Pacific Graphics 2018
H. Fu, A. Ghosh, and J. Kopf
(Guest Editors)

Volume 37 (2018), Number 7

Mumford-Shah Mesh Processing using the Ambrosio-Tortorelli Functional

Nicolas Bonneel[*,1], David Coeurjolly[*,1], Pierre Gueth[*,2], Jacques-Olivier Lachaud[*,3]

[*]joint first authors [1]CNRS, Univ. Lyon [2]Arskan [3]Université Savoie Mont Blanc

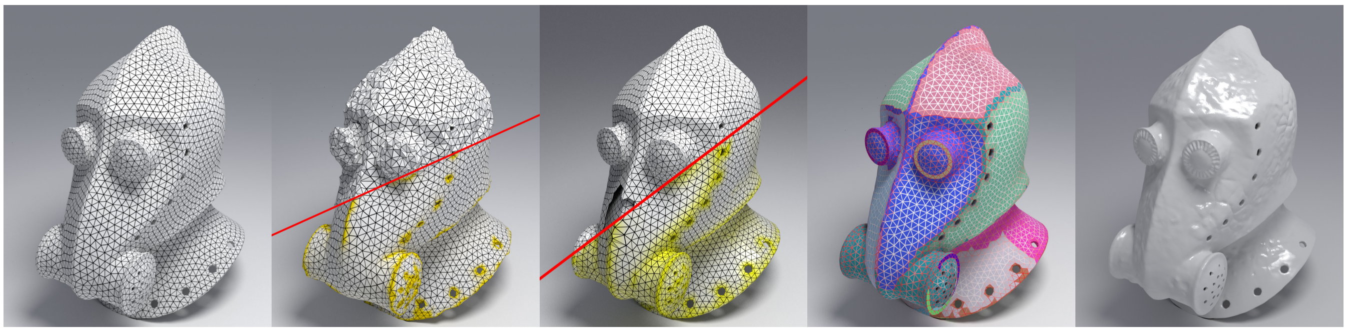


Figure 1: *Our discretization of the MS functional along with our new vertex projection technique allows for applications such as mesh denoising, segmentation, inpainting and normal map embossing. (b) Our method removes heavy noise while preserving sharp features shown in yellow. (c) We remove vertices highlighted in cyan from the original mesh (a): our method finds sharp features (shown in yellow) and sucessfully inpaints the missing area. (d) We decompose the original mesh (a) into piecewise smooth segments whose boundaries are characterized by sharp features shown on edges in red. (e) We emboss a normal map into the mesh vertices.*

Abstract
The Mumford-Shah functional approximates a function by a piecewise smooth function. Its versatility makes it ideal for tasks such as image segmentation or restoration, and it is now a widespread tool of image processing. Recent work has started to investigate its use for mesh segmentation and feature lines detection, but we take the stance that the power of this functional could reach far beyond these tasks and integrate the everyday mesh processing toolbox. In this paper, we discretize an Ambrosio-Tortorelli approximation via a Discrete Exterior Calculus formulation. We show that, combined with a new shape optimization routine, several mesh processing problems can be readily tackled within the same framework. In particular, we illustrate applications in mesh denoising, normal map embossing, mesh inpainting and mesh segmentation.

1. Introduction

The Mumford-Shah (MS) functional, originally intended for image segmentation [MS89], is a well-known functional which models an image as a piecewise-smooth function. Minimizing this functional recovers both a piecewise-smooth image that can be useful for applications such as image denoising, and the set of discontinuities can be useful for edge detection. Since the nineties, this model has seen a tremendous success in image processing and has found many applications from image segmentation [TYW01, VC02] to denoising [TYW01], inpainting [ES02], magnification [TYW01], deblurring [BSK06] or registration [BAS10].

This functional has however attracted little attention in geometry processing, where similar problems are encountered. Notable exceptions include the work of Zhang et al. [ZZWC12] that makes use of a convexified version of the MS model for mesh segmentation, and the finite element method of Tong and Tai [TT16] to construct feature lines on meshes. This lack of interest might have come from several challenges that need to be overcome. First, the model itself is not suited to a computational resolution. Hence, numerical methods only solve approximate MS models. Second their computational complexity makes algorithms tuned for regular grids difficult to scale to large unstructured meshes. Third, the MS formulation, once adapted to manifolds, exhibits differential operators that need careful handling when discretized over triangular meshes. Last, the MS model represents a piecewise-smooth *scalar function* over a domain; it is thus not obvious what this function should be in the context of mesh processing and for which applications.

Fortunately, recent progress on the first two issues allows to paint a brighter picture. On the computational side, various approximations of the MS functional have recently been proposed. In this paper, we are interested with an accurate approximation of the MS functional: the Ambrosio-Tortorelli's functional [AT90]. Recent advances have further made calculus on meshes practical and affordable. In particular, Discrete Exterior Calculus (DEC) [Hir03] has become popular for easily formulating and solving differential equations on meshes. We will formulate the MS functional as a discrete AT model in the language of DEC, in the spirit of the method of Coeurjolly et al. [CFGL16].

Regarding the choice of scalar function to use in the MS functional, it obviously depends on the targeted applications. For our applications, we take advantage of our DEC formulation to consider each component of the normal vector at each point of the surface as a 0-form stored on the faces of a dual mesh. This results in a set of three scalar functions, generalizing MS to vector functions [FI14]. We will jointly minimize MS over these three scalar functions. Overall, this essentially allows to smooth out the normals of an input mesh so they better match the underlying continuous surface being approximated while preserving mesh discontinuities.

Benefitting from these advances, we formulate a set of classical mesh processing problems using the MS functional applied to the manifold's normal vector field. From the resulting regularized normal vector field, we further introduce a shape optimization routine as an easy-to-implement projection operator which deforms the geometry so that geometric normals match the obtained regularized normals. We illustrate our method on a number of applications, such as mesh denoising, normal map embossing, mesh inpainting and mesh segmentation.

2. Related work

The Mumford-Shah Model. Mumford and Shah described a functional representing an input image by a piecewise-smooth approximation [MS89]. This functional reads:

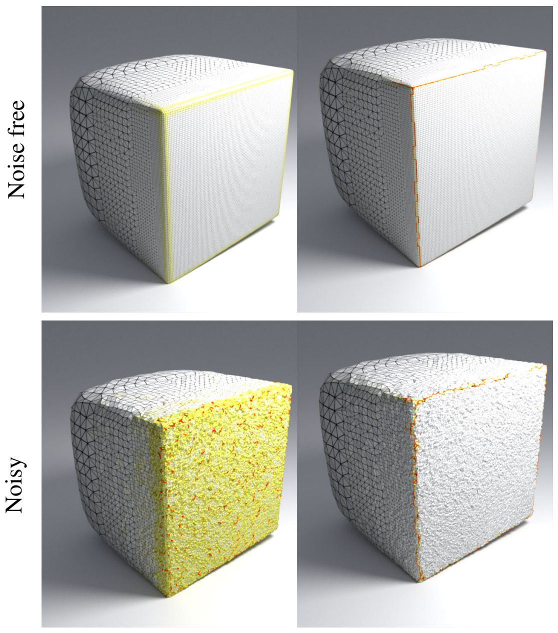
$$MS[u,C] = \alpha \int_{\Omega} (u-g)^2 \mathrm{d}x + \beta \int_{\Omega \setminus C} |\nabla u|^2 \mathrm{d}x + \gamma \int_{C} \mathrm{d}s, \qquad (1)$$

with g the input image on a two-dimensional planar domain Ω, u its (unknown) approximation, and C a set of (unknown) curves describing the set of discontinuities. The parameters of the model intuitively model the tightness α of the approximation to the input image, the smoothness β of the approximation, and the length of discontinuities γ. While for segmentation purpose, it is often assumed that C forms a closed curve or is the boundary of some partition of the space, this is not a requirement of the model nor the optimal solution to it. In addition, this model is not restricted to images: in the general case, Ω need not be a plane and can be an arbitrary surface, and g a real-valued function over this surface.

Unfortunately, except in limited scenarios, this functional is non-convex and very difficult to optimize.

Optimizing the MS functional. To avoid these difficulties, various approximations to MS have been proposed, and among them, the most natural ones are convex envelopes or convex approximations to MS functional. When restricting the problem to foreground / background extraction, the MS functional can be convexified by replacing the term accounting for the length of segment boundaries by the total variation of the gradient of a segment binary indicator function [CEN06]. Instead of a piecewise-smooth assumption, this assumes a piecewise constant function and precludes open segment boundaries or isolated discontinuities. This can be extended to multiple segments but still necessarily produces disjoint sets. While this assumption can be appropriate for certain segmentation applications and has indeed been sucessfully used for mesh segmentation [ZZWC12], this may not be suitable in our context where our function of interest is the normal field that varies relatively smoothly across the surface [ZZWC12] and may exhibit internal discontinuities (see Fig. 5, first row). Similarly, Tsai et al. [TYW01] directly optimize the MS functional using level sets, but also requires closed segments boundaries. These approaches thus lack the generality we are targeting. We instead resort to an early approximation from Ambrosio and Tortorelli [AT90] that relies on Γ-convergence results, and provably converges towards the MS functional. Its numerical optimization is nonetheless far from being trivial. For instance, Chambolle and Dal Maso [CDM99] and Bourdin and Chambolle [BC00] employ a Finite Element Method with adaptive mesh refinement and edge alignment is required for its optimization. Even with such advanced technique, these numerical methods are very sensitive to noise [FLT16]. Fortunately, new discretization schemes for the Ambrosio-Tortorelli (AT) model have been designed on grids [FLT16, CFGL16] and do not require FEM anymore nor adaptive mesh to get piecewise smooth solutions. This discretization has seen applications for image restoration [FLT16] and feature extraction on voxel-based digital geometries [CFGL16]. A first discrete differential calculus version of AT has been employed by Pokrass *et al.* [PBB11] to solve the partial matching of non-rigid 3D shapes. Their discretization is different from ours since discontinuities and values live on vertices, while the cross-term is evaluated on faces (see below), resulting in smoother features. This is fine for their specific objective but limits its range of applications. In contrast our approach allows coarse-to-fine detection of features for a variety of geometry processing tasks. An alternative finite element discretization of AT has been recently proposed for triangle meshes [TT16]. This approach is the closest one to our formulation but it is not able to recover piecewise smooth patchs on noisy data as illustrated in Figure 2. We propose a specific Discrete Exterior Calculus discretization of this functional, which achieves sharper and more robust features, as we show in our experiments.

Applications in image processing. The MS functional originated from the image processing community as a way to represent an image as a piecewise-smooth function. In the last 40 years, it has led to numerous applications, even if most of the time it is only simpler variants of the MS functional that are solved. Image segmentation [TYW01, VC02] and denoising [TYW01] are the two main applications that directly follow from the variables being optimized for (that is, discontinuity boundaries and piecewise smooth approximations). Image inpainting [ES02] is rendered possible by



(a) [TT16] (b) Our method

Figure 2: *We compare our features (b) to that extracted by the finite element discretization of Tong and Tai [TT16] (a) using the same parameters ($\lambda = 0.06, \alpha = 0.07$). The feature function is shown in yellow, while extracted feature lines are shown in red. Our method is able to detect features even at relatively high noise levels.*

removing the data attachment term inside the inpainted area. A 3x image magnification [TYW01] is performed by interlacing the input image in a 3x upsampled grid, and inpainting the missing pixel values. Bar et al. extend a deconvolution functional by summing it with the MS functional [BSK06], hence leading to a new image deconvolution method with piecewise smoothness prior. Similarly, Ben-Ari and Sochen integrate the MS model in a stereo-matching functional, resulting in piecewise smooth disparity maps [BAS10]. We believe the MS model has the potential to produce a similar number of applications in geometry processing, and this paper presents an initial set of applications.

Variational methods in mesh processing. Variational methods for mesh processing have long been studied. A variational method optimizes a given functional over a space of functions. The most common examples of such functionals used in mesh processing include Lloyd's functional [CSAD04], Total Variation [ZWZD15], the variational problem associated with the Poisson equation [YZX*04], the curvature energy [Zor05] or the MS functional [CFGL16,ZZWC12]. Individual mesh processing applications have their own set of priors, hypotheses and constraints, and it is illusory to search for an all-purpose functional. However, we show that the MS functional appropriately solves a number of problems and is of value to the geometry processing community.

3. Discretizing Mumford-Shah over Surfaces

This section describes our model for mesh processing, that extends the model of Coeurjolly et al. [CFGL16] tailored for voxel-based geometries. In particular, we briefly discuss the AT approximation, and expose our discretization over triangular meshes within the DEC framework.

3.1. Ambrosio-Tortorelli's Approximation

Ambrosio and Tortorelli reformulates the MS functional [AT90] by working on a smoothness indicator function rather than the set of discontinuities C. This indicator function, denoted by v in the following, is optimized such that it takes the value $v(x) = 1$ wherever $g(x)$ is smooth, and $v(x) = 0$ on discontinuities. Therefore we call it the *feature* function. In their model, this function v is smooth and kept *close to* 1 where g is smooth, and its oscillations are prevented by keeping ∇v close to 0. Their functional can be written as follows:

$$AT_\varepsilon[u,v] = \int_\Omega \alpha(u-g)^2 + |v\nabla u|^2 + \lambda\varepsilon|\nabla v|^2 + \frac{\lambda}{\varepsilon}\frac{(1-v)^2}{4}\,\mathrm{d}x. \quad (2)$$

This functional now depends on the same parameters α and λ (β is set to 1 without loss of generality), but also requires a parameter ε. Parameter λ is still such that $1/\lambda$ represents the length of discontinuities, and parameter ε controls the smoothness of the feature v. Ambrosio and Tortorelli proved that when ε tends to zero, minimizers of this functional exactly corresponds to those of the *MS* functional. However, the problem is now quasi convex, and the integration domain is fixed and does not involve unknown curves.

3.2. Minimizing the Energy

For a fixed ε, since the energy (2) is quasi convex, we can minimize it by alterning minimizations over u and v. We follow the AT energy formulation proposed in [CFGL16]. The feature scalar field v is discretized as a primal $0-$form, that is a collection of scalar values associated with each vertex of the mesh. The normal vector fields (both u and g) are each discretized as three dual $0-$forms, one for each coordinate of the $\mathbb{R}^3$ embedding space. Note that Focardi and Iurlano showed that AT in the vectorial case also converges towards an approximation of MS [FI14]. Each of those three dual $0-$forms can be seen as a scalar value associated with each primal face (or dual vertex) of the mesh. Denoting d and $\bar{\mathrm{d}}$ the primal and dual exterior derivatives, it follows that $\mathrm{d}v$ is a primal $1-$form and $\bar{\mathrm{d}}u$ corresponds to three dual $1-$forms. Norms are induced by the natural inner products between $k-$forms. We rewrite then the AT functional as a sum of inner products on primal and dual 0 and 1-forms:

$$\begin{aligned} AT_\varepsilon[u,v] =& \alpha\langle u-g, u-g\rangle_{\bar{0}} + \langle v\bar{\mathrm{d}}u, v\bar{\mathrm{d}}u\rangle_{\bar{1}} \\ &+ \lambda\varepsilon\langle \mathrm{d}v, \mathrm{d}v\rangle_1 + \frac{\lambda}{4\varepsilon}\langle 1-v, 1-v\rangle_0 . \end{aligned} \quad (3)$$

Note that we have discretized (2) such that our feature function v does not become identically one when epsilon tends to 0, and remains zero around feature edges. To achieve such discontinuities, the function v acts as a rescaling of the derivatives of u and not as a rescaling of the squared gradient norm. Hence it operates directly

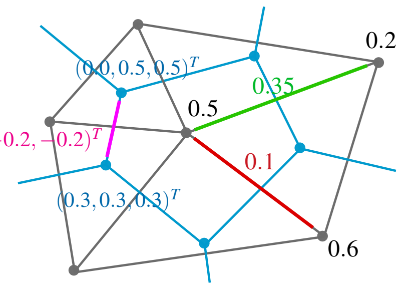


Figure 3: *Notations on a primal and dual mesh for the energy optimization (edge orientation not shown for the sake of clarity): v is a $0-$form on vertices. In red (resp. green), we have the Av values (resp. Mv values) on edges. In blue, the triple of dual $\bar{0}-$forms, u on dual vertices and the Bu values in magenta on dual edges.*

on edges in (3) and it does not operate on vertices as would give a standard DEC discretization of (2). Our discretization of AT is thus more related to discontinuous FEM than to continuous and linear basis elements. For instance, the FEM approach of Tong and Tai [TT16] oversmoothes features (function v becomes almost 1 everywhere) and requires post-processing to find its valleys.

We write the AT_ε energy in matrix form, using the following notations using thec classical discrete exterior calculus (DEC) framework where the dual vertices are circumcircle of primal triangles [CdDS13]: the Hodge star $\star_i$ is written as the diagonal matrix S_i , the vector diagonalization operator is denoted by Diag, differential operators d_0 and $\bar{d}_1$ are respectively denoted by matrices A and B, and the matrix M is the operator averaging incident 0-form values at each 1-cell ($M := \text{abs}(A)/2$). We thus have:

$$\begin{aligned} AT_\varepsilon[u,v] =& \alpha(u-g)^T S_{\bar{0}}(u-g) + u^T B^T \operatorname{Diag}(Mv) S_{\bar{1}} \operatorname{Diag}(Mv) Bu \\ &+ \lambda\varepsilon v^T A^T S_1 A v + \frac{\lambda}{4\varepsilon}(1-v)^T S_0 (1-v). \end{aligned} \tag{4}$$

The cross term ($v\nabla u$ in Eq. (2) and its discretizations) can indifferently be written with the vectors v or u inside or outside the inner product. Observe that the energy gradient with respect to u (5) (resp. v (6)) is a linear operator with respect to u (resp. v). Thus the alternating minimization method amounts to alternating between the resolution of the following linear systems:

$$\nabla_u AT_\varepsilon[u,v] = 0 \Leftrightarrow$$
$$\left[\alpha S_{\bar{0}} - B^T \operatorname{Diag}(Mv) S_{\bar{1}} \operatorname{Diag}(Mv) B\right] u = \alpha S_{\bar{0}}\, g, \tag{5}$$

$$\nabla_v AT_\varepsilon[u,v] = 0 \Leftrightarrow$$
$$\left[\frac{\lambda}{4\varepsilon} S_0 + \lambda\varepsilon A^T S_1 A + M^T \operatorname{Diag}(Bu) S_{\bar{1}} \operatorname{Diag}(Bu) M\right] v = \frac{\lambda}{4\varepsilon} S_0, \tag{6}$$

As long as the diagonal Hodge stars S_0 and $S_{\bar{0}}$ are non-degenerate, the left-hand sides are full rank, yielding a unique minimum at each iteration. It is a known result in convex analysis linked to block coordinate descent algorithms [Ber99, Prop. 2.7.1], that these iterations must converge to a stationary point of AT_ε.

Convergence and stability are improved by progressively reducing ε rather than directly solving the problem with a small ε. We hence perform the above minimization repeatedly, dividing ε by 2 at each iteration (we typically start at $\varepsilon_0 = 2$ and decrease until $\varepsilon = 0.25$). This allows to better infer the general position of features at a coarse scale (large ε), and then to better precisely delineate them at a fine scale (small ε). This coarse-to-fine optimization is especially important in the inpainting case, where the value of ε_0 should be around the radius size of the inpainted region.

4. Deforming a Mesh to a Prescribed Normal Field

Given a regularized normal field u over an input mesh (e.g., given by a first AT minimization), we wish to deform the mesh by moving its vertices such that the triangles normal vectors match the prescribed normal field u. We perform this operation by minimizing an energy E consisting of three terms: a normal matching term E_m, a fairness term E_f, and a data attachment term E_d, described in this section. Note that some terms may appear in various forms in other works [Tau01, FDCO03, PAH*07, SRML07, BDS*12]. We describe them for self-containedness and propose a fairness energy to take into account the feature field. Our method then proceeds by alternately solving for the unknown positions of the deformed mesh with a fixed feature field v and solving the AT functional to update the feature field v and piecewise smooth normals u.

Matching normals. Let p be the vertices positions of the deformed mesh, and $F = \{f_i\}_{i=1..N}$ the set of triangles where f_i^j denotes the j^{th} vertex of the i^{th} triangle, with $j \in \{0,1,2\}$. As we are deforming the mesh, the topology (and hence the set of triangles) of the deformed mesh is the same as that of the input mesh. We denote by $|p|$ the number of vertices. Our goal is to find p such that per-triangle geometric normals match the prescribed normal field u. We formulate this condition by imposing that each edge of each triangle be orthogonal to u:

$$\begin{aligned} E_m &= \sum_{f_i \in F} ((p_{f_i^1} - p_{f_i^0}) \cdot u)^2 + ((p_{f_i^2} - p_{f_i^1}) \cdot u)^2 + ((p_{f_i^0} - p_{f_i^2}) \cdot u)^2 \\ &= \sum_{f_i \in F} E_m(f_i)\,. \end{aligned}$$

Note that the summation is performed over all triangles and not over edges since u is known per triangle. This induces sharper discontinuities. Its gradient with respect to each triangle vertex position is given by:

$$\begin{aligned} \nabla_{p_{f_i^j}} E_m(f_i) &= 2((2p_{f_i^j} - \sum_{k \neq j} p_{f_i^k}) \cdot u)u \\ &= C(f_i)p\,, \end{aligned}$$

where $C(f_i)$ is a 3×9 matrix, that can be assembled in a $3|p| \times 3|p|$ matrix C for all vertices of all triangles of the mesh.

Fairness. We additionally enforce the piecewise smoothness of the deformed mesh by adding a fairness term. This term forces neighboring triangles to share similar geometric normals, at least wherever the feature $v = 1$ and corresponds to a thin plate energy. Its effect can be appreciated in Fig. 4. For an edge $e = (p_{i_1}, p_{i_2})$ between triangles $t_1 = (p_{i_1}, p_{i_2}, p_{i_3})$ and $t_2 = (p_{i_1}, p_{i_2}, p_{i_4})$, the fairness energy reads:

$$E_f(e) = \left(\frac{v(p_{i_1}) + v(p_{i_2})}{2}\right)^2 \|p_{i_1} + p_{i_2} - p_{i_3} - p_{i_4}\|^2 ,$$

and we obtain E_f by summing over all edges $E_f = \sum_e E_f(e)$. Fixing v, the gradient of $E_f(e)$ with respect to p_{i_1} (resp. p_{i_2}) is:

$$\begin{aligned}\nabla_{p_{i_1}} E_f(e) &= 2\left(\frac{v(p_{i_1}) + v(p_{i_2})}{2}\right)^2 (p_{i_1} + p_{i_2} - p_{i_3} - p_{i_4}) \\ &= D(e_i)p ,\end{aligned}$$

where $D(e_i)$ is a 3×12 matrix. These matrices again be assembled in a $3|p| \times 3|p|$ matrix D by summing contributions of all edges.

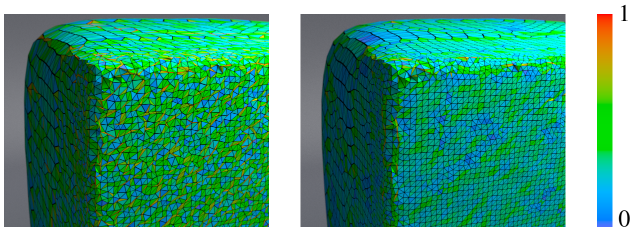


Figure 4: *Denoising the model of Fig. 2 without (a) and with (b) a small fairness term E_f ($w_1 = 0.5$). The fairness term favors more regular near-Delaunay meshes. The colormap indicates the aspect ratio of each triangle ($1 - \frac{\alpha_T}{\pi/3}$ with α_T the minimum angle of triangle T) .*

Data attachment. We finally prevent vertices to depart too much from their original position, and add a data attachment term. Denoting q the original position of the vertices, this reads:

$$E_d = \sum_i \|p_i - q_i\|^2 .$$

Its gradient is $\nabla_p E_d = 2(p - q)$.

Solving for positions. Denoting $E = E_m + w_1 E_f + w_2 E_d$ the weighted sum of the energies, we can now obtain p by solving $\nabla_p E = 0$. This amounts to solving the linear system

$$(C + w_1 D + w_2 \mathrm{Id})p = w_2 q ,$$

where Id is the $3|p| \times 3|p|$ identity matrix. One can easily show that by definitions, operators C and D are positive-semidefinite. As long as $w_2 > 0$, the overall left-hand linear operator is thus positive-definite and efficient inversion algorithms can be used to solve the system. Except for inpainting which requires large values of w_1 as discussed next, most results were obtained with $w_1 = 1$, though decreasing w_1 can be useful for large noise or unreliable vertex positions. We have kept w_2 fixed and small throughout all our experiments, with $w_2 = 0.05$. Note that meshes are uniformly scaled to fit the unit ball, so that parameters are comparable across meshes.

5. Applications

Inspired by image processing, our method serves various geometry processing applications illustrated in this section. This includes restoring noisy meshes, segmenting meshes, inpainting missing data, or embossing a normal map into the mesh vertices.

5.1. Denoising

Given an input noisy mesh, our method smoothes out noise while preserving features. For this application, we alternate between solving for the MS functional to regularize the normal vector field, and conforming the mesh vertices to the estimated normals. In our examples, our method achieves visually satisfactory results within 2 to 6 such iterations (see Fig. 6). In this experiment, we have considered both synthetic shapes perturbed using a Gaussian noise in order to have a ground-truth, and more realistic noisy LiDAR shapes. We illustrate these results in Fig. 5 and show the effect of varying λ in Fig. 7. In practice, decreasing λ produces longer discontinuities, and thus less smooth denoising results. Numerical and visual comparisons with state-of-the-art denoising techniques can be found respectively in Table 1 and in Fig. 5.

We compare our features to that of Tong and Tai [TT16] who solve a similar AT functional via a Finite Elements discretization, using a noisy non-uniform mesh (Fig. 2) and meshes with varying amount of noise (Fig. 8). We show that our features are more robust to noise, making our method suitable for denoising applications. The MS functional reconstructs a piecewise smooth vector field in terms of direction and orientation. Starting from a smooth surface with some few flipped normal vectors, the MS functional will reorient the vectors as incorrect orientations introduce spurious discontinuities in the normal field. Combined with the projection step (Sect. 4) where the fairness term unfolds the surface, we do not observe flipped triangle in our experiments.

5.2. Segmentation

We segment meshes into piecewise smooth parts. Note that, contrary to semantic mesh segmentation, our segmentation is purely geometric and bears no semantic meaning. This can find applications in mesh parameterization where seams should remain as little visible as possible [RNLL10], or as a preprocessing for further feature extraction (see the mesh segmentation survey of Attene et al. [AKM*06]). However, this makes numerical evaluation difficult since no ground truth segmentation is available, in contrast to semantic segmentation benchmarks [CGF09].

To perform this segmentation, we make use of our MS solver to obtain the feature field v for each vertex of the mesh. We further clamp v since it can occasionally take values outside of the range $[0,1]$ due to approximations. We then make use of the method of Keuper et al. [KLB*15] which solves a minimum multicut problem in the triangle adjacency graph using the Kernighan Lin method. This method takes as input the probability of two adjacent triangles to belong to different segment, which amounts to bringing the feature v from vertices to edges. For each edge, we obtain a splitting probability by averaging the value $1 - v$ of both vertices of the edge, while forcing this probability to 0.1% when adjacent triangles have their (regularized) normals further than 5° apart. Segmentation results can be seen in Fig. 9.

In Fig. 10, we illustrate the differences between our geometric segmentation and state-of-the-art semantic segmentation methods. Our method is agnostic to the underlying semantics and produces geometrically consistent segmentations with piecewise

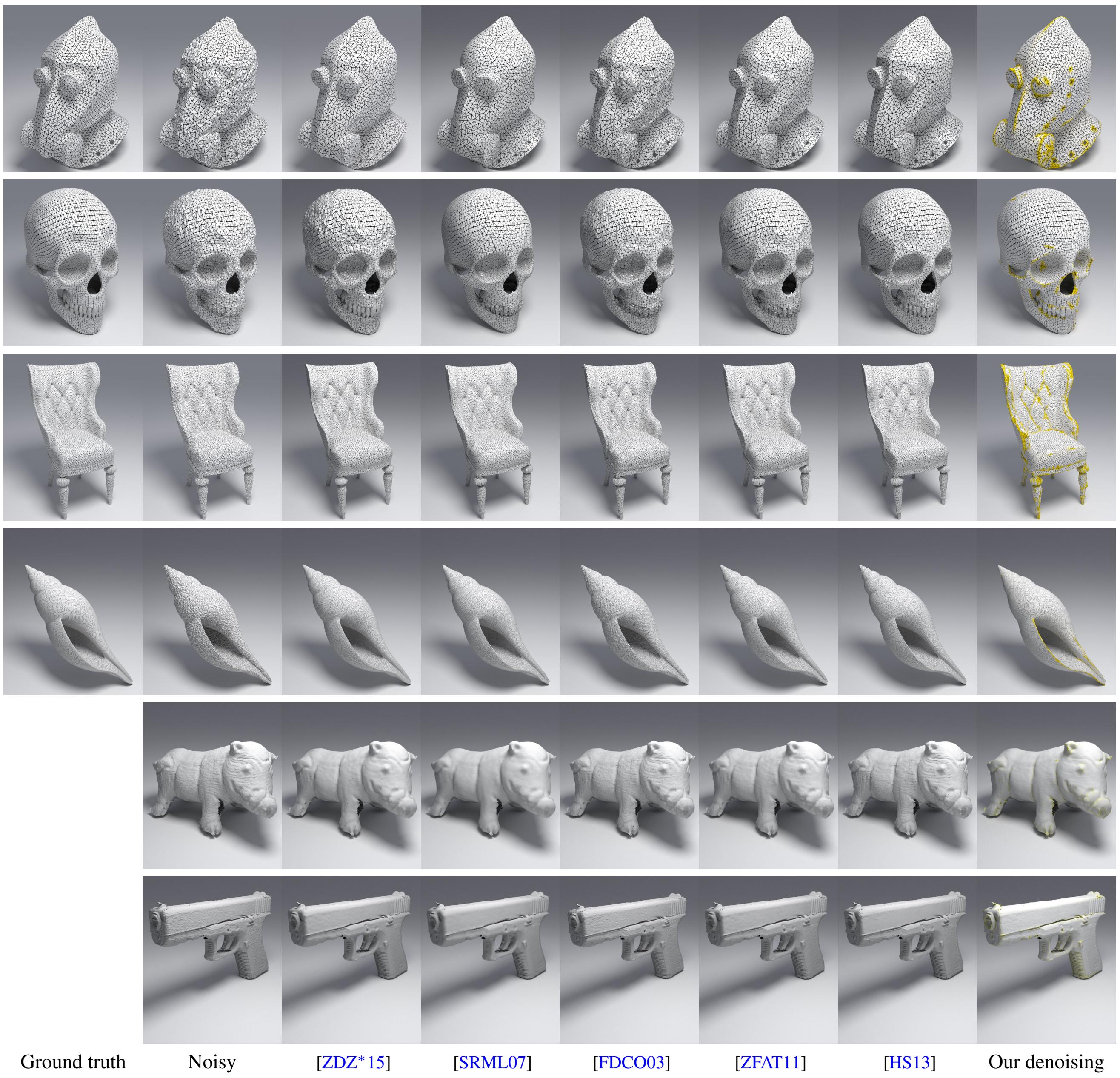


Figure 5: *Denoising results and comparisons. For our method, features (v) are shown in yellow. The last two rows illustrate denoising on LiDAR acquired geometries where no ground-truth information is available.*

smooth parts minimizing a well-defined AT energy, while semantic segmentation methods often produce semantically meaningful parts from heuristics or learning from large manually segmented databases [CGF09].

5.3. Inpainting

Inpainting seeks to fill in missing areas of a mesh. To perform mesh inpainting, one should first triangulate the missing area. The first step is to recover the topology of the missing area, which we perform using the hole filling algorithm [Lie03] implemented in CGAL [FP09]. Then, we set the data attachment terms α and w_2 to a large value (typically, $\alpha = w_2 = 1000$) outside of the inpainted area to prevent known vertices from moving, and w_2 to zero inside. Since the hole filling tends to oversmooth the inpainted area, we set λ to one tenth of its original value inside the inpainted area, hence favoring longer features. We finally solve the MS problem and conform the mesh to the obtained normals. Inpainting results are shown in Fig. 11 and Hausdorff distance evaluation is given in Table 2.

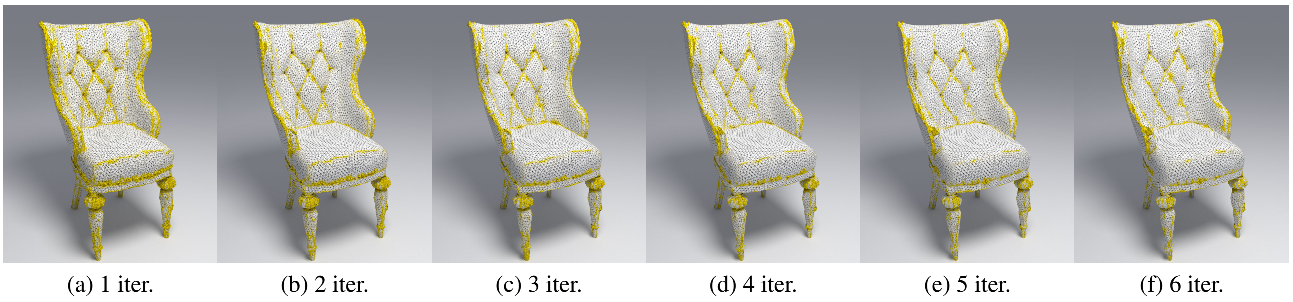


Figure 6: *Influence of the number of MS / vertex projection iterations.*

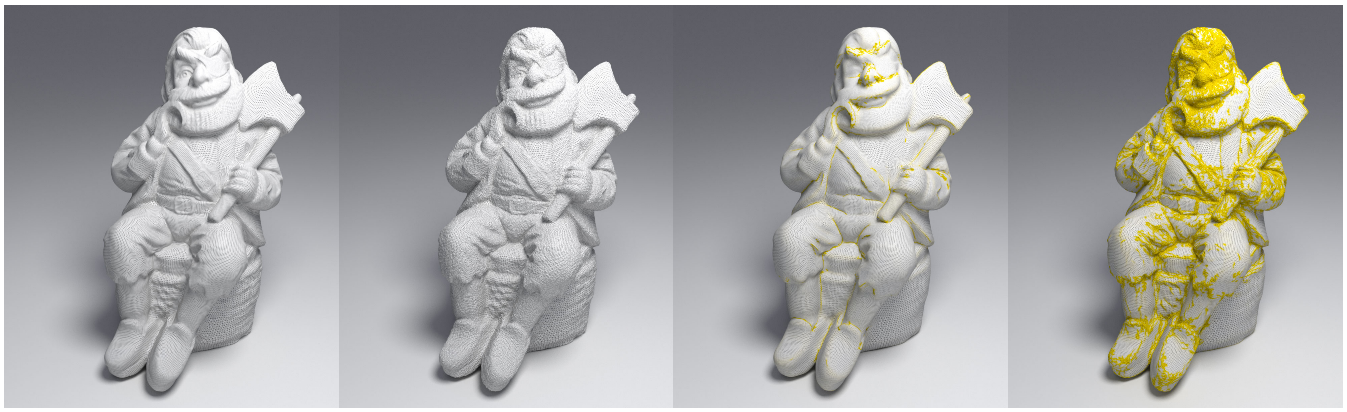


Figure 7: *Influence of λ on denoising results. First row: Original and noisy models. Second row: denoising with $\lambda = 0.05$ (left) and $\lambda = 0.01$ (right). Decreasing λ produces longer features, and more discontinuities.*

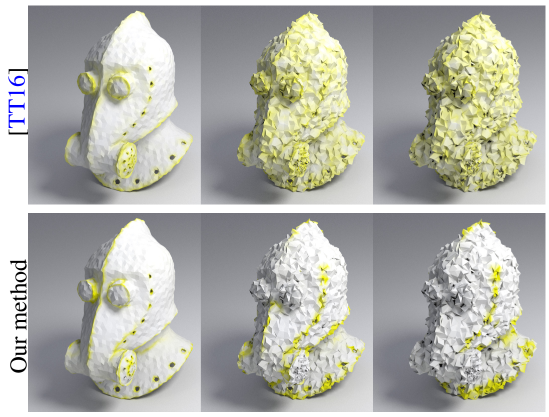


Figure 8: *We compare our method (bottom) to that of Tong and Tai [TT16] on meshes with weak, moderate and extreme noise levels. Our approach remains robust even to extreme noise levels.*

5.4. Embossing Normal Maps

Normal mapping is a useful technique for interactive applications, and allows for efficient low-polygon rendering by encoding geometric details as variations in normals stored in a high resolution texture. However, when it comes to realistic or physically-based rendering, this comes at the expense of several shortcomings. First, as these *shading* normal variations merely fake geometric details, silhouettes remain coarse, hence precluding renderings of close-ups. Second, as shading normals are not aligned with geometric normals, this can produce artifacts in physically-based renderers such as light leaks or black regions (Fig. 12, (b)), or artifacts due to the lack of energy conservation if not handled with particular care [Vea98, SHHD17]. Our projection step allows to recover finely detailed models for offline rendering from coarse assets along with their normal map. First, we increase the resolution of the input model (for subdivision surfaces, we apply a Catmull-Clark scheme). We then look up the normal value at each vertex from the normal map, and finally apply the projection step described in Sec. 4. Several embossing results are shown in Fig. 12.

6. Discussions

6.1. Implementation and Performance

We implemented our method in C++ using the DEC operators within the libDCC library [CdDS13]. Our implementation is mono-threaded, although fast parallel sparse linear solvers could be used instead [GJ*10]. Still, our unoptimized code already produces results in a reasonable amount of time. It processes meshes of nearly 300k triangles in less than 15 minutes, or in less than a

	Hausdorff Distance ($\times 100$)						Perceptual Metric					
	Z15	S07	F03	Z11	H13	Our	Z15	S07	F03	Z11	H13	Our
Chair	**0.0677**	0.1221	0.0762	0.0636	0.0786	0.1695	0.587	0.539	0.577	**0.531**	0.557	0.546
Skull	0.1082	0.1773	0.1180	**0.1017**	0.1078	0.1689	0.635	0.626	0.650	0.623	0.603	**0.592**
Helmet	0.1748	0.2595	0.1691	**0.1408**	0.1874	0.1963	0.577	0.524	0.596	0.510	0.558	**0.507**
Pirate	0.0853	0.1170	0.0898	0.0892	**0.0838**	0.0860	**0.277**	0.404	0.304	0.326	0.387	0.318
Shell	0.0386	0.0500	0.0641	**0.0331**	0.0464	0.1324	**0.346**	0.499	0.698	0.410	0.410	0.349

Table 1: *We provide numerical evaluation with state-of-the-art methods, with respect to the Hausdorff distance (RMS, normalized by the bounding box diameter), and the perceptual metric of Lavoué et al. [LGD*06] (ranks are given as cell colors from dark blue –rank 1– to light-blue –rank 6–). Our method performs better according to perceptual distances. This is due to a slight shrinkage of our denoising results which drastically impacts purely geometric metrics. Parameters were not adjusted to perform well under these metrics. Columns are named following: Z15 [ZDZ*15], S07 [SRML07], F03 [FDCO03], Z11 [ZFAT11], and H13 [HS13], and AT is our method.*

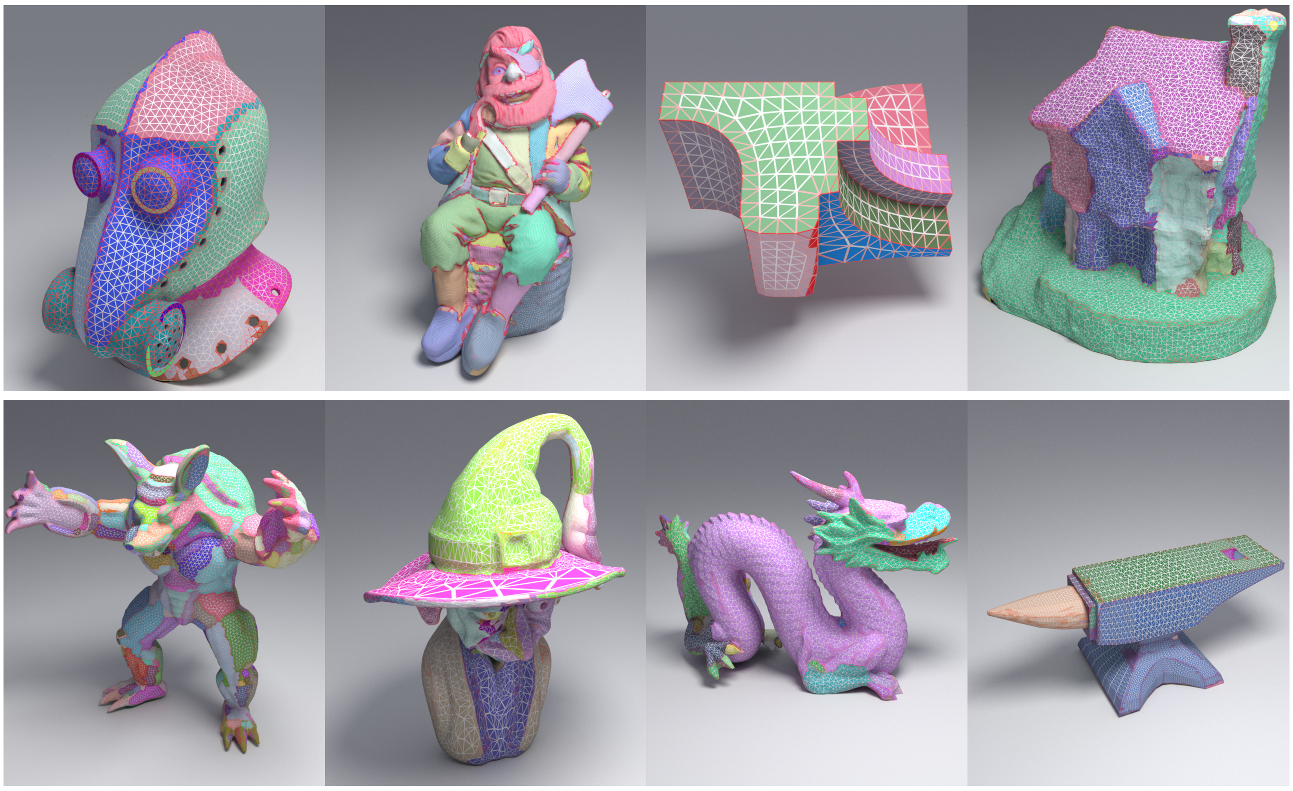


Figure 9: *Segmentation results. Edge splitting probabilities are shown in red.*

	CGAL	Our
Helmet	0.160	0.147
Knight	1.179	1.167
Fandisk	1.004	0.953
Witch	7.503	7.098
Anvil	7.910	5.126

Table 2: *Hausdorff distance ($\times 1000$) between the original shape and CGAL filling and our inpainting.*

minute if only the projection step is required (see Table. 3). Our implementation is available within our supplemental materials.

6.2. Limitations

First of all, to better capture the geometry, our approach requires to have meshes with enough triangles. For instance, low resolution shapes may lead to unrelevant segmentations (see Fig. 13, first row). For a more extreme case, since the feature vector v is represented as a 0−form we may highlight as feature all edges of a thin triangular strip (see Fig. 13, second row).

We also believe the normal map embossing would benefit from a mesh refinement strategy that adapts to the local frequency content of the normal map. Some details from the normal map indeed appear to be lost, and this is due to a too coarse underlying mesh at some places. This could be investigated in the future.

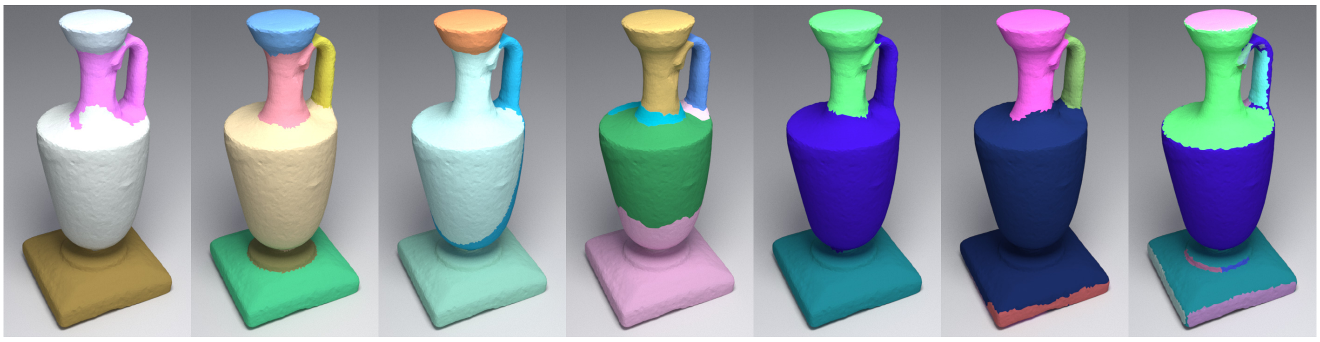

(a) Normalized Cuts [GF08]
(b) Randomized Cuts [GF08]
(c) Random Walks [LHMR08]
(d) Shape Diameter Function [SSCO08]
(e) Lifted Multicut [KLB*15]
(f) Projective ConvNet [KAMC17]
(g) Our result

Figure 10: *Comparison of semantic segmentations (a–f) with our geometric segmentation (g). While semantic segmentation produces parts of meaningful semantics such as the base, the neck, or the handle of this jar, our geometric segmentation detects sharp edges and results in piecewise smooth segments regardless of the underlying semantic.*

Model	# vertices	# triangles	MS (s)	Projection (s)
Helmet	10,022	20,247	28.9	2.5
Dragon	13,504	27,224	203	N/A
Witch	14,255	28,524	151	8
Anvil	25,202	50,400	273	N/A
Shell	30,036	60,068	120	10
Chair	57,933	115,910	214	15
Lion	60,066	120,128	248	14
Skull	81,051	162,120	315	18
Pirate	136,770	273,536	832	46
Boar	223,251	444,809	1,133	53
Gun	392,747	784,785	2,189	87
Skull (HR)	324,251	648,480	N/A	104
Armor	447,052	806,144	N/A	65

Table 3: *Performance of the MS solver and vertex projection steps with respect to the model complexity, using an unoptimized monothreaded implementation.*

Finally, we do not explicitly prevent triangle flips in our projection step (Sec. 4). If flipping occurs, other work have specifically formulated their variational problem to prevent flipping [ZDH*, WFL*15] and could be an inspiration for alleviating this issue.

6.3. Conclusions and Future Work

We have introduced a discretization of a functional for mesh processing along with a series of preliminary applications, which produces state-of-the-art results in mesh denoising, segmentation, inpainting and embossing. We leave as a future work a more extensive validation of each individual application, and rather emphasize the broad scope of our method. In particular, we expect the features detected by our Mumford-Shah discretization to be useful for feature sensitive remeshing [NLG15]. We also believe that stitching meshes together [SBSCO06] and mesh super-resolution [TYW01] could be performed similarly to our inpainting application. More generally, we expect the success of the MS model in geometry processing could parallel its success in image processing.

Acknowledgments

This work has been partly funded by COMEDIC ANR-15-CE40-0006 research grant.

References

[AKM*06] ATTENE M., KATZ S., MORTARA M., PATANÉ G., SPAGNUOLO M., TAL A.: Mesh segmentation-a comparative study. In *Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on* (2006), IEEE, pp. 7–7. 5

[AT90] AMBROSIO L., TORTORELLI V. M.: Approximation of functional depending on jumps by elliptic functional via Γ-convergence. *Comm. Pure and App. Math. 43*, 8 (1990). 2, 3

[BAS10] BEN-ARI R., SOCHEN N.: Stereo matching with Mumford-Shah regularization and occlusion handling. *IEEE Trans. on Pattern Analysis and Machine Intelligence 32*, 11 (2010), 2071–2084. 1, 3

[BC00] BOURDIN B., CHAMBOLLE A.: Implementation of an adaptive finite-element approximation of the mumford-shah functional. *Numerische Mathematik 85* (2000), 609–646. 2

[BDS*12] BOUAZIZ S., DEUSS M., SCHWARTZBURG Y., WEISE T., PAULY M.: Shape-up: Shaping discrete geometry with projections. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 1657–1667. 4

[Ber99] BERTSEKAS D. P.: *Nonlinear programming*, 2nd ed. Athena scientific, 1999. 4

[BSK06] BAR L., SOCHEN N., KIRYATI N.: Semi-blind image restoration via mumford-shah regularization. *IEEE Trans. on Image Proc. 15*, 2 (2006). 1, 3

[CdDS13] CRANE K., DEGOES F., DESBRUN M., SCHRÃŰDER P.: Digital geometry processing with discrete exterior calculus. In *ACM SIGGRAPH 2013 courses* (2013). 4, 7

[CDM99] CHAMBOLLE A., DAL MASO G.: Discrete approximation of the mumford-shah functional in dimension two. *Mathematical Modelling and Numerical Analysis 33*, 4 (1999), 651–672. 2

[CEN06] CHAN T. F., ESEDOGLU S., NIKOLOVA M.: Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM Journal of Applied Mathematics 66*, 5 (2006), 1632–1648. 2

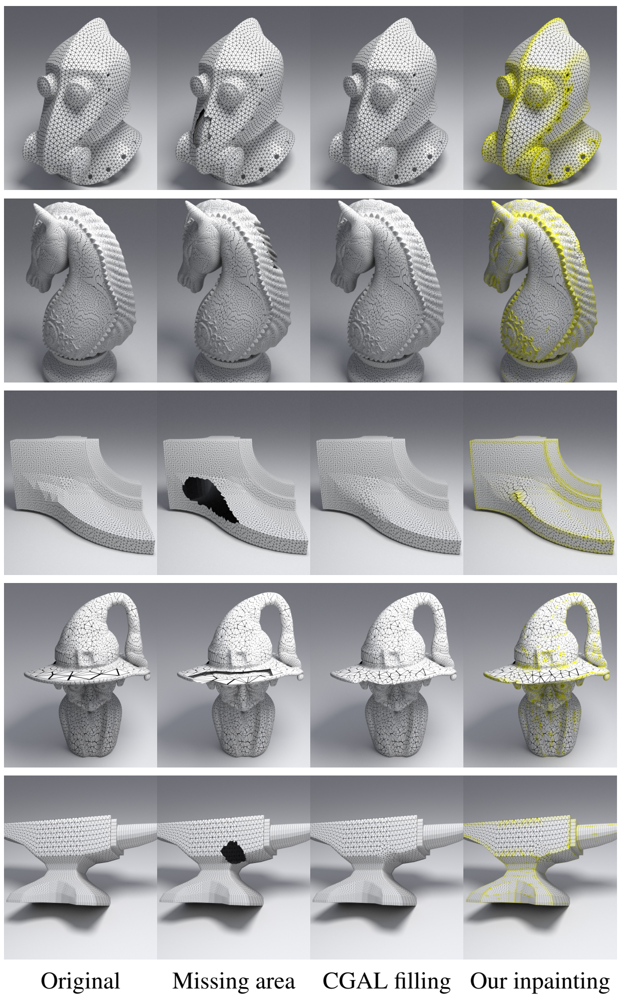


Figure 11: *Inpainting results. The recovered feature field v is shown in yellow.*

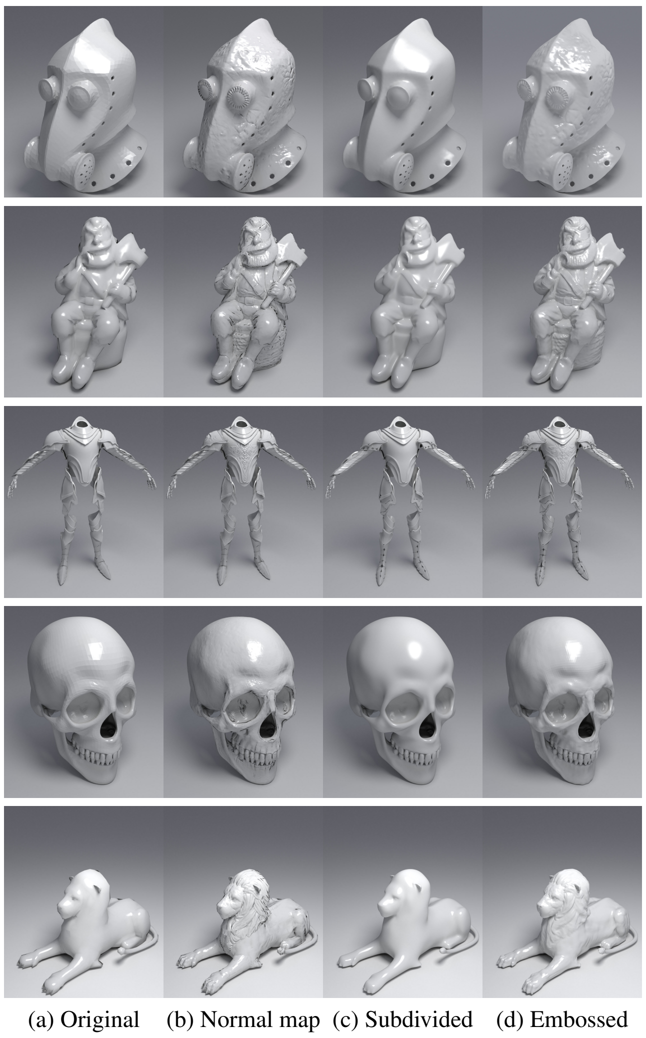


Figure 12: *Embossing results. We subdivide (c) the input model (a), and emboss the normal map (c) onto the subdivided model (d). When shading normals are not aligned with geometric normals, physically-based renderers can produce artifacts such as black regions (b) if not handled with particular care.*

[CFGL16] Coeurjolly D., Foare M., Gueth P., Lachaud J.: Piecewise smooth reconstruction of normal vector field on digital data. *Comput. Graph. Forum (Pacific Graphics) 35*, 7 (2016), 157–167. 2, 3

[CGF09] Chen X., Golovinskiy A., Funkhouser T.: A benchmark for 3d mesh segmentation. In *ACM Trans. Graph.* (2009), vol. 28, ACM, p. 73. 5, 6

[CSAD04] Cohen-Steiner D., Alliez P., Desbrun M.: Variational shape approximation. *ACM Trans. Graph. (SIGGRAPH)* (2004). 3

[ES02] Esedoglu S., Shen J.: Digital inpainting based on the Mumford-Shah-Euler image model. *Eur. J. Appl. Math. 13* (2002), 353–370. 1, 2

[FDCO03] Fleishman S., Drori I., Cohen-Or D.: Bilateral mesh denoising. In *ACM Trans. Graph.* (2003), vol. 22. 4, 6, 8

[FI14] Focardi M., Iurlano F.: Asymptotic analysis of ambrosio–tortorelli energies in linearized elasticity. *SIAM Journal on Mathematical Analysis 46*, 4 (2014), 2936–2955. 2, 3

[FLT16] Foare M., Lachaud J.-O., Talbot H.: Image restoration and segmentation using the ambrosio-tortorelli functional and discrete calculus. In *Pattern Recognition (ICPR), 2016 23rd International Conference on* (2016), IEEE, pp. 1418–1423. 2

[FP09] Fabri A., Pion S.: Cgal: The computational geometry algorithms library. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (2009), ACM, pp. 538–539. 6

[GF08] Golovinskiy A., Funkhouser T.: Randomized cuts for 3d mesh analysis. In *ACM Trans. Graph.* (2008), vol. 27, p. 145. 9

[GJ*10] Guennebaud G., Jacob B., et al.: Eigen v3. http://eigen.tuxfamily.org, 2010. 7

[Hir03] Hirani A. N.: *Discrete Exterior Calculus.* PhD thesis, Pasadena, CA, USA, 2003. AAI3086864. 2

[HS13] He L., Schaefer S.: Mesh denoising via l 0 minimization. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 64. 6, 8

[KAMC17] Kalogerakis E., Averkiou M., Maji S., Chaudhuri S.: 3d shape segmentation with projective convolutional networks. *Proc. CVPR, IEEE 2* (2017). 9

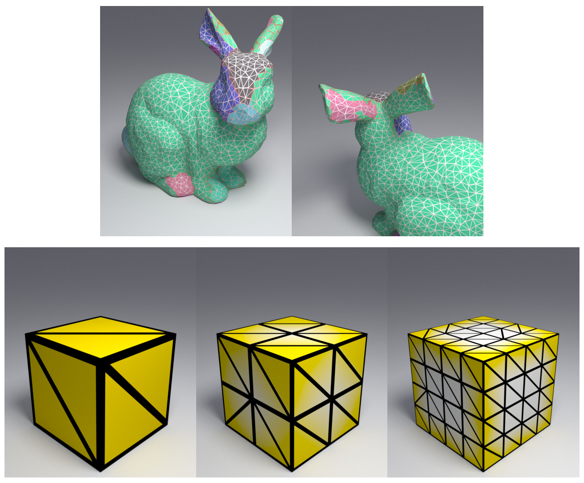


Figure 13: *First row: multicut segmentation fails to capture the bunny ears geometry. Second row: a coarse mesh does not capture features (in yellow) computed per vertex accurately. As we subdivide the mesh, features become more accurate.*

[KLB*15] Keuper M., Levinkov E., Bonneel N., Lavoué G., Brox T., Andres B.: Efficient decomposition of image and mesh graphs by lifted multicuts. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 1751–1759. 5, 9

[LGD*06] Lavoué G., Gelasca E. D., Dupont F., Baskurt A., Ebrahimi T.: Perceptually driven 3d distance metrics with application to watermarking. In *SPIE Optics+ Photonics* (2006), International Society for Optics and Photonics, pp. 63120L–63120L. 8

[LHMR08] Lai Y.-K., Hu S.-M., Martin R. R., Rosin P. L.: Fast mesh segmentation using random walks. In *Proc. ACM Symp. on Solid and Physical Modeling* (2008), pp. 183–191. 9

[Lie03] Liepa P.: Filling holes in meshes. In *Eurographics/ACM SIGGRAPH Symp. on Geometry Proc.* (2003), pp. 200–205. 6

[MS89] Mumford D., Shah J.: Optimal approximation by piecewise smooth functions and associated variational problems. *Commun. Pure Applied Mathematics* (1989), 577–685. 1, 2

[NLG15] Nivoliers V., Lévy B., Geuzaine C.: Anisotropic and feature sensitive triangular remeshing using normal lifting. *Journal of Computational and Applied Mathematics 289* (2015), 225–240. 9

[PAH*07] Pottman H., Asperl A., Hofer M., Kilian A., Bentley D.: *Architectural geometry*, vol. 724. Bentley Institute Press Exton, 2007. 4

[PBB11] Pokrass J., Bronstein A. M., Bronstein M. M.: A correspondence-less approach to matching of deformable shapes. In *Int. Conf. Scale Space and Variational Meth. in Comp. Vis.* (2011). 2

[RNLL10] Ray N., Nivoliers V., Lefebvre S., Lévy B.: Invisible seams. In *Comp. Graph. Forum* (2010), vol. 29. 5

[SBSCO06] Sharf A., Blumenkrants M., Shamir A., Cohen-Or D.: Snappaste: an interactive technique for easy mesh composition. *The Visual Computer 22*, 9 (2006), 835–844. 9

[SHHD17] Schüssler V., Heitz E., Hanika J., Dachsbacher C.: Microfacet-based normal mapping for robust Monte Carlo path tracing. *ACM Trans. Graph. (SIGGRAPH Asia) 36*, 6 (2017). 7

[SRML07] Sun X., Rosin P., Martin R., Langbein F.: Fast and effective feature-preserving mesh denoising. *IEEE transactions on visualization and computer graphics 13*, 5 (2007). 4, 6, 8

[SSCO08] Shapira L., Shamir A., Cohen-Or D.: Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer 24*, 4 (2008), 249. 9

[Tau01] Taubin G.: Linear anisotropic mesh filtering. *Res. Rep. RC2213 IBM 1*, 4 (2001). 4

[TT16] Tong W., Tai X.: A variational approach for detecting feature lines on meshes. *J. of Comp. Math. 34*, 1 (2016). 1, 2, 3, 4, 5, 7

[TYW01] Tsai A., Yezzi A., Willsky A. S.: Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification. *IEEE Trans. on Image Processing 10*, 8 (2001), 1169–1186. 1, 2, 3, 9

[VC02] Vese L. A., Chan T. F.: A multiphase level set framework for image segmentation using the Mumford and Shah model. *International Journal of Computer Vision 50*, 3 (Dec 2002), 271–293. 1, 2

[Vea98] Veach E.: *Robust Monte Carlo methods for light transport simulation*. Stanford University Stanford, 1998. 7

[WFL*15] Wang P.-S., Fu X.-M., Liu Y., Tong X., Liu S.-L., Guo B.: Rolling guidance normal filter for geometric processing. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 173. 9

[YZX*04] Yu Y., Zhou K., Xu D., Shi X., Bao H., Guo B., Shum H.-Y.: Mesh editing with poisson-based gradient field manipulation. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 644–651. 3

[ZDH*] Zhang J., Deng B., Hong Y., Peng Y., Qin W., Liu L.: Static/dynamic filtering for mesh geometry. *IEEE Transactions on Visualization & Computer Graphics*, 1, 1–1. 9

[ZDZ*15] Zhang W., Deng B., Zhang J., Bouaziz S., Liu L.: Guided mesh normal filtering. *Computer Graphics Forum (Pacific Graphics) 34* (2015). 6, 8

[ZFAT11] Zheng Y., Fu H., Au O. K.-C., Tai C.-L.: Bilateral normal filtering for mesh denoising. *IEEE Transactions on Visualization and Computer Graphics 17*, 10 (2011), 1521–1530. 6, 8

[Zor05] Zorin D.: Curvature-based energy for simulation and variational modeling. In *International Conference on Shape Modeling and Applications 2005 (SMI' 05)* (June 2005), pp. 196–204. 3

[ZWZD15] Zhang H., Wu C., Zhang J., Deng J.: Variational mesh denoising using total variation and piecewise constant function space. *IEEE transactions on visualization and computer graphics 21*, 7 (2015), 873–886. 3

[ZZWC12] Zhang J., Zheng J., Wu C., Cai J.: Variational mesh decomposition. *ACM Trans. Graph. 31*, 3 (2012), 21:1–21:14. 1, 2, 3