A Constraint-Based System to Ensure the Preservation of Sharp Geometric Features in Hexahedral Meshes

Franck Ledoux[1], Nicolas Le Goff[1], Steve J. Owen[2], Matthew L. Staten[2], and Jean-Christophe Weill[1]

[1] CEA, DAM, DIF, 91920 Arpajon, France
{nicolas.le-goff,franck.ledoux,jean-christophe.weill}@cea.fr
[2] Sandia National Laboratories, Albuquerque, USA
{mlstate,sjowen}@sandia.gov

Summary. Generating a full hexahedral mesh for any 3D geometric domain is still a challenging problem. Among the different attempts, the octree-based methods are the most efficient from an engineering point of view. But the main drawback of such methods is the lack of control near the boundary. In this work, we propose an *a posteriori* technique based on the notion of the fundamental mesh in order to improve the mesh quality near the boundary. This approach is based on the resolution of a constraint problem defined on the topology of the CAD model that we have to discretize.

Keywords: hexahedral mesh modification, topology of CAD models, fundamental sheets, sheet insertion, constraint-based system.

1 Introduction

Computational simulations depend upon the numerical approximation method used, such as finite element methods or finite volume methods. Such methods rely on a discretization of the physical domain into a “mesh”. Depending on the numerical approximation methods, hexahedral meshes are preferred to tetrahedral meshes due to their layered structure that can be aligned along the boundary of the 3D geometric domain to be meshed. While this structure is an important feature of hexahedral meshes [13], it is also the origin of the issues in writing generic and robust meshing algorithms.

The automatic generation of hexahedral meshes for any geometric 3D object is still an open problem. In recent decades, different approaches have been followed to generate a full hexahedral mesh for any geometric 3D object. Starting from a pre-meshed boundary surface, pure geometric approaches like plastering [2] or H-morph [15] algorithms have been proposed, while some other works have been based on pure topological approaches [20, 3, 8].

In both cases, the success was limited: some unfilled cavities remain or inverted cells and negative Jacobian cells can be generated inside the mesh. By relaxing the constraint of working with a pre-meshed boundary, inside-out algorithms [9, 18] or unconstrained plastering [19] have provided encouraging examples of all-hex meshes for arbitrary geometries. For instance, inside-out algorithms do not consider the global topological structure and generally put the worst quality elements near the boundaries while the quality of elements in these areas can be very important for numerical approximation methods.

Among the different families of hexahedral meshing algorithms, inside-out or octree-based algorithms [9] are certainly the most efficient from an engineering point of view. Quite generic and "simple" to implement, they offer an automatic method to get a hexahedral mesh for any 3D geometric domain, however, the mesh quality and structure is very good inside the domain and quite poor near the boundary. In order to improve the quality near the boundary and especially for sharp features, recent works about inside-out algorithms suggest performing a post-processing stage where some sheets are inserted along the boundary to get better shaped hexahedra [18], or local transformation patterns are applied [9]. In this paper, we focus on this post-processing stage where some sheets are inserted inside the mesh to improve the mesh structure along the boundary. To do that our approach is based on the notion of the *fundamental mesh* which is a theoretical classification of the layers or sheets required to capture the model's geometric topology [6, 5]. This classification allows us to link geometric features and global topological constraints in a unique concept known as *fundamental sheets*, or *fun sheets* for short. It is also particularly appropriate for CAD models where numerous sharp features (corners and ridges) are present. The process of inside-out or octree-based algorithms preserving sharp features of a CAD model[14, 9, 16] can be split into two main steps:

1. considering a geometric domain Ω and an initial hexahedral mesh M, the cells of M that are inside $\partial\Omega$ (totally or partially depending on the algorithm) are kept[1]. The boundary nodes, edges and faces are then classified onto $\partial\Omega$ to fullfill Ω.
2. Some topological operations on M are performed in order to improve the quality of the hexahedral elements near $\partial\Omega$.

For this type of algorithms, the robustness of the first step remains to be proved, while the second step is done applying heuristics based on geometric criteria. In this work, we suggest a new way for this second step: our method consists in solving an integer constraint-based system to define fundamental sheets to insert and then to insert those into M.

In this paper, and mainly in Section 3, we focus on the resolution of the constraint-based system. The formulation of the problem we give allows us to ensure an automatic, reliable and terminating process. We also explain how

[1] Some mesh refinement processes can be performed to capture small geometric features.

we define the sheets to insert in Section 3. In Section 4, some examples are given before concluding. But in order to present this work some basic mesh structure definitions and the notions of fundamental sheets and fundamental mesh are mandatory. They are given in Section 2.

2 Background and Theoretical Mesh Foundations

Unlike tetrahedral meshes, hexahedral meshes have an inherent layered structure, which makes both local modification [1, 4, 7] and automatic generation of all-hexahedral meshes difficult. This structure is formally defined as the dual, or the Spatial Twist Continuum [11]. It is the main cause of robustness issues with previous attempts at all-hexahedral meshing which rely on a pre-meshed boundary[2]. As a consequence, the development of a reliable all-hexahedral meshing algorithm for arbitrary geometries can not be done without taking this structure into account. But it must also be "connected" to the geometric features of the geometric domain that is to be meshed. In order to formalize this connection, the notion of the fundamental mesh was introduced in [6, 5]. The work of this paper relies on this notion that we review in this section.

2.1 Topological Definitions

A hexahedral mesh is composed of sets of hexahedra, quadrilaterals, edges and nodes, that are connected to form a strong topological structure. The dual of this structure can be modeled as a simple arrangement of surfaces[21] where:

- each surface corresponds to a layer of hexahedra, or *primal sheets*,
- the intersection of two surfaces is a column of hexahedra, or a *primal*[3] *chord*,
- the intersection of three surfaces is a single hexahedron.

Note that self-intersecting sheets are of course possible. To illustrate these notions, let us consider Fig. 1-a, where two primal sheets intersect each other. Their intersection forms a primal chord, which is is exhibited on Fig. 1-b.

2.2 Mesh Classification onto Geometric Models

Meshes are used to discretize a geometric domain Ω. The notion of "discretization" requires that any geometric point in Ω belongs to only one mesh cell and the mesh wholly fills Ω. But in order to perform FEM analysis onto

[2] Under the acceptable conditions of having a volume isomorphic to a ball and an even number of quadrilaterals on the boundary.

[3] The term *primal* is used in opposition to the term *dual*, *sheets* and *chords* being originally terms used in the dual of hexahedral meshes.

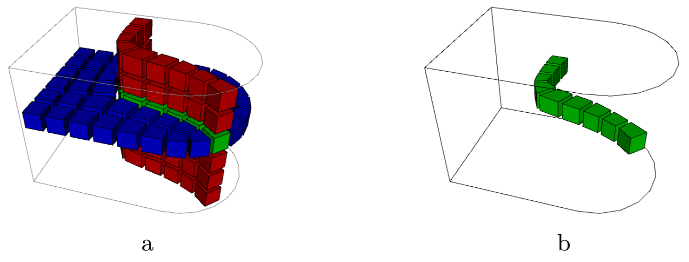


Fig. 1. In a, two primal sheets (respectively in red and blue) intersect each other along a chord (in green) that is shown alone in b

CAD models it is necessary to distinguish the nodes, edges and faces that are on $\partial\Omega$. To do that, we need to handle *mesh surfaces* and *mesh lines*. Let M be a mesh discretizing a bounded domain Ω, a mesh surface of M is a set of pairwise adjacent faces forming a 2-manifold, and a mesh line of M is a set of pairwise adjacent edges forming a 1-manifold.

In order to represent the 3D bounded domains we want to discretize, a boundary representation is mostly used. A 3-dimensional geometric object is then a 3-tuple (S,C,V) where

1. S is a non-empty set of geometric surfaces enclosing a 3-dimensional space and such that $\forall(s_1, s_2) \in S^2,\ s_1 \cap s_2 = \emptyset$;
2. C is the non-empty set of curves adjacent to one or more surfaces in S;
3. V is the non-empty set of vertices adjacent to one or more surfaces in S.

We can now associate geometric surfaces and geometric curves respectively to mesh surfaces and mesh lines. This association is an extension of the classification notion introduced in [17].

Definition 1 (Geometric association). *Let M be a hexahedral mesh discretizing the BRep geometric object $G = (S, C, V)$.*

- *A mesh surface s_M is associated to a geometric surface $s \in S$* iff *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 is associated to a geometric curve $c \in C$* iff *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, this definition indicates that if two geometric surfaces s_1 and s_2 of a BRep geometric object share a curve c, then the edges of the mesh line associated to curve c are both associated to surfaces s_1 and s_2 too.

2.3 Fundamental Hexahedral Meshes

Fundamental meshes characterize what it means for a mesh to have well-shaped hexahedra near the boundary.

Definition 2 (Fundamental primal chord). *Let M be a hexahedral mesh of a 3D geometric object $G = (S, C, V)$, let $c_G \in C$ be a geometric curve of G and $(s_{G_1}, s_{G_2}) \in S^2$ be the adjacent surfaces of c_G. Let F_{G_1} and F_{G_2} be the sets of faces adjacent to c_G and belonging to s_{G_1} and s_{G_2} respectively, let H_c be the set of all the hexahedra sharing a face of F_{G_1} (respectively F_{G_2}), a primal chord c is a fundamental primal chord of c_G associated to s_{G_1} (respectively s_{G_2})* iff *H_c is included into c and c is simply connected locally to curve c_G.*

This definition ensures that if you consider a geometric curve c_G delimiting two geometric surfaces, then, on both surfaces, quadrilaterals adjacent to curve c_G belong to a single primal chord (see Fig. 2). This definition specifies nothing about the hexahedra having an edge associated to curve c_G but no faces on the geometric boundary. Indeed, any number of chords can be fundamental to the same geometric curve. In order to define the notion of fundamental sheets, we now introduce the definition of geometric surfaces capture.

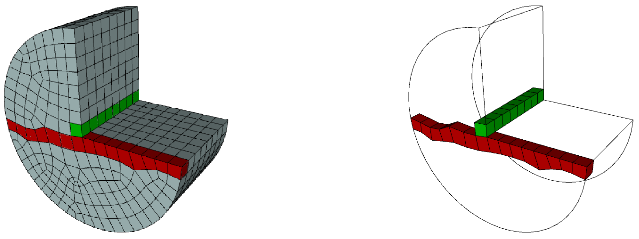

a b

Fig. 2. Two examples of fundamental chords

Definition 3 (Primal sheet capturing a geometric surface). *Let M be a hexahedral mesh of a 3D geometric object $G = (S, C, V)$, let $s_G \in S$ be a geometric surface of G and s_M be the quadrilateral surface mesh associated to s_G, let H_s be the set of all the hexahedra incident to a face of s_M, a primal sheet P is a sheet capturing s_G* iff

1. *$H_s \subseteq P$ and,*
2. *P is simply connected locally to surface s_G.*

The first condition ensures that the geometric surface is captured by a single primal sheet. For instance, let us consider Fig. 3-a, all the hexahedra belonging to the green primal sheet are along a single curved surface. On the contrary, in Fig. 3-b, this surface is partially captured by three primal sheets.

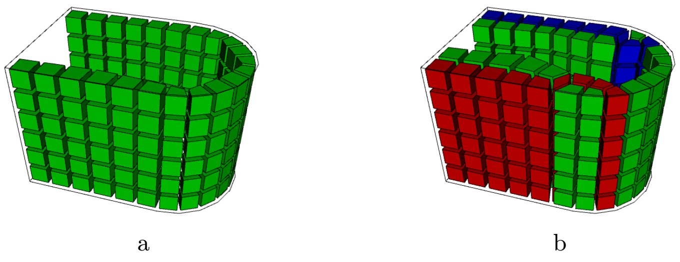


Fig. 3. The same geometric 3D domain where a curved surface is captured by a single fundamental sheet in a and three non-fundamental sheets in b

The second condition guarantees that the corresponding dual sheet is locally a 2-manifold.

Using Definitions 2 and 3, we will distinguish three types of fundamental sheets.

Definition 4 (fundamental sheets). *Let M be a hexahedral mesh of a geometric object $G = (S, C, V)$, $s_G \in S$ be a geometric surface of G, s_M be the quadrilateral surface mesh corresponding to s_G, H_s be the set of all the hexahedra sharing a face of s_M and $c_G \in G$ be a geometric curve of G. Then a primal sheet P is a:*

- **level 1 fundamental** *sheet for s_G* iff *P captures s_G and every hexahedron $h \in P - H_s$ participates to capture another geometric surface of G;*
- **level 2 fundamental** *sheet for s_G* iff *P captures s_G and P is not a level 1 fundamental sheet;*
- **level 3 fundamental** *sheet for c_G* iff
  - *There exists a fundamental chord c and a level 1 or level 2 fundamental sheet P_2 such that c is the intersection of P and P_2,*
  - *P is neither a level 1 nor a level 2 fundamental sheet.*

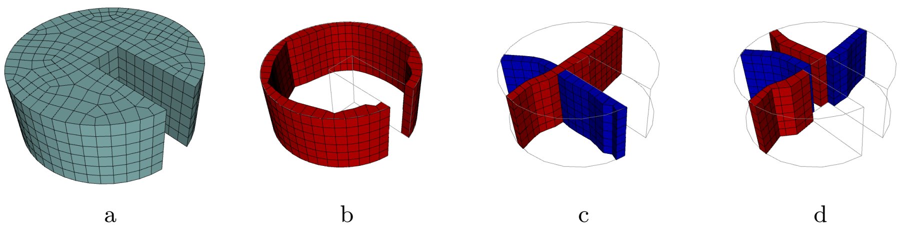


Fig. 4. A hexahedral mesh (in a) and one level 1 fundamental sheet (in b), two level 2 fundamental sheets (in c) and four level 3 fundamental sheets (in d)

Fig. 4 shows examples of fundamental sheets. In Fig. 4-a, the hexahedral mesh is given. In Fig. 4-b, a level 1 fundamental sheet captures the cylindrical surface of the geometric model. In Fig. 4-c, two level 2 fundamental sheets participate to capture geometric surfaces before diving into the mesh volume. In Fig. 4-d, four level 3 fundamental sheets are shown. They participate, by pair of one red sheet and one blue sheet, in capturing geometric curves. Fig. 5 provides the dual representation corresponding to the fundamental primal sheets of Fig. 4.

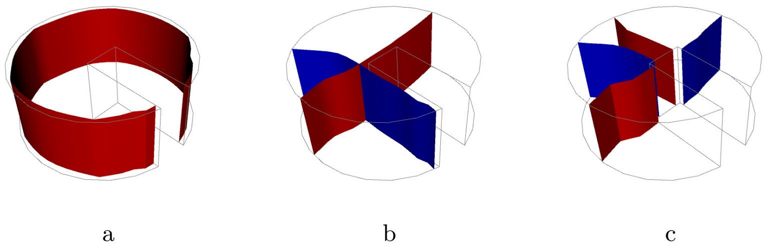


Fig. 5. Dual representation of the fundamental sheets given in Fig. 4

Considering the three levels of fundamental sheets defined previously, a mesh of a 3D geometric object $G = (S, C, V)$ is a fundamental mesh of G *iff*

1. All the geometric surfaces in S are captured by a level 1 or level 2 fundamental sheet in M;
2. All the geometric curves in S are captured by level 3 fun sheets in M.

3 A Constraint-Based System to Get Fundamental Meshes

Considering a hexahedral mesh M discretizing a geometric model $G = (S, C, V)$, our aim is to transform M into a fundamental mesh of G. The definition of fundamental meshes being based on the existence of fundamental sheets, the mesh transformation is performed by sheet insertions [10]. In order to determinate which sheets to insert, we consider the geometric curves and their incidence relationships to geometric vertices and we solve a constraint system to label every geometric curve. The label associated to a curve c indicates a configuration of several fundamental sheets that participate to locally capture c. Once a label is associated to each curve, we automatically deduce the sets of level 1, level 2 and level 3 fundamental sheets to insert.

3.1 Curve Labeling

In this part, we consider the geometric vertices and geometric curves as defining a weighted graph $G = (V^G, E^G)$, where $V^G = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices and $E^G = \{e_1, e_2, \ldots, e_n\}$ is a set of edges. A weight is then associated to each graph edge and will correspond to the labeling of the curves. In this way, solving our problem consists in solving a problem of graph theory.

Curve Labeling Definition

Depending on the number of fundamental chords that capture a geometric curve, we assign a number to every geometric curve of the geometric model. This number can take any strictly positive value, but for geometric meaning it should range from 1 to 4. Fig 6 depicts the four cases we handle focusing on the dual representation of the fundamental sheets. We consider that a curve c can be locally captured by:

1. two level 1 fundamental sheets (see Fig. 6-a),
2. one level 1 fundamental sheet and two level 3 fundamental sheets (see Fig. 6-b),
3. two level 2 fundamental sheets that intersect each others and two level 3 fundamental sheets (see Fig. 6-c),
4. two level 2 fundamental sheets that do not intersect each other and two level 3 fundamental sheets (see Fig. 6-d).

The "locality" term is due to the fact that locally to a curve, it is impossible to distinguish a level 1 from a level 2 fundamental sheet and a level 3 from a level 2 fundamental sheet. Moreover a propagation of constraints along geometric curves and surfaces can alter this classification.

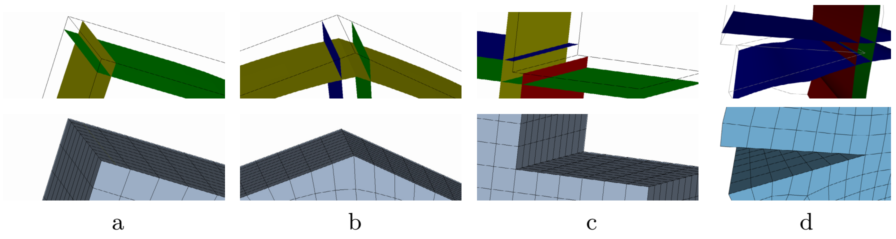


Fig. 6. The four types of curves we handle

Ideally, each geometric curve should be captured by one, two, three or four chords depending on the dihedral angle between its incident geometric surfaces. Considering a dihedral angle $\alpha \in [0, 2\pi]$, the curve should be labeled 1 if $\alpha < 3\pi/4$, 2 if $3\pi/4 \leq \alpha < 3\pi/2$, 3 if $3\pi/2 \leq \alpha < 7\pi/4$ and 4 if $7\pi/4 \leq \alpha < 2\pi$.

However, locally to a vertex some curves' labeling are invalid due to the topological structure of hexahedral meshes. As a consequence we currently handle a limited number of configurations for 3, 4 and 5-valent vertices. Those

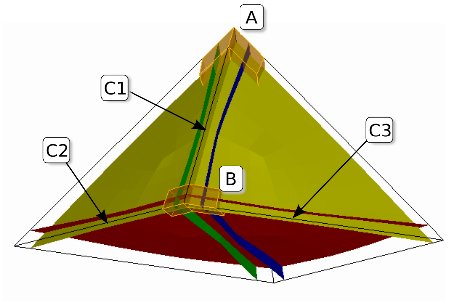


Fig. 7. Propagation of geometric vertices' constraints along geometric curves

configurations are shown in Table 1; for each configuration is mentioned the number of fundamental sheets inserted and the number of hexahedra that capture the geometric vertex. The labeling configurations allowed are local to each vertex, and conflicts can spawn across the geometric model. It is important to understand that the configurations given in Table 1 are those we generate with our approach starting from any hexahedral mesh. Our algorithm can thus be applied to modify any hexahedral mesh.

Hence, labeling the curves of a geometric model G is a global problem. For instance, consider the example of Fig. 7 where are represented four dual fundamental sheets for a square-based pyramid. Since each base corner of this pyramid is a 3-valent geometric vertex with angles between curves lower or equal than $\pi/2$, ideally, it will be meshed with a single hexahedron. It means curves C1, C2 and C3 would have label 1. However, the top vertex of the pyramid is 4-valent, which requires that at least two of the curves connected to the pyramid's top vertex must have label 2. As a result, the initial label of C1 becomes 2 during the solving of the constraint system. It means that the B vertex that ideally would have been captured by three level 1 fundamental sheets (putting one hexahedron at the corner) will instead be captured by one level 1 fundamental sheet, and two level 3 fundamental sheets that will capture curve C1, effectively putting two hexahedra at the corner.

Mathematical Formulation of the Problem

Considering a geometric model $G = (S, C, V)$, assigning a number to each curve of C can be interpreted as solving the following minimization problem:

$$F = \min \sum_{i=1}^{|C|} \lambda_{w_g^i - w_c^i} \tag{1}$$

where for all i in $[1, |C|]$,

- w_c^i is the *decision variable* corresponding to the computed label we will assign to the $i^{\mathtt{th}}$ curve,

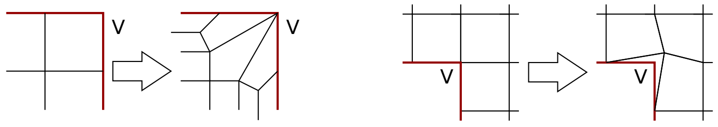


Fig. 8. The labeling of vertex V is increased (left) or decreased by 2 (right)

- w_g^i is the ideal geometric label we would assign to the i^{th} curve without considering the other geometric curves incident to common vertices, and
- $\lambda_{w_g^i - w_c^i}$ is a cost term designed to favor good geometric configurations and discard invalid ones.

This formulation deserves some explanations. The label w_c^i is associated to the i^{th} curve and is chosen so as to comply with allowed labeling configurations at each vertex without any conflicts and should be "close" to the ideal w_g^i; the closeness is defined by the cost term $\lambda_{w_g^i - w_c^i}$. Depending on the sign of $w_g^i - w_c^i$, we may not accept the resulting mesh. For instance, let us consider the 2D case of Fig. 8:

- In a, the ideal labeling for vertex V is 1. If we assign a weight of 3 (a difference of 2), we get 3 quadrilaterals that capture V. The obtained mesh is not the best one but it remains valid.
- In b, the ideal labeling for vertex V is 3 and we compute a weight of 1 (also a difference of 2), we get just one quadrilateral capturing V. This unique quadrilateral is inverted and thus inappropriate for finite element methods.

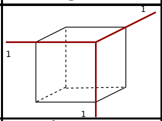
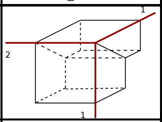
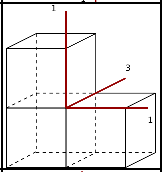
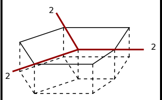
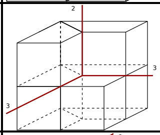
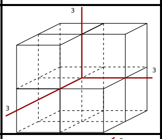
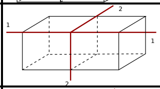
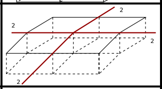
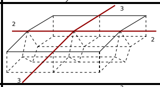
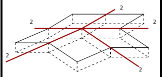
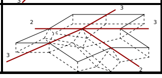
We arbitrarily choose $\lambda_0 = 0$, $\lambda_{-1} = 1$, $\lambda_{-2} = 2$, $\lambda_{-3} = 3$, $\lambda_1 = 1$, $\lambda_2 = 10^3$ and $\lambda_3 = 10^6$. The two latter values are set in order to heavily penalize (and thus avoid when possible) some labeling geometric configurations, namely the case when the number of chords capturing a curve is reduced compared to the ideal labeling. This cost term can be set freely and should reflect the behavior that the user wants and allows.

System Solving

Considering a given geometric model G and a hexahedral mesh discretizing it, we have to solve Problem 1. First we initialize the ideal geometric label w_g^i for every geometric curve. Let c be such a curve, this is achieved by computing the dihedral angle along all the mesh edges discretizing c and keeping the maximal one for each curve. Once this initialization is performed, our solution relies on building a research tree where:

- Each node n of the tree contains an array of weights associated to the curves and an array of booleans describing whether the weight on the corresponding curve has already been constrained or remains free to be modified;

Table 1. Validity rules for 3, 4 and 5-valent geometric vertices

Edge numb.	Configuration Configuration	Level 1 2 3	Hex. Hex.	Edge numb.	Configuration Configuration	Level 1 2 3	Hex. Hex.
1-1-1		3 0 0	1	1-1-2		2 0 2	2
1-1-3		0 2 2	3	2-2-2		1 0 3	3
2-3-3		0 2 2	2	3-3-3		0 3 3	7
1-2-1-2		2 0 2	2	2-2-2-2		1 0 4	4
2-3-2-3		0 2 4	6	2-2-2-2-2		1 0 5	5
2-2-3-2-3		0 2 5	7				

- The weight array at the root node contains the ideal weights w_g^i for all i in $[1, |C|]$, and no curve is constrained. As a consequence the objective function F reaches its minimum, i.e. zero, for the root.
- An internal node at depth k, i.e. neither the root nor a leaf, corresponds to a state where assigned labeling to curves incident to vertices up to the $k^{\mathtt{th}}$ one comply with a local configuration at each vertex and has no conflict with previously constrained curves.
- A leaf corresponds to a valid configuration.

In order to create such a research tree, and thus solve Problem (1), we use a depth-first algorithm where the creation of nodes is as follows. Let n be a node of depth k. That means that the current curves' labeling is valid for nodes V_1 to V_k and that a value v_{min} corresponding to the minimum value computed for F exists. Value v_{min} is equal to the cost term $\lambda_{w_g^i - w_c^i}$ of a valid configuration if a first complete branch to a leaf has been computed, or infinity otherwise. Let C_1, C_2, $\ldots$, C_m be the configurations allowed for vertex V_{k+1}, then n might have a child for each configuration C_i, $1 \leq i \leq n$, with any curve connected to V_{k+1} receiving weights as specified by C_i. In practice, some configurations do not lead to the creation of a child node:

- no child is created when C_i imposes a different weight on an already constrained curve;
- if applying C_i leads to a computed value of F greater than v_{min}, the corresponding child node is not created. It is a simple way to reduce the number of configurations to evaluate, and it allows us to keep the best solution in the meaning of F. This optimization relies on Lemma (1).

Lemma 1. *Considering a research tree as defined previously, the value of F for a node is lower or equal than the value of F for any of its children.*

Proof. Let us consider a node n_j of depth $k+1$ such that its father is the node n_i. As a child node of n_i, the labeling of the edges incident to vertex V_{k+1} in n_j is either the same as in n_i, or it can be different depending on the validity rule that spawned n_j. In the first case we have $F(n_j) = F(n_i)$ and in the second case where E_{ij} is the set of curves that have been renumbered by w_c^l with l in $[1, |E_{ij}|]$ when going from n_i to n_j we have

$$F(n_j) = F(n_i) + \sum_{l=1}^{|E_{ij}|} \lambda_{w_g^l - w_c^l}$$

Since the cost term λ is always positive, we have $F(n_j) \geq F(n_i)$. □

Lemma 2. *The proposed algorithm always returns a valid labeling.*

Proof. A solution can always be computed by assigning all the curves' labeling to 2. Table 1 shows that indeed, for 3, 4 or 5-valence vertices such a solution is valid. This solution consists in inserting a unique level 1 fundamental sheet that encloses all the existing hexahedral cells and captures all the geometric surfaces and several level 3 fundamental sheets to capture curves. □

In order to illustrate this algorithm, let us consider Fig. 9 where a small portion of the research tree is shown. At the root node, the objective function F returns zero since $w_g^j = w_c^j$ for any geometric curve j. We work vertex by vertex; starting from the 4-valent vertex A at the top of the pyramid, we build 5 children (only 4 are represented), each one corresponding to a configuration allowed in Table 1 (while there are only three possible rules appearing in the table, rotation is allowed). We then follow the first branch and choose a configuration for B by first exploring the 1-2-1-2 rule. Such a rule will later bring conflict with configurations around other vertices, so we switch to the 2-2-2-2 branch for solving B. By successively solving constraints at each vertex a solution is found (see the longest branch represented in Fig. 9). Once a first solution is found, we set the value of v_{min} to the value of the objective function F for this solution. Then, we cover the rest of the tree to find a better solution.

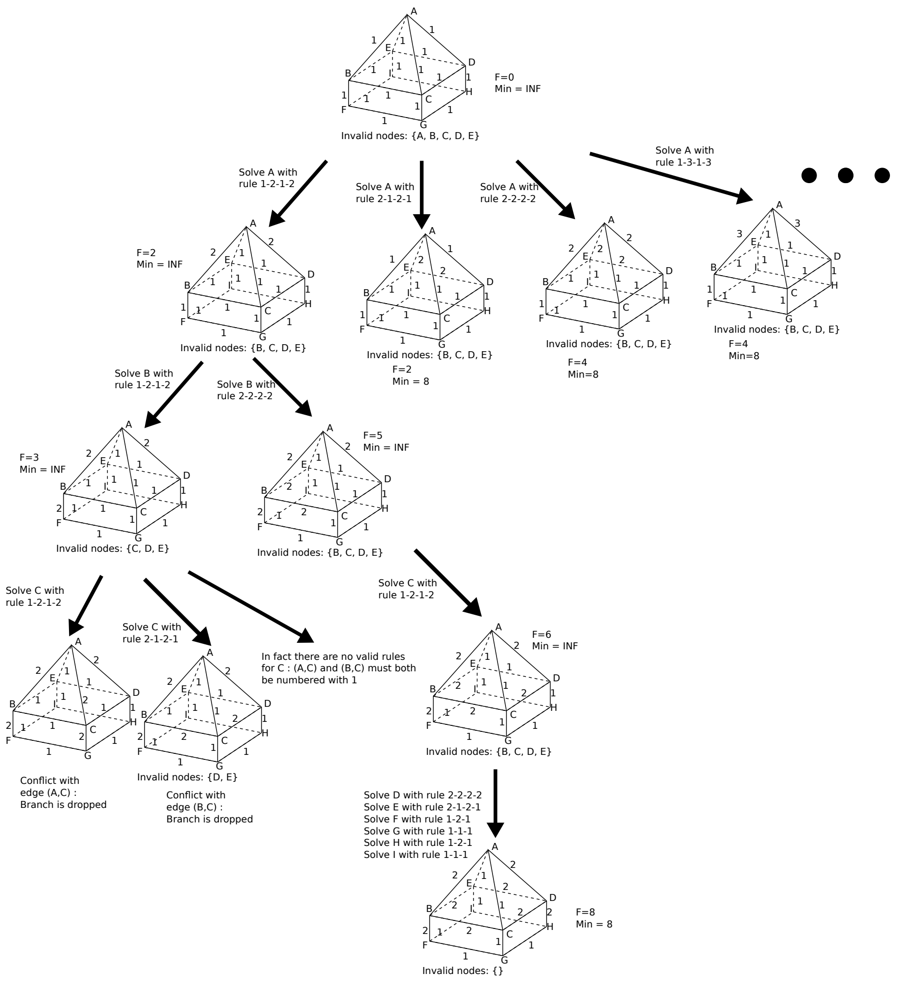

F=0
Min = INF
Invalid nodes: {A, B, C, D, E}
Solve A with rule 1-2-1-2
Solve A with rule 2-1-2-1
Solve A with rule 2-2-2-2
Solve A with rule 1-3-1-3
F=2
Min = INF
Invalid nodes: {B, C, D, E}
Invalid nodes: {B, C, D, E}
F=2
Min = 8
Invalid nodes: {B, C, D, E}
F=4
Min=8
Invalid nodes: {B, C, D, E}
F=4
Min=8
Solve B with rule 1-2-1-2
Solve B with rule 2-2-2-2
F=3
Min = INF
Invalid nodes: {C, D, E}
F=5
Min = INF
Invalid nodes: {B, C, D, E}
Solve C with rule 1-2-1-2
Solve C with rule 2-1-2-1
In fact there are no valid rules for C : (A,C) and (B,C) must both be numbered with 1
Solve C with rule 1-2-1-2
F=6
Min = INF
Invalid nodes: {B, C, D, E}
Conflict with edge (A,C) : Branch is dropped
Invalid nodes: {D, E}
Conflict with edge (B,C) : Branch is dropped
Solve D with rule 2-2-2-2
Solve E with rule 2-1-2-1
Solve F with rule 1-2-1
Solve G with rule 1-1-1
Solve H with rule 1-2-1
Solve I with rule 1-1-1
F=8
Min = 8
Invalid nodes: {}

Fig. 9. A portion of the research tree

3.2 Deduction of Fundamental Sheets to Be Inserted

At the end of the previous step we get a curve labeling indicating the type of each curve. Using this labeling, we can deduce some fundamental sheets to insert by traversing the geometric surfaces of the geometric model $G = (S, C, V)$ to be meshed. Let $s \in S$ be a geometric surface, then:

- if all its incident curves are labeled 1 then a single level 1 fundamental sheet must be inserted in order to inflate all the mesh faces classified on s. The *inflating* process corresponds to the an adaptation of the pillowing algorithm [12] where a path of mesh faces is opened to insert a complete layer of hexahedra.

- if some incident curves are labeled 1 and some others are labeled 3, then a single level 2 fundamental sheet must be inserted to capture this surface. It will inflate all the mesh faces classified on s plus some inner faces.
- if some incident curves are labeled 1 and some others are labeled 2, then a level 1 fundamental sheet must be inserted in order to inflate all the mesh faces classified on s. It will also inflate the mesh faces classified on the incident surfaces of s, sharing a curve labeled 2 with s.
- if some incident curves are labeled 1 or 3 and some others are labeled 2, then it is like the previous case but with a level 2 fun sheet to insert.
- Finally along each curve labeled 2 or 3, inner sheets must be added in order to capture geometric curves. To do that it is necessary to define a path P of mesh faces forming a 2-manifold inside the volume. The faces of P are then inflated to insert a level 2 or level 3 fundamental sheet. Each 2-labeled curve ends on a 3-valent vertex or is incident to a 2-labeled curve around 4 or 5-valent vertices.

The definition of the sheets to insert is totally automatic following the previous given rules. First, for each geometric surface $s \in S$, we build a set of mesh faces that define the path to insert the fundamental sheet capturing s. Then, for each geometric curve $c \in C$, we check if it is totally captured (one sheet on both sides); if it is not, some sets of faces are also defined corresponding to level 3 fundamental sheets. Once all the sets of mesh faces are defined, there remains only to apply sheet insertion by inflating each set of faces.

Considering the example of Fig. 9, let us now indicate the fundamental sheets to be inserted (see Fig. 10). The first step consists in inserting the level 1 fundamental sheets. The face [F,G,H,I] is surrounded by 1-labeled curves thus a level 1 fundamental sheet is inserted (see Fig. 9-a). Faces [A,B,C], [A,C,D], [B,C,G,F] and [C,G,H,D] are pairwise incident along 2-labeled curves and are surrounded by 1-labeled curves. Hence, a single level 1 fundamental sheet is used to capture all of four (see Fig. 9-c). For the same reason, faces [A,B,E], [A,E,D], [B,F,I,E] and [D,E,I,H] are captured by a single level 1 fundamental sheet (see Fig. 9-d).Eventually, the paths of 2-labeled curves $\{$[B,C],[C,D],[D,E],[E,B]$\}$ and $\{$[G,C],[C,A],[A,E],[E,I]$\}$ require to insert level 3 fundamental sheets (see Fig. 9-d and Fig. 9-e).

4 Results and Discussion

We show two examples to illustrate the proposed capability. In both cases, we insert fundamental sheets in an initial THex mesh, i.e. a hexahedral mesh built by splitting each tetrahedron of a tetrahedral mesh into four hexahedra. The first example shown in Fig. 11 is a diamond-shaped geometric domain, which has two opposite 4-valent vertices and eight 3-valent vertices. Some views of the final mesh are given in Fig. 11-a and Fig. 11-b. In Fig. 11-c and 11-d, two points of view are used to highlight the pairs of level 3 fundamental sheets that go across the volume. To drive the sheet insertion inside the

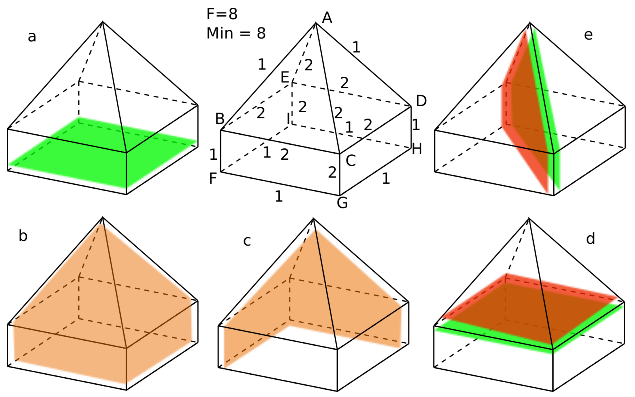


Fig. 10. Fundamental sheets to insert

volume, we use an advancing front algorithm that depends on the underlying hexahedral mesh. The result can thus differ depending on the mesh but the success of sharp feature detection does not rely on the base mesh. The second example is called the hook model. Snapshots from four view angles are given in Fig. 12. We can see that the patterns of THex meshes are no longer along curves where the insertion of fundamental sheets allows us to get surface chords on both sides of each geometric curve. In Fig. 13 are given the three sets of fundamental sheets.

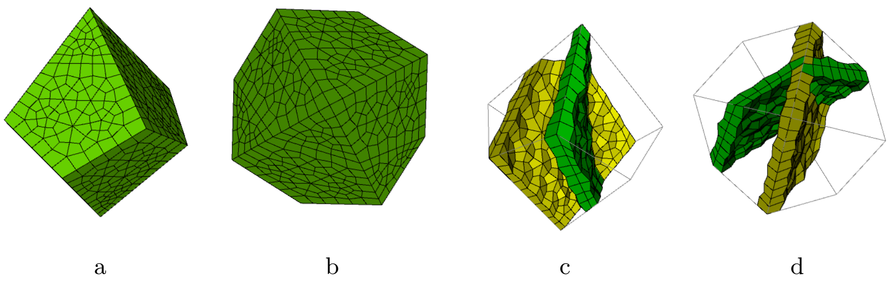


Fig. 11. A diamond-shaped domain discretized with a THex mesh then improved by inserting fundamental sheets

The different experiments we led highlighted important issues about the robustness of our approach and the future improvements to add:

- The resolution of the constraint system always finishes as expected and proved (see lemma (2));
- The definition of the set of faces to inflate depends on the base mesh and it should be improved for inner sheets when the initial mesh is too coarse.

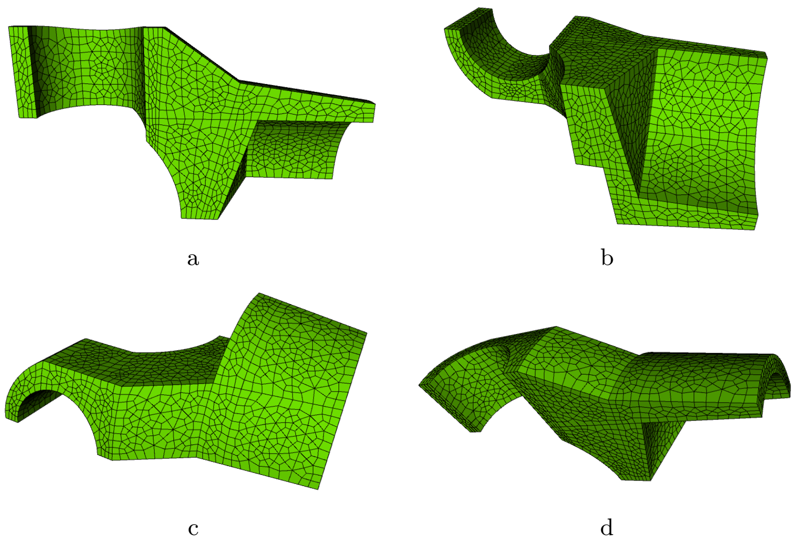


Fig. 12. The resulting mesh for the hook model

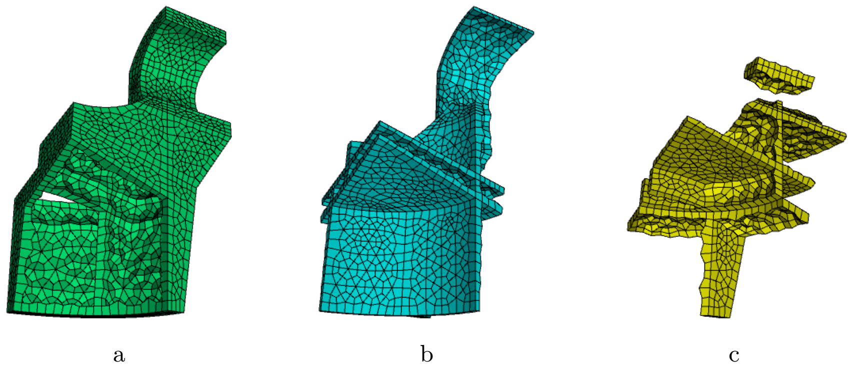


Fig. 13. Level 1, level 2 and level 3 fundamental sheets for the hook model respectively in a, b and c

Inner sheets must form a 2-manifold, follow the structure of the initial mesh and must take care of it propagating out of the geometric domain. Moreover, some level 2 and level 3 fundamental sheets could be connected inside the domain. It is currently not handled. From our point of view, there is no theoretical limitation to this process but it remains technical work to do.

- Once the set of faces is defined, the sheet insertion process always works[4].

5 Conclusion and Future Work

In this work, we proposed an *a posteriori* technique based on the notion of fundamental mesh in order to modify hexahedral meshes to get boundary-aligned layers of hexahedra. The approach consists in two parts: solving a constraint system defined on the topology of the CAD model to discretize; then introducing fundamental sheets inside the mesh. The first step is totally automatic and always terminates. It can be easily extended to support new configurations. Automatic rules are also given for the second step of the algorithm and first results allow us to assess the adequacy of the approach.

However, the second step of the algorithm deals with geometric propagation of surfaces and needs to be improved in the future to minimize its sensitivity to the initial mesh discretization. Another extension of this work will be to handle multi-domain geometries. The resolution of the constraint system will remain unchanged as fundamental sheets should match between adjacent domains, while the definition of the sheets to insert will have to be done on all the domains and no longer on a single one.

References

1. Bern, M., Eppstein, D., Erickson, J.: Flipping cubical meshes. Engineering with Computers 18(3), 173–187 (2002)
2. Blacker, T.D., Meyers, R.J.: Seams and wedges in plastering:a 3d hexahedral mesh generation algorithm. Engineering with Computers 2(9), 83–93 (1993)
3. Folwell, N.T., Mitchell, S.A.: Reliable whisker weaving via curve contraction. In: proceedings of the 7th Int. Meshing Roundtable, pp. 365–378 (1998)
4. Jurkova, K., Ledoux, F., Kuate, R., Rickmeyer, T., Tautges, T.J., Zorgati, H.: Local topological modifications of hexahedral meshes; part ii: Combinatorics and relation to boy surface. In: ESAIM Proceedings Cemracs 2007, vol. 24, pp. 34–45 (2008)
5. Kowalski, N., Ledoux, F., Staten, M.L., Owen, S.J.: Fun sheet matching: Towards automatic block decomposition for hexahedral meshes. Engineering with Computers (2011)
6. Ledoux, F., Shepherd, J.: Topological and geometrical properties of hexahedral meshes. Engineering with Computers, 419–432 (2010)
7. Ledoux, F., Shepherd, J.: Topological modifications of hexahedral meshes via sheet operations: a theoretical study. Engineering with Computers, 433–447 (2010)
8. Ledoux, F., Weill, J.-C.: An extension of the reliable whisker weaving algorithm. In: Proceedings of the 16th Int. Meshing Roundtable, pp. 215–232. Springer (2007)

[4] Note that getting a robust sheet insertion process with mesh classification (i.e. geometry association) is technically difficult to implement.

9. Maréchal, L.: Advances in octree-based all-hexahedral mesh generation: Handling sharp features. In: Clark, B.W. (ed.) Proceedings of the 18th Int. Meshing Roundtable, pp. 65–84. Springer (2009)
10. Merkley, K., Ernst, C., Shepherd, J.F., Borden, M.J.: Methods and applications of generalized sheet insertion for hexahedral meshing. In: Proceedings of the 16th Int. Meshing Roundtable, pp. 233–250 (2008)
11. Mitchell, S.A.: A characterization of the quadrilateral meshes of a surface which admit a compatible hexahedral mesh of the enclosed volume. In: Proceedings of the 13th Annual Symposium on Theoretical Aspects of Computer Science, pp. 465–476 (1996)
12. Mitchell, S.A., Tautges, T.J.: Pillowing doublets: Refining a mesh to ensure that faces share at most one edge. In: Proceedings of the 4th Int. Meshing Roundtable, pp. 231–240. Sandia National Laboratories (October 1995)
13. Murdoch, P., Benzley, S.E.: The spatial twist continuum: A connectivity-based method for representing all hexahedral finite element meshes. In: Proceedings of the 4th Int. Meshing Roundtable, number SAND95-2130, Albuquerque. Sandia National Laboratories (1995)
14. Owen, S.J., Shepherd, J.F.: Embedding features in a cartesian grid. In: Proceedings of the 18th Int. Meshing Roundtable, pp. 117–138 (2009)
15. Owen, S.J., Sunil, S.: H-morph: An indirect approach to advancing front hex meshing. Int. Journal for Numerical Methods in Engineering 1(49), 289–312 (2000)
16. Qian, J., Zhang, Y.: Sharp feature preservation in octree-based hexahedral mesh generation for cad assembly models. In: Proceedings of the 19th Int. Meshing Roundtable, pp. 243–262 (2010)
17. Remacle, J.-F., Shephard, M.S.: An algorithm oriented mesh database. Int. Journal for Numerical Methods in Engineering 58(2) (2003)
18. Shepherd, J.F.: Conforming hexahedral mesh generation via geometric capture methods. In: Clark, B.W. (ed.) Proceedings of the 18th Int. Meshing Roundtable, pp. 85–102. Springer (2009)
19. Staten, M.L., Kerr, R.A., Owen, S.J., Blacker, T.D., Stupazzini, M., Shimada, K.: Unconstrained plastering - hexahedral mesh generation via advancing-front geometry decomposition. Int. Journal for Numerical Methods in Engineering 81, 135–171 (2010)
20. Tautges, T.J., Blacker, T.D., Mitchell, S.A.: The whisker weaving algorithm: A connectivity-based method for constructing all-hexahedral finite element meshes. Int. Journal for Numerical Methods in Engineering 39, 3327–3349 (1996)
21. Tautges, T.J., Knoop, S.: Topology modification of hexahedral meshes using atomic dual-based operations. In: Proceedings of the 12th Int. Meshing Roundtable, pp. 415–423. Sandia National Laboratories (September 2003)