Surface Relief Analysis for Illustrative Shading

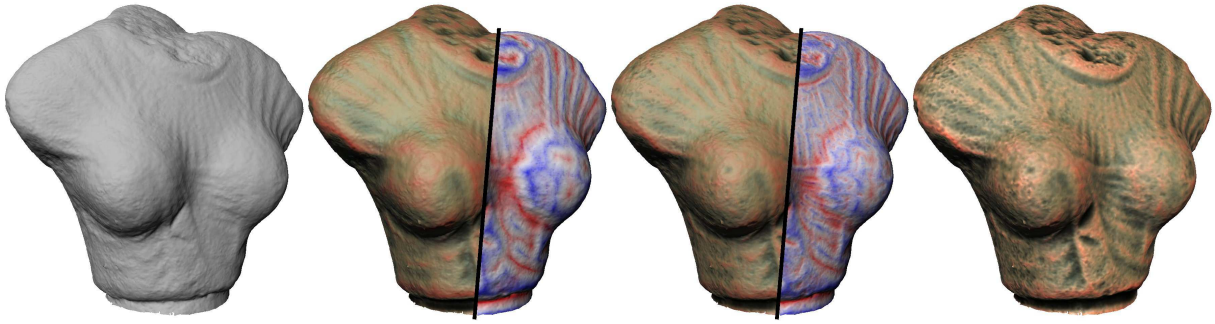
Lucas Ammann, Pascal Barla, Xavier Granier, Gael Guennebaud, Patrick Reuter

Eurographics Symposium on Rendering 2012
Fredo Durand and Diego Gutierrez
(Guest Editors)

Volume 31 (2012), Number 4

Surface Relief Analysis for Illustrative Shading

Lucas Ammann†, Pascal Barla, Gaël Guennebaud, Xavier Granier, Patrick Reuter
Inria - Univ. Bordeaux - IOGS - CNRS



(a) Diffuse shading (b) Coarse scale only (c) All scales (d) + Radiance scaling

Figure 1: ***Feature-based shading:*** *Given a detailed surface (a), we analyze its relief to locate relief features in the neighborhood of each surface point (b). We focus on three types of features: convexities, concavities, and inflexions, shown on the right half with blue, red and white colors respectively. Extracted information is used to assign them different shading functions: here we use three different lit-spheres, shown on the left half. An additional accessibility shading effect helps convey relief cavities. Features are extracted and combined at multiple scales to depict relevant relief details (c). Finally, radiance scaling is added to enhance the relief based on the curvature at each feature (d).*

Abstract

In this paper, we present an analysis technique that leverages the complexity found in detailed 3D models for illustrative shading purposes. Given a smooth base surface with relief, it locates relief features (concavities, convexities and inflections) around each surface point and at multiple scales, using cubic-polynomial fitting. This object-space, per-vertex information is then used to guide a variety of shading techniques including normal enhancement, feature visualization, accessibility shading and radiance scaling. Thanks to this approach, features at multiple scales are easily combined, filtered and shaded, allowing users to explore surface relief in real-time.

1. Introduction

During the last decade, the bulk of computer graphics applications has made use of 3D objects exhibiting an ever-increasing amount of shape details. Such geometric complexity may either come from densely scanned real-world objects, or it may be created by artists in modern digital modeling and sculpting software. In both cases, there is a growing need for shading tools that help visualize, enhance, or exaggerate complex surface relief patterns automatically.

This need firstly arises in scientific illustration domains. In Cultural Heritage, example applications include the study of detailed inscriptions found in engraved stones or of shallow traces over human bones. In Cartography, mountain relief is acquired with an ever-increasing resolution, which then requires specific coloring and shading techniques for a proper depiction. The same requirement has recently emerged with manually-created geometry. Artists nowadays spend a lot of time using modeling and sculpting software to work out fine shape details such as skin pores, wrinkles or grain. Surface shading techniques that properly depict this geometric complexity without requiring additional manual effort are then necessary.

In this paper, we consider *relief* in a standard way as a height field relative to a smooth (and not necessarily planar) base surface. This is a reasonable assumption both in scientific applications where surface details are often very shallow (as in Archeology) or naturally defined as height data (as in Cartography), and in artistic applications where most surface details are stored in displacement maps.

A relief *feature* is then meant to be a region on the base surface where the relief is prominent. In particular, we are in-

† {ammann,barla,guenneba,granier,reuter}@labri.fr

terested in three types of features — convexities, inflections and concavities — and the ways these are nested at multiple scales. Take for instance a relief ridge pitted with small cavities, as shown on the shoulder of the statue in Figure 1(a): a point anywhere inside one of the small pits belongs to a concave feature at a small scale, and to a convex feature at a larger scale. Conveying surface relief through illustrative shading then amounts to assigning a proper shading value to each surface point, in a way that depends on the multi-scale features it belongs to. This raises two major issues: 1) how to locate relief features in the neighborhood of a surface point? 2) how to combine features found at various scales (i.e., at increasing neighborhood sizes)?

We address these issues by introducing an efficient bottom-up, multi-scale relief analysis technique based on cubic polynomial fitting. It locates relief features (concavities, convexities and inflections) in a neighborhood *around* each surface point, and provides a straightforward solution to the multi-scale combination problem. We illustrate the benefits of our approach by adapting shading techniques to take advantage of *non-local* relief information available at each surface point. An example is shown in Figure 1(b-d), where shading color is assigned based on the distance to the closest feature center, first at a single (large) scale, then at multiple scales. The result is a far more legible depiction of relief features occuring at various scales (including the small pits on the shoulder). Our system only requires a short pre-process for the completion of the bottom-up analysis step. It then allows users to explore surface relief in real-time, by varying the contribution of each scale, or by modifying shading and filtering parameters.

2. Previous work

One of the earliest shading methods explicitly designed to convey shape features is *accessibility shading* [Mil94]. Together with its most popular variant *ambient occlusion* (e.g., [PG04]), these methods produce images that depict cavities by darkening hardly-accessible surface points. However, they have two main drawbacks. First, they demand long pre-processing times when done accurately in object space. But most importantly, they provide limited control in terms of surface feature depiction: shading and analysis are not separated and mainly convey deep concavities with very smooth shading variations.

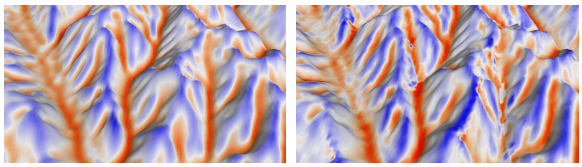
More accurate surface measurements are obtained with differential geometry operators [dC76]. In particular, the second- and third-order tensors provide information about curvature and curvature variations at a surface point. Such measurements only apply to small surface neighborhoods though (infinitesimal neighborhoods for smooth surfaces). They have thus traditionally been used for identifying surface points that exactly lie on extremal curves such as *ridges and valleys* [OBS04] or *demarcating curves* [KST08]. Some methods have used similar operators for larger surface neighborhoods (e.g., [CP05, CPG09]) by first fitting a local plane to the surface, and then fitting a quadratic function on relative height values. When the surface "folds-over" itself, complex issues arise that can be overcome using total least-square fits [GGG08]. Such analysis methods are limited to purely local surface feature measurements, while we are interested in locating salient relief features, which we recall are to be found in the *neighborhood* of each surface point. In particular, these methods provide no simple way to know the distance to the nearest feature center of a given type, to combine features found at multiple scales, or to filter out less prominent features (e.g., shallow relief).

Another approach for analysing relief at multiple scales consists in decomposing surface normals into different layers, which also avoids fold-over issues. Such a decomposition has first been applied to *normal enhancement* [CST05, ZCF*10], where a single relief layer is manipulated to exaggerate or attenuate surface details through shading. Similar enhancement techniques have been applied to *polynomial texture maps* [MWGA06]. In both cases, only two scales are considered. These methods have been extended to multiple scales in the *exaggerated shading* technique [RBD06]. Different levels of smoothed normals are used to align a single light source at grazing angles. Shading values are then computed at each scale through half-Lambertian shading, and combined with a weighted sum to exaggerate multi-scale surface relief. The drawback with this approach is that scale combination depends on the choice of light direction and shading parameters, which not only produces artifacts when the light is moved around, but also makes it difficult to control which type of feature is depicted.

Normal variations in image space have also been used to convey surface details by letting them drive variations of incoming radiance (e.g., [VPB*10b]). This is an improvement compared to exaggerated shading, since it works for arbitrary materials and illumination and is devoid of temporal artifacts. However, the method is confined to a single scale per point, and provides limited control over the type of depicted feature as before.

When an accurate depiction of surface relief is targeted, it is preferable to decompose surface *geometry* into base and relief layers. Various methods have been proposed to perform this decomposition (e.g., [ZTS09]). The goal of this paper is *not* to present a novel decomposition though, but rather to provide analysis solutions specifically targeted to the relief layer. Only a few existing techniques have tackled this problem. The *prominent field* technique [KST09b] combines output of second- and third-order tensors to identify a direction field along which prominent relief features are likely to be located. This direction field has proven to be useful for feature-aware smoothing, curvature-based shading, and more recently line-based rendering [KST09a]. A similar solution has been proposed to identify feature lines in image-space, using fitting techniques [VVC*11]. The advantage of using a direction field is that it locates more accurately surface features around a point of interest. Unfortu-

nately, as depicted in Figure 2 using a *single* direction also introduces artifacts around field singularities, and since the field is likely to change at different scales, it is not clear how to perform scale combination.



(a) Mean curvature (b) Directional curvature

Figure 2: ***Artifacts due to singularities****. As opposed to isotropic measurements (a), the use of a directional field to analyse relief produces artifacts at it singularities (b).*

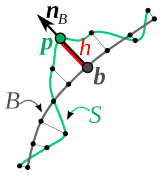
3. Overview

Our approach focuses on surface relief and assumes input geometry to be decomposed into base and relief layers. We first locate relief features in the neighborhood of a surface point, at a number of scales and directions. This is done thanks to cubic polynomial fitting as explained in Section 4.1. We then combine features together and filter them in a way that preserves only the most pertinent data. This is permited by a weighting scheme based on the information conveyed by individual fits, as explained in Section 4.2. Shading techniques are finally adapted to depict properties of the relief features found around each surface point, as described in Section 5.

Details of our GPU implementation are given in Section 6: it requires a short pre-process for analysis, after which it runs in real-time with interactive scale combination, filtering and shading abilities. Our solution does not rely on any parametrization of the base surface, and we demonstrate our system on both height-fields and detailed 2D manifolds. Moreover, for dynamic scenes, there is no need to update the analysis during animation since in most cases, only the base surface is deformed.

4. Relief analysis

Our analysis takes as input a surface S defined as a smooth base surface B displaced along its normal field $\boldsymbol{n}_B$ by a height function h. Mathematically, any point $\boldsymbol{b} \in B$ yields a point $\boldsymbol{p}$ on the surface S with:



$$\boldsymbol{p} \;=\; \boldsymbol{b} + h(\boldsymbol{b})\boldsymbol{n}_B(\boldsymbol{b})\,.$$

The scalar function h corresponds to the *relief* of the surface. For instance, sculpting tools are able to directly provide a base surface with a displacement map. With acquired objects though, the base and relief layers must be extracted in pre-process. The presentation of this step is *out of the scope* of our paper, and we refer to, e.g., Zatzarinni et al. [ZTS09] for such a base/relief decomposition technique. For the sake of simplicity, and without loss of generality, we assume that B is provided as a dense (regular or irregular) triangular mesh, while h is given as a scalar value per vertex.

4.1. 1D Analysis

The key idea of our approach is to analyse relief along 1D neighborhoods of the base surface, in multiple directions around each point $\boldsymbol{b}$. The main reasons for this choice are that 1) the localisation of relief feature centers (e.g., curvature extrema) around $\boldsymbol{b}$ is made considerably simpler in one dimension, and 2) no 2D parametrization is required. This strategy is also significantly faster for large scales.

In practice, we perform the analysis at mesh vertices, in multiple directions θ_i uniformly distributed in the $[0,\pi[$ range. We define $\boldsymbol{t}_i = (\cos\theta_i, \sin\theta_i)$ as the corresponding direction vector that lies in the tangent plane of the base surface. A 1D neighborhood is obtained as the intersection of the base surface with the plane spanned by $\boldsymbol{n}_B$ and $\boldsymbol{t}_i$, and going through $\boldsymbol{b}$ (see Figure 3(a)). The reference direction $\theta_0 = 0$ can be chosen arbitrarily for each point $\boldsymbol{b}$ without any impact on quality because, in our approach, we combine multipe directions to compute a single feature property.

For each direction, this intersection yields a *base polygonal parametric curve* $\boldsymbol{b}_i$ which is parametrized in term of arc-length on the base surface. Every $\boldsymbol{b}_i(t)$ corresponds to a point on the base surface at a distance t along the polygonal curve, with associated height value $h(\boldsymbol{b}_i(t)) = h \circ \boldsymbol{b}_i(t)$.

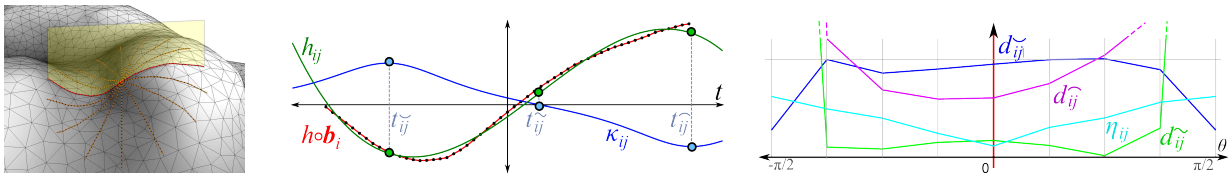
Polynomial fitting The 1D relief function $h \circ \boldsymbol{b}_i$ may contain a variety of features at multiple scales. In order to identify them, we approximate $h \circ \boldsymbol{b}_i$ by a set of cubic polynomials fitted for multiple support sizes s_j. Higher values of s_j correspond to coarser scales. Cubic-polynomials are particularly well suited to our purpose since they exactly exhibit a pair of convex and concave regions separated by an inflection. It permits to locate and characterize such neighboring features analytically.

A polynomial is given by $h_{ij}(t) = u_0 + u_1 t + u_2 t^2 + u_3 t^3$. As illustrated in Figure 3(b), h_{ij} is computed by a standard least square minimization applied to a finite set of samples t_k uniformly spread in the range $[-s_j/2, s_j/2]$:

$$h_{ij} \;=\; \arg\min_P \sum_k \left(P(t_k) - h(\boldsymbol{b}_i(t_k))\right)^2 . \qquad (1)$$

Details on how to efficiently solve these minimization problems are given in Section 6, where we show how to take advantage of sampling regularity and incremental calculus.

Feature extraction One of the main reasons for making use of an analytic representation is that it permits to identify entire relief features. The fitted cubic polynomial provides us with analytic expressions of the first three order derivatives



(a) Neighborhood sampling (b) Polynomial fitting at point p in 1 direction (c) Error measures at point p in all directions

Figure 3: ***1D analysis.*** *(a) The curves $\boldsymbol{b}_i$ on the detailed surface S are extracted in multiple directions by intersecting planes defined relative to the base surface B. (b) The height along a curve $h \circ \boldsymbol{b}_i$ on a support of size s_j is fitted with a cubic polynomial h_{ij} from which we compute the curvature function κ_{ij} and feature center locations $t_{ij}^{\star}$ ($\star \in \{\frown, \smile, \sim\}$) analytically. (c) For each direction, we compute the distance $d_{ij}^{\star}$ to a particular feature center and the normal deviation η_{ij} of the detailed surface.*

h'_{ij}, h''_{ij}, and h'''_{ij}, as well as the curvature κ_{ij}:

$$\kappa_{ij}(t) = \frac{h''_{ij}(t)}{(1+h'_{ij}(t)^2)^{\frac{3}{2}}}. \tag{2}$$

We use these differential quantities to identify the positions of the convexity ($t_{ij}^{\frown}$), concavity ($t_{ij}^{\smile}$), and inflexion ($t_{ij}^{\sim}$) points. We define the first two as the locations of the curvature extrema which are obtained as the zero-crossings of the curvature variation. More precisely, $t_{ij}^{\frown}$ and $t_{ij}^{\smile}$ are the roots of the rational polynomial $\kappa'_{ij}(t)$ for which explicit formulas are given in the Appendix. We observe that these roots do not coincide with the local extrema of the cubic defined as the zero-crossings of the first-order derivative h'_{ij}, even though they are often very close (see Figure 3(b)). This choice is particularly important when the cubic is monotone but not linear: curvature extrema can still be found whereas the value extrema do not exist. Curvature extrema thus constitute a more robust choice in general. The position of the inflexion point $t_{ij}^{\sim}$ is given by the zero-crossing of the second-order derivative ($P''_{ij}(t_{ij}^{\sim}) = 0$).

Normalized distances and error measure Features extracted from the polynomial approximation may not always be *pertinent*. First, specific feature locations may be identified outside the support used to fit the relief data. We thus compute a truncated and normalized distance for each type of relief feature ($\star \in \{\frown, \smile, \sim\}$):

$$d_{ij}^{\star} = \left\lfloor \frac{2|t_{ij}^{\star}|}{s_j} \right\rceil, \tag{3}$$

where $\lfloor x \rceil$ is a function that clamps x to the $[0,1]$ range.

Second, the detailed surface normals may substantially deviate from the plane used to intersect the base surface geometry. Indeed, with a large normal deviation, a small change in direction implies a large change in the fitted heights. We take into account this source of error by measuring for each fit the average normal deviation η_{ij}:

$$\eta_{ij} = \frac{1}{s_j} \sum_k |\boldsymbol{n}(t_k) \cdot (\boldsymbol{t}_i \times \boldsymbol{n}_B)|, \tag{4}$$

where $\boldsymbol{n}$ is the normal of the detailed surface S, and $\boldsymbol{t}_i \times \boldsymbol{n}_B$ is the normal of the plane supporting the cubic. Both types of error measures are visualized in Figure 3(c) as a function of direction θ_i at a single scale j.

4.2. Combination and Filtering

Our multi-scale and multi-direction analysis provides a dense relief description around a surface point, which ensures that most nearby relevant relief features are properly identified by at least one fitted polynomial. We now present combination and filtering mechanisms that make such information exploitable by subsequent shading techniques.

Feature combination The key idea of our approach is to combine scalar feature values f_{ij} instead of polynomials themselves, to yield a single scalar F. For instance f_{ij} could be taken to be the convexity curvature $\kappa_{ij}(t_{i,j}^{\frown})$, yielding an average curvature K. Combination is done in two steps. We first average feature values at each scale over all directions:

$$F_j = \sum_i \widehat{w}_{ij} f_{ij}, \quad \widehat{w}_{ij} = \frac{w_{ij}}{\sum_{i'} w_{i'j}}, \tag{5}$$

using $w_{ij} = (1-\eta_{ij})(1-d_{ij}^{\star})$ for the weights. The factor $1-\eta_{ij}$ favors directions of low normal deviation for which extracted feature values are most pertinent. This may be seen as a generalization of principal curvature directions, since for a quadratic surface, they exactly correspond to the directions of zero normal deviation. For choices of f_{ij} involving either $t_{ij}^{\frown}$, $t_{ij}^{\smile}$ or $t_{ij}^{\sim}$, the weights $1-d_{ij}^{\star}$ also decrease as the

(a) Algebraic fit (b) Using $\kappa_{i,j}(0)$ (c) Using $\kappa_{i,j}(t_{i,j}^{\circ})$

Figure 4: ***Curvature comparisons.*** *The mean curvature obtained from the least square fit of an algebraic sphere (a) yields similar results than averaging our curvature $\kappa_{i,j}(0)$ at the current point (b). On the other hand, averaging the curvature of the closest convexity or concavity ($\kappa_{i,j}(t_{i,j}^{\circ})$) exhibits a much more segmented curvature measurement (c).*

(a) Coarse scale only (b) Fine scale only (c) All scales (ours) (d) All scales (uniform)

Figure 5: ***Scale combinations****: Diffuse shading is darkened proportional to the distance to curvature inflections, in (a) with only a coarse scale, in (b) using only a fine scale, and in (c) and (d) combining all* 3 *scales together using our adaptive weights and uniform weights respectively. The use of a single scale fails to convey all the relief features in a legible way.*

feature location approaches the limit of the support. In practice, any decreasing functions of η_{ij} and $d^{\star}_{ij}$ could be used.

The F_j from Equation 5 are then averaged over scales:

$$F = \sum_j \alpha_j \widehat{W}_j F_j, \quad \widehat{W}_j = \frac{W_j}{\sum_{j'} W_{j'}}. \tag{6}$$

with $W_j = \sum_i w_{ij}$. This weighting provides for a natural balance between features at each scale: the influence of a given scale depends on the average pertinence of the angular analysis at this scale. The parameter α_j is introduced for user-controlled filtering. By default it is set to 1, and its adjustment will be discussed below.

Figure 4 illustrates our approach at individual scales (Equation 5). If we take $f_{ij} := \kappa_{ij}(0)$, it boils down to a local estimation of mean curvature. Like total least square fits [GGG08], it permits to avoid fold-over issues at large scales (we use 8 directions for each scale). However, the main benefit of our approach is in estimating properties of features in the neighborhood of each surface point. To illustrate this, we choose $f_{ij} := \kappa_{ij}(t^{\circ}_{i,j})$, which corresponds to the curvature of the closest relevant feature, i.e., the closest convexity ($t^{\circ}_{i,j} = t_{\widehat{ij}}$) or concavity ($t^{\circ}_{i,j} = t_{\widetilde{ij}}$). It results in a much more segmented measurement of curvature, each point displaying the curvature of the convexity it belongs to.

Figure 5 illustrates our approach at multiple scales (Equation 6). Here we use another feature value: $f_{ij} := 1 - d^{\sim}_{ij}$, the distance to the closest inflection point. Contrary to exaggerated shading [RBD06], our scale combination is independent of the choice of lighting or shading model. More importantly, our blending weights (the $\widehat{W}_j$ in Equation 6) are locally adjusted over the object surface based on fitting pertinence, while their are based solely on scale, thus leading to a more legible end-result.

User-controlled filtering Users may desire to select specific features during the combination process; this is made possible through the α_j parameter introduced in Equation 6. In practice, α_j decreases the contribution of the scale s_j relative to the other scales and to a fall-back value F_B. This is done thanks to a simple linear interpolation:

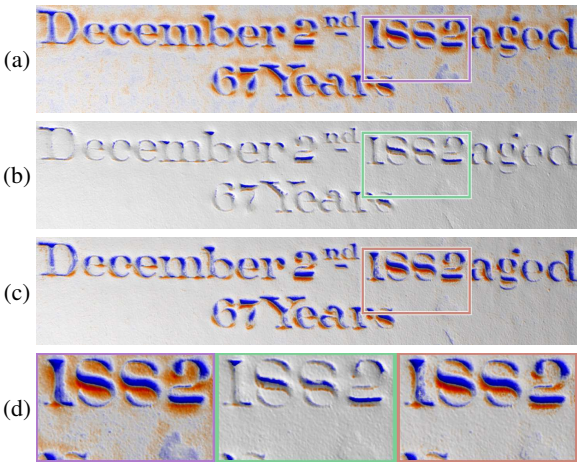
$$F_\alpha = \mathrm{lerp}(F_B, F, \alpha) = F_B + (F - F_B)\alpha, \tag{7}$$

where α is the average of all the α_j. In other words, this procedure ensures that in regions where none, or little, of the user-selected features are found, we fall back to a "background" value F_B.

The choice of α_j depends on the kind of features the user wants to preserve. We suggest the following *feature filter*:

$$\alpha_j = \psi\left(\operatorname*{std}_i(\eta_{ij}) \sum_i \widehat{w}_{ij}\kappa_{ij}(t_{\widehat{ij}})\right), \tag{8}$$

where ψ is a user controlled sigmoid function allowing for a smooth transition between the selected and rejected parts. In this paper we used a cubic *smooth step* function. The first factor retains anisotropic relief features defined as the standard deviation of the normal deviation η_{ij} over the set of directions. The second factor selects regions for which the closest relevant feature has a high mean curvature. Because we make use of non-local curvature measurements, entire



(a) (b) (c) (d)

Figure 6: ***Feature filtering****. In (a), surface relief height is displayed with a red (lowest) to blue (highest) gradient (i.e., we use $f_{ij} := h_{ij}(0)$ in Equation 5). It is filtered based on (b) the average curvature at the evaluation point ($\alpha_j = \psi_\kappa(\sum_i \widehat{w}_{ij}\kappa_{ij}(0))$, and (c) our* feature filter*. The zoom insets (d) shows how our non-local criteria better preserves feature integrity.*

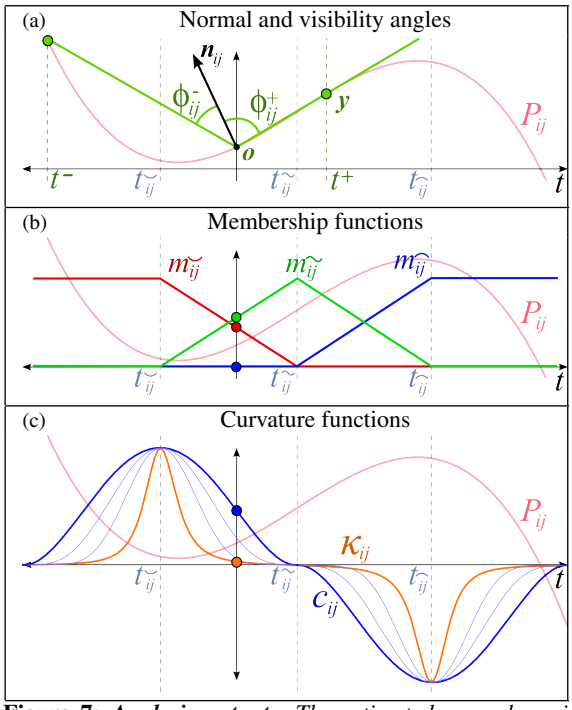


Figure 7: ***Analysis outputs****. The estimated normal $\boldsymbol{n}_{ij}$ is used both for enhancement, and for computing visibility angles $\phi_{ij}^{\pm}$ for accessibility shading. The membership functions $m_{ij}^{\star}$ (in red, green and blue for concavities, inflections and convexities) serve to identify features for direct visualization. The curvature function c_{ij} (in blue) is used instead of κ_{ij} (in orange) as it offers better user control. The thin blue lines represent variations of the exponent γ.*

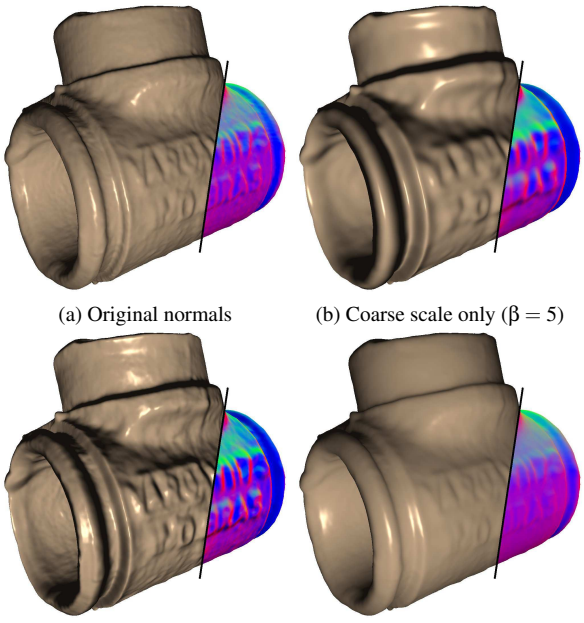
features are selected even if filtering is applied per surface point. As illustrated in Figure 6, it preserves much better relief structure compared to filtering based on local mean curvature, which tends to cut-off relief features. In our experience, this filter works remarkably well to select engraved symbols.

5. Shading techniques

In the previous section, extracted relief features have been displayed with simple colors for clarity of exposition. We now explain how more complex illustrative shading techniques are adapted to take advantage of our analysis. Thus, all subsequent figures present variants of our approach. In all results, we use 8 directions and between 3 and 5 scales.

5.1. Normal enhancement

The availability of relief normals is necessary for all shading techniques regardless of the target application. However, using the original relief normal raises coherence issues whenever filtering is activated (i.e., $\alpha < 1$) . Our solution consists in estimating an averaged normal $\boldsymbol{N}$ alongside the computation of F. We compute one local normal $\boldsymbol{n}_{ij}$ for each fitted polynomial h_{ij} (see Figure 7(a)) and transform them back to



(a) Original normals (b) Coarse scale only ($\beta = 5$)

(c) All scales ($\beta = 5$) (d) All scales ($\beta = 1$)

Figure 8: ***Normal enhancement:*** *The original normals of an object (a) are enhanced at a coarse scale (b), then at all scales (c-d) using different enhancement parameter values.*

world space. The coordinates of $\boldsymbol{n}_{ij}$ in the plane spanned by h_{ij} are directly given by:

$$\boldsymbol{n}_{ij} = \frac{(-u_1, 1)}{\|(-u_1, 1)\|}, \tag{9}$$

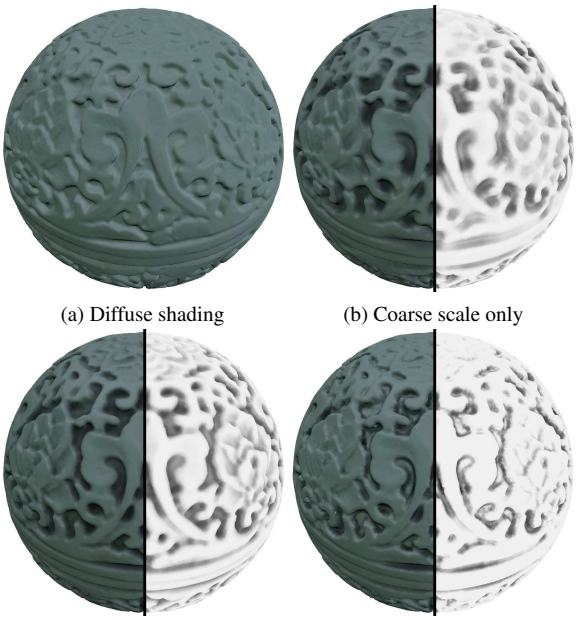
where u_1 is the linear coefficient of h_{ij}.

The averaged normal $\boldsymbol{N}$ is then obtained by applying Equation 6 to each coordinate of $\boldsymbol{n}_{ij}$ in world-space and then renormalizing. The final filtered normal is computed via spherical interpolation: $\boldsymbol{N}_\alpha = \text{slerp}(\boldsymbol{n}_B, \boldsymbol{N}, \beta\alpha)$, where $\beta \in \mathbb{R}^+$ is a user-controlled parameter reproducing the *normal enhancement* technique [CST05]. With $\beta = 1$, we obtain the estimated normal. For $\beta \in [0, 1]$, relief is attenuated, while for $\beta > 1$, relief is exaggerated.

This is illustrated in Figure 8 on an acquired pipe model. The sole benefits of our relief analysis in this case is that it provides a precise control over which surface features are enhanced through scale combination and filtering. Note that the normals of the original detailed surface are easily reconstructed by using the finest scale only and $\alpha = 1$.

5.2. Accessibility shading

The entire fitted polynomial may also be used for local visibility evaluation. We propose a variant of accessibility shading inspired by ambient occlusion. It computes a local visibility value $v_{ij} \in [0, 1]$ for each 1D polynomial as the integra-



(a) Diffuse shading (b) Coarse scale only

(c) All scales (d) + *feature filter*

Figure 9: ***Accessibility shading:*** *Starting from (a) standard diffuse shading, we add accessibility shading (using* $\lambda = 8$*) (b) at a coarse scale first, then (c) at all scales, and finally (d) we apply our* feature filter.

tion of the cosine term over the range of visible directions:

$$v_{ij} = \int_{\phi_{ij}^-}^{\phi_{ij}^+} \cos(\phi)|\sin(\phi)|d\phi = 1 - \frac{1}{2}\left(\cos^2(\phi_{ij}^-) + \cos^2(\phi_{ij}^+)\right). \quad (10)$$

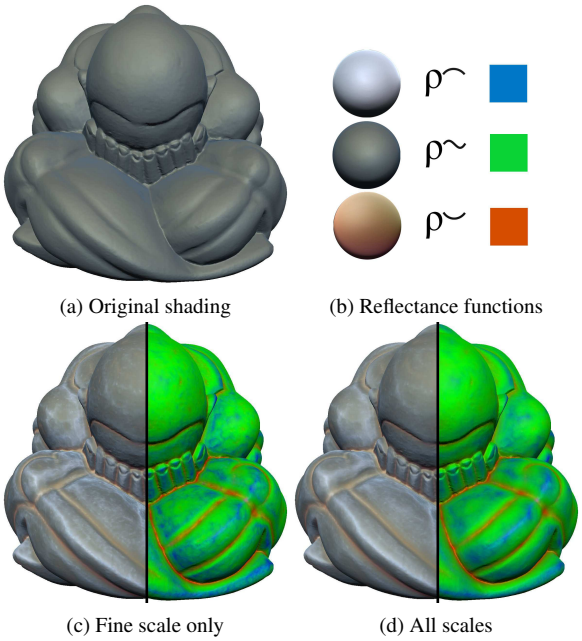
The integration bounds $\phi_{ij}^{\pm}$ correspond to the minimum and maximum visibility angles around the normal $\boldsymbol{n}_{ij}$ (Figure 7(a)). Their derivation is given in the Appendix.

The local visibility terms v_{ij} are combined using Equation 6 to yield an average visibility V which can be arbitrarily scaled by a factor $\lambda \in]0,1]$ to exaggerate the effect. The final filtered occlusion is computed via linear interpolation: $V_\alpha = \text{lerp}(1, V, \alpha)$.

Figure 9 illustrates this technique on a spherical box. Our approach has two advantages over classical accessibility techniques when applied to surface relief: it is fast; and it provides an accurate control over which cavities are conveyed thanks to both multi-scale combination and user-controlled filtering. This effect emphasizes the non purely-local aspect of our analysis, since it exploits the entire polynomial information included in the support size.

5.3. Feature visualization

In scientific illustration, shading is used to help visualize relief features unambiguously. To this end, we assign one



(a) Original shading (b) Reflectance functions

(c) Fine scale only (d) All scales

Figure 10: ***Feature visualization:*** *When using a single lit-sphere (a) to render a statue, we miss many of its features. Instead, we assign one lit-sphere per feature, showing results either at a coarse scale (b), or a fine scale (c), or all scales combined (d), which provides a more balanced depiction of surface details.*

shading function per feature type ($\rho^{\frown}$, $\rho^{\smile}$ and $\rho^{\sim}$), and blend them based on proximity. To do so, we define three membership functions $m_{ij}^{\frown}$, $m_{ij}^{\smile}$ and $m_{ij}^{\sim}$ that tell us how close we are relative to each feature center inside each 1D fitted polynomial. Let's suppose that $t_{ij}^{\smile} < t_{ij}^{\frown}$; then we have:

$$m_{ij}^{\frown}(t) = \left\lfloor \frac{t_{ij}^{\sim} - t}{\Delta_{ij}} \right\rfloor ; m_{ij}^{\smile}(t) = \left\lfloor \frac{t - t_{ij}^{\sim}}{\Delta_{ij}} \right\rfloor ; m_{ij}^{\sim}(t) = \left\lfloor 1 - \frac{|t_{ij}^{\sim} - t|}{\Delta_{ij}} \right\rfloor$$

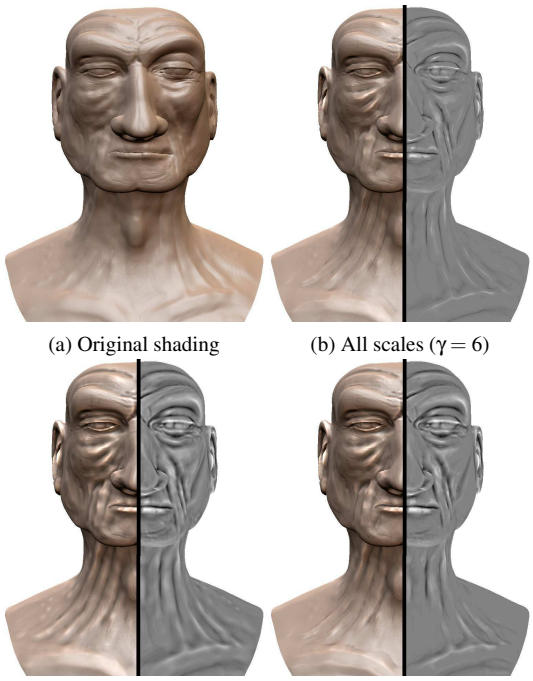
where $\Delta_{ij} = |t_{ij}^{\smile} - t_{ij}^{\frown}|/2$, and recall that $\lfloor x \rceil$ is a function that clamps x to the $[0,1]$ range. In the case $t_{ij}^{\smile} > t_{ij}^{\frown}$, memberships are obtained by exchanging values for $m_{ij}^{\smile}$ and $m_{ij}^{\frown}$.

As shown in Figure 7(b), the membership functions form a partition of unity: $\forall t, m_{ij}^{\frown}(t) + m_{ij}^{\smile}(t) + m_{ij}^{\sim}(t) = 1$. In practice, we evaluate these memberships only at $t = 0$ and plug the resulting values into Equation 6. Since feature combination is a linear process, the resulting averaged memberships $M^{\frown}$, $M^{\smile}$ and $M^{\sim}$ also form a partition of unity. This is a handy property, since it permits to unambiguously assign a reflectance to each membership triplet:

$$\bar{\rho} = M^{\frown}\rho^{\frown} + M^{\smile}\rho^{\smile} + M^{\sim}\rho^{\sim}. \quad (11)$$

The final filtered reflectance is obtained by linear interpolation: $\bar{\rho}_\alpha = \text{lerp}(\rho_B, \bar{\rho}, \alpha)$, with ρ_B the base reflectance.

Figure 10 demonstrates feature visualization on a statue model using lit-sphere shading [SMGG01]. We illustrate another important advantage of our approach here: it provides



(a) Original shading (b) All scales ($\gamma = 6$)

(c) All scales ($\gamma = 0.2$) (d) + *feature filter*

Figure 11: ***Radiance Scaling:*** *We enhance the original shading (a) of a detailed face model by applying radiance scaling to all scales (b-c) using different values for sharpness, and then adding our anisotropic* feature filter *(d) to select the most prominent features. The grey images show the remapped curvature function c_{ij}.*

distance information to nearby concavities, convexities and inflections within a *single* analysis. Again, this is made possible thanks to our ability to detect these features in the neighborhood of each surface point.

5.4. Radiance scaling

Recent shading techniques have demonstrated enhancement capabilities with arbitrary material and illumination. For instance, *radiance scaling* [VPB*10a] correlates shading intensity variations produced by each incoming light direction with surface normal variations. Thanks to our approach, curvature extrema are easily located around a surface point. It allows us to not only convey the curvature of nearby features, but also provides a more precise control. Indeed, as can be seen in Figure 7(c), the highest values of κ_{ij} (in orange) are mostly located in a narrow neighborhood around each feature location. Thus, instead of directly using κ_{ij}, we remap it to a curvature function c_{ij} given by

$$c_{ij}(t) \;=\; \kappa_{ij}(t)\left(1-\left\lfloor\frac{t\text{-}t^{\circ}_{ij}}{\Delta_{ij}}\right\rceil^{2}\right)^{\gamma}, \qquad (12)$$

where Δ_{ij} is the same as in Equation 11, $\gamma \in \mathbb{R}^{+}$ is a user-specified parameter that controls the sharpness of concave and convex regions, and, as before, t°_{ij} corresponds to the location of the closest relevant feature ($t^{\smile}_{ij}$ or $t^{\frown}_{ij}$). As illustrated in Figure 7(c), $c_{ij}(t) = \kappa_{ij}(t)$ for $t \in \{t^{\smile}_{ij}, t^{\frown}_{ij}, t^{\sim}_{ij}\}$, but exhibits a smoother behavior, in a way controlled by γ.

The curvatures c_{ij} are combined using Equation 6 to obtain the average curvature C. The final filtered curvature is computed via simple interpolation ($C_\alpha = \mathrm{lerp}(0, C, \alpha)$ as in Equation 7), and it is used in the radiance scaling technique to modulate incoming light intensities at each surface point.

Figure 11 illustrates this technique applied to a detailed face model sculpted in Autodesk Mudbox©. The main advantage of our approach here is to provide a direct control over the magnitude (via filtering) and the thickness (through γ) of conveyed features. Each of them cannot be achieved using a purely local curvature estimator.

Figures 1 and 12 show more sophisticated shading examples, where our illustrative shading techniques are combined in different ways. The accompanying video and image archive present all our results with more parameter variations and in high resolution.

6. Implementation

In our prototype implementation, the analysis (Section 4.1) is performed on mesh vertices in a preprocess and updated every time a new scale is added. The cubic coefficients for each h_{ij} along with normal deviations η_{ij} are stored in vertex buffers. By trading memory for speed, this strategy enables real-time feature combination (Section 4.2) in a vertex shader followed by per-pixel illustrative shading (Section 5).

The fitting of the polynomials is a critical step of our algorithm. With a naive implementation, complexity would depend on the number of scales. We propose a more efficient strategy that makes complexity depend on the largest scale only (see Table 6 for performance statistics). Given a vertex $\boldsymbol{v}$ and a direction θ_i, the key idea is to perform fitting for all scales incrementally, from the smallest to the largest scale. Relief samples t_k are then taken at equal intervals Δ_t, which we set to half the average edge-length in our system. In this way, the samples t_k are the same at all scales and the fitting of a cubic (Equation 1) boils down to a normal equation:

$$\boldsymbol{A}_j \boldsymbol{u}_{ij}(\boldsymbol{v}) = \boldsymbol{r}_{ij}(\boldsymbol{v}),$$
$$\boldsymbol{A}_j = \textstyle\sum_k L(t_k)^T L(t_k), \quad \boldsymbol{r}_{ij}(\boldsymbol{v}) = \textstyle\sum_k h(\boldsymbol{b}_i(t_k)) L(t_k)^T,$$

where $\boldsymbol{u}_{ij}(\boldsymbol{v})$ is the vector of unknown polynomial coefficients, and $L(t_k) = (1, t_k, t_k^2, t_k^3)$ is the cubic basis vector. From these equations, it is clear that the covariance matrices $\boldsymbol{A}_j$ only depend on their corresponding scale s_j. They are thus computed and pre-inverted once and for all. Likewise, the right hand sides $\boldsymbol{r}_{ij}(\boldsymbol{v})$ and deviation measures η_{ij} (Equation 4) are easily computed incrementally from $\boldsymbol{r}_{i(j\text{-}1)}(\boldsymbol{v})$ and $\eta_{i(j\text{-}1)}$. Fitting is performed on the GPU using a minimalistic half-edge data structure to walk on the surface. Table 6 reports some performance statistics of our OpenCL implementation running on a Nvidia GTX 480 graphics card.

(a) Original shading (b) Feature weights (c) RS + AS (d) RS + AS + *feature filter*

Figure 12: *The original diffuse shading (a) of the scanned portion of a cave wall hardly reveals the engraved shape. After the analysis, and assignment of different lit-spheres to the extracted features (b) we can provide a more legible rendering using radiance scaling (RS) and accessibility shading (AS) (c). Filtering removes most of the unwanted features (d).*

model (#vertices)	scales {35}	{75}	{35,55,75}	base/relief decomposition
Box (180k)	2.4	5.2	5.8	12.5
Head (667k)	11.8	25.2	26.9	66.3

Table 1: ***Analysis timings*** *for different scales (given in terms of number of samples) and 8 directions. The last column reports the time in second to decompose the input mesh into a base/relief layers with our implementation of [ZTS09].*

7. Discussion

Depicting surface relief through illustrative shading requires to gain knowledge and control over relief features at various scales. Our approach permits to identify, filter and combine three types of relief features at the small expense of a short, automatic pre-process; after which all subsequent manipulations and rendering occur in real-time. Our current implementation works at the vertex level, but it would be easily extended to take 3D models with displacement maps as input, performing the analysis and storing its outcome per texel. Another limitation is that we interpolate averaged feature values F per pixel, which lets appear tesselation in close-up views. One solution could be to interpolate h_{ij} per pixel; however, the interpolation of such directional cubic polynomials is not straightforward. Our method could also be adapted to other kinds of surface representations, provided we can intersect the surface with a plane, and march on the resulting curve. This includes for instance raw point clouds reconstructed with implicit fitting methods.

Apart from these technical issues, the main limitation of our approach is that it assumes that relief information can be represented by a single height-field. Although this is true for many kinds of surface reliefs, this is not always the case. One solution would be to interleave our multi-scale analysis with 3D multi-scale decompositions (e.g., [PKG06]), though it might require longer preprocessing times. Our approach also neglects the curvature of the base surface. While it is assumed to be flat most of the time, one could imagine to cut the neighborhoods at regions of sharp transitions [SGW06].

8. Conclusion

We have presented a novel analysis technique of surface relief that identifies three types of features in the neighborhood of each surface point, and provides multi-scale combination mechanisms that favor most pertinent features. The direct benefit of our approach is to gain a direct global control over the type of relief features depicted through illustrative shading techniques: the influence of each scale is easily tuned, features are efficiently filtered out based on their importance and/or based on our anisotropic metric, and a variety of feature properties can be selected for illustration. This is made possible by our cubic polynomial fitting approach that identifies features at multiple scales and orientations, without the necessity of a complex parametrization.

Our system thus provides a convenient solution for the depiction of surface relief, either for its exploration as in scientific illustration, or for its exaggeration as in visual effects. It relies on a base/relief decomposition though, and we believe that different solutions to this problem may be chosen depending on the target application, considering how relief has been generated. Has it been engraved or eroded? Does it include cracks? Is it composed of different materials? Although we believe concavities, convexities and inflections are very common features, they might not always be adapted. An exciting avenue of future work would be to consider alternative feature types for expert applications. As long as the fitting functions used to identify them are one-dimensional, our method will be applicable with little changes. Finally, we believe such relief analysis technique could be used in many other applications, for instance, for geometric-aware texture synthesis [MKC*06].

Acknowledgments

This work has been supported by the ANR SeARCH project (ANR-09-CORD-019), and the European Community's Seventh Framework Program [FP7 – 2007/2013] under the Grant Agreement 270404. We would also like to thanks Robert Vergnieux, Romain Vergne and Nicolas Mellado for their insightful discussions. The models of figures 4, 8, 9, 10 are provided courtesy of AIMSHAPE Shape Repository, and figure 5 courtesy of M. Kolomenkin.

References

[CP05] CAZALS F., POUGET M.: Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design 22*, 2 (2005). 2

[CPG09] CIPRIANO G., PHILIPS JR G., GLEICHER M.: Multi-Scale Surface Descriptors. *IEEE Trans. Visualization and Comput. Graph. 15*, 6 (2009), 1201–1208. 2

[CST05] CIGNONI P., SCOPIGNO R., TARINI M.: A simple normal enhancement technique for interactive non-photorealistic renderings. *Computers & Graphics 29* (2005), 125–133. 2, 6

[dC76] DO CARMO M. P.: *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, NJ, 1976. 2

[GGG08] GUENNEBAUD G., GERMANN M., GROSS M.: Dynamic sampling and rendering of algebraic point set surfaces. *Computer Graphics Forum 27*, 2 (2008). 2, 5

[KST08] KOLOMENKIN M., SHIMSHONI I., TAL A.: Demarcating Curves for Shape Illustration. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 27*, 5 (2008). 2

[KST09a] KOLOMENKIN M., SHIMSHONI I., TAL A.: On edge detection on surfaces. In *IEEE conf. Comput. Vision and Pattern Recognition (CVPR)* (June 2009), Ieee, pp. 2767–2774. 2

[KST09b] KOLOMENKIN M., SHIMSHONI I., TAL A.: Prominent field for shape processing of archaeological artifacts. In *IEEE Int. Conf. Comput. Vision Workshop (ICCVW)* (Sept. 2009), Ieee, pp. 915–922. 2

[Mil94] MILLER G.: Efficient algorithms for local and global accessibility shading. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1994), SIGGRAPH '94, ACM, pp. 319–326. 2

[MKC*06] MERTENS T., KAUTZ J., CHEN J., BEKAERT P., DURAND F.: Texture transfer using geometry correlation. In *Rendering Techniques* (2006), pp. 273–284. 9

[MWGA06] MALZBENDER T., WILBURN B., GELB D., AMBRISCO B.: Surface Enhancement Using Real-time Photometric Stereo and Reflectance Transformation. In *Eurographics Symposium on Rendering* (2006), Eurographics. 2

[OBS04] OHTAKE Y., BELYAEV A., SEIDEL H.-P.: Ridge-valley lines on meshes via implicit surface fitting. In *ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), SIGGRAPH '04, ACM, pp. 609–612. 2

[PG04] PHARR M., GREEN S.: *GPU Gems*. Addison-Wesley, 2004, ch. Ambient Occlusion. 2

[PKG06] PAULY M., KOBBELT L. P., GROSS M.: Point-based multiscale surface representation. *ACM Trans. Graph. 25* (April 2006), 177–193. 9

[RBD06] RUSINKIEWICZ S., BURNS M., DECARLO D.: Exaggerated shading for depicting shape and detail. *ACM Trans. Graph. (Proc. SIGGRAPH) 25*, 3 (July 2006), 1199. 2, 5

[SGW06] SCHMIDT R., GRIMM C., WYVILL B.: Interactive decal compositing with discrete exponential maps. *ACM Trans. Graph. 25* (July 2006), 605–613. 9

[SMGG01] SLOAN P.-P. J., MARTIN W., GOOCH A., GOOCH B.: The lit sphere: a model for capturing npr shading from art. In *No description on Graphics interface 2001* (Toronto, Ont., Canada, Canada, 2001), GRIN'01, Canadian Information Processing Society, pp. 143–150. 7

[VPB*10a] VERGNE R., PACANOWSKI R., BARLA P., GRANIER X., SCHLICK C.: Radiance Scaling for Versatile Surface Enhancement. In *Proc. symposium on Interactive 3D and games (I3D)* (2010), ACM. 8

[VPB*10b] VERGNE R., PACANOWSKI R., BARLA P., GRANIER X., SHLICK C.: Improving Shape Depiction under Arbitrary Rendering. *IEEE Trans. Visualization and Comput. Graph. PrePrint*, 99 (Dec. 2010), 1–12. 2

[VVC*11] VERGNE R., VANDERHAEGHE D., CHEN J., BARLA P., GRANIER X., SCHLICK C.: Implicit Brushes for Stylized Line-based Rendering. *Comp. Graph. Forum (Proc. Eurographics) 30*, 2 (2011). 2

[ZCF*10] ZHANG X., CHEN W., FANG J., WANG R., PENG Q.: Perceptually-motivated Shape Exaggeration. *The Visual Comp. (Proc. CGI) 26*, 6-8 (2010), 985–995. 2

[ZTS09] ZATZARINNI R., TAL A., SHAMIR A.: Relief Analysis and Extraction. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 28*, 5 (2009), 1–10. 2, 3, 9

Appendix

Relief analysis - feature locations

As explained in Section 4.1, the convexity and concavity locations $t^{\frown}$, $t^{\smile}$ are the roots of the rational polynomial $\kappa'(t)$, where κ (Equation 2) corresponds to the curvature of a cubic polynomial $h(t) = u_0 + u_1 t + u_2 t^2 + u_3 t^3$. If $u_3 \neq 0$, then there exist only two real roots, $t^{\frown}$ and $t^{\smile}$, given by:

$$\pm \frac{\sqrt{3\sqrt{9u_3^2u_1^2 - 6u_3u_2^2u_1 + u_2^4 + 5u_3^2 - 6u_3u_1 + 2u_2^2}}}{3\sqrt{5}u_3} - \frac{u_2}{3u_3} \qquad (13)$$

The inflexion position $t^{\sim}$ is then obtained as the unique root of the second-order derivative polynomial ($h''(t^{\sim}) = 0$):

$$t^{\sim} = \frac{-u_2}{3u_3} = \frac{t^{\frown} + t^{\smile}}{2} \quad . \qquad (14)$$

On the other hand, if $u_3 = 0$ but $u_2 \neq 0$, then there is only a unique concavity or convexity point at $t = -u_1/2u_2$, and no inflexion point can be detected. If u_2 is also zero, then no feature points can be extracted. In these cases, the respective undefined $d_{ij}^{\star}$ values are set to 1.

Accessibility shading - integration bounds

The computation of the accessibility term of Equation 10 relies on the determination of the cosine of the two integration bounding angles $\phi^{\pm}$. In the case of a cubic polynomial $h(t)$, they are given by the clamped dot product between the normal direction $\boldsymbol{n}$ (as defined in Equation 9), and the two normalized vectors $(\boldsymbol{x}^{\pm} - \boldsymbol{o})$, where $\boldsymbol{x}^{\pm} = (t^{\pm}, h(t^{\pm}))$ are the two *extremity points* of the curve visible from the *observer* location $\boldsymbol{o} = (0, -u_0)$ (see Figure 7). Since the curve is clamped to the range $[-s_j/2, s_j/2]$, they either coincide with the boundaries of the curves, or with the point $\boldsymbol{y} = (t_y, h(t_y))$ for which the vector $(\boldsymbol{y} - \boldsymbol{o})$ is tangent to the curve at $\boldsymbol{y}$. It is obtained by solving for the cubic equation $(-h'(t_y), 1) \cdot (\boldsymbol{y} - \boldsymbol{o}) = 0$, which admits a single non null root: $t_y = -u_2/(2u_3)$. The two positions t^-, t^+ of the visible extremities $\boldsymbol{x}^{\pm} = (t^{\pm}, h(t^{\pm}))$ are finally given by:

$$(t^-, t^+) = \begin{cases} \left(-\frac{s_j}{2}, \min\left(\frac{s_j}{2}, t_y\right)\right) & \text{if } t_y > 0, \\ \left(\max\left(-\frac{s_j}{2}, t_y\right), \frac{s_j}{2}\right) & \text{otherwise.} \end{cases} \qquad (15)$$