Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition 38 (2005) 1139–1151
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/patcog

A new subdivision based approach for piecewise smooth approximation of 3D polygonal curves

Guillaume Lavoue*, Florent Dupont, Atilla Baskurt

LIRIS - UMR 5205 CNRS, 8 boulevard Niels Bohr, 69622 Villeurbanne Cedex, France

Received 8 July 2004

Abstract

This paper presents an algorithm dealing with the data reduction and the approximation of 3D polygonal curves. Our method is able to approximate efficiently a set of straight 3D segments or points with a piecewise smooth subdivision curve, in a near optimal way in terms of control point number. Our algorithm is a generalization for subdivision rules, including sharp vertex processing, of the Active B-Spline Curve developed by Pottmann et al. We have also developed a theoretically demonstrated approach, analysing curvature properties of B-Splines, which computes a near optimal evaluation of the initial number and positions of control points. Moreover, our original Active Footpoint Parameterization method prevents wrong matching problems occurring particularly for self-intersecting curves. Thus, the stability of the algorithm is highly increased. Our method was tested on different sets of curves and gives satisfying results regarding to approximation error, convergence speed and compression rate. This method is in line with a larger 3D CAD object compression scheme by piecewise subdivision surface approximation. The objective is to fit a subdivision surface on a target patch by first fitting its boundary with a subdivision curve whose control polygon will represent the boundary of the surface control polyhedron.

Keywords: Polygonal curve; Subdivision curve; Approximation; Sharp vertex; Compression

1. Introduction

The context of this work is the Semantic-3D project (http://www.semantic-3d.net), supported by the French Research Ministry and the RNRT (Réseau National de Recherche en Télécommunications). The objective is the low bandwidth transmission, in a visualization objective, of CAD objects represented by 3D meshes, with multi-resolution and adaptivity properties. Meshes are optimized in terms of triangle numbers and original NURBS information is not available. In this context, a 3D compression algorithm is needed but the optimized tessellation and the need of a low bandwidth transmission make this problematic very complex. The chosen approach is to convert the original object into a set of light patches represented by subdivision surfaces. This representation will bring a high compression rate adapted to a low bandwidth and to a multi-resolution displaying because of subdivision properties. Moreover, the model will be adaptive because of the prior decomposition into surface patches. In addition, this representation provides a compact and structured description of 3D objects that might be convenient for 3D image processing tasks such as indexing or characterization. This approximation problematic can be linked with the inverse problem for subdivision surfaces. Within this context, we present an efficient algorithm dealing with the inverse problem for subdivision curves, whose purpose is two-fold: firstly it represents a sub-problem of the surface case, and secondly, subdivision curves have the property to represent the boundary of a

* Corresponding author. Tel.: +33 4 72 44 83 95; fax: +33 4 72 43 15 36.
E-mail address: glavoue@liris.cnrs.fr (G. Lavoue).

0031-3203/$30.00
doi:10.1016/j.patcog.2005.02.002

subdivision surface. Thus, dealing first with the boundary of a patch and then with its interior, should be an efficient solution. The visual aspect of the final compressed objects is important and have led us to introduce the notion of sharp features in our algorithms. In this context, our curve approximation algorithm is able to deal with sharp vertices, and to preserve them.

Section 2 details the whole 3D-object compression scheme, whereas subdivision curves are presented in Section 3. Section 4 presents the related work about smooth curve fitting and Sections 5–7 deal with the different parts of our method, the initial curve computation, the optimization scheme and the footpoint determination. Finally, results are presented and discussed in Section 8 and an example of the surface approximation case is presented in Section 9.

2. Presentation of the whole compression process

The whole process can be decomposed into the following parts:

2.1. Decomposition into patches

Firstly the CAD objects are segmented into surface patches. The used method is based on the curvature tensor field analysis and presents two distinct complementary steps: a region-based segmentation which decomposes the object into near constant curvature patches, and a boundary rectification based on curvature tensor directions, which corrects boundaries by suppressing their artifacts or discontinuities. These methods are detailed in Refs. [1] and [2]. Resulting segmented patches, by virtue of their properties (known curvature, clean boundaries) are particularly adapted to subdivision surface fitting (see Fig. 1).

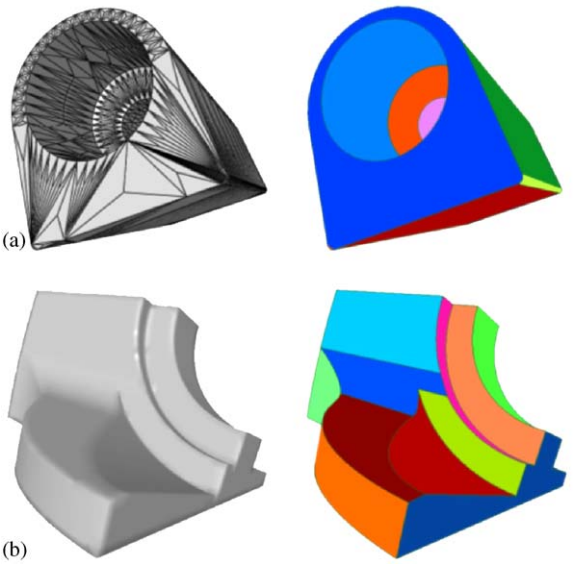

Fig. 1. Segmentation of Swivel (a) and Fandisk (b) objects into patches adapted to subdivision surface fitting.

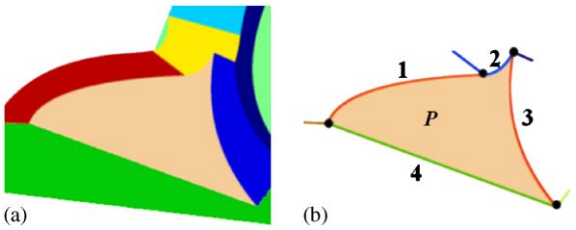

Fig. 2. Extraction of pieces of boundary (1,2,3,4) of a segmented patch (*P*) from the Fandisk object.

2.2. Patch approximation

One of the most relevant problem in the fact of approximating an object by patches is the apparition of cracks because each patch will be approximated by a different surface whose boundary will not be perfectly matched with the others. A solution is to add constraints during the fitting process but the complexity will highly increase. Indeed if each patch has constraints with its neighbors, the algorithm will become a global optimization problem. Another solution is to treat these cracks after the fitting process but the fitted patches will be modified compared with the first approximation. Our solution is simpler and more effective. For each patch the subdivision surface approximation problem is divided into two sub-problems: a piecewise approximation of the patch boundary and the construction of the final subdivision surface by interpolation of the found boundary and approximation of the interior data, like the approach proposed by Schweitzer [3]. In order to prevent our model from cracks, for each patch the boundary is divided into pieces of boundary corresponding to the different adjacencies with its neighboring regions (see Fig. 2). Once each piece of boundary has been approximated by a subdivision curve (this inverse problem is treated in this paper), the corresponding control polygons are put together to form the control polygon of the whole boundary. According to subdivision properties, this control polygon will represent the boundary of the control polyhedron of the subdivision surface representing the corresponding patch. Then, for each patch, a subdivision control mesh is created using its boundary information. The final control mesh defining the whole surface will comprise control meshes of all regions.

3. Subdivision curve presentation

The subject of this paper is the approximation of a polygonal curve with a piecewise smooth subdivision curve. A subdivision curve is created using iterative subdivisions of a

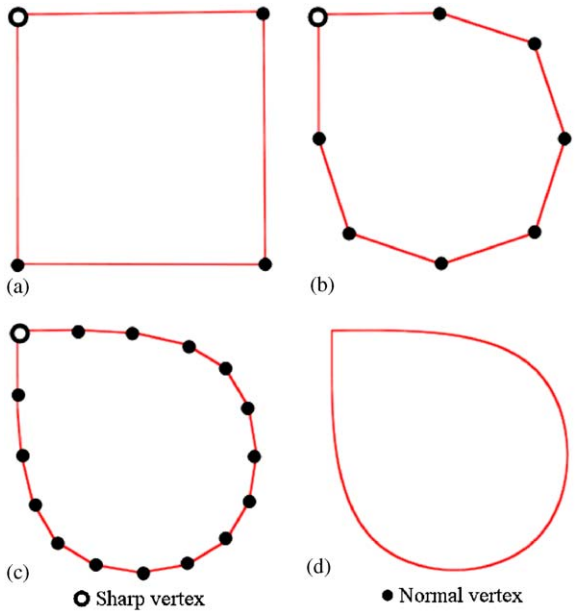

Fig. 3. Example of subdivision curve with one sharp vertex. (a) Control polygon. (b and c) 2 iterations of subdivision, (d) limit curve.

control polygon. In this paper we use the subdivision rules defined for subdivision surface by Hoppe et al. [4] for the particular case of *crease* or boundary edges: new vertices are inserted at the midpoints of the control segments and new positions P_i' for the control points P_i are computed using their old values and those of their two neighbors using the mask [5]:

$$P_i' = \frac{1}{8}(P_{i-1} + 6P_i + P_{i+1}). \quad (1)$$

With these rules, the subdivision curve corresponds to a uniform cubic B-spline, except for its end segments. We also consider specific rules (those defined by Hoppe [4] for *corner* vertices) to handle sharp parts and extremities:

$$P_i' = P_i. \quad (2)$$

This subdivision curve will coincide with the boundary generated by commonly used subdivision surface rules like Catmull–Clark [6] or Loop [7]. An example of subdivision curve is presented in Fig. 3.

4. Related work and framework

This inverse problem for subdivision curve ties up with the smooth parametric curve approximation problematic, with the additional difficulty that a subdivision curve does not have a parametric formulation and cannot be evaluated at any point. But this shortcoming can be solved using different techniques like does Schweitzer [3]. The author considers the subdivision curve as being composed of a sequence of cubic Bezier segments, except at each end. This number of segments increases when subdividing. Thus any point of the curve can be evaluated by founding the Bezier segment (given by repeated subdivisions) which includes the considered parameter, and then applying B-Spline basis functions. Most of the smooth curve approximation methods are based on a data parameterization. Let V_i being the sequence of p points to approximate, and S, our B-spline or subdivision curve, the usual approaches compute the control points that minimize an error function F:

$$F = \sum_{i=1}^{p} \| S(\xi_i) - V_i \|^2 \quad (3)$$

with ξ_i the parameter value assigned to the data point V_i. The minimization of this over determined quadratic system is generally solved by a least square method, which leads to a square linear system. The main problem lies in the choice of the parameters ξ_i which highly influence the result. Three methods are commonly used to assign the approximation parameter locations (see Ref. [8]): Uniform, Chord Length and Centripetal parameterizations. But none of these solutions is optimal and readily adapted. Schweitzer [3], in its subdivision curve approximation algorithm, starts with a Chord Length parameterization and then corrects the parameters by considering at each iteration the parameters of the data point projections on the approximating subdivision curve. Some authors have proposed more sophisticated and efficient iterative parameter correction procedures, notably the intrinsic parameterization of Hoschek [9], which was improved by Saux and Daniel [10]. The algorithm firstly minimizes the system with respect to S, considering initial ξ_i values, and secondly with respect to ξ_i (separately for each parameter value). Another solution was proposed by Speer et al. [11] which considers a global approach, both the control points and the parameters are considered as unknowns. But fundamentally this parameterization remains a problem because the correction procedures are time consuming and take many iterations to converge (> 50) [12].

In this context, Pottmann et al. [13] have proposed a new and very efficient approach inspired by the active contour models of Kass et al. [14]: the Active B-Spline Curve. Their approximation scheme does not require parameterization. The idea is to make an active initial B-Spline converge towards a target curve by minimizing local approximate squared distances from the target curve. It is not a point to point distance minimization but a point to curve minimization which allows a very fast convergence (< 10 iterations) (see Section 8). We have extended this optimization scheme to handle piecewise smooth subdivision curves (see Sections 5 and 6).

Unfortunately this method is very dependent of the initial active curve. Thus, the original method from Pottmann was extended by Yang et al. [12] to permit a dynamic control point insertion or removal. The choice of the initial number

and placement of the control points is a remaining problem in the approximation algorithms. Schweitzer [3] starts with 4 control points and then increases iteratively the number according to the resulting error. Saux and Daniel [10,15] consider initially a high number of control points and then determine the minimum number using a dichotomic method. Finally, Yang et al. [12] heuristically determine this number by considering the local direction monotony of the target curve. But their method leads to a generally too high number of control points, so a removal algorithm is generally needed. Concerning the positions of the initial control points, the initial parameterization for the initial curve determination is usually the Chord Length [12,3] or the Centripetal [10] and thus gives rather poor initial results. However these control point initial placement and number are critical because, better is the initial approximation and faster will be the convergence algorithm. A control point insertion or removal, for instance, is a very time consuming task because it will influence a significant part of the curve and thus require an other cycle of iterations. Within this context, we have developed an efficient determination method for the number and positions of initial control points, based on curvature analysis and theoretical foundations (see Section 5). One other shortcoming of the Active B-Spline Curve from Pottman et al. was raised in Ref. [12], and concerns the wrong matching problem. The distance function used for the optimization has often multiple local minima and some of these minima can lead to a wrong matching of the target curve. This problem occurs especially for highly concave or self-intersecting target curves, or when the initial active curve is too different from the target. Our algorithm will increase the stability and prevent from these problems, by introducing an active pseudo-parameterization of the target data, called Active Footpoint Parameterization (see Section 7).

5. Initial control point processing

5.1. Sharp vertex detection

Correct number and positions of the control points of the initial active curve are critical for our convergence algorithm. Our initial control point placement method is based on the curvature analysis of the target curve. Thus prior to starting the algorithm we must detect and take into account sharp vertices. Indeed even if, in practice, a curvature value is associated with sharp vertices, the curvature is not theoretically defined on these features. That is why we process a sharp vertex detection by fixing a threshold on the angle of the two segments sharing each vertex. Each sharp vertex will represent a sharp control point in the final control polygon representing the final approximating subdivision curve. Thus we cut the target curve at each sharp vertex and apply our algorithm on each so extracted piece of curve. Finally, correspondingly found control polygons are put together, with associated sharp control points tagged as *sharp*, to form the final subdivision curve control polygon.

5.2. Theoretical foundations

Our subdivision curve (once cutted at its sharp vertices) represents a uniform cubic B-Spline curve except for its end segments (see Section 3), therefore except at its ends, the curve is composed with polynomial curve segments S_i. We have studied the behavior of the curvature on such a segment, in order to make the connection between the optimal number of control points and the curvature of the target curve.

Theorem 1. *Considering a uniform cubic B-Spline segment, local curvature maxima are necessarily located at the extremities.*

Proof. Each uniform cubic B-Spline segment S_i is associated with 4 control points $(P_{i-3}, P_{i-2}, P_{i-1}, P_i)$. For any parameter value u such as $0 \leqslant u \leqslant 1$, the corresponding position $S_i(u)$ on this segment is defined by:

$$S_i(u) = \tfrac{1}{6}((1-u)^3 P_{i-3} + (3u^3 - 6u^2 + 4) P_{i-2} + (-3u^3 + 3u^2 + 3u + 1) P_{i-1} + u^3 P_i). \quad (4)$$

Thus the second derivative vector is:

$$\ddot{S}_i(u) = (1-u) P_{i-3} + (3u-2) P_{i-2} + (-3u+1) P_{i-1} + u P_i. \quad (5)$$

The curvature $C(u)$, at each parameter, is defined by $C = \|\ddot{S}_i(u)\|$. Our goal is to study the variation of this curvature, thus we have calculated its first derivative which can be expressed as:

$$\dot{C}(u) = \frac{f(u)}{\sqrt{g(u)}} \quad (6)$$

with f linear in u and g quadratic in u. Thus the equation $\dot{C}(u) = 0$ has one or zero solution $u_0 \in [0, 1]$, therefore the curvature is either monotonic, or has one extremum over the cubic segment (not located at an extremity).

- If the curve segment is monotonic there exists one curvature maximum at one extremity ($u = 0$ or $u = 1$), therefore the theorem is verified.
- If the curve has one extremum, we will determine if it is a maximum or a minimum. For this purpose, we have determined the sign of the second derivative $\ddot{C}(u)$ of the curvature. We found numerically that:

$$\forall u \in [0, 1], \quad \ddot{C}(u) \geqslant 0 \quad (7)$$

thus $\dot{C}(u)$ is increasing $\forall u \in [0, 1]$, and therefore $\dot{C}(u) \leqslant 0$ for $u \in [0, u_0]$ and $\dot{C}(u) \geqslant 0$ for $u \in [u_0, 1]$. As a result $C(u)$ is decreasing before u_0 and increasing after and therefore $C(u_0)$ represents a local minimum.

Thus curvature maxima are located at the extremities, therefore the theorem is verified. □

Formal and limit calculation have been made using the software Waterloo Maple©.

According to Theorem 1, a local maximum of curvature located over the target curve is associated with the extremity of a B-Spline segment and therefore there is necessary at least one associated control point whose limit position is at the extremum. So, for n local curvature maxima we can affirm that at least n initial control points are needed.

5.3. Algorithm

Our initial curve processing algorithm is the following:

(1) The curvature C is calculated for each vertex V_i of the target curve. The curvature is defined as the opposite of the radius of the circle circumscribed around the triangle defined by V_{i-1}, V_i and V_{i+1}.

$$C(V_i) = \frac{\sin(\varphi_i)}{\frac{1}{2}d_i}, \tag{8}$$

where φ_i is the angle between $\overrightarrow{V_{i-1}V_i}$ and $\overrightarrow{V_iV_{i+1}}$ and d_i is the norm $\| \overrightarrow{V_{i-1}V_{i+1}} \|$.

(2) The curvature is smoothed by a Median Filter [16] (a common image denoising filter), associated with a 1-neighbourhood. Then, after a quantification, the n local maxima are extracted by the *Top Hat* algorithm, a morphological filter described by Meyer in Ref. [17].

(3) The number of control points is initialized to n, increased by 2 for the extremities if the curve is open.

(4) The placement of the n control points is determined with a linear $n \times n$ system. Indeed, for a subdivision curve, the limit position P_i' of a control point P_i can be processed according to its neighbors:

$$P_i' = \frac{1}{6}(P_{i-1} + 4P_i + P_{i+1}). \tag{9}$$

Since we know that these limit positions must coincide with the local curvature maxima, we obtain the linear $n \times n$ system.

Fig. 4 shows this initialization process: the variation of the curvature (a) with the determination of the 5 local maxima, and the corresponding initial subdivision curve (b) processed using the positions of these maxima. We can observe that this initialization curve is very satisfying considering the target curve, thus the number of convergence iterations will be considerably reduced. Moreover, we have asserted that this number of control points is minimum and therefore no control point removal will be needed for the further optimization algorithm.

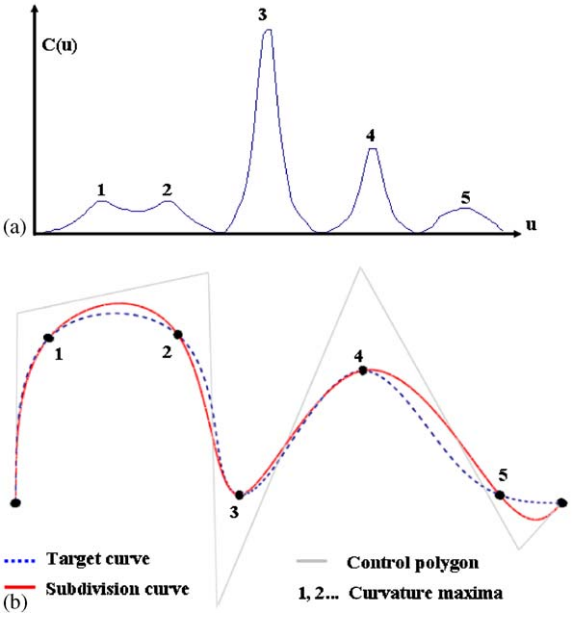

Fig. 4. Example of initial control point processing. (a) Curvature variation over the target curve, (b) corresponding maxima and initial subdivision curve with the associated control polygon.

There exist some special cases, in which our rule for the initialization is modified:

- If the curvature is constant and not null or monotonic, thus we consider 4 initial control points (2 and the extremities for open curves), with their limit positions uniformly distributed over the target curve.
- If the curvature is null on the whole curve, we consider a straight line linking up the extremities (this case is not possible for closed curve).

Of course, this method is not suited for target curves of which curvature does not fully characterize the shape. Common examples are spirals or helixes which have no local maxima of curvature but exhibit a highly curly appearance. For these curves, our method will start with 4 control points, whereas they need much more, so the process will be slowed down by the control point insertion mechanisms. However this kind of curve remains marginal in our application.

6. Optimization scheme

Once the initial subdivision curve has been processed, the optimization algorithm fits this curve to the target data by displacing iteratively the control points P_i. We have extended the method from Pottmann et al. [13] for subdivision rules. This method relies on the distance function of the data curve Ψ, which assigns to each point its shortest

distance to Ψ. In practical terms, not the distance function itself, but the local quadratic approximants of the squared distance function are considered.

6.1. Local quadratic approximants of the squared distance function to 2D and 3D curves

Considering a 2D space Π and a smooth curve Ψ, the Frenet frame (e_1, e_2) at a curve point $\Psi(t)$ is defined as follows: $e_1 = \dot{\Psi}/\|\dot{\Psi}\|$ is the unit tangent vector and e_2 the associated unit normal vector. Considering a point p in Π, and its corresponding footpoint $\Psi(t_0)$ (associated with the shortest distance d of the curve), thus the coordinates of p in the Frenet frame defined in $\Psi(t_0)$ are $(0, d)$. Then the local quadratic approximant $F_d(p)$ of the squared distance of p to the curve Ψ is given by:

$$F_d(x_1, x_2) = \frac{d}{d+\rho} x_1^2 + x_2^2, \tag{10}$$

where x_1 and x_2 are the coordinate of p with respect to the Frenet frame and ρ is the curvature radius at $\Psi(t_0)$. The reader may refer to Ref. [18] for a detailed derivation and proof of this formula. In the case of a 3D space, considering p and the associated footpoint $\Psi(t_0)$, a cartesian coordinate system (e_1, e_2, e_3) is defined such as: $e_1 = \dot{\Psi}/\|\dot{\Psi}\|$ is the unit tangent vector, e_3 is in the direction of $p - \Psi(t_0)$ and $e_2 = e_3 \wedge e_1$. In this frame, the local quadratic approximant is given by [13]:

$$F_d(x_1, x_2, x_3) = \frac{d}{d+\rho} x_1^2 + x_2^2 + x_3^2. \tag{11}$$

In our case, the target curve is sampled and therefore not continuous like Ψ. However Pottmann et al. [13] have shown that their estimator is still valid in this case, considering discrete values for ρ as for the Frenet Frame.

6.2. Optimization algorithm for subdivision rules

The optimization process is the following, for each iteration:

- Several sample points S_k are chosen on the subdivision curve, and the associated footpoints O_k are calculated on the target curve. In our case, sample points are the vertices of the subdivision curve at a finer level l_0, after application of several steps of subdivision. Sample points S_k can be computed as linear combinations of the control points P_i (see Section 3):

$$S_k = C_k(P_1, P_2, \ldots, P_n). \tag{12}$$

  The functionals C_k are determined using iterative multiplications of the subdivision matrices associated with our subdivision rules. For n control points P_i, the $n \times 2n$ subdivision matrix M_1 which gives the $2n$ vertices S_j after one step of subdivision has the form (see Eq. (1)):

$$M_1 = \frac{1}{8} \begin{pmatrix} . & . & . & . & . & . \\ . & 4 & 4 & 0 & 0 & . \\ . & 1 & 6 & 1 & 0 & . \\ . & 0 & 4 & 4 & 0 & . \\ . & 0 & 1 & 6 & 1 & . \\ . & 0 & 0 & 4 & 4 & . \\ . & . & . & . & . & . \end{pmatrix}. \tag{13}$$

  Thus the functionals C_k, for the level l_0, are the lines of the matrix C such as:

$$C = \prod_{l=1}^{l_0} M_l \times L_{l_0}. \tag{14}$$

  L_{l_0} is the limit matrix which give the limit positions of the considered control points at the level l_0 (see Eq. (9)).
- For each S_k the local quadratic approximant F_d^k of the squared distance function of S_k to the target curve, is computed according to the Frenet frame at O_k.
- New positions of control points are processed by minimizing the sum of the local quadratic approximants:

$$F = \sum_k F_d^k(S_k) = \sum_k F_d^k(C_k(P_1, P_2, \ldots, P_n)). \tag{15}$$

  The minimization of this quadratic function in the new position of the control points, leads to the resolution of a linear squared system.

These iterations are repeated until the approximation error or the variation of the approximation error is lower than a given threshold. The convergence of the algorithm is very fast. Fig. 5 presents three iterations of the algorithm: Fig. 5a shows the initial position of the subdivision curve with a sample point S_k and the corresponding footpoint O_k, whereas Figs. 5b and c present the new positions of the control points after, respectively, 1 and 2 iterations of the optimization algorithm. At the second iteration the target curve is perfectly fitted.

7. Footpoint determination

7.1. The wrong matching problem

The footpoint determination algorithm used in Refs. [13] and [12], consists, for each sample point, in considering the smallest distance point on the target curve. Ref. [12] precomputes the discrete distance field using the Fast Marching Method in order to increase the speed, but the result is the same. Considering this method, a problem will occur for a self intersecting target curve or when a part of this target curve is very close to another part: sample points will be associated with incorrect footpoints belonging to wrong parts of the curve. This wrong matching was also observed by Yang et al. [12] who found no general solution.

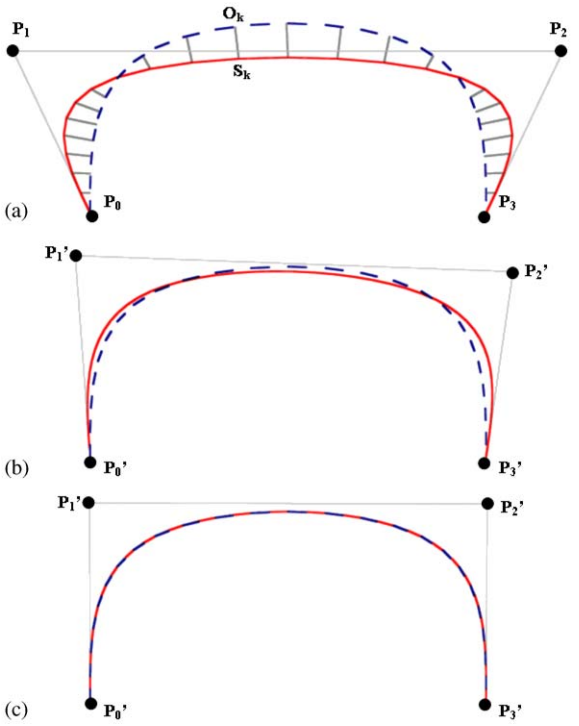

Fig. 5. Example of the optimization procedure. (a) Initial subdivision curve, (b and c) resulting curve after, respectively, 1 and 2 optimization iterations.

As a consequence, either the convergence of the algorithm will slow down because a higher number of iterations is required, or the convergence will become impossible. This problem is illustrated in Fig. 6. We have developed an efficient solution for this problem, based on the determination of Generalized Footpoints (GF) and their Active Parameterization.

7.2. Generalized footpoint computation

The footpoint O_k corresponding to a point S_k is defined as the point belonging to the target curve and associated with the shortest distance from S_k. Since the target curve is usually defined by a highly sampled polygonal curve, the footpoint determination consists generally in computing point to segment distances and considering the projected point associated with the shortest distance. In the continuous case, considering a smooth curve Ψ (at least C^1), a footpoint O_k is necessarily issued from an orthogonal projection of S_k onto the curve. We introduce GF as the set of points O_k^g issued from an orthogonal projection of S_k. In our polygonal case, GF are more complex to determine since the normals are not continuous but piecewise constant. The determination process is the following: each point T_i of the target curve is associated with a parameter t_i (Chord Length Parameterization), such as:

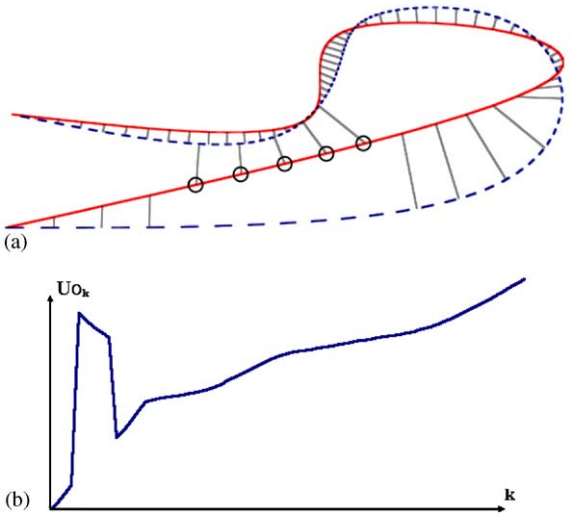

Fig. 6. Illustration of the bad footpoint placement problematic. (a) Subdivision curve with misplaced footpoints (corresponding to the surrounded sample points), (b) footpoint parameter distribution, with an evident discontinuity corresponding to the misplaced ones.

$$t_1 = 0 \quad \text{and} \quad t_n = 1,$$
$$t_{i+1} = \frac{\|T_{i+1} - T_i\|}{\sum_{i=1}^{n-1} \|T_{i+1} - T_i\|}. \tag{16}$$

For each point T_i of the target curve, we consider the two incident segments Seg_1 and Seg_2 and the associated projections p_k^1 and p_k^2 of the considered S_k on lines carrying Seg_1 and Seg_2. Different cases are considered:

- $p_k^1 \in Seg_1$ (resp. $p_k^2 \in Seg_2$) thus p_k^1 (resp. p_k^2) is considered as a GF (see Fig. 7a).
- $p_k^1 \notin Seg_1$ and $p_k^2 \notin Seg_2$, and they are not on the same side of their respective segments, thus T is considered as a GF (see Fig. 7b).
- $p_k^1 \notin Seg_1$ and $p_k^2 \notin Seg_2$, and they are on the same side of their respective segments, there is no associated GF (see Fig. 7c).

Fig. 8 shows an example of the determined GF O_k^g for a sample point S_k. Only one of them corresponds to the correct one within our optimization procedure, this choice is detailed in the next subsection.

7.3. Active footpoint parameterization

Parameter values U_{O_k} are associated with footpoints O_k, they are computed by a linear interpolation between the parameters t_i of the target curve points which surround the considered footpoints. However these parameters are not

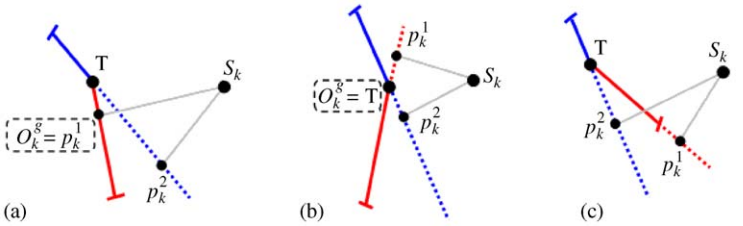

Fig. 7. Generalized Footpoints (*GF*) determination mechanism for a piece of target curve consisting of two segments, (a and b) determination of a *GF*, (c) no *GF*.

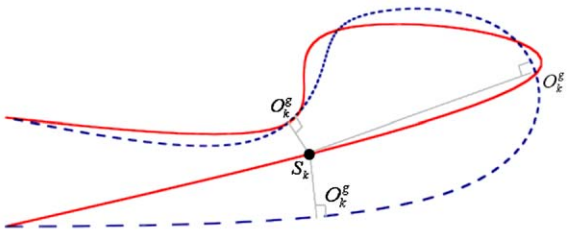

Fig. 8. Example of the set of Generalized Footpoints O_k^g for a sample point S_k.

linked with the parametric correspondence introduced by traditional fitting methods [9,10], they just aim to select the appropriate footpoints. We have studied the variation of these footpoint parameter values U_{O_k} for sample points S_k along the active curve. We consider the set of footpoints O_k as correct if the U_{O_k} distribution is strictly increasing. Fig. 6a shows misplaced footpoints, the corresponding U_{O_k} distribution is presented in Fig. 6b. An evident discontinuity appears in the distribution which notifies the wrong matching problem. Our goal is to find a footpoint distribution which gives a strictly increasing U_{O_k} distribution. Our algorithm, so called Active Footpoint Parameterization is the following:

- We compute GF (see Section 7.2), for each sample point of the subdivision curve.
- For each sample point S_k, we consider among its GF O_k^g, the smallest increasing one:

$$O_k = \underset{O_k^g}{\operatorname{argmin}}(\|O_k^g - O_{k-1}\|), \quad (O_k^g - O_{k-1}) > 0. \tag{17}$$

- If the found O_k is too high compared with O_{k-1}, thus it is considered incorrect and eliminated, the corresponding S_k is not considered in the optimization process for the current iteration, because no coherent footpoints have been found. It is the same if no O_k can be found.

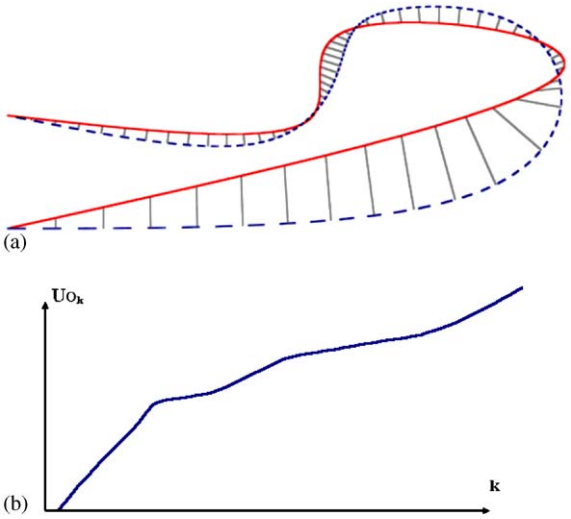

Fig. 9. Illustration of Active Footpoint Parameterization result for the footpoint determination, (a) subdivision curve with corresponding correct footpoints, (b) footpoint parameter distribution.

With this algorithm, the set of determined footpoint parameters is strictly monotonic and increasing, whatever the curve to treat. Thus wrong matching is eliminated and the optimization procedure is prevented from instability or oscillations. As a consequence Fig. 9a shows the curve presented in Fig. 6a, with correct footpoints determined with our Active Footpoint Parameterization method. Results are now correct, resulting footpoints are coherent and adapted to the optimization procedure. The U_{O_k} distribution, presented in Fig. 9b, is increasing and much more correct that the distribution in Fig. 6b.

8. Complete algorithm and results

The whole subdivision curve approximation algorithm is the following:

- Detection of the sharp vertices and decomposition of the input target curve into smooth parts (see Section 5.1).

- For each smooth part:
  (1) Initialization of the subdivision curve, according to the curvature of the target curve (see Section 5.3).
  (2) Computation of the correct footpoints, using the Active Footpoint Parameterization (see Section 7). The number of sample points on the subdivision curve is chosen by the user, in our examples we consider vertices of the curve subdivided twice.
  (3) Optimization procedure (see Section 6). The subdivision curve is moved toward the target curve, by minimizing a sum of quadratic distances. The approximation error E is computed.
  (4) (2) and (3) are repeated m times until $E < \varepsilon$ or $m < m_0$. ε and m_0 are, respectively, a maximum fixed error and a maximum iteration number.
  (5) If $E < \varepsilon$ then the process is terminated, else a new control point is inserted onto the subdivision curve, where the local error is maximum and the process goes to step (2).
- Smooth parts are put together with associated *sharp* tagged control points.

The control point insertion algorithm is the following: first, the sample point S_k associated with the maximum error is extracted. Since it comes from several steps of subdivision from the initial control polygon, it is easy to locate control points P_i and P_j, of which limit positions S_i and S_j surround S_k. Then the new control point P_k is inserted by carrying forward the ratio $\frac{S_i S_k}{S_i S_j}$:

$$P_k = P_i + \frac{S_i S_k}{S_i S_j} \times \overrightarrow{P_i P_j}\,. \tag{18}$$

We have conducted many tests on different target curves from different natures, in order to demonstrate the efficiency of our method. We present here several examples about the different characteristics of our algorithm. Most of the presented curves are 2D (except for Fig. 14) in order to improve visibility but our algorithm work as well on 3D curves. The average error E is defined by the mean of the distances from each sample point to its corresponding footpoint. All curves considered in the experiments were normalized in a bounding box of length equal to 1.

8.1. Initial control points placement examples

First, we have conducted experiments about the efficiency of the subdivision curve initialization presented in Section 5. For different target curves, we have considered 2 different initial curves, one processed with our algorithm and the other, with the same number of control points with limit positions evenly sampled on the target curve. Results are presented on Fig. 10. On one hand, Fig. 10c presents the initial curve computed with our curvature based method (10 control points) whereas its optimization result which converges after several optimization iterations is in Fig. 10d. On the other hand, the other initial curve, computed by a regular sampling of the same number of control points, and the corresponding convergent result are presented in Figs. 10a and b.

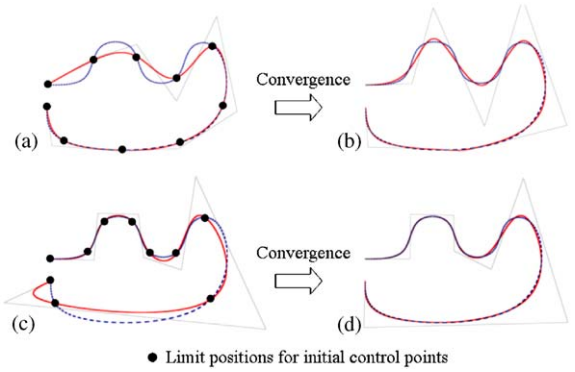

Fig. 10. Optimization results starting from two different initial subdivision curves. (a and b) Initial evenly distributed control points and corresponding final convergent result (after 7 iterations). (c and d) Initial control point processing using curvature and corresponding final convergent result (after 7 iterations).

The resulting curve corresponding to our curvature based initialization is very closed to the target curve (the resulting error is 2.302×10^{-3}) and particularly, is closer than the regular sampling one of which resulting error is 5.168×10^{-3}. These resulting errors appear more clearly in Fig. 11. The evolution of the optimization results is presented for each iteration. The error associated with the curvature based initialization is always lower than the other. Our initialization procedure provides a near optimal set of control points in term of placement, and a number generally sufficient for a good approximation.

Other results for the two target curves of Fig. 14 are presented in Table 1. *Curve14a* corresponds to Figs. 14a and b and *Curve14c* corresponds to Figs. 14c and d. For both target curves, the error is much lower at each iteration for the curvature-based initial curve (CBI). In the case of *Curve14c* the approximation error for the regular sampling initial curve (RSI) will finally reach the value 4.38×10^{-3} ($\approx$ error value for the 3rd iteration of the CBI curve), but only at the 25th iteration, thus our curve initialization has considerably increased the convergence speed. In the case of *Curve14a*, the approximation errors presented at the 6th iteration for both RSI and CBI curves are approximately the convergence values. Thus in this case our curve initialization has permitted a better approximation, even with an infinite number of iterations.

8.2. Active footpoint parameterization examples

We have tested the efficiency of our footpoint determination algorithm (see Section 7). We have conducted experiments on curves with close branches (see Fig. 12) or

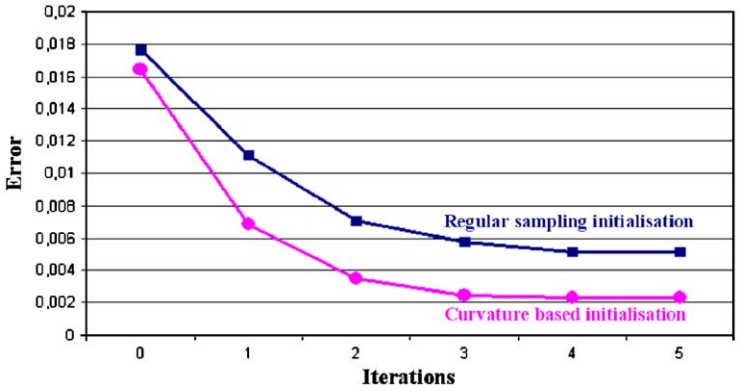

Fig. 11. Optimization result evolution for several iterations, starting from two different initial subdivision curves: Curvature-based initialization (CBI) and regular sampling initialization (RSI).

Table 1
Error ($\times 10^{-3}$) evolution for several iterations, for the target curves presented in Fig. 14 and different initialization methods

Iterations	0	1	3	6
Curve14a, RSI	17.09	8.62	5.09	5.05
Curve14a, CBI	12.35	3.15	1.90	1.85
Curve14c, RSI	24.3	16.22	13.87	12.41
Curve14c, CBI	26.48	8.10	4.38	4.38

self-intersections (see Fig. 13). Fig. 12a presents a target curve with close branches and the corresponding initial curve with classic footpoint determination, wrong matching problems appear and thus the final curve after several iterations converges toward a bad approximation (see Fig. 12b). On the other hand, our footpoint determination algorithm, presented in Fig. 12c leads to a very satisfying approximation (see Fig. 12d), moreover the convergence was obtained very rapidly after only 4 iterations. Concerning the self-intersected curve presented in Fig. 13, results are also very satisfying. Bad convergence results are observed for the classic footpoint determination (see Figs. 13a and b) whereas our method gives a very good approximation (the convergence was obtained after 5 iterations) by carrying out the footpoint matching successfully.

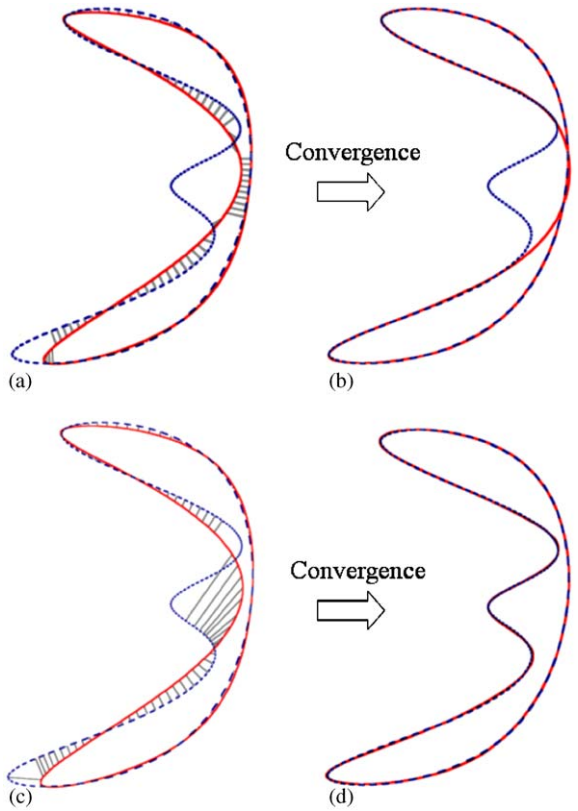

Fig. 12. Effect of the Active Footpoints Parameterization on a curve with close parts. Initial curve with associated footpoints computed by the classical method (a) and by our Active Footpoints Parameterization (c) and results of the optimization process (b and d) (4 iterations).

8.3. Complicated curve examples and compression rate analysis

We have tested our algorithm for *complicated* target curves in order to test the efficiency of our method whose final purpose is to decrease the amount of data in our final compression objective. Examples are presented for a curve with several concavities and self-intersections (Figs. 14a and b) and for a 3D curve with a complex shape (Figs. 14c and d). Figs. 14a and c present target curves with the initial subdivision curves computed according to the curvature analysis presented in Section 5. Final subdivision curves processed according to the complete algorithm described at the beginning of this section are presented on

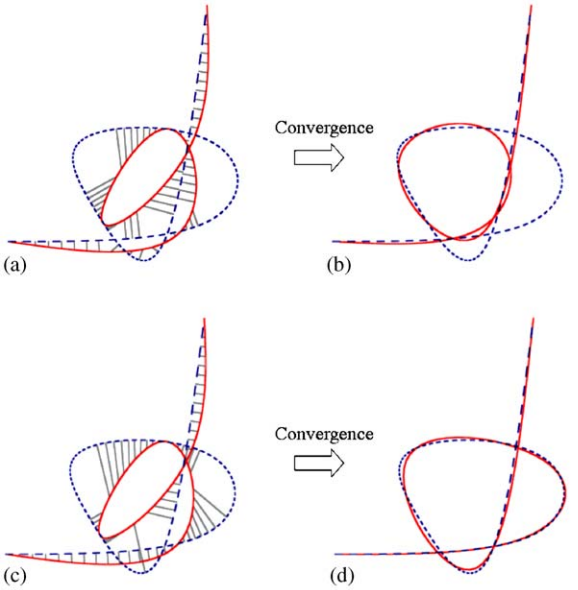

Fig. 13. Effect of the Active Footpoints Parameterization on a self-intersecting curve. Initial curve with associated footpoints computed by the classical method (a) and by our Active Footpoints Parameterization (c) and results of the optimization process (b and d) (5 iterations).

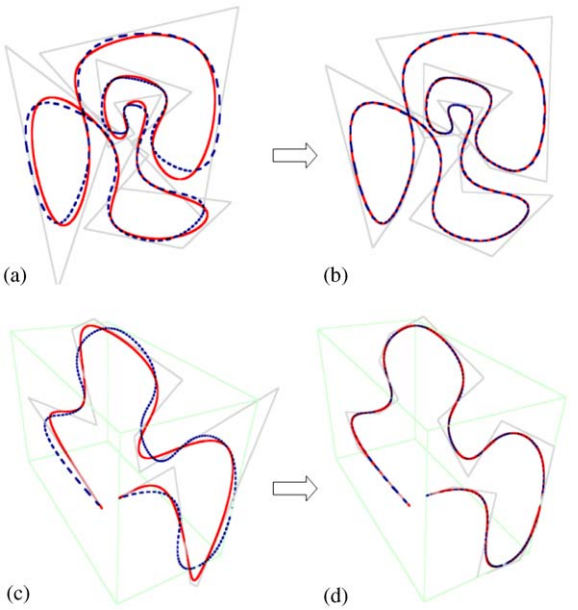

Fig. 14. Approximation results for a highly concave, self-intersecting target curve (a and b) and a complex 3D one (c and d) whose bounding box is represented. (a and c) Initial subdivision curves. (b and d) Results after the whole algorithm.

Figs. 14b and d. Chosen sample points, for the footpoint determination, are vertices of the curve after two subdivision steps and the error tolerance is $\varepsilon = 1 \times 10^{-3}$.

Table 2
Resulting errors (E), Final control points numbers (*CtrlNb*) and compression rates (CR) associated to curves approximation of Fig. 14

	(a)	(b)	(c)	(d)
CtrlNb	18	21	13	19
E ($\times 10^{-3}$)	12.34	0.99	26.48	0.86
CR	89.5%	87.7%	93.2%	90.2%

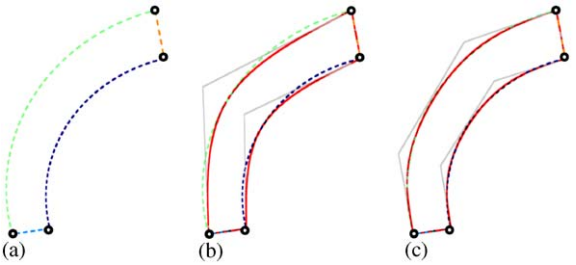

Fig. 15. Approximation results for a target curve with sharp vertices. (a) Decomposition of the target curve. (b) Initial subdivision curves. (c) Results after the whole algorithm.

Table 2 presents the results. The numbers of points of the target curves are 171 for Figs. 14a and b and 194 for Figs. 14c and d. At the end of the algorithm, final numbers of control points (*CtrlNb*) of the approximated subdivision curves are, respectively, 21 and 19. This is equivalent to compression rates (CR) of 87.7% and 90.2%. These are very satisfying results regarding to the small approximation errors E (respectively, 0.99×10^{-3} and 0.86×10^{-3}). All experiments were conducted on a PC, with a 2 Ghz XEON bi-processor. Processing times were 922 ms for Fig. 14b and 875 ms for Fig. 14d. Initial curve processing times were about 16 ms for each.

We also have conducted tests with target curves containing sharp vertices. An example is shown on Fig. 15. The target curve contains 78 vertices, whereas the control point number of the final control polyhedron is 8, with 4 points tagged *sharp*. The associated compression rate is 89.7%. We can ignore the amount of data carried by the flags embedded in each control point.

9. Future work for surface approximation

The subdivision curve approximation algorithm presented in this paper represents the first step in our surface mesh compression objective, by piecewise subdivision surface approximation (see Section 2). An example of this process is presented in Fig. 16.

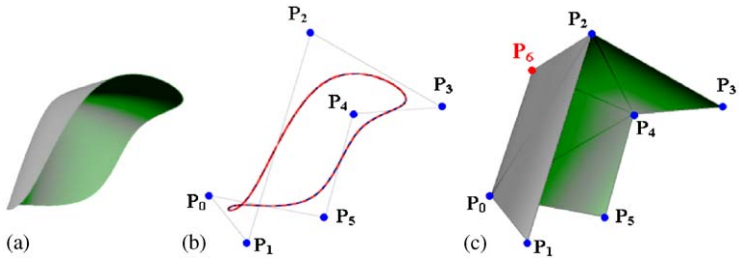

Fig. 16. Surface compression mechanism. (a) A mesh to compress. (b) Extraction and approximation of the boundary. (c) Construction of the final subdivision surface control polyhedron.

For a given surface patch, we first extract the boundary (see Fig. 16b), and determine the approximating subdivision curve, containing 6 control points in the example (P_0, P_1, P_2, P_3, P_4, P_5). This control polygon represents the boundary of the searched subdivision control polyhedron, thus we will use it, as a foundation to determine the approximating subdivision surface. In Fig. 16c, the control polyhedron was determined by adding a control point (P_6) to those of the boundary, meshing correctly these control points, and optimizing the placement of P_6 according to the target surface.

10. Conclusion

We have presented in this paper an efficient algorithm, based on the analysis of the data, for the inverse problem of curve subdivision. For any polygonal target curve, our algorithm computes the approximating piecewise smooth subdivision curve optimized in terms of the number of control points. A curvature analysis supported with theoretical foundations permits to compute near optimal numbers and placements of control points for the initial curve construction. The optimization scheme, based on the local quadratic approximations of the squared distance of curves, is an extension for subdivision rules of that presented by Pottmann et al. [13] and dealing with B-Splines. Our original footpoint determination method based on an active parameterization, allows to overcome the wrong projections occurring particularly for self-intersecting curves. Thus the stability of the method is highly increased by this good convergence guaranty. Many experiments demonstrate the efficiency of the method for approximation or compression of polygonal curves. This approximation method is involved in a larger surface compression scheme. Target objects are CAD meshes, previously segmented into surface patches. Our purpose is to determine the best approximating subdivision surface for each patch. The method presented in this paper approximates the boundary of a patch with a subdivision curve of which control polygon has the property to represent the boundary of the control polyhedron of the approximating subdivision surface. We plan now to develop the surface approximation algorithm taking as input the target surface patches and their associated subdivision boundary curves.

References

[1] G. Lavoue, F. Dupont, A. Baskurt, Constant curvature region decomposition of 3d-meshes by a mixed approach vertex-triangle, J. WSCG (WSCG'04 Proceedings) 12 (2004) 245–252.
[2] G. Lavoué, F. Dupont, A. Baskurt, Curvature tensor based triangle mesh segmentation with boundary rectification, Comput. Graph. Int. (CGI' 2004), IEEE Press, 2004, pp. 10–17.
[3] J. Schweitzer, Analysis and application of subdivision surfaces, Ph.D. Thesis, University of Washington, 1996.
[4] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, W. Stuetzle, Piecewise smooth surface reconstruction, in: Computer Graphics, SIGGRAPH 94 Proceedings, vol. 28, 1994, pp. 295–302.
[5] J. Warren, H. Weimer, Subdivision Methods For Geometric Design: A Constructive Approach, Morgan Kaufmann, Los Altos, CA, 2002.
[6] E. Catmull, J. Clark, Recursively generated b-spline surfaces on arbitrary topological meshes, Comput. Aided Design 10 (6) (1978) 350–355.
[7] C. Loop, Smooth subdivision surfaces based on triangles, Master's Thesis, Utah University, 1987.
[8] E. Lee, Choosing nodes in parametric curve interpolation, Comput. Aided Design 21 (6) (1989) 363–370.
[9] J. Hoschek, Intrinsic parametrization for approximation, Comput. Aided Geom. D. 5 (1) (1988) 27–31.
[10] E. Saux, M. Daniel, An improved hoschek intrinsic parametrization, Comput. Aided Geom. D. 20 (8–9) (2003) 513–521.
[11] T. Speer, M. Kuppe, J. Hoschek, Global reparametrization for curve approximation, Comput. Aided Geom. D. 15 (9) (1998) 869–877.
[12] H. Yang, W. Wang, J. Sun, Control point adjustment for b-spline curve approximation, Comput. Aided Design 36 (7) (2004) 539–552.
[13] H. Pottmann, S. Leopoldseder, M. Hofer, Approximation with active b-spline curves and surfaces, Pacific Graphics 02, IEEE Press, New York, 2002, pp. 8–25.
[14] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, Int. J. Comput. Vision 1 (1988) 321–332.

[15] E. Saux, M. Daniel, Data reduction of polygonal curves using b-splines, Comput. Aided Design 31 (8) (1999) 507–515.
[16] E. Davies, Machine Vision: Theory, Algorithms, Practicalities, Microelectronics and Signal Processing Series, Academic Press, New York, 1991.
[17] F. Meyer, Automatic screening of cytological specimens, Comput. Vision, Graphics Image Process. 35 (1986) 356–369.
[18] H. Pottmann, M. Hofer, Geometry of the squared distance function to curves and surfaces, in: H.-C. Hege, K. Polthier, (Eds.), Visualization Math. III, Springer, 2003, pp. 223–244.

About the Author—GUILLAUME LAVOUÉ received in 2002 his Engineering Degree in Electronic, Telecommunication and Computer Science from CPE-Lyon (France) and his M.S. degree in Image Processing from the university Jean Monnet of St-Etienne (France). He is actually Ph.D. student at LIRIS Laboratory in the University Claude Bernard of Lyon (France). His research interests include 3D digital image processing, geometric modeling and more precisely 3D compression and subdivision surfaces.

About the Author—FLORENT DUPONT received his B.S. and M.S. degree in 1990, and his Ph.D. in1994 from INSA of Lyon, France. Since 1998, he is Associate Professor. He now works at LIRIS Laboratory in the University Claude Bernard of Lyon, France. His technical research concerns 3D digital image processing, 3D compression and discrete geometry.

About the Author—ATILLA BASKURT received his B.S. degree in 1984, his M.S. in 1985, and his Ph.D. in 1989, all in electrical engineering from INSA of Lyon, France. From 1989 to 1998, he was Associate Professor at INSA of Lyon. Since 1998, he has been with the University Claude Bernard of Lyon, France, where he is a professor in electrical and computer engineering. He leads the images & videos group of LIRIS research laboratory at this University. This group performs image and video analysis and segmentation for image compression, image retrieval, shape detection, and identification. His technical research and experience includes digital image processing, image compression and segmentation, image indexing and retrieval, especially for multimedia applications.