Fitting Polynomial Surfaces to Triangular Meshes with Voronoi Squared Distance Minimization

Vincent Nivoliers[1,2], Dong-Ming Yan[1,3], and Bruno Lévy[1]

[1] Project ALICE / Institut National de Recherche en Informatique et en Automatique (INRIA) Nancy Grand Est, LORIA
[2] Institut National Polytechnique de Lorraine (INPL)
[3] Geometric Modeling and Scientific Visualization Center, King Abdullah University of Science and Technology (KAUST)
Vincent.Nivoliers@loria.fr, Dongming.Yan@inria.fr, Bruno.Levy@inria.fr

This paper introduces Voronoi Squared Distance Minimization (VSDM), an algorithm that fits a surface to an input mesh. VSDM minimizes an objective function that corresponds to a Voronoi-based approximation of the overall squared distance function between the surface and the input mesh (SDM). This objective function is a generalization of Centroidal Voronoi Tesselation (CVT), and can be minimized by a quasi-Newton solver. VSDM naturally adapts the orientation of the mesh to best approximate the input, without estimating any differential quantities. Therefore it can be applied to triangle soups or surfaces with degenerate triangles, topological noise and sharp features. Applications of fitting quad meshes and polynomial surfaces to input triangular meshes are demonstrated.

1 Introduction

We focus on the problem of fitting a surface $\mathcal{S}$ to an input mesh $\mathcal{T}$, under the following assumptions :

- An initial template $\mathcal{S}^{(0)}$ is available. For instance, if $\mathcal{T}$ is a topological sphere, $\mathcal{S}^{(0)}$ can be initialized as the bounding box of $\mathcal{T}$ (see Figure 1-center). For higher genus, some existing automatic or interactive methods may be used to construct the template $\mathcal{S}^{(0)}$ (see Section 4);
- the reconstructed surface $\mathcal{S}$ can be a polygon mesh or a polynomial surface;

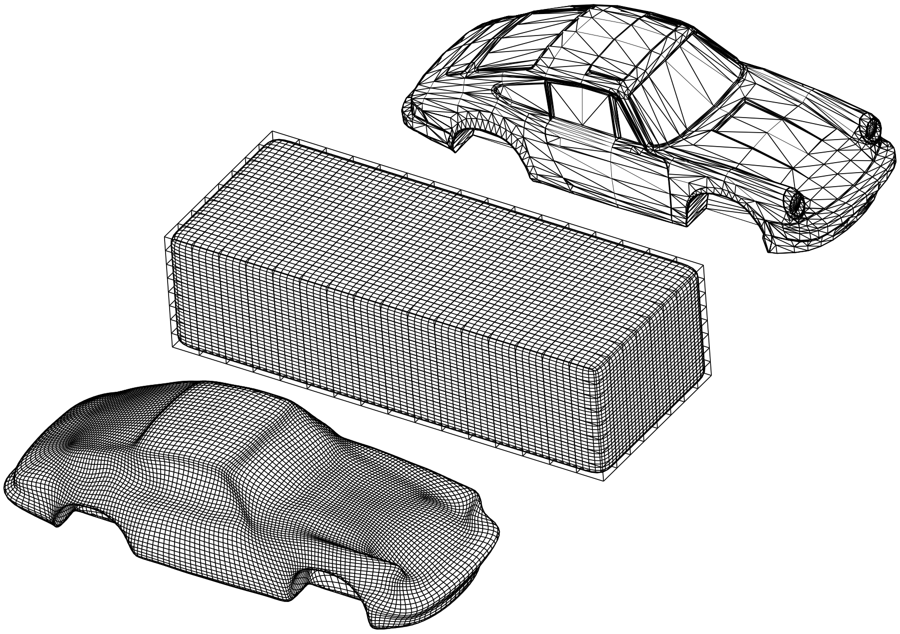


Fig. 1: *Given an input mesh $\mathcal{T}$ (top,* 2065 *vertices and* 4114 *facets) and a control mesh (898 vertices and 896 quads) in an initial position (center), VSDM minimizes the squared distance between $\mathcal{T}$ and the polynomial surface $\mathcal{S}$ defined by the control mesh. The Hausdorff distance between the result (bottom) and the input mesh (top) is* 0.554% *of the bounding box diagonal. Other views of the same data are shown further in the paper.*

- the input mesh $\mathcal{T}$ may have degenerate triangles and/or topological degeneracies such as T-junctions, holes or topological noise.

We introduce VSDM (Voronoi Squared Distance Minimization), an algorithm that fits the template $\mathcal{S}$ to the input mesh $\mathcal{T}$ by minimizing an objective function $\tilde{F}$ of the set of coordinates that determines $\mathcal{S}$, i.e. the vertices of a polygon mesh or the control points of a polynomial surface. Figure 1 shows an example of fitting a polynomial surface to an input triangulated mesh.

This paper makes the following contributions :

- definition of $\tilde{F}$ (Section 3.1), and proof that it converges to the integrated squared distance (Appendix A);
- solution mechanism to minimize $\tilde{F}$ (Sections 3.3, 3.4);
- some applications to quad mesh fitting and polynomial surface fitting (Section 3.5).

advantages:

1. VSDM can fit a surface $\mathcal{S}$ to an input mesh $\mathcal{T}$ even if the initialization $\mathcal{S}^{(0)}$ is far away from $\mathcal{T}$ (typically a bounding box);
2. unlike methods based on parameterization, VSDM can process meshes with sharp angles and skinny triangles;
3. VSDM adapts the orientation of the control mesh in a way that best approximates the input surface, without requiring computation of its curvature tensor.

limitations/uncovered aspects:

1. VSDM may generate pinchouts or overlaps, for instance when trying to fit a simple template $\mathcal{S}^{(0)}$ to a surface that has long protrusions / high Gaussian curvature. This can be fixed in most cases by designing a better template $\mathcal{S}^{(0)}$;
2. we do not prove the C^2 continuity of the objective function. However, in our empirical studies, the numerical optimization behaves well (see discussion in Section 3.4) ;
3. Dynamically modifying the topology of $\mathcal{S}$ is not adressed here. These topics will be studied in future works.

2 Background and previous work

Methods based on parameterization

Fitting splines was the motivation of early works in mesh parameterization for objects homeomorphic to a disc [8]. For fitting splines to objects of arbitrary genus, it is possible to use a parameterization defined over a base complex [7, 20], polycube maps [21, 22] or global parameterization methods [19, 12]. The relations between the curvature of the surface and the metric defined by the parameterization is studied in [10] and used to compute an anisotropic mesh that minimizes the approximation error. Since they require the estimates of differential quantities (gradients, curvature, shape operator ...), the methods above cannot be applied to meshes with degeneracies (skinny triangles, multiple components, holes, sharp creases). Our methods that directly minimizes the squared distance does not suffer from this limitation.

Methods based on point-to-point distances

To remesh surfaces, "shrink-wrap" methods [5, 9] iteratively project the template onto the input mesh while minimizing a regularization criterion. A similar idea can be applied to subdivision surfaces [15, 16], using an exact algorithm to find closest points on the subdivision surface and the exact evaluation of the subdivision surface. The "dual domain relaxation" method [25] uses some variants of Laplace surface editing to fit a template to the input mesh. Since they are based on point-to-point distances, the methods above can mostly do small corrections on the geometry, and have difficulties converging when the initialization is far away from the target surface. In contrast, VSDM can successfully fit a control mesh to a surface.

Squared distance minimization (SDM)

SDM was proposed by Pottmann et al. [18] for curve and surface fitting. The SDM framework fits a surface $\mathcal{S}$ to an input mesh $\mathcal{T}$ by minimizing an approximation of the objective function $E(\mathbf{X})$:

$$\begin{array}{l} E(\mathbf{X}) = F_{\mathcal{S}\to\mathcal{T}}(\mathbf{X}) + \lambda R(\mathbf{X}) \\ \text{where:} \\ F_{\mathcal{S}\to\mathcal{T}}(\mathbf{X}) = \int_{\mathcal{S}(\mathbf{X})} \parallel \mathbf{x} - \Pi_{\mathcal{T}}(\mathbf{x}) \parallel^2 d\mathbf{x} \\ \qquad R(\mathbf{X}) = \|\mathbf{L}\mathbf{X}\|^2 \end{array} \tag{1}$$

In this equation, $\mathbf{X} = [\mathbf{x}_i]_{i=0}^n$ denotes the coordinates that determine $\mathcal{S}$ and $\Pi_{\mathcal{T}}(\mathbf{x})$ denotes the projection of $\mathbf{x}$ onto $\mathcal{T}$, i.e. the point of $\mathcal{T}$ nearest to $\mathbf{x}$. The term $R(\mathbf{X})$ is a quadratic regularization energy and $\mathbf{L}$ a discretization of the Laplacian. The regularization factor λ lets the user choose a tradeoff between the smoothness of $\mathcal{S}$ and the fitting criterion.

Wang et al. [23] showed that SDM can be characterized as a quasi-Newton method and they applied it to B-spline curve fitting. Cheng et al. [2, 3] proposed a subdivision surface fitting algorithm based on SDM. In the methods above, the approximation of the integral is based on both a point-sampling $\mathbf{X} = [\mathbf{x}_i]_{i=1}^n$ of $\mathcal{S}$ and a point-sampling $\mathbf{Y} = [\mathbf{y}_j]_{j=0}^m$ of $\mathcal{T}$. The approximation has several variants that correspond to Taylor expansions of different orders (see Figure 2).

The squared distance between $\mathbf{x}_i$ and $\mathcal{T}$ can be replaced by :

- Point Distance (PD): the squared distance to the nearest sample $\mathbf{y}_j$ (order 0 approximation);
- Tangent Distance (TD): the squared distance to the nearest point on the tangent plane of the nearest sample (order 1 approximation);

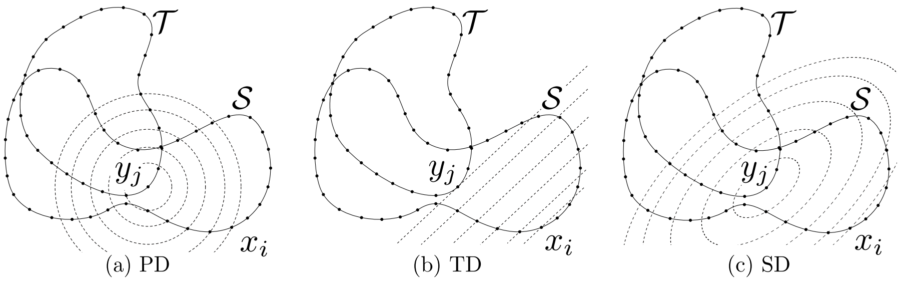


Fig. 2: *Illustration of different approximations of SDM. The dashed lines represent iso lines of the distance function to the sample*

- Squared Distance (SD): the order 2 approximation of the squared distance around $\mathbf{y}_j$.

3 Voronoi Squared Distance Minimization

SDM requires an accurate estimation of the curvature tensor on $\mathcal{T}$, which may be not available if $\mathcal{T}$ is a triangle soup or a CAD mesh with many skinny triangles. Therefore, to allow processing such degenerate input meshes, VSDM directly uses the local geometry of $\mathcal{T}$ around $\mathbf{y}_j$ by integrating the squared distance function over a small patch (a restricted Voronoi cell), as explained in the next subsection.

In addition, we note that $F_{\mathcal{S}\to\mathcal{T}}$ vanishes whenever $\mathcal{S}$ matches a subset of $\mathcal{T}$ (instead of the totality of $\mathcal{T}$). Therefore, to avoid degenerate minimizers that partially match $\mathcal{T}$, we propose to minimize a symmetrized version of SDM given by $F_{\mathcal{S}\to\mathcal{T}} + F_{\mathcal{T}\to\mathcal{S}}$. The benefit of the symmetrized formulation is demonstrated later (subsection 3.3).

3.1 Definition

VSDM minimizes an approximation of the following objective function :

$$F(\mathbf{X}) = F_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) + F_{\mathcal{S}\to\mathcal{T}}(\mathbf{X}) + \lambda R(\mathbf{X}) \tag{2}$$

Let us consider the first term $F_{\mathcal{T}\to\mathcal{S}}$. Using a sampling $\mathbf{X}$ of $\mathcal{S}$, we make the following approximation $\|\mathbf{y} - \Pi_{\mathcal{S}}(\mathbf{y})\| \simeq \min_i \|\mathbf{y} - \mathbf{x}_i\|$. Replacing the integrand of $F_{\mathcal{T}\to\mathcal{S}}$ gives :

$$\begin{aligned} F_{\mathcal{T}\to\mathcal{S}} &= \int_{\mathcal{T}} \|\mathbf{y} - \Pi_{\mathcal{S}}(\mathbf{y})\|^2 d\mathbf{y} \\ &\simeq \int_{\mathcal{T}} \min_i \|\mathbf{y}-\mathbf{x}_i\|^2 d\mathbf{y} \\ &= \sum_i \int_{\Omega_i \cap \mathcal{T}} \|\mathbf{y}-\mathbf{x}_i\|^2 d\mathbf{y} \end{aligned}$$

where Ω_i denotes the 3D Voronoi cell of $\mathbf{x}_i$. We shall now give the definition of the approximation $\tilde{F}$ of F minimized by VSDM :

$$\begin{aligned} \tilde{F}(\mathbf{X}) &= \tilde{F}_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) + \tilde{F}_{\mathcal{S}\to\mathcal{T}}(\mathbf{X}) + \lambda \underbrace{\mathbf{X}^t\mathbf{L}^2\mathbf{X}}_{R(\mathbf{X})} \\ &\text{where :} \\ \tilde{F}_{\mathcal{T}\to\mathcal{S}} &= \sum_{\mathbf{x}_i \in \mathbf{X}} \int_{\mathcal{T}\cap\Omega_i} \| \mathbf{y} - \mathbf{x}_i \|^2 \, d\mathbf{y} \\ \tilde{F}_{\mathcal{S}\to\mathcal{T}} &= \sum_{\mathbf{y}_j \in \mathbf{Y}} \int_{\mathcal{S}\cap\Omega_j} \| \mathbf{x} - \mathbf{y}_j \|^2 \, d\mathbf{x} \end{aligned} \tag{3}$$

The matrix $\mathbf{L}$ is the uniform graph Laplacian of $\mathcal{S}$. The influence of the regularization factor λ is illustrated in Figure 3. Ω_i denotes the Voronoi cell of $\mathbf{x}_i$ in the Voronoi diagram of $\mathbf{X}$, and Ω_j the Voronoi cell of $\mathbf{y}_j$ in the Voronoi diagram of $\mathbf{Y}$ (see Figure 4).

3.2 Convergence to the continuous objective function

The VSDM approximation replaces the nearest point on $\mathcal{S}$ with the nearest sample of $\mathbf{X}$ (in the term $F_{\mathcal{T}\to\mathcal{S}}$) and the nearest point on $\mathcal{T}$ with the nearest sample of $\mathbf{Y}$ (in the term $F_{\mathcal{S}\to\mathcal{T}}$). The accuracy of the approximation depends

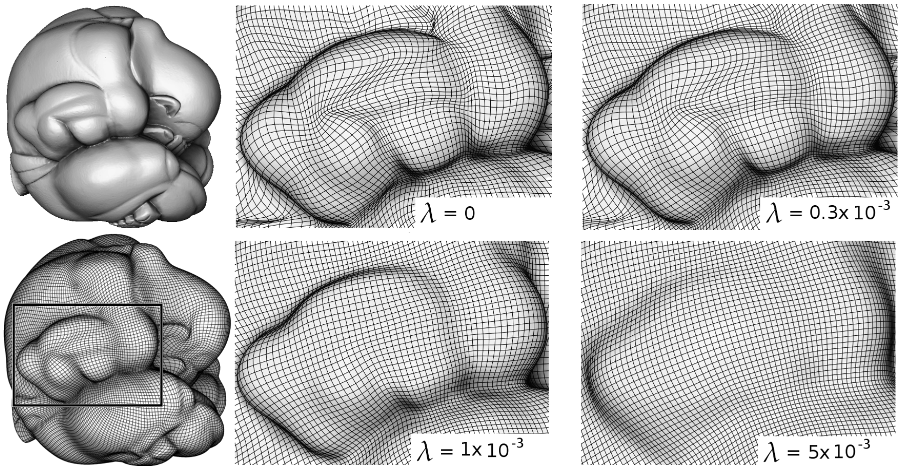
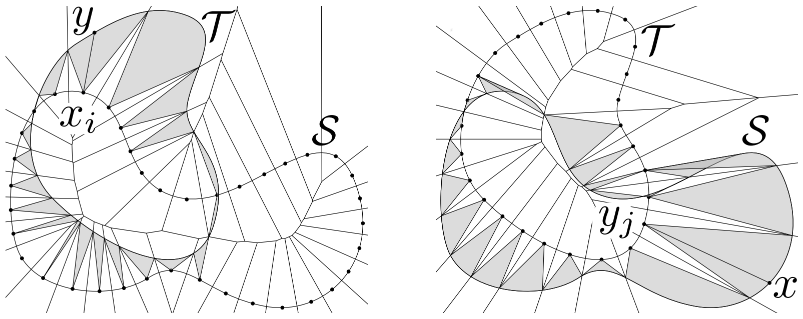


Fig. 3: *Influence of the regularization factor λ on subdivision surface fitting.*



(a) $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$

(b) $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$

Fig. 4: *Illustration of the terms of the VSDM objective function. The shaded regions represent for each sample $\mathbf{x}_i$ (resp. $\mathbf{y}_j$) the portion $\mathcal{T} \cap \Omega_i$ (resp. $\mathcal{S} \cap \Omega_j$) of the other surface whose squared distance with respect to the sample is integrated.*

on the *density* of the point sets $\mathbf{X}$ and $\mathbf{Y}$ used to sample $\mathcal{S}$ and $\mathcal{T}$ respectively. The density of a sampling is formalized by the notion of ε-sampling [1]. A point set $\mathbf{X}$ is an ε-sampling of a surface $\mathcal{S}$ if for any point $\mathbf{x}$ of $\mathcal{S}$ there is a point $\mathbf{x}_i$ in $\mathbf{X}$ such that $\|\mathbf{x}_i - \mathbf{x}\| < \varepsilon \ \mathrm{lfs}(\mathbf{x})$ where $\mathrm{lfs}(\mathbf{x})$ denotes local feature size (distance to medial axis of $\mathcal{S}$). $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$ satisfies the following property (proved in Appendix A).

Property 1. Given $\mathbf{X}$, an ε-sampling of $\mathcal{S}$, we have :

$$\lim_{\varepsilon\to 0} \tilde{F}_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) = F_{\mathcal{T}\to\mathcal{S}}(\mathbf{X})$$

The same property is satisfied by the symmetric term $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$ if $\mathbf{Y}$ is an ε-sampling of $\mathcal{T}$. Therefore, if $\mathbf{X}$ and $\mathbf{Y}$ are dense enough, $\tilde{F}$ is a good approximation of F. Note that ε-sampling is not defined for non-smooth surfaces. In all our experiments, we took $\mathbf{X}$ as the vertices of $\mathcal{S}$ and $\mathbf{Y}$ as a sampling of $\mathcal{T}$ with the same number of vertices as $\mathbf{X}$, optimized by CVT [24].

3.3 Need for the symmetrized objective function

The term $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$ of $\tilde{F}$ corresponds to the objective function minimized by Restricted CVT. Therefore, omitting the term $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$ results in the objective function $\tilde{F}_{\mathcal{T}\to\mathcal{S}} + \lambda R(\mathbf{X})$, that can be minimized by a straightforward modification of the CVT quasi-Newton algorithm used in [14, 24], i.e. by adding the term $\lambda \mathbf{X}^t \mathbf{L}^2 \mathbf{X}$ to the objective function and $2\lambda \mathbf{L}^2 \mathbf{X}$ to the gradient. However, as noted before, the function $F_{\mathcal{T}\to\mathcal{S}}$ reaches a minimum whenever $\mathcal{S}$ is a superset of $\mathcal{T}$. Therefore, a minimizer of $F_{\mathcal{T}\to\mathcal{S}}$ may have spurious parts, as

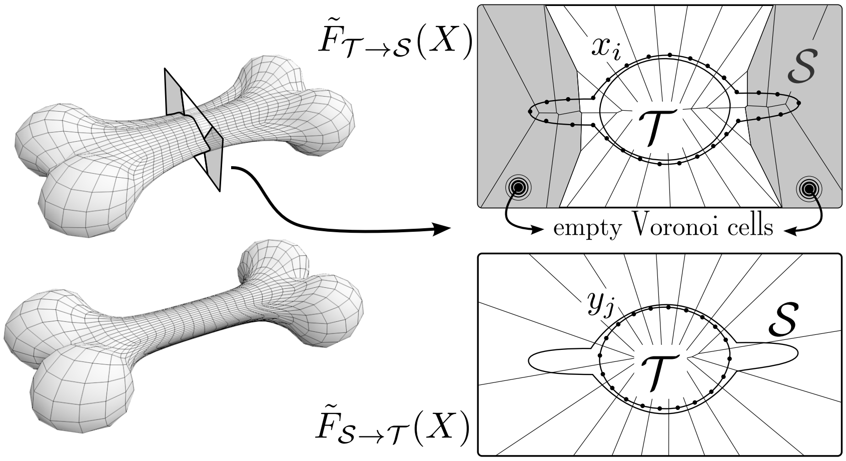

$\tilde{F}_{\mathcal{T}\to\mathcal{S}}(X)$
x_i
$\mathcal{S}$
$\mathcal{T}$
empty Voronoi cells
y_j
$\mathcal{S}$
$\mathcal{T}$
$\tilde{F}_{\mathcal{S}\to\mathcal{T}}(X)$

Fig. 5: *Top: the minimizer of $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$ has spurious parts that cannot be eliminated since their Voronoi cells do not intersect the input mesh $\mathcal{T}$. Bottom: the symmetrized $\tilde{F} = \tilde{F}_{\mathcal{T}\to\mathcal{S}} + \tilde{F}_{\mathcal{S}\to\mathcal{T}}$ detects and eliminates them.*

shown in Figure 5. These spurious parts correspond to the set $\mathcal{S} - \Pi_{\mathcal{S}}(\mathcal{T})$, that does not yield any term in $F_{\mathcal{T}\to\mathcal{S}}$. In terms of the discretization $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$, they correspond to Voronoi cells that have an empty intersection with $\mathcal{T}$.

3.4 Solution Mechanism

To minimize the function $\tilde{F} = \tilde{F}_{\mathcal{T}\to\mathcal{S}} + \tilde{F}_{\mathcal{S}\to\mathcal{T}} + \lambda R(\mathbf{X})$ in Equation 3, VSDM uses the L-BFGS algorithm [17, 13]. L-BFGS is a Newton-type algorithm, that uses successive evaluation of the function and its gradient to compute an approximation of the inverse of the Hessian. Although only the gradient is required in the computation, the objective function needs to be C^2 to ensure the proper convergence of the L-BFGS algorithms. We discuss here about the continuity of the three terms of $\tilde{F}$:

- The term $R(\mathbf{X})$ is a quadratic form (C^∞) ;
- the term $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$ corresponds to the *quantization noise power*, which is the objective function minimized by a centroidal Voronoi tesselation. It is of class C^2, except in some rarely encountered degenerate configurations (see [14] for a proof) ;
- the term $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$ is obtained by permuting the roles of the constant and variables in $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$. We will study its continuity in future work. Experimentally, it is regular enough for obtaining a stable behavior of L-BFGS.

In practice, implementing L-BFGS requires to evaluate $\tilde{F}(\mathbf{X}^{(k)})$ and $\nabla\tilde{F}(\mathbf{X}^{(k)})$ for a series of iterates $\mathbf{X}^{(k)}$ (see Algorithm 1) :

```
(1) $\mathbf{X}^{(0)} \leftarrow$ vertices of $\mathcal{S}^{(0)}$
(2) $\mathbf{Y} \leftarrow \varepsilon$-sampling of $\mathcal{T}$   ;   Compute $Vor(\mathbf{Y})$
while minimum not reached do
    (3) Compute $Vor(\mathbf{X}^{(k)})$, $Vor(\mathbf{X}^{(k)})|_{\mathcal{T}}$ and $Vor(\mathbf{Y})|_{\mathcal{S}}$
    (4) Compute $\tilde{F}_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}^{(k)})$ and $\nabla\tilde{F}_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}^{(k)})$
    (5) Compute $R(\mathbf{X}^{(k)})$ and $\nabla R(\mathbf{X}^{(k)})$
    (6) Compute $\tilde{F}_{\mathcal{S}\to\mathcal{T}}(\mathbf{X}^{(k)})$ and $\nabla\tilde{F}_{\mathcal{S}\to\mathcal{T}}(\mathbf{X}^{(k)})$
    (7) Compute $\tilde{F}(\mathbf{X})$ and $\nabla\tilde{F}(\mathbf{X})$
    (8) $\mathbf{X}^{(k+1)} \leftarrow \mathbf{X}^{(k)} + \mathbf{p}^{(k)}$   ;   Update $\mathcal{S}$ from $\mathbf{X}^{(k+1)}$
end
```

Algorithm 1: *fitting a polygon mesh using VSDM.*

In order to make our work reproducible, we further detail each step :

(2): the sampling $\mathbf{Y}$ of $\mathcal{T}$, used by $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$, is computed by the CVT algorithm in [24], with the same number of vertices as in $\mathbf{X}$;

(3): the Restricted Voronoi Diagrams $Vor(\mathbf{Y})|_{\mathcal{S}}$, $Vor(\mathbf{X}^{(k)})|_{\mathcal{T}}$ are computed as in [24] ;

(4),(5): $F_{\mathcal{T}\to\mathcal{S}}$ is the CVT objective function and R the regularization energy. The gradients are given by Equation 4 :

$$\begin{array}{ll} \nabla|_{\mathbf{x}_i}\tilde{F}_{\mathcal{T}\to\mathcal{S}} & = 2m_i(\mathbf{x}_i - \mathbf{g}_i) \\ \nabla R(\mathbf{X}) & = \nabla \mathbf{X}^t\mathbf{L}^2\mathbf{X} = 2\mathbf{L}^2\mathbf{X} \end{array} \tag{4}$$

where m_i and $\mathbf{g}_i$ denote the volume and the centroid of the restricted Voronoi cell $\Omega_i \cap \mathcal{T}$ [6] ;

(6): the term $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$ is obtained by exchanging the roles of $\mathcal{S}$ and $\mathcal{T}$ in $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$ and using the point set $\mathbf{Y}$ instead of $\mathbf{X}$. The computation of this term and its gradient are explained in the next paragraph ;

(8): $\mathbf{p}^{(k)}$ denotes the step vector computed by L-BFGS.

The function $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$ depends on the Voronoi diagram of $\mathbf{Y}$ restricted to $\mathcal{S}$ (see Figure 6). Each restricted Voronoi cell $\Omega_j \cap \mathcal{S}$ (colored polygons) is decomposed into a set of triangles. One of them $T = (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$ is highlighted. Each triangle T of the decomposition of $\Omega_j \cap \mathcal{S}$ contributes the following terms to $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$ and $\nabla\tilde{F}_{\mathcal{S}\to\mathcal{T}}$:

$$\tilde{F}^T_{\mathcal{S}\to\mathcal{T}} = \frac{|T|}{6} \sum_{1\le k\le l\le 3} (\mathbf{c}_k - \mathbf{y}_j)\cdot(\mathbf{c}_l - \mathbf{y}_j), \qquad \frac{d\tilde{F}^T_{\mathcal{S}\to\mathcal{T}}}{d\mathbf{X}} = \sum_{k=1}^{3} \frac{d\tilde{F}^T_{\mathcal{S}\to\mathcal{T}}}{d\mathbf{c}_k}\frac{d\mathbf{c}_k}{d\mathbf{X}} \tag{5}$$

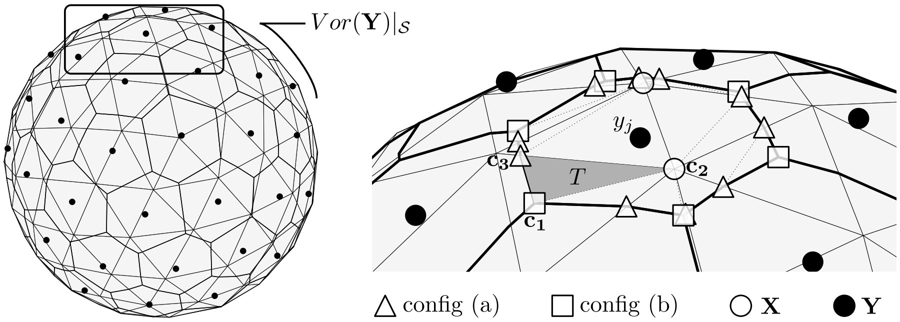


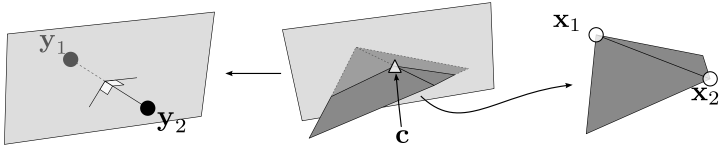
Fig. 6: *Computing the gradient of the symmetric term $\nabla \tilde{F}_{\mathcal{S}\to\mathcal{T}}$: configurations of the vertices of $Vor(\mathbf{Y})|_{\mathcal{S}}$.*

where $d\mathbf{A}/d\mathbf{B} = (\partial a_i/\partial b_j)_{i,j}$ denotes the Jacobian matrix of $\mathbf{A}$.

The set of possible configurations for a vertex $\mathbf{c}_k$ is similar to the combinatorial structure of the L_p-CVT function [11], with the exception that the roles of the variables and constants are exchanged. Each configuration yields a Jacobian matrix that propagates the derivatives of $\tilde{F}^T_{\mathcal{S}\to\mathcal{T}}$ from the $\mathbf{c}_k$'s to the $\mathbf{x}_i$'s. There are 3 possible configurations (see overview in Figure 6) :

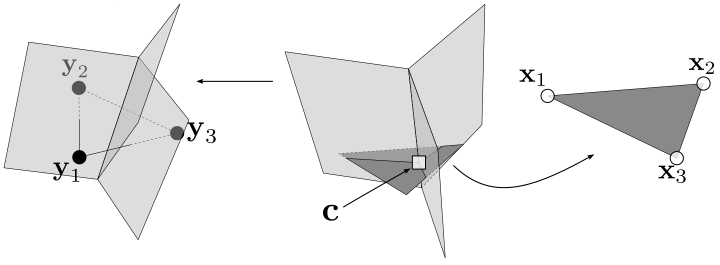
→ ○ $\mathbf{c}$ is a vertex $\mathbf{x}_i$ of $\mathcal{S}$ (then $d\mathbf{c}/d\mathbf{x}_i = \mathbf{I}_{3\times 3}$) ;

→ △ $\mathbf{c}$ has configuration (a) :



$\mathbf{c}$ corresponds to the intersection between the bisector of $[\mathbf{y}_1, \mathbf{y}_2]$ (left, plane shown in blue) and an edge $[\mathbf{x}_1, \mathbf{x}_2]$ of $\mathcal{S}$ (right). The Jacobian matrices $d\mathbf{c}/d\mathbf{x}_1$ and $d\mathbf{c}/d\mathbf{x}_2$ are given in Appendix B, Equation 13;

→ □ $\mathbf{c}$ has configuration (b) :



$\mathbf{c}$ corresponds to the intersection between the three bisectors of $[\mathbf{y}_1, \mathbf{y}_2]$, $[\mathbf{y}_2, \mathbf{y}_3]$, $[\mathbf{y}_3, \mathbf{y}_1]$ (left) and a facet $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ of $\mathcal{S}$ (right). The Jacobian matrices $d\mathbf{c}/d\mathbf{x}_1$, $d\mathbf{c}/d\mathbf{x}_2$ and $d\mathbf{c}/d\mathbf{x}_3$ are given in Appendix B, Equation 14.

3.5 Fitting Polynomial Surfaces

We consider now the problem of fitting a polynomial surface defined by its control mesh $\mathcal{C}$. At each iteration, we compute a polygonal approximation $\mathcal{S}$ of the polynomial surface. The vertices $\mathbf{X}$ of $\mathcal{S}$ are given as linear combinations of the control points $\mathbf{P}$:

$$\mathbf{X} = \mathbf{MP}$$

where $\mathbf{X} = [x_1 y_1 z_1 \dots x_n y_n z_n]^t$ denotes the coordinates at the vertices of $\mathcal{S}$, $\mathbf{P}$ denotes the coordinates at the control points. One may use the exact evaluation of the polynomial surface, or simply use the approximation obtained by subdividing the control mesh several times with De Casteljau's rule.

Polynomial surface fitting is done by minimizing the function $G(\mathbf{P}) = \tilde{F}(\mathbf{MP})$. This can implemented with a change of variable in the VSDM algorithm (see Algorithm 2) :

```
$\mathbf{P}^{(0)} \leftarrow$ vertices of $\mathcal{C}^{(0)}$
$\mathbf{Y} \leftarrow \varepsilon$-sampling of $\mathcal{T}$    ;    Compute $Vor(\mathbf{Y})$
while minimum not reached do
    $\mathbf{X} \leftarrow \mathbf{MP}^{(k)}$    ;    Update $\mathcal{S}$ from $\mathbf{X}$
    Compute $\tilde{F}(\mathbf{X})$ and $\nabla\tilde{F}(\mathbf{X})$ as in Algo. 1, steps (3) to (7)
    Compute $\nabla G(\mathbf{P}) = \mathbf{M}^t \nabla\tilde{F}(\mathbf{X})$
    $\mathbf{P}^{(k+1)} \leftarrow \mathbf{P}^{(k)} + \mathbf{p}^{(k)}$
end
```

Algorithm 2: *Fitting a polynomial surface.*

3.6 Feature-Sensitive Fitting

Using the algorithm above for fitting polynomial surfaces may result in oversmoothing sharp creases (Figure 7 center). However, this can be improved by injecting *normal anisotropy* [11] into the objective function $\tilde{F}$ (Figure 7 right). This changes the terms $\tilde{F}_{\mathcal{T}\to\mathcal{S}}$ and $\tilde{F}_{\mathcal{S}\to\mathcal{T}}$ as follows :

$$\begin{array}{ll} \tilde{F}^s_{\mathcal{T}\to\mathcal{S}} & = \sum\limits_{\mathbf{x}_i\in\mathbf{X}} \quad \sum\limits_{T\subset\mathcal{T}\cap\Omega_i} \int\limits_T \| \mathbf{A}_s(\mathbf{N}_T)(\mathbf{y}-\mathbf{x}_i) \|^2 \, d\mathbf{y} \\ \tilde{F}^s_{\mathcal{S}\to\mathcal{T}} & = \sum\limits_{\mathbf{y}_j\in\mathbf{Y}} \quad \sum\limits_{T\subset\mathcal{S}\cap\Omega_j} \int\limits_T \| \mathbf{A}_s(\mathbf{N}_j)(\mathbf{x}-\mathbf{y}_j) \|^2 \, d\mathbf{x} \\ \text{where :} & \\ \mathbf{A}_s(\mathbf{N}) & = (s-1) \begin{pmatrix} \mathbf{N}_x[\mathbf{N}]^t \\ \mathbf{N}_y[\mathbf{N}]^t \\ \mathbf{N}_z[\mathbf{N}]^t \end{pmatrix} + \mathbf{I}_{3\times 3} \end{array} \tag{6}$$

where the parameter $s \in (0, +\infty)$ specifies the importance of normal anisotropy. The normals are sampled from the input surface $\mathcal{T}$ in both terms, $\mathbf{N}_T$ is the normal of the triangle T, and $\mathbf{N}_j$ the normal to $\mathcal{T}$ at $\mathbf{y}_j$. Normal anisotropy is used in all the examples shown below.

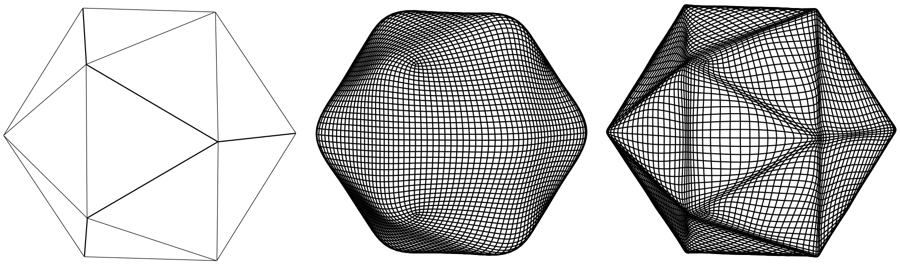


Fig. 7: *Influence of the feature-sensitive fitting on meshes with sharp creases (from left to right: original mesh, result without and with normal anisotropy).*

3.7 Implementation

For the Delaunay triangulation, we use CGAL (`www.cgal.org`). For the Restricted Voronoi Diagram computation (Section 3.4) and the normal anisotropy (previous subsection), we use the implementation provided with [11].

4 Results

We shall now show some results obtained with VSDM. In the results herein, the regularization term is set to $\lambda = 0.2 \times 10^{-3}$, the normal anisotropy is set to $s = 50$ and subdivision surfaces are approximated by subdividing the control mesh twice. Figures 8, 9, and 10 show the result obtained with an initial toroidal grid. Note on Figure 12 how the spacing of the iso-parameter line adapts to the features. Scanned meshes from AimAtShape can also be efficiently processed (see Figure 13). For each model, the result was obtained in less than 3 minutes on a 2 GHz machine.

Discussion and future work

The examples shown in the previous section were obtained automatically, by using the bounding box (or a toroidal mesh) as the initial control mesh. However, for shapes with an arbitrary genus or a complicated geometry, an initial control mesh is needed. Designing an initial control mesh may be also required to improve the quality of the surface. In future work, we will study the generation of an initial control mesh and/or the dynamic modification of the control mesh during the optimization.

Acknowledgements

The authors wish to thank Sylvain Lefebvre for a discussion (about an unrelated topic) that inspired this work, Rhaleb Zayer, Xavier Goaoc, Tamy

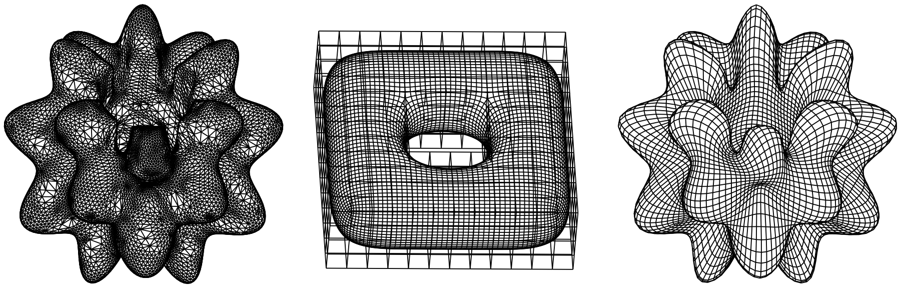


Fig. 8: *Fitting a polynomial surface to an object with toroidal topology. Left: input mesh (16.8k vertices, 33.6k facets); Center: initial control mesh (512 vertices and 512 quads) and surface; Right: result. The Hausdorff distance between the resulting surface and the input mesh is* 1.221% *(relative to the diagonal length of the bounding box, measured by Metro [4]).*

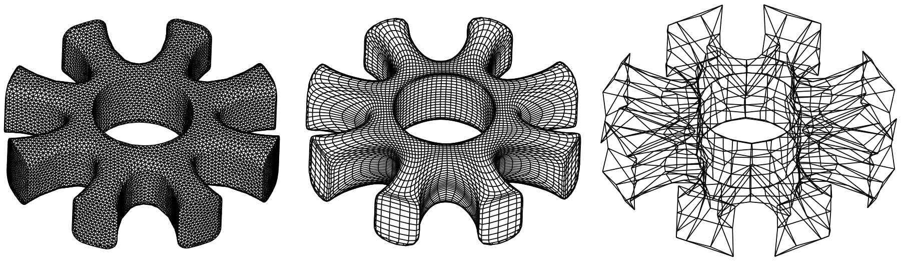


Fig. 9: *Another example, using the same initial toroidal control mesh as in Figure 8. Left: input mesh (10k vertices, 20k facets); Center: result; Right: result (control mesh with 512 vertices and 512 quads). Hausdorff distance is* 0.473% *bbox. diag.*

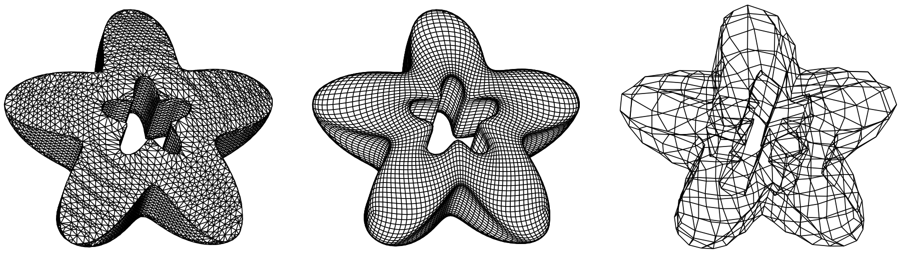


Fig. 10: *Another example, still using the same initial toroidal control mesh. Left: input mesh (5.2k vertices and 10.4k facets); Center: result; Right: result (control mesh with 512 vertices and 512 quads). The Hausdorff distance is* 0.699% *bbox. diag.*

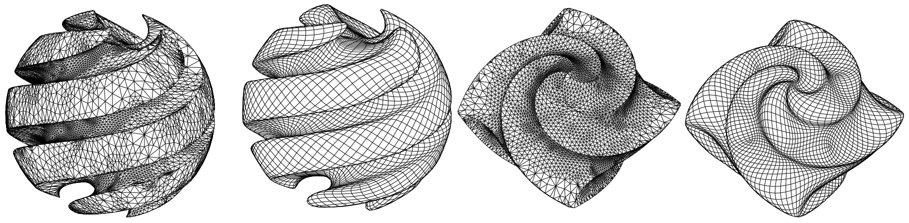


Fig. 11: *Other examples with geometrical shapes. Initialization from bounding box (386 vertices and 384 quads). Left: input mesh of sharp sphere (10.4k vertices, 20.9k facets) and result. Right: input mesh of octa-flower (7.9k vertics and 15.8k facets) and result. Hausdorff distances are* 1.072% *and* 0.706% *bbox. diag., respectively.*

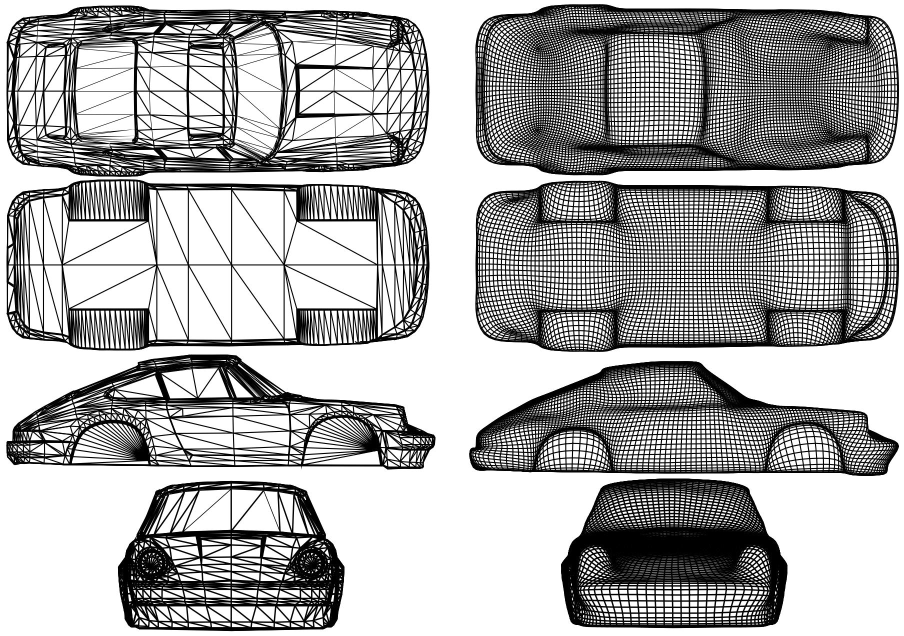


Fig. 12: *Different views of the example shown in Figure 1. The Hausdorff distance between the input and result is* 0.554% *of the bounding box diagonal.*

Boubekeur, Yang Liu and Wenping Wang for many discussions, Loic Marechal, Marc Loriot and the AimAtShape repository for data. This project is partly supported by the European Research Council grant GOODSHAPE ERC-StG-205693 and ANR/NSFC (60625202,60911130368) Program (SHAN Project).

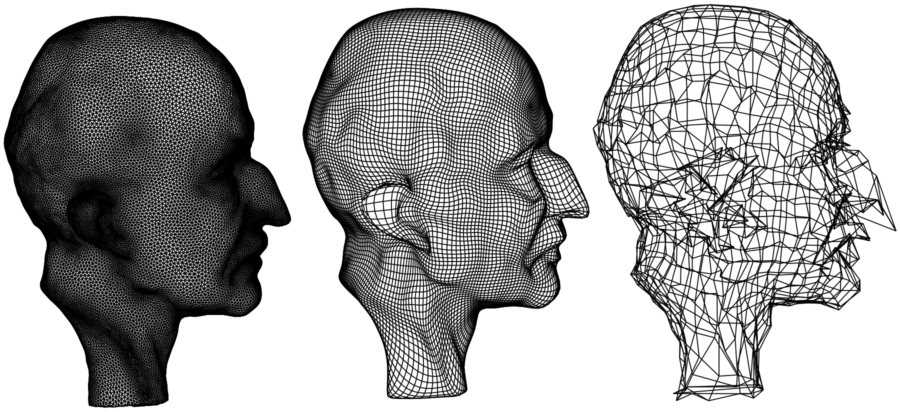


Fig. 13: *Fitting a Catmull-Clark subdivision surface to the statue of Max Planck (52.8k vertices and 105.6 facets). Initialization from bounding box (6257 vertices and 6255 quads). The Hausdorff distance is* 0.386% *of the bounding box diagonal.*

A Convergence to squared distance - Error bound

The following section proves that if $\mathbf{X}$ is an ε-sampling of $\mathcal{S}$ then:

$$\lim_{\varepsilon \to 0} \tilde{F}_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) = F_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) \tag{7}$$

Lemma 1. *Let* $\mathbf{y}$ *be a point of* $\mathcal{T}$ *and* $\mathbf{x}_i$ *its nearest point in* $\mathbf{X}$ *(see Figure 14). Let* $d = \| \mathbf{y} - \Pi_{\mathcal{S}}(\mathbf{y}) \|$ *and* $\tilde{d} = \| \mathbf{y} - \mathbf{x}_i \|$*. Then for* $\varepsilon < 2$ *the following bound is sharp:*

$$\tilde{d}^2 - d^2 \leq \varepsilon^2 lfs(\Pi_{\mathcal{S}}(\mathbf{y}))(lfs(\Pi_{\mathcal{S}}(\mathbf{y})) + d)$$

Proof. Let $\mathcal{B}_{Vor}$ be the ball centered at $\mathbf{y}$ passing through $\mathbf{x}_i$. This ball contains no point of $\mathbf{X}$.

Let $\mathcal{B}_{lfs}$ be the ball tangent to $\mathcal{S}$ at $\Pi_{\mathcal{S}}(\mathbf{y})$ on the opposite side of $\mathbf{y}$ and of radius $\mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))$ and c_{lfs} its center. Since $\mathbf{X}$ is an ε-sampling this ball also contains no point of $\mathbf{X}$.

Finally let $\mathcal{B}_{\varepsilon}$ be the ball centered at $\Pi_{\mathcal{S}}(\mathbf{y})$ of radius $\varepsilon \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))$. Since $\mathbf{X}$ is an ε sampling this ball contains no point of $\mathcal{S}$, and therefore no point of $\mathbf{X}$.

Since $\Pi_{\mathcal{S}}(\mathbf{y})$ is the nearest point of $\mathbf{y}$ on $\mathcal{S}$, $\mathbf{y}$, $\Pi_{\mathcal{S}}(\mathbf{y})$ and c_{lfs} are aligned and the problem is completely symmetric around the line joining them. Figure 14 shows a cut containing this axis.

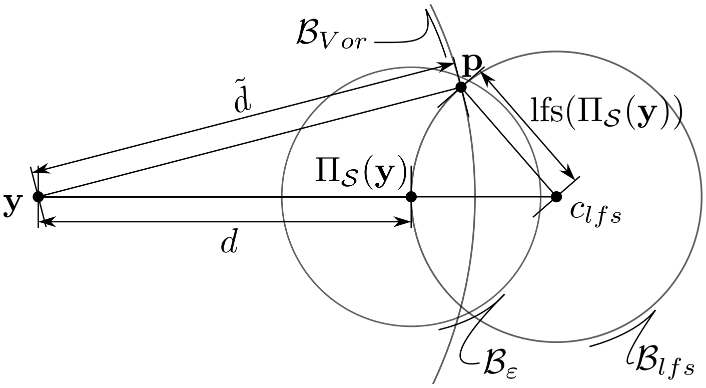

$\mathcal{B}_{Vor}$
$\mathbf{p}$
$\tilde{d}$
$\mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))$
$\Pi_{\mathcal{S}}(\mathbf{y})$
$\mathbf{y}$
c_{lfs}
d
$\mathcal{B}_{\varepsilon}$
$\mathcal{B}_{lfs}$

Fig. 14: *Configuration of the nearest point of* $\Pi_{\mathcal{S}}(\mathbf{y})$.

The bound follows from the fact that $\mathcal{B}_\varepsilon \not\subset \mathcal{B}_{Vor} \cup \mathcal{B}_{lfs}$. Let p be a point of $\mathcal{B}_{Vor} \cap \mathcal{B}_{lfs}$. This point exists since $\varepsilon < 2$ and $\mathcal{B}_\varepsilon$ is not included in $\mathcal{B}_{Vor}$. Using this point the previous condition can be reformulated as $p \in \mathcal{B}_\varepsilon$.

Using triangular identities in $(\Pi_{\mathcal{S}}(\mathbf{y}), c_{lfs}, \mathbf{p})$, we have:

$$\| \Pi_{\mathcal{S}}(\mathbf{y}) - \mathbf{p} \|^2 = 2\mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))^2(1 - \cos\alpha) \tag{8}$$

with α being the $(\mathbf{p}, c_{lfs}, \Pi_{\mathcal{S}}(\mathbf{y}))$ angle. Using the same identities in $(\mathbf{x}, c_{lfs}, \mathbf{p})$ we obtain:

$$\begin{aligned} \tilde{d}^2 &= (d + \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y})))^2 + \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))^2 \\ &\quad - 2(d + \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y})))\mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))\cos\alpha \\ \tilde{d}^2 - d^2 &= 2(d + \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y})))\mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))(1 - \cos\alpha) \end{aligned}$$

Using equation 8, $(1 - \cos\alpha)$ can be replaced:

$$\tilde{d}^2 - d^2 = (d + \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y})))\frac{\| \Pi_{\mathcal{S}}(\mathbf{y}) - \mathbf{p} \|^2}{\mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))} \tag{9}$$

Finally since $\mathbf{p}$ is inside $\mathcal{B}_\varepsilon$, we have:

$$\| \Pi_{\mathcal{S}}(\mathbf{y}) - \mathbf{p} \| \leq \varepsilon \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y})) \tag{10}$$

This finally provides the result:

$$\tilde{d}^2 - d^2 \leq \varepsilon^2 \mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y}))(\mathrm{lfs}(\Pi_{\mathcal{S}}(\mathbf{y})) + d) \tag{11}$$

This bound is sharp since it is reached whenever $\mathcal{S}$ is exactly $\mathcal{B}_{lfs}$ and $\mathbf{x}_i$ is located at $\mathbf{p}$.

This lemma leads to a global bound:

Proposition 1. *If $\mathcal{S}$ is different from a plane and bounded, then:*

$$\tilde{F}_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) - F_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) \leq \varepsilon^2 |\mathcal{T}| \sigma_{\mathcal{S}}(\sigma_{\mathcal{S}} + d_{\mathcal{H}}(\mathcal{T}, \mathcal{S}))$$

where $\sigma_{\mathcal{S}} = \sup\{lfs(\mathbf{x}), \mathbf{x} \in \mathcal{S}\}$

Proof. Since $\mathcal{S}$ is not a plane and bounded, σ exists. In addition, the definition of the Hausdorff distance gives us $d \leq d_{\mathcal{H}}(\mathcal{T}, \mathcal{S})$.

$$\begin{aligned}
e &= \tilde{F}_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) - F_{\mathcal{T}\to\mathcal{S}}(\mathbf{X}) \\
&= \int_{\mathcal{T}} \min_i \parallel \mathbf{y} - \mathbf{x}_i \parallel^2 d\mathbf{y} - \int_{\mathcal{T}} \parallel \mathbf{y} - \Pi_{\mathcal{S}}(\mathbf{y}) \parallel^2 d\mathbf{y} \\
&= \int_{\mathcal{T}} \min_i \parallel \mathbf{y} - \mathbf{x}_i \parallel^2 - \parallel \mathbf{y} - \Pi_{\mathcal{S}}(\mathbf{y}) \parallel^2 d\mathbf{y} \\
&\leq \int_{\mathcal{T}} \varepsilon^2 \sigma_{\mathcal{S}}(\sigma_{\mathcal{S}} + d_{\mathcal{H}}(\mathcal{T}, \mathcal{S})) d\mathbf{y} \\
&\leq \varepsilon^2 |\mathcal{T}| \sigma_{\mathcal{S}}(\sigma_{\mathcal{S}} + d_{\mathcal{H}}(\mathcal{T}, \mathcal{S}))
\end{aligned} \tag{12}$$

B Gradients of the symmetric term $\nabla \tilde{F}_{\mathcal{S}\to\mathcal{T}}$

Configuration (a) :

$$\begin{array}{l}
\frac{d\mathbf{c}}{d\mathbf{x}_1} = \mathbf{e}\mathbf{w}_1^t + (1-u)\mathbf{I}_{3\times 3} \\
\frac{d\mathbf{c}}{d\mathbf{x}_2} = \mathbf{e}\mathbf{w}_2^t + u\mathbf{I}_{3\times 3}
\end{array}
\quad \text{where :} \left\{ \begin{array}{r@{\;}l}
\mathbf{e} &= (\mathbf{x}_2 - \mathbf{x}_1) \\
\mathbf{n} &= (\mathbf{y}_2 - \mathbf{y}_1) \\
k &= \mathbf{n}.\mathbf{e} \\
h &= \frac{1}{2}\mathbf{n}.(\mathbf{y}_1 + \mathbf{y}_2) \\
u &= \frac{1}{k}(h - \mathbf{n}.\mathbf{x}_1) \\
\mathbf{w}_1 &= -\frac{1}{k^2}(h - \mathbf{n}.\mathbf{x}_2)\mathbf{n} \\
\mathbf{w}_2 &= \frac{1}{k^2}(h - \mathbf{n}.\mathbf{x}_1)\mathbf{n}
\end{array} \right. \tag{13}$$

Configuration (b) :

$$\begin{array}{l}
\frac{d\mathbf{c}}{d\mathbf{x}_1} = \mathbf{e}\mathbf{w}_1^t \\
\frac{d\mathbf{c}}{d\mathbf{x}_2} = \mathbf{e}\mathbf{w}_2^t \\
\frac{d\mathbf{c}}{d\mathbf{x}_3} = \mathbf{e}\mathbf{w}_3^t
\end{array}
\quad \text{where :} \left\{ \begin{array}{r@{\;}l}
\mathbf{e} &= (\mathbf{y}_1 - \mathbf{y}_2) \times (\mathbf{y}_1 - \mathbf{y}_3) \\
\mathbf{n} &= (\mathbf{x}_1 - \mathbf{x}_2) \times (\mathbf{x}_1 - \mathbf{x}_3) \\
k &= \mathbf{n}.\mathbf{e} \\
\mathbf{w}_1 &= ((\mathbf{x}_2 - \mathbf{x}_3) \times (\mathbf{x}_1 - \mathbf{c}) + \mathbf{n})/k \\
\mathbf{w}_2 &= ((\mathbf{x}_3 - \mathbf{x}_1) \times (\mathbf{x}_1 - \mathbf{c}))/k \\
\mathbf{w}_3 &= ((\mathbf{x}_1 - \mathbf{x}_2) \times (\mathbf{x}_1 - \mathbf{c}))/k
\end{array} \right. \tag{14}$$

References

1. N. Amenta and M. Bern. Surface reconstruction by Voronoi filtering. *Discrete and Computational Geometry*, 22(4):481–504, 1999.

2. K.-S. D. Cheng, W. Wang, H. Qin, K.-Y. K. Wong, H.-P. Yang, and Y. Liu. Fitting subdivision surfaces using SDM. In *Pacific Graphics conf. proc.*, pages 16–24, 2004.
3. K.-S.D. Cheng, W. Wang, H. Qin, K.-Y.K. Wong, H.-P. Yang, and Y. Liu. Design and analysis of methods for subdivision surface fitting. *IEEE TVCG*, 13(5), 2007.
4. P. Cignoni, C. Rocchini, and R. Scopigno. Metro: measuring error on simplified surfaces. *Comp. Graphics Forum*, 17(2):167–174, 1998.
5. H. Delingette. General object reconstruction based on simplex meshes. *IJCV*, 32(2):111–146, 1999.
6. Q. Du, V. Faber, and M. Gunzburger. Centroidal Voronoi tessellations: applications and algorithms. *SIAM Review*, 41(4):637–676, 1999.
7. M. Eck and H. Hoppe. Automatic reconstruction of B-spline surfaces of arbitrary topological type. In *Proc. ACM SIGGRAPH*, pages 325–334, 1996.
8. M.S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231–250, 1997.
9. L. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel. A shrink wrapping approach to remeshing polygonal surfaces. *Comp. Graphics Forum*, 18(3), 2001.
10. D. Kovacs, A. Myles, and D. Zorin. Anisotropic quadrangulation. In *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling*, pages 137–146, 2010.
11. B. Lévy and Y. Liu. L_p centroidal Voronoi tessellation and its applications. *ACM TOG (Proc. SIGGRAPH)*, 29(4):Article No. 119, 2010.
12. W.-C. Li, N. Ray, and B. Lévy. Automatic and interactive mesh to T-spline conversion. In *Symposium on Geometry Processing*, pages 191–200, 2006.
13. D.C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming: Series A and B*, 45(3):503–528, 1989.
14. Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, and C. Yang. On centroidal Voronoi tessellation — energy smoothness and fast computation. *ACM Trans. on Graphics*, 28(4):Article No. 101, 2009.
15. W. Ma, X. Ma, S.-K. Tso, and Z. Pan. A direct approach for subdivision surface fitting. *Comp. Aided Design*, 36(6), 2004.
16. M. Marinov and L. Kobbelt. Optimization methods for scattered data approximation with subdivision surfaces. *Graphical Models*, 67(5):452–473, 2005.
17. J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2006.
18. H. Pottmann and S. Leopoldseder. A concept for parametric surface fitting which avoids the parametrization problem. *Comp. Aided Geom. Design*, 20(6), 2003.
19. N. Ray, W.C. Li, B. Lévy, A. Scheffer, and P. Alliez. Periodic global parameterization. *ACM Trans. on Graphics*, 25(4):1460–1485, 2006.
20. J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe. Inter-surface mapping. *ACM TOG (Proc. SIGGRAPH)*, 23(3):870–877, 2004.
21. M. Tarini, K. Hormann, P. Cignoni, and C. Montani. Polycube-maps. *ACM TOG (Proc. SIGGRAPH)*, 23(3):853–860, 2004.
22. H. Wang, Y. He, X. Li, X. Gu, and H. Qin. Polycube splines. *Comp. Aided Design*, 40(6):721–733, 2008.
23. W. Wang, H. Pottmann, and Y. Liu. Fitting B-spline curves to point clouds by curvature-based sdm. *ACM Trans. on Graphics*, 25(2):214–238, 2006.
24. D.-M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. *Comp. Graphics Forum (Proc. SGP)*, 28(5):1445–1454, 2009.
25. I-C. Yeh, C.-H. Lin, O. Sorkine, and T.-Y. Lee. Template-based 3D model fitting using dual-domain relaxation. *IEEE TVCG*, 17(8):1178–1190, 2011.