Frame Fields with Prescribed Boundary Singularity Graph for Hexahedral Meshing

David Desobry

Université de Lorraine, Inria, F-54000 Nancy, France

Abstract

We present a method for enforcing the boundary singularity graph of a frame field before optimizing it. By controlling the singularity indices at the boundaries, this technique can generate hexahedral meshes with specified boundary edge valences. In situations where feature lines in a CAD model intersect at sharp angles, the traditional approach of aligning the frame field with surface normals leads to non-meshable boundary singularities. Instead, we align the frame field with a precomputed direction field that is specifically designed to produce the valid boundary singularities given in input. We present an algorithm to generate this direction field and provide heuristics to obtain the necessary input boundary singularities. We demonstrate the effectiveness of this approach in preventing the collapse of sharp features through a standard hexahedral meshing pipeline, and we show that modifying the boundary singularities of a frame field can also enhance the performance of current frame field optimization techniques in constructing a valid interior singularity graph.

Keywords: Geometry processing, Frame field, Hexahedral remeshing

1. Introduction

Volume meshes are widely used in industry because they accurately represent the geometry of complex shapes and are suitable for finite element analysis (FEA) and computational fluid dynamics (CFD) simulations. Tetrahedra and hexahedra are the most commonly used elements for meshing volumes. While automatic generation of tetrahedral meshes has reached a level of maturity that allows its usage in industrial contexts, hexahedral mesh generation still requires heavy user interaction. One key difference between these two elements is that hexahedral meshes are more structured and have relatively few irregular edges. An edge is *regular* if it is adjacent to exactly 4 hexahedra inside the mesh and 2 at the boundary. The set of irregular edges forms a *singularity* graph whose properties have been extensively studied [1, 2] but a full characterization of this graph is still lacking.

In this paper, our goal is to improve the usability of the global parametrization method (or frame field based method [3]) for converting a tetrahedral mesh into a hexahedral mesh. This method decomposes the meshing problem into three steps: 1) generating an orthogonal frame field to fix the element orientations and the singularity graph, 2) computing the global parametrization such that the coordinate gradients follow the frames directions, and 3) using a "quantization" algorithm to assign integer values to the isovalues of the parametrization's boundaries and singularities, which allows the extraction of a hex-mesh. One weakness of this method is that a frame field singularity graph may not always be transposable to a hex-mesh singularity graph.

A 3D orthogonal frame is typically represented by a set of three orthonormal directions. The key idea is that a frame has the same symmetry as a cube. Thus, it is an excellent proxy for determining a singularity graph. However, only a subset of all possible frame field singularity graphs can be used for hexmeshing [1, 2]. In fact, most frame field generation techniques constrain one direction of the frame to be normal to the boundary and optimize for smoothness inside the shape [4, 5]. As a consequence, the boundary edge singularity indices are simply set as a rounding of the dihedral angle of the boundary faces. Typically, if the dihedral angle φ_{ij} of a boundary edge ij falls within the range of $\pi - \pi/4$ and $\pi + \pi/4$, a smoothed frame field that aligns with the boundary normals snaps it to a π angle and treats the boundary edge as regular. Instead, if the dihedral angle falls outside of this range, the edge is considered singular, and its index is determined by the nearest multiple of $\pi/2$. However, when dealing with CAD models that have feature lines meeting at acute or strongly obtuse angles, singularity indices determined by rounding the geometric dihedral angle often result in invalid singularity graphs. A particularly telling example is seen in geometries with dihedral angles smaller than $\pi/4$ (Figure 1). On the one hand (Figure 1, left), a frame field aligned with the boundary normals snaps it to a 0 angle, resulting in a valence 0 edge that cannot be meshed. On the other hand (Figure 1, right), a frame field aligned with a carefully chosen vector field snaps it to a $\pi/2$ angle, which corresponds to a valence 1 edge in the output hexahedral mesh. This demonstrates that we can influence the boundary edge singularities of a frame field by adjusting its boundary alignment constraints. Furthermore, modifying the singularity graph on the boundaries also affects the singularity graph *within* the model, which can be beneficial.

In this paper, we propose a simple algorithm to generate frame fields with prescribed edge valences on boundary edges. The key idea is to constrain the boundary frame field to align with a direction v which may differ from the boundary normal n so that the rounding leads to the desired edge valence. Then, we use the symmetric frame field optimization of Ray *et al.* [5] to smoothly interpolate the field and obtain the internal singularity graph. If this frame field has the desired singularities, it cannot be used directly for meshing as it is not aligned with the boundary. Therefore, we compute a non-symmetric frame field with the same singularity graph but, this time, aligned with the boundary normals. The hexmesh is extracted from this field using the state-of-the-art method [6, 7].

We demonstrate that our method can be used to address the problem of meshing CAD models with feature lines intersecting at sharp angles (Fig 2). In addition, we propose heuristics for choosing edge valences in order to avoid the limit cycle problems

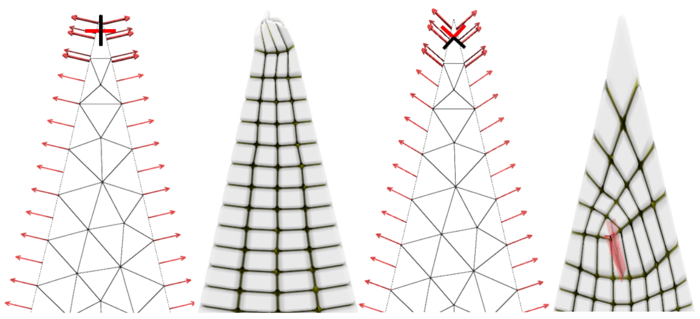

Figure 1: In the case of meshing a low-angle edge, an orthogonal frame field aligned with the surface normals satisfies the two constraints with opposite direction vectors of a frame (top, left), leading to a non-meshable valence 0 edge. We compute new direction constraints such that a frame field aligns with our directions with two orthogonal vectors of the frame (top, right), resulting in a meshable valence 1 edge. The global smoothness of our direction constraints enables the production of a valid interior singularity (red line, bottom right) using a frame field optimization method [5].

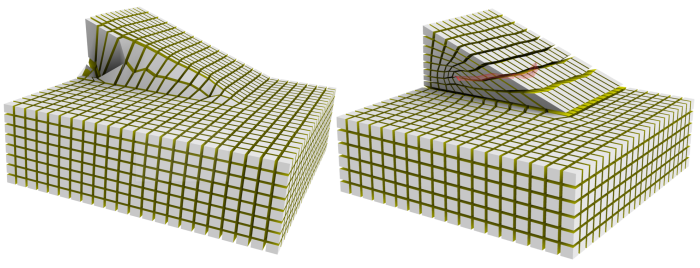

Figure 2: Because the slope's angle is too low, an orthogonal frame field following the boundary normal directions fails to separate the slope's end from the slope's support, resulting in a degenerated result (left). Instead, our direction constraints obtained with our method separate the slope and its support with a 90° angle, resulting in a valence 3 edge at the slope's end. The red line corresponds to a necessary valence 3 interior singularity line produced by frame field optimization [5] with our direction constraints.

described in [2].

To summarize, our algorithm follows the steps:

1. Compute new boundary constraints v (Sec. 4);
2. Generate a smooth frame field with boundary direction constraint v using Ray *et al.* [5];
3. Compute a *non-symmetric* frame field with fixed singularities and normal boundary constraints (Sec. 3);
4. Extract a seamless parametrization [6] and a hexahedral mesh [7].

2. Related work

2.1. Singularity graphs

Pietroni *et al.* [3] define the singularity graph of a hexahedral mesh as the interior edges that do not have a valence of 4, as well as the boundary edges that do not have a valence of 2. The structure of a hexahedral mesh is determined by the singularity graph; the simpler the singularity graph, the more structured the hexahedral mesh. The benefit of frame field based methods is that they produce hex meshes with the simplest singularity graphs. If local characterization of an hexmesh singularity graph is well understood [8, 9], fewer works have attempted to find global characterization [1]. To the best of our knowledge, the only way to check that a given graph corresponds to an hexmesh is by explicitly building the mesh. This is very impractical and is a major challenge for hex meshing methods. Therefore, singularity graph modification techniques [8, 9, 1] are limited to local corrections of common defects. These methods cannot be used to prescribe a boundary singularity graph because this may significantly alter the inner graph. Frame field generation from fixed singularities [1, 10] requires a complete and valid singularity graph as input. Hence, it is not possible to utilize these methods with just an input boundary singularity graph.

2.2. 3D frame field generation

The goal of frame field optimization is to generate a valid interior singularity graph by interpolating the input boundary alignment constraints. The most common representation of an orthogonal frame uses 9 coefficients in the spherical harmonic decomposition [4, 5]. Many methods have been proposed to compute smooth frame fields. They often rely on three steps [4, 8, 5]: first, a loose convex relaxation is used as an initialization step; second, each set of 9 coefficients is projected in the space of orthogonal frames; and third, an optional non-convex smoothing procedure is used.

Many other algorithms are possible. Palmer *et al.* [11] suggest alternating between a diffusion step and a projection step to obtain a smoother field. Vaxman *et al.* [12] uses a Boundary element method to interpolate boundary frames to the interior.

Unfortunately, the necessary conditions exposed in [6] for a frame field singularity graph to be compatible with a hexmesh are extremely challenging to be used in a constructive way. Thus, there are no available algorithms guaranteeing the validity of a frame field singularity graph. In particular, all these methods for interpolating and smoothing orthogonal direction fields fail to properly handle acute dihedral angles at the boundary. In practice, the boundary singularity indices of the final hexmesh are determined by rounding the dihedral angle.

Desobry *et al.* [13] is one of the only options available that can correctly handle feature lines intersecting at acute angles. The idea is to replace the 3D frame field by a field of three independent directions, which are represented by 14 spherical harmonic coefficients. In contrast to orthogonal frames, these fields can utilize non-orthogonality to impact the output valence. However, aligning with the boundary normals still precludes the creation of boundary edge valences that deviate significantly from what the surface dihedral angles suggest. Moreover, this approach is computationally expensive as it requires solving a challenging non-convex optimization problem.

3. Hexahedral meshing pipeline

In this section, we present our pipeline for frame field generation. Our method allows us to generate hexahedral meshes with specified boundary edge valences by aligning a frame field with the direction vectors from Sec. 4 instead of the surface normals (Sec. 3.2). As a result, our frame field has the desired boundary singularity graph, but its orientation on the boundary is not ideal for extracting high-quality hexahedrals aligned with the input surface. To resolve this, we perform a second optimization to realign the frame field with the surface normals while keeping its singularities fixed (Sec. 3.3). With a valid frame field topology and a new geometry aligned with the surface normals, standard hexahedral meshing methods can be used, as outlined in Sec. 3.4.

3.1. Notations

We have, as input, a tetrahedral mesh (V, E, F, T) where we denote V the set of vertices, E the set of edges, F the set of triangles, and T the set of tetrahedra. The boundary of the mesh is a triangle mesh (V_b, E_b, F_b) with V_b the set of boundary vertices, E_b the set of boundary edges, and F_b the set of boundary triangles.

We denote n_i as the normal vector of boundary facet $i \in F_b$, and e_{ij} as the edge separating a facet i from one of its three neighbors $j \in \mathcal{N}(i)$.

3.2. Frame field topology

Each tetrahedron $i \in T$ is endowed with three orthonormal vectors stored as columns of the rotation matrix $E_i \in \mathrm{SO}(3)$. We use the algorithm of Ray *et al.* [5] to compute a smooth symmetric frame field with boundary alignment constraints. A symmetric frame is represented by 9 spherical harmonic coefficients and then reprojected in the space of orthogonal frames.

The transformation to go from one frame $i \in T$ to an adjacent frame $j \in T$ can be decomposed into the rotation $R_{ij} \in \mathrm{SO}(3)$ and the matching matrix $g_{ij} \in \mathrm{SO}(3)$:

$$E_i = R_{ij} E_j g_{ij}. \tag{1}$$

The matching matrices g_{ij} take into account the change of symmetry and belong to the orientation-preserving octahedral symmetry group Γ, the cardinal of Γ is 24. The rotation of the frame R_{ij} accounts for the part of the rotation that is independent of the symmetry jump. Both matrices are heavily correlated because, for each matching matrix, the rotation R_{ij} is uniquely defined.

As noticed in [1], the singularity graph is fully characterized by the frame matching matrices g. Thus, to obtain meshable singularities, we must correctly identify the symmetries between adjacent frames. In practice, given a symmetric frame field, the matching matrices are unknown and are computed in the least squares sense:

$$g_{ij} = \underset{g \in \Gamma}{\arg\min} \left\| E_i - E_j g \right\|_F^2, \tag{2}$$

implying that R_{ij} should be as close as possible to identity.

However, at the boundary and even more so along a feature edge, the frames are fixed, and the matchings are heavily constrained by the alignment with both the normal direction and the feature edge direction. More precisely, for two adjacent boundary facets $i, j \in F_b$, one column of E_i must be colinear to n_i and one column of E_j must be colinear to n_j. Also, as E_i, E_j satisfy Eq. (1), the rotation R_{ij} aligns the column of E_j colinear to n_j with a column of E_i (or its opposite). If these column directions are the facet normals n_i and n_j (or $-n_i$ and $-n_j$), then ij is a regular edge of valence 2. More generally, the valence k_{ij} of the output hexmesh at the boundary edge ij is directly related to the rotation R_{ij} (and to the matching g_{ij}) by the formula:

$$k_{ij} = 2 - \frac{2}{\pi} \mathrm{atan2}\left(\langle n_i \times (R_{ij} n_j), \frac{e_{ij}}{\left| e_{ij} \right|} \rangle, \langle n_i, R_{ij} n_j \rangle \right). \tag{3}$$

In the case of a very acute dihedral angle (as in Fig. 1), the matching given by Eq. (2) imposes that the rotation R_{ij} should be as close as possible to identity. Hence, the normals on both sides of the hard edge will match as opposite and, as stated by Eq. (3), imply that this edge is adjacent to zero hex.

To avoid these catastrophic rounding errors, we constrain the symmetric frame field to be aligned to a direction v at the boundary such that the matching matrices computed with Eq. (2) produce the desired edge valences. The computation of the boundary field v is detailed in Sec. 4.

3.3. Smooth frame field with fixed matching

Now that we have a frame field E with the correct matching matrices, we would like to realign the boundary frames with their corresponding surface normals. To do so, we need to identify, for each boundary frame E_i, the column vector that aligns with the direction v_i and transfer this constraint to the normal vector n_i. Thus, a boundary-aligned frame $\bar{E}$ must satisfy the constraint:

$$n_i^\top \bar{E}_i = v_i^\top E_i, \quad \forall i \in F_b.$$

With these boundary constraints, we would like to find the smoothest frame field whose matching matrices are the g_{ij} obtained with Eq. (2):

$$\begin{array}{ll} \min_{\bar{E}} & \sum_{i \in T} \sum_{j \in \mathcal{N}(i)} \left\| \bar{E}_i - \bar{E}_j g_{ij} \right\|_F^2, \\ \text{subject to:} & n_i^\top \bar{E}_i = v_i^\top E_i, \forall i \in F_b. \end{array} \tag{4}$$

The optimization problem in Eq. (4) does not impose that $\bar{E}_i$ is a rotation matrix. Thus, we solve a Procrustes problem at each tet to project the matrix in the space of rotation matrices using a singular value decomposition.

3.4. Hexahedral meshing method

The boundary aligned frame field $\bar{E}$ and its matching matrices g can be used to generate hexmeshes with the standard frame field based pipeline [3].

The CubeCover algorithm [6] produces seamless parameterizations by using the three gradient directions provided by the vectors of an input frame field. To achieve this, the algorithm generates three scalar fields, which correspond to a volume parameterization. The scalar fields are linear per tetrahedron, and the relationship between fields across interface facets is defined by "matching matrices" [6]. By using the matching matrices g_{ij}, CubeCover generates parametrization with the same singularity graph as the input frame field. Once an integer seamless parameterization is obtained, a hexahedral mesh is extracted using Hexex [7].

The images of the hexahedral meshes in this paper are displayed using hexalab [14], which shows both the surface of the hexahedral mesh and the interior singularity lines. These lines are colored red or green, depending on whether the interior singularity edge has a valence of three or five. Since no postprocessing smoothing is applied to the hexahedral meshes after extraction with hexex, the red and green lines seen in the images correspond directly to the interior singularity graph of the frame field generated from the direction constraints of Sec. 4. Hence, these lines follow the edges of the input tetrahedral mesh.

4. Computing boundary constraints from boundary valences

In practice, choosing the closest matching matrix, as in Eq (2), is equivalent to picking edge valences by rounding of boundary dihedral angles. For a dihedral angle φ_{ij} deduced from boundary alignment constraints of a frame field, the associated edge valence with the pipeline described in Sec 3 is $\mathrm{round}(2\varphi_{ij}/\pi)$. In

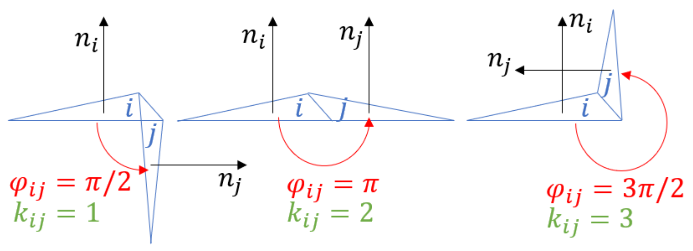

Figure 3: The dihedral angle φ_{ij} and the geometrical edge valence k_{ij} are calculated using the two facet normals n_i and n_j adjacent to the boundary edge ij.

this section, our goal is to find a new field of boundary direction constraints v so that the rounding of φ_{ij} leads to the user-prescribed boundary valences k_{ij}. To do so, we rely on greedy optimization to minimize two objectives, one constraining the dihedral angles and the other imposing to preserve the boundary curvature.

Because of the rounding, we want to be certain that the maximum angle deviation from the target dihedral angle is smaller than $\pi/4$. For this reason, we do not use a least squares objective but rather a minimization of a maximum angle deviation.

The discussion of how to choose the k_{ij} valences is differed to the Section 5.

4.1. Energies

We have as input a tetrahedral mesh whose boundary facets are triangles; we use the same notations as Sec 3.1. We denote as $\varphi(n_i, n_j)$ the dihedral angle between two planes whose normals are n_i and n_j and intersect along the edge e_{ij}.

$$\varphi(n_i, n_j) = \pi - \operatorname{atan2}\left(\langle n_i \times n_j, \frac{e_{ij}}{|e_{ij}|}\rangle, \langle n_i, n_j\rangle\right).$$

Figure 3 illustrates all of these variables.

At a triangle i, we would like our new direction constraint v to define a dihedral angle matching the boundary valences k_{ij}:

$$\delta_i^d(v) := \max_{j\in\mathcal{N}(i)} \left|\varphi(v_i, v_j) - k_{ij}\pi/2\right|. \tag{5}$$

This objective alone is insufficient because it is entirely independent of the surface curvature. For a sphere, minimizing Eq. (5) would lead to a constant vector field v from which it would be impossible to generate a valid inner singularity graph. Thus, we also desire that the new boundary field v stays as close as possible to the initial curvature. With this goal in mind, we define, at each half dual edge ij of the boundary, a pair of ideal normals $\bar{n}_{ij}$ and $\bar{n}_{ji}$. Both normals should be close, in the least squares sense, to the original normals but also satisfy the dihedral angle constraint. More formally, our pair of ideal normals is the solution to the optimization problem:

$$\begin{array}{ll} \min_{\bar{n}_{ij},\bar{n}_{ji}\in\mathbb{R}^3} & |\bar{n}_{ij} - n_i|^2 + |\bar{n}_{ji} - n_j|^2 \\ \text{subject to:} & \varphi(\bar{n}_{ij}, \bar{n}_{ji}) = k_{ij}\pi/2 \\ & |\bar{n}_{ij}| = |\bar{n}_{ji}| = 1 \end{array} \tag{6}$$

Let R_{ij} be the rotation of angle $\pi - k_{ij}\pi/2$ and of axis the edge vector e_{ij}, we use it to find the solutions of Eq. (6) and obtain our ideal half-edge normals:

$$\bar{n}_{ij} = \frac{n_i + R_{ij}n_j}{|n_i + R_{ij}n_j|}.$$

The smoothness energy measures the maximum angular distance to our ideal half-edge normals:

$$\delta_i^n(v) := \max_{j\in\mathcal{N}(i)} \arccos\left(v_i^\top \bar{n}_{ij}\right). \tag{7}$$

The angle deviation at triangle i is defined as the maximum of the angular distance from the target dihedral angles (Eq. (5)) and the angular distance from the ideal half-edge normals (Eq. (7)):

$$\mathcal{E}_i(v) := \max(\delta_i^d(v), \delta_i^n(v)).$$

The maximum angle deviation among all triangles i for a given set of direction vectors v is denoted as follows:

$$\mathcal{E}(v) := \max_{i\in V_b} \mathcal{E}_i(v).$$

Our objective in the following subsection Sec. 4.2 is to find a set of direction vectors v that minimizes $\mathcal{E}(v)$. The dihedral angle rounding guarantees that a smooth orthogonal frame field constrained by the directions v has a boundary singularity graph corresponding to the input boundary edge valences k_{ij} whenever $\mathcal{E}(v) < \pi/4$ (see Sec. 6.2).

4.2. A greedy optimization method

Since we would like to minimize a non-smooth objective function, we rely on a simple greedy algorithm.

We initialize the directions v_i to be equal to the normal vectors n_i and we iteratively replace v_i by the unit vector u_i which minimizes the angle deviation $\mathcal{E}_i(v)$ when v_j are considered constant for all $j \neq i$. The vector u_i is determined by the neighboring vector directions at the time of modification. Its exact computation is detailed in Sec. 4.3.

After the vector v_i is modified, we need to update the vectors of the neighboring triangles. Thus, our greedy algorithm uses a queue to determine the facet indices that must be updated. First, the queue is filled with all facet indices. Then, while the queue is not empty, we pick and remove the front index. If a vector u_i lowers the current facet energy, we replace the direction vector v_i with u_i and add all the adjacent facets to the queue (if they are not already in). This procedure is summarized in the pseudo-code of Alg. 1.

Algorithm 1 Greedy boundary vector optimization

```
v ← initial normal vectors of boundary facets.
Q ← boundary facet indices
while Q is not empty do
    i ← Q.pop_front()
    u ← MinFacetDirection(i)
    if $\mathcal{E}(u) < \mathcal{E}(v)$ then
        $v_i \leftarrow u_i$
        for all $j \in \mathcal{N}(i)$ do
            Q.push(j)
        end for
    end if
end while
```

4.3. Minimization of a single vector

In order to use Alg. 1, we need to solve the subproblem of minimizing $\mathcal{E}(v)$ for the variable v_i while considering all other

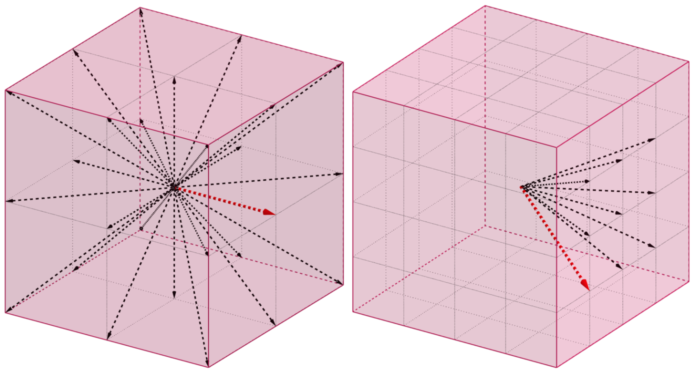

Figure 4: In the initial step, the direction vector that minimizes $\mathcal{E}_i$ is chosen among 26 direction vectors, each represented by points with integer coordinates at the surface a cube centered on the origin with a size of 2. Then, the size of the cube is doubled to identify the optimal direction vector (displayed in red, right) from the integer points that surround the previous (red, left) solution. This procedure is repeated 9 times to reach a direction vector that is nearly optimal.

components fixed. Our energy only depends on the angle between vectors, so the length of v_i does not change the value of $\mathcal{E}$, only the direction matters. Thus, we use an iterative searching algorithm by sampling directions of integer coordinated on the surface of a centered cube of edge length 2^n for increasing $n \in \mathbb{N}$.

We start with a cube of length $n = 2$ resulting in 26 sampled directions. At the first stage, we exhaustively test all directions and keep the one with the lowest energy. In the next iteration, we double the size of the cube and test only the 8 (or 6 at a corner) adjacent directions with integer coordinates, as illustrated in Fig. 4. We again keep the best candidate. We iterate this procedure 9 times.

A cube of length 2^p has a total of $24{\times}4^p{+}2$ sampled directions. Instead of testing all directions, our dichotomy method only tests $9k + 26$ direction vectors after p iterations to find an approximation of a local minimizer v_i. In practice, we stop at $p = 9$. This procedure is summarized in the pseudo-code of Alg. 2.

Algorithm 2 Iterative searching algorithm

```
function MinFacetDirection($v$, $i$)
    $N = 2$
    $st \leftarrow (-1, -1, -1)$
    $end \leftarrow (1, 1, 1)$
    for $iter \leftarrow 0$ to 10 do
        $pt \leftarrow$ MinPointOfCubeSurface($v, i, st, end, N$)
        $N \leftarrow 2 \cdot N$
        $st \leftarrow 2 \cdot pt - (1, 1, 1)$
        $end \leftarrow 2 \cdot pt + (1, 1, 1)$
    end for
    return normalizedVec3($pt$)
end function
function MinPointOfCubeSurface($v, i, st, end, N$)
    $u \leftarrow v$
    $pt \leftarrow (1, 0, 0)$
    for $x \leftarrow st_x$ to $end_x$ do
        for $y \leftarrow st_y$ to $end_y$ do
            for $z \leftarrow st_z$ to $end_z$ do
                if $\max(|x|, |y|, |z|) == N/2$ then
                    $u_i \leftarrow$ normalizedVec3($x, y, z$)
                    if $\mathcal{E}_i(u) < \mathcal{E}_i(v)$ then
                        $v_i \leftarrow u_i$
                        $pt \leftarrow (x, y, z)$
                    end if
                end if
            end for
        end for
    end for
    return $pt$
end function
```

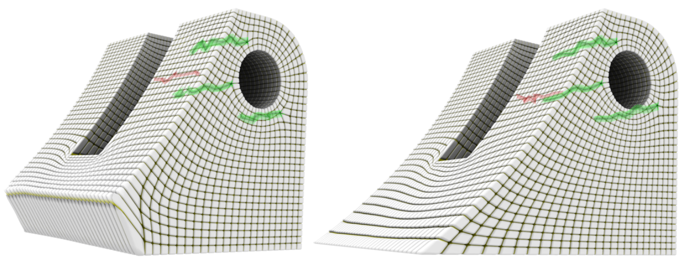

Figure 5: An orthogonal frame field following the boundary surface normals near a low-angle edge considers the edge as an output valence 0 edge, which leads to a degenerancy result (left). We solve it by computing new direction constraints that act as if the low-angle edge was in fact a $\pi/2$ angle edge, which an orthogonal frame field has no problem dealing with (right).

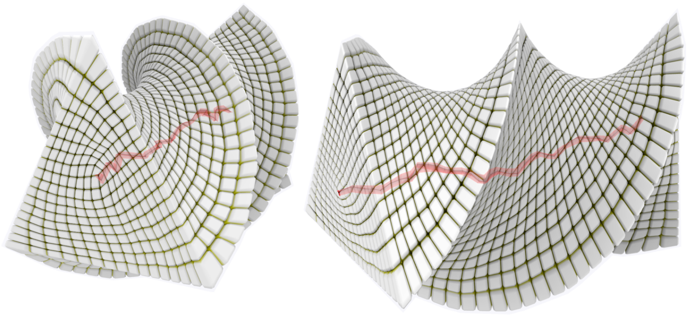

Figure 6: Adding twist during extrusion permits to generate a lot of non-orthogonality in many different directions. Our method constructs a hexahedral mesh result as if the extrusion were done without twist.

5. Different strategies to determine boundary edge valences

In this section, we provide three heuristics for choosing boundary edge valences. Sec. 5.1 focuses on reproducing the boundary edge valences suggested by the surface normals (see Fig 3) except on low-angle edges to avoid the failure case $k_{ij} = 0$. However, low-angle edges are not the only cases in which the boundary naturally creates a non-meshable configuration. In Sec. 5.2, we present a heuristic that uses the surface geometry to solve non-orthogonality problems that are not related to low-angle edges. The Sec. 5.3 focuses on CAD models whose geometry cannot be used to determine a valid boundary singularity graph. These are frequently associated with cases of extreme non-orthogonality of feature edges, and we show how to solve such cases using edge valences given as input. Finally, in Sec. 5.4, we show examples of CAD models in which modifying the boundary singularity graph is optional but helps the frame field optimization method [5] to find a valid interior singularity graph.

5.1. Low-angle edges

If the input tetrahedral mesh has dihedral angles φ_{ij} less than $\pi/4$, an orthogonal frame field aligned with surface normal di-

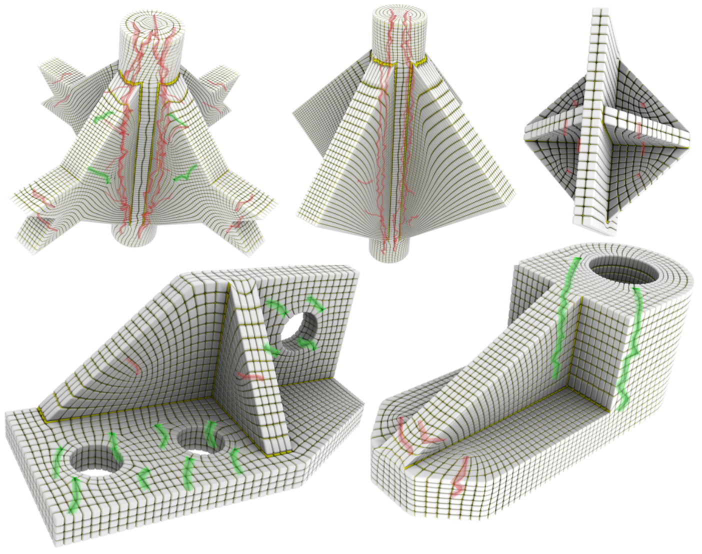
Figure 7: When a dihedral angle is close to $p\pi/2+\pi/4$ with $p \in \mathbb{N}$, we remark that choosing either $k = p$ or $k = p + 1$ results in a valid boundary singularity graph, while the other does not. To obtain the above results, our heuristic is to choose the edge valence $k = p + 1$ if the dihedral angle φ is greater than $p\pi/2 + 0.9\pi/4$.

rections has a boundary singularity graph corresponding to a valence $k_{ij} = 0$ edge in the output hexahedral mesh, which is not meshable (see Fig. 5, left).

We propose a simple heuristic consisting of assigning an edge valence of $k_{ij} = 1$ whenever the dihedral angle φ_{ij} is smaller than $\pi/4$. For a boundary edge $ij \in E_b$, we compute the edge valences as:

$$k_{ij} = \max\left(\text{round}\left(\frac{2\varphi_{ij}}{\pi}\right), 1\right).$$

As a result, direction constraints v computed from these valences (Sec 4) eliminate low-angle issues while not creating new ones, as illustrated in Figure 5 (right) and Figure 6. This means that using these v directions constraints is always preferable to using surface normal n directions constraints.

5.2. *Ambiguous dihedral angles*

Many of the problems we encounter with the MAMBO dataset [15] are caused by non-orthogonality issues on edges with dihedral angles $\varphi \approx p\pi/2 + \pi/4, p \in \mathbb{N}$. In this case, choosing the value $k = p$ can lead to boundary edge valences that are incompatible with hex meshing. This meshing problem can be solved by systematically increasing the edge valence by one whenever the dihedral angle is too close to $\pi/4$ modulo $\pi/2$:

$$k_{ij} = \max\left(\text{round}\left(\frac{2\varphi_{ij}}{\pi} + 0.05\right), 1\right).$$

Figure 7 shows 5 hexahedral meshes successfully generated thanks to this simple heuristic.

5.3. *CAD models with user prescribed edge valences*

For hexahedral meshing of CAD models, the user can also add to the feature edge description the desired edge valence. By doing so, our method can construct boundary direction constraints according to these input valences. If we achieve a maximum deviation angle $\mathcal{E}(v)$ smaller than $\pi/4$ (Sec. 6.2), we ensure that a frame field following these direction constraints will have a valid

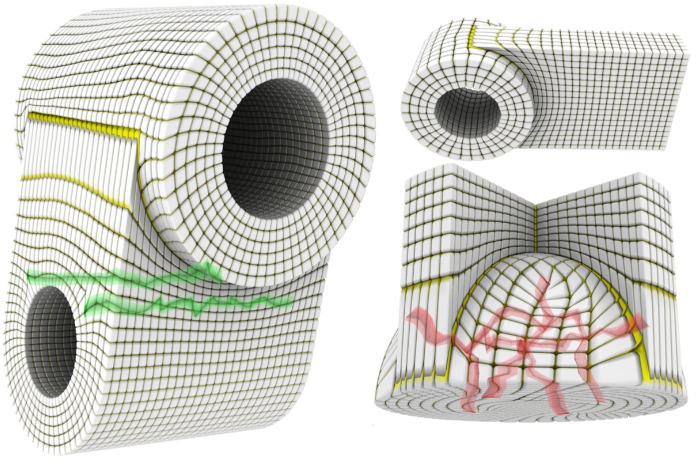
Figure 8: In the case of extreme non-orthogonality of feature edges in CAD models, it is challenging to determine geometrically the feature edge valences. To obtain these results, we prescribed the boundary valences to the feature edge descriptions of the input models.

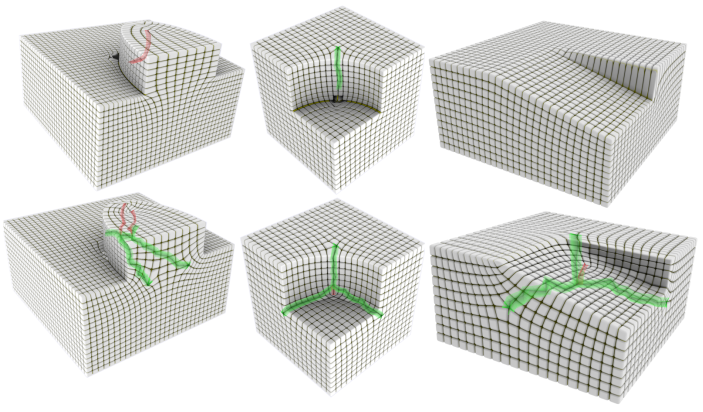
Figure 9: Replacing all valence 3 edges by valence 2 edges can transform non-meshable singularity graphs into valid ones.

boundary singularity graph (corresponding to the valid edge valences given on feature edges). Figure 8 shows examples of CAD models where we need the boundary edge valences in input as our heuristics fail to produce valid meshes.

5.4. *Modification of inner singularity graphs*

Reberol *et al.* [2] illustrate several configurations where computing a smooth frame field leads to non-meshable singularity graphs. They propose a solution to this problem by changing the geometry of the input tetrahedral mesh. Rather than changing the geometry of the input, we simply replace valence 3 edges with valence 2 edges. For the models in Figure 9, we used:

$$k_{ij} = \min\left(\text{round}\left(\frac{2\varphi_{ij}}{\pi}\right), 2\right),$$

on all boundary edges.

Figure 9 shows three examples in which we avoided the problem of invalid singularities by simply changing the k_{ij} values on some boundary edges. However, these modifications to boundary singularities have a negative impact on the Hausdorff distance between the input tetrahedral mesh and the output hexahedral mesh. Moreover, the hexahedral mesh partially loses feature edge alignment.

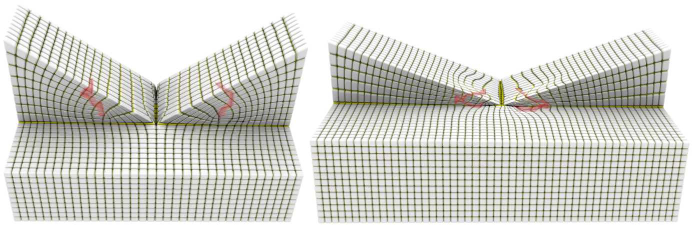

Figure 10: To compute a valid hexahedral mesh, a prescribed edge valence of 4 on the slopes intersection edges is required. Starting from edge interior angles of 4.37 (left) and 3.87 (right), the orthogonal frame field optimization [5] produces a valid interior singularity graph as if the interior angles of the slopes intersection edges were close to 2π. Our algorithm reduces the maximum deviation angles $\delta_{\max}$ on the boundary direction constraints from 1.91 (left) and 2.41 (right) to 0.38 and 0.48.

6. Results

We tested our algorithm on the 74 "basic" models of the MAMBO dataset [15]. Figure 11 shows the computation times with respect to the number of tets for each model in our dataset. Out of the three main steps of our algorithm (computation of new boundary constraints, generation of a symmetric frame field, and realignment with the boundary normals), the computation of a symmetric frame field with Ray *et al.* [5] is the most expensive.

The computation of the boundary constraints from edge valences relies on the greedy optimization described in Section 4. Although we have no proof of convergence or a bound to the optimal solution, this scheme works well in practice. Figure 12 compares, for all our models, the maximal angle deviation $\mathcal{E}$ evaluated with the initial boundary normals and with the vector field v, output of Algorithm 1. We always succeed in lowering the angle deviation well below the bar of $\pi/4$, thereby avoiding the rounding error.

6.1. Comparisons with [13]

Desobry *et al.* [13] propose a representation of non-orthogonal direction fields that can solve some meshability issues induced by strongly non-orthogonal corners. They rely on a smooth non-convex optimization process, initialized with an orthogonal frame field computed with Ray *et al.* [5].

If this method provides an alternative solution to boundary hexmeshability, it is not as flexible as our method and can only fix a smaller range of problems. For example, when the dihedral angle φ_{ij} of a boundary edge is in the range $[p\pi/2, (p+1)\pi/2]$, a hexahedral mesh computed from a non-orthogonal direction field will have a boundary edge valence of p or $p+1$, depending on how the field is constrained with the surface's normal directions.

Our method can generate boundary edge valences with arbitrary values. Figure 10 shows double slopes models with intersection edges having an interior angle in the range $[\pi, 3\pi/2]$. With prescribed edge valences of 4 on the intersection edges, our method builds direction constraints that make possible the computation of a valid hexahedral mesh. A non-orthogonal frame field aligned with the normal directions cannot produce this valid boundary singularity graph because the intersection edges can at best be of valence 3.

In terms of performances, Desobry *et al.* [13] report a computation time roughly 1000 times slower than an orthogonal frame field generation. As demonstrated by Figure 11, our method is much faster as the computation time is largely dominated by the orthogonal frame field computation.

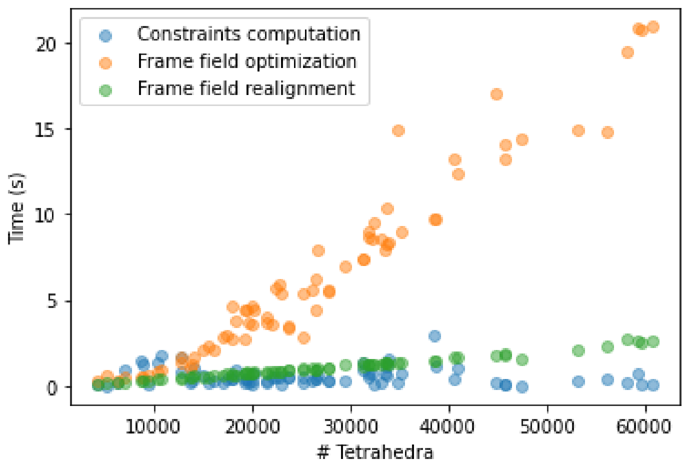

Figure 11: Computation time of the three steps of our frame field generation algorithm: computation of boundary constraints, orthogonal frame field optimization and frame field realignment. Our algorithm adds a very small time overhead to standard frame field generation methods.

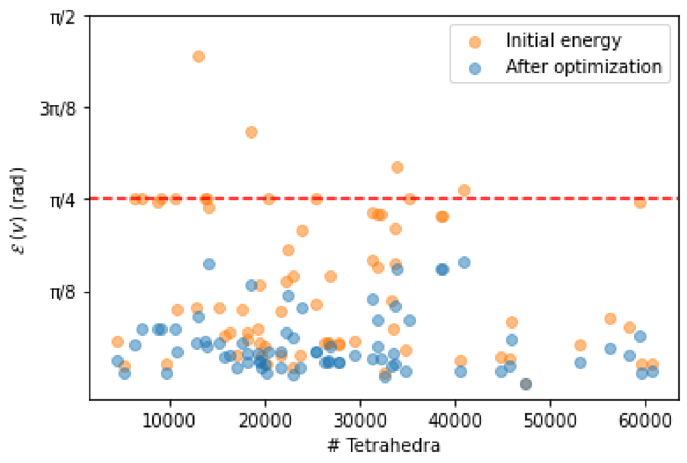

Figure 12: Comparison between the initial energy and the energy after optimization. On "B" models of the mambo dataset [15] and with input edge valences obtained with the heuristic of Sec 5.1, we obtain directions v such that the maximum deviation angle $\mathcal{E}(v)$ is below $\pi/4$ after optimization. This means that the boundary singularity graph of our frame field corresponds to the input edge valences.

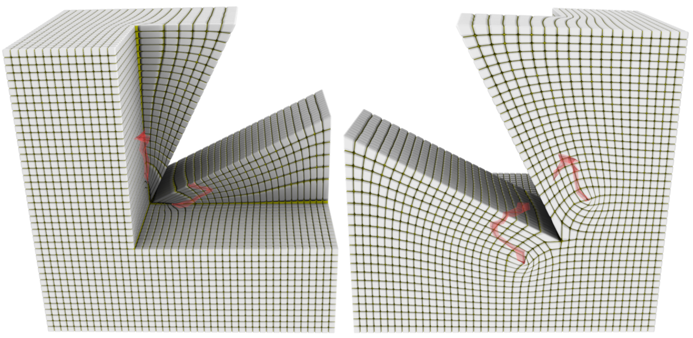

Figure 13: Our method produces boundary constraints that result in a boundary edge valence of 5, simulating a volume surface that self-intersects (interior angle of $5\pi/2 > 2\pi$) around the edge of the two slopes. The initial deviation angle of $\mathcal{E} = 2.2$ is reduced to $\mathcal{E} = 0.46$ after optimization.

6.2. *Failure cases*

The feature edge valence k_{ij} of a hexahedral mesh produced using the method described in Sec. 3 depends on the dihedral angle $\varphi(v_i, v_j)$ of the two direction constraints of the adjacent facets i and j as outlined in Sec. 4.1. The goal of Equation (5) is to reach a value of $\varphi(v_i, v_j)$ which is at worst at a distance of $\pi/4$ of $k_{ij}\pi/2$. However, as $\varphi(v_i, v_j)$ is an angle between 0 and 2π, our target edge valence cannot exceed 4. To overcome this limitation, for valences $k_{ij} \geq 5$, Equation (5) must be modified to take into account the lowest value modulo 2π:

$$\delta_i^d(v) = \max_{j\in\mathcal{N}(i)} \min_{p\in\mathbb{Z}} \left|\varphi(v_i, v_j) - k_{ij}\pi/2 + 2p\pi\right|.$$

This modification is demonstrated in Figure 13 where a boundary edge valence of 5 is desired and achieved. In this case, the initial maximum deviation angle $\mathcal{E} = 2.2$, which corresponds to an initial dihedral angle of $5\pi/2 - 2.2$, is reduced to $\mathcal{E} = 0.46 < \pi/4$. However, the higher starting maximum deviation angle is, the more difficult it becomes to reduce it to a value below $\pi/4$. For this reason, valences strictly larger than 6 may be difficult to realize.

7. Conclusion

In this paper, we present a method for computing frame field with prescribed boundary singularities by modifying boundary alignment constraints. These fields can then be used in the standard hexmeshing pipeline. Simple heuristics can prevent non-meshable configurations that are recurrent in CAD models, both at the boundary and inside the mesh. Our algorithm is simple and adds little computation overhead to standard orthogonal frame field generation algorithms.

The biggest limitation of our method is that we only rely on heuristics for choosing boundary valences. We do not provide a complete characterization of meshable boundary singularities. Moreover, there is no assurance that the interior singularity graph is also valid. The invalid singularity graphs exhibited in Sec. 5.4 are common in CAD models, but they cannot always be treated by simply changing the boundary singularity graph.

References

[1] H. Liu, P. Zhang, E. Chien, J. Solomon, D. Bommes, Singularity-constrained octahedral fields for hexahedral meshing., ACM Trans. Graph. 37 (4) (2018) 93–1.

[2] M. Reberol, A. Chemin, J.-F. Remacle, Multiple approaches to frame field correction for cad models, arXiv preprint arXiv:1912.01248.

[3] N. Pietroni, M. Campen, A. Sheffer, G. Cherchi, D. Bommes, X. Gao, R. Scateni, F. Ledoux, J. Remacle, M. Livesu, Hex-mesh generation and processing: a survey, ACM transactions on graphics 42 (2) (2022) 1–44.

[4] J. Huang, Y. Tong, H. Wei, H. Bao, Boundary aligned smooth 3d cross-frame field, ACM transactions on graphics (TOG) 30 (6) (2011) 1–8.

[5] N. Ray, D. Sokolov, B. Lévy, Practical 3d frame field generation, ACM Transactions on Graphics (TOG) 35 (6) (2016) 1–9.

[6] M. Nieser, U. Reitebuch, K. Polthier, Cubecover–parameterization of 3d volumes, in: Computer graphics forum, Vol. 30, Wiley Online Library, 2011, pp. 1397–1406.

[7] M. Lyon, D. Bommes, L. Kobbelt, Hexex: Robust hexahedral mesh extraction, ACM Transactions on Graphics (TOG) 35 (4) (2016) 1–11.

[8] Y. Li, Y. Liu, W. Xu, W. Wang, B. Guo, All-hex meshing using singularity-restricted field, ACM Trans. Graph. 31 (6). `doi:10.1145/2366145.2366196`.
URL `https://doi.org/10.1145/2366145.2366196`

[9] T. Jiang, J. Huang, Y. Wang, Y. Tong, H. Bao, Frame field singularity correction for automatic hexahedralization, IEEE Transactions on Visualization and Computer Graphics 20 (8) (2013) 1189–1199.

[10] E. Corman, K. Crane, Symmetric moving frames, ACM Transactions on Graphics (TOG) 38 (4) (2019) 1–16.

[11] D. Palmer, D. Bommes, J. Solomon, Algebraic representations for volumetric frame fields, ACM Transactions on Graphics (TOG) 39 (2) (2020) 1–17.

[12] J. Solomon, A. Vaxman, D. Bommes, Boundary element octahedral fields in volumes, ACM Trans. Graph. 36 (4). `doi:10.1145/3072959.3065254`.
URL `https://doi.org/10.1145/3072959.3065254`

[13] D. Desobry, Y. Coudert-Osmont, E. Corman, N. Ray, D. Sokolov, Designing 2d and 3d non-orthogonal frame fields, Computer-Aided Design 139 (2021) 103081.

[14] M. Bracci, M. Tarini, N. Pietroni, M. Livesu, P. Cignoni, Hexalab. net: An online viewer for hexahedral meshes, Computer-Aided Design 110 (2019) 24–36.

[15] F. Ledoux, Mambo dataset (2019).
URL `https://gitlab.com/franck.ledoux/mambo`