Semi-Sharp Subdivision Surface Fitting Based on Feature Line Approximation

Guillaume Lavoué, Florent Dupont

Semi-sharp subdivision surface fitting based on feature lines approximation

Guillaume Lavoué [a,b,*], Florent Dupont [a,c]
[a] Université de Lyon, CNRS
[b] INSA-Lyon, LIRIS, UMR5205, F-69621, France
[c] Universite Lyon 1, LIRIS, UMR5205, F-69622, France

This paper presents an algorithm for approximating arbitrary polygonal meshes with subdivision surfaces, with the objective of preserving the relevant features of the object while searching the coarsest possible control mesh. The main idea is to firstly extract the feature lines of the object, and secondly construct the subdivision surface over this network. Control points are created by approximating these lines while the connectivity is built with respect to the anisotropy of the object. Our algorithm reinforces the similarity between the subdivision surface and the original shape by affecting an integer sharpness degree to each control edge in order to accurately reproduce the different curvature radii of corresponding fillets and blends.

1. Introduction

Finding an optimal and concise representation for a 3D model is particularly crucial in computer graphics and computer-aided design. Indeed, obtaining a synthetic representation of a shape, usually defined as a *redundant* dense polygonal mesh, is of interest for many applications: animation, compression, recognition and reverse engineering. Subdivision surfaces combine a lot of properties quite relevant for this issue: this model is very *compact*, can represent an *arbitrary topology*, allows a *local control* and is intrinsically *multi-resolution*. For these reasons, subdivision surfaces are more and more popular in computer graphics and have been integrated to the MPEG4 standard. In this context, approximating a dense verbose polygonal mesh with that model becomes even more pertinent.

Hence, we present an algorithm for subdivision surface fitting from arbitrary polygonal meshes, and particularly scanned models; this algorithm follows our previous approach [1] which was specifically designed for piecewise smooth hand-made CAD models. Our main objective is not to focus on approximation error but rather to preserve the relevant features of the object while searching the coarsest possible control mesh. Our algorithm involves three main parts: a first step extracts a smooth feature line network from the object, using segmentation and smoothing of the patch boundaries. A second process approximates these feature lines with subdivision curves and creates a coarse base mesh by linking corresponding control points. Finally, a relaxing process affects a sharpness degree to each control edge and optimizes the control point positions in order to fit the target object. The whole algorithm is summarized in Fig. 1. Our main contributions are the following:

- The global subdivision surface fitting framework based on feature line extraction and approximation, allowing to obtain a near optimal vertex number.
- The feature line extraction, which extracts smooth feature lines from arbitrary objects in a very simple way.
- The base mesh construction which is (1) independent of the connectivity of the target surface, (2) correct even for complex topology and (3) adapted to the anisotropy of the target object.
- The sharpness relaxation, which assign to each control edge a semi-sharpness degree according to the curvature of the target surface.

2. Previous work

2.1. Subdivision surface fitting

Many authors have investigated subdivision surface fitting, since this issue is quite interesting for compression, reverse engineering, etc. Most of the existing algorithms rely on the same scheme: first a coarse base control mesh is constructed by one of the following approaches: simplifying the original dense mesh [2–5], face clustering [6], triangulating an octree partition [7], shrinking an initial mesh towards the surface [8] or global parameterization and quad dominant remeshing as considered by Lévy et al. [9].

* Corresponding author. Tel.: +33 4 72 43 71 36.
E-mail address: glavoue@liris.cnrs.fr (G. Lavoué).

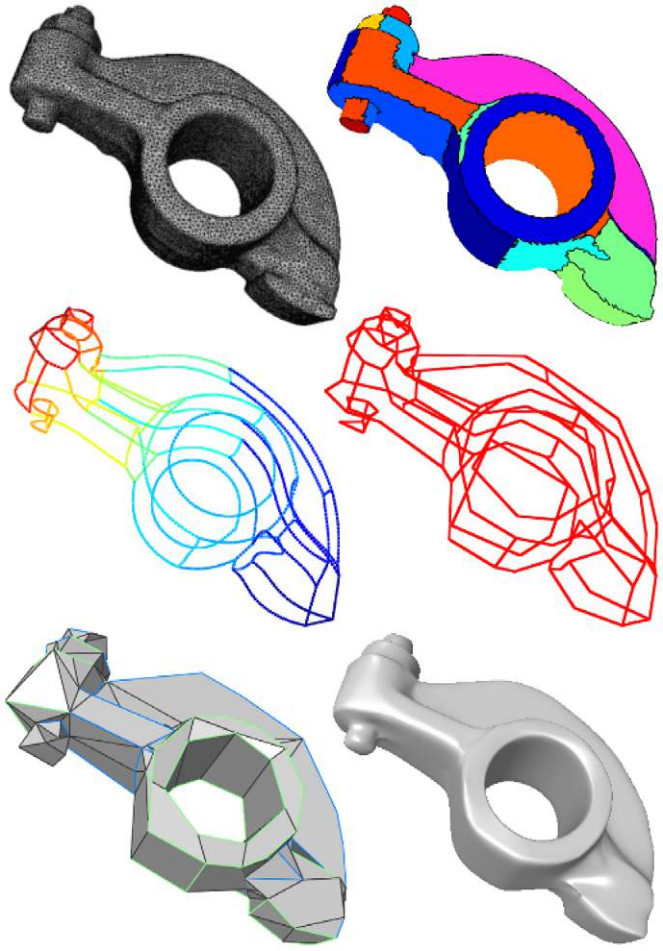

Fig. 1. Overview of our subdivision surface fitting algorithm. *Top row*: The input *RockerArm* mesh (15K vertices) and result of the segmentation (35 regions). *Middle row*: Feature line network (*left*) and control polygon network created from subdivision curve approximation (*right*). *Bottom row*: Subdivision control mesh (160 vertices) with different sharpness degrees and associated limit surface.

Once this coarse base mesh has been constructed, a second step optimizes its geometry in order to fit exactly the target object using a global [10,2,4,3] or local [8,11] distance minimization. Since the base mesh does not own a perfect connectivity, several authors have also investigated connectivity optimization procedures. Hoppe et al. [10] optimize the connectivity by trying to collapse, split or swap each edge of the control polyhedron. Their algorithm produces high quality models but need of course an extensive computing time. Recently, Marinov and Kobbelt [4] subdivide faces associated with high errors and flip some edges to regularize vertex valences, similar to [7]. Lavoué et al. [1] also consider such local enrichment process piloted by the error distribution. Finally, Lévy et al. [9] optimize the connectivity of the control mesh by analysing curvature directions of the target surface.

2.2. Feature preserving remeshing

Very recent remeshing techniques [12–17] are mostly quad-dominant and outline the importance for the connectivity of a mesh to follow the salient features of the object and to align with the geometry (sharp edges, lines of curvature etc.). The main reason is given by D'Azevedo [18]: the convergence is improved by such alignment, for both remeshing or fitting. This is linked to the concept of optimality: for a given number of elements, a mesh best approximates a smooth surface if its connectivity follows the lines of curvature. This feature alignment issue is as well relevant for subdivision surface fitting: the control mesh must respect and follow the features of the object and especially if the goal is to provide fewer vertices as possible. Our algorithm considers segmentation to find feature lines, and then curvature tensor analysis to complete the connectivity. Our algorithm bears similarities with the recent remeshing technique from Marinov and Kobbelt [16] which samples equidistantly feature lines to remesh the target object. However, our objective is not to produce a *nice* control mesh but rather an *efficient* subdivision surface that correctly approximates a given shape while containing as less control points as possible.

2.3. Lofting

This feature line preservation shares some similarities with another class of algorithms, often referred to as *lofting* and of particular interest for designers. These algorithms start from a network of curves and generate a smooth surface which interpolates this network. Subdivision surfaces represent a powerful framework for this task [19–22]. Our algorithm bears many similarities with a lofting scheme since it considers above all feature lines to construct the subdivision surface.

A lofting algorithm has to resolve three principal difficulties: firstly the curve network (usually a set of B-Splines) has to be compatible with a smooth or piecewise smooth surface, in other words, is it possible to construct a smooth surface over this network? Schaefer et al. [22] define the curves with control polygons associated with a specific subdivision scheme to insure this property. Since our feature lines directly come from a real object to approximate, we do not encounter compatibility problems.

The second major issue is the *skinning* step, which consists is constructing the connectivity of the base control mesh. Basically many authors consider that the curve network is defined by a control polygon network, thus for each cycle of polygons bounding each patch, they construct the corresponding polyhedron by linking boundary control points and creating some others in order to have mostly quadrilateral and regular faces. Our scheme considers only boundary control points and links them according to curvature directions in order to obtain the most compact and optimal base mesh.

Most of the authors consider Catmull–Clark subdivision surfaces in their lofting algorithm; since this kind of surface is not theoretically capable of interpolating a net of curves, they modify the original scheme by considering special rules near the curve network [19,22], or by introducing portions of surface that specifically define the curve like the *polygonal complexes* from Nasri [20,21]. In our case we consider a simple existing subdivision scheme [23], since we do not search a perfect interpolation but rather a quite good approximation.

2.4. Subdivision semi-sharpness

Many subdivision rules exist, some of them are adapted for triangular control meshes, like Loop [24], and others are adapted for quadrilateral ones, like Catmull–Clark [25]. We have chosen the hybrid quad/triangle scheme developed by Stam and Loop [23], since we want to adapt the control mesh connectivity to the shape of the input object and thus we may obtain faces from different degrees. This scheme reproduces Catmull–Clark on quad regions and Loop on triangle regions. The control mesh is firstly linearly subdivided and then each point is replaced by a linear combination of itself and its direct neighbours following a smoothing mask (see Fig. 2, left).

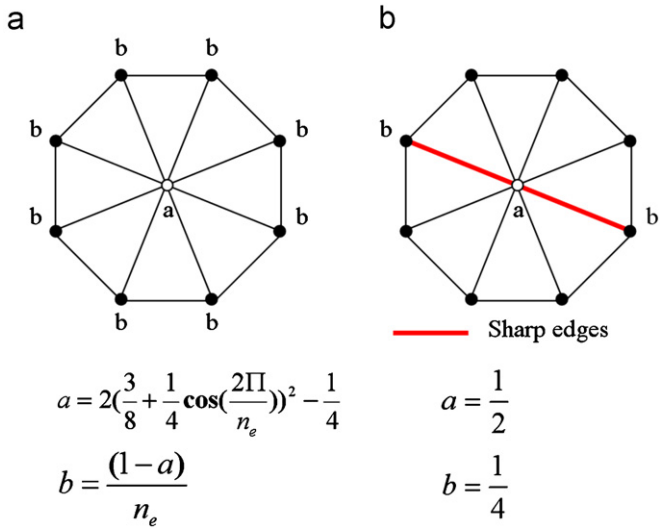

Fig. 2. Smoothing masks for Loop [24] subdivision rules. (a) Standard vertex. (b) *Sharp* (or boundary) vertex. *a* and *b* represent, respectively, the weights associated with a vertex and its neighbourhood in the smoothing operation.

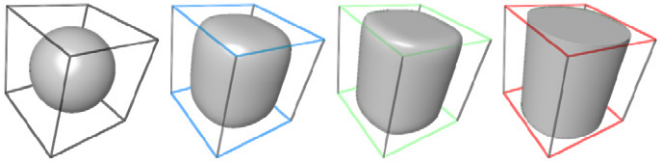

Fig. 3. Example of semi-sharp edges. The control mesh is the cube drawn in wireframe. Smooth edges are black and colour edges are semi-sharp. *From left to right*: Sharpness degree: 0 (black), 1 (blue), 2 (green) and infinite (red).

Special rules have been introduced by Hoppe et al. [10] for handling boundary edges, in such a way that the boundary curve of the limit surface does not depend on any interior control vertices. These rules can also be used to introduce sharp creases (see Fig. 3, right). Fig. 2 (right) illustrates the smoothing coefficients for vertices shared by two sharp edges.

These infinite sharp edges (see Fig. 3, right), introduced by Hoppe et al. [10] are quite convenient to represent piecewise smooth surfaces. However, real-world surfaces are never infinitely sharp, thus DeRose et al. [26] have introduced *semi-sharp* edges whose sharpness is an integer *sh* which can vary from zero (meaning smooth) to infinite. This semi-sharp subdivision is then processed by using sharp rules during the first *sh* subdivision steps, followed by the use of smooth rules for subsequent subdivision steps.

With this semi-sharpness concept, the same control edge can represent blends or fillets associated with different curvature radii (see Fig. 3). Our algorithm assigns to each edge of the control mesh the appropriate sharpness degree; to our knowledge no other existing algorithm carries out such a process.

3. Overview

Given an input polygonal mesh, we firstly process a decomposition into several regions R_i, using a modified version of the segmentation algorithm (VSA) from Cohen-Steiner et al. [27] (see Fig. 1, top right). We then extract the network of corresponding boundaries and apply a smoothing mask to obtain a smooth feature line network (see Section 4, see Fig. 1, middle left).

The feature lines are then approximated with subdivision curves, near optimal in terms of control points number, using the algorithm from Lavoué et al. [28]. We thus obtain a control polygon network (see Fig. 1, middle right). Each region R_i is then treated separately: from the control polygon surrounding the region, we create edges and facets by linking control points with respect to its lines of curvature. We obtain a set of control meshes M_i which are then assembled together; boundary edges between them are marked as *sharp* in order to fit correctly the input object after subdivisions. We obtain the *sharp control mesh* (see Section 5).

At this point, we have created a piecewise smooth subdivision surface with sharp edges at the emplacement of extracted feature lines (boundaries between regions R_i). Since this sharpness does not necessarily correspond to the object aspect, a process relaxes the sharpness by associating to each control edge an integer *sharpness degree* instead of a Boolean value (sharp or not). For this task we analyse the curvature of the input surface around feature lines. Finally, since changing the sharpness of the control mesh induces a shrinking of the limit surface, we perform a geometric optimization by iteratively relocating control points in order to minimize a global quadratic distance to the input surface (see Section 6, see Fig. 1, bottom left).

This algorithm improves over previous work [1] with the following improvements:

- Previous algorithm was limited to carefully designed CAD mechanical objects with optimized triangulation, whereas our new algorithm can approximate arbitrary models and particularly scanned objects.
- The feature line extraction of our previous algorithm was only suited to CAD models: it was based on a specific CAD segmentation step [29]. Our new algorithm produces smooth feature lines for arbitrary models.
- The creation of the base control mesh, in particular the *edge score* (see Section 5.2), was improved regarding to robustness and quality of the connectivity.
- Previous algorithm was creating sharp edges along feature lines. While this is well suited to some specific objects, it is not for all. Our new algorithm introduces sharpness relaxation which assigns to each control edge the appropriate integer sharpness degree.
- A geometric optimization was processed for every region separately and thus with few degrees of freedom. We introduce a global optimization process which allows minimizing a global asymmetric error between the target object and the approximating subdivision surface.

4. Feature line extraction

Feature lines of a surface carry the visually most salient characteristics. They are usually described as local extrema of principal curvatures along corresponding principal directions. Several algorithms exist to extract smooth feature lines from an input mesh [30,31], they are mostly based on computation of high order derivatives of principal curvatures, then thresholding and smoothing processes.

Existing algorithms are quite complex and provide a set of smooth lines which are not especially connected and thus do not always form a partition of the mesh; however, our algorithm needs a partition into regions to construct the topology and connectivity of the base control mesh. Hence, we define our

feature lines as the boundaries of a set of regions R_i issued from an appropriate segmentation of the mesh.

4.1. Segmentation

The goal is to obtain a partition of the mesh, such as the corresponding boundaries represent a coherent set of feature lines which aligns to geometric salient parts and does not contain too many unnecessary lines. Hence, the segmentation has to create a partition such as each region R_i bears a geometrical meaning.

The VSA algorithm from Cohen-Steiner et al. [27] complies well with our requirements: it is fully automatic and decomposes the mesh into a set of regions, as flat as possible, aligning with the geometric structure of the input mesh and capturing its anisotropy. The VSA is a global optimization method, starting from initial seeds and updating the shape of the regions at each iteration. Of course, according to the initial seed positions the algorithm can fall to a local minimum. Hence we adopt an incremental technique, quite similar to the *farthest-point initialization* proposed by the authors: we add one region at a time, perform a partitioning (with a fixed number of iterations), and then add a new region at the maximum error position. Fig. 4 (left) illustrates the segmentation of the *RockerArm* model into 45 regions.

Regarding to our feature line extraction objective, the standard VSA algorithm is not completely satisfactory, since it produces only approximately flat regions. For instance the cylindrical part at the centre of the object in Fig. 4 (left) is cut into several regions whereas we would rather require one single region, according to our feature line definition (local extrema of principal curvatures along corresponding principal directions). Moreover, each region is further approximated with a subdivision surface constructed by linking its boundary control points; this kind of subdivision surface is able to represent not only flat regions but also anisotropic parts (i.e. surfaces with clearly notable curvature direction, like elliptic or parabolic parts for instance).

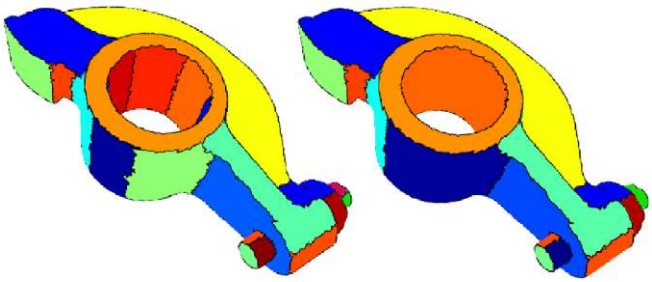

Fig. 4. Segmentation of the *RockerArm* model. Results of the VSA (45 regions) (*left*). Results after merging (35 regions) (*right*).

These reasons have led us to process a region merging after the VSA, according to anisotropy similarity: we merge two regions R_i and R_j if they share the same curvature value and the same minimum curvature direction. We simply process an anisotropy similarity score (*ASC*) between two regions R_i and R_j:

$$ASC(R_i, R_j) = \|d^i_{min} - d^j_{min}\| \times |c^i_{max} - c^j_{max}| \quad (1)$$

d^i_{min} and c^i_{max} are, respectively, the minimum curvature direction and maximum curvature value of the ith region. These values are calculated by averaging vertex values over the region. For each vertex, the curvature tensor has been calculated using the normal cycle algorithm [32] and the principal curvature values and directions have been extracted; they correspond, respectively, to the eigenvalues and eigenvectors of the curvature tensor.

We then construct a region adjacency graph, and merge iteratively pairs of regions associated with smallest scores. The merging operation stops when a fixed number of regions (chosen by the user) is obtained or when a minimum score threshold is reached; in practice such threshold is hard to find *a priori* but can be learned from training data for instance. Fig. 4 (right) illustrates the partition after 10 merging operations processed on the segmented model on the left. Regions associated with a similar anisotropy have been correctly merged.

4.2. Feature line smoothing

Our feature line network is represented by the network of boundaries between regions of the partition. This network is composed with sets of connected pieces of boundary (polygonal curves) separated by anchor vertices (vertices adjacent to at least three regions). These paths between regions are quite jagged since the connectivity of a scanned model, for instance, does not exactly follow the natural features of the object, on the contrary to hand-designed CAD parts. Our further process needs smooth feature lines, particularly for the curve approximation step. Thus we process a Laplace smoothing of the polygonal curves representing each piece of boundary (anchor vertices do not move). The main disadvantage of Laplace curve smoothing is the shrinkage effect that deviates the boundary polyline from the surface; fortunately it is not a problem if the curves do not lie precisely on the mesh since our goal is not to produce a perfect fitting but rather to have a correct approximation associated with a very coarse control mesh. Moreover, the further global optimization process

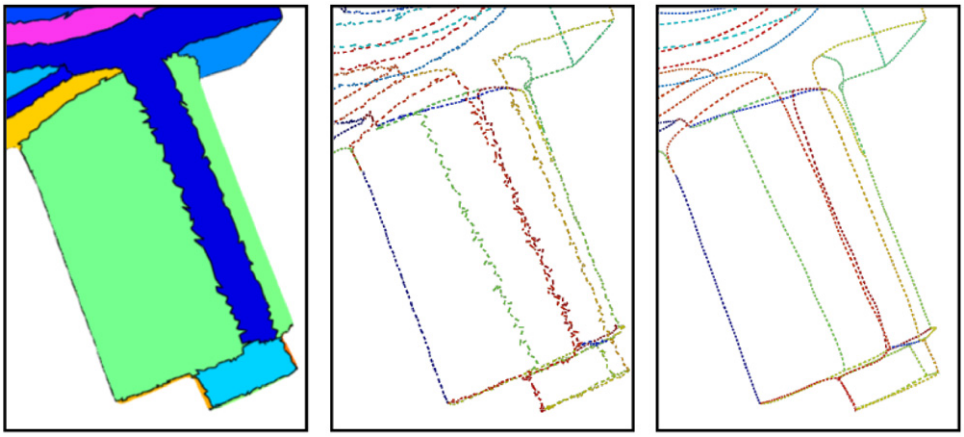

Fig. 5. A part of the segmented *Blade* mesh (15K vertices, 45 regions) (*left*), jagged feature line network (*middle*), smoothed feature line network (*right*).

(see Section 6.2) will reduce this shrinkage resilient error. Fig. 5 shows the segmented *Blade* object (left), the jagged boundary network (centre) and the feature line network after smoothing (right).

5. *Sharp control mesh* construction

Our goal is to create the coarsest possible base control mesh, while respecting the shape and the main features of the input object. We consider that a *correct* approximating subdivision surface should respect/approximate the main feature lines of the object, and thus has to contain at least the number of control vertices necessary to approximate these feature lines. Hence our process is the following: we first approximate the feature line network with subdivision curves and then edges and facets of the base control mesh are created by linking only their *feature control points*. With this process we create a near optimal control mesh since only necessary vertices are created, moreover, we link them with respect to the lines of curvature of the object.

5.1. Feature line approximation

To create the set of *feature control points*, we approximate the network of feature lines with subdivision curves. Each smooth line between two anchor points is approximated separately (see Fig. 6), then we obtain a control polygon network.

We use the approximation algorithm from Lavoué et al. [28]: given a smooth polyline and a maximum error value, this algorithm creates an approximating subdivision curve with a minimal number of control points.

5.2. Local control mesh construction

At this point, we process each region separately: for each segmented region (see Fig. 7, top left), the closed cycle B of control polygons corresponding to its boundary is extracted from the network (see plain lines in Fig. 7, bottom left). Our task is to form a local control mesh P whose boundary is exactly B and without any additional control points (see Fig. 7, bottom right), thus we construct P by creating control edges (and thus facets) linking vertices of B (see dotted lines in Fig. 7, bottom left).

Each region has been extracted due to planarity or anisotropy similarity criteria, thus there exist basically two classes of regions: planar or parabolic (i.e. anisotropic). For the planar case our objective is to create facets with correct proportions; thus we chose control edges associated with the smallest lengths similarly to the lofting algorithms from Nasri [20,21]. For the parabolic case we create edges with respect to the anisotropy and therefore edges coherent with the minimum curvature directions of the region (see Fig. 7, top right). In order to take into account these two cases, a score S is processed for each potential control edge E.

Edge score definition: The mechanism is illustrated by Fig. 8: for each potential edge E, we consider its vertices P_i, P_j and the projections $\tilde{P}_i$, $\tilde{P}_j$ of their respective limit positions on the patch boundary. Then the pseudo geodesic path between these limit

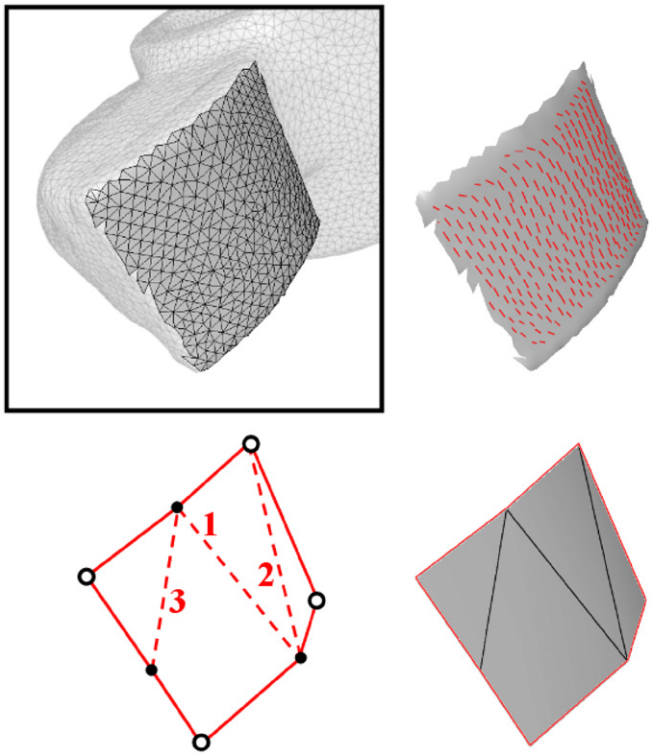

Fig. 7. Local control mesh construction. *Top row*: A region from the RockerArm object and its directions of minimum curvature. *Bottom left*: The closed cycle of control polygons corresponding to its boundaries and the created edges (dotted segments). *Bottom right*: The corresponding local control mesh.

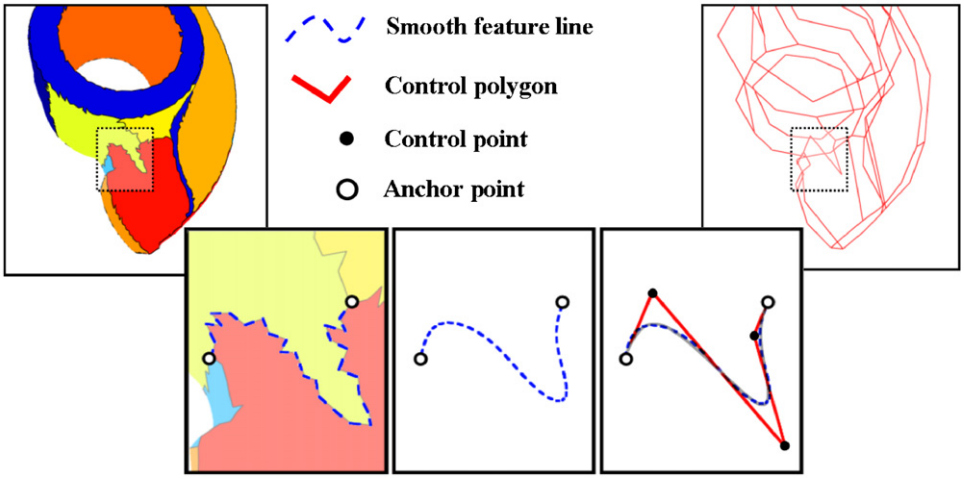

Fig. 6. Construction of the control polygon network. *From left to right*: results of the segmentation, the jagged boundary between two anchor points, result after smoothing, subdivision curve approximation (five control points), the complete control polygon network.

positions is calculated by applying the Dijkstra algorithm on the vertices of the target region (see Fig. 8, right); this greedy algorithm [33] computes the shortest path between two points of a graph. Finally, the curvature tensors of the n vertices V_i of this path are extracted, and particularly the minimum curvature directions. The score $S(E)$ is then defined as follows:

$$S(E) = \frac{\sum_{i=1}^{n} \theta min_i}{n} \times \sum_{i=1}^{n-1} \|V_{i+1} - V_i\| \quad (2)$$

with θmin_i the angle between the minimum curvature direction of the vertex V_i and the segment $\tilde{P}_i\tilde{P}_j$. The first term favours edges coherent with minimum curvature lines and the second term prevents incoherent edges while favouring short ones.

Connectivity construction: Once scores S have been associated to every potential edges between all the pairs of boundary control points from B, the algorithm constructs the potential edge associated with the smallest score (dotted segment associated with number 1 in Fig. 7), and the contour is cut along this edge, creating two sub-contours. This algorithm is repeated recursively on sub-contours until it remains only plane contours (the corresponding control points all lie in the same plane). The planarity is determined by a threshold on the dot products of the contour segments with its average normal. Then for each plane contour, we check its convexity, if it is convex, we create a facet, and if not, we decompose it into convex parts, using the algorithm from Hertel and Mehlhorn [34]. Our algorithm is also applicable to regions with holes and thus associated with several cycles of control polygons. The solution for such cases was proposed and detailed in our previous algorithm [1]: a single oriented contour including every boundary control polygons is constructed, linking them by creating edges and doubling some control points.

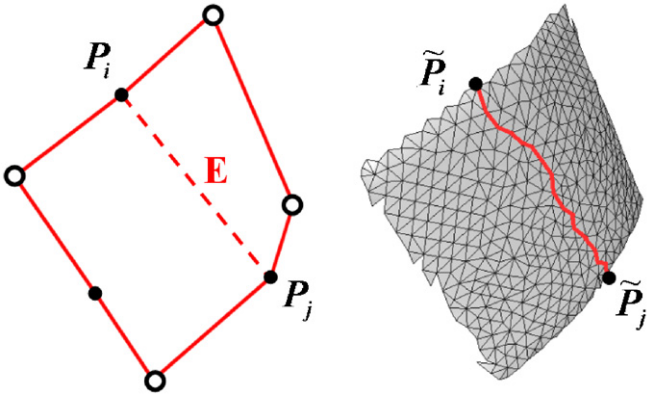

Fig. 8. Mechanism for edge score definition.

5.3. Local control mesh assembly

Once local control meshes corresponding to every regions have been constructed, they are glued together to form a global control mesh. Boundary edges are tagged as *sharp*, to insure that boundary constrains are respected between patches. Moreover, this insures that boundary edges match the feature lines after subdivision.

The obtained subdivision surface is piecewise smooth and gives a quite good approximation of the object (see Fig. 9), without any global optimization process. At this point the main drawback is that we introduce sharp edges in the resulting subdivision surface (at the boundaries between patches) which can produce unpleasant discontinuities for smooth objects. However, such piecewise smooth reconstruction of scanned mechanical parts can be required for CAD applications.

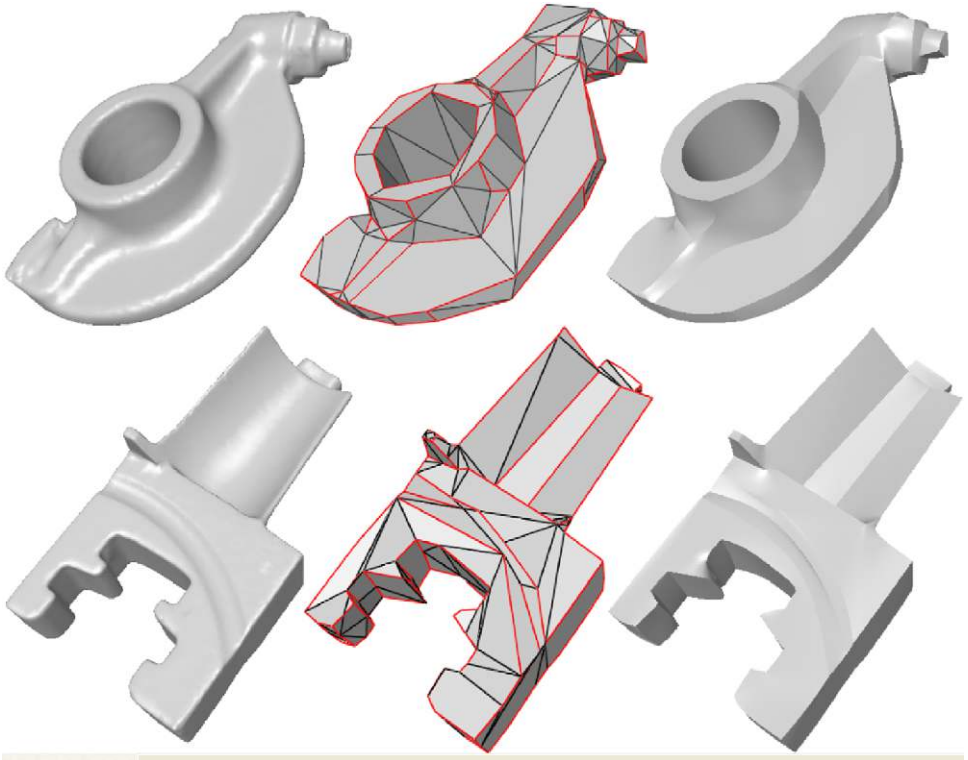

Fig. 9. Sharp control mesh examples. *Top row*: The RockerArm model (15K vertices), the associated control mesh (160 vertices) with sharp edges in red and the limit surface. *Bottom row*: The Blade model (15K vertices), the associated control mesh (187 vertices) and the limit surface.

We have to notice that this framework focuses on obtaining the coarsest possible control mesh and thus our objective is not to obtain nice shaped quadrangles or regular connectivity (in terms of vertex valence). Hence the created control mesh can possibly have high degree facets or high valence vertices. However, in our experiments, this has not perturbed the results nor induced visible artefact.

6. Semi-sharp control mesh construction

The first part of our algorithm (see previous section) produces a piecewise smooth approximating subdivision surface, associated with a quite coarse control polyhedron. We now have to optimize this surface: firstly the sharpness of control edges is relaxed in order to reproduce the curved aspect of the target object and then a global geometry optimization displaces the control points to minimize an asymmetric quadratic error.

6.1. Sharpness relaxation

Since most of the natural 3D objects, especially scanned datasets, are rather smooth, the sharp creases introduced by the first step of our process have to be relaxed. In order to reproduce more precisely the shape aspect of the target mesh while keeping the same control point number, we associate to each sharp edge, an integer sharpness degree from 0 (smooth) to 3 (sharp) according to the rules introduced by DeRose et al. [26] (see Section 2.4). We have considered an integer sharpness degree instead of a real value by reason of simplicity, indeed this mechanism has thus easily been integrated to the further geometric optimization (see next section). Moreover, the maximum sharpness degree has been limited to 3 in order to speed-up the process, knowing that this value is sufficient to represent a visually sharp feature.

For each *infinite* sharp edge, introduced by the previous process, we determine the appropriate degree so as to reproduce the curvature radius of the corresponding fillet or blend on the target mesh. Fig. 10 illustrates this process: In order to automatically determine the sharpness degree $Sh(E_i)$ of the control edge E_i, we associate this edge with different values from 0 to 3. We then compare the curvature radii of the different resulting surfaces (the three pictures at bottom right) with the target object (the picture at bottom left). Finally, we choose the degree that produces the most similar curvature radius ($Sh(E_i) = 2$ for the example).

Practically, starting from the control polyhedron P_∞ containing infinite sharp edges, we create four copies P_0, P_1, P_2 and P_3 where sharp edges are all associated with a degree, respectively, equal to 0,1, 2 and 3. These control meshes are then subdivided (three iterations) to produce dense meshes P_0^s, P_1^s, P_2^s and P_3^s. For each sharp control edge E_i, we choose the appropriate degree by the following process:

- For each mesh P_j^s, we extract the vertices issued from subdivisions of E_i and we compute the mean $C_j(E_i)$ of their maximum curvature values.
- E_i is associated with a boundary between two regions issued from the segmentation process (see Section 4.1); thus we extract from the original mesh the vertices from this boundary

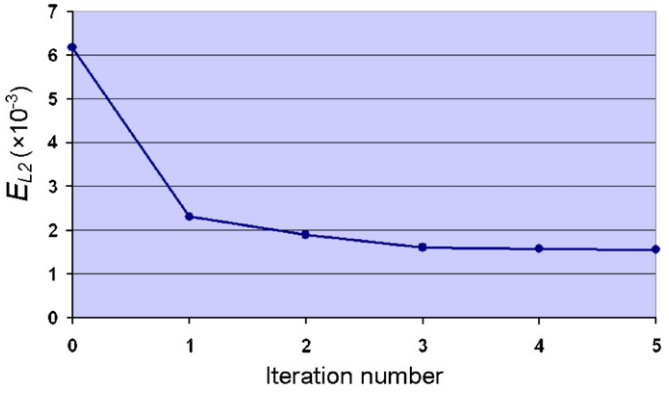

Fig. 11. Evolution of the root mean square error E_{L2}, along with the number of optimization iterations, for the RockerArm approximating surface.

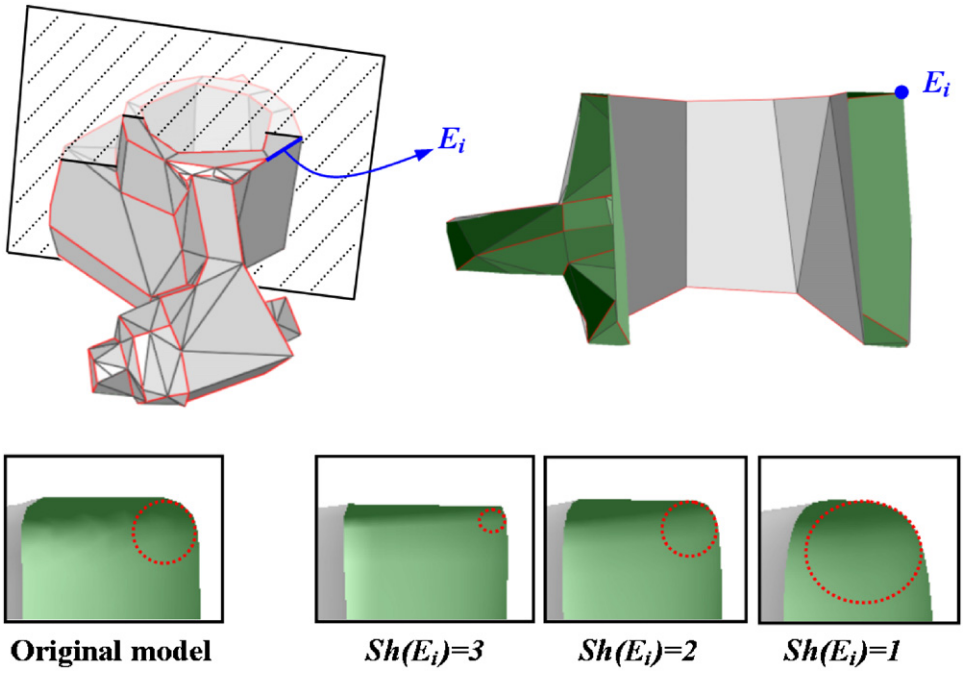

Fig. 10. Mechanism for sharpness degree determination.

and compute the mean $C_{Orig}(E_i)$ of their maximum curvature values.

- We choose the sharpness degree j such as $C_j(E_i)$ is the most similar to $C_{Orig}(E_i)$.

$$Sh(E_i) = argmin_j(|C_j(E_i) - C_{Orig}(E_i)|) \quad (3)$$

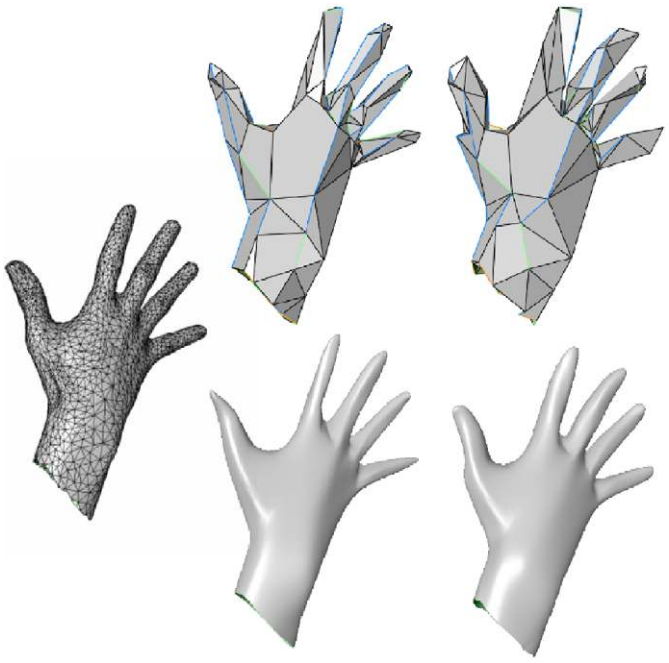

Fig. 12. *Left column*: The *Hand* model. *Middle column*: Control mesh before geometry optimization (top) and the limit surface (bottom). *Right column*: The output control mesh after geometry optimization (three iterations) and the limit surface.

6.2. Geometry optimization

Relaxing the sharpness of the control edges induces a shrinking of the limit surface, and thus the control points have to be moved in order to match correctly the target object. For this task, a global geometry optimization is conducted, which relocates iteratively the control points by minimizing a sum of quadratic distances to the target surface.

(1) Several sample points S_k are chosen on the subdivision surface, they correspond to vertices of the subdivided polyhedron at a finer level l_0. The associated footpoints (projections of the sample points on the target surface) are extracted. Sample points S_k can be computed as linear combinations of the initial control points P_i^0 (see the subdivision rules presented in Section 2.4 and Fig. 2); they correspond to control points $P_i^{l_0}$ at the finer level l_0.

$$S_k = C_k(P_1^0, P_2^0, \ldots, P_n^0) \quad (4)$$

(2) The functions C_k are determined using iterative multiplications of the subdivision matrices associated with our subdivision rules including semi-sharpness processing (see Section 2.4 and Fig. 2 for the *sharp* subdivision rules).

(3) For all S_k, we express the squared distance F_d^k to the target surface using the quadratic distance approximants defined by Pottmann and Leopoldseder [35]. The minimization of their sum F gives the new positions of the control points P_i^0.

$$F = \sum_k F_d^k(S_k) = \sum_k F_d^k(C_k(P_1^0, P_2^0, \ldots, P_n^0)) \quad (5)$$

The minimization of this quadratic function leads to the solution of a linear squared system.

The point to surface quadratic distance approximants from Pottmann and Leopoldseder [35] (recently used for subdivision surface fitting by Marinov and Kobbelt [4] and Cheng et al. [7])

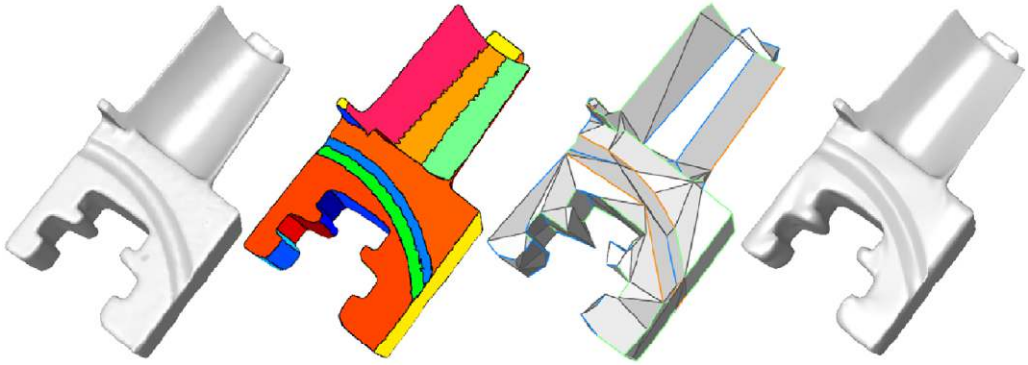

Fig. 13. Segmentation, output control mesh and limit surface for the *Blade* model.

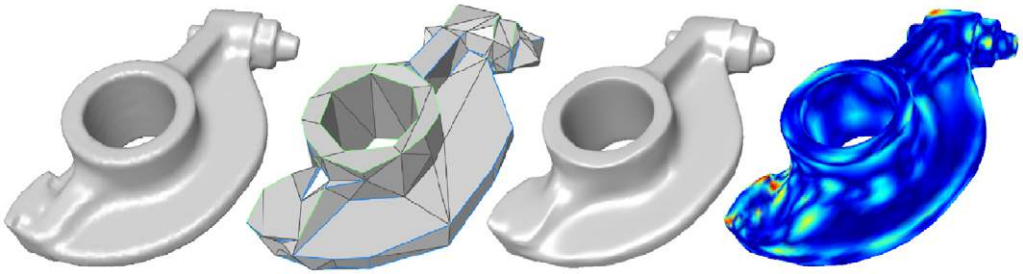

Fig. 14. The *RockerArm* model, output control mesh, limit surface and its distance map to the original shape.

leads to a much faster convergence than traditional point to point distance (used by Ma et al. [3], for instance).

Steps (1)–(3) are repeated for three iterations since it has proven to be sufficient to obtain good visual results and a good trade-off between processing time and quality. Fig. 11 illustrates, for the RockerArm model, the evolution of the root mean square error E_{L2} of the approximating surface, along with the number of iterations; the convergence is very fast and the precision gain is marginal after three iterations. Depending on the case, the convergence may be not completely reached, but our objective is not to focus on a very punctilious approximation. Concerning the choice of the number of sample points S_k, we have chosen $l_0 = 2$ refinements for all examples in this article. As for each refinement, the number of vertices increases by a factor of at least four, the number of equations is about sixteen times the number of unknowns. That ensures a stable solution when solving Eq. (5) in the least squares sense. Fig. 12 illustrates a result of the optimization algorithm (three iterations).

7. Experiments and results

We have tested our algorithm on several models from different natures: scanned mechanical models (see Figs. 13 and 14), scanned organic data (see Fig. 12) and hand-designed model with sharp edges (see Fig. 15). Table 1 presents some statistics about our algorithm.

Properties of the approximating subdivision surfaces: The algorithm provides extremely coarse subdivision control meshes (less than 200 vertices for the presented examples) even for complex shapes. Moreover, corresponding limit surfaces present a very satisfying visual similarity with original objects. Thank to the semi-sharpness optimization, limit surfaces well reproduce the curved aspect of the original shapes. This is particularly visible on the scanned mechanical pieces which present rolling ball blends with a large variety of curvature radii.

The approximation errors: Mean (E_{L1}) and root mean squared (E_{L2}) approximation errors are quite small, while maximal errors ($E_{L\infty}$) are larger. This is due to our approximation mechanism: since we want to obtain very coarse control meshes, it sometimes lacks degrees of freedom to well approximate some parts of the object, particularly tiny details. The distance map from the limit surface to the original shape presented in Fig. 14 well illustrates this repartition of the error in some localized areas (red parts). A solution to this drawback could be to conduct a local enrichment of the mesh in such high error parts, or to consider a finer segmentation. However, our objective is not to focus on approximation error but rather to preserve the main features of the object.

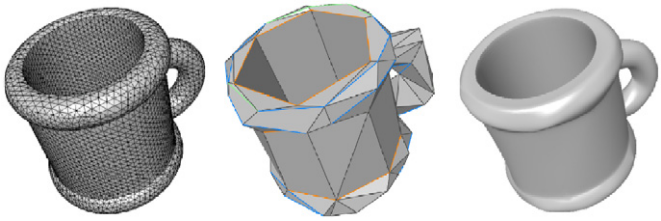

Fig. 15. The *Cup* model, output control mesh and limit surface.

Processing time: Processing times are illustrated in the last three columns of Table 1. The whole first part of the algorithm (feature line extraction and sharp control mesh construction, see Sections 4 and 5) takes about 30 s for a mesh with 30K faces (on a 2 GHz XEON bi-processor). The sharpness relaxation is also quite fast (less than 10 s for the examples); however, the geometry optimization (see Section 6.2) can take several minutes, particularly because of the multiplications of the large subdivision matrices. We could consider a local process to optimize the positions of vertices after the sharpness relaxation, in particular some rules could be found that calculate the shrinking due to the sharpness change.

Applications: Applications of our fitting process are numerous: even if the approximation is not highly precise, such coarse control meshes may represent a good start point for reverse engineering of scanned mechanical parts. In particular such control polyhedron may facilitate B-Spline surface retrieval; indeed several methods build a network of B-Spline patches starting from a subdivision control polyhedron [36,37]. The semi-sharp subdivision representation is particularly used for character animation, in the context of 3D movies [26]. Our algorithm can retrieve the semi-sharp control mesh from a scanned humanoid or other organic model. Applying animation parameters to this coarse control mesh is far easier than animating the original dense mesh. Finally, the approximating subdivision surface can be considered for compression, indeed the control mesh is extremely compact in terms of amount of data and leads to a quite satisfying approximation after subdivisions. For instance the encoding of the RockerArm object with the compression algorithm from Touma and Gotsman [38] (12 bits precision) gives a 35 kbytes binary stream; the encoding of the associated control mesh (see Fig. 14) with 12 bits quantization associated with prediction for geometry, the Facefixer algorithm [39] for connectivity and 2 bits per edge for the sharpness degrees gives about 900 bytes. This kind of lossy high rate compression is particularly adapted for transmission on low bandwidth channel, moreover, the properties of subdivision surface allow to display the object to the desired resolution according to the terminal capacity for instance.

Comparison with other algorithms: We have compared our results with two algorithms: (1) simplification then geometric optimization, a basic scheme followed by many authors [2–4] and (2) the approach from Kanai [5]. Table 2 and Fig. 16 illustrate the results. For both algorithms (1) and (2), we have created subdivision surfaces associated with 200 control points against only 160 for our method; thus the three models are associated with approximatively the same data size (our model contains less vertices but the supplementary sharpness information to encode).

Table 1
Statistics of our subdivision surface fitting algorithm for various 3D models.

	NbReg	F/V Orig	F/V Ctrl	E_{L1}	E_{L2}	$E_{L\infty}$	Ctrl (4–5)	Relax (6.1)	Optim (6.2)
Blade	45	30K/15K	260/187	1.25	2.45	25.4	00:32	00:09	04:36
RockerArm	45	30.2K/15.1K	253/160	1.19	1.60	8.0	00:25	00:08	02:59
Hand	20	5K/2.5K	200/119	2.13	3.71	23.3	00:07	00:06	00:45
Cup	25	11.3K/5.7K	215/123	1.93	2.64	26.1	00:19	00:06	02:05

Number of regions from the segmentation, face/vertex number from original surface and control mesh. Approximation error ($\times 10^{-3}$), objects are normalized in a unit cube. Processing times (min:s) of sharp control mesh construction, sharpness relaxation and geometry optimization.

In terms of geometric approximation errors, our method provides better results than both others, for E_{L1}, E_{L2} and $E_{L\infty}$. For instance, the root mean square error (E_{L2}) is 1.60×10^{-3} for our algorithm against 2.20×10^{-3} for the simplification-based approach and 5.63×10^{-3} for the Kanai algorithm [5]. The Hausdorff distance ($E_{L\infty}$) is also much better with our approximation (8.0×10^{-3} against respectively 14.7×10^{-3} and 31.1×10^{-3}).

Table 2
Face/vertex numbers from control mesh (F/V Ctrl) and approximation error ($\times 10^{-3}$) for different algorithms, for the *RockerArm* model.

	F/V Ctrl	E_{L1}	E_{L2}	$E_{L\infty}$
Simplif-optim	400/200	1.39	2.20	14.7
Kanai [5]	400/200	4.07	5.63	31.1
Our method	253/160	1.19	1.60	8.0

Regarding the visual quality of the approximating shape (see Fig. 16), the subdivision surface associated with our semi-sharp control mesh appears once again much better than others. In particular, the centre cylindrical part is very similar to the original one in comparison with both other algorithms; the extremities of the model are also very nicely approximated and present no visible artifact, contrarily to both other methods. Some tiny details are nevertheless missing at the bottom of the model presented in the second row, by reason of the high coarseness of the control mesh.

Fig. 17 illustrates the control meshes corresponding to the approximation; compared with the other methods, the control edges produced by our algorithm well follow the main lines of the

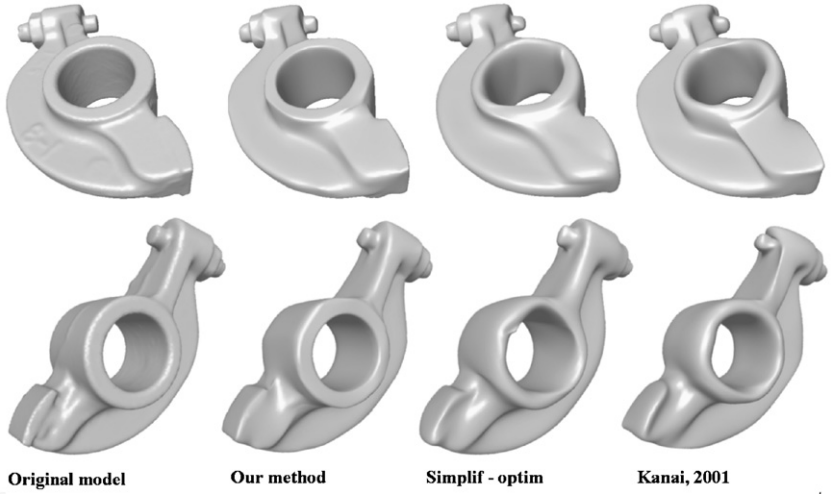

Fig. 16. Comparison of the approximation results of the *RockerArm* object for different algorithms.

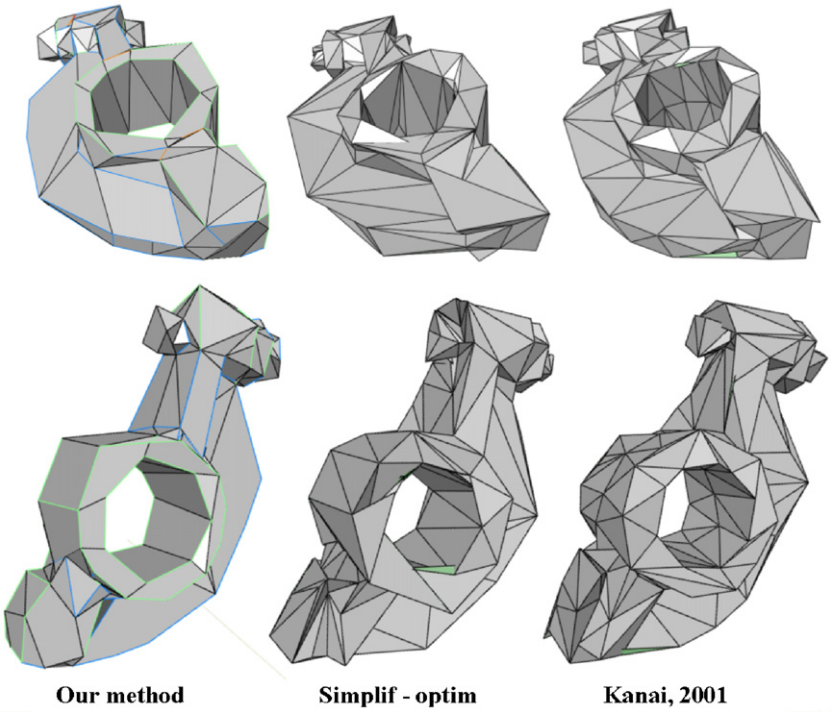

Fig. 17. Comparison of the control meshes corresponding to the approximation of the *RockerArm* object for different algorithms.

shape. Moreover, for both other algorithms, the control meshes present thin and elongated triangles, and also some degenerated cases like faces crossing themselves, or flipping. These phenomena are mainly caused by the geometric optimization steps which can produce unstable results. In our case, the base mesh (before optimization) is constrained by the feature lines of the object and is very close to the final result, hence the geometry optimization does not introduce such artifacts.

8. Conclusion

We have presented an original subdivision surface fitting algorithm based on feature line approximation, anisotropy analysis for connectivity construction and edge sharpness relaxation. These mechanisms yield to a subdivision surface associated with a very coarse control polyhedron and respecting the visual aspect and the relevant features of the object. This approximating surface is quite pertinent regarding to many applications: reverse engineering, animation or compression.

In the case of noised 3D objects we obtain of course a smoothed approximation, indeed our objective is not to represent accurately each detail because the size of the control mesh will blow up. However, it could be interesting to associate this smooth approximation to a multi-resolution bump map or vector field. We also plan to drastically reduce the computing time by suppressing the global geometry optimization; a solution could be to evaluate quantitatively the shrinking (direction and strength) induced by the sharpness relaxation and to displace the vertices accordingly.

References

[1] Lavoué G, Dupont F, Baskurt A. A framework for quad/triangle subdivision surface fitting: application to mechanical objects. Computer Graphics Forum 2007;26(1):1–14.

[2] Lee A, Moreton H, Hoppe H. Displaced subdivision surfaces. In: ACM SIGGRAPH; 2000. p. 85–94.

[3] Ma W, Ma X, Tso S-K, Pan Z. A direct approach for subdivision surface fitting from a dense triangle mesh. Computer Aided Design 2004;36(16):525–36.

[4] Marinov M, Kobbelt L. Optimization methods for scattered data approximation with subdivision surfaces. Journal of Graphical Models 2005;67(5): 452–73.

[5] Kanai T. Meshtoss-converting subdivision surfaces from dense meshes. In: 6th international workshop on vision, modeling and visualization; 2001. p. 325–32.

[6] Marinov M, Kobbelt L. Automatic generation of structure-preserving multiresolution models. Computer Graphics Forum 2005;24(3):479–86.

[7] Cheng K-S-D, Wang W, Qin H, Wong K-Y-K, Yang H-P, Liu Y. Fitting subdivision surfaces to unorganized point data using SDM. In: IEEE Pacific graphics; 2004. p. 16–24.

[8] Susuki H. Subdivision surface fitting to a range of points. In: IEEE Pacific graphics; 1999. p. 158–67.

[9] Li W-C, Ray N, Lévy B. Automatic and interactive mesh to t-spline conversion. In: Eurographics/ACM SIGGRAPH symposium on geometry processing; 2006.

[10] Hoppe H, DeRose T, Duchamp T, Halstead M, Jin H, McDonald J, et al. Piecewise smooth surface reconstruction. In: ACM SIGGRAPH, 28; 1994. p. 295–302.

[11] Litke N, Levin A, Schroder P. Fitting subdivision surfaces. In: IEEE Visualization; 2001. p. 319–24.

[12] Alliez P, Cohen-Steiner D, Devillers O, Lévy B, Desbrun M. Anisotropic polygonal remeshing. ACM Transactions on Graphics 2003;22(3):485–93.

[13] Boier-Martin I, Rushmeier H, Jin J. Parameterization of triangle meshes over quadrilateral domains. In: Eurographics/ACM SIGGRAPH symposium on geometry processing; 2004. p. 193–203.

[14] Tong Y, Alliez P, Cohen-Steiner D, Desbrun M. Designing quadrangulations with discrete harmonic forms. In: Eurographics/ACM SIGGRAPH symposium on geometry processing; 2006. p. 201–10.

[15] Ray N, Li W-C, Lévy B, Sheffer A, Alliez P. Periodic global parameterization. ACM Transactions on Graphics 2006;25(4):1460–1485.

[16] Marinov M, Kobbelt L. A robust two-step procedure for quad-dominant remeshing. Computer Graphics Forum 2006;25(3):537–46.

[17] Dong S, Bremer P-T, Garland M, Pascucci V, Hart JC. Spectral surface quadrangulation. In: ACM SIGGRAPH; 2006. p. 1057–66.

[18] D'Azevedo EF. Are bilinear quadrilaterals better than linear triangles? SIAM Journal On Scientific Computing 2000;22(1):198–217.

[19] Levin A. Interpolating nets of curves by smooth subdivision surfaces. In: ACM SIGGRAPH; 1999. p. 57–64.

[20] Nasri A-H. Interpolating an unlimited number of curves meeting at extraordinary points on subdivision surfaces. Computer Graphics Forum 2003;22(1):87–98.

[21] Nasri A-H, Abbas A, Hasbini I. Skinning Catmull–Clark subdivision surfaces with incompatible cross-sectional curves. In: IEEE Pacific graphics; 2003. p. 102–11.

[22] Schaefer S, Warren J, Zorin D. Lofting curve networks using subdivision surfaces. In: Eurographics/ACM SIGGRAPH symposium on geometry processing; 2004. p. 103–14.

[23] Stam J, Loop C. Quad/triangle subdivision. Computer Graphics Forum 2003;22(1):79–85.

[24] C. Loop, Smooth subdivision surfaces based on triangles. Master's thesis, Utah University; 1987.

[25] Catmull E, Clark J. Recursively generated B-Spline surfaces on arbitrary topological meshes. Computer-Aided Design 1978;10(6):350–5.

[26] DeRose T, Kass M, Truong T. Subdivision surfaces in character animation. In: ACM SIGGRAPH. New York: ACM; 1998. p. 85–94.

[27] Cohen-Steiner D, Alliez P, Desbrun M. Variational shape approximation. In: ACM SIGGRAPH; 2004. p. 905–14.

[28] Lavoué G, Dupont F, Baskurt A. A new subdivision based approach for piecewise smooth approximation of 3d polygonal curves. Pattern Recognition 2005;38(8):1139–51.

[29] Lavoué G, Dupont F, Baskurt A. A new cad mesh segmentation method, based on curvature tensor analysis. Computer-Aided Design 2005;37(10): 975–87.

[30] Ohtake Y, Belyaev A, Seidel H-P. Ridge-valley lines on meshes via implicit surface fitting. In: ACM SIGGRAPH; 2004. p. 609–12.

[31] Hildebrandt K, Polthier K, Wardetzky M. Smooth feature lines on surface meshes. In: Eurographics/ACM SIGGRAPH symposium on geometry processing; 2005. p. 85–90.

[32] Cohen-Steiner D, Morvan J. Restricted Delaunay triangulations and normal cycle. In: 19th annual ACM symposium on computational geometry; 2003.

[33] Dijkstra E-W. A note on two problems in connexion with graphs. In: Numerische Mathematik; 1959. p. 269–71.

[34] Hertel S, Mehlhorn K. Fast triangulation of simple polygons. In: Proceedings of the international FCT-conference on fundamentals of computation theory. Lecture notes in computer science, vol. 158. Berlin: Springer; 1983. p. 207–18.

[35] Pottmann H, Leopoldseder S. A concept for parametric surface fitting which avoids the parametrization problem. Computer Aided Geometric Design 2003;20(6):343–62.

[36] Peters J. Biquartic c1-surface splines over irregular meshes. Computer-Aided Design 1995;27(12):895–903.

[37] Peters J. Patching Catmull–Clark meshes. In: ACM SIGGRAPH; 2000. p. 255–8.

[38] Touma C, Gotsman C. Triangle mesh compression. In: Graphics Interface; 1998. p. 26–34.

[39] Isenburg M, Snoeyink J. Face fixer: compressing polygon meshes with properties. In: ACM SIGGRAPH; 2001. p. 263–70.