PhD thesis proposal: Polycube-dominant meshing

I. Proposal's context

Numerical simulations solve physical equations to better understand complex phenomena. It is a key aspect in both industry and science: the idea is to replace costly physical experiments (e.g. wind tunnel in fluid dynamics) with computer simulation. This dramatically reduces the overall cost of product development, because it allows to prune hypotheses before undertaking expensive real actions such as the realization of a mechanical part, on site investments such as drilling of oil wells, and then submitting it to tests. One of the central points in numerical simulations is the ability to represent the functions (temperature, pressure, speed, etc.) on the studied objects. The most versatile way to do so is to discretize the domain of the interest into the cells of a mesh.

Today, a majority of pure or highly dominant hexahedral meshes is still hand-made, because no method has proven yet to be versatile and largely applicable enough with an acceptable accuracy, thus inducing large costs and time overheads. The image below shows an example of a hand-made mesh; it took more than one month of engineer time to build it. Our ambition is to reduce this time.

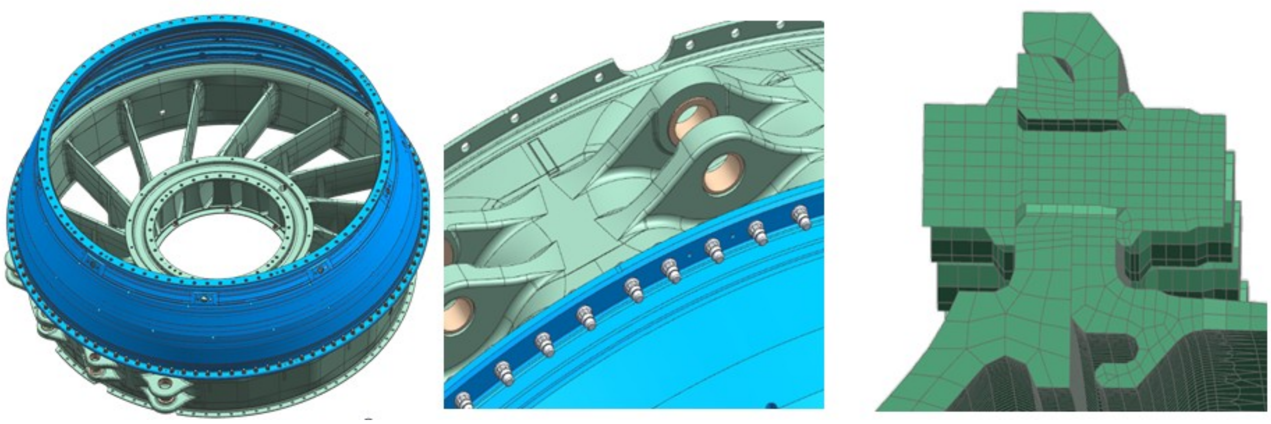

Figure 1: Industrial assembly model: geometry (left), expected mesh (right), so as to be able to simulate contact areas accurately.

A new and original approach to generate hexahedral meshes was proposed recently by the team [Sokolov2016, Ray2018]. It comes from the observation that good quality hexahedral meshes look like a deformed grid almost everywhere. The idea is to define a deformation of the object such that if the final hexahedral mesh (the result) undergoes this deformation, it will match a unit, axis aligned grid. The direct application of this idea computes this deformation, applies its inverse to the unit grid, and obtains a hexahedral mesh. In practice, we introduce more degrees of freedom by considering global parameterizations instead of deformation. In this case, the deformation functions have some discontinuities that make it possible to represent a much larger family of hexahedral meshes: the deformed grid can be cut and glued to itself in a non-trivial way (refer to the Figure 2).

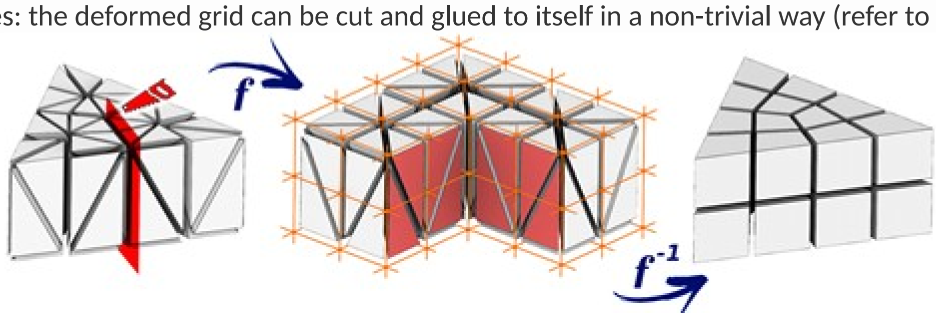

Figure 2: Global parameterization example: the input is cut open along the red plane (left), then it is deformed (middle), and the pre-image of the orange grid produces a hexahedral mesh (right).

Global parameterizations are constructed in two steps: the *Frame Field (FF)* step defines the orientation of the grid at each point of the domain, and the *Cube Covering (CC)* step generates a global parameterization that will align the final mesh with the orientation (FF) defined in the first step. When both steps succeed, the result is a very regular full hexahedral mesh. Unfortunately, the FF step may generate an orientation field with a topological structure (its singularity graph) that is not valid for the second step, and the CC step often generates invalid mappings with (locally) negative Jacobians.

II. Organisation of the project

The project will be split into four main parts:

- *Months 1-6: Design a GP-based hexahedral-dominant prototype. The first task will be to set up a software that generates conforming hexahedral dominant meshes. The developed prototype will be an adaptation of the current Inria implementation of [Ray2018].*

- *Months 6-20: Improve global parameterization algorithms*. With the robustness technical barrier being lifted by the first task, we can attack the mesh quality criteria. For this task we are mainly interested by the case where all the meshing is handled through global parameterizations.

  To narrow down this vast research domain, for this project we will consider a particular case of global parameterizations without discontinuities, *polycubes* (Figure 3) i.e. a union of unit, axis aligned, cubes [Gregson2011]. In these settings, the orientation field does not constrain the hexahedral mesh by a complex singularity graph, but only by the function that maps each point of the domain boundary to its normal in the deformed domain, that is necessary aligned with one axis (x, -x, y, -y, z or -z). The interests of this specific case are two-fold: results are often good enough to generate hexahedral meshes, and it gives a simpler context to study the robustness issues of the CC step.

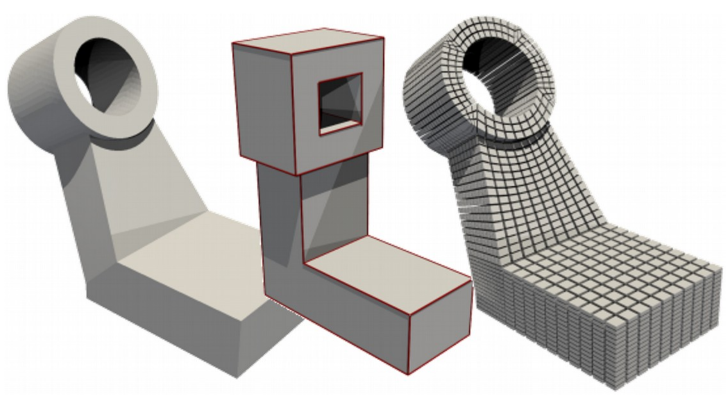

Figure 3: A domain, its polycube, and the final mesh.

  We will start this task by implementing a frame field without singularities, then we will improve the robustness of the CC step by introducing feasibility constraints. We will try different strategies to discover these feasibility: by observation of failure cases, or by a general characterization of all possible ways to produce degeneracy inspired by what was done for the 2D case [Campen2015]. The last element of the task will be to inject these constraints into the solver. This task will provide us some experience to better address the general (non polycube) case.

- *Months 21-28: Combine grid placement with other meshing tools*. While for the previous task we have supposed that GP handles all the meshing, it is unreasonable to rely on GP only. There are many cases where it would be better to combine it with other meshing tools.

Placing a (deformed) grid inside a volume gives a very regular hexahedral mesh. However, it is very difficult to fill the whole volume with such a grid: the boundary alignment constraints may be locally (fillet, chamfer) or globally (thread on screws) incompatible. In these cases, a global parameterization algorithm simply fails to mesh the model. To deal with this issue, we will integrate the grid placement algorithms in a more complete meshing process. To do so, we will analyze the conflicting constraints and solve them by pre-processing and post-processing. We will also try to better exploit the combinatorial structure of the grid, despite possible geometric inversions, as it is locally done in [Lyon2016].

We can actually generate hexahedral dominant meshes by a post-processing step that remeshes the part of the model that is not successfully filled by hexahedra. It makes the process robust, but it would be much better to detect conflicting constraints earlier, before it deteriorates the global parameterization. Indeed, the hexahedral meshes have a “very rigid” combinatorial structure that makes it possible for a (local) constraint to have an impact on the whole mesh. This new way to think about the problem leads to interesting theoretical questions: what are the conditions on the boundary conditions for a valid global parameterization to exist.

Once again, for this project we will address the pre-processing in the particular case of polycubes. In this case, most existing polycube methods already have some specific pre-processing and post-processing methods to manage local conflicting constraints. Their pre-process changes the function that determines, for each point of the boundary, what would be the normal after deformation (x, -x, y, -y, z or -z). Their post-processing is a pillowing step that better distributes the angular distortion due to the singularities, by pushing them inside the domain. These solutions are able to fix simple (and frequent) conflicting configurations, but always have failure cases that are difficult to predict. Moreover, they are mostly designed for organic models where difficulties are very different from what we can observe on our CAD models; we have many conflicting situations that come from configurations of feature edges that meet on a single point.

We will focus on conflicting configurations of the feature edges. The starting point will be to observe failure cases, define the regions that will be impacted, and remove them from the domain to be remeshed (the post-processing will deal with it). Then, we will better manage these regions by replacing them by local hexahedral mesh patterns, and we will try to get an exhaustive classification of conflicting cases.

- *Months 29-36: Push polycube singularities graph deep inside the domain.* Meshes generated from polycubes are typically highly distorted near the (degenerated) singularity graph located on the domain boundary. A common practice is to better distribute this distortion by a pillowing step, that will move the singularity graph inside the domain. We would like to go further in this direction, by pushing the singularity graph deeper inside the domain. We consider this option as a way to generate valid singularity graphs in the general case.

  The first step will be to produce a first layer of hexahedra on the object’s boundary, then to move it inside the domain by insertion/deletion of hexahedron layers. A way to determine layers to add/remove will be to move the singularities graph in order to minimize the rotation of the hexahedra orientation, then solve once again a mixed integer problem to determine the optimal number of hexahedron layers to insert/remove according to the new singularity graph position.

III. References related to the project

[Campen2015] M. Campen, D. Bommes and L. Kobbelt. 2015. Quantized global parametrization. ACM Transactons on Graphics 34.

[Gregson2011] J. Gregson, A. Sheffer, E. Zhang, All-Hex Mesh Generation via Volumetric PolyCube Deformation, Computer Graphics Forum 2011.

[Lyon2016] M. Lyon, D. Bommes, L. Kobbelt, HexEx: Robust Hexahedral Mesh Extraction, ACM Transactions on Graphics, 2016

[Ray2018] N. Ray, D. Sokolov, M. Reberol, F. Ledoux, B. Lévy, Hex-dominant meshing: Mind the gap!, Computer-Aided Design, Volume 102, 2018.

[Sokolov2016] D. Sokolov, N. Ray, L. Untereiner and B. Lévy. 2016. Hexahedral-Dominant Meshing. ACM Transactions on Graphics 35.