BUILDING DIRECTION FIELDS ON THE MEDIAL OBJECT TO GENERATE 3D DOMAIN DECOMPOSITIONS FOR HEXAHEDRAL MESHING

Dimitrios Papadimitrakis[1], Cecil G. Armstrong[1], Trevor T. Robinson[1], Alan Le Moigne[2], Shahrokh Shahpar[3]

[1]School of Mechanical and Aerospace Engineering, The Ashby Building, Queen's University of Belfast, UK, BT95AH, dpapadimitrakis01@qub.ac.uk
[2]Group Business Services – IT, Product Development System, Rolls-Royce plc, Derby, England, UK, DE248BJ
[3]Innovation Hub, Central Technology, Future Methods, Rolls-Royce plc, Derby, England, UK, DE248BJ

ABSTRACT

In this work, a novel method for creating decompositions of general 3D domains, suitable for hexahedral mesh generation is presented. To accomplish this, frames and cross-fields are generated on top of the medial object of the domain. The geometrical and topological information carried by the medial object, together with the directional information of the frames/crosses help analyze the domain. By generating frames and cross fields on medial vertices, edges and faces based on touching vectors, a directional field is constructed on top of the medial object. Based on it, critical lines in the domain called, singularity lines, are identified. Starting from these, a complete line network is created on the interior of the domain. This network is extruded to the boundary in order to create the boundary of high-quality partitioning surfaces that are used to decompose the domain into regions appropriate for a high-quality hexahedral mesh. Examples are given to validate the method.

Keywords: Hex mesh, Medial Object, Frame fields, Cross fields, Singularity, Decomposition

1. RELATED WORK

Generating a structured mesh comprised of hexahedral elements has been a topic of research since the 1970s. Mapping methods were the first approach to be investigated for simple geometries. Either by solving partial differential equations [1] or by using algebraic interpolation techniques [2], [3], they aim to create the mesh by mapping a regular mesh in the parametric space to the physical space. Nowadays these methods are mostly used after the computational domain has been decomposed into mappable sub-regions or blocks.

Plastering [4], is the 3D equivalent of the paving method in 2D [5]. Starting from a boundary quad mesh it gradually constructs hex elements in an advancing front. While good element quality could be achieved close to the boundary, voids are created inside the volume which, often, cannot be meshed with hex elements. To avoid additional constraints, Staten et al. [6] proposed an extension which does not rely on a boundary mesh. By advancing fronts inwards and generating meshes on inner voids that define the boundary mesh, they created hexahedral meshes for rather simple geometries.

One of the most popular hex mesh generation algorithms is sweeping. The basic concept of this algorithm is that a quad mesh on a source face is extruded to a target face along a specified direction [7]. This one to one sweeping algorithm has been further improved through the years [8] to handle complex shapes and even cases where there is more than one source and/or target faces [9], [10]. However, when dealing with many to many type sweeps, there remains a level of domain decomposition/imprinting required to generate the meshable regions.

Price et al. [11] used the 3D medial object to create a decomposition of a solid into sub regions that can be easily meshed by the midpoint subdivision technique. Using the topological and adjacency information the medial object provides, they came up with a set of 13 meshable solid primitives, each with at most 8 faces, with each face having between three and six sides. These primitives are placed along medial vertices, edges and surfaces to create the decomposition of the solid. In their second work [12] the authors extend their work so that geometries with medial surface degeneracies, shallow and concave edges can also be analyzed. However, problems such as N-valent vertices and objects with two sided faces still need to be investigated. LayTracks3D [13] is a more recent work which also relies on the 3D medial object. By combining the medial object and the advancing front method hex-dominant meshes are created. Using medial axis junction curves together with the medial radii, simpler regions called corridors are created. After meshing medial surfaces inside the corridors, a further subdivision is provided with the creation of the so-called tracks. The advancing front method is then used to form the final hex dominant mesh. The author also gives information on how this method can be extended to create all-hex meshes and how it can be used for the meshing of assemblies with the creation of imprints on the medial object. The main drawbacks of these technique are a) the robust computation of the medial surface is still challenging b) a small change in object geometry can radically change the topology of the medial surface and, therefore, the resulting mesh. Finally, one more algorithm that uses skeletons of models to generate hexahedral meshes is that described by Livesu et al. in [14]. In this work high quality hexahedral meshes are generated after a tubular structure that resembles the initial geometry is constructed based on a curve skeleton. However, the method described is limited in models that admit a skeletal representation.

Nieser et al. [15] first proposed a method for generating a hexahedral mesh of a solid subjected to boundary alignment constraints by finding a volume parameterization of a manually created block decomposition of the domain. The notion of a frame field built upon a tetrahedral mesh that guides the parameterization was introduced and several conditions of the so called singularities and the gradient of the frame field where given. The automatic, fast and robust generation and correction of frame fields for general domains was the focus of many works to follow [16]–[20]. Liu et al. [21]recently proposed a method to generate frame fields with manually prescribed singularity graphs, thus giving the opportunity to manually correct topological invalid singularity graphs and then generate a correct frame field. The fundamental properties of hex meshes were first studied by Price et al. [11], [12]. In [21] these properties were used from a slightly different perspective to derive local and global conditions that are necessary for a hex-meshable frame field. Fogg et al. [22] also proposed conditions that a network of singularities must respect. However, the topology of the singularities of the frame fields that can be automatically computed by current methods are still not guaranteed to imply a valid hexahedral mesh even for simple geometries [23].

Having obtained a high-quality frame field, Kowalski et al. [24] suggested that the singularity graph of the frame field can be immediately used to create a domain partitioning and then a mappable block structure to avoid the expensive calculation of a volumetric parameterization. Shang et al. [25] described a different way of using a frame field on a tetrahedral mesh. In their approach, it is used not to guide a volumetric parameterization or a domain partitioning, but to drive the generation of mesh sheets in terms of the spatial twist continuum. This results in a more robust and parallelizable method that does not depend in heavy numerical libraries. However, the final mesh is boundary dependent since it relies on an initial surface quadrilateral mesh. Wang et al. [26] used frame fields in order to guide the creation of dual surfaces using an underlying hex mesh generated by a hex-to-tet method. By isolating singularities and boundary features, these surfaces generate block decompositions that respect the geometry and topology of the domain. However, many of their steps are heuristic and not guaranteed to work in all cases.

Another successful technique for producing all-hex meshes with block structure is that of Polycubes. In such methods, a solid formed from a union of cubes (Polycube) is created and represents the initial model. A hexahedral mesh can be easily created in the Polycube and then mapped back to the model to produces the final hexahedral mesh. High quality meshes are produced robustly even for complex models. However singularities merge close to the boundary reducing the quality in the regions that are important from a simulation perspective. The main challenge in these approaches is the robust generation of the Polycube structure, which remains an open problem. State of the art methods for generating Polycubes are [27] and [28].

Finally, in [29] Lim et al. propose generating multi-block decompositions for 2D domains based on an evolutionary algorithm. In order to generate blocks of high quality a set of fixed boundary points that capture all important geometric features is created. Based on this set, a new set of candidate points on the interior of the domain is generated. Quad meshes are generated based on those points and evaluated in an evolutionary fashion until the best block is derived. The results obtained are comparable to the state-of-the-art. The authors also discuss the extension of the method to 3D.

In this work, a method for the automatic decomposition of a general domain is proposed which combines the merits of the medial object and frame field approach. Continuing the work of [30], it is explained how frames and cross-fields can be constructed on top of the medial object in order to create an internal line network attached to singularity lines. This line network is then extended to guide the generation of partition surfaces that decompose the domain.

2. PRELIMINARIES

Before the method is explained some definitions are given in order to familiarize the reader with the concepts related to the work.

Medial object

The medial object is the locus of the center of an inscribed sphere of maximal diameter as it rolls around the interior of an object. A sphere is maximal if there is no other inscribed sphere that contains it. The medial object is made up of medial surfaces, edges and vertices. In non-degenerate cases each medial surface is constructed by centers of spheres that touch two faces of the object, each medial edge by centers of spheres that touch three faces of the object and each medial vertex by centers of spheres that touch four faces of the object. Another possible configuration is the so called finite contact where the inscribed sphere is in contact with a finite portion of the boundary, like when the sphere rolls along the axis of a cylinder. One more case is that of curvature contact, where the curvature of the inscribed sphere and the minimum curvature of the surface are the same as, for example, at the foci of an elliptical extrusion. The vector that starts from a point on the medial object and ends at a position of contact of the corresponding maximal sphere with the boundary is called touching vector. The medial object has some important properties which are important for mesh generation. These are

- One to one correspondence with the domain.
- Dimensional reduction: the medial object is a 2D object.
- It is orientation independent.
- It identifies parts of the object boundary in geometric proximity.

Mesh singularities

In a quad mesh, singularities are called the nodes of the mesh on which more or less than four mesh edges are connected. The most common situations are those where three or five edges are incident to the node. These are referred to as positive and negative singularities accordingly. Similarly, in a hexahedral mesh, mesh edges where more or less than four hexahedral elements are connected are called singular edges. These edges connect to each other to form singularity lines. The number and the position of singularities affect the quality and the "flow" of the mesh. In general, a small number of singularities is preferred. Singularity lines also describe how the decomposition will look. In this work, two different types of categorization of singularities are used. The first one to describe the type of the singularity regarding the number of mesh elements connected to it. The second one to describe the nature of the singularity with respect to the medial object. It is important to note that the definition of a singularity is different from the work in [31]. Here, boundary edges and vertices where the number of attached elements corresponds to that implied by the dihedral angle are not considered to be singular. For example, a concave boundary edge of 270 degrees where three hex elements join is not considered to be singular.

Positive singularity: Five partition surfaces emanate from the singularity, e.g. Figure 1e. They are highlighted in blue hereafter.
Negative singularity: Three partition surfaces emanate from the singularity, e.g. Figure 9 left. They are highlighted in red hereafter.

Type-1 singularity: The singularity lies on the medial object.
Type-2 singularity: The singularity runs perpendicular to a medial surface.

Figure 1 shows the medial object, the mesh singularities, the partition surfaces and the singularity lines for a block with a hole. In (b) the medial object of the solid in (a) can be seen. In (c) a high-quality block structured mesh is generated. The points on the top surface from which the yellow lines emanate are mesh singularities. The yellow lines are the wireframe of the base-complex of the domain. In (d) the partition surfaces are given in yellow. Finally, in (e) the singularity lines on the interior of the domain are depicted. Partition surfaces emanate from these lines. Although in Figure 1 partition surfaces and singularity lines are depicted as part of an already existing hexahedral mesh with a block structure, in this work we present a method to identify singularities, build partition surfaces based on them, and decompose the domain into regions that can be used to generate a hexahedral mesh.

Partition surfaces

Partition surfaces are surfaces that emanate from singularities. These surfaces imply a partitioning of the domain in smaller regions and are bounded by singularities and by boundary curves. The wireframe of this partitioning is referred in the literature as the base complex of the domain. This is essential to generate a high-quality hexahedral mesh. Five partition surfaces emanate from each positive singularity and three partition surfaces emanate from each negative.

Frames

A frame consists of 3 mutual perpendicular unit vectors together with their opposite vectors $\{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, -\boldsymbol{u}, -\boldsymbol{v}, -\boldsymbol{w}\}$. These vectors represent the orientation in 3D space of a cube that has its faces normal to them. In terms of a mesh, the orientation of each mesh element can be thought to be approximately represented by such a frame or a cube. Figure 2 shows this representation.

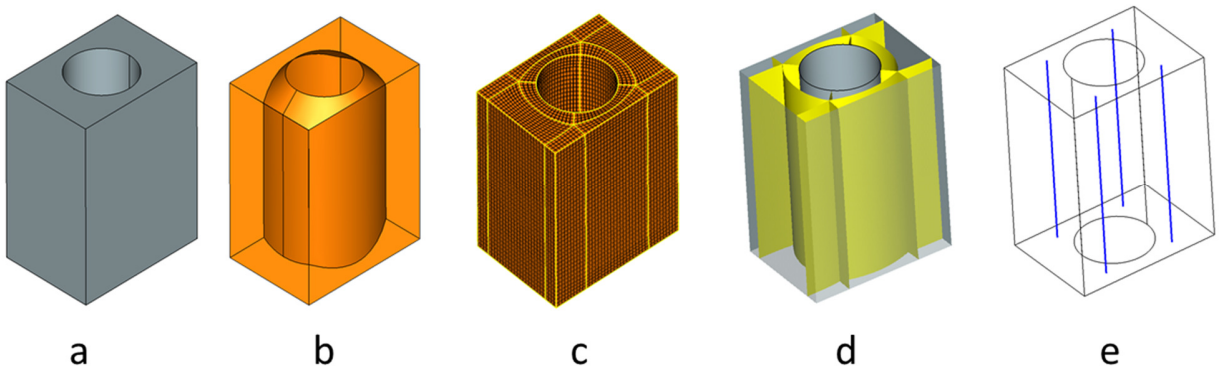

Figure 1: Object (a), medial object (b), Hex mesh (c). Yellow lines define the boundary of the base complex. Partition surfaces (d). Singularity lines (e).

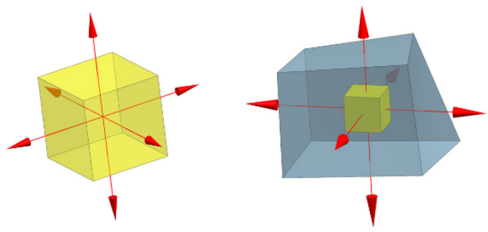

Figure 2: Cube represented by a frame (a). Mesh element represented by a cube and the corresponding frame (b).

3. OVERVIEW OF THE METHOD

Decomposing a general domain into blocks is a tedious task which requires both geometrical and topological information. Although many methods have been proposed in the literature, they all have their strengths and weaknesses. Inspired by previous research on medial object and frame fields, in this work a method is proposed that uses the directional information of a frame along with the structure that the medial object provides in order to decompose a domain into simple regions. Generating high-quality partition surfaces that define those regions is crucial. To achieve that, high-quality singularity lines and boundary lines are identified to form the boundary of the partition surfaces. These lines are produced after a line network is constructed on the interior of the domain based on an analysis that uses the medial object and directions defined through the touching vectors. Contrary to other methods, this work builds the boundaries of partition surfaces from the interior of the domain and is not constrained by a boundary cross-field. The following are the main steps of the method. Figure 3 shows the results after each step.

1) Generate frames along medial edges and medial vertices based on touching vectors.
2) Generate cross-fields on medial surfaces based on those frames.
3) Identify singularity lines lying on the medial object and being normal to it.
4) Trace streamlines that emanate from singularity lines on the medial object to construct a complete line network on the interior of the domain.
5) Project the line network to the boundary to generate boundary lines.
6) Define partition surfaces with the aid of singularity lines and boundary lines.
7) Decompose the domain based on the partition surfaces.

The input of the method is the medial object of the domain with a triangular mesh on each medial surface and the output is the internal line network and a set of partition surfaces. This work focuses mainly on the steps 1-6 aiming in the creation of the interior line network and on the generation of partition surfaces. Based on these surfaces, a domain decomposition is created with the use of the commercial software CADfix.

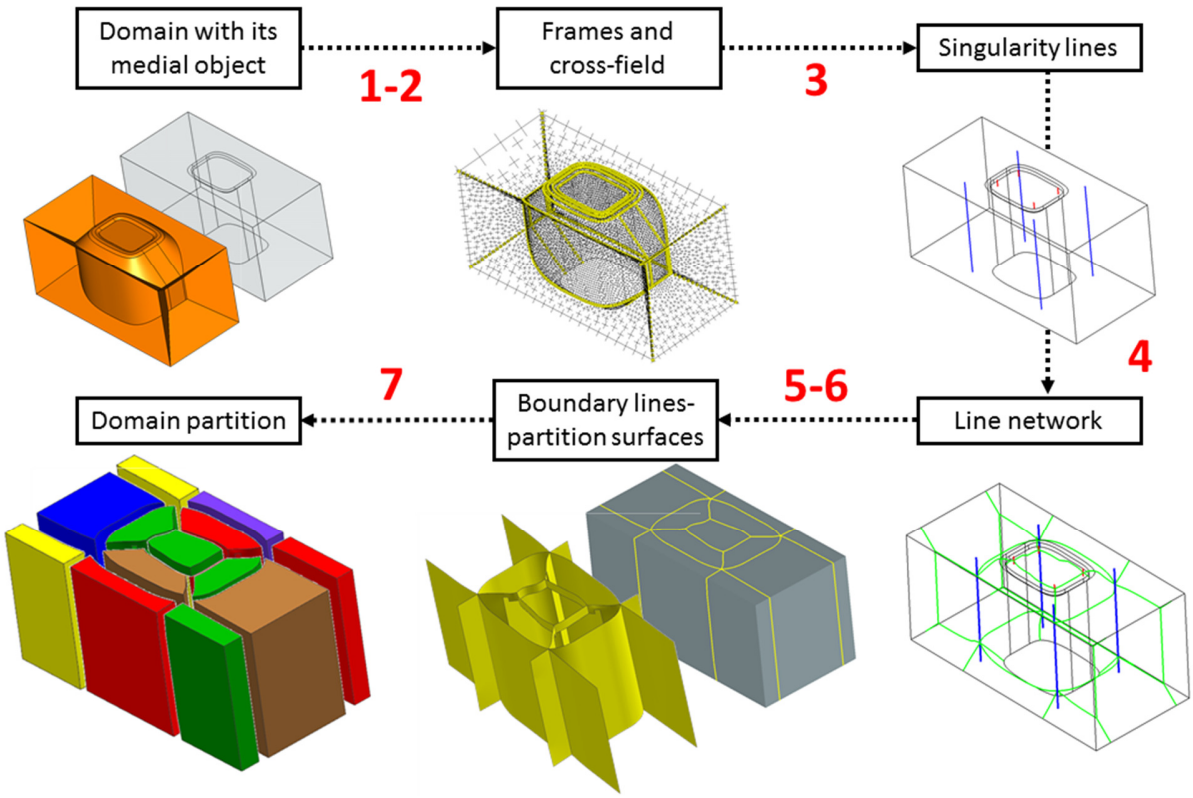

Figure 3: Steps of the method.

4. GENERATING FRAMES

Functional representation of frame:

To generate frames on the medial object, a functional representation of frames restricted to the unit sphere is used which exhibits their 24 symmetries. As described by [Ray et al], this function can be decomposed onto the basis of nine spherical harmonics namely $B = (Y_{4,-4}, Y_{4,-3}, \dots, Y_{4,4})$. Such a decomposition gives the opportunity to describe the function of each frame as $F = Ba$ where the representation vector a has length nine and describes the influence of each harmonic. To find the difference between two frames i and j with different orientations, the integral $\int_{S^2}(F_i(x) - F_j(x))^2 dx$ is calculated on the unit sphere S^2. Since the function basis B is orthonomal, this integral can be further simplified as $\|(a_i - a_j)\|^2$. The representation vector can be expressed by Euler angles that orient it in three-dimensional space relative to a global reference frame. The proximity of two frames can now be simply described by the difference of two vectors.

Frames on medial edges and vertices:

Orienting frames on medial edges and vertices depends on the touching vectors together with the boundary entities that they are associated with. Both the number of the touching entities and their type needs to be considered. Each touching vector represents a boundary entity. If this entity is a face, then the touching vector represents a mesh element on this face which has a rotational degree of freedom. If the entity is an edge or a vertex, then the touching vectors represents a mesh element on this edge or vertex which has no degrees of freedom. Since at each point of a medial edge/vertex, at least three touching vectors exist, a frame must be constructed that represents all the corresponding boundary entities. In order to accomplish that, firstly, a frame must be constructed for each boundary entity. Below it is explained how a frame that fits n vectors can be created.

Frames based on vectors

Let $\vec{t}_i$, $i = (1, \dots, n)$ be n vectors in 3D space. Each pair of vectors $(\vec{t}_i, \vec{t}_j)$, $\{ i \neq j \; and \; \vec{t}_i \cdot \vec{t}_j \neq 0 \}$ defines a plane P_{ij} with normal vector $\vec{t}_n = \vec{t}_i \times \vec{t}_j$. Vectors $\vec{t}_{ni} = \vec{t}_n \times \vec{t}_i and \; \vec{t}_{nj} = \vec{t}_n \times \vec{t}_j$ are sufficient to create two frames $\{\vec{t}_i, \vec{t}_n, \vec{t}_{ni}\}$ and $\{\vec{t}_j, \vec{t}_n, \vec{t}_{nj}\}$ both of which lies on the plane P_{ij}. The first frame corresponds to a representation vectors a_{ij} and the second one to a representation vector a_{ji}

based on the functional representation of frames described before. By doing the same for each pair of non-collinear vectors, $l_i, (l_i \leq n-1)$ frames are created for each vector $\vec{t}_i$, by solving n minimization problems,

$$\min \sum_{j=1}^{l_i} \|a_i - a_{ij}\|^2 \ , i = 1, \dots, n \tag{1}$$

Each representation vectors represents a frame that best fit all frames that were created based on vector $\vec{t}_i$. A set of frames $F = \{a_1, \dots, a_n\}$ is created.
By solving one more minimization problem described by the equation

$$\min \sum_{i=0}^{n} \|a - a_i\|^2 \tag{2}$$

a frame that fits all frames on the set F can be identified. This is the frame that fits all vectors $\vec{t}_i$.

Having defined a way to calculate a frame that best fits n vectors, it can now be explained how frames are generated on medial edges and vertices. For all touching vectors that correspond to a boundary face a frame can be calculated based on equation (1), where the vectors that are fitted are the touching vectors. For each touching vector that corresponds to boundary edge/vertex, a frame can be created again based on equation (1), where the vectors that are fitted are the normal vectors of the boundary faces that are topological parents of the boundary edge/vertex. Having identified a frame for each touching vector, a single frame can be calculated by solving equation (2). This optimization problem can be relaxed if only the more constrained entities are taken into account each time. If for example a medial edge is associated with two boundary faces and one boundary edge the frame that corresponds to the boundary edge can directly be used. Figure 4 shows three examples of a cube/frame that is generated to fit three touching vectors (indicated in red). In (a), a cube perfectly fits the vectors since they are all normal to each other. In this special case the planes defined by each pair of touching vectors corresponds to the dual face of the cube. In (b), two of the touching vectors are collinear with opposite directions while the third one is normal to them. Only two of them are required to generate the frame. In (c), a more general case where the vectors are neither collinear nor normal to each other is depicted.

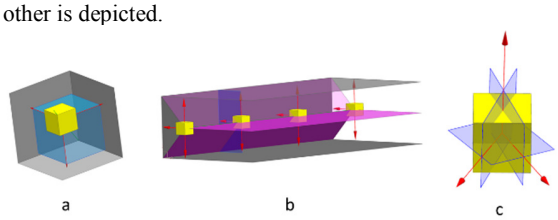

Figure 4: Cube/frame generated by touching vectors for three different configurations. When all touching vectors are normal to each other (a), when two of them are collinear (b), for a general configuration (c).

Frames calculated by this procedure are not forced to be aligned with medial edges. The only thing that constrains them is the orientation of the planes defined by the touching vectors.

5. GENERATING CROSS-FIELDS

In order to create a complete line network on the interior of the domain that will be used as a skeleton to construct partition surfaces, cross-fields are generated on medial surfaces. Such cross-fields define orientations along medial surfaces in the interior of the domain. Methods to generate cross-fields are vast in the literature. In the current work, crosses are generated by a propagation procedure described in [32]. The generation of the cross field will depend on boundary crosses on medial edges and vertices. Such crosses should lie on planes that are tangent to the medial surface at each point. Since boundary frames are generated to approximate all touching vectors there is no guarantee that they will lie on the medial surface. Frames are modified on medial edges and vertices so that they are normal to the medial surface. Let a be the representation vector of a frame F on a medial edge or vertex. Let also $\vec{n}$ be the normal vector of the medial surface. To find the frame F_n with representation vector a_n that is aligned with the normal vector $\vec{n}$ and is closest to the frame F in terms of the functional representation, the function $E = \|a_n - a\|^2$ is minimized. An example of such frames is shown in Figure 5 on the left. The orientation of boundary crosses depends on the information carried by touching vectors. Such information may include hard boundary constraints due to edges or vertices that restrict frame orientation. While crosses are forced to lie on medial surfaces, they are not forced to be tangential on medial edges. This is expected to result in a simpler block topology than that of [13] where meshes on medial surfaces are forced to align with medial edges. Figure 5 shows an example of a 2D cross field generated on a medial surface. When boundary crosses are aligned with medial edges (right), a negative singularity emerges which, in the current method, is avoided (left).

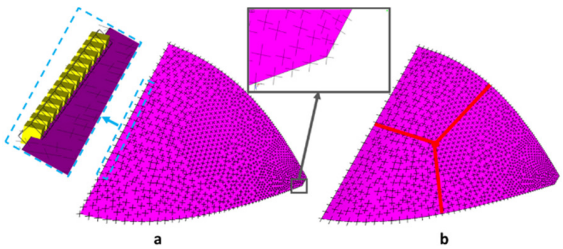

Figure 5: Frames are rotated to be normal to the medial surface. The cross field is not constrained to align with medial edges (a). When it is, more singularities are identified (b).

6. GENERATING LINE NETWORK

Singularities

As discussed in a previous section, singularities can be distinguished by whether they lie on the medial object (Type-1) or they pass through a medial surface (Type-2). This section explains how such singularities can be identified.

Type-1: The procedure of generating singularities that lie on the medial object consists of two steps. In the first step, positions on medial edges where singularities enter the medial object are identified. In the second step, these lines are traced along the medial surfaces, based on the directional information of the cross-field, until they meet another singularity, the boundary, or they connect back to themselves.

By considering a medial edge as being part of one of its parent medial surfaces, touching vectors can be organized in groups of two. Each of them is associated with one boundary entity. By calculating the rotation of adjacent frames calculated by (1), the positions where a singularity is needed to create a well-structured decomposition that respects only those two boundary entities is identified. If, for example, those two touching vectors are associated with two boundary faces, then the singularity will enter the body through the medial edge and run across the medial surface which lies between those two boundary faces. Figure 6 shows how frames that correspond to the touching vectors along two neighboring points a and b on a medial edge can be compared. Here, one combination of three frames out of the four that correspond to each touching vectors is depicted. A similar analysis can be made for all combinations of three touching vectors. If the best fitting frame is calculated on many points along the medial edge based on the touching vectors, the position of the singularity can be visualized as a cube that suddenly "flips" (Figure 7). This "flip" occurs when the angle of the touching vectors passes through 45 or 135 degrees. Each time, a singularity that lies on the corresponding medial surface is sought.

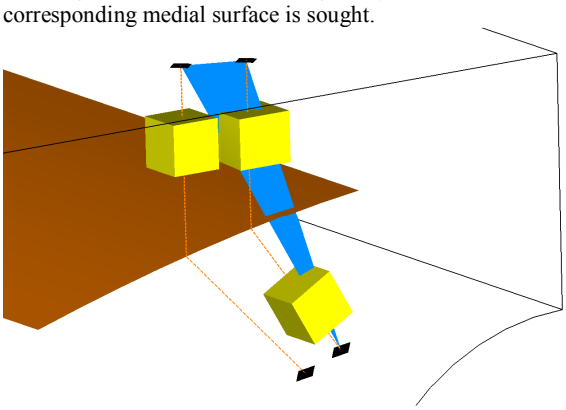

Figure 6: Identifying singularities entering medial surfaces by analyzing neighboring frames.

Having found the position where the singularity enters the medial face, then the singularity is created by tracing along the cross-field on the medial surface. Figure 7 shows an example of an elongated 5 sided block with a curved side edge. A positive singularity is traced along the medial surface.

Type-1 singularities are associated with the boundary of the domain through the medial object. Each end point of the singularity can be projected to a certain boundary entity in order to form a line that starts from the boundary and finishes on the boundary. The only exception is that of singularities that form loops and connect to themselves. The projection depends on the medial edge or the medial vertex on which the end point lies. From all the boundary entities with which this medial edge/vertex is associated with the one whose touching vector forms the smallest angle with the direction of the singularity line is chosen. This guarantees that the singularity will be connected to the boundary smoothly. On Figure 7(left) it can be seen how the end points of a Type-1 singularity are connected to the boundary. Furthermore, a Type-1 singularity is also associated with the boundary entities of the medial surface, on which it lies. If the singularity line is projected to these boundary entities, boundary lines or points are created. These lines/points will be used to generate partition surfaces as will be described in the following section.

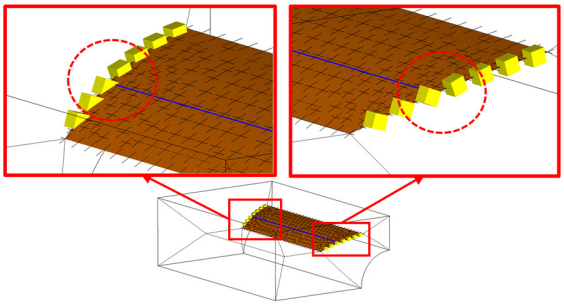

Figure 7: Example of a positive singularity traced on a medial surface. The singularity enters where frame orientations flips.

In a similar way, medial vertices can indicate positions where a singularity will enter the medial object through a medial edge. In this case touching vectors on the medial vertex are organized in groups according to the medial edge that is analyzed. Figure 8 shows an example of an elongated pentagonal prism where the medial edge in the middle carries a positive singularity line. A front view is also given where the frames can be seen to rotate around the medial edge.

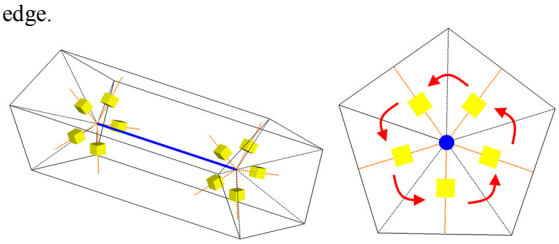

Figure 8: Identifying singularities carried by medial edges. Frames for each touching vector are analyzed.

Type-2: Since crosses on medial surfaces are placed on nodes of an underlying triangular mesh, singularities are identified by analyzing rotations of crosses on each triangular mesh element. Singular points are identified on medial surfaces. Singularity lines are then created by extruding these points to the boundary entities associated with this medial surface. Since singularity lines should end on boundary faces and not on boundary edges or vertices, only medial surfaces that lie between two boundary faces are considered. Figure 9 shows an example of two short prisms. On the left, a negative singularity lies on the interior of the triangular prism. On the right, a positive singularity lies on the pentagonal prism.

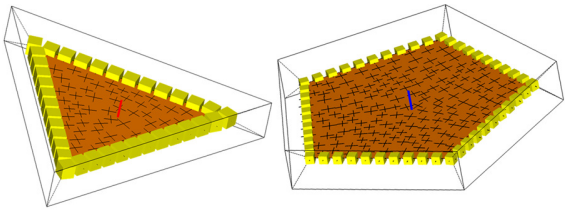

Figure 9: Singularities normal to medial surfaces are identified by cross fields. The simple examples of a triangular and pentagonal prism are illustrated.

Streamlines

In 2D, five/three streamlines emanate from each positive/negative singularity. These streamlines follow the cross-field and form the decomposition of the domain by connecting to other singularities or to the boundary. In 3D, five partition surfaces emanate from each positive singularity line and three from each negative. In this work, instead of directly generating these surfaces, their bounding lines are first created. These lines can then be used to generate the surfaces. Similarly to 2D, streamlines are emanated from each singularity. In Figure 18, these streamlines are depicted in green. Again, Type-1 and Type-2 singularities are treated in a different way. Streamlines are traced on top of the medial object. On Figure 10 it is depicted how partition surfaces on semicircular plate intersect with the medial object of the domain defining lines that emanate from singularities and travel across different medial entities. Since it is the partition surfaces that are to be created it is logical to think reversely and first try to generate the green lines and then, based on them, define the partition surfaces.

Streamline types

Type-1: Each end of a Type-1 singularities can be treated locally as a 2D singularity and five/three streamlines can be initiated there, depending on whether the singularity is positive or negative. Thus, from each Type-1 singularity ten/six streamlines will be traced. These traces can lie on medial faces or medial edges and are traced until they join to a singularity or they meet a boundary edge. The cross-field on the medial object provides the directional information to guide these traces. The initial direction of the traces depends on the local structure of the medial object. Figure 11 shows the directions in one end of a positive singularity. Traces tr1 and tr4 follow the touching vectors and connect to boundary faces BF1 and BF2 respectively. Traces tr2 and tr3 on the other hand, run across medial faces parallel to the boundary faces to create high-quality blocks. Finally, trace tr5 runs across the medial edge that lies between BF1 and BF2. A similar configuration would exist on the other end of the singularity line. It is also depicted how tracing along the medial object's entities provides a more global view. Trace tr5 from the top singularity line connects to the bottom one after travelling along medial edges.

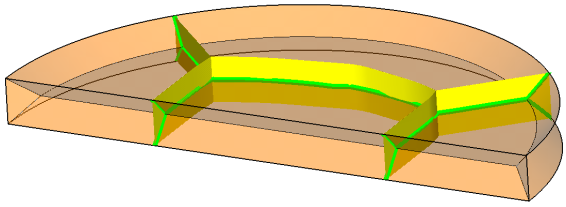

Figure 10: Partition surfaces (yellow) intersecting with the medial object to define streamlines.

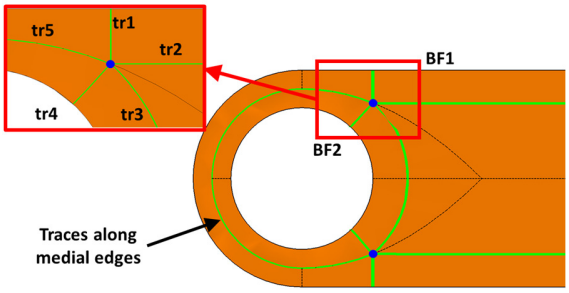

Figure 11: Streamlines are initiated based on the structure of the medial object. They can lie on medial surfaces and medial edges. The structure of the medial object can be crucial in connecting singularities together.

Type-2: Type-2 singularities are treated like in 2D. Thus three/five new traces will be initiated from each negative/positive singular point on a medial surface. The directions of the traces depend on the cross-field, as described in [32], and are traced until they join to another singularity or they meet a boundary edge. In this case, all traces start on the same medial surface. An example is given in Figure 16 which shows streamlines from four negative singularities.

Boundary association

In order to create the boundary lines that will support the partition surfaces, streamlines, like singularities, must be associated with the boundary. These associations depend on the type of the singularity and on the connectivity of the medial object with the boundary.

Each streamline that emanates from a Type-1 singularity will support the generation of one partition surface. The local nature of this partition surface (in the region of the endpoint

of the singularity) can be described by its normal vector $\vec{n} = \overrightarrow{t_1} \times \vec{t}$, where $\overrightarrow{t_1} = \overrightarrow{t_{vb}}$ is the vector that connects the end point of the singularity, to which the streamline is connected to, to the boundary of the domain and $\vec{t}$ is the tangent vector of the streamline at this endpoint. Since a streamline lies on a medial surface or a medial edge, it can be associated with, at least, two boundary entities. Regarding that, one partition surface will be created based on each streamline, choosing to which boundary entities the streamline will be associated to, depends on maintaining a smooth partition surface. If $\overrightarrow{t_v}$ is the touching vector that connects the starting point of the streamline with a boundary entity, then, if $\overrightarrow{t_v} \times \vec{t} \cong \vec{n}$, this boundary entity is associated with the streamline. This condition will ensure that the associations will result in smooth partition surfaces. If, on the other hand, this condition does not hold, then the partition surface will form a dihedral angle along the streamline. In Figure 12 the blue positive singularity can be seen to follow the direction of y-axis. Two of the five streamlines emanating from this singularity are shown in green. Figure 13 shows how streamline tr2 cannot be associated with both boundary entities. Only when projected to the top is the singularity parallel to the yellow surface generated. When it is projected to the right a surface perpendicular to the singularity is created which does not correspond to a partition surface that emanates from the singularity. On Figure 14, on the other hand, tr1 can be associated with both boundary entities since both projections generate surfaces that are parallel to the singularity.

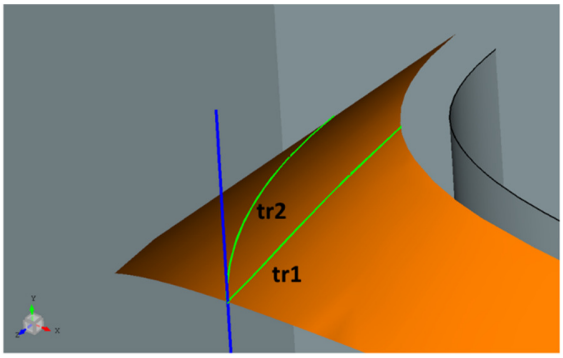

Figure 12: Two of the five streamlines that emanate from the blue positive singularity lie on the same medial surface and thus can be associated to two boundary entities. The association must be done so that the partition surfaces will be parallel to the singularity.

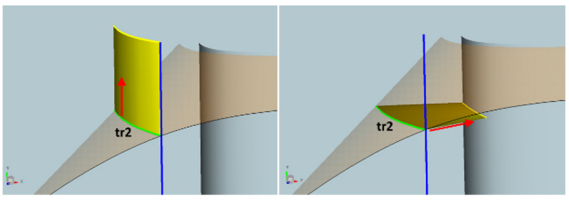

Figure 13: When tr2 is projected on the top, the corresponding surface respects the singularity line (left). When projected to the right the corresponding surface does not. Only projections that respect the singularity are kept so that the resulting partition surfaces will emanate from singularities.

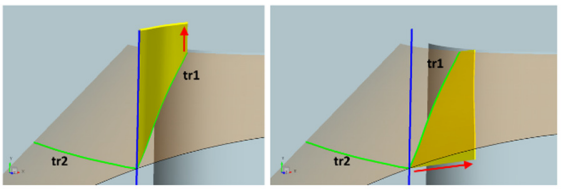

Figure 14: When tr1 is projected on both boundaries, the resulting surfaces are parallel to the singularity line and thus they can both assist on the creation of a partition surface that respect the singularity line. Tr1 is associated with both boundary entities.

Each streamline that emanates from a Type-2 singularity on the other hand is associated with both boundary entities that the medial surface, on which it lies, is associated with.
Tracing streamlines depends on the frames on medial edges and vertices and on the cross fields on medial surfaces. However, since the medial object consists of many different surfaces, edges and vertices, a trace might have to travel along many of them until it connects to another singularity or it meets the boundary. Medial edges and vertices indicate positions where a trace "jumps" from one medial entity to another. The way this transition will take place is important since these lines will form the structure to create partition surfaces. If a transition is smooth, then that will result in generating high-quality partition surfaces.
When a streamline passes from a medial entity to another the association to the boundary must be identified again. From all the boundary entities of the new medial entity, those that maintain the smoothness of the partition surface are chosen. A new trace is then initiated on the new medial entity. It is also important to note that when this transition occurs, more than one new traces might be initiated on different medial entities. For example on a non-degenerate medial edge three medial surfaces are connected. When a trace lying on one of them meets the medial edge, two new traces will be initiated. The association with the boundary must be derived for each new trace. On Figure 15, a trace from a negative singularity can be seen to break into two new traces when it meets a medial edge. It can also be seen how the direction of the new traces are such that the partition surface (light yellow) will continue smoothly.

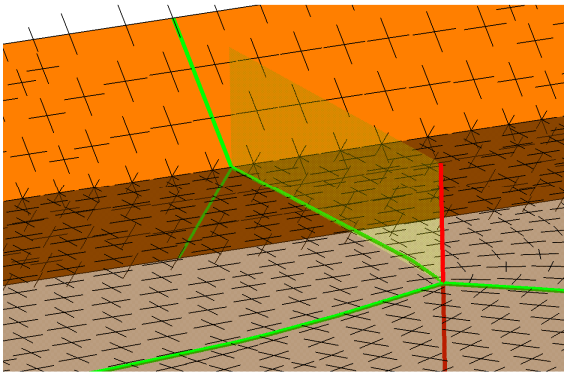

Figure 15: A streamline meeting a medial edge breaks into two new traces.

Local control

Tracing Type-1 and Type-2 singularities and their streamlines on the medial object gives the flexibility to handle different regions separately. Adjustments, corrections and simplifications of the line network can be accomplished separately on each medial surface/edge. If, for example, two streamlines on the same medial surface pass close to each other Figure 16 (a), they can be connected by manipulating only the traces on this medial surface before they propagate to different medial surfaces. The sudden jump on the streamlines in Figure 16 (b) appears because streamlines where forced to join. Lines like those can then be smoothed in order to increase the quality of the final block decomposition. By joining such lines spiral effects can be avoided. Increasing the cross-field density would force streamlines to pass closer to each other and thus produce a smaller step when joined in the cost of a more expensive cross-field computation. Joining lines can guided by a distance parameter that depends on the local radius of the maximum inscribed sphere of the medial object. Here the value of $r/3.0$ was heuristically chosen. Since this radius is a direct measure of the local thickness of the domain, it is a good candidate to decide whether streamlines need to be joined to simplify the final decomposition.

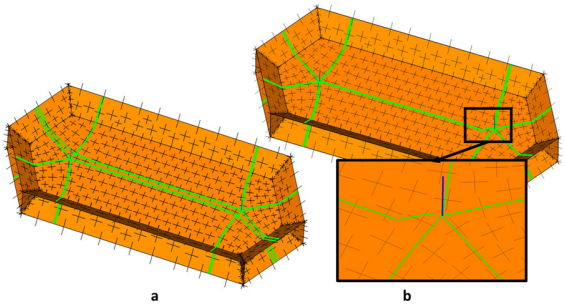

Figure 16: Traces from four Type-2 negative singularities are connected to create a simple topology on the interior. The line network can be locally adjusted by analyzing a selected medial surface.

Line network

After all singularity lines have been identified and all their streamlines have been traced, a complete line network is generated on the interior of the domain. This line network is strongly related to the medial object and each of the lines is associated with certain boundary entities. The network consists of the singularity lines and all the streamlines that emanate from them. Moreover, since the medial object is connected to the boundary, all streamlines are guaranteed to be connected to the boundary, or to another streamline, or to a singularity line. In Figure 17, an example of a model with a tip clearance is given. The tip is not flat and a cavity sits inside the solid tip. Four positive and four negative singularities are identified. The complete internal line network for this geometry is depicted in Figure 18. Streamlines are indicated with green color. All lines are smoothed to support the generation of a high-quality partitioning.

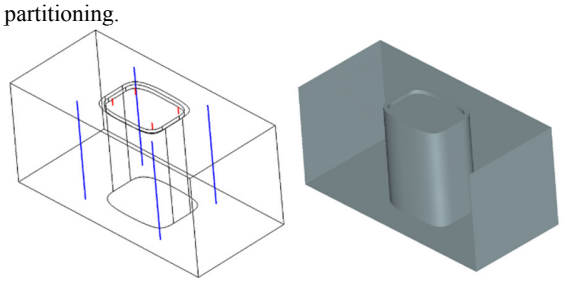

Figure 17: Singularity lines. View of the cavity.

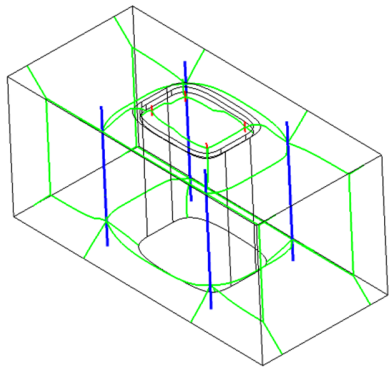

Figure 18: Line network consisting of singularity lines (red and blue) and streamlines (green). Streamlines lie on the medial object.

7. BUILDING DECOMPOSITIONS

Boundary curves

The line network consists of streamlines that lie on the medial object and of singularity lines. Furthermore, since each medial entity is associated with parts of the object's boundary, this association is inherited to the line network as described in section 6. Through this association, boundary curves can be created by projecting each streamline and each

singularity line to the boundary. Boundary curves that correspond to the same streamline will support the construction of one partition surface with the assistance of the singularity line. In Figure 19 an example is given for a positive singularity. Green lines represent trace lines that lie on the medial object. When projected to the boundary they produce the yellow boundary lines. On the right, a detail is given for the region around the concavities. The medial object structure captures such features and, as a result, the boundary lines take them into account too. These boundary lines start from singularities and, either meet the other end of the singularity or join to another singularity. Boundary lines together with singularities create loops of lines to define partition surfaces. In Figure 20, all boundary lines are depicted in yellow. For good quality surfaces to be generated it is important that singularities and boundary curves are smooth and thus a smoothing step is important. During smoothing the connectivity with the boundary should be maintained.

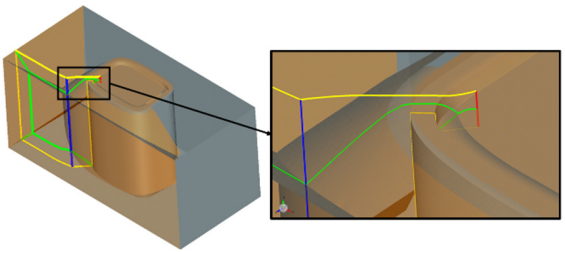

Figure 19: The green streamlines that lie on the medial object are extruded to form the yellow boundary lines (a). The medial object captures all features of the domain and so do the streamlines and the boundary lines. An example of two concavities is given in (b).

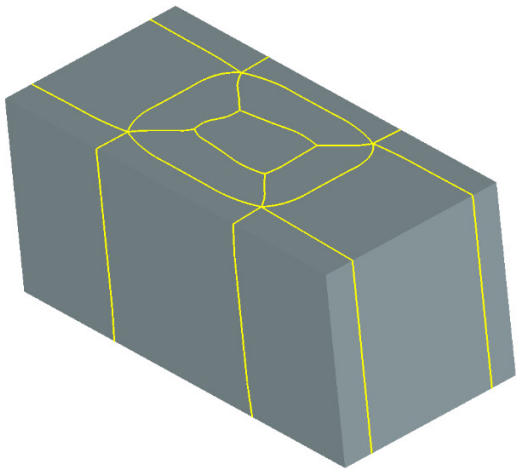

Figure 20: Boundary loops for the model of Figure 17.

Partition surfaces

The boundary curves and singularity lines define the boundaries of high-quality partition surfaces. Since boundary lines were generated using all the features of the domain it is expected that the partition surfaces will respect them too. Figure 21(a) shows partition surfaces on the interior of the model of Figure 17. Figure 21(b) depicts how the partition surface respects the two concavities. Since the medial object captures such features of the domain, so do the streamlines and consequently, the partition surfaces too. In Figure 23(a) the partition surfaces can be seen separately.

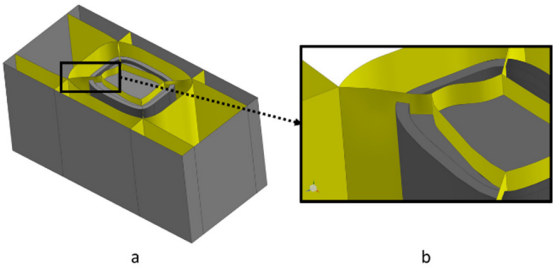

Figure 21: High-quality partition surfaces are generated (a). These surfaces respect features of the domain (b).

Generate regions

The partition surfaces that were generated in the previous step are used to decompose the domain. At this stage, the generation of regions suitable for hexahedral meshing is not fully automated. After automatically generating the line network and the partition surfaces, regions are constructed with the aid of the commercial software CADfix. In order to do that, the intersections between partition surfaces are identified. In Figure 22, an example of such intersections is given. In (a), two partition surfaces from two positive singularities on the left are highlighted in yellow. In (b), partition surfaces from two positive singularities on the right are also shown. In (c) the intersections with the yellow partition surfaces are given in purple. In (d) a top view of the intersection is given. These curves, together with the boundary curves and the singularity lines, define the boundaries of the regions that will be created and hex-meshed. These regions have no further singularities. The generation of two such blocks can be seen on the right of Figure 22. After the regions have been defined, the user can prescribe the density of the mesh through the meshing environment of CADfix and generate a hexahedral mesh. Figure 23(b) shows the decomposition implied by the partition surfaces for the model of Figure 17. It is important to note that although no singularities exist in the regions after decomposition, however, not all of the regions have a simple block structure. Due to concavities, some of the regions need to be decomposed further in order to have only simple blocks. An example of such a region is given in Figure 24(a). Although this region is not mappable, a good quality mesh can be created by sweeping (b).

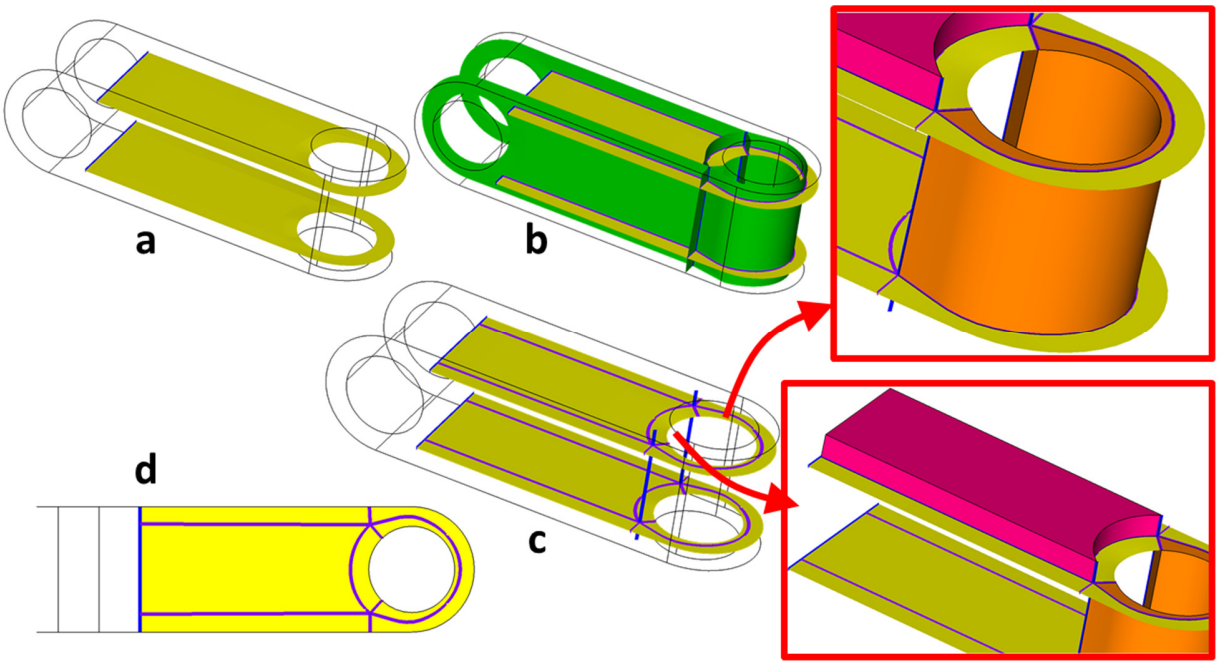

Figure 22: Partition surfaces from two positive singularities (a). Partition surfaces from two other positive singularities are highlighted in green (b). Intersections between them are shown in purple (c) and (d). In the right, two blocks are given. These are bounded by singularity lines, intersections between partition surfaces and boundary lines.

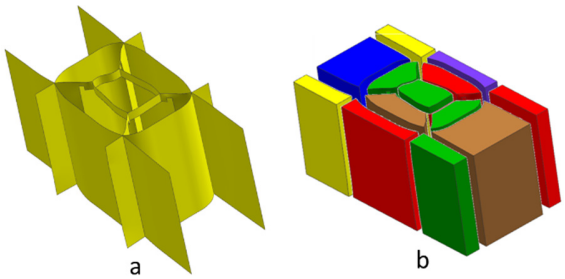

Figure 23: Partition surfaces (a). Decomposition (b).

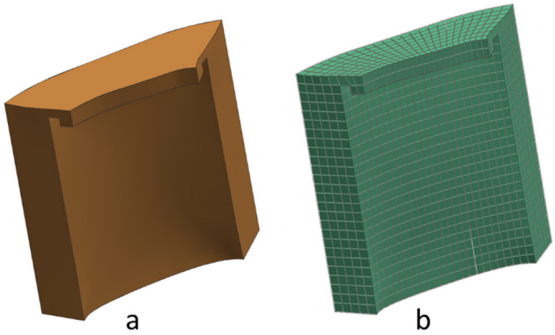

Figure 24: Region that is not a simple block (a). A high-quality mesh generated via sweeping (b).

8. RESULTS

The proposed method has been tested in a series of models and produced decompositions suitable for the generation of all hexahedral meshes. The medial object and all the meshes are created using the commercial software CADfix. Using the provided API the method has been implemented in Python. The first model is a simple thin semicircular plate shown together with its medial object in Figure 25.

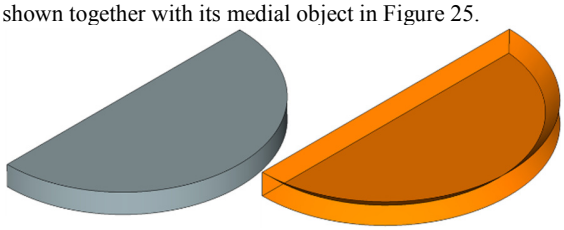

Figure 25: Thin semicircular plate with its medial object

Two negative singularities are identified on the central medial surface. The streamlines are traced until they meet the boundary or they connect to other singularities. Boundary lines are formed by extruding to the boundary these lines and, together with the singularities they form the partition surfaces. A block decomposition is generated and a high-quality mesh is created. Figure 26 shows these steps. Figure 27 shows the second model, a thin circular plate with two holes, together with its medial object. In this model, six positive and two negative singularities are identified. Figure

28 shows the streamlines and the crosses generated on top of the medial object.

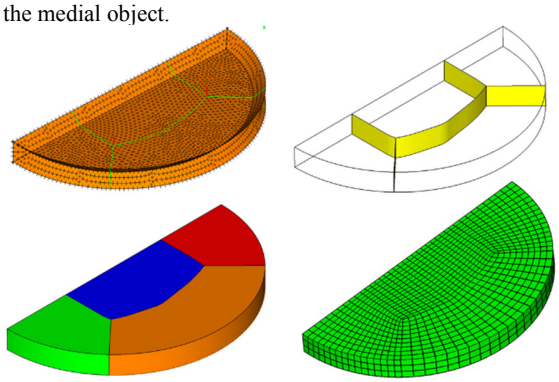

Figure 26: Streamlines (top left), partition surfaces (top right), block decomposition (bottom left) and hex-mesh (bottom right) for the thin semicircular plate.

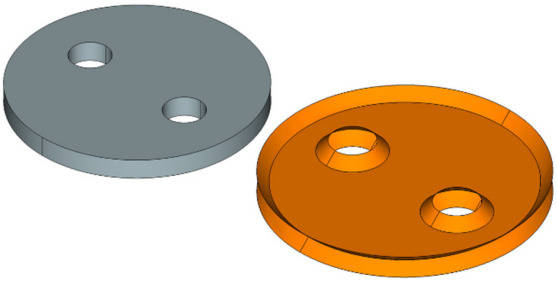

Figure 27: Circular plate with two holes together with its medial object.

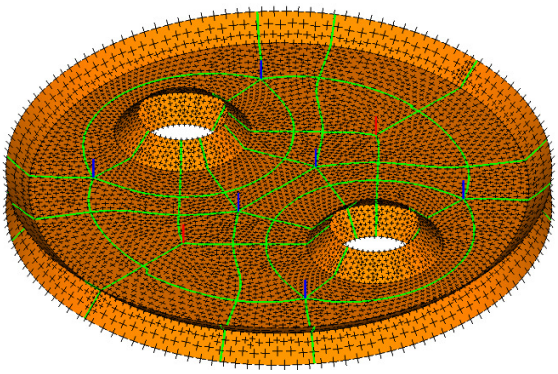

Figure 28: Cross-fields on the medial object and streamlines traced. Six positive and two negative singularities are identified.

The boundary lines that are created based on these streamlines are depicted in Figure 29.

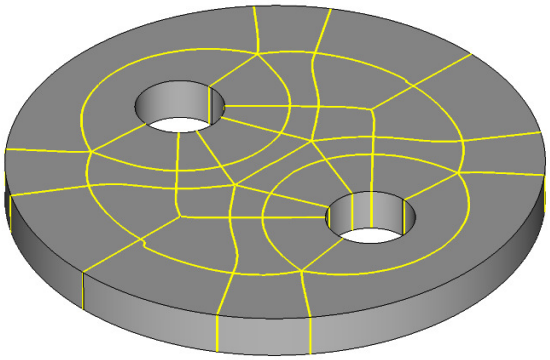

Figure 29: Boundary lines generated for the plate with the two holes.

A more complex model is shown in Figure 30. The concave features of this model make the decomposition quite challenging for a non-expert.

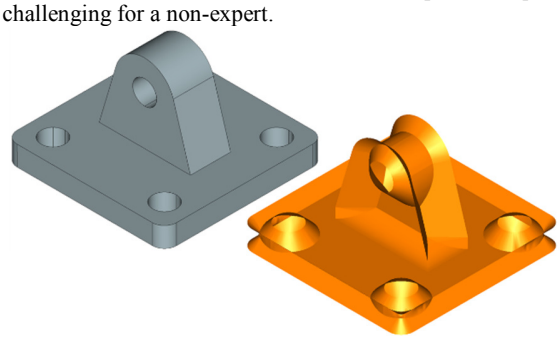

Figure 30: Solid model together with its medial object.

Two details of the medial object are given in Figure 31 to understand how the concavities affect the medial object. In Figure 31(a), the medial surface that maps the top to the bottom boundary face is highlighted. The loop formed by the four concave boundary edges results in a rectangular hole on the medial surface. As it can be seen in Figure 31(b), the medial object curves around the concavities. This proves to be really helpful since it will allow streamlines to curve around the concavities.

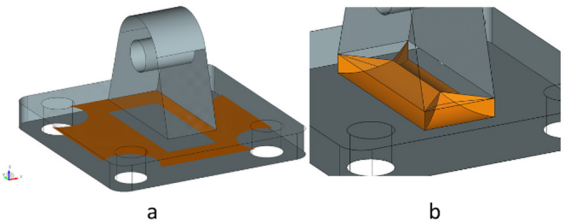

Figure 31: Concavities result in a hole in a medial surface (a). A detail of the medial surfaces around the concave boundary edges (b).

Figure 32 shows the cross-field on the medial surface of Figure 31(a). Four positive singularities are identified and the corresponding streamlines are shown in green. Although

the streamlines belong to the same medial surface, the hole created by the concavity separates the singularities on the left from the singularities on the right. However, the medial object's connectivity around the concavities, shown in Figure 31(b), allows the streamlines to be connected on different medial surfaces.

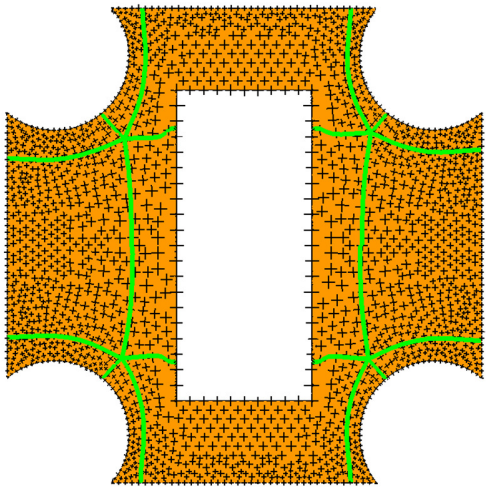

Figure 32: Concave edges result in a hole on the medial surface. The streamlines on the left side are not aware of those on the right.

In Figure 33 it can be seen how the structure of the medial object around the concavity makes it possible for the streamlines to connect to each other. It can also be seen how they connect to another blue positive singularity on the top. A detail is also given which shows how the streamline breaks into two near the concavity. One continues and connects to the other streamline coming from the right while the second one turns 90 degrees following the concave boundary until it connects to the positive singularity on the top.

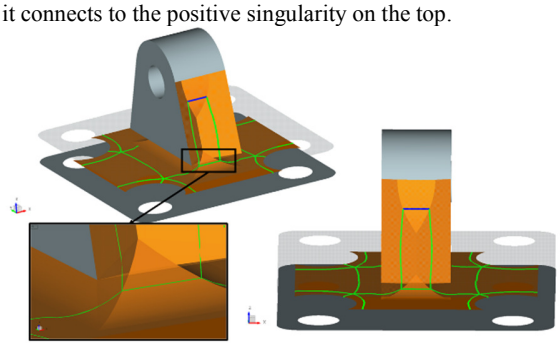

Figure 33: Streamlines traced from medial surface to medial surface on the medial object connect to each other although the concavity separated them. A streamline can be seen to "break" in two around the concavity.

Streamlines like these are "geometry aware" and help in generating a decomposition that captures all important features of the domain. In Figure 34 the partition surface that corresponds to the streamlines of Figure 33 is given. The positive singularities on which it is attached are shown in blue.

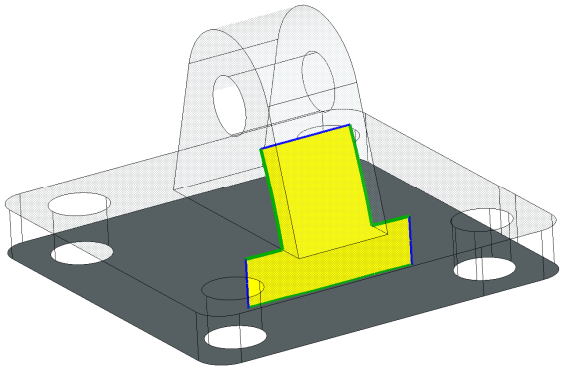

Figure 34: Partition surface attached to three positive singularities respecting the concave boundary edges.

The final decomposition of this model is given in Figure 35. Most regions are simple blocks. However, around the concavities the regions are a bit more complex (Figure 36(a)), and were created based on surfaces like the one depicted on Figure 34. A good quality mesh can still be created by sweeping like that shown in Figure 36(b) or alternatively the region could be further decomposed as shown in Figure 36(c). However, no further singularities exist in these regions. At this stage, such regions are not treated automatically. It is part of on-going research to automatically detect concave features and construct extra partition surfaces.

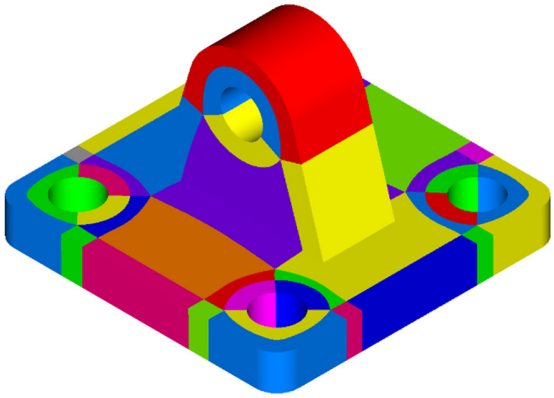

Figure 35: Decomposition of the model of Figure 27.

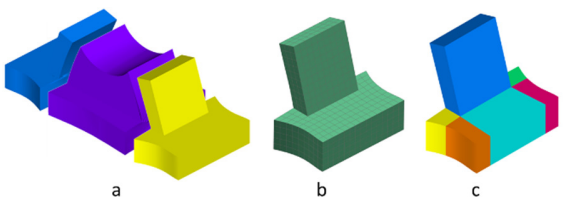

Figure 36: Non-simple block regions (a), mesh of yellow region created by sweeping (b), further decomposition into blocks of yellow region (c).

The mesh for the model of Figure 30 is given in Figure 37.

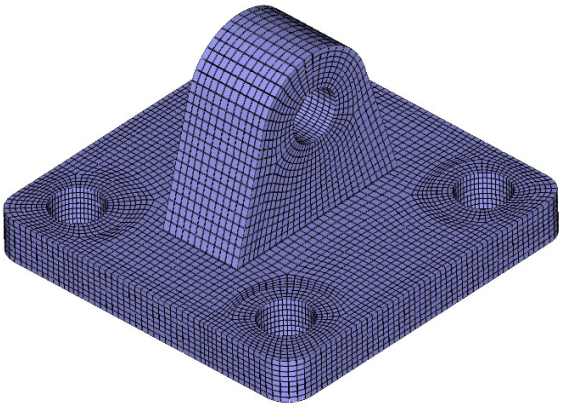

Figure 37: Mesh for the model of Figure 30. (Min. / Avg. Scaled Jacobian: 0.68 / 0.97)

Finally streamlines for one more example are given in Figure 38 (b). Partition surfaces for this geometry where shown in Figure 22. Four positive singularity lines are identified. The block decomposition that was generated based on the partition surfaces and the final hexahedral mesh are given in Figure 39.

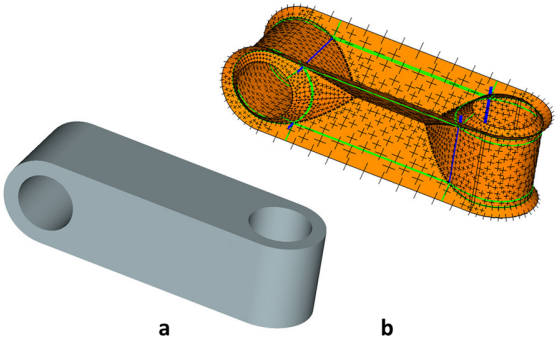

Figure 38: Model with two through holes in orthogonal directions (a). The cross-field on the medial object together with singularity lines and streamlines (b).

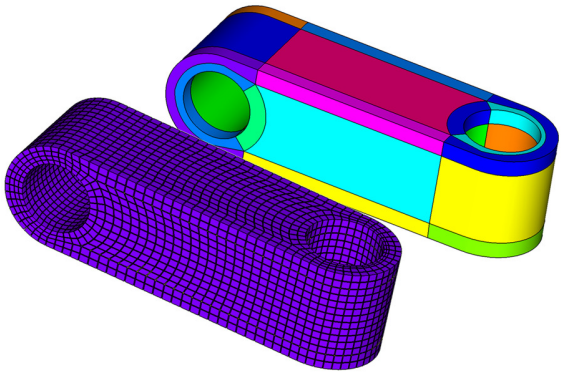

Figure 39: Final block decomposition and hexahedral mesh for the model of Figure 38. (Min. / Avg. Scaled Jacobian: 0.69 / 0.94)

9. DISCUSSION

Generating high-quality partition surfaces to create decompositions sufficient for all-hexahedral meshing can be proven to be difficult. Capturing all geometrical and topological features of the domain is challenging. In the current work, this issue was addressed by first generating a set of boundary curves and singularity lines which bound those surfaces. These lines are generated by projecting on the boundary a line network created on top of the medial object of the domain. Since the medial object captures all geometrical and topological features of the domain, the partition surfaces will respect them too. Furthermore, since the medial object by itself separates the domain into regions, a line network with simple topology can be created by modifying it locally on each medial surface. This simplifies the decomposition. Instead of manipulating partition surfaces, changes can now be done on curves on the medial object. Furthermore, by placing singularities on the medial object, they are pushed to the interior of the domain, far from the boundary. This ensures that the decomposition and the final mesh will have high quality close to the boundary. This is, in general, preferable for numerical simulations. Moreover, since singularity lines do not depend on an underlying tetrahedral mesh, noisy patterns that are common in frame-field methods are avoided and in general smooth lines are created. By generating frames based on touching vectors and not by aligning them with medial edges, unnecessary singularities are avoided. In general, compared to the current state of the art where the fixed cross-field topology of the boundary restricts the decomposition process, an attempt is made to construct it on cross-fields built on the interior and the project it to the boundary.

The medial object of the domain proves to be a really helpful framework on which singularities can be traced. At the same time, it captures efficiently all geometric features of the geometry and thus can provide important information in reasoning a high-quality decomposition. However, the medial object by itself is difficult to construct and no method exists that can guarantee a robust and efficient computation of the medial object of an arbitrary complex domain.

This work aims, not only on describing a method by which arbitrary domains can be decomposed for hexahedral meshing, but also to bring together the merits of two different methods. This can help to gain further knowledge regarding the long-standing problem of block-decomposition. The medial object provides a topological and geometrical connectivity on the interior of the domain that could be beneficial. Having understood what exactly is needed, then a, more easily generated, imprecise medial object could be used to assist existing methods that rely on frame-fields.

The method was tested on a number of models and produced good quality partitioning of the domains. However, there are still issues that need to be addressed. Concave features can result in decompositions that are not simple blocks like those shown in Figure 24 and Figure 36. Extra partition surfaces are needed to fully decompose the domains into blocks. Generating such surfaces by exploring the imprints of the concavities on the medial object is a topic of future research.

At this stage, the lack of a robust method to generate the exact medial object for every possible domain is the main drawback of the method as it relies on it. Handling concavities to generate pure block regions is another issue that needs to be addressed. Finally, degenerate cases where multiple points on the boundary map to a single point/line on the medial object (like for example a sphere, a cylinder or a blended convex edge) need to be further examined to make the method more complete.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the financial support provided by Innovate UK via the GEMinIDS (project 113088), a UK Centre for Aerodynamics project. The authors acknowledge Rolls-Royce for granting permission to publish this paper.

REFERENCES

[1] J. Thompson, Z. U. a Warsi, C. W. Mastin, and J. F. Thomson, *Numerical grid generation: foundations and applications*, vol. 1. North-Holland, 1985.

[2] W. J. Gordon and C. A. Hall, "Construction of curvilinear co-ordinate systems and applications to mesh generation," *Int. J. Numer. Methods Eng.*, vol. 7, no. 4, pp. 461–477, Jan. 1973.

[3] R. E. Smith and L. E. Eriksson, "Algebraic grid generation," *Comput. Methods Appl. Mech. Eng.*, vol. 64, no. 1–3, pp. 285–300, Jan. 1987.

[4] S. Cannan, "Plastering - A new approach to automated, 3-D hexahedral mesh generation," in *33rd Structures, Structural Dynamics and Materials Conference*, 1992.

[5] T. D. Blacker and M. B. Stephenson, "Paving: A new approach to automated quadrilateral mesh generation," *Int. J. Numer. Methods Eng.*, vol. 32, no. 4, pp. 811–847, Jun. 1991.

[6] M. L. Staten, R. A. Kerr, S. J. Owen, T. D. Blacker, M. Stupazzini, and K. Shimada, "Unconstrained plastering-hexahedral mesh generation via advancing-front geometry decomposition," *Int. J. Numer. Methods Eng.*, vol. 81, no. 2, pp. 135–171, Jan. 2010.

[7] P. M. Knupp, *Next-Generation Sweep Tool: A Method for Generating All-Hex Meshes on Two-and-One-Half Dimensional Geometries.* 1998.

[8] S. Cai and T. J. Tautges, "One-to-one sweeping based on harmonic S-T mappings of facet meshes and their cages," *Eng. Comput.*, vol. 31, no. 3, pp. 439–452, Jul. 2015.

[9] M. A. Scott, S. E. Benzley, and S. J. Owen, "Improved many-to-one sweeping," *Int. J. Numer. Methods Eng.*, vol. 65, no. 3, pp. 332–348, Jan. 2006.

[10] E. Ruiz-Gironés, X. Roca, and J. Sarrate, "A new procedure to compute imprints in multi-sweeping algorithms," in *Proceedings of the 18th International Meshing Roundtable, 2009*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 281–299.

[11] M. A. Price, C. G. Armstrong, and M. A. Sabin, "Hexahedral mesh generation by medial surface subdivision: Part I. Solids with convex edges," *Int. J. Numer. Methods Eng.*, vol. 38, no. 19, pp. 3335–3359, Oct. 1995.

[12] M. A. Price and C. G. Armstrong, "Hexahedral mesh generation by medial surface subdivision: Part ii. solids with flat and concave edges," *Int. J. Numer. Methods Eng.*, vol. 40, no. 1, pp. 111–136, Jan. 1997.

[13] W. R. Quadros, "LayTracks3D: A new approach for meshing general solids using medial axis transform," *Compuer-Aided Design.*, vol. 72, pp. 102–117, 2016.

[14] M. Livesu, A. Muntoni, E. Puppo, and R. Scateni, "Skeleton-driven Adaptive Hexahedral Meshing of Tubular Shapes," *Comput. Graph. Forum*, vol. 35, no. 7, pp. 237–246, 2016.

[15] M. Nieser, U. Reitebuch, and K. Polthier, "CUBECOVER - Parameterization of 3D volumes," *Comput. Graph. Forum*, vol. 30, no. 5, pp. 1397–1406, Aug. 2014.

[16] J. Huang, Y. Tong, H. Wei, and H. Bao, "Boundary aligned smooth 3D cross-frame field," in *ACM Transactions on Graphics*, 2011, vol. 30, no. 6, p. 1.

[17] J. Huang, T. Jiang, Y. Wang, Y. Tong, and H. Bao, "Automatic Frame Field Guided Hexahedral Mesh Generation, Technical Report," 2012.

[18] Y. Li, Y. Liu, W. Xu, W. Wang, and B. Guo, "All-hex meshing using singularity-restricted field," *ACM Transactions on Graphics*, vol. 31, no. 6, p. 1, Nov. 2012.

[19] N. Ray, D. Sokolov, and B. Lévy, "Practical 3D Frame Field Generation," *ACM Trans. Graph.*, vol. 35, no. 6, p. 233:1--233:9, Nov. 2016.

[20] J. Solomon, A. Vaxman, and D. Bommes, "Boundary Element Octahedral Fields in Volumes," *ACM Transactions on Graphics*, vol. 36, no. 3, May 2017.

[21] H. Liu, P. Zhang, E. Chien, J. Solomon, and D. Bommes, "Singularity-constrained Octahedral Fields for Hexahedral Meshing," *ACM Transactions on Graphics*, vol. 37, no. 4, p. 93:1--93:17, Jul. 2018.

[22] R. T. Fogg HJ, Sun L, Makem JE, Armstrong CG, "Singularities in structured meshes and cross-fields," *Computer-Aided Design*, vol. 105, pp. 11–25, 2018.

[23] F. L. Ryan Viertel, Matthew L Staten, "Analysis of Non-Meshable Automatically Generated Frame Fields.," Albuquerque, NM (United States), 2016.

[24] N. Kowalski, F. Ledoux, and P. Frey, "Smoothness driven frame field generation for hexahedral meshing," *CAD Computer-Aided Design.*, vol. 72, pp. 65–77, Mar. 2016.

[25] F. Shang, Y. Gan, and Y. Guo, "Hexahedral mesh generation via constrained quadrilateralization," *PLoS One*, vol. 12, no. 5, p. e0177603, May 2017.

[26] R. Wang, C. Shen, J. Chen, H. Wu, and S. Gao,

"Sheet operation based block decomposition of solid models for hex meshing.," *Computer-Aided Design.*, vol. 85, pp. 123–137, 2017.

[27] J. Gregson, A. Sheffer, and E. Zhang, "All-hex mesh generation via volumetric polycube deformation," *Eurographics Symp. Geom. Process.*, vol. 30, no. 5, pp. 1407–1416, Aug. 2011.

[28] X. Fang, W. Xu, H. Bao, and J. Huang, "All-hex Meshing Using Closed-form Induced Polycube," *ACM Transactions on Graphics*, vol. 35, no. 4, p. 124:1--124:9, Jul. 2016.

[29] S. (2018). A. blocking of S. using E. A. Lim, C.W., Yin, X., Zhang, T., Goh, C.K., Alejandro, L.B., Moreno, & Shahpar, "Automatic blocking of Shapes using Evolutionary Algorithm.," in *Proceeding of the 27th International Roundtable*, 2018.

[30] D. Papadimitrakis, C. G. Armstrong, T. T. Robinson, S. Shahpar, and A. Le Moigne, "A Combined Medial Object and Frame Approach to Compute Mesh Singularity Lines," in *27th International Meshing Roundtable.*, 2018.

[31] J. S. and D. B. Liu, Heng, Paul Zhang, Edward Chien, "Singularity-constrained octahedral fields for hexahedral meshing," *ACM Transactions on Graphics 37*, p. 93:1-93:17, 2018.

[32] H. J. Fogg, C. G. Armstrong, and T. T. Robinson, "Automatic generation of multiblock decompositions of surfaces," *Int. J. Numer. Methods Eng.*, vol. 101, no. 13, pp. 965–991, Mar. 2015.