VI International Conference on Adaptive Modeling and Simulation
ADMOS 2013
J. P. Moitinho de Almeida, P. Díez, C. Tiago and N. Parés (Eds)

AN IMPRINTING ALGORITHM TO INSERT GEOMETRIC DETAILS INTO HEXAHEDRAL MESHES

Nicolas Le Goff, Franck Ledoux, Jean-Christophe Weill

*CEA, DAM, DIF, F-91297 Arpajon, France
e-mail: nicolas.le-goff@cea.fr, franck.ledoux@cea.fr, jean-christophe.weill@cea.fr

Key words: Parametric studies in Numerical Simulation, Hexahedral Meshes, Adaptive model, Sheet operations, Mesh Imprint

Abstract. In numerous computational engineering applications, hexahedral meshes may be preferred over tetrahedral meshes. However, automatic hexahedral meshing remains an unsolved issue and thus generating a hexahedral mesh is known as a time-consuming stage that requires a lot of user interactions in the simulation process. A possible way for designing and optimizing a CAD model or a geometric shape requires parametric studies where the shape is enriched by inserting geometric details into it. Then we must "adapt" the initial mesh and not generate it anew for each new detail taken into account. In order to perform such studies with hexahedral meshes, we provide an imprinting method allowing us to automatically add geometric details into an existing mesh. This addition is done using geometric projections, sheets (layers of hexahedral elements) insertions and combinatorial algorithms while preserving the hexahedral mesh structure as best as possible.

1 INTRODUCTION

The definition of a real mechanical piece using only numerical modeling and simulation has been increasingly used for several years. A lot of research efforts have been put into the quality control of the numerical solutions and into the design of sophisticated, complex and coupled modeling, which leads to increasingly time-consuming computations. Most of these simulations rely on the finite element method (FEM) or the finite volume method (FVM). Both of them require that the geometric model be discretized by a mesh. In most cases, they are purely tetrahedral or hexahedral, that is to say exclusively composed of tetrahedral elements or hexahedral elements. In this work, we focus on the generation of hexahedral meshes, and more precisely on the adaptation of an existing hexahedral mesh to fit new geometric features that are inserted into a CAD model during an adaptive simulation process.

The classical process for designing and optimizing a geometric shape requires parametric studies where the shape is modified and/or enriched by adding geometric details. Considering a first shape with an associated mesh, we want to "adapt" the initial mesh and not to regenerate it from scratch for each new part taken into account (see Fig. 1). In order to perform such studies with hexahedral meshes, we provide an imprinting method that allows us to automatically add geometric details into a hexahedral mesh. This addition is done using geometric projections, sheets (layers of hexahedral elements) insertions and combinatorial algorithms, while preserving the hexahedral mesh structure as best as possible. Some authors have studied the insertion of complex geometric models into an existing grid or octree structure in order to get the initial mesh [8, 12, 4, 9, 5, 6, 13]. In our work, we focus on CAD models where sharp features are numerous and must be preserved; corners and ridges are typically difficult to capture in an existing mesh. The main contributions of our work are:

- Contrary to existing algorithms [8, 12, 4, 9, 5, 6, 13], our method can be applied onto any unstructured hexahedral meshes, it is not restricted to grids or octrees;
- While these algorithms only use a grid or octree to discretize the inner volume of one or several geometrical domains, we discretize both the inner and outer volumes;
- Both the initial geometric domain and the geometric details to be inserted have several corners and ridges.

The remainder of this paper is organized as follows: Section 2 gives an overview of our algorithm while introducing necessary terminology. Section 3 discusses the detailed algorithm for properly capturing the new geometric entities into the mesh. Section 4 explains how to improve the mesh quality in the vicinity of the inserted details and to improve the robustness of our algorithm. Section 5 draws conclusions and outlines future works.

2 MAIN STEPS OF THE IMPRINTING ALGORITHM

2.1 Background notions

A traditional representation [3] of a hexahedral mesh is to consider a 4-tuple (H, F, E, N) where H is a non-empty set of hexahedra, F is the non-empty set of all quadrilaterals adjacent to one or more hexahedra in H, E is the non-empty set of all edges adjacent to one or more hexahedra in H and N is the non-empty set of all nodes adjacent to one or more hexahedra in H. Hexahedra are 3-dimensional cells, or 3-cells, quadrilaterals are 2-cells, edges are 1-cells and nodes are 0-cells. In this work, the geometric domain Ω that we want to discretize is a 3-dimensional geometric object represented by its boundary. It is thus a BRep object described as a 3-tuple (S,C,V) [2] where S is a non-empty set of geometric surfaces enclosing a 3-dimensional space and such that $\forall(s_1, s_2) \in S^2,\ s_1 \cap s_2 = \emptyset$, C is the non-empty set of curves adjacent to one or more surfaces in S and V is the non-empty set of vertices adjacent to one or more surfaces in S.

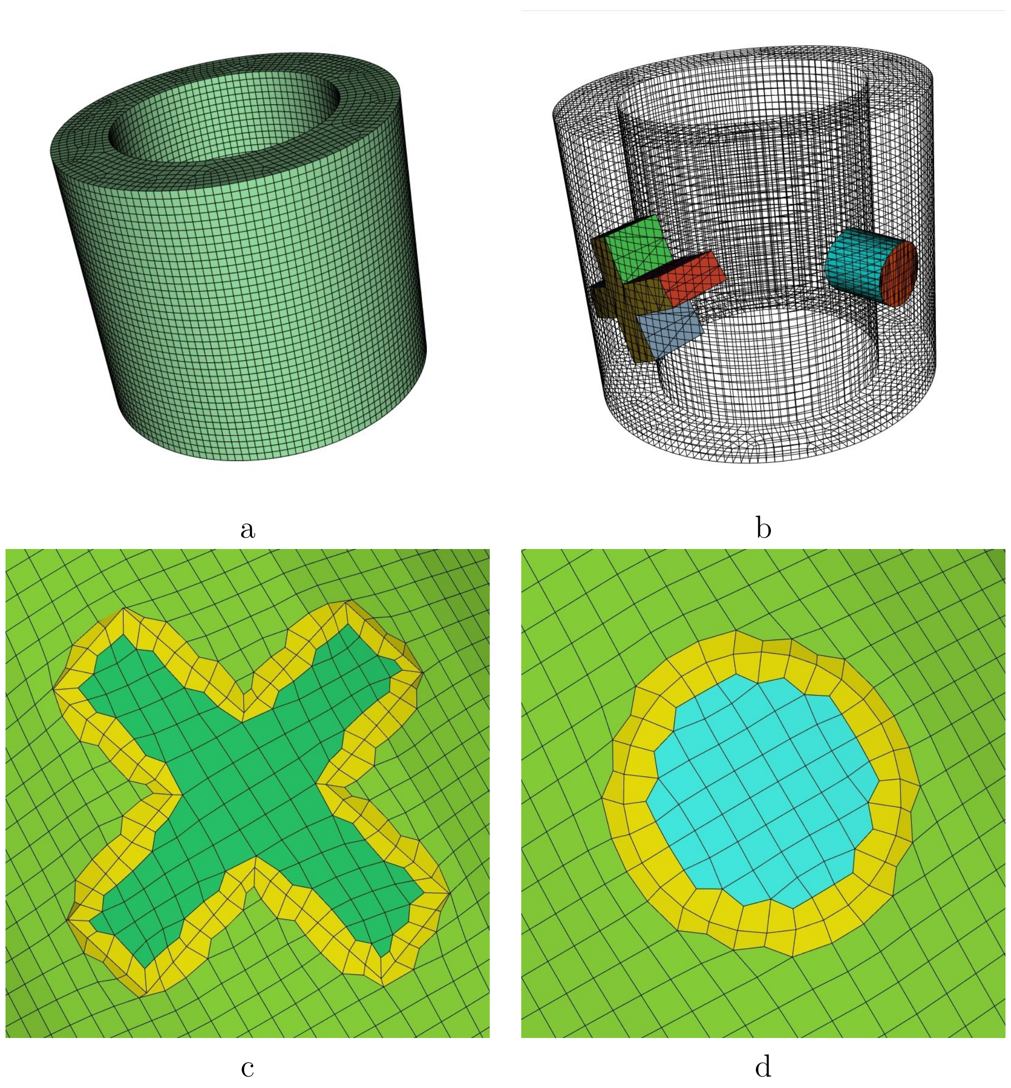

Figure 1: A hexahedral mesh is modified in order to add some geometrical details that can be relevant for the numerical study or to get a more geometric-sharp model. In (a), the first mesh was obtained using a sweeping algorithm [1]. In (b), two shapes, a cylinder and a cross shapes are added. In (c) and (d), close-up of the imprint on the side of the original mesh resulting from the insertion of respectively the cross and the cylindre shapes.

Let $M = (H, F, E, N)$ be a hexahedral mesh discretizing[1] the BRep object $G = (S, C, V)$. In order to initialize boundary conditions for FEM and FVM methods, it is mandatory to associate[2] each i-cell to a j-dimensional geometric entity with $j \geq i$. To get a valid association, some constraints must be satisfied:

- A mesh surface $s_M \subseteq F$, i.e. a set of pairwise adjacent faces of F forming a 2-

[1]The notion of discretization is not detailed in this paper. See for instance [3].

[2]This association is similar to the classification notion introduced by Remacle and Shephard in [7].

manifold, must be associated to each geometric surface $s \in S$. It means that all the faces in s_M, all the edges and nodes adjacent to a face of s_M are geometrically on surface s within a tolerance, and s_M discretizes surface s (i.e. every point $x \in s$ is contained in exactly one face, $f \in s_M$, and s_M wholly fills s.

- A mesh line $l_M \subseteq E$, i.e. a set of pairwise adjacent edges of E forming a 1-manifold, must be associated to each geometric curve $c \in C$. It means that all the edges in l_M and all the nodes adjacent to an edge of l_M are geometrically on curve c within a tolerance, and l_M discretizes curve c (i.e. every point $x \in c$ is contained in exactly one edge, $e \in l_M$, and l_M wholly fills c.

Implicitly, it means that if two geometric surfaces s_1 and s_2 of a BRep object share a curve c then the edges of the mesh line associated to c are also associated to s_1 and s_2 and such a line of edges separates the two sets of faces associated to s_1 and s_2.

2.2 Overview of the algorithm

Starting from a hexahedral mesh $M = (H, F, E, N)$ that discretizes a BRep geometric object $G = (S, C, V)$, the aim of our algorithm is to adapt M in order to discretize both G and G_2, where $G_2 = (S_2, C_2, V_2)$ is a new geometric object fully enclosed into G. The global process of our method is the following one:

1. Cells of H are split into two sets: those inside G_2, denoted H_2, and those outside; cells that are intersected by the geometric object will either be classified as inside or outside depending on a few criteria(see Section 3.1). Some refinement patterns can be applied to ensure the right topology of H_2 (see Section 4.1);

2. Each vertex of V_2 is captured by a node located on the boundary of H_2 (see Section 3.2);

3. Each curve of C_2 is captured by a line composed of edges of E located on the boundary of H_2 of which the endnodes capture the endpoints of c (see Section 3.3);

4. Each surface s of S_2 is captured by a mesh surface composed of faces of F located on the boundary of H_2 and delimited by mesh lines capturing the bounding curves of s (see Section 3.4);

5. Layers of hexahedra are inserted along the boundary of H_2 in order to improve the quality of elements (see Section 4.2).

3 CAPTURING GEOMETRIC ENTITIES INTO A HEXAHEDRAL MESH

3.1 Extraction of inner cells

The first step consists in choosing which hexahedra of the original mesh M will be considered as being part of the inserted geometric entity G_2. Thanks to the fact that G_2

is fully enclosed into G, hexahedra are divided into two categories: inside or outside the geometric detail, and the set of faces of M delimiting the two areas will be considered as the discrete boundary of G_2 in M.

- We first identify the hexahedra intersected by G_2 and mark the remaining cells as either inside or outside. Let H_2 be the set composed of intersected and inner hexahedra;
- The intersected hexahedra of H_2 will then be classified as inside or outside depending on whether more or less than half of their volume is located inside the geometric detail; those classified as outside are removed from H_2. This is done by, for each hexahedron h of H_2, taking a set of points S_h located inside the cell and determining if most of them reside within G_2 or not. For each point P in S_h the projected point P_{S_2} on the surfaces S_2 is computed, then the sign of the scalar product between $\mathbf{P_{S_2}P}$ and the outward normal to the surface at P_{S_2} determines whether the point is inside or outside (negative is inside, positive is outside).
  Currently we take an arbitrary number of 27 points located inside each intersected hexahedron using trilinear interpolation; Gauss points or some other quadrature rule could be used.

At the end of this step, the hexahedra of M are separated into two sets: those inside G_2 and those outside. In the following steps, our algorithm is restricted to selecting boundary nodes, edges and faces among the discrete boundary of G_2 in M.

3.2 Vertices' classification

A boundary node $n \in N$ will be associated to each vertex $v \in V_2$ considering a distance criterion, meaning the nearest node of N will be chosen for each vertex v of V_2. A node cannot be associated to more than one vertex, and in case of conflict, for example if two vertices both have the same nearest node, vertices' classification is done on a first-come, first-served principle.

3.3 Curves' classification

Curves' classification is done in two steps.

First for each vertex $v \in V_2$ we associate an edge to every curve adjacent to said vertex (see Fig. 2-a-b). Let C_v be the list of curves adjacent to v, ordered around v in a direct order. Let n be the node associated to v and E_n be the set of boundary edges of M adjacent to n. We are looking for the list of ordered edges $L_n \subseteq E_n$, ordered around n in a direct order, that best matches C_v. We define such a list as the list of ordered edges that maximizes the cost function:

$$f(L_n) = \sum_{i=1}^{|C_v|} \mathbf{C_v}[i].\mathbf{L_n}[i]$$

where $\mathbf{C_v}[i]$ and $\mathbf{L_n}[i]$ are the vectors of respectively the ith curve/edge of C_v/L_n pointing outward from v/n. This phase is not mandatory but it allows us to select a better solution near the vertices, which is typically where a good selection will improve robustness by avoiding crossing between lines of selected edges; this non-crossing property is mandatory for the algorithm used during the surfaces' classification phase.

The second step builds the remainder of the lines for each curve $c \in C_2$ (see Fig. 2-c-d), by starting from one of the curve's endpoints and building a contiguous line of boundary edges to reach the other endpoint. A set of selectable edges is computed as the boundary edges part of every hexahedron intersected by c, and a shortest path algorithm is used where each edge is weighted by its Hausdorff distance to the curve [10]. This way, we extract a suitable line of edges. Let us note that we do not start and end at the endpoints of c, but rather we start from the first edges associated to c at its endpoints during the previous step.

In case of curves that do not have endpoints (circles for example in the cylindrical shape inserted, see Fig. 2-f) we arbitrarily put a few points on the curve and associate them to boundary nodes , then build lines of edges that connect all those nodes using the same method as described above, i.e. a weighted shortest path algorithm applied on a restricted set of edges.

3.4 Surfaces' classification

Surfaces' classification is fairly straightforward once the edges' lines have been determined. Sets of faces are delimited by the lines, and for each set of faces s_M delimited by a set of lines L_M the corresponding surface $s \in S_2$ is the surface delimited by the curves the lines in L_M are associated to (see Fig. 2-e). This is sufficient to characterize all the surfaces of S_2 but in two cases:

- When there are no curves, for example if the geometric detail is a sphere, there is only one surface s in G_2 which is then associated to s_M;
- When there are only two surfaces, hence delimited by the same set of curves we have to choose an order of traversal for the curves and lines of edges and discriminate between the two surfaces by determining which surface is on the left or on the right.

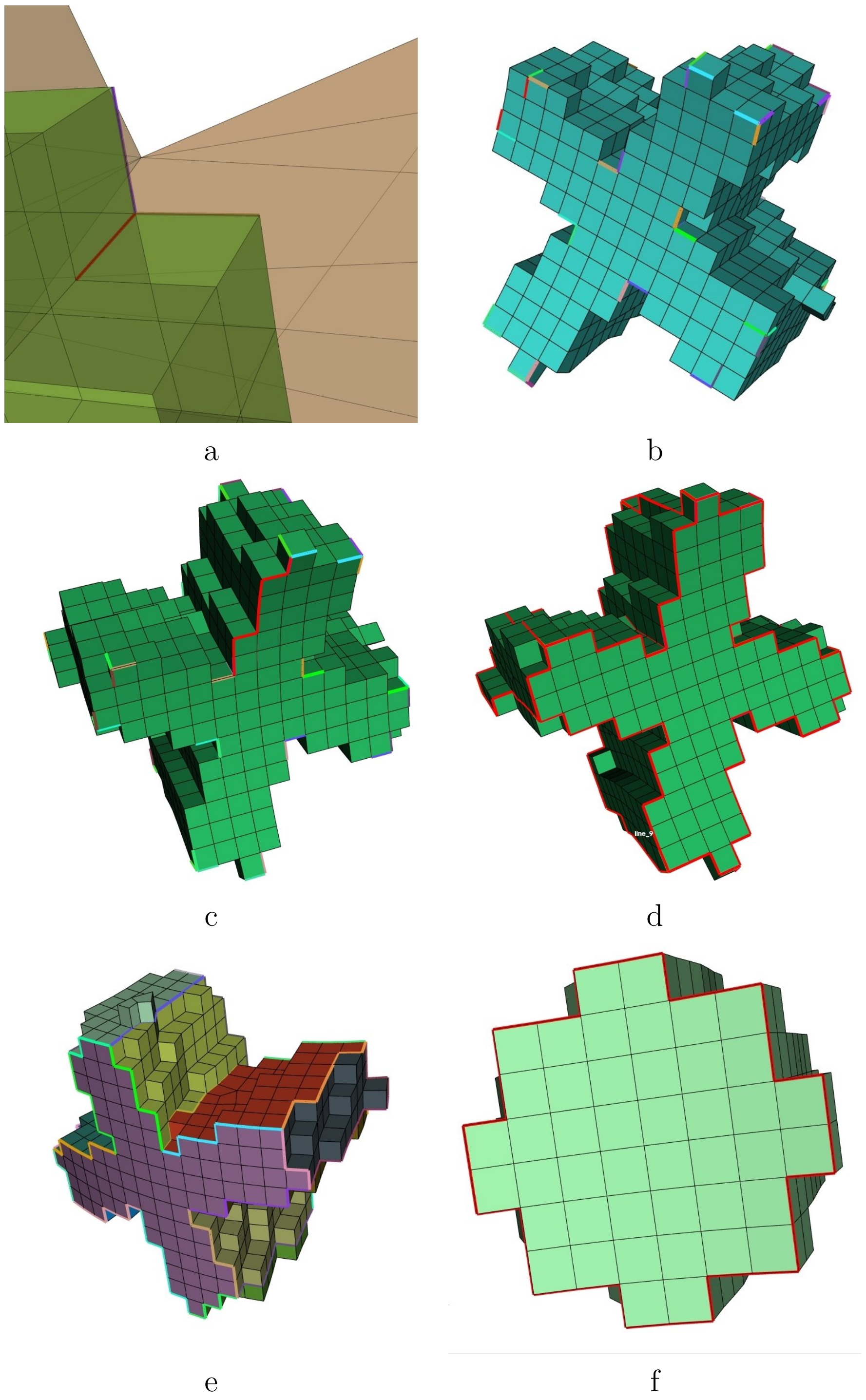

Figure 2: Curves' classification, from a best combination around each vertex to building a line of edges. In (a), close-up of the combination of edges that best matches the 3 curves of the cross shape at this vertex. In (b), edges at every vertex have been associated to curves. In (c), a line of edges (in red) has been associated to a curve. In (d), every curve in the cross shape has been associated to a line of edges. In (e), surfaces are classified to sets of faces. In (f), a curve of the cynlindrical shape is associated to a line of edges despite having no endpoints.

4 ROBUSTNESS AND QUALITY IMPROVEMENT

At this point in the paper the main contributions of our work have been outlined; the geometric detail G_2 has been inserted into the initial mesh and its surfaces, curves and vertices have been associated to mesh entities, but in order to be more robust and obtain a resulting mesh of better quality our algorithm needs to apply the following steps:

4.1 Refinement

The quality and robustness of the geometric detail classification strongly depends on the initial mesh. We use a 3-refinement strategy similar to the refinement used in [14] in order to get a valid result at the end of the first step of our algorithm. Indeed, such a refinement ensure that the topology of the set of inner hexahedra of H associated to G_2 will be the same as the topology of $G2$. In Fig. 3 the mesh is refined in the thin areas using a criterion based on whether at least two non-neighbor surfaces intersect a hexahedron. That allows the algorithm to better capture the thin top and bottom parts, and to disjoin the two parts on the right side of the model. Otherwise depending on the position and the size of the hexahedra near the thin space on the right of the model the space would not be captured, meaning the selected hexahedra in this area would form one block instead of two, and the corresponding surfaces and curves would not be classified.

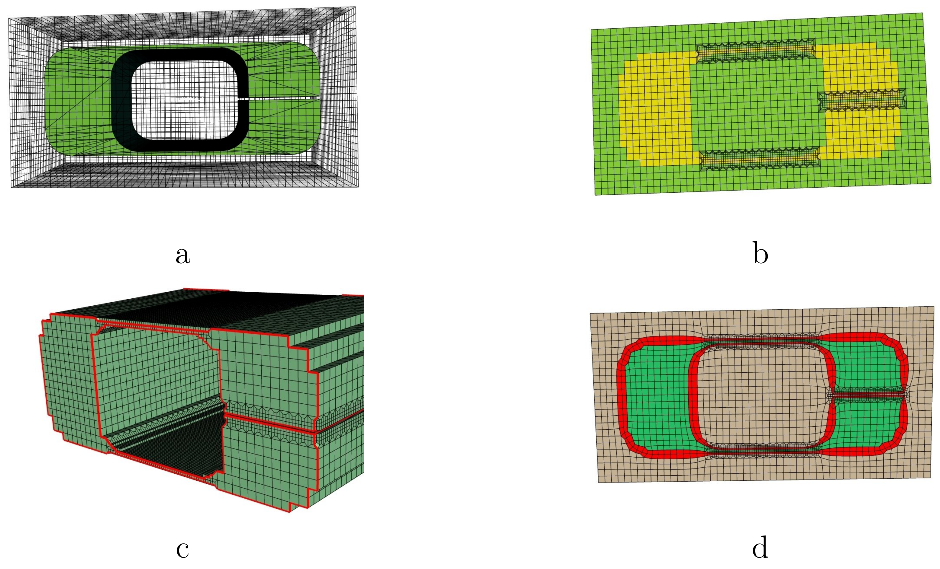

Figure 3: A hexahedral mesh is refined in order to facilitate the geometric detail insertion. In (a), the geometric detail is represented inside the original mesh, which is a regular cartesian grid. In (b), three areas are refined, around the thin parts of the geometric detail; the inside hexahedra are represented in yellow, the ouside ones in green. In (c), after curves' classification. In (d), after sheets insertions, sheets represented in red.

4.2 Sheet insertion

After having classified the vertices, curves and surfaces of the geometric detail G_2, sheets can be inserted in order to offer good quality elements near the surface of the geometric detail, and to provide boundary-aligned elements in case the numerical simulation favours such a feature in a mesh (see Fig. 4). Depending on the requirements on the resulting mesh, fundamental sheets and chords [2] can be inserted.

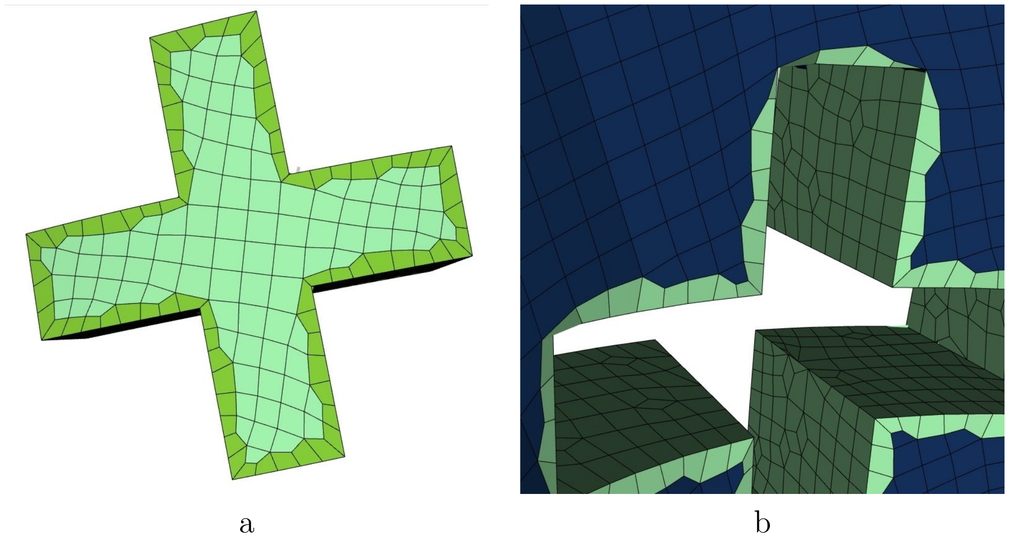

Figure 4: Sheet insertions after classification of the cross shape inside the original mesh. In (a), one sheet was inserted around the cross shape, inside the geometric detail. In (b), a sheet was inserted around the shape but this time located on the side of the outer volume.

4.3 Mesh smoothing

A laplacian smoothing constrained by the geometric classification was applied to the examples shown in this paper. But to get better quality, it seems mandatory to apply more evolved algorithms. Indeed, as we insert sharp geometric objects, non convex areas with sharp ridges appear. In such areas,algorithms merging untangling technics and geometric smoothing should be used [11]. In order to select a suitable method, we need to further study the impact of the geometric constraint on the smoothing method.

5 CONCLUSIONS

In this work we introduced a method to insert a geometric detail into an existing mesh. The approach consists in selecting an initial good set of hexahedra, so as to simplify the curves' and surfaces' classification steps that could prove overly difficult otherwise. This is a strictly *a priori* selection, and slight changes could be applied to the selection, i.e. adding or removing a select few hexahedra in order to improve quality or robustness.

A lot of work remains to be done concerning robustness; for example we have at the moment ignored the possibility that during vertices' classification a vertex v could have more adjacent curves than there are adjacent boundary edges to the nearest node, not to mention any boundary nodes, hence an impossiblity to classify curves. Such an issue could be resolved by refining the mesh around good nodes candidates, thus adding adjacent edges to those nodes. Same wise the hexahedra selection must form a 3-manifold, as that is an essential property for the surfaces' classification step; the geometric criteria that we currently use, i.e. keeping hexahedra which are at least half located inside G_2 is not sufficient and needs to be supplemented with topological criteria. Concerning quality, sheets insertion needs to be further developped in order to adress and correct badly shaped cells that can have several edges or even faces classified on the same curve or surface. This could be done considering [2].

REFERENCES

[1] P. Knupp J. Shepherd, S.A. Mitchell and D.R. White. Methods for multisweep automation. In proceedings of the 9th International Meshing Roundtable, pages 77–87, 2000.

[2] F. Ledoux, N. Le Goff, S. J. Owen, M. L. Staten, and J.-Ch. Weill. A constraint-based system to ensure the preservation of sharp geometric features in hexahedral meshes. In proceedings of the 21st International Meshing Roundtable, pages 315–332. Sandia National Laboratories, September 2012.

[3] F. Ledoux and J. Shepherd. Topological and geometrical properties of hexahedral meshes. Engineering with Computers, 26(4):419–432, 2010.

[4] L. Maréchal. Advances in octree-based all-hexahedral mesh generation: Handling sharp features. In Brett W. Clark, editor, proceedings of the 18th International Meshing Roundtable, pages 65–84. Springer, 2009.

[5] S. J. Owen and J. F. Shepherd. Embedding features in a cartesian grid. In proceedings of the 18th International Meshing Roundtable, pages 116–138. Sandia National Laboratories, October 2009.

[6] J. Qian and Y.e Zhang. Sharp feature preservation in octree-based hexahedral mesh generation for cad assembly models. In proceedings of the 19th International Meshing Roundtable, pages 243–262. Sandia National Laboratories, October 2010.

[7] J.-F. Remacle and M.S. Shephard. An algorithm oriented mesh database. International Journal for Numerical Methods in Engineering, 58(2), 2003.

[8] R. Schneiders. An algorithm for the generation of hexahedral element meshes based on a octree technique. In proceedings of the 6th International Meshing Roundtable, pages 183–194, 1997.

[9] J. F. Shepherd. Conforming hexahedral mesh generation via geometric capture methods. In Brett W. Clark, editor, proceedings of the 18th International Meshing Roundtable, pages 85–102. Springer, 2009.

[10] R. Shonkwiler. Computing the hausdorff set distance in linear time for any lp point distance. Information Processing Letters, 38:201–207, 1991.

[11] T.J. Wilson, J. Sarrate, X. Roca, R. Montenegro, and J.M Escobar. Untangling and smoothing of quadrilateral and hexahedral meshes. In B.H.V Topping, editor, Proceedings of the Eighth International Conference on Engineering Computational Technology. Civil-Comp Press, Stirlingshire, UK, Paper 36, 2012.

[12] K. H. Lee Y. Su and A. Senthil Kumar. Automatic hexahedral mesh generation for multi-domain composite models using a hybrid projective grid-based method. Computer-Aided Design, 36:203–215, 2004.

[13] T. J. R. Hughes Y. Zhang and C. L. Bajaj. An automatic 3d mesh generation method for domains with multiple materials. Computer Methods in Applied Mechanics and Engineering, 199(5-8):405–415, January 2010.

[14] Y. Zhang and C. Bajaj. Adaptive and quality quadrilateral/hexahedral meshing from volumetric data. In proceedings of the 13th International Meshing Roundtable, pages 365–376. Sandia National Laboratories, September 2004.