Robust diameter-based thickness estimation of 3D objects

Rolland-Nevière Xavier, Gwenaël Doërr, Pierre Alliez

Robust Diameter-Based Thickness Estimation of 3D Objects

Xavier Rolland-Nevière[a,*], Gwenael Doërr[a], Pierre Alliez[b]

[a]*Technicolor R&D France, Cesson-Sévigné, France*
[b]*Inria Sophia Antipolis - Méditerranée, France*

Abstract

We propose a robust thickness estimation approach for 3D objects based on the Shape Diameter Function (SDF). Our method first applies a modified strategy to estimate the local diameter with increased accuracy. We then compute a scale-dependent robust thickness estimate from a point cloud, constructed using this local diameter estimation and a variant of a robust distance function. The robustness of our method is benchmarked against several operations such as remeshing, geometric noise and artifacts common in triangle soups. The experimental results show a more stable local thickness estimation than the original SDF, and consistent segmentation results on defect-laden inputs.

Keywords: 3D Object, Surface, Mesh, Thickness, Robustness, Shape Diameter Function.
2000 MSC: 68U05, 68W25, 68U20, 65D18

1. Introduction

Estimating the local thickness of complex 3D objects is a multi-faceted problem with a variety of applications. In computer graphics, algorithms such as mesh partitioning or curve skeleton extraction can successfully rely on a local thickness estimate such as the so-called *Shape Diameter Function* (SDF) [1]. In these applications, the robustness of the local thickness estimate is still a challenge. In this paper the notion of robustness relates to the resilience of the thickness estimate to both editing and processing operations applied to the shape, as well as to defect-laden inputs.

1.1. Robustness against operations

A shape can be altered to meet the limited computational capabilities of heterogeneous computer hardware, by, e.g., matching a target level of detail. Assuming an input shape provided as a surface mesh, this goal commonly involves processing operations such as mesh simplification [2]. In this context, any thickness estimate should ideally be consistent for all levels of detail. A shape can also be animated, involving complex distortions. We expect that articulated animations have only minor effects on the thickness overall, as changes only occur at the joints which are in general a small subset of a shape. In this context, a local thickness estimate should ideally be consistent across all poses of an animation.

1.2. Robustness against artifacts

When digitizing, the original physical shape is only known through sampling and approximation. A triangle mesh is an instance of such piecewise-linear approximation of a surface. In addition to the inherent uncertainty of any measurement device and imperfections of the acquisition process, some imperfect algorithms along the geometry processing pipeline may produce a range of artifacts such as gaps, holes, non-manifold parts and triangle soups. While a thread of research has focused on repairing defect-laden data or removing artifacts, there is currently no definitive solution to such defects [3]. Moreover, some applications gather data from heterogeneous inputs and thus require converting between shape representations. These conversions also lead to artifacts such as handles or disconnected components. Ideally, a thickness estimation would provide results that are both robust and consistent for all these cases.

2. Related Work

One definition of the local thickness of a 3D shape is based on its medial axis transform (MAT). The MAT was initially introduced to represent 2D shapes through the loci of maximally inscribed circles [4]. In 3D the medial axis $\mathcal{M}$ is defined as the loci of centers of maximally inscribed spheres. The MAT of a 3D object is defined from the medial axis and the set of sphere radii, which defines a scalar field onto $\mathcal{M}$. On the boundary of a smooth 3D object each point has a unique corresponding point on the medial axis, which is the center of the maximally inscribed sphere tangent to the boundary point. Through this correspondence one can map the radii of the spheres centered at the medial axis onto the surface, thus defining the local thickness for each boundary point [5] (Figure 1). Extracting the medial axis of a surface is complex [6] however, as it is very sen-

*Technicolor R&D France, 975, avenue des Champs Blancs - CS 17616, 35576 Cesson-Sévigné, FRANCE - Phone: +33299273052
Email addresses: Xavier.Rolland-Neviere@technicolor.com (Xavier Rolland-Nevière), Gwenael.Doerr@technicolor.com (Gwenael Doërr), pierre.alliez@inria.fr (Pierre Alliez)

sitive to small variations of the surface. This issue is critical when dealing with defect-laden inputs as small irregularities on a smooth surface may create large spikes on the medial axis. To alleviate this issue some robust variants taking advantage of the notion of scale have been proposed [7].

An approach to compute an intuitive and pose-invariant local thickness from surface meshes was explored with the *Shape Diameter Function* (SDF), based on statistics upon local diameter estimates. Given a query boundary point $\mathbf{q}$, a single local diameter estimate is defined as the length of the segment joining $\mathbf{q}$ and the first intersection between the input input mesh and a ray shot from $\mathbf{q}$ and aligned to a vector located inside an inward cone. The diameter does not rely on computing the medial axis in order to alleviate the aforementioned issue. It is experimentally shown that the SDF is stable with respect to articulated deformations and provides an effective means to consistently partition surface meshes over multiple poses. Its robustness with respect to noise and defect-laden inputs can however be improved. The original SDF has also been improved in terms of computational complexity by performing down-sampling followed by efficient interpolation [8]. This procedure improves the computational time of the original SDF, at the cost of a lower robustness for segmentation.

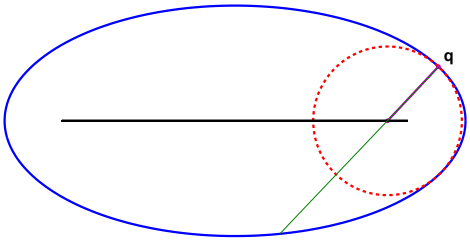


Figure 1: **Thickness on an ellipse.** The medial axis is depicted in black. The thickness (purple line) is defined as the radius of the maximal inscribed ball (red dotted line) associated to a boundary point $\mathbf{q}$. The diameter is depicted in green. Notice that taking half the diameter value is different from the computing the thickness, as it yields a larger value.

Contributions

Extending the SDF approach, we propose a robust method to estimate the local thickness of a 3D object bounded by a surface mesh. Inspired by ideas introduced for robust medial axis extraction, we devise a scale-dependent estimation method. As contributions we (i) improve the accuracy of the original SDF, and (ii) provide several experimental evidences that illustrate the robustness of the proposed approach. These results also show benefits for robust shape segmentation. In the following the term "thickness" relates to our robust scale-dependent diameter-based thickness, while "mathematical thickness" relates to the radius of the maximal inscribed sphere (MAT).

3. Algorithm

The input to our algorithm is a surface triangle mesh - denoted by $\mathcal{S}$ - bounding the input 3D object. $\mathcal{S}$ may contain defects such as noise or holes. The algorithm for computing the thickness of $\mathcal{S}$ comprises two main steps. First, we compute a cloud $\mathcal{D}$ of half-diameter points $\mathbf{d}_i$ (Section 3.1). This step extends a curve skeleton extraction technique [1] originally presented as a direct extension of the SDF computation. Second, we define a robust scale-dependent thickness function t_k (defined for arbitrary query points on $\mathcal{S}$) using a noise- and outlier-robust distance function between each query point $\mathbf{q}_i$, and the half-diameter point cloud (Section 3.2).

Algorithm 1 provides a general overview of our method.

1: **procedure** THICKNESS(Input mesh $\mathcal{S}$; sampling size n; boundary point queries Q; scale parameter k)
2: Random sampling of $\mathcal{S}$ with n points
3: **for all** Sample points $\mathbf{p}_i$ **do**
4: Probe mesh volume at sample $\mathbf{p}_i$
5: Compute a local estimation of the diameter
6: Create a half-diameter point $\mathbf{d}_i$ and add it to $\mathcal{D}$.
7: **end for**
8: **for all** Query points $\mathbf{q}_i$ in Q **do**
9: Search appropriate k nearest neighbors $\mathbf{d}_i$ in $\mathcal{D}$.
10: Compute robust distance function
11: **return** scale-dependent thickness t_k
12: **end for**
13: **end procedure**

Algorithm 1: Overview of our thickness estimation algorithm.

3.1. Half-Diameter Points

The procedure for generating half-diameter points is summarized by Algorithm 2. The main input to the algorithm is a surface triangle mesh $\mathcal{S}$, which is first uniformly point sampled in order to generate a set of boundary points $\mathcal{P}$. Section 4.1.1 provides implementation details on this sampling step.

Given a boundary sample point $\mathbf{p}_i$, the original SDF is computed by: (i) casting random rays inside an inward-oriented cone along the normal and computing their intersection with the input surface $\mathcal{S}$, which defines a series of segments; (ii) measuring statistics δ based on a weighted average and the variance of the lengths of these segments; and (iii) smoothing the final thickness function over a small neighborhood and normalizing the output in log-space [1].

While this algorithm yields a sufficiently discriminative diameter estimate for mesh segmentation, we introduce several methodological differences to improve its accuracy and robustness, starting from the random re-sampling of $\mathcal{S}$.

3.1.1. SDF Ray-casting Strategy

Although effective when dealing with normal distributions of lengths, the outlier-robust statistics δ computed in the SDF often yields counter-intuitive estimations of the diameter. Consider the following dummy example: with two (infinite) parallel planes, the diameter can be estimated exactly and is equal to (twice) the mathematical thickness. Most importantly, this configuration does not involve scale-dependent quantities. Therefore, any thickness computation algorithm should provide an

estimate as close as possible to half the diameter value. However, averaging multiple lengths inside a cone systematically overestimates the diameter. When dealing with mechanical parts and piecewise flat surfaces, this situation is quite common and δ always over-estimates th expected value. A more accurate value for δ would then be (half) the minimum length of the rays cast inside the cone.

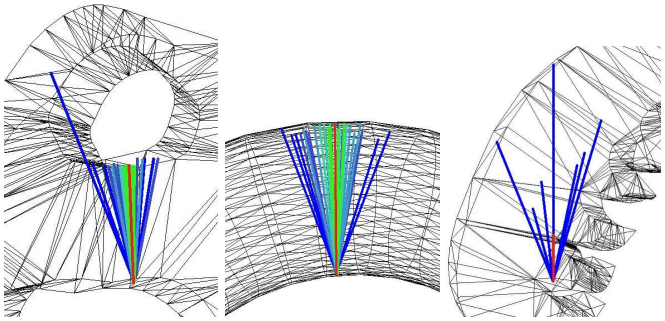
In the case of a tubular section with a circular cut, the mathematical thickness also coincides with half the length of a ray cast along the normal. This value is however not given by the minimum lengths of all possible random rays cast inside the cone. At the bifurcation of a Y-shape, the larger the opening of the cone, the more noise is added to the diameter estimation δ when taking an average ray length. In this case, notice that the outlier-robust strategy of the SDF, which is based on the median ray length, is inappropriate: the closest value to the mesh diameter (and the mathematical thickness) is given by a single minimal estimate among many noisy larger ones.

In all these examples, the larger the half-opening ϕ of the cone, the more δ differs from the mathematical thickness or from the mesh diameter. Nevertheless, a large opening angle is necessary to capture more information on the shape and to detect a masking feature such as protuberances or dents depicted by Figure 2(c). All possible configurations are not addressed when designing statistics to compute δ from the R ray lengths. However, in cases where the notion of scale is not needed, and when the mathematical thickness and the mesh diameter are identical, the accuracy and relevance of δ can be measured by its closeness to, e.g., the mathematical thickness chosen as "ground-truth".

3.1.2. Adaptive Ray Casting

To alleviate the dilemma between a large and a small opening angle of the cone to probe the local volume, we adopt an adaptive method when casting the rays inside the cone. Such a method provides a more conservative estimation of the diameter than the SDF. The aperture angle is initially set to a large value ϕ, and R rays are cast inside this cone. Let $l_{\min}$ be the minimum ray length. This ray casting procedure is iteratively repeated while decreasing the opening angle by a step η, yielding each time a new length $l_{\min}$. At each step the stability of $l_{\min}$ is estimated by computing its absolute growth rate r (see Algorithm 2). If r is valued above a threshold τ, the process ends and δ is set to the previous $l_{\min}$. Otherwise, δ is set to a final $l_{\min}$ after I iterations. In practice, we set $\phi = 25°$, $R = 5$, $\eta = 2°$, $\tau = 0.8$ and $I = 10$. Thus, the procedure either ends with a variation of $l_{\min}$ larger than 80%, or when the aperture of the cone reaches $\phi - I\eta = 5°$.

The results of this procedure are illustrated by Figure 2. In Figure 2(a), δ is eventually set to half the length given by the ray cast along the normal. In Figure 2(b), the tubular section has a circular cut and our procedure ensures that the half-opening of the cone is very small when estimating the final $l_{\min}$, thus improving the accuracy of the estimate: with the previously described parameter settings, $r < \tau$ holds true until ϕ reaches its minimum value of 5°. Figure 2(c) shows that the adaptive closing algorithm stops when a large variation in the minimum ray length is detected. In this particular configuration, our procedure ensures that δ does not get too large.

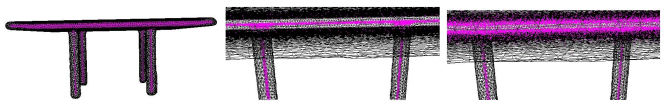


(a) Parallel planes (b) Torus (c) Dent

Figure 2: **Adaptive cone opening.** Illustrations of 3 specific configurations. Rays cast inside the adaptive cones are shown with different colors. Blue rays correspond to the first iterations (larger cones), green rays to the last iterations (smaller cones). The (double) diameter estimation δ is depicted as a red segment along the normal. 2(a) and 2(b): the adaptive closing reaches a very small opening angle, which improves the estimation of the diameter, as the average length of the deep blue rays would not provide such an accurate estimate. 2(c): the algorithm stops after the first iteration, as a large variation in the minimum ray length is detected.

3.1.3. Half-Diameter Points Construction

We create the half-diameter cloud, a point set approximating the middle of the shape using the diameter information δ (Figure 3). This step is similar to the one proposed for extracting a curve skeleton [1]: the sample points are projected into the shape using their normal direction at half δ, thus creating the point set denoted by $\mathcal{D}$.



(a) Table mesh (b) Adaptive cone (c) SDF

Figure 3: **Half-diameter cloud.** 3(a): half-diameter points for a mesh of a table. 3(b): close-up on the points (purple), representing projections of samples at half their estimated δ value along the normal, using the adaptive-opening cone algorithm. This configuration corresponds to the parallel planes case. 3(c): same close-up, but points are the centers of facets projected at half their SDF value. In this case, two distinct parallel planes of points are created, due to the systematic overestimation of the actual diameter.

3.2. Robust Thickness Estimation t_k

Another difference between the proposed thickness approximation and the SDF is that none of its post-processing operations (the bilateral smoothing and the normalization) are performed on δ. These operations were initially designed to counterbalance the variations due to the pose. Instead, our approach uses a variant of a robust distance function [9] to compute a scale-dependent thickness t_k from the point cloud $\mathcal{D}$. Such an approach addresses both the issue of robustness and the constraint of having a single and intuitive parameter.

procedure HALF-DIAMETER POINTS(Input mesh $\mathcal{S}$; boundary point queries $\mathcal{Q}$; sampling size n; cone aperture ϕ; rays R; iterations I; step η; variation threshold τ)
  for all facets in $\mathcal{S}$ **do**
    Generate (uniformly) a sampling set $\mathcal{P}$
    for all $\mathbf{p}_i \in \mathcal{P}$ **do**
      $r \leftarrow 0, j \leftarrow 0$
      Cast R random rays from $\mathbf{p}_i$ inside cone $(\mathbf{p}_i, \mathbf{n}_{\mathbf{p}_i}, \phi)$ ▷ Cone defined by apex $\mathbf{p}_i$, direction as normal at $\mathbf{p}_i$, aperture denoted by ϕ.
      $l^B_{\text{min}} \leftarrow$Compute minimum ray lengths
      while $j < I$ **do**
        $j \leftarrow j + 1, \phi \leftarrow \phi - \eta$
        Cast R rays from $\mathbf{p}_i$ inside cone $(\mathbf{p}_i, \mathbf{n}_{\mathbf{p}_i}, \phi)$
        $l_{\text{min}} \leftarrow$Compute minimum ray lengths
        $r \leftarrow |l^B_{\text{min}} - l_{\text{min}}|/l^B_{\text{min}}$
        if $r > \tau$ **then**
          break
        else
          $l^B_{\text{min}} \leftarrow l_{\text{min}}$
        end if
      end while
      Add projection of $\mathbf{p}_i$ along $\mathbf{n}_{\mathbf{p}_i}$ at $\frac{1}{2} l^B_{\text{min}}$ to $\mathcal{D}$
    end for
  end for
  return Set of half-diameter points $\mathcal{D}$ ($|\mathcal{D}| = n$)
end procedure

Algorithm 2: Half-diameter points computation.

3.2.1. *Defining the thickness t_k*

Let $\mathcal{D}$ be the set of half-diameter points, computed from the input surface mesh $\mathcal{S}$, and $n = |\mathcal{D}|$ (which equals the sampling size). $\mathcal{Q}$ denotes the set of query points where the thickness is estimated. Given $k \in [\![1; n]\!]$, the thickness $t_k(\mathbf{q})$ ($\mathbf{q} \in \mathcal{Q}$) is defined on the input surface as follows:

$$\forall \mathbf{q} \in \mathcal{Q},\ t_k(\mathbf{q}) = \sqrt{\frac{1}{k} \sum_{i \in [\![1,k]\!]} \|\mathbf{q}\mathbf{d}_i\|^2} \tag{1}$$

where $\mathbf{d}_i$ denotes the i^{th} closest point to $\mathbf{q}$ in $\mathcal{D}$ that verifies the condition:

$$\mathcal{S} \cap [\mathbf{q}\mathbf{d}_i] = \{\mathbf{q}\} \tag{2}$$

Equation (2) ensures that half-diameter points and boundary query points are mutually visible. This is required to avoid the issue depicted by Figure 4.

3.2.2. *Computing t_k*

We compute $t_k(\mathbf{q})$ by iteratively retrieving the next closest point $\mathbf{d}_i$ ($\mathbf{d}_i \in \mathcal{D}$) to $\mathbf{q}$. If Equation (2) holds true for $\mathbf{d}_i$, we increment i and add $\|\mathbf{q}\mathbf{d}_i\|^2$ to $t_k(\mathbf{q})$. This procedures either ends when i reaches k or when all points in $\mathcal{D}$ have been queried. In Equation (1), $\frac{1}{k}$ is replaced by $\frac{1}{v(\mathbf{q})}$, where $v(\mathbf{q})$ is the number of points in $\mathcal{D}$ which are visible from $\mathbf{q}$.

The main issue to compute t_k is related to the computational efficiency: if $k > v(\mathbf{q})$, computing the thickness involves useless queries to all points in $\mathcal{D}$. Although there is no way to exactly estimate $v(\mathbf{q})$ without testing all half-diameter points, we introduce two tests to speed up the computation: a preliminary check and an upper-bound to the neighbor search.

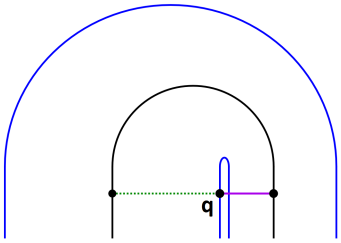


Figure 4: **Visibility issue between a boundary point and the half-diameter point cloud.** This dummy shape (blue) is formed by two parallel tubes joined by a circular connection. Notice that, at query point $\mathbf{q}$, directly using the nearest points in the point cloud (black) implies measuring the purple solid line, which yields an irrelevant thickness estimation. Instead, the furthest visible points (green dotted line) should be used to estimate t_k.

Before performing any other computation, we check:

$$\mathbf{q}\mathbf{d}_i \cdot \mathbf{n}_{\mathbf{q}} \geq 0$$

where $\mathbf{n}_{\mathbf{q}}$ denotes the normal at $\mathbf{q}$ oriented inward. This condition is always verified by points visible from $\mathbf{q}$. Since this test is substantially faster than a ray-shape intersection query, it generally improves the efficiency of the process.

We also add an upper-bound d_{max} to the acceptable distance $\|\mathbf{q}\mathbf{d}_i\|$, thus limiting the neighbor search to a sphere of radius d_{max} centered at $\mathbf{q}$. When reaching a point $\mathbf{d}_i$ which does not verify Equation (2), the computation of t_k terminates if $\|\mathbf{q}\mathbf{d}_i\| \geq d_{\text{max}}$. Since the upper-bound is only applied when the search returns unusable points, half-diameter points outside the radius d_{max} may still be used in $t_k(\mathbf{q})$. This heuristic is automatically dropped when the first point in $\mathcal{D}$ visible from $\mathbf{q}$ is farther than d_{max}, so that t_k remains well-defined.

Empirically, d_{max} is set to $\frac{1}{5}$ of the largest diagonal of the bounding box. In practice, this heuristic roughly speeds up the computation by one order of magnitude. Algorithm 3 details the work flow of this algorithm, whose output t_k is scale-dependent.

3.2.3. *Properties of t_k*

For a given size of the sampling set n, the parameter k provides the user a means to trade robustness for discriminative capability of the thickness estimate. Large values of k increase the robustness through a lower sensitivity to outliers, but they also mean a low influence of small scale features since they are considered outliers compared to large scale features. In a nutshell, the scale of the thickness estimation is entirely controlled by the $\frac{k}{n}$ ratio.

Figure 5 illustrates this behavior for the Hippo model and a fixed value of n. With a low k value, the importance of small scale features is enhanced for the toes and ears. This is indicated by the deep blue parts on the mesh (deep blue patches correspond to small values of t_k). With larger values of k, small scale features become less significant, and larger parts, such as the torso, become more important: deep blue parts on the mesh

procedure DiameterToThickness(Input mesh $\mathcal{S}$; half-diameter points $\mathcal{D}$; boundary point query $\mathbf{q}$, scale parameter k, upper-bound $d_{\max}$)
  $t_k \leftarrow 0$, $j \leftarrow 0$
  while $j < k$ AND $\mathcal{D} \neq \emptyset$ **do**
    Pop next nearest neighbor $\mathbf{d}_i \in \mathcal{D}$ from $\mathbf{q}$
    if $\mathbf{n_q} \cdot \mathbf{qd}_i \geq 0$ **then** ▷ $\mathbf{n_q}$ is the normal at $\mathbf{q}$.
      if $\mathcal{S} \cap [\mathbf{qd}_i] = \{\mathbf{q}\}$ **then**
        $j \leftarrow j + 1$; update t_k
      else if $\|\mathbf{qd}_i\| \geq d_{\max}$ AND $j \neq 0$ **then**
        break
      end if
    end if
  end while
  return $t_k(\mathbf{q})$
end procedure

Algorithm 3: k-nn based thickness computation.

are no longer visible, as they have been replaced by a global red patch (larger thickness values). Notice that, in some mesh parts, t_k does not change when increasing k because of the visibility condition.

Since the thickness should be pose-invariant, very large values of k may also induce issues: the probability that half-diameter points are not visible after a pose operation increases with their distance. In addition, the computation times directly depends on the parameters k and n.

Table 1 summarizes the main differences between our algorithm and the original SDF algorithm.

Parameter	SDF	t_k
Sampling	Facet centers	Random sampling
Cone	Large	Adaptive-opening
Diameter	Outlier-robust average	Minimum length
Postprocessing	Bilateral smoothing Normalization	Robust distance function

Table 1: Main stages of the thickness computation compared to the SDF.

4. Implementation Detail

This section provides details on our technical choices for implementing the thickness estimation t_k. Our algorithm is implemented in C++ with components from the CGAL library [10]. The ray casting and intersection queries use an AABB tree data structure. The robust distance function uses an incremental neighbor search based on a kD-tree. Both processes are multi-threaded through OpenMP. On most 3D objects of our database (with a few thousands vertices), the algorithm takes around 30 seconds on a PC with two quad-core processors clocked at 2.93 GHz and 12 GB RAM.

4.1. Algorithmic Choices

4.1.1. Surface Sampling

We observed that when dealing with anisotropic meshes or low-complexity meshes, using only the facet centers to estimate the diameter produces biased or incorrect results with high dependency to the input discretization. The first step of our thickness estimation algorithm is thus a dense re-sampling of the input surface mesh $\mathcal{S}$ to generate a set of n points samples denoted by $\mathcal{P}$. A mesh-independent solution consists in generating boundary points by casting random rays inside the bounding box of the object and computing intersection points with its surface. However, for shapes with very fine levels of detail this method requires a very dense sampling to avoid overlooking parts of the object. Our default random sampling method is thus based on uniform sampling of each triangular facet, with a a number of samples proportional to the contribution of the facet to the total area of the input surface mesh. In Section 5.2.1, we detail how we set the value for n for benchmarking, as the scale of the thickness estimation is controlled by the ratio $\frac{k}{n}$. Fixing n thus leads to a unique parameter for scale selection. In addition, when the set of point queries $\mathcal{Q}$ is known before thickness estimation, we add $\mathcal{Q}$ to $\mathcal{P}$.

Note that such a choice departs from the recent down-sampling approach [8] for computing SDF values. Our goal is to improve both the robustness and accuracy of the thickness estimation, while the down-sampling strategy aims at decreasing the computation time of the SDF. Still, our initial re-sampling strategy could also be used to reduce the number of points at which the diameter will be estimated, e.g. by setting n to a lower value than the number of facets.

4.1.2. Cone sampling

In the original SDF approach, the diameter estimation δ is obtained through an outlier-robust weighted average of the lengths of rays cast inside a cone. The weights are designed to compensate for the bias in the casting of random rays: uniformly generating angles in $[0, \phi]$, where ϕ denotes the half opening angle of the cone, yields a non-uniform ray sampling of the cone.

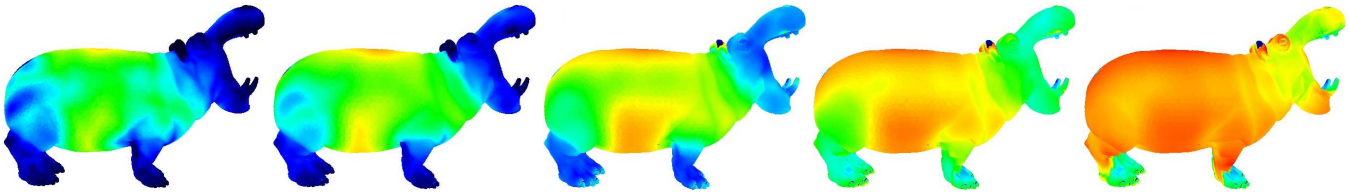
In our implementation the random rays are uniformly generated inside the cone. Let $\mathbf{p}$ denote a point on the surface, $\mathbf{n_p}$ the normal vector at $\mathbf{p}$ (unitary, pointing inward), $(\mathbf{x_p}, \mathbf{y_p})$ two orthogonal unit vectors spanning the tangent plane to the surface at $\mathbf{p}$ and $\mathbf{r}$ the direction vector for the random ray. The cone is defined by its apex $\mathbf{p}$, its direction $\mathbf{n_p}$ and its aperture ϕ. Let $\text{rnd}[a, b]$ denote a uniform random number generated within $[a, b]$. $\mathbf{r}$ is then defined as follows:

$$\mathbf{r} = \mathbf{n_p} + \lambda(\cos(\theta)\mathbf{x_p} + \sin(\theta)\mathbf{y_p}) \tag{3}$$

$$\lambda = \sqrt{\text{rnd}[0, \tan^2\phi]} \qquad \theta = \text{rnd}[0, 2\pi[$$

In addition, we always cast a ray in the direction of $\mathbf{n_p}$.

Notice that by requiring that (i) the query points in $\mathcal{Q}$ at which t_k is to be estimated are used in the sampling procedure, (ii)



(a) $k = 5$ (b) $k = 10^2$ (c) $k = 10^3$ (d) $k = 5 \times 10^3$ (e) $k = 10^4$

Figure 5: **Influence of $\frac{k}{n}$ on t_k.** Variations in the local thickness t_k for different values of k in the Hippo mesh for a fixed $n = 10^5$ value. The color map is the same for all meshes. For $k = 5$, 5(a), small thickness values are relatively important with respect to others. These are located on the legs, the toes, and the ears. Conversely, for $k = 5 \times 10^3$, 5(d), larger thickness values are more important, as small values have almost disappeared: the thickness estimated in the mid- and large scale features (e.g. legs, torso) has increased, indicated by the change from blue to green and red. Nevertheless, the visibility condition between half-diameter points and query points creates an upper-bound for t_k, especially noticeable at the ears. In these parts, t_k quickly becomes constant, and the model remains deep blue.

the ray casting procedure always include the normal, (iii) δ is set to half the minimum ray length, and (iv) the projection of the samples is along the normal, we ensure that there is always at least one point $\mathbf{d}_i$ for which Equation (2) is matched. The thickness $t_k(\mathbf{q})$ is thus defined for all points in $\mathcal{Q}$. Moreover, if $\mathcal{S}$ is a watertight surface mesh, all points in $\mathcal{D}$ are inside the mesh.

4.2. Complexity

The complexity of the algorithms mainly depends on the number of ray casting queries, which itself depends on the number of facets $|\mathcal{S}|$. For the SDF, the complexity is $O(R|\mathcal{S}|)$, R rays being cast at each facet center. For our method a worst case scenario has a complexity of $O(n(RI + |\mathcal{Q}|))$, where n denotes the number of sampling points on the surface, and I is the maximum number of iterations of the adaptive ray casting procedure. This corresponds to a configuration where: (i) the adaptive ray casting always reaches the smallest opening angles; (ii) for each query point in $\mathcal{Q}$, all the medial points in $\mathcal{M}$ have to be checked against Equation (2). Conversely, the best case scenario involves $O(2nR+|\mathcal{Q}|k)$ ray casting queries: (i) the adaptive ray casting always ends after 1 iteration; (ii) Equation (2) holds true for all query points in $\mathcal{Q}$ and their k nearest medial points. In our experiments we observe that our implementation of the original SDF approach runs between 2 and 5 times faster than t_k on most meshes. These results however greatly depend on the input mesh and the number of visibility checks performed (Section 3.2.2), and is a direct function of the parameter settings described below.

5. Experiments

In the following, we denote by δ_{SDF} the thickness computed using the original SDF method, with the following modifications: (i) we use the unbiased ray casting strategy described in Section 4.1.2, and therefore do not use any weight-based compensation mechanism; (ii) we do not perform the log-based normalization, as we benchmark the accuracy of the estimate and require the actual thickness measurements.

5.1. Setup

5.1.1. Database

The performance of δ_{SDF} and t_k are benchmarked on a database of 392 meshes. Most of them are watertight [11], and the number of facets ranges from a few hundreds to around 100k. These meshes contain both articulated and non articulated shapes, as well as mechanical parts, as partially shown by Figure 6.

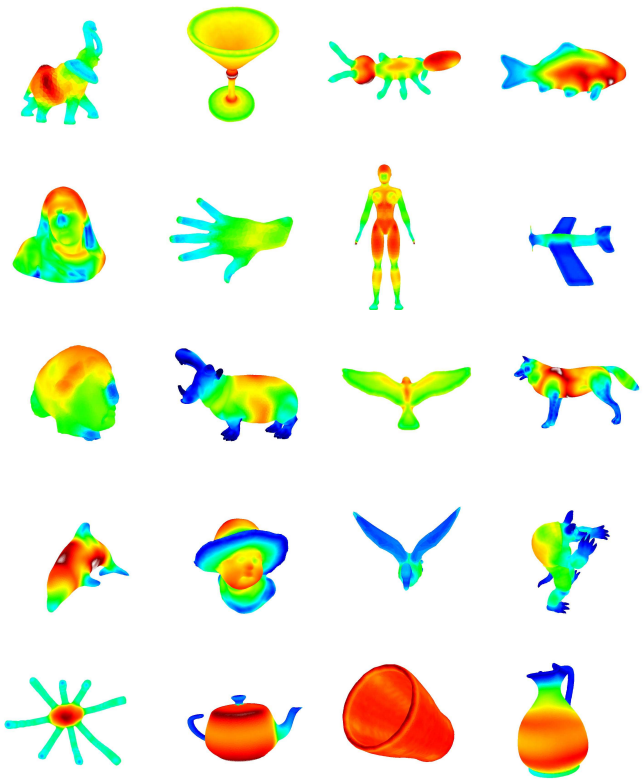


Figure 6: **Subset of meshes in our database.** Estimated thickness values are depicted with a color ramp ranging from blue to red. Note that the colormap has not been normalized between meshes.

When benchmarking against specific distortions such as, e.g., addition of noise, a smaller subset of the database is used, described in Table 2. This subset contains 4 articulated meshes

and 1 mechanical mesh, with large variance in the number of facets and feature sizes.

Mesh	# Facets	# Vertices	Properties
Elephant	5.5×10^3	2.8×10^2	Thin features
"U"	168	86	Mechanical part
Fish	15×10^3	7.5×10^2	Smooth surface
Giraffe	27×10^3	9.2×10^2	Anisotropic
Armadillo	30×10^3	15×10^3	Many small bumps

Table 2: Meshes used for benchmarking against distortions.

5.1.2. Performance Metrics

We benchmark the (i) instability, (ii) accuracy, and (iii) robustness of the estimation algorithms against distortions such as noise addition, simplification, etc. The performances are always benchmarked both locally for a facet and globally for the entire mesh.

Instability. The instability measures the intrinsic uncertainty of a thickness estimate, as both δ_{SDF} and t_k are based on a stochastic approach. In the following, $\mu^q(f)$ denotes the average thickness estimated at the center of facet f over q runs of each algorithm. For instance, denoting by $\delta^i_{\mathrm{SDF}}(f)$ the estimated δ_{SDF} at trial i, $\mu^q(f) = \frac{1}{q}\sum_{i=1}^{q} \delta^i_{\mathrm{SDF}}(f)$. In addition, $\sigma^q(f)$ denotes the standard deviation of the estimation.

1. The *(intrinsic) local instability* $I_q(f)$ is measured by the coefficient of variation of an estimate at a facet center, as follows:
$$I^q(f) = \frac{\sigma^q(f)}{\mu^q(f)}. \tag{4}$$
2. The *(intrinsic) global instability* I^q is defined as the average local instability over the surface mesh $\mathcal{S}$, as:
$$I^q = \frac{1}{|\mathcal{S}|}\sum_{f\in\mathcal{S}} I^q(f). \tag{5}$$

Note that the instability is defined using several runs of an algorithm on the same mesh. We set $q = 4$ in all experiments.

Accuracy. In general, the accuracy of a method is defined with regard to a so-called "ground-truth". Given a ground-truth defined per facet, the following metrics are defined:

1. The *local accuracy* is measured for a given mesh facet by computing the relative error between the averaged output of an algorithm and the ground-truth at the facet center. Denote by $g(f)$ the ground-truth at the center of facet f. We define the local accuracy $a^q(f)$ as:
$$a^q(f) = \frac{|\mu^q(f) - g(f)|}{g(f)}. \tag{6}$$
2. The *global accuracy* on a mesh is then given by averaging the local accuracy over all facets.
$$a^q = \frac{1}{|\mathcal{S}|}\sum_{f\in\mathcal{S}} a^q(f). \tag{7}$$

For the thickness computation, defining a ground-truth is a complex task. In the following, the mathematical thickness, which is defined at point $\mathbf{p}$ as the radius of the maximal ball associated with $\mathbf{p}$ (see Section 2), is selected. This choice requires computing the medial axis analytically, which is feasible for canonical shapes such as spheres, torii and infinite cylinders. However, for more complex shapes or noisy inputs, the notion of scale comes into play, and the mathematical thickness becomes an inappropriate ground-truth. Therefore, computing the accuracy with regard to the mathematical thickness would be irrelevant in most cases.

Robustness. The robustness of a method against distortions which preserve the connectivity is evaluated through the average relative error. Denote by f a facet of mesh $\mathcal{S}$ and f' its image in a distorted mesh $\mathcal{S}'$. f' is well-defined, since the distortion induces a one-to-one mapping between $\mathcal{S}$ and $\mathcal{S}'$.

1. The *local (per facet) error* is defined as:
$$R^q_{\mathcal{S},\mathcal{S}'}(f) = \frac{|\mu^q(f') - \mu^q(f)|}{\mu^q(f)}. \tag{8}$$
2. The *global error* of an algorithm is then measured by:
$$R^q_{\mathcal{S},\mathcal{S}'} = \frac{1}{|\mathcal{S}|}\sum_{f\in\mathcal{S}} R^q_{\mathcal{S},\mathcal{S}'}(f). \tag{9}$$

Notice that the larger R^q, the lower the robustness. In other words, R^q can be seen as a measurement of the inconsistency of an algorithm for a given distortion.

For modifications that do not preserve the mesh connectivity, i.e., when the mapping between f and f' is lost through local mesh operators, depicting the thickness using identical color maps enables a visual comparison. A quantitative comparison is made using Equation (8), where f' is chosen as the nearest facet to f in $\mathcal{S}'$. Notice that in this case $R^q_{\mathcal{S},\mathcal{S}'} \neq R^q_{\mathcal{S}',\mathcal{S}}$. Finally, another evaluation metric consists in comparing the normalized histograms of the thickness over $\mathcal{S}$ and $\mathcal{S}'$.

5.2. Comparison with the Shape Diameter Function

For comparison we define two comparable baseline parameter settings for δ_{SDF} and t_k. We then benchmark (i) the accuracy, (ii) the instability, and (iii) the robustness to pose for both algorithms. These last two criteria are already mentioned in the original SDF algorithm.

5.2.1. Parameters

Using a robust distance function instead of the bilateral smoothing of the SDF presents the immediate advantage of reducing the number of parameters for this part of the algorithm. However, setting the parameter k for computing $t_k(\mathbf{p})$ and setting the parameters of the bilateral filter are unrelated. Directly comparing the benefits and drawbacks of increasing k or the number of smoothing iterations is therefore not meaningful.

To compute δ_{SDF}, we start with a single diameter estimation for every facet center, and then apply i iterations of bilateral

smoothing with a window of size w. In a regular 4-1 subdivision mesh, the size of the surface patch involved in this computation is $m = 6iw(iw+1)$, with m the number of facets. In practice, we choose $m = 72$ by setting $i = 3$ and $w = 1$, i.e. 3 iterations of a bilateral smoothing based on a 1-ring spatial neighborhood. Since the trade-off between robustness and accuracy for the estimation of local quantities depends on m, the same value is used for t_k when comparing results between algorithms.

Sampling size. Regarding t_k, we first set a normalized target sampling size $\tilde{n} = 10^5$ in all experiments. For a given mesh $\mathcal{S}$, let A be the area of the surface, b the length of the space diagonal of the bounding box and $|\mathcal{S}|$ the number of facets in $\mathcal{S}$. To ensures that the sampling size is scale-invariant, the actual sampling size is defined as follows:

$$n = \tilde{n}\frac{A}{b^2}. \tag{10}$$

k-nn settings. Medial points being projections of uniformly sampled points, the area a of the surface boundary that is required to create k medial points is defined as follows, where d denotes the sampling density, which is set to be uniform over the mesh.

$$a = \frac{k}{d} \tag{11}$$

Equation (11) can then be rewritten using m, the average number of facets in area a as:

$$m\frac{A}{|S|} = \frac{k}{\frac{n}{A}} \implies k = \frac{nm}{|S|}. \tag{12}$$

As we use $m = 72$ and $\tilde{n} = 10^5$ for our experiments, Equation (10) and (12) show that k can be automatically set, its value ensuring a meaningful comparison between δ_{SDF} and t_k. In the following, we simply denote t_k as t.

Output. Finally, we set $\mathcal{Q}$, the query points at which t is estimated, as all the facet centers. Therefore, for both δ_{SDF} and t, all parameters are automatically set, and a single value is eventually assigned to each facet.

Table 3 summarizes the baseline parameter settings.

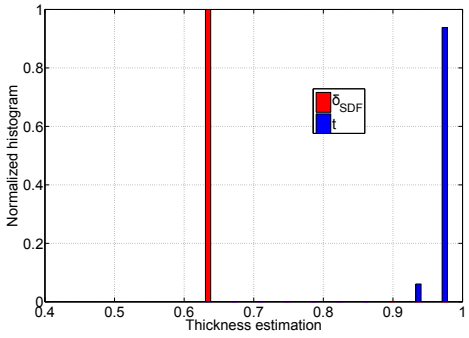
Parameter	δ_{SDF}	t
Sampling	-	$\tilde{n} = 10^5$ (Eq. (10))
R (rays)	30	5
Cone opening	$\phi = 60°$	Adaptive opening (Section 3.1) $\phi = 25°$, $\eta = 2°$ $\tau = 0.8$, $I = 10$
Postprocessing	Bilateral filter window $w = 1$ iteration $i = 3$	Robust distance $m = 72$ (Eq. (12))
Output	Single estimate per facet	

Table 3: Baseline parameter setting to establish meaningful comparisons between the results of δ_{SDF} and t.

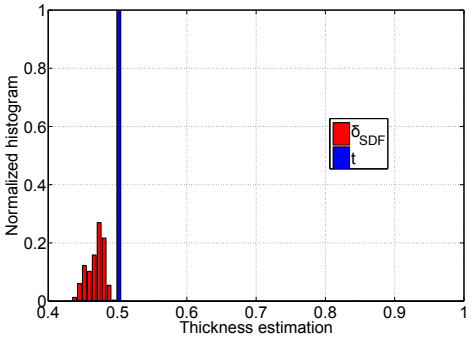
5.2.2. Accuracy

Sphere and Torus. We compare the accuracy of t and δ_{SDF} for a sphere of radius 1.0 (1,740 facets), and a torus of minor radius 0.5 and major radius 2.0 (3,200 facets). Figure 7 depicts a normalized distribution of the per-facet values. Averaging multiple ray lengths in the case of a sphere yields a substantial underestimation of the actual radius: δ_{SDF} is around 0.65 all over the sphere instead of the expected unitary value. Among all rays cast inside a cone along the normal direction, only one yields the correct value (the actual radius), while the others yield smaller values. The global instability I^4 (i.e., over 4 iterations of the algorithm) of δ_{SDF} is around 2.5%, while it is about 1.6% for t.

For the torus, averaging ray lengths as in δ_{SDF} provides a better estimation of the half-section than for the radius of the sphere (theoretical value at 0.5), but leads to a slightly scattered distribution of δ_{SDF} around 0.48. The global instability I^4 is in this case around 3.0% against 0.03% for t. Finally, note that the inaccuracy of the estimation process is also due to the sampling, since both meshes are only approximations of the unit sphere and torus.



(a) Unitary Sphere



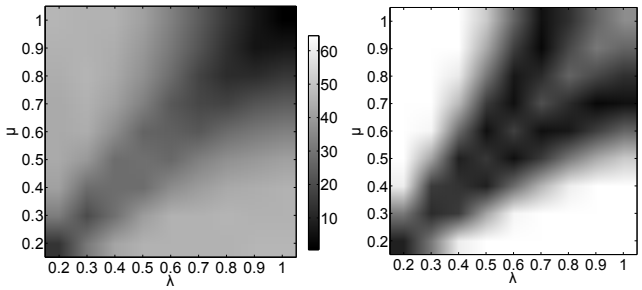
(b) Torus (major radius 2.0, minor radius 0.5)

Figure 7: **Accuracy for a sphere and a torus.** Distribution of δ_{SDF} and t using a mesh of the unit sphere 7(a) and a torus 7(b)). δ_{SDF} underestimates the radius of the sphere and exhibits a scattered distribution in the case of the torus.

Ellipsoids. Both algorithms are evaluated on 81 ellipsoids (20k facets each), parameterized by their eccentricity. Their centers are at the origin, with two semi-axis (λ, μ) ranging from 2×10^{-1} to 1 (step: 10^{-1}) and the third semi-axis constantly set to $c = 1$. For each ellipsoid, the thickness t and δ_{SDF} are exper-

imentally estimated as well as their local accuracy with regard to the mathematical thickness computed in closed form.

Figure 8 depicts in gray-scale the global error between each thickness estimation algorithm and the mathematical thickness. All values are normalized in order to depict the results with the same scale. On average, t is closer to the mathematical thickness, which is visually verified as the diagram 8(a) is darker than the diagram 8(b). However, when only looking at the diagonal from $(\lambda, \mu) = (1; 1)$ to $(\lambda, \mu) = (0.2; 0.2)$, δ_{SDF} presents better results. These cases correspond to cigar-like ellipsoids, with one axis set to 1.0, and a circular cut ($\lambda = \mu$). In the upper-right part of the diagram, when the ellipsoids are very close to the unit sphere, t is more accurate than δ_{SDF}, which is consistent with the results shown by Figure 7. Finally, when transforming the unit sphere to a plate-like ellipsoid (λ or μ set to 1, i.e. only considering the first row or the first column of the diagrams), the accuracy of both algorithms drops abruptly, albeit less for t.



(a) t (b) δ_{SDF}

Figure 8: **Accuracy benchmark on ellipsoids.** Global accuracy (average relative error) between the estimated thickness, i.e. δ_{SDF} and t, and the mathematical thickness for various ellipsoids with semi-axis $(\lambda, \mu, 1.0)$. The upper-right part of the diagram corresponds to sphere-like ellipsoids. On the lower-left, the ellipsoids have a cigar-like shape. On the upper-left and lower-right, the ellipsoids are plate-like shaped. Since both λ and μ have the same range, both diagrams are symmetrical.

5.2.3. Stability

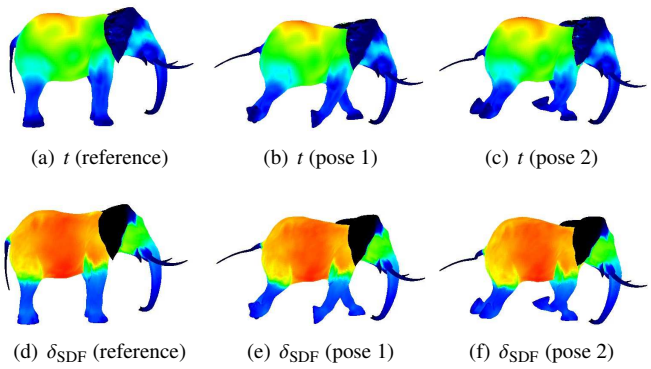
We measure the global instability I^4 of both thickness estimation methods on the entire mesh database. For δ_{SDF}, the instability is on average 3.4%. It is around 0.58% for t. Although these results confirm the stability of our thickness estimate, our computation time is about 3 times slower (around 30 seconds for every mesh instead of 10 seconds for δ_{SDF}). Figure 9 summarizes the results of the instability benchmark.

Figure 9(a) shows the global instability for every mesh in our database (392 points). For δ_{SDF}, the values are more scattered around the average 3.4% than for t, which only exhibits 3 outliers (above 2% global instability). These outliers correspond to mechanical parts: the point close to the diagonal represents the "U" shape, while the points in the upper-part of the diagram stand for bowls or plates. All the articulated meshes are located around the point $(0.5; 3)$.

Figure 9(b) shows the local instability of both methods for the Armadillo mesh. The average instability is around 0.5% for t and 2.8% for δ_{SDF}. Both methods exhibit local outliers, but their range are slightly different. For t, some facets have an instability below 0.1%, and a single facet has an instability above 10%. For δ_{SDF}, all values are above 0.1%, and many above 10%. In other words, the local instability of t: (i) has a lower upper-bound than δ_{SDF}, with very few unstable estimates; (ii) is lower than the local instability of δ_{SDF} for most facets (the point cloud is mostly above the diagonal); (iii) has a few highly stable estimates.

5.2.4. Robustness to Pose

A series of poses of the Elephant mesh are used to benchmark the robustness of δ_{SDF} and t. This mesh has 85k facets and contains some self-intersections at the ears. Figure 10 shows the local thickness estimates on the reference mesh and two differently posed versions. This first experiment assesses the visual consistency of both algorithms over different poses.



(a) t (reference) (b) t (pose 1) (c) t (pose 2)

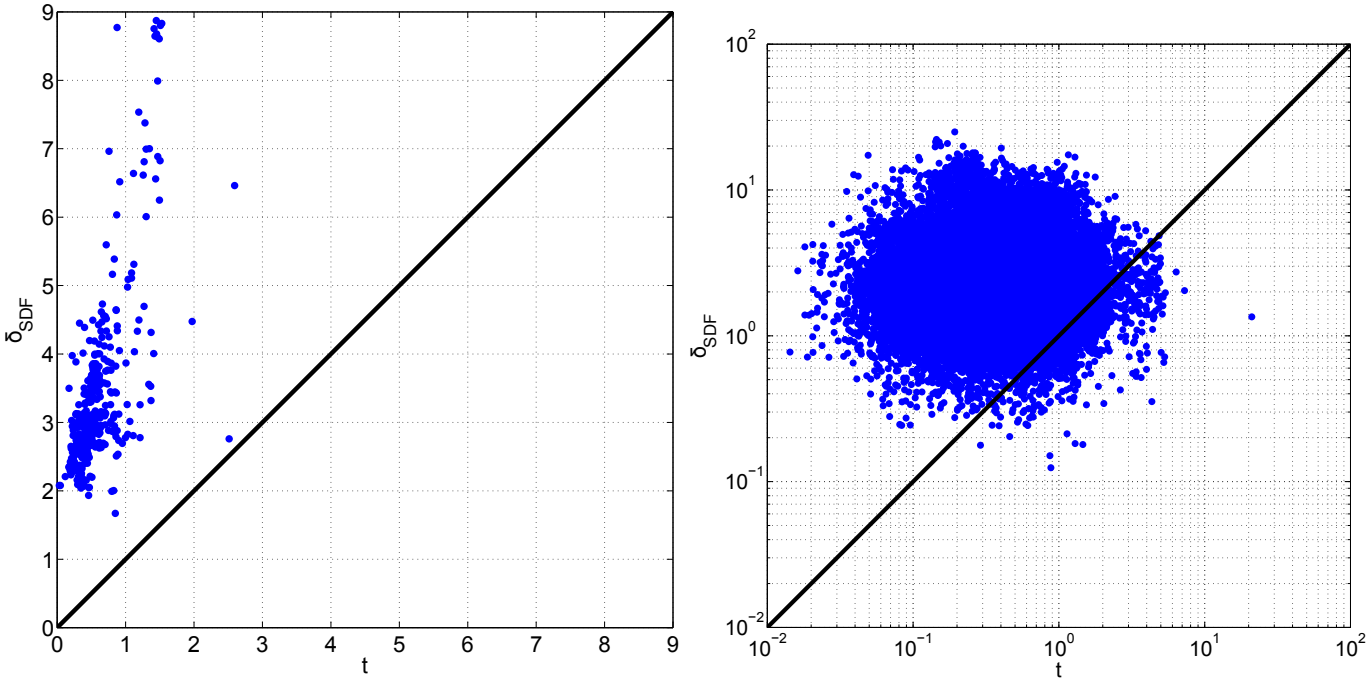
(d) δ_{SDF} (reference) (e) δ_{SDF} (pose 1) (f) δ_{SDF} (pose 2)

Figure 10: **Estimated thickness for 3 poses of an elephant.** The same range of colors is used for all poses. Both δ_{SDF} and t yield visually consistent results. Note that the estimated thickness values greatly differ between the two algorithms.

A quantitative analysis of the robustness is depicted on Figure 11. Figure 11(a) first illustrates the global robustness of δ_{SDF} and t using the global error R^4 and all the various poses of the Elephant mesh. Note that the reference values in Equation (9) correspond to the ones computed on the reference pose (depicted on Figure 10(a) and Figure 10(d)). The error is upper-bounded by 16% for δ_{SDF} and by 11% for t. For almost all poses, t is more consistent than δ_{SDF} (points above $y = x$). Figure 10(b) and Figure 10(e) show the local thickness for the single pose where δ_{SDF} is more consistent than t.

Figure 11(b) illustrates the local robustness of both methods for one pose of the mesh. This pose corresponds to the point with coordinates $(10.73; 12.54)$ on the global error diagram (Figure 11(a)), i.e. one of the worst result for both methods. The actual estimated thickness can be seen on Figure 10(c) and Figure 10(f).

The local error for δ_{SDF} and t exhibits a large number of outlier values, i.e. facets for which the estimated thickness is highly modified, or conversely almost exactly identical between poses. The few facets with a large error (above 10^3%) could correspond to the joints of the model, at which the local thickness greatly varies between poses. However, these facets are not the same for δ_{SDF} and for t. Moreover, none of them are located at the joints, but only on the ears of the elephant. Furthermore,



(a) Global instability (%) for every mesh in our database.

(b) Local instability (%) for the Armadillo mesh.

Figure 9: **Global instability and local instability of δ_{SDF} vs. t.** Points above the diagonal line $y = x$ represent facet centers for which δ_{SDF} is more unstable than t. 9(a): for a small number of meshes, the global instability of δ_{SDF} gets very large (around 7%), while it always stays below 3% for t. 9(b) the logarithmic scale denotes a large scattering, with the local instability ranging from 10^{-2}% to 10^{2}%.

the largest local instability values (computed with Equation (6)) are located in the same regions. This indicates a correlation between large local errors, large instability of the estimates, and the self-intersections of the mesh (which are located near the ears).

5.3. Benchmarking vs. Distortions

This section presents the results on the robustness of t w.r.t. a variety of synthetic distortions. Without a direct comparison with δ_{SDF}, the parameter settings established in Section 5.2.1 is no longer needed. In particular, setting k as a function of the number of facets $|\mathcal{S}|$ is ill-suited for benchmarking against e.g. simplification. Similarly, making parameters dependent on the surface decreases the robustness against e. g. noise, as the parameters are altered over different noise magnitudes.

In the following, we simply change k so as to use a constant $\frac{k}{n}$ value. In other words, the scale at which the estimations are performed stays constant.

5.3.1. Affine transformations

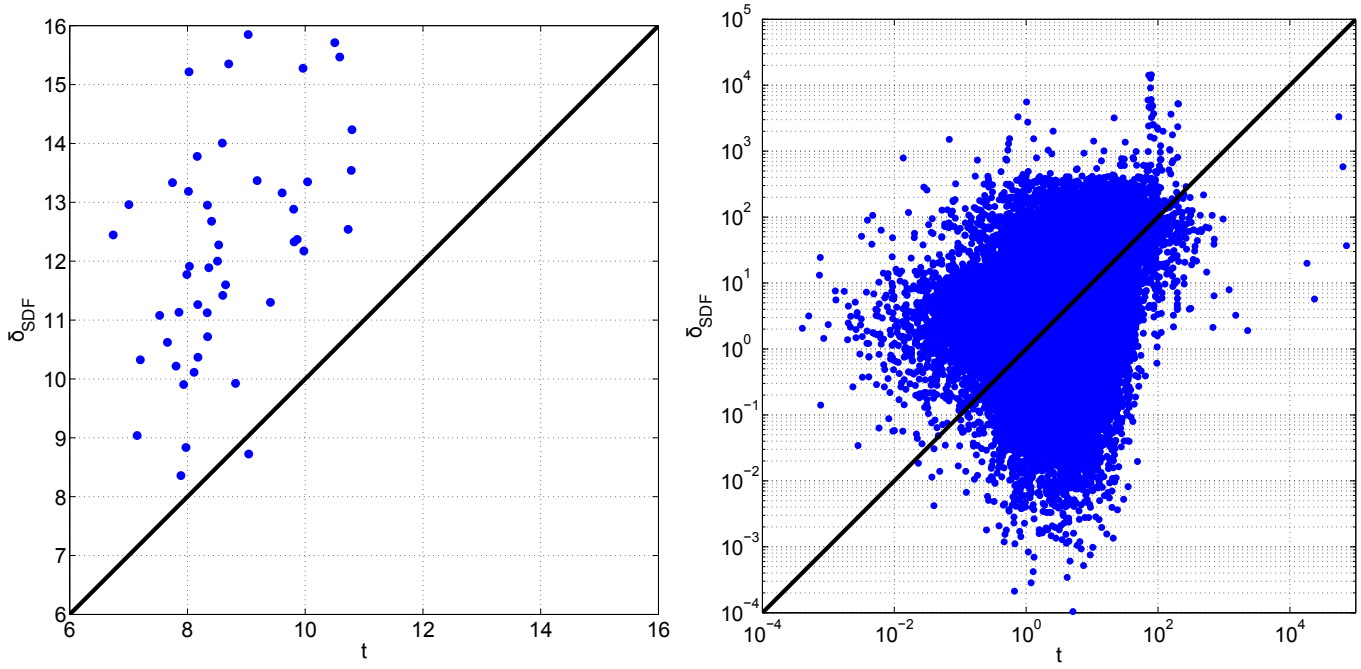
The most basic affine transformations in $\mathbb{R}^3$ consists in rigid transforms (rotations, translations), for which all the algorithms are in theory invariant. For a scaling operation with ratio α, we simply compute an estimate $\tilde{\alpha} = \frac{1}{|\mathcal{S}|}\sum_{f\in\mathcal{S}}\frac{\mu^m(f)}{\mu^m(f')}$ and compare it to α, using all the 392 meshes in our database. With α ranging in $[10^{-3}; 10^{3}]$, the relative error between the estimation $\tilde{\alpha}$ and the ground-truth is always below 0.01%.

5.3.2. Uniform Geometric Noise

The local thickness is estimated for the 5 meshes in Table 2 after modifying the vertex coordinates by adding a uniform noise vector along the normal. Denote by $\mathbf{n}$ the local unit normal. The noise vectors are locally generated uniformly within $[-\frac{s}{2}\mathbf{n}; \frac{s}{2}\mathbf{n}]$, with s the noise magnitude, corresponding to a ratio of the longest diagonal length of the bounding box. Figure 12 summarizes the results of this benchmark.

Figure 12(a) shows the global error vs. noise levels. For articulated meshes and $s \leq 0.1\%$, t yields on average consistent results. Above $s = 0.5\%$, the thickness estimation exhibits low robustness, as the global error quickly increases. Conversely, for the mechanical part, t shows a larger initial global error, but a slower decrease in performance with larger values of s. For $s \leq 0.1\%$ (around 2%), this is due to the high (intrinsic) global instability of the algorithm as shown in Section 5.2.3. For all the articulated meshes, this global instability stays around 0.5%, for all levels of noise.

Figure 12(b) presents the local error for the Giraffe mesh. It shows that increasing s yields a global decrease in performance, as the local error increases for 90% of the facets. In particular, the large increase in the global error at $s = 0.5\%$ comes through a drop in robustness for all facets, and not for some specific parts of the mesh. Finally, Figure 12(c) illustrates the location of the local error at $s = 2\%$ on the distorted mesh. Small error values are located on large features (torso), while small features, such as ears, exhibit the largest errors. This is explained by the fact that the additive noise is generated without taking into account the feature size: small features are relatively more



(a) Global error (%) over a series of pose of an elephant mesh.

(b) Local error (%) for a single pose of an elephant mesh.

Figure 11: **Robustness of t and δ_{SDF} w.r.t. pose.** 11(a): global error in the thickness estimation on a series of poses of the Elephant mesh. Every point represents a different pose, with its position w.r.t. the diagonal line indicating whether t or δ_{SDF} shows greater consistency. The local (per-facet) error of the pose represented by the point (10.73; 12.54) is shown on Figure 11(b). Both algorithms present a similar behavior, with an average error around 10%, and large outliers: some values are above 10^3%.

distorted than larger features. This phenomenon is magnified by the size of the space diagonal of the bounding box w.r.t. most of the mesh features.

5.3.3. Smoothing

The robustness of the thickness estimate is benchmarked against a common mesh smoothing method [12] for the meshes in Table 2 (Figure 13). We monitor the global error when increasing the number of smoothing iterations (Figure 13(a)), and provide a close-up on the distribution of local error for the Armadillo (Figure 13(b)), as well as its location (Figure 13(c)).

For 3 out of 4 articulated models, t shows a large consistency (Elephant, Armadillo and Giraffe curves in Figure 13(a)) over smoothing. For the U mesh, t conversely yields much lower consistency, with a global error above 10% after a single smoothing iteration. Finally, the results for the Fish mesh are significantly better than all the others, with a global error below 0.2% even after 20 smoothing iterations.

These results are closely correlated with the distortion introduced by the smoothing iteration and estimated through the Root Mean Square (from the distorted mesh to the original mesh) metric (RMS) [13]. For the U mesh, the RMS after 1 smoothing iteration is estimated around 4×10^{-2}, while it is only in the order of 10^{-4} for the articulated meshes. This is caused by the sharp features of the mechanical parts, which are heavily distorted by the smoothing process. Similarly, the RMS for the Fish mesh stays very low (4×10^{-5} after 1 iteration and 4×10^{-4} after 20 iterations), as the original mesh does not present any sharp features.

Finally, the close-up on the distributions of the local error and the actual mesh of the Armadillo (Figure 13(b) and Figure 13(c)) shows that t is not only robust against smoothing operations at a global level, but also at a local level: the range of the local error for 90% of the facets stays approximately within two order of magnitude, i.e. between 0.1% and 10%. The largest local error values are also correlated with the small bumps on the mesh and the small extremities, e.g. the fingers, which are heavily modified by the smoothing process.

5.3.4. Triangle soup

A triangle soup distortion is obtained by (i) disconnecting all facets of the mesh, (ii) creating holes by shrinking the triangles using a constant ratio r. Figure 14 reports the consistency results of t against this type of distortions. Figure 14(a) shows the global error for the 5 benchmarked meshes. Notice that although the connectivity has changed, an obvious 1 to 1 mapping exists between facets in the original and in the modified mesh. Computing the global error R^4 with Equation (9) is therefore still straightforward. The curve corresponding to the U mesh clearly presents inconsistencies, as the error decreases with the magnitude of the distortion. For all articulated meshes, t shows consistency until $r = 30\%$. An example for the Armadillo mesh and $r = 40\%$ is given in Figure 14(b).

These figures show that t relies loosely on the mesh connectivity, since all facets have been disconnected. This provides robustness in the presence of small holes and cracks.

5.3.5. Simplification

A practical thickness estimation has to provide consistent results for different levels of detail of the same mesh (Figure 15). On Figure 15(a), t is computed on the benchmarked meshes with levels of simplification ranging from 90% to 0.01% (indicating the percent of remaining edges w.r.t. the original mesh). For the U (resp. the elephant) mesh, with 252 (resp. 8,337) edges, this simplification process quickly reaches too large a level of distortion for the model to be recognizable. The estimation of the global error becomes then meaningless, and the curves exhibit incoherences (decreased error with increased simplification). Moreover, computing a mapping between facets in the distorted and the original mesh based on their distance also create issues with the global error computation. For the Armadillo, the Fish and the Giraffe, t shows robustness until a 5% simplification ratio.

Figure 15(b) and 15(c) display the actual thickness computed on the original Fish mesh and a simplified version (5×10^{-2}% remaining edges), showing the consistency of t at a local level. Figure 15(d) presents the local error between these two versions: the errors are mostly located on the small features which are highly altered by the simplification process.

5.3.6. Remeshing

Finally, the robustness of t is benchmarked against a complete re-meshing process. We first estimate the local thickness on a watertight triangular mesh of a face with a hat. The mesh is then re-tessellated with a regular quadrangle mesh [14], then transformed back into a triangular mesh by splitting facets. t is then estimated once more, and the local error is depicted on Figure 16. The global error is about 0.23%, with a maximum around 2%. This shows the consistency of t w.r.t. remeshing operations.

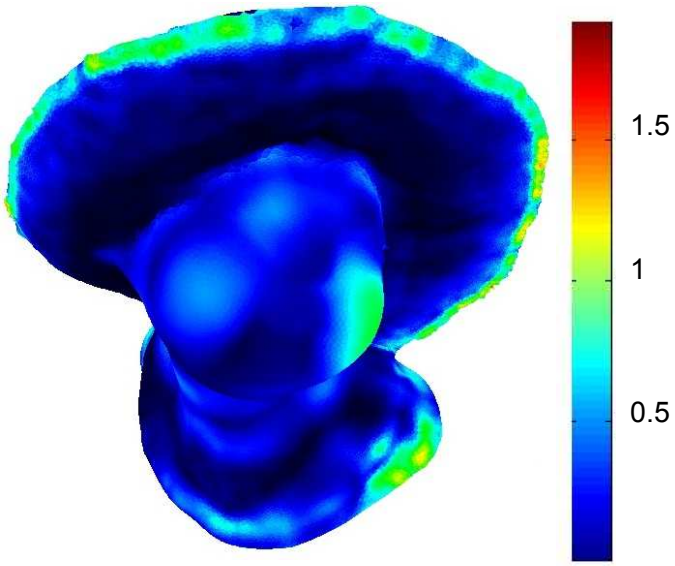


Figure 16: **Robustness to remeshing operations.** Local error (%) between t originally estimated on a mesh (face with hat) and a remeshed version. The local error stays below 2% all over the mesh.

5.4. Segmentation

One of the main potential application of a local thickness estimation is to enable a robust and efficient mesh partitioning. This was originally achieved by using a soft clustering of the SDF values, followed by a graph-cut computation [1]. Note that the segmentation method itself also improves the robustness of the processing pipeline, as regrouping facets into patches has an averaging effect.

The benefits of using t for segmentation purposes are first illustrated by Figure 17. We applied the segmentation process on an Elephant mesh (89k facets) with 3% noise (Figure 17(a)). As a result, the segmentation based on δ_{SDF} (Figure 17(b)) is highly modified and creates 44 small segments. The segmentation relying on t (Figure 17(c)) provides a more intuitive partitioning of the mesh.

Regarding the robustness of mesh segmentation, Figure 18 shows the median relative error in the number of segments for the t-based and the δ_{SDF}-based segmentation when applying different types of distortions to the meshes in our database: (i) an increasing number of smoothing iterations (Figure 18(a)); (ii) an increasing number of edge simplifications (Figure 18(b)). These results show that even after a large number of smoothing iterations, e.g. 50, the number of segments created by the t-based segmentation algorithm is still very close to the original ones, around than 2% variation. In a similar manner, the segmentation based on t shows a large consistency, even after heavy simplification on the input meshes.

6. Conclusion

We have introduced a robust thickness estimation method based on a shape diameter estimation. We modified the original SDF algorithm [1] by (i) introducing an adaptive scheme when sampling the local volume of a mesh; (ii) replacing the bilateral smoothing by a robust distance function to a cloud of half-diameter points, thus creating a scale-dependent robust thickness estimate over the mesh surface.

The rationale for changing the initial methodology was to increase (i) the accuracy with respect to a ground-truth defined through an exact medial axis extraction (for canonical shapes such as spheres or ellipsoids); (ii) the intrinsic stability of the stochastic process; and (iii) the robustness of the estimate against alterations of an original mesh. Our experiments confirm the accuracy of our thickness estimation, illustrated on some canonical examples for which an analytical ground-truth (the mathematical thickness) is both well-defined and meaningful. They also indicate a tangible improvement in the stability of the estimation process: with comparable parameter settings, the SDF-based thickness estimation shows on average an instability one order of magnitude larger than our method. We have also benchmarked our method against a wide range of modifications such as pose, noise addition, triangle soup, simplification and remeshing. The results show that for some distortions, our method exhibits an increased robustness with respect to the original strategy.

Our method has room for improvement: its application is in general limited to articulated shapes such as humanoids or animals. On mechanical parts, we have shown in Figure 3 that our adaptive strategy could be improved. Some experiments also indicate that the robustness of our method could be improved by finding a more appropriate outlier-removal strategy. Finally, improving the stability and the robustness of the thickness estimation makes it suitable to other applications such as robust segmentation or robust watermarking.

Acknowledgements

We wish to thank the authors of the original SDF algorithm [1] for making their code publicly available and providing us with their database of 3D objects. We also thank Ilker O. Yaz for his implementation of the SDF and segmentation algorithm available as a CGAL software component. We further thank David Bommes for help with remeshing. This work was funded by Technicolor. Pierre Alliez is funded by the European Research Council (ERC Starting Grant "Robust Geometry Processing", Grant agreement 257474).

[1] L. Shapira, A. Shamir, D. Cohen-Or, Consistent mesh partitioning and skeletonisation using the shape diameter function, Visual Computer 24 (2008) 249–259.

[2] M. Garland, Multiresolution modeling: Survey & future opportunities, in: EUROGRAPHICS '99 – State of the Art Reports, pp. 111–131.

[3] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Lévy, Polygon Mesh Processing, AK Peters, 2010.

[4] H. Blum, A Transformation for Extracting New Descriptors of Shape, in: W. Wathen-Dunn (Ed.), Models for the Perception of Speech and Visual Form, MIT Press, Cambridge, 1967, pp. 362–380.

[5] A. Tagliasacchi, Skeletal representations and applications, CoRR abs/1301.6809 (2013).

[6] D. Attali, J.-D. Boissonnat, H. Edelsbrunner, Stability and computation of medial axes, a state-of-the-art report, in: Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration, Springer, 2009, pp. 109–125.

[7] J. Giesen, B. Miklos, M. Pauly, C. Wormser, The scale axis transform, in: Proceedings of the 25th annual Symposium on Computational Geometry, ACM, 2009, pp. 106–115.

[8] M. Kovacic, F. Guggeri, S. Marras, R. Scateni, Fast approximation of the shape diameter function, in: Proceedings Workshop on Computer Graphics, Computer Vision and Mathematics (GraVisMa) 2010, volume 5.

[9] F. Chazal, D. Cohen-Steiner, Q. Mérigot, Geometric Inference for Measures based on Distance Functions, Research report RR-6930, INRIA, 2010.

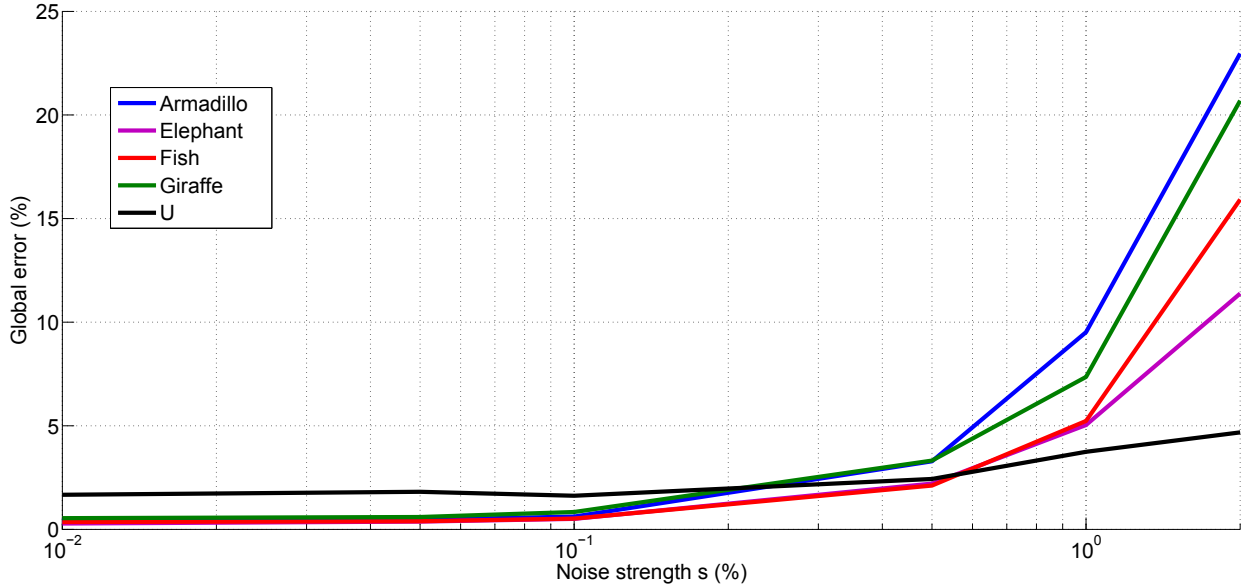
[10] CGAL, Computational Geometry Algorithms Library, 2012. http://www.cgal.org.

[11] Watertight models, 2007. http://watertight.ge.imati.cnr.it/.

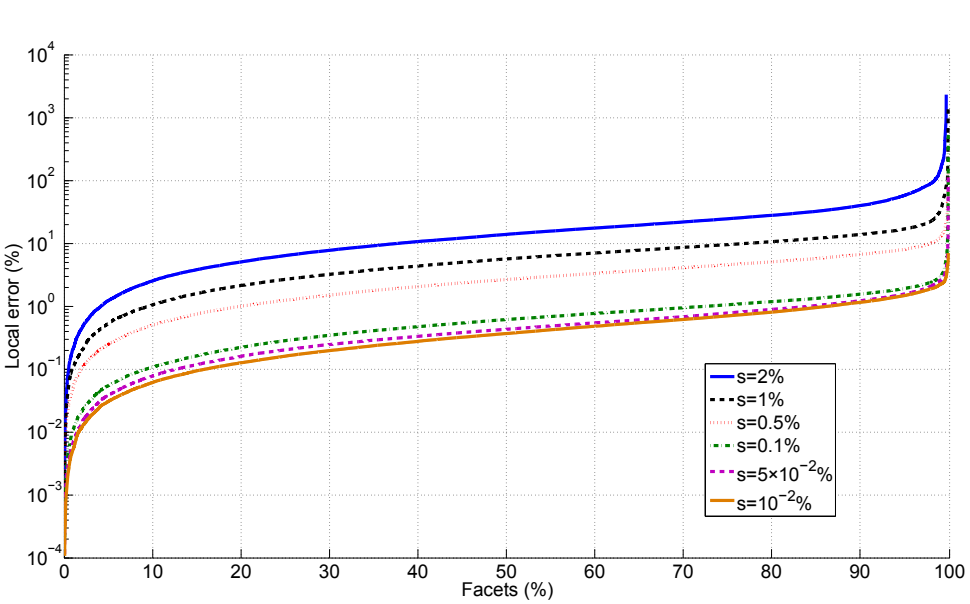
[12] G. Taubin, A signal processing approach to fair surface design, in: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH '95, pp. 351–358.

[13] P. Cignoni, C. Rocchini, R. Scopigno, Metro: measuring error on simplified surfaces, Technical Report 2, 1998.

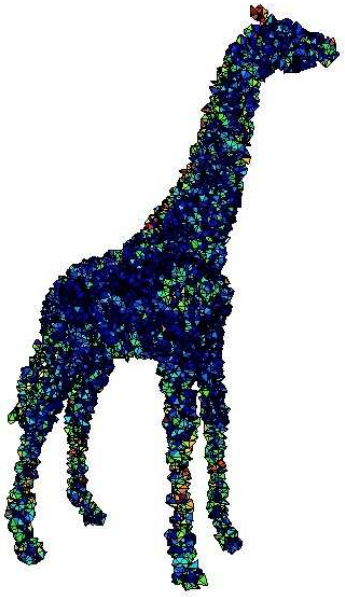
[14] L. Velho, D. Zorin, 4-8 subdivision., Computer-Aided Geometric Design 18 (2001) 397–427.



(a) Global error for increasing noise magnitudes.

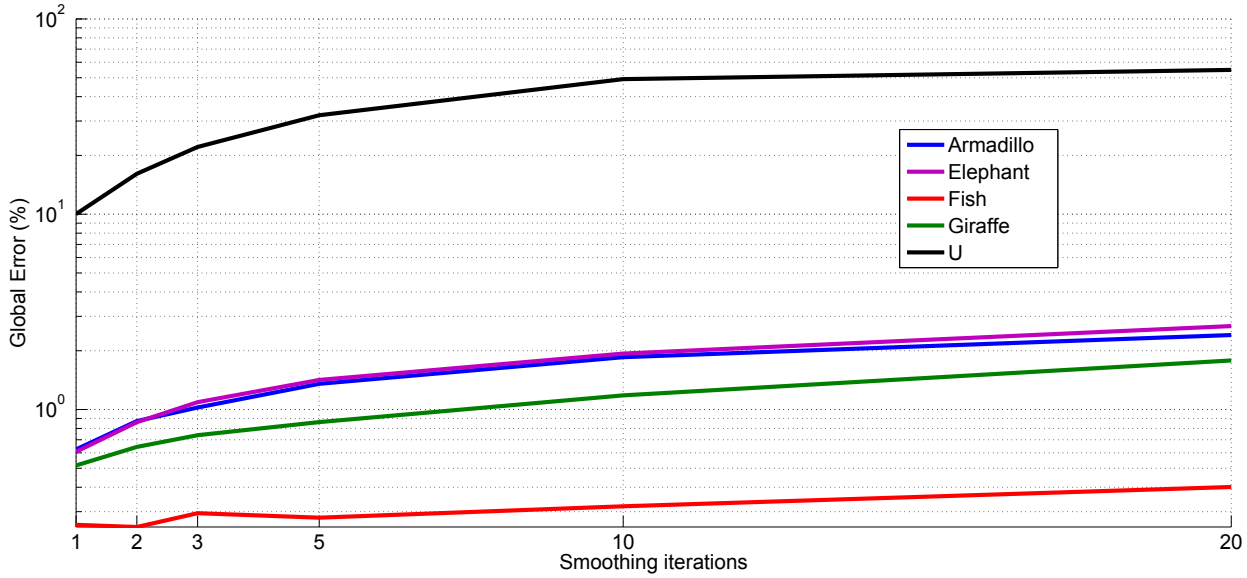


(b) Distribution of the local error for the giraffe mesh and different noise magnitudes.

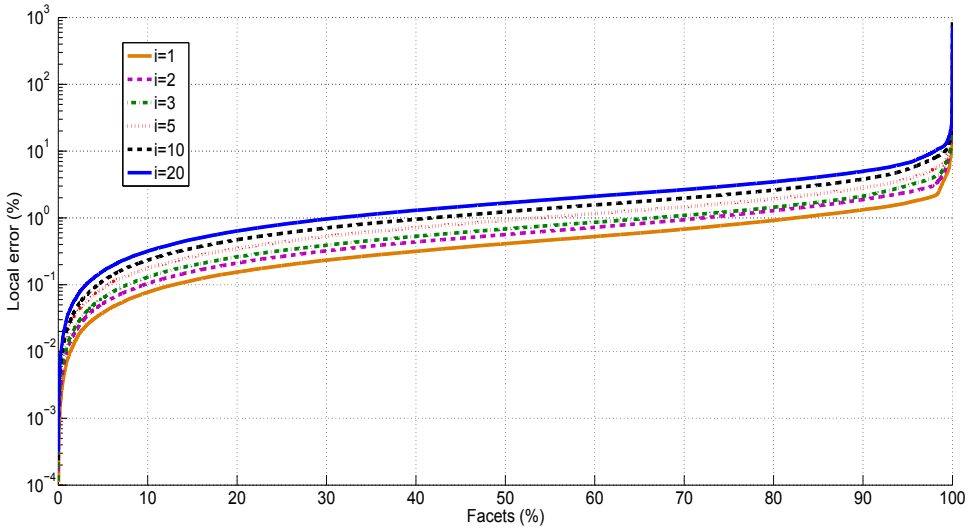


(c) Local error on the giraffe mesh with 2% noise.

Figure 12: **Robustness of t w.r.t. uniform geometric noise.** 12(a): global error between the thickness estimated on 5 series of reference meshes and their versions with additive noise is displayed. On all articulated models and for levels of noise $s \leq 0.1\%$ (s is defined w.r.t. the space diagonal of the bounding box), t yields a constant global error around 0.5%, indicating a large robustness. For $s > 0.1\%$, the global error increases rapidly, and t becomes less consistent. The U mesh (mechanical part) shows very different results! the error is systematically larger for small levels of noise (around 2%), but the drop in robustness for larger noise magnitudes is slower. 12(b): distribution of the local error for one of the articulated mesh (Giraffe) and different values of s. Note that the per-facet values are sorted along the horizontal axis for each curve. All distributions are very similar: (i) for 80% of the facets (part of the curves between $x = 10\%$ and $x = 90\%$), the error stays within the same order of magnitude, e.g. between 1% and 10% for the largest noise magnitude; (ii) 10% of the facets exhibit outlier values (parts of the curves below $x = 5\%$ and above $x = 95\%$), with very low or very large errors. 12(c): local error for $s = 2\%$ depicted on the mesh. The upper-part of the local error distribution (red facets) corresponds to the small features, such as the ears. Using a scale-independent additive noise on this models creates very large distortions, as the giraffe exhibits a wide range of feature sizes (small for the tail, legs and ears, and large for torso), and the space diagonal of the bounding box is very large.



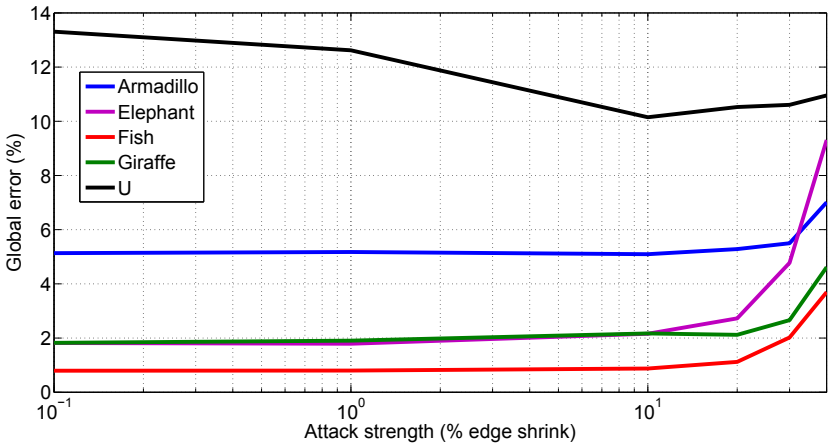
(a) Global error against an increasing number of smoothing iterations



(b) Distribution of the local error for the Armadillo and different number of smoothing iterations i

(c) Local error on the Armadillo after 20 smoothing iterations

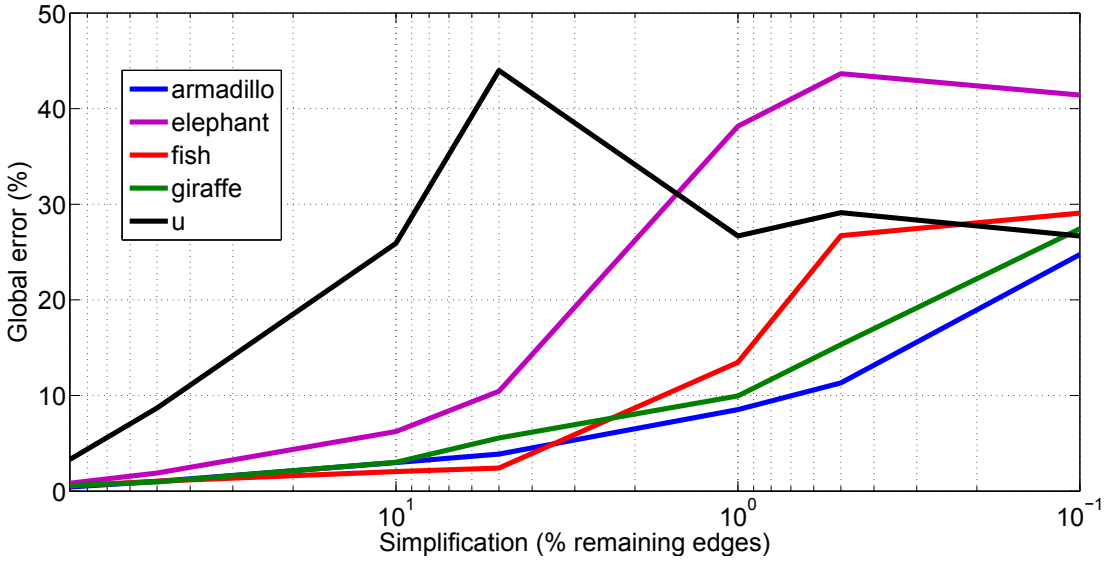
Figure 13: **Robustness of t w.r.t. to smoothing.** We apply i iterations of the Taubin filter with parameters $(\lambda, \mu) = (0.5, -0.53)$ on 5 meshes. 13(a): global error in t for increasing values of i. For the Elephant, Giraffe and Armadillo mesh, the global error has the same evolution, ranging from 0.5% and reaching around 3% after 20 smoothing iterations. The Fish mesh presents a much smaller global error than all the others. Finally, the global error is much larger for the only mechanical part in the benchmark database. 13(b): distribution of the local error for the Armadillo. The ratio of outliers is extremely low, but the part of the curves below $x = 2\%$ (resp. above $x = 98\%$), reaches significantly larger (resp. lower) values than the average, as highlighted by the logarithmic scale. On Figure 13(c), these values are displayed in the case $i = 20$: large errors (pale blue, green and red) in the thickness estimation mainly lie in the extremities, e.g. toes and fingers, which are more altered by the smoothing process. Note the pattern of average-valued local error (medium blue) all over the mesh. It corresponds to a pattern of small bumps on the original Armadillo surface.



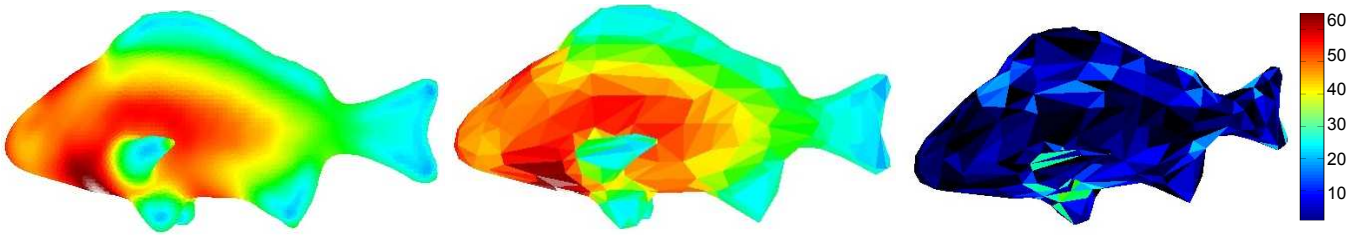
(a) Global error against triangle soup distortion.

(b) Local error for the Armadillo mesh with facets shrink 40%.

Figure 14: **Robustness of t w.r.t. to triangle soup distortion.** We benchmark t against triangle soup-like distortions for 5 meshes. 14(a): facets are disconnected, then holes are created by shrinking edges with increasing ratios. Even with 30% shrink, the thickness estimate t only shows a 5% global error with regard to the original mesh for all articulated models. The curve corresponding to the mechanical part (U mesh) is however inconsistent, though it indicates a larger sensitivity towards cracks and holes. 14(b): estimated thickness on the original Armadillo (left), and on a triangle soup version (right) with $r = 40\%$. No perceivable change appears on the mesh.



(a) Global error depending on simplification level

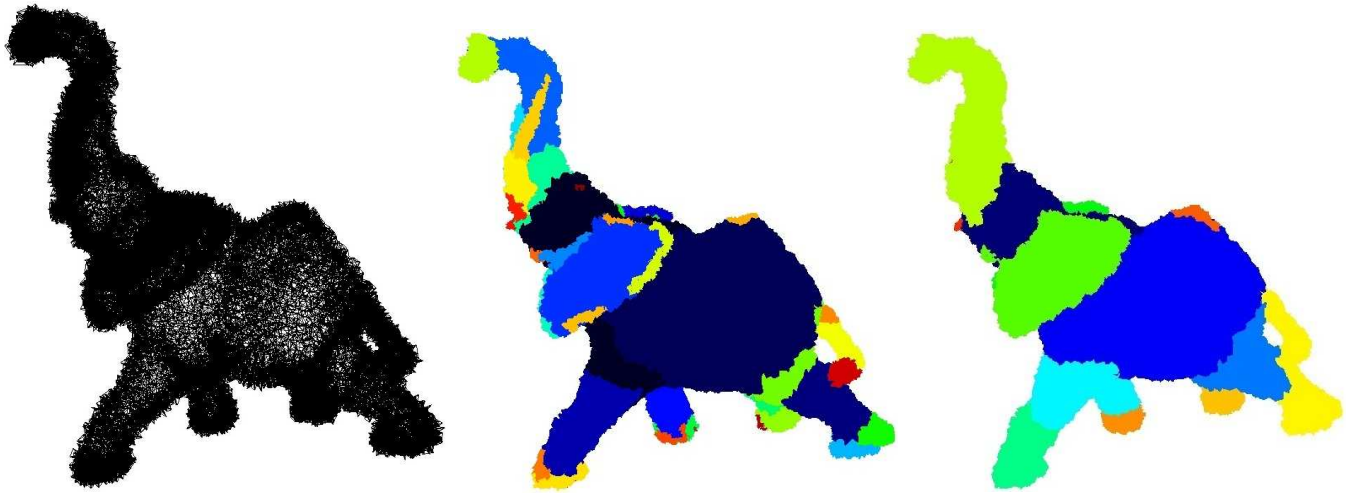


(b) t (original fish)

(c) t (simplified fish, 5% remaining edges)

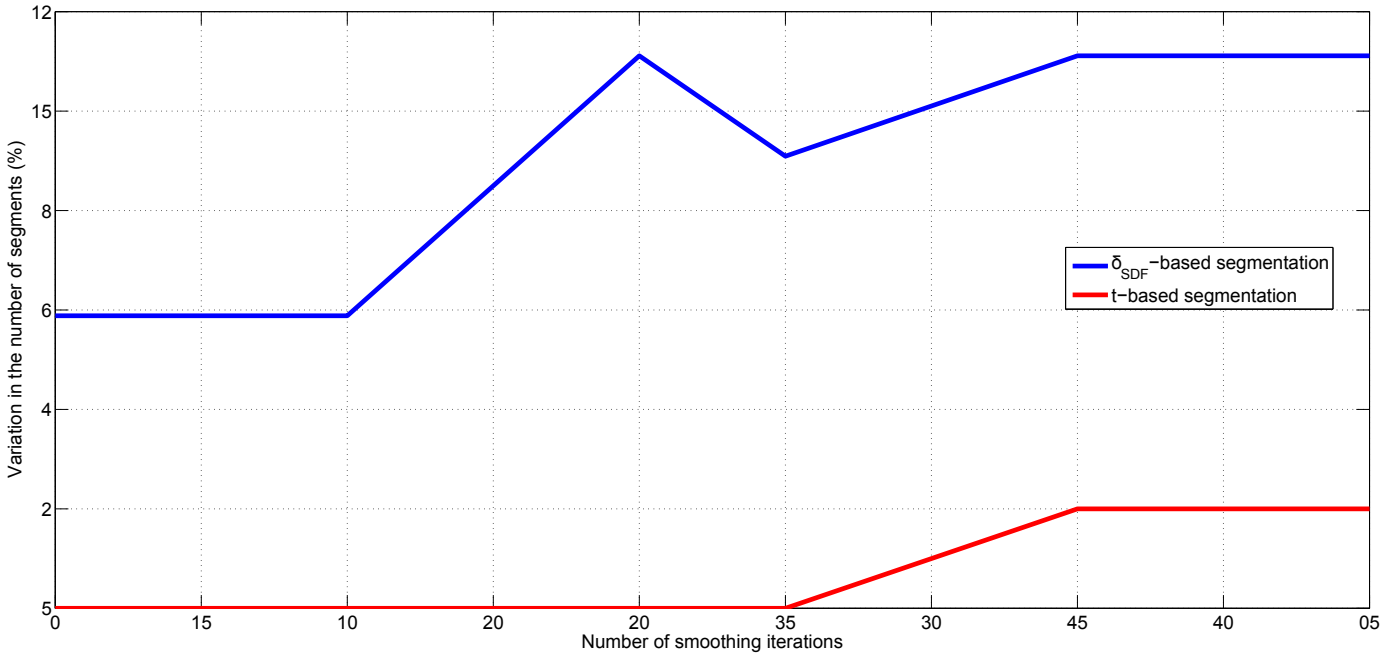
(d) Local error with 5% remaining edges

Figure 15: **Robustness of t w.r.t. to simplification.** 15(a): global error for 5 series of meshes with increasing levels of simplification. For 3 models, t shows consistency until reaching a 5% simplification ratio. For the 2 other meshes, the simplification quickly reaches a point where the estimation of the error becomes meaningless, as the models are visually unrecognizable. 15(b) and 15(c): local consistency of t for the Fish model. 15(d): quantitative measure of this consistency, as the local error is upper-bounded by 30%, mostly on the small features, which are heavily modified by the simplification process. On the other parts of the mesh, the local error remains below 10%.

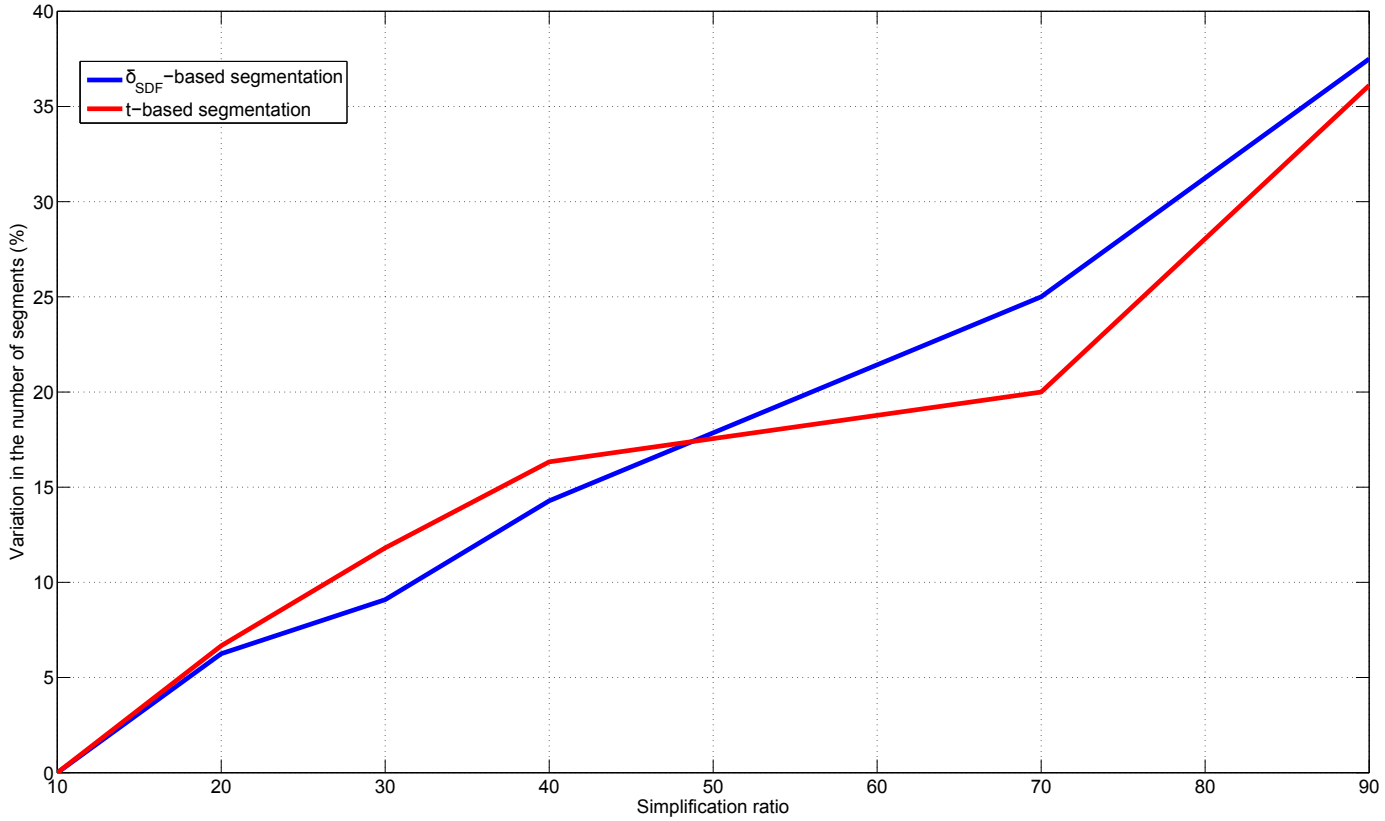


(a) Elephant mesh with uniform geometric noise (b) δ_{SDF}-based segmentation (c) t-based segmentation

Figure 17: **Segmentation using a mesh with uniform geometric noise** - Segmentation using soft clustering and k-way graph-cut [1] for a distorted Elephant mesh (Figure 17(a)) with 89K facets. We applied a random noise with 3% magnitude, as in Section 5.3.2. Figure 17(b) shows the results of the segmentation based on δ_{SDF} (44 distinct partitions), while Figure 17(c) shows the segmentation based on the thickness estimation t (8 distinct partitions). The latter yields more natural results. Notice that for t, the two tusks of the elephant are in the same segment as the trunk. In the original mesh, these parts are indeed merged together. The k-way graph cut partitioning is then inefficient to separate these features, as it cannot distinguish them using a topological criterion.



(a) Smoothing



(b) Simplification

Figure 18: **Variations in the number of segments** - Figure 18(a) monitors the average variations in the number of segments (output by the segmentation algorithm) over multiple large meshes in our database, when applying an increasing number of smoothing iterations. Figure 18(b) shows the average variation for consecutive simplifications of the input meshes (ratio expressed in % of remaining edges). For this type of distortion, the performance of both methods are very close.

Vitae

Pierre Alliez is Senior Researcher at Inria Sophia Antipolis - Mediterranée. He has authored several scientific publications and several book chapters on topics commonly referred to as geometry processing: Mesh compression, surface reconstruction, mesh generation, surface remeshing and mesh parameterization. He is an associate editor of the Computational Geometry Algorithms Library and an associate editor of the ACM Transactions on Graphics. He was co-chair of the Symposium on Geometry Processing in 2008 and of Pacific Graphics in 2010. He was awarded in 2011 a Starting Grant from the European Research Council on Robust Digital Geometry Processing.

Xavier Rolland-Nevière received a French Engineering degree in Telecommunications Systems from Telecom-Bretagne, France in September 2011. At the same time he was awarded a M.Sc. in Signal Processing from the University of Rennes 1, France. He is a Ph.D. candidate at Inria Sophia Antipolis - Mediterranée, and is currently with the Security & Content Protection Labs of Technicolor (ex Thomson) in Cesson-Sévigné (France). His research interests include watermarking, 3D mesh processing and video processing.

Gwenal Dorr is a Senior Scientist on Content Protection at Technicolor R&D France and is a Distinguished Member of its fellow Network. He has authored several scientific publications on topics commonly referred to as multimedia security: digital watermarking, content fingerprinting, passive forensics, digital rights management, etc. He is an associate editor of the IEEE Signal Processing Letters, the IEEE Transactions on Circuits and Systems for Video Technology, and the EURASIP Journal on Image and Video Processing. He will co-chair the Technical Committee on Information Forensics and Security from the IEEE Signal Processing Society in 2013.