this function. Specifically, we modify the mesh by changing the coordinates of a vertex v using a parameter α as follows:

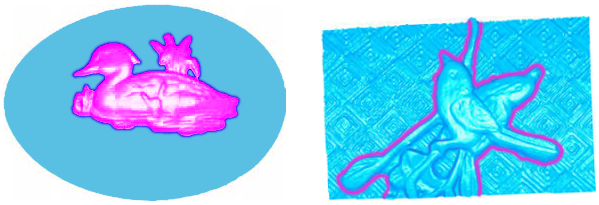
$$v(\alpha) = b_v + \alpha \cdot h(v)\vec{n}_B(b_v).$$

As a result, a relief vertex is exaggerated by alpha, whereas a background vertex, having a near-zero height, is hardly exaggerated.

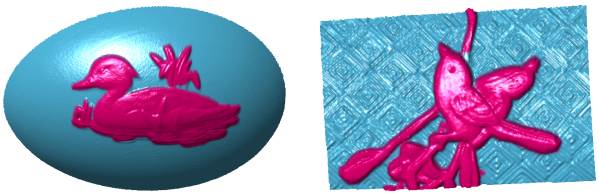
The advantage of this method, apart from simplicity, is that the mesh topology remains unchanged. Only the geometry (the coordinates of the vertices) are modified. Since this is very quick to perform once the height function is computed, it provides a way to experiment with a variety of heights. Global mesh manipulations can be created if we choose the same α for all vertices. Note that if the we choose $\alpha = 0$, the height is zero, and the details disappear all together. Some examples are demonstrated in Figures 12–14. Specifically, Figure 12 shows a synthetic example, while figures 13–14 show archaeological artifacts, where exaggeration or dampening help emphasize either the relief or the base surface, respectively. Some previous work such as [Rusinkiewicz et al. 2006] propose a local lighting model that can exaggerate details, but our method actually changes the geometry to accentuate the reliefs.



(a) Input



(b) [Liu et al. 2007a] [Liu et al. 2007b]



(c) Our result

Figure 11: *Left: A porcelain relief on a smooth background. Right: A porcelain relief on a textured background. Note how the relief is clearly separated from the background texture.*

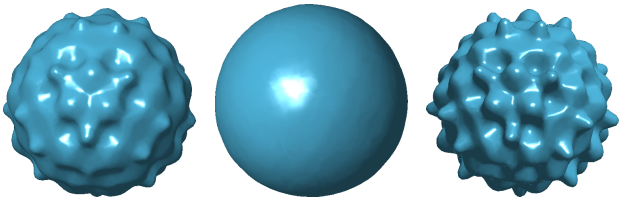


Figure 12: *A bumpy sphere whose details are dampened (middle) or exaggerated (right).*

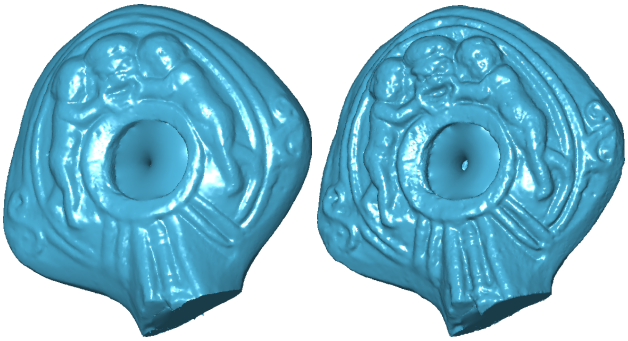


Figure 13: *A Hellenistic lamp (left) having its relief exaggerated (right) to accentuate the details.*

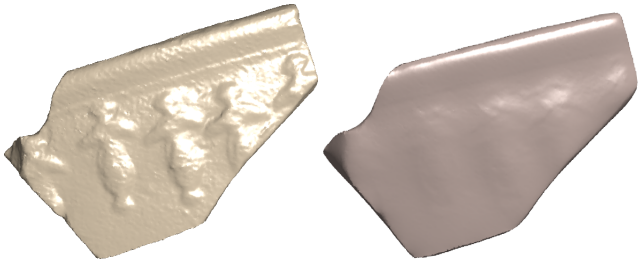


Figure 14: *A Hellenistic vase piece (left) having its relief dampened (right).*

8.3 Relief cut & paste

Copying and pasting sub-parts of models is an intriguing problem, which results in powerful and useful modeling tools [Funkhouser et al. 2004]. Our approach allows us to "cut & paste" reliefs and details, without the need to actually cut the surface and then seamlessly stitch sub-surfaces together – two highly challenging operations.

The key idea is to utilize the fact that reliefs can be expressed in terms of a height function. Thus, we can "glue" reliefs to other objects simply by transferring the height function of the relief, without having to modify the connectivity of the mesh. Instead, the vertices move according to the transferred height function, in the direction of the base's normals, very similar to bump mapping. This makes the process very quick and simple to compute.

Specifically, after computing the height function, we find a planar parameterization of both the source relief area and the target area of the object on to which we want to copy the relief. This is done using [Graphite 2009]. We merge the two parameterized domains to create a mesh compatible with both domains. Next, we interpolate the height values of the source relief to vertices of the target patch. Lastly, we move the corresponding target mesh vertices according to this newly calculated height function.

We can use this procedure to decorate models, to embed a logo onto an object, to copy details from one archaeological artifact to another, and more. For instance, in the porcelain industry, decorative reliefs can be extracted from old vases in order to create new items that match the old design. Figures 15–16 show some examples. Notice the robustness of our scheme – both the base and the relief are noisy, yet the results are pleasing.

8.4 Curve drawing

Curves drawn on objects convey prominent and meaningful information about the shape [DeCarlo et al. 2003; Yoshizawa et al. 2005; DeCarlo and Rusinkiewicz 2007; Judd et al. 2007; Kolomenkin et al. 2009]. They have been shown to be useful in a wide spectrum of applications. We propose a new type of curve, which incorporates real semantic meaning when the input is known to be a relief surface. In particular, in archaeological drawing, the reliefs are drawn by hand and printed on reports of excavators. The main purpose of these drawings is to aid the archaeologists to visualize the artifacts and compare them without actually holding them in their hands. In [Kolomenkin et al. 2008] it is proposed to replace the manual drawing with an automatic one. However, it is often the case that the curves generated appear broken.

Our approach can be utilized to draw prominent curves. They are defined as iso-contours of the height function $\{p \in M | h(p) = \theta\}$, whose value is the threshold used for extracting the reliefs (Sec-

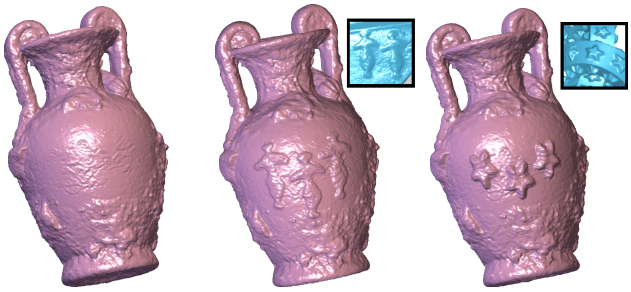


Figure 15: *Decorating a vase with the relief of the broken vase (Figure 5) and the stars knot (Figure 9). Note how the pasted reliefs contain the texture of the target surface naturally.*

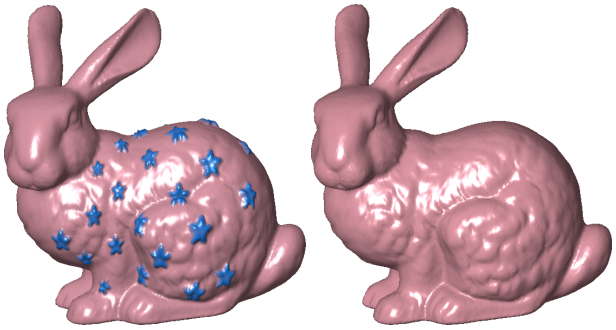


Figure 16: *Decorating the surface of the bunny with details from the stars knot model.*

tion 5). These curves present two benefits. First, noisy surfaces are handled well. Second, as iso-contours, the curves are guaranteed to be closed. Although these curves are not as appealing as some of the well-known NPR curves, certain applications, in particular shape analysis ones, may benefit from using such curves. Figure 17 illustrates that our curves nicely bound the features (such as the letters on the coin) and generates more informative drawings than the alternatives – relief edges using the method of [Kolomenkin et al. 2009] or ridges & valleys [Yoshizawa et al. 2005].

9 Discussion

Running times: Our algorithm is implemented in Matlab and C and our experiments were executed on a 1.6Ghz Pentium 2-processor machine with 2Gb of memory. For a model of $60K$ vertices, it takes about 16 seconds to extract the reliefs, out of which 14 seconds are devoted for the normal estimation. Hence, if interactive refinement is needed, most of the computation can be done in a preprocessing stage and stored.

Limitations: There are some types of reliefs that cannot be expressed as a base and a height function and hence might not be extracted by our method. Though this does not happen often for reliefs in archaeology, such models can be generated by CAD software. In addition, if the texturing of the background is too deep compared to the height of the relief, the algorithm might classify high areas of the textured background as relief. Other types of reliefs that are not handled by our algorithm are hierarchical reliefs, i.e., reliefs that can themselves contain reliefs of smaller scales. In this case the algorithm can be adapted by running it recursively on the relief part.

Another limitation of our method is that if the relief is adjacent to a sharp edge, we may classify part of the relief as a sharp edge and remove it. This can be seen in Figure 7, where near the broken edge of the piece, some parts of the dancers are classified as sharp edges. Conversely, if the reliefs is mixed with sharp features, these features might be classified erroneously. Finally, although automatic selection of thresholds worked in most cases, it is still a heuristic and further investigation in this area is needed.

10 Conclusions

This paper presented a method for extracting reliefs and details from relief surfaces. The method is based on defining the relief as a height function over a base surface. Though the decoupling into a base and a relief is unknown, we have shown that this function can nevertheless be computed by using normal estimation of the base surface. We also demonstrated how the method can benefit several applications, including relief segmentation, detail exaggeration and dampening, detail copy & paste, and curve drawing on meshes.

In the future we would like to examine more the automatic selection of threshold values for various cases. It could also be possible to build a GMM with more than two Gaussians and have several levels of reliefs. Other possible applications that could benefit from relief extraction are model editing where first the reliefs are removed, then the object undergoes some modification, after which the reliefs are returned. More generally, the separation of details from a background surface is semantically correct and can assist reverse engineering and complex editing operations.

Acknowledgements

We would like to thank Ralph Martin for sharing his models and results. This research was supported in part by the Israel Science Foundation (ISF) 628/08 and 315/07, the Goldbers Fund for Electronics Research, the Olendorff Foundation and the Israel Ministry of Science grant 3-3421.

References

ANDERSON, S., AND LEVOY, M. 2002. Unwrapping and visualizing cuneiform tablets. *IEEE Computer Graphics and Applications*, 82–88.

ATTENE, M., KATZ, S., MORTARA, M., PATANE, G., SPAGNUOLO, M., AND A.TAL. 2006. Mesh segmentation - a comparative study. In *Shape Modeling International (SMI)*, 7–7.

CHEN, X., GOLOVINSKIY, A., AND FUNKHOUSER, T. 2009. A benchmark for 3d mesh segmentation. *ACM Trans. Graph. 28*, 3.

DECARLO, D., AND RUSINKIEWICZ, S. 2007. Highlight lines for conveying shape. In *NPAR*, 63–70.

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

DEMPSTER, A., LAIRD, N., AND RUBIN, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological) 39*, 1, 1–38.

FUNKHOUSER, T., KAZHDAN, M., SHILANE, P., MIN, P., KIEFER, W., TAL, A., RUSINKIEWICZ, S., AND DOBKIN, D. 2004. Modeling by example. *ACM Trans. Graph. 23*, 3, 652–663.

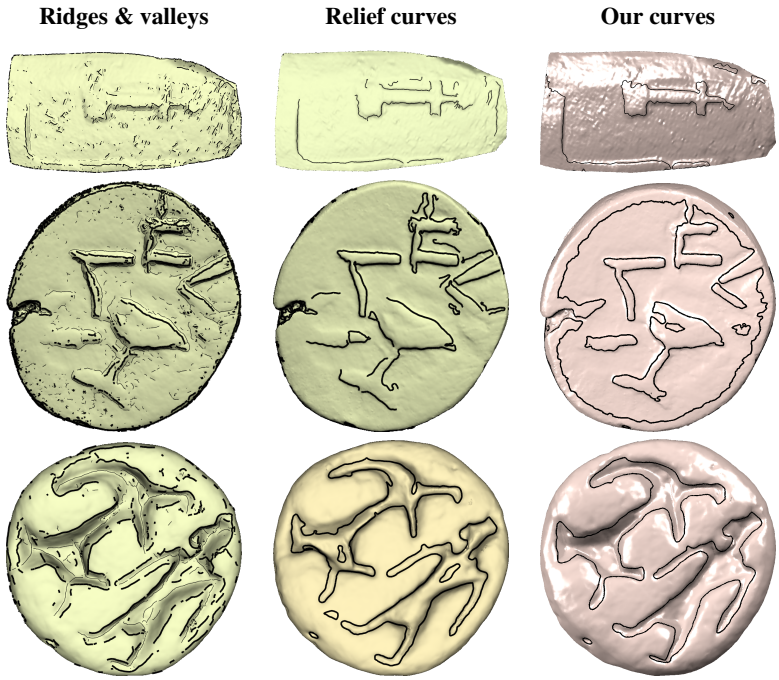


Figure 17: *Comparison of different methods for curve drawing on noisy and weathered input: Top – Hellenistic stamped amphora handle from the first century BCE. Middle – a coin from second century AD. Bottom – a seal from the early Iron Age (also seen in Figure 1). Although the compared results could possibly be optimized using more parameters tuning, our curves are very simply to obtain as the boundary of the extracted reliefs and are guaranteed to be a closed curve.*

GRAPHITE. 2009. *Graphite research platform for computer graphics user manual*. ALICE project-team NRIA Nancy Grand-Est / Loria, http://alice.loria.fr/WIKI/index.php/Graphite/Graphite.

HECKBERT, P., AND GARLAND, M. 1999. Optimal triangulation and quadric-based surface simplification. *Journal of Computational Geometry: Theory and Applications 14*, 1-3, 49–65.

HESTENES, M., AND STIEFEL, E. 1952. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards 49*, 6, 409–436.

JUDD, T., DURAND, F., AND ADELSON, E. 2007. Apparent ridges for line drawing. *ACM Trans. Graph. 26*, 3, 19:1–19:7.

KOLOMENKIN, M., SHIMSHONI, I., AND TAL, A. 2008. Demarcating curves for shape illustration. *ACM Transactions on Graphics 27*, 5, 157:1–9.

KOLOMENKIN, M., SHIMSHONI, I., AND TAL, A. 2009. On edge detection on surfaces. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

LIU, S., MARTIN, R., LANGBEIN, F., AND ROSIN, P. 2006. Segmenting reliefs on triangle meshes. *ACM Symposiam Solid and Physical Modeling*, 7–16.

LIU, S., MARTIN, R., LANGBEIN, F., AND ROSIN, P. 2007. Background surface estimation for reverse engineering of reliefs. *International Journal of CAD/CAM 7*.

LIU, S., MARTIN, R., LANGBEIN, F., AND ROSIN, P. 2007. Segmenting geometric reliefs from textured background surfaces. *Computer-Aided Design and Applications*, 565–583.

LIU, S., MARTIN, R., LANGBEIN, F., AND ROSIN, P. 2007. Segmenting periodic reliefs on triangle meshes. In *Math. of Surfaces XII*, 290–306.

MANGAN, A., AND WHITAKER, R. 1999. Partitioning 3d surface meshes using watershed segmentation. *IEEE Transactionson Visualization and Computer Graphics 5*, 4, 308–321.

OHTAKE, Y., BELYAEV, A., AND SEIDEL, H. 2002. Mesh smoothing by adaptive and anisotropic gaussian filter. In *Vision, Modeling, and Visualization*, 203–210.

RUSINKIEWICZ, S., BURNS, M., AND DECARLO, D. 2006. Exaggerated shading for depicting shape and detail. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3, 1199–1205.

SHAMIR, A. 2008. A survey on mesh segmentation techniques. *Computer Graphics Forum 27*, 6, 1539–1556.

SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., AND SEIDEL, H. 2004. Laplacian surface editing. In *Symposium on Geometry processing*, 179–188.

VOLLMER, J., MENCEL, R., AND MULLER, H. 1999. Improved Laplacian smoothing of noisy surface meshes. In *EuroGraphics*, 131–138.

YOSHIZAWA, S., BELYAEV, A., AND SEIDEL, H. P. 2005. Fast and robust detection of crest lines on meshes. In *ACM Symposium on Solid and Physical Modeling*, 227–232.

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