Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 82 (2014) 253 – 265

Procedia Engineering

www.elsevier.com/locate/procedia

23rd International Meshing Roundtable (IMR23)

Curvature-Adapted Remeshing of CAD Surfaces

Franco Dassi[a], Andrea Mola[b], Hang Si[c]

[a]*Politecnico di Milano, Piazza Leonardo da Vinci 32, I-20133 Milano, Italy*
[b]*Scuola Internazionale Superiore di Studi Avanzati, Via Bonomea 265, 34136, Trieste, Italy*
[c]*Weierstrass Institute, Mohrenstr. 39, 10117, Berlin, Germany*

Abstract

A common representation of surfaces with complicated topology and geometry is through composite parametric surfaces as is the case for most CAD modelers. A challenging problem is how to generate a mesh of such a surface that well approximates the geometry of the surface, preserves its topology and important geometric features, and contains nicely shaped elements. In this work, we present an optimization-based surface remeshing method that is able to satisfy many of these requirements simultaneously. This method is inspired by the recent work of Lévy and Bonneel (*Proc. 21th International Meshing Roundtable*, October 2012), which embeds a smooth surface into a high-dimensional space and remesh it uniformly in that embedding space. Our method works directly in the 3d spaces and uses an embedding space in $\mathbb{R}^6$ to evaluate mesh size and mesh quality. It generates a curvature-adapted anisotropic surface mesh that well represents the geometry of the surface with a low number of elements. We illustrate our approach through various examples.

Peer-review under responsibility of organizing committee of the 23rd International Meshing Roundtable (IMR23)
Keywords: surface mesh generation; surface remeshing; curvature-adapted; anisotropic; mesh optimization; edge flip

1. Introduction

A composite parameterized surface consists of smoothly parameterized patches that meet at their common boundaries (curves and vertices). It is one of common representations to specify a surface which may have complicated geometry and topology. It can represent both manifold and non-manifold surfaces. In particular, it is the default representation in most of CAD modelers.

Let Σ be a composite parameterized surface in $\mathbb{R}^3$. We consider the following problem: How to generate a surface mesh of Σ with the following properties:

- The topology and geometric features of the original surface are preserved.
- Having a small number of elements with respect to a desired resolution.
- Having nicely shaped elements.

E-mail address: Franco Dassi, franco.dassi@polimi.it, Andrea Mola, mola.andrea@gmail.com, Hang Si, Hang.Si@wias-berlin.de

Peer-review under responsibility of organizing committee of the 23rd International Meshing Roundtable (IMR23)
doi:10.1016/j.proeng.2014.10.388

Such meshes are useful in many applications, such as visualization, geometric processing, and numerical simulation.

It has been shown that a surface mesh which consists of anisotropic elements fits many of the above requirements simultaneously. In this work, we consider how to generate a mesh that respects an intrinsic natural of the surface, i.e., the curvature. Given a composite parameterized surface, our goal is to generate a curvature-adapted surface mesh of that surface, i.e., more curved regions of the surface will be meshed by anisotropic elements, while nearly flat regions will be meshed by almost equilateral elements.

Surface remeshing has been an active research subject for nearly two decades, see a nice survey given by Alliez *et al.* [1]. Previous methods can be roughly characterized into two groups: parameterized remeshing in 2d [11,20] and direct remeshing in 3d [8,21,22]. The focus of most of these methods is to generate 3d isotropic surface meshes. Many recent methods have been developed for creating anisotropic surface meshes [16–19,23]. Most of these methods use (or reconstruct) a metric tensor field to describe the anisotropy of the surface.

In this paper, we propose a new method for remeshing 3d composite parameterized surfaces based on the idea of higher dimensional embedding [2,3,5]. We use the normal information of the surface, and embed the surface into $\mathbb{R}^6$. Our method directly optimizes a triangular mesh of the surface in such a way that its triangles are as uniform as possible in $\mathbb{R}^6$. It will result a curvature-adapted mesh of the surface. Our method will naturally result an anisotropic mesh for a surface which has rapidly varying principle curvatures. Moreover, it easily preserves important geometric (sharp) features of the surface.

Our method does have the following limitations: (i) It only ensures the mesh quality in the embedding space, but not the usually mesh quality (such as the smallest angle and aspect ratio) in $\mathbb{R}^3$; (ii) It does not handle anisotropy because of physics, eg, PDEs, which may be an arbitrary user-defined metric tensor field.

Our re-meshing method will create new nodes or display existing nodes whenever the quality requirements are not fulfilled. In both cases the underlying CAD surface is queried to obtain the correct values for the local surface normals, and the new nodes located on the surface. In addition to use the default parameterizations of the surface patches, we implemented a set of basic geometrical operations which can query the mesh and CAD data structures. In particular, unit vectors identifying the directional normal to the object surface are used for the evaluation of the mesh quality in the embedding space. Such tools represent a first generalization step towards the treatment of arbitrary geometries, of the CAD interface methods developed for quadrilateral surface mesh generation, see [25].

The remainder of this paper is organized as follows: Section 2 presents the methodology of our surface remeshing algorithm. The proposed method is described in detail in Section 3. Section 4 describes a set of geometric operations to query the CAD surfaces. Some experimental results are demonstrated in Section 5. Finally, a summarization and outlook of future work are given in Section 6.

2. The Methodology

2.1. Surface Embedding in $\mathbb{R}^6$

The re-meshing method presented in this paper is inspired by the method of Lévy and Bonneel [2]. The basic idea is pioneered by Cañas and Gortler [3] and is originated in the application of feature characterization [4,5] from image processing [6]. In order to re-mesh a surface in $\mathbb{R}^3$, it first maps it into a high-dimensional space, i.e., $\mathbb{R}^n, n > 3$, then it re-meshes the embedded surface uniformly in that space, and transform the mesh back to $\mathbb{R}^3$.

Clearly, the choice of the map greatly affects the properties of resulting mesh. For a smooth surface, it is natural to consider the Gauss map of the surface. Given a surface Ω in $\mathbb{R}^3$, one can use the components of the normals of the surface as the codimension, and embed it into $\mathbb{R}^6$ via the embedding: $\Phi : \Omega \to \mathbb{R}^6$ defined by:

$$\Phi(\mathbf{x}) = (x, y, z, s\,n_x, s\,n_y, s\,n_z)^t, \tag{1}$$

where (n_x, n_y, n_z) denotes the unit normal to Ω at $\mathbf{x}$, and $s \in [0, +\infty)$ is a user-specified constant. This embedding Φ allows us to approximate the geodesic edge lengths in Ω by the Euclidean edge lengths in $\Phi(\Omega)$. Each edge length in $\Phi(\Omega)$ is determined by two parts:

- its Euclidean length in $\mathbb{R}^3$; and
- the variation of the normals of its endpoints, scaled by the parameter s.

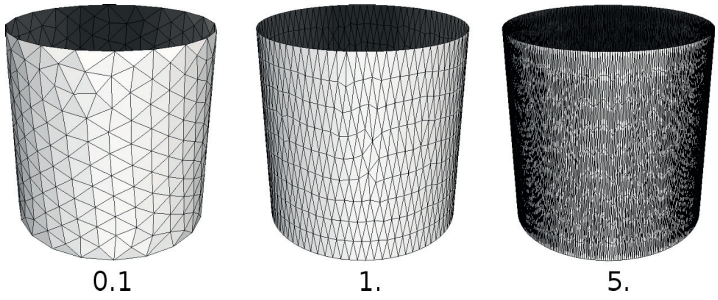

Fig. 1. An experiment of using different values of s to re-mesh the surface of a cylinder.

By this transformation, in flat regions of Ω, the lengths of edges remain the same in $\Phi(\Omega)$. While in regions which have rapidly varying normals, the lengths of edges in $\Phi(\Omega)$ become much larger than theirs in $\mathbb{R}^3$. Since the distance in $\mathbb{R}^6$ are affected by the normals, an isotropic mesh of the surface $\Phi(\Omega)$ in $\mathbb{R}^6$, when transformed back into $\mathbb{R}^3$, will become a curvature-adapted anisotropic mesh of Ω. This concept has been successfully applied in generating curvature-adapted surface meshes [2,7].

The constant s in (1) distorts the evaluation of the lengths and the angles in the embedded space. It is an important parameter that governs the level of anisotropy, see Figure 1 for an example.

2.2. *Surface Remeshing in* $\mathbb{R}^6$

By embedding a surface in higher dimensions motivates the new problem: *How to generate an isotropic good quality surface mesh in this embedding space?* A direct generalization of available methods in 3d is possible. But this will be impractical due to the $d!$ cost of memory requirement.

Lévy and Bonneel [2] overcome this difficulty by using their *Vorpaline* (Voronoi Parallel Linear Enumeration) technique to compute a restricted centroidal Voronoi diagram (CVT) embedded in 6d. It directly computes the 6d Voronoi cells by iterative half-space clipping. It requires only the nearest neighbor informations for a point set in $\mathbb{R}^6$. This method creates a curvature-adapted anisotropic surface mesh. However, it is a global mesh optimization method, which means that it does not support local refinement and mesh modifications. It may not preserve all important geometric features. For example, sharp features may be smoothed and disappear in the resulting mesh.

An alternative approach to perform the surface remeshing in $\mathbb{R}^6$ is proposed in [14]. This method directly optimizes an initial surface mesh in $\mathbb{R}^3$, but uses the quantities (edge lengths and angles) from $\mathbb{R}^6$. This method fits in the well-developed mesh adaptation framework. In principle, any isotropic surface remeshing method that work in $\mathbb{R}^3$ can just as well be applied in this method. With this method, local mesh refinement and modifications are supported. Moreover, geometric (sharp) features can be easily preserved. In the next section, we extend this method for meshing composite parametrized surfaces.

3. The Re-meshing Approach

The basic idea behind the anisotropic mesh adaptation is to distort the distance. In this framework we are going to achieve this goal moving from the standard scalar product in $\mathbb{R}^6$. Consider a surface Ω and two points $A, B \in \Omega$, we apply the map Φ and we have:

$$\Phi(A) = (x_A,\, y_A,\, z_A,\, sn_A,\, sv_A,\, sw_A)^t\,,$$
$$\Phi(B) = (x_B,\, y_B,\, z_B,\, sn_B,\, sv_B,\, sw_B)^t\,,$$

where x_A, y_A, z_A and x_B, y_B, z_B are the $\mathbb{R}^3$ coordinates of A and B, respectively, and n_A, v_A, w_A and n_B, v_B, w_B are the components of the unit normal vectors to Ω at A and B, respectively. Then, the standard scalar product in the embedded space is

$$(A, B)_{6d} := x_A x_B + y_A y_B + z_A z_B + s^2(n_A n_B + v_A v_B + w_A w_B)\,. \tag{2}$$

Via this scalar product, we could define both the length of a segment AB, $l_{AB} := \sqrt{(A-B, A-B)_{6d}} = \|A-B\|_{6d}$ and the angle

$$\cos(\angle ACB) := \frac{(A-C, B-C)_{6d}}{(A-C, A-C)_{6d}(B-C, B-C)_{6d}},$$

where C is another point of the surface Ω.

3.1. Local Mesh Operations

The proposed algorithm applies a series of local surface mesh modifications directly on the mesh. The most well-known and commonly used local modifications are: edge-flip, edge-collapse, vertex insertion, and vertex smoothing (Figure 2). These operations are already extensively discussed in literature, see [8–11]. In this section, we briefly describe how we implemented these operations.

Consider a surface Ω and its discretization Ω_h composed by triangular elements. Let T be a triangle of Ω_h, we define the quality of T as the minimum 6d-angle of T. By this definition, the best triangle will be the equilateral triangle in 6d and the maximized minimum 6d-angle 60^o.

Given a surface Ω it is possible to define some geometric quantities such as the curvature and the normals of a surface $\Omega' \subset U_\delta$ parallel to Ω and U_δ is a shell around Ω, see [12]. To capture how close the mesh is approximating the CAD geometry, we introduce the concept of energy of an edge and a triangle of the mesh.

Definition 3.1. *Let T be a triangle in Ω_h. We define the energy of T as the following quantity*

$$E_T := \frac{1}{|T|}\int_T (1 - n \cdot n_T)^2, \tag{3}$$

where $|T|$ is the area of the triangle T, n_T is the normal of the triangle T and n is the extension of the normal to the surface Ω.

Definition 3.2. *Let e be an edge of Ω_h. We define the energy of e as the following quantity*

$$E_e := \frac{1}{|e|}\int_e (1 - n \cdot n_{T_1})^2 + (1 - n \cdot n_{T_2})^2, \tag{4}$$

where $|e|$ is the length of the edge e, T_1 and T_2 are the triangles of the mesh Ω_h that shares the edge e whose normals are n_{T_1} and n_{T_2}, respectively, and n is the extension of the normal to the surface Ω.

Definition 3.3. *Let T_1 and T_2 be two adjacent triangles. We define the total energy of these triangles be the quantity:*

$$E_{T_1,T_2} := E_{T_1} + E_{T_2} + \sum_{e \in \mathcal{S}} E_e, \tag{5}$$

where $\mathcal{S}$ is the set of edges of both T_1 and T_2, E_{T_1} and E_{T_2} are the energy of the triangles T_1 and T_2, respectively, and E_e are the energies of the edges in $\mathcal{S}$.

Definition 3.4. *Let T be a triangle of Ω_h and α be an angle. We say that this triangle is inverted if the maximum angle between the normal to the triangle, n_T, and the normal to the surface Ω evaluated at the vertices of the triangle is greater than α, in this work we consider $\alpha = \pi/2$.*

Edge-flip is the most efficient and effective local operation to improve simultaneously the geometry approximation and the quality of the surface mesh. An edge-flip on an edge e_1 will remove the two faces T_1 and T_2 that share at e_1 and replace them with two new faces T_1' and T_2' whose common edge is e_2. As a result, edge e_1 is replaced by e_2.

In our algorithm, we want to flip an edge e_1 if the new triangles are "better" than the old ones with regard to both geometry approximation and the 6d mesh quality. Specifically, we will flip e_1 if one of the following conditions is met: (a) (geometric approximation) one of the faces adjacent to the edge is inverted and the new configuration is associated

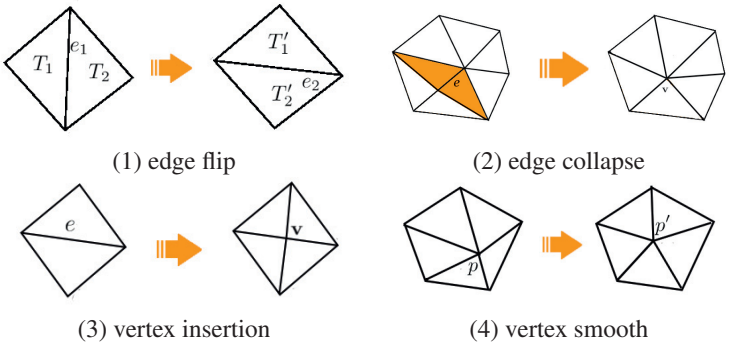

Fig. 2. The common local mesh operations.

with a lower total energy; and (b) (mesh quality) if both T_1 and T_2 are not inverted and the smallest 6d-angle of the two new configuration is larger.

Unfortunately, it is not always possible to do an edge-flip. Specifically, an edge e_1 is *flippable* if it satisfies all these constraints: (i) the new edge e_2 does not already belong to the mesh; and (ii) the edge e_1 does not belong to a sharp feature of the surface Ω.

Our edge-flip algorithm uses two stacks S and S_1. Initially, the stack S keeps all edges to be checked and flipped, and the stack S_1 is empty, and it will keep edges which are not flippable. Edges in S_1 are tried again if any flip has been done. Once an edge is get flipped, we push the link edges of the new triangles into stack S_1, hence flips may propagate to the neighboring edges. The flip process stops when no edge is flippable.

Edge Collapse is a common operation for simplifying meshes. An edge collapse unifies the two endpoints of the edge and two adjacent faces of this edge vanish. The unification of the two endpoints can be done at either one of the endpoints or at a new location inside the cavity of adjacent faces of this edge. In this work we choose one of the endpoints as the new vertex. Then, we push all the link edges of this vertex into a stack, and use the edge-flip algorithm to locally improve the mesh.

Edge Splitting is a major operation in our mesh adaptation. In this operation a vertex is put at the middle point of an edge e and the triangles adjacent to e are split.

The creation of a new point in an edge is a key task. The new point must be on the CAD surface. In Section 4, we will describe how to project a point on CAD models in a good way. Note that that this operation may lead to an undesired mesh configuration such as inverted elements. We experimentally show that our edge flip algorithm automatically fixes these undesired features, see [14] for more details.

Vertex Smoothing. Given vertex p, the smoothing operation finds a new location such that the local mesh quality is improved without changing the mesh topology.

A generic smoothing method moves a point to a new location

$$p' = p + \alpha \sum_{p_i \in \omega_p} f(d(p\,, p_i))u_i\,, \tag{6}$$

here α is a constant, f is a function $f : \mathbb{R} \to \mathbb{R}$, ω_p is the set of vertexes that are connected to p and u_i are the unit vectors that identifies the direction from p_i to p, see Figure 2. Finally, d is the distance between p and p_i

Different smoothing methods are characterized by different choices of the parameter α and the function f in Equation (6). For example the classical Laplacian smoothing, [13], is defined by $\alpha = 1/\#\omega_p$ and $f(d) = -d$, where $\#\omega_p$ is the cardinality of the set ω_p. In this paper we use the smoothing algorithm proposed in [14]. The basic idea behind this smoothing technique is using the distance d evaluated in the embedded space.

Like the splitting algorithm, once we have find a new location for p, the point is projected on the real surface in order to lie as closely as possible on the surface.

3.2. The Re-meshing Procedure

Let Ω be a CAD surface, i.e., it is a composite parameterized surface, i.e.,

$$\Omega = \bigcup_{i=1}^{n} \Omega_i \,, \quad \text{and} \quad \Omega_i \cap \Omega_j = \begin{cases} \emptyset \\ \gamma_{ij} \end{cases} , \tag{7}$$

where the curve γ_{ij} is the common boundary of patches Ω_i and Ω_j. Such curves usually represent important geometric and sharp features of the surface, hence they are needed to be preserved.

The proposed re-meshing algorithm takes as input an initial triangular mesh Ω_h^i of Ω, a desired 6d-length, L, a minimum 6d-angle, θ_{min} and a value of s. Then it starts to modify Ω_h^i with the basic local mesh operation described in the previous subsection in order to get a final mesh Ω_h^f as uniform as possible in the embedded space, i.e., a mesh where $\|e\|_{6d} \approx L \; \forall e \in \mathcal{E}$, where $\mathcal{E}$ is the skeleton of the mesh Ω_h^f.

SAMPLING(Ω, Ω_h, Q, L)
Data: Q is a queue of triangles in Ω_h.

```
1:  while Q is non-empty do
2:     pop a face f from Q;
3:     Let e be the longest edge of f;
4:     if ||e||_6d > 1.5 L, then
5:        split e by adding v ∈ Ω into Ω_h;
6:        update Q;
7:     end if
8:  end while
9:  improve the mesh quality by edge-flips;
10: put all the triangles in Q;
11: while Q is non-empty do
12:    pop a face f from Q;
13:    Let e be the shortest edge of f;
14:    if ||e||_6d < 0.5 L, then
15:       collapse e onto one of the end-points;
16:       update Q;
17:    end if
18: end while
19: improve the mesh quality by edge-flips;
```

Algorithm 1: The sampling algorithm.

The proposed re-meshing procedureconsist of two main phases:

(i) **Sampling** – split and collapse the edges in the mesh that are too long or too short with respect to the desired 6d-length, see Algorithm 1;

(ii) **Optimizing** – remove the 6d-angles that are lower than θ_{min} and smooth the nodes in order to make the 6d-length of the edges as uniform as possible, see Algorithm 2.

Note that the vertex smoothing operation is in the most internal loop of the optimization step, see line 3 in Algorithm 2. To increase the speed of the entire re-meshing process, we decide to move not all of the vertexes of the mesh, but only the ones that are furthest away from their desired positions.

The proposed re-meshing procedure applies Algorithm 1 and 2 iteratively to get the final mesh Ω_h^f. Algorithm 3 summarizes this re-meshing procedure.

OPTMIZING(Ω, Ω_h, L, θ_{min}, I, J)
Data: I and J are user-specified iterations.

1: **for** $j \in \{1, ..., \mathrm{I}\}$ **do**
2: **for** $k \in \{1, ..., \mathrm{J}\}$ **do**
3: Smooth the 30% of the vertexes;
4: improve the mesh quality by edge-flips;;
5: **end for**
6: Collapse edges for removing angles $< \theta_{min}$;
7: improve the mesh quality by edge-flips;
8: **end for**

Algorithm 2: The optimizing algorithm.

RE-MESHING(Ω, Ω_h, L, θ_{min}, K, I, J)
Data: K is user-specified iterations.

1: **for** $j \in \{1, ..., \mathrm{K}\}$ **do**
2: call SAMPLING(Ω, Ω_h, Q, L)
3: call OPTMIZING(Ω, Ω_h, L, θ_{min}, I, J)
4: **end for**

Algorithm 3: The re.meshing algorithm.

4. Enhanced Geometric Operations on CAD Surfaces

A common strategy for generating grids on CAD surfaces is to exploit the manifold space for the generation of new nodes. In this framework, any node is created in the parametric space of each —NURBS or B-spline— CAD patch, resulting in a point which will be automatically located on the desired surface. Despite this clear advantage, such approach presents several drawbacks when applied to industrial CAD geometries.

- It is very common in the CAD framework that the geometrical models are characterized by surfaces composed by patches which are not logically connected, i.e., not water-tight. Any small gap or overlap among the patches will also be found in the mesh generated.
- Even the presence of a continuous junction between two adjacent patches, the mesh will present non conformal elements unless the surface parameterization coincide on the shared edge (see Fig. 3).

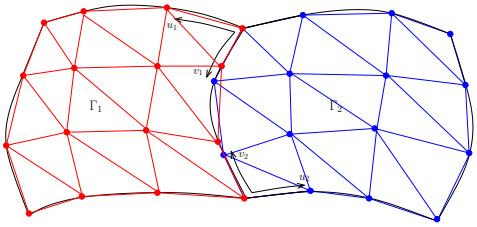

Fig. 3. Neighboring parametric surfaces with non consistent parameterizations at the common edge. Using such parameterizations to obtain the nodes of computational grid might result in a non conformal interface between the grids generated on the two CAD surfaces. The situation can get even worse if the faces are not even connected.

The latter —very strong— constraint is too restrictive for the common practice of CAD model design. Yet, even in presence of a CAD model fulfilling such requirement, the mesh generated with this approach will present a series of edges constrained to be on the patches junctions. In most cases such junctions are not geometrically meaningful, as they connect faces with continuous normal (see Fig. 6). Thus, constraining the mesh nodes to be located on the patches junctions is unacceptable, especially when the model contains extremely small patches.

In this work, we employed a different approach which consists of generating new nodes in the physical, three dimensional space in which the CAD model is embedded. We implemented a surface projector which queries the CAD data structure to obtain the surface projection of the newly generated points. The developed surface projector consists of the projection of the specified point on the surface along a direction prescribed by the user. In practice, this projection is realized by intersecting a straight line passing through the desired point and directed along the specified direction, with each of the surface patches. The projected point is selected as the closest intersection with respect to the original point. This operation only needs to compute line-surface intersections which can be done efficiently. In addition, this projection algorithm leads to higher quality meshes, as the axis of projection is not solely dependent on the underlying CAD surface, but can be selected to preserve mesh quality. We decide to get the projection axis from the triangular surface mesh. More precisely we use the normal approximation at a mesh vertex proposed in [15]. Figure 4 illustrates the cell refinement procedure obtained adopting the surface projection in the mesh normal direction. It is worth noting that the projection algorithm described is completely independent of the parametrization on each patch composing the CAD surface. Thus, the grid generated with this strategy will not have nodes located on the patches junctions, nor non conformal interfaces across the patches. Once the actual projection is performed, the projector function implemented is also able to provide the local geometrical features of the surface needed by the re-meshing procedure. In particular, the surface normal unit vector and mean curvature value at the projected point can be evaluated through the projector.

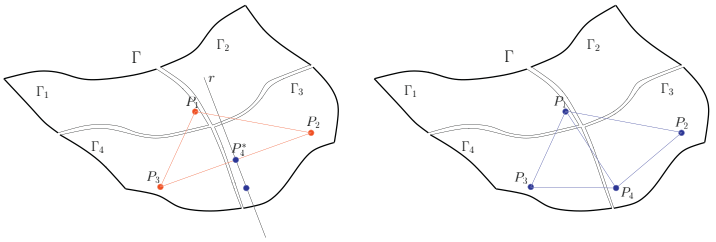

Fig. 4. Triangle refinement procedure which makes use of the surface projection in the mesh normal direction. On the left, the new node P_4^* is created on the mesh edge flagged for refinement. The mesh normal direction is estimated via the method proposed in [15]. A straight line r having such direction and passing from P_4^* is created and intersected with the surface Γ. On the right, the two cells resulting from the refinement are shown.

To fully support the mesh generation algorithm, a strategy for the generation of new nodes on sharp edges of the CAD model must also be implemented. To preserve the geometrical features of the specified object, it is in fact important that whenever the grid is refined or modified in correspondence with a sharp edge the new or displaced nodes are still located on the edge. Thus, we implemented an arc-length projector which, once assigned two points defining an arc on a CAD curve, is able to identify the point splitting such arc at a specified length fraction. Figure 5 shows such an example. Such arc-length projector is employed in the re-meshing procedure to split all the mesh edges located on CAD curves and flagged for refinement. Also in this case, the new nodes of the grid are generated without any dependence on the local CAD curve parameterization.

The surface and arc-length projectors described represent the only geometrical operations effectively carried out on the CAD model during the re-meshing procedure. Yet, for the re-meshing procedure and the projectors to work correctly, the information contained in the CAD model and mesh data structured must be properly organized. Firstly, it is important to identify the sets of surfaces in the model, that are mutually separated by sharp edges. Each of such sets (which have for most geometries null mutual intersections) is then used to create a separate surface projector. Finally, all the mesh cells are assigned to one of the projectors created. This ensures that the new points on mesh cells located on the two sides of a sharp edge of the geometry will be safely placed on the respective sides of the edge, preserving its sharpness through each mesh refinement. In a similar way, all the curves defining the sharp edges in the CAD model are identified. Then, the identified curves are checked for possible duplicates (i.e.: curves that coincide up to a specified tolerance), and eventually concatenated into the smallest set of smooth curves. These curves will be used for the generation of arc-length projectors. Finally, all the mesh edges on such curves are assigned to the corresponding arc-length projector.

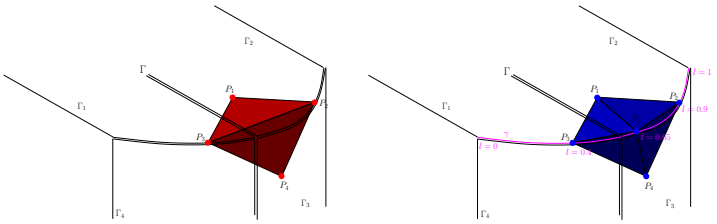

Fig. 5. A sketch of the sharp edge refinement procedure. In the picture, the CAD surface Γ is composed of unconnected patches $\Gamma_1, \Gamma_2, \Gamma_3$ and Γ_4. The surface normal has a significant jump between patches Γ_1 and Γ_4, and patches Γ_2 and Γ_3. Thus, CAD surface Γ presents a sharp edge. In the preliminary CAD processing phase, the boundaries of patches $\Gamma_1, \Gamma_2, \Gamma_3$ and Γ_4 lying along the sharp edge are checked for possible duplicates and concatenated in a single curve γ. In this example, the curvilinear abscissa on γ is used to refine the edge $\overline{P_2P_3}$ of the original grid (left picture). The new point P_5 splits the arc of γ between P_2 and P_3 in two arcs of equal length (right picture).

5. Application Examples

In this section, several examples are presented to illustrate the properties of the proposed method. The inputs are surfaces from CAD models that present sharp features. The statistical information and the CPU times are given in Table 1. To evaluate the level of anisotropy of the resulting mesh, we consider the so-called aspect ratio: $q(T) := 2\,\frac{r_T}{R_T}$, where T is a triangle of the mesh, r_T and R_T are the radii of the inscribed and circumscribed circle, respectively. If $q(T) \approx 0$, the triangle T is stretched, while, for triangles close to the equilateral one, we have $q(T) \approx 1$.

	Examples	1	2
1.	L	50	500
2.	# vertexes in Ω_h^i	681	113
3.	# triangles in Ω_h^i	1280	192
4.	# vertexes in Ω_h^f	2185	2526
5.	# triangles in Ω_h^f	1158	4679
6.	Sampling Time (sec.)	2	113
7.	Optimizing Time (sec.)	78	336
8.	Minimum Aspect Ratio	2.112e-03	4.059e-03

Table 1. Statistics of Examples. Here we call Re-meshing(), see Algorithm 3, with different surfaces Ω and $K = 3, I, J = 4$.

Example 1. The CAD model is shown in Figure 6. Not all the curves in this model have to be considered as sharp features, such as the curves γ_{12} and γ_{34} which join the surface Ω_1 and Ω_3 with Ω_2 and Ω_4, respectively. Such curves do not have to be preserved. The initial and final mesh of our method are shown in Figure 7. Several views of the details of the final mesh and a histogram of the distribution of the minimum 6d-angles are shown in Figure 8.

Example 2. The input geometry of this example is a CAD model of a ship (Figure 9). It is composed by many smooth surface patches, joined along their common curves. Note that not all curves in this model need to be preserved. The initial and final meshes of this example are shown in Figure 10. Some details of the final mesh are displayed in Figure 11 and Figure 12, respectively.

Example 3. In this example, a CAD model that used by the work of Levy and Bonnel [2] was tested. It is shown in Figure 13. We used the `Gmsh` library [24] (`http://geuz.org/gmsh/`) to produce an initial surface mesh of this model. Our method was then used to sample and optimize the mesh. We compared our resulting mesh with that given

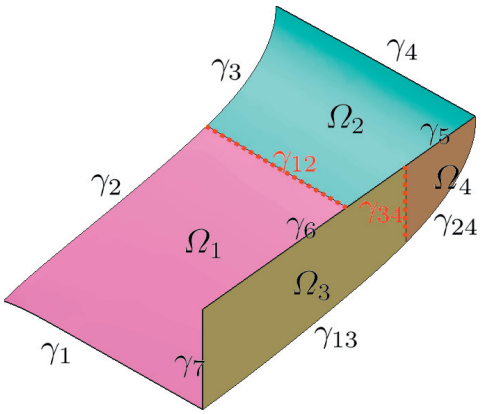

Fig. 6. The CAD surface of Example 1. It consists of the curves, γ_*, and the smooth surface patches, Ω_*. The highlighted curves γ_{12} and γ_{23} are not treated as sharp features.

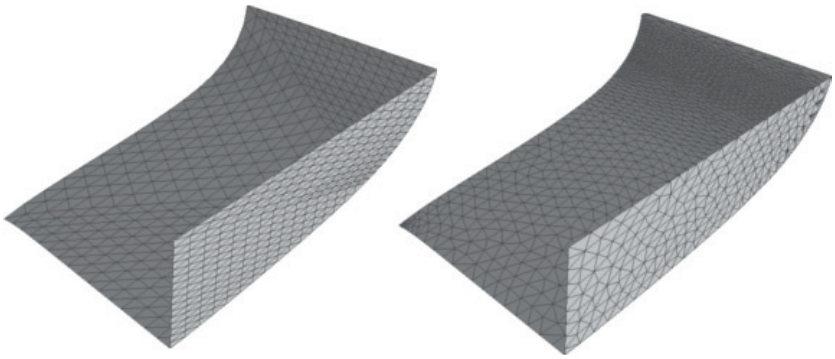

Fig. 7. Left: the initial mesh. Right: the final adapted mesh.

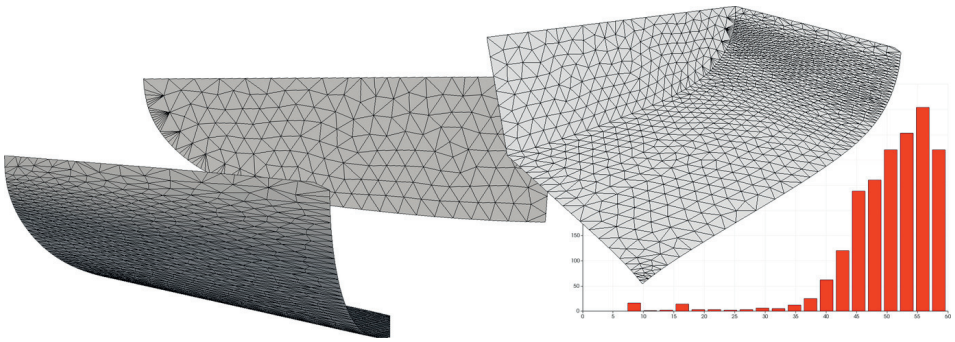

Fig. 8. Details of the mesh presented in Figure 7 and the histogram of the distribution of the minimum 6d-angle of the adapted mesh.

in [2]. Some detail of the difference are shown in Figure 14. One can observe that our method well preserved the geometry and did not oversample the surfaces.

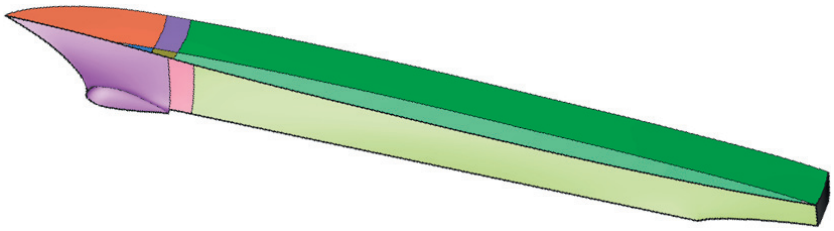

Fig. 9. The CAD surface of Example 2. Different colors highlight the smooth surface patches of the ship.

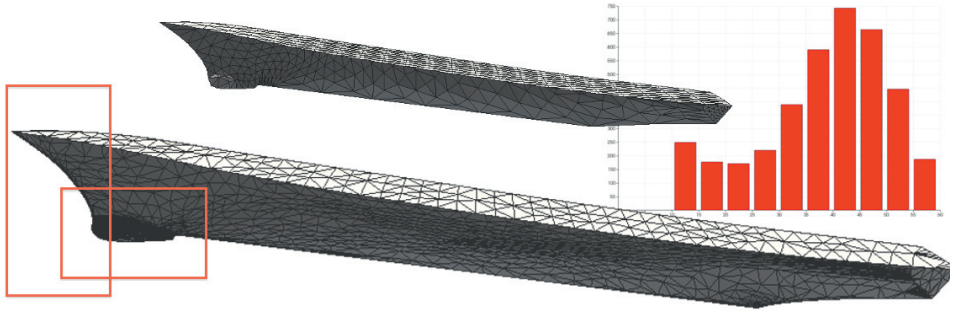

Fig. 10. Top: the initial mesh. Bottom: the final adapted mesh and the histogram of the minimum 6d-angles of the triangles of the adapted mesh.

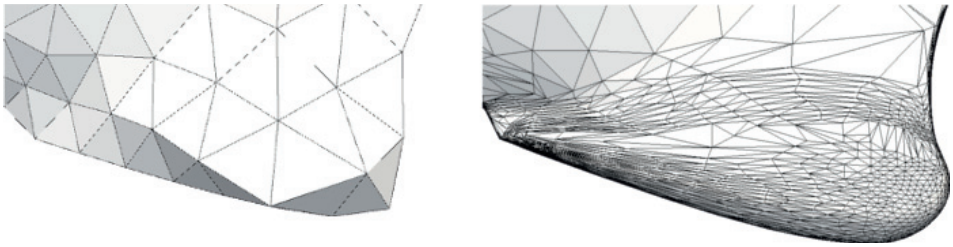

Fig. 11. Mesh detail of Figure 10. Left: the initial mesh. Right: the final mesh.

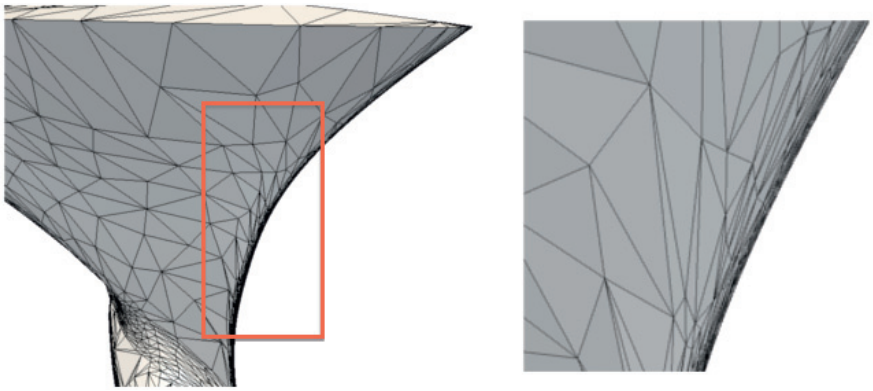

Fig. 12. Mesh detail of Figure 10. Left: detail of the final mesh. Right: detail of the rectangular region of the left.

6. Conclusion

In this paper, we presented a curvature-adapted surface remeshing method for remeshing CAD surfaces. It is based on the idea of high-dimensional embeddings of surfaces. This method is in principle simple. It fits the well-developed

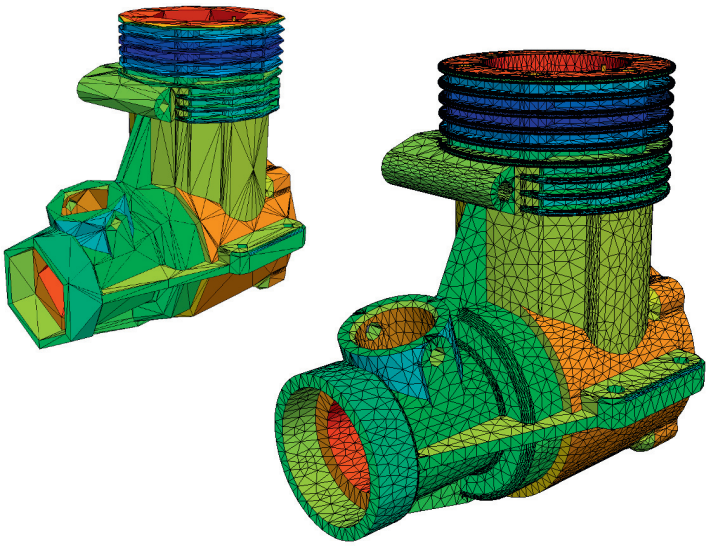

Fig. 13. **Example 3**. The initial mesh (left) and the optimized mesh (right).

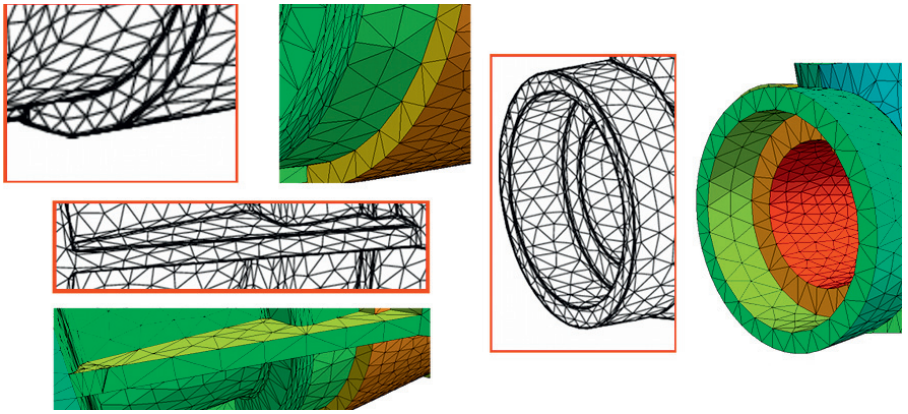

Fig. 14. **Example 3**. Detail of the mesh generated by the method in [2] (shown in red boxes) and the same detail of the mesh obtained by our method (colored).

mesh adaptation framework. It has several notable advantages. For instances, important geometrical features are preserved; it is robust in handling strong anisotropy, and it is easy to implement. Our experimental results showed that this method is able to produce well-adapted meshes for various CAD surfaces.

There are many questions which are still open. A very important theoretical question is: How well does this mapping Φ approximate the geodesic distances in 3d surfaces? Are there upper or lower bounds on distance variations by this mapping? A theoretical study of these question could lead to more efficient methods, and will result much smaller mesh size. The edge-flip algorithm we described in this paper appears very useful in improving both geometry approximation and mesh quality. However, its termination is not yet proven. We found that the threshold angle for checking inverted faces is very crucial. A good value will give edge-flip more freedom and may produce highly stretched triangles. In practice, many surfaces are given as a polygonal mesh, i.e., the original geometry is not available. A good recovery and estimation of the surface normals are necessary in order to achieve good results. We

plan to implement such feature into our code. Finally, the running time for our implementation is far from optimal. There are many possibilities to improve it.

Acknowledgments

We sincerely thank our reviewers for their precious comments, suggestions, and corrections of our manuscript.

References

[1] P. Alliez, M. Attene, C. Gotsman, and G. Ucelli, Recent advances in remeshing of surfaces, D. L. and M. Spagnuolo, editors, Shape Analysis and Structuring, Springer, 2008, pp. 53–82.
[2] B. Lévy and N. Bonneel, Variational anisotropic surface meshing with Voronoi parallel linear enumeration, Proceedings 21st International Meshing Roundtable, 2012, pp. 349–366.
[3] G. D. Canas and S. J. Gortler, Surface remeshing in arbitrary codimensions, Vis. Comput., 2006, 22(9):885–895.
[4] H. Pottmann, T. Steiner, M. Hofer, C. Haider, and A. Hanbury, The isophotic metric and its application to feature sensitive morphology on surfaces, T. Pajdla and J. Matas, editors, Comput. Vis., 2004, volume 3024 of Lecture Notes in Computer Science, pp. 560–572.
[5] Y.-K. Lai, Q.-Y. Zhou, S.-M. Hu, J. Wallner, and H. Pottmann, Robust feature classification and editing.IEEE Trans. Vis. Comput. Graph., 2007, 13(1): pp. 34–45.
[6] R. Kimmel, R. Malladi, and N. Sochen, Images as embedded maps and minimal surfaces: Movies, color, texture, and volumetric medical images, Int. J. Comput. Vis., 2000, 39(2):111–129.
[7] D. Kovacs, A. Myles, and D. Zorin, Anisotropic quadrangulation, Proceedings of the 14th ACM Symposium on Solid and Physical Modeling, 2010, pp. 137–146.
[8] H. Hoppe, T. DeRose, D. Tom, M. John, and W. Stuetzle, Mesh optimization, Technical Report 93-01-01, Department of Computer Science & Engineering, University of Washington, Seattle, Washington 98195, 1993.
[9] H. Hoppe, Progressive meshes, Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, SIGGRAPH '96, 1996, pp. 99–108.
[10] H. de Cougny and M. S. Shephard, Surface meshing using vertex insertion, Proceedings of the 5th International Meshing Roundtable, 1996, pp. 243–256.
[11] P. J. Frey and H. Borouchaki, Geometric surface mesh optimization, Comput. Vis. Scie., 1998, 1(3): pp.113–121.
[12] D. Gilbarg and N. S. Trudinger, Elliptic partial differential equations of second order, Springer, 2001, 224.
[13] D. Field, Laplacian smoothing and Delaunay triangulation, Comm. App. Numer. Meth, 1988, 4: pp. 709–712.
[14] F. Dassi and H. Si, A curvature-adapted anisotropic surface re-meshing method, Tetrahedron IV Proceedings, 2013, to appear.
[15] K. Hyoungseok and K. Hosook, New computation of normal vector and curvature, WSEAS Trans. Comp., 2009, 8(10): pp. 1661–1670.
[16] P. S. Heckbert and M. Garland, Optimal triangulation and quadric-based surface simplification, Comput. Geo., 1999, 14(1-3): pp. 49–65.
[17] X. Jiao, A. Colombi, X. Ni, and J. C. Hart, Anisotropic mesh adaptation for evolving triangulated surfaces, Proceedings 15th International Meshing Roundtable, 1989, pp. 173–190.
[18] S.-W. Cheng, T. K. Dey, and E. A. Ramos, Anisotropic surface meshing, Proceedings ACM-SIAM Sympos, Discrete Algorithms, 2006, pp. 202–211.
[19] M. S. Floater and K. Hormann, Anisotropic Delaunay meshes of surfaces, To appear in ACM Trans. Graph., 2012.
[20] H. Pottmann, T. Steiner, M. Hofer, C. Haider, and A. Hanbury, Surface Parameterization: a Tutorial and Survey, Advances in Multiresolution for Geometric Modelling, Springer Berlin Heidelberg, 2005, pp. 157-186.
[21] S. J. Owen, D. R. White, and T. J. Tautges, Facet-based Surfaces for 3d mesh generation, Proceedings 11th International Meshing Roundtable, 2002, pp. 297–311.
[22] V. Surazhsky and C. Gotsman, Explicit Surface Remeshing, Proceedings of Eurographics Symposium on Geometry Processing, 2003, pp. 17-28.
[23] P. Laug and H. Borouchaki, High quality geometric meshing of CAD surfaces, Proceedings 20th International Meshing Roundtable, 2011, pp. 63–80.
[24] C. Geuzaine and J. F. Remacle, Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities, Int. J. Numer. Meth. Engng., 2009, 79: pp. 1309–1331.
[25] A. Mola and L. Heltai and A. De Simone, A fully nonlinear potential model for ship hydrodynamics directly interfaced with CAD data structures, International Society of Offshore and Polar Engineers (ISOPE) 2014 Busan Conference Proceedings, 2014, to appear.