Tensor-Guided Hex-Dominant Mesh Generation with Targeted All-Hex Regions

Ved Vyas and Kenji Shimada

Department of Mechanical Engineering
Carnegie Mellon University, Pittsburgh, PA, USA
ved@cmu.edu, shimada@cmu.edu

Abstract. In this paper, we present a method for generating hex-dominant meshes with targeted all-hex regions over closed volumes. The method begins by generating a piecewise-continuous metric tensor field over the volume. This field specifies desired anisotropy and directionality during the subsequent meshing stages. Meshing begins with field-guided tiling of individual structured hexahedral fronts wherever suitable and in regions of interest (ROI). Then, the hexahedral fronts are incorporated into an existing hex-dominant meshing procedure, resulting in a good quality hex-dominant mesh. Presently, many successful hex meshing methods require significant preprocessing and have limited control over mesh directionality and anisotropy. In light of this, hex-dominant meshes have gained traction for industry analyses. In turn, this presents the challenge of increasing the hex-to-tet ratio in hex-dominant meshes, especially in ROI specified by analysts. Here, a novel three-part strategy addresses this goal: generation of a guiding tensor field, application of topological insertion operators to tile elements and grow fronts towards the boundary, and incorporation of the fronts into a hex-dominant meshing procedure. The field directionality is generated from boundary information, which is then adjusted to specified uniform anisotropy. Carefully placed streamsurfaces of the metric field are intersected to shape new elements, and the insertion operators maintain mesh integrity while tiling new elements. Finally, the effectiveness of the proposed method is demonstrated with a non-linear, large deformation, finite element analysis.

1 Introduction

In this paper, we present a method for generating hex-dominant meshes with targeted all-hex regions over closed volumes. A three-part strategy is employed to achieve this: generation of a guiding metric field, application of topological insertion operators to insert elements and grow the hex fronts outward from starting interior points, and indirect incorporation of the fronts into a final hex-dominant mesh. The field governs element directionality and anisotropy over the volume for the tiling and final meshing stages. During the tiling stage, insertion operators maintain mesh integrity and attempt to prevent the mesh from marching into itself as they place new hexes. New

elements are formed from the intersections of strategically placed stream-surfaces of the metric field. The nodes of the fronts are then incorporated into a rectangular bubble-packing process [24] as fixed bubbles, from which a tetrahedral and then final hex-dominant mesh are obtained.

The method proposed in this work possesses many salient features. First, a new method is described for constructing geometry-based tensor fields with boundary sensitivity and user-input anisotropy. These fields can be applied to any meshing algorithm which is suitably equipped. Second, both the tiling and bubble-packing steps produce valid meshes that conform to these fields. The tiling process is also flexible enough to start on the interior, and with the help of the field and boundary conformity operations, capture the boundary. Finally, the incorporation of the tiled hex fronts into the hex-dominant process allows targeting of all-hex regions in ROIs.

Hexahedral meshes are commonly preferred in many types of 3D Finite Element Analyses. However, robust all-hexahedral mesh generation has not yet been achieved with the guarantees, quality, and control of arbitrary anisotropy/directionality provided tetrahedral and hex-dominant meshing schemes.

Current practical methods for direct hexahedral mesh generation are at most semi-automatic, requiring model decomposition and application of topology specific methods such as (sub)mapping and sweeping among others [1]. Other methods, such as the grid based methods of Schneiders et al. [2] or Zhang et al. [3], are only subject to topology constraints if boundary features (for instance curves and vertices) need to be captured. Although they do provide some degree of anisotropy and size gradation using templates, there is no mechanism for controlling arbitrary directionality. Furthermore, 3D advancing front methods such as Plastering [4] start at the boundary while placing elements. Unfortunately, this leads to unmeshable voids and stitching problems between fronts. Unconstrained plastering is a promising extension that aims to avoid these difficulties, but is still under development [5].

In light of this, hexahedral-dominant meshing schemes have gained traction for semi- and fully automatic mesh generation. Rectangular bubble-packing, by Yamakawa and Shimada [24], is an indirect hex-dominant method that is capable of generating high quality hex-dominant meshes. The resulting meshes conform to a specified tensor field and have good hexahedral volume ratios. The method begins by packing rectangular solid cells and then generating a tetrahedral mesh using the cells' centers. This is followed by merging groups of tetrahedra into good hexahedral elements. Optionally, tets can be merged into prisms and possibly pyramids to produce a fully conformal mesh. That is, hanging edges which are present in a mixed hex-tet mesh will be eliminated through the use or prisms and pyramids. Alternatively, special subdivision templates have been developed to produce conformal meshes without pyramids [29].

From the inherent drawbacks and strengths of the methods mentioned here, the authors have reached the observation that a successful hex-dominant meshing strategy should include:

1. Using fields to control directionality and anisotropy.
2. Tailoring such fields to be boundary aligned and to incorporate additional directionality and anisotropic size specifications.
3. Meshing from the inside out by starting at a suitable location(s).
4. Deciding marching directions for meshing ahead of time (via the field).
5. Meshing element by element in such a way that the mesh remains topologically valid and does not march into itself.
6. Complementing difficult regions for all-hex with good quality hex-dominant regions.

Some of these points, such as the need for a boundary sensitive overlay for grid-based methods, have been noted by Blacker for all-hex meshing [6]. Furthermore, field generation has been performed for application to other methods such as BubbleMesh [7, 8, 9].

The method presented here attempts to utilize these desired characteristics through field generation and application of topological insertion operators to insert elements and grow the mesh outward from a starting interior point. The metric field incorporates boundary normal information and user-input uniform anisotropy, and is capable to providing information on element shaping and marching directions for meshing. This allows for a relatively simple tiling algorithm based on topological insertion operators. The insertion operators maintain mesh integrity while attempting to prevent the mesh from marching into itself. They also govern the shaping and via specific intersections of streamsurfaces of the eigenvector fields that are obtained from the metric field.

Due to the difficulties posed by applying this process to entire volumes, the proposed method attempts to incorporate the traced hex elements into a hex-dominant mesh. The resulting mesh is boundary conformal, approximately metric conformal, good quality, and has increased hex-to-tet ratios in the tiled regions. Therefore, predominantly hex meshes can be created in ROIs, for instance where boundary conditions are applied or important stresses/quantities are present.

Presently, the proposed method operates on closed volumes with a facet-based boundary representation equipped with sufficient guiding feature curves. It can be applied to analytic-surface based CAD models using the graphics facets and feature curve extraction. In addition, in this paper only fields of uniform anisotropy are considered. The basic approach, however, can be extended to non-uniform anisotropic fields.

The paper is organized as follows: Section 2 discusses related work that leads up to and supports the proposed method. Section 3 presents an overview of the methods used, and also covers preliminaries. Generation of metric tensor fields is described in Section 4, and meshing via hex tiling and packing

is covered in Section 5. Section 6 contains results and our discussion, and the paper is concluded with a brief look at further work in Section 7.

2 Related Work

Riemannian metric tensors have become a popular means of anisotropic mesh control [7, 8, 10]. In the context of 3D mesh generation, such second-order tensors can be represented as 3×3, symmetric, positive-definite matrices in a local coordinate system. Given a metric tensor M, the dot (inner) product of two vectors $\mathbf{x}$ and $\mathbf{y}$ can be written as $\mathbf{x}^{\mathrm{T}}\mathsf{M}\mathbf{y}$ [11]. This can be used to obtain the norm of a vector under the metric via $\left(\mathbf{x}^{\mathrm{T}}\mathsf{M}\mathbf{x}\right)^{1/2}$, as well as the angle between two vectors. Additionally, the length l of a parametric curve $\mathbf{r}(t)$ where $t \in [t_0, t_1]$, can be expressed as:

$$l = \int_{t_0}^{t_1} \left[M_{ij} \frac{dr^i}{dt} \frac{dr^j}{dt} \right]^{1/2} dt = \int_{t_0}^{t_1} \left[\dot{\mathbf{r}}(t)^{\mathrm{T}} \mathsf{M}\left(\mathbf{r}(t)\right) \dot{\mathbf{r}}(t) \right]^{1/2} dt \,, \tag{1}$$

where M can vary spatially and $i, j = 1, 2, 3$.

A metric tensor represented by a matrix M has a decomposition given by the spectral theorem [12]:

$$\mathsf{M} = \mathsf{Q}\Lambda\mathsf{Q}^{\mathrm{T}} \,. \tag{2}$$

Here, Q is an orthogonal eigenvector basis and Λ is a diagonal eigenvalue matrix. Using the ellipsoid interpretation of the metric, the eigenvalues are the inverse-squares of the ellipsoid's axis lengths.

These properties have made metrics an appealing method for representing mesh directionality (eigenvector directions) and anisotropy (corresponding eigenvalues). Researchers have applied metrics to generate and adapt meshes [7, 13].

There is a body of work directed toward generating directional fields on surfaces for the purpose of texture synthesis, painterly rendering, and surface meshing; refer to [14, 15].

Other algorithms relevant to this work include advancing front methods and grid-based methods, as previously mentioned. In addition to this, our underlying element-shaping strategy is inspired by the previous work of Alliez et al. [16] and Tchon et al. [17] on remeshing and pseudo-mesh generation using tensor fields and streamlines, respectively.

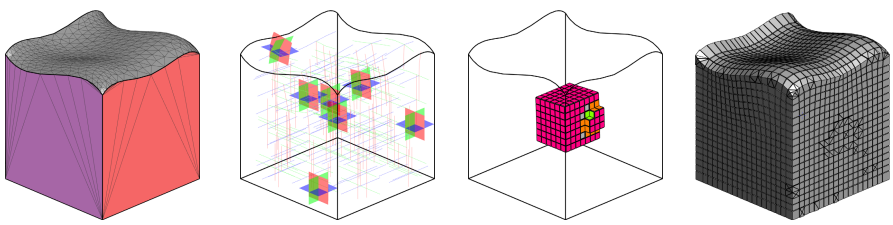
The proposed work is unique in that it assigns the tasks of element shaping and choosing marching directions to a field that is generated prior to meshing. Additionally, insertion operators have been developed that allow systematic insertion of elements in a valid manner, while meshing from the inside-out. Finally, by placing tiled fronts throughout the volume, this method is able to combine with existing methods to produce good quality hex-dominant meshes with augmented hex-to-tet ratios where desired.

3 Technical Approach and Preliminaries

The overall strategy is described in this section, which can be decomposed into three main steps:

1. Metric Field Generation: An iterative technique is used to solve for metric tensors on the nodes of a background mesh. The technique produces a boundary-aligned field, which is then adjusted to uniform anisotropy. With appropriate interpolation, this provides a piecewise-linear field over the volume.
2. Field-Guided Hex Tiling: The element shaping strategy attempts to form unit hexes that are aligned with the metric eigenvector streams and approximately adhere to the lengths encoded in the metric eigenvalues. This is achieved by creating new hex nodes at the intersections of approximately unit length streamsurface triplets. The meshing process begins by inserting a seed hex on the interior of the volume. During tiling, insertion operators determine how new hexes are extended from the current mesh. This continues until the boundaries are reached, at which point new elements are locally snapped to boundary surfaces, feature curves, and feature vertices to enable boundary conformity.
3. Hex-Dominant Mesh Generation: Packing of rectangular solid cells is augmented to incorporate the tiled hex fronts. The hex nodes are packed as fixed cells, and with some specific pre- and post-processing, the process is continued to obtain a hex-dominant mesh.

Figure 1 below depicts some of the main steps of the whole process, in the order described above.



(a) Input geometry (b) Streamlines and streamsurfaces of the generated tensor field (c) Intermediate stage of hex front tiling (d) Final hex-dominant result

Fig. 1. Overview of the whole process

Before elaborating on each of these steps, several techniques and constructs that support field generation and meshing need to be discussed.

3.1 Metric Field Representation, Support, and Operations

The metrics used in this work are stored as symmetric 3×3 matrices with directionality and anisotropy encoded through the eigenvectors and eigenvalues. This is convenient for tasks such as length measurement and interpolation. However, streamline generation requires the eigen-decomposition in order to use the eigenvector fields. Therefore, both representations are employed at different times. The eigen-decomposition is obtained via the Jacobi rotation method [18], and 2 is used to go back to the matrix form.

During meshing it is often necessary to query eigenvector directions, find the best eigenvector corresponding to an input direction, and to relate the eigenvectors of two closely aligned metrics. In this work, eigenvectors are specified by an index ($e = 1, 2, 3$) and sign ($\sigma = \pm 1$) with respect to a given metric. The eigenvector (index-sign pair with respect to a specific metric) that best corresponds to an input direction is taken to be the one that has the largest dot product with it. For example, if the input direction is the normal of a patch of mesh faces, this facilitates finding the best eigenvector to trace streamlines in the normal direction.

When relating two metrics, one of two methods is used. The first consistently orders eigenvectors based on eigenvalue magnitudes, so associativity between the eigenvectors of both metrics is determined by eigenvalue magnitude. This is used to define the three eigenvector fields which are traced during tiling. The second is employed during field generation to snap the eigenvectors of one tensor to the eigenvectors of another. It compares directions as mentioned before to determine the associativity.

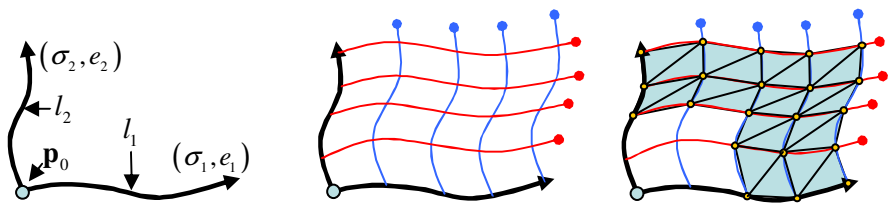
The generated metric fields are piecewise-linear in component space. That is, the six independent components of the tensor field vary linearly over space. As a support, a convex, constrained Delaunay background tet mesh is used. With known surface normals, a flood fill is used to mark interior and exterior tets. Additionally, the metric tensors are stored on the nodes of the background mesh.

Point location in the background mesh is performed by walking [19]. The convexity requirement prevents walks from falsely terminating at non-convex boundaries. It is a suitable choice, considering that most of the field operations sequentially access face-adjacent tets. Another benefit is that the barycentric coordinates can be reused during interpolation. Once the target point is located, point inside/outside can be determined from the status of its enclosing tetrahedron.

Linear interpolation of the enclosing tet's nodal tensors (in component space) yields the local metric components. This is a relatively inexpensive interpolation kernel, but note that it can lead to undesirable properties such as swelling [20] and locally isotropic tensors. The latter case leads to umbilics in the tensor field [21]. The eigenvector directions change very rapidly in the proximity of umbilics, which is a problem when forming elements there. Therefore, tiling avoids such regions.

Streamlines are generated by specifying a starting point, an eigenvector index-sign pair (σ, e), and a desired length under the metric field. Fourth-order, fixed step-size Runge-Kutta [22] is used to generate successive points along a streamline while accumulating the metric-lengths of the new segments [12]. Due to the sign ambiguity of eigenvectors, the direction that most closely matches the established marching direction is chosen; similar considerations are made in [17]. The process is stopped once the measured length meets the desired length, or if the integrator is stalled. Depending on the use, either the entire polyline or just the endpoint is stored.

The next step is to generate streamsurfaces along two eigenvector directions, (σ_1, e_1) and (σ_2, e_2), with nominal lengths l_1 and l_2, respectively (see Figure 2). Primary streamlines are traced according to these initial directions and lengths from the initial point, $\mathbf{p}_0$. The primary streamlines are then resampled to have $n+1$ points including the original first and last points.



(a) Primary streamlines emanating from the initial point

(b) Secondary streamlines emanating from the primary streamlines

(c) Final triangulated quarter-band obtained from streamline net

Fig. 2. Formation of streamsurfaces ($n = 4$)

From each resampled point on the (σ_1, e_1) streamline, a sufficiently long streamline is generated in the local direction corresponding to (σ_2, e_2), and vice-versa. These secondary streamlines should be sufficiently long to account for rapidly converging/diverging eigenvector directions.

Due to the nature of these fields, it is not necessary for the secondary streamlines to intersect. Therefore, an optimization process is used to find the closest points on each pair of streamlines. Let $\mathbf{poly}_1$ and $\mathbf{poly}_2$ be arrays of m_1 and m_2 points, respectively, that contain the points from each streamline. Then the following parameterization provides a piecewise-continuous representation of each $(i = 1, 2)$:

$$\mathbf{p}_i(u) = \mathbf{poly}_i[u_0]\,(1 - u') + \mathbf{poly}_i[u_0 + 1]\,(u')\ , \tag{3}$$

where $u_0 = floor(u)$, $u' = u - u_0$, and $u \in [0, m_i - 1)$. One unit interval in the parameter space spans one line segment. Defining separate parameters

for each curve, s and t, we can formulate an objective function from the squared-distance between the two curves:

$$f = \|\mathbf{p}_1(s) - \mathbf{p}_2(t)\|^2 \ . \tag{4}$$

A steepest descent scheme with adaptive step-sizing is used to find the parameters $t^\star$ and $s^\star$ that minimize this function. The gradient of f can be written as:

$$\nabla f = 2\left([\mathbf{p}_1(s) - \mathbf{p}_2(t)] \cdot \dot{\mathbf{p}}_1(s), [\mathbf{p}_1(s) - \mathbf{p}_2(t)] \cdot \dot{\mathbf{p}}_2(t)\right)^{\mathrm{T}} \ . \tag{5}$$

Central differencing is used to evaluate the tangent vectors of the two curves, $\dot{\mathbf{p}}_1(s)$ and $\dot{\mathbf{p}}_2(t)$. The average of the locations evaluated from the optimal parameters is returned as the closest point.

The net of closest points forms a grid, which is split into triangles to build the streamsurface. In practice, the portion of the net within some length along the primary directions is not generated to lower cost. The streamsurfaces generated in this manner also occupy a single "quadrant," so they are referred to as "quarter-bands."

This concludes discussion of the preliminaries, and in the next sections the methods for field generation and hexahedral tiling will be addressed.

4 Generating Metric Tensor Fields

The key to the proposed method is to separate the tasks of controlling element shape, selecting marching directions, and obtaining boundary alignment from the meshing algorithm. This allows for a simpler meshing algorithm, while incurring more effort during this stage. The following subsections describe a "form-fitting" approach to first generate individual surfaces fields, and then the final boundary-aligned volume field. It is designed to match the intuitive notion of how directionality should vary over a "principally blocky" volume.

4.1 Boundary Field Form-Fitting

In the first step, "scaffold triads" are placed along feature curves and at feature vertices. These triads are aligned with the local features as best as is possible, and specify required directionality but not eigenvector ordering or anisotropy. Many geometric features can be captured this way, however, the orthogonality of the triads limits capture of features such as knife edges or a vertex with more than four incident feature curves.

Then, form-fitting is applied independently to each surface. For a given surface, one of its bounding curve's nodes is selected to begin. This node's metric tensor is initialized with the directions from its scaffold triad, and the following procedure is performed:

```
while not all bounding nodes have been visited do
   Get the metric tensor of the current node, and snap its directions to the closest directions
   on the node's scaffold.
   Mark the node as visited and as a boundary condition (BC) node.
   Solve an approximate surface Laplacian over the current surface, with the current set of
   tensor BCs.
   Move to the next unvisited, adjacent node on the bounding node-loop(s) of the surface.
end while
```

The stencil for the approximate surface Laplacian is:

$$\mathsf{M}^{k+1} = \left(\sum_{j=1}^{N} w_j \right)^{-1} \sum_{j=1}^{N} w_j \mathsf{T}_j \mathsf{M}_j^k \mathsf{T}_j^{\mathrm{T}} , \tag{6}$$

where M^{k+1} is the updated tensor at the current node, w_j is the Floater's mean value coordinate [15] associated with neighbor node $j = 1...N$, T_j is the transformation from the surface normal at neighbor node j to the surface normal at the current node, and M_j^k is the tensor at neighbor node j.

4.2 Interior Form-Fitting

By this point, all surface fields have been established. The fields of adjacent surfaces may not match up in terms of ordered eigenvectors, but they are directionally-compatible due to the shared scaffold triads used while generating them. The tensors on the interior of each surface are snapped to the local surface normal to ensure alignment, and then all surface tensors are converted to scaffold triads. A similar procedure is then employed to march over unvisited boundary nodes and complete the volume field:

```
Pick a starting node on the boundary and generate a complete metric tensor with directions
from its scaffold.
Set this node as visited.
Push node onto boundary-node queue.
while boundary-node queue not empty do
   Set current node to the front node of the queue.
   Pop the front of the queue.
   Snap the current node's metric tensor to its scaffold directions.
   Set the current node as a BC node.
   Solve tensor-component Laplacian over volume with BCs.
   for each node adjacent to the current node do
      if current adjacent node is on the boundary and is unvisited then
         Push onto queue.
         Set visited flag.
      end if
   end for
end while
```

This concludes the description of the form-fitting procedure. After this, all surface tensors are set as boundary conditions to solve for the exterior volume field. This allows streamlines and surfaces to extend slightly past the

boundary as necessary. The nodal tensors of the completed field are modified to represent the desired anisotropy, and the umbilics of the field are extracted using a technique described in [25]. We have modified the second technique presented in that work to extract connected umbilic structures (curves, surfaces, and sub-volumes) as well as discrete points for full- and transverse-isotropy, for the piecewise-linear tensor fields generated here. In the next stage, the tensor field and umbilic information is passed on to the tiling and hex-dominant meshing algorithms.

5 Hex-Dominant Mesh Generation

Individual hex fronts are generated in regions that are free of umbilics and in regions of interest. A hex front is initiated by inserting a seed element on the interior or just within the boundary of the volume. This exposes six hex faces on the "skin," or collection of 2D elements on the boundary of the current hex mesh. The next elements to be inserted will use some of the current skin faces and add new ones, after which the skin is updated. As the skin grows, different combinations of existing skin faces may be used to form new elements. To avoid combinations that might lead to self-intersection or topological invalidity, insertion operators are used to plan and execute insertions that utilize groups of existing skin faces, referred to as insertion face groups (IFG). The IFGs, in turn, are identified by studying their skin nodes. When an operator is used to perform an insertion, depending on the IFG type, quarter-bands are appropriately placed and intersected to obtain the new nodes that define the target element shape. As appropriate, boundary-conformity operations may be invoked to associate proximal mesh entities with boundary features.

Once the fronts have been tiled in ROIs and away from field umbilics, there are two possible strategies for completing the hex-dominant mesh. The first is to perform a Boolean subtraction of the fronts from the volume, generate a hex-dominant mesh in the remaining volume, and then unite the meshes. This is similar to the Hex-Tet algorithm by Meyers et al. [27]. The second is to incorporate the front nodes into the rectangular bubble-packing process, which has been selected for this work. Although it does not guarantee 100% hex recovery, in practice the distribution of front nodes leads to significant recovery. Furthermore, it may allow for better overall element quality and transitions between hex and hex-dominant regions. The potential for non-conformal configurations with hanging-nodes and other complications noted in [24] are also avoided by taking this route.

5.1 Topological Insertion Operators and Face Groups

The rules that determine what types of elements to insert are fundamentally based on the local element-vertex connectivities of the skin vertices and faces. These vertex types can be enumerated by considering a central vertex in a

structured hex mesh. The vertex is initially surrounded by eight elements, and by removing different permutations of elements, a unique set of configurations of skin faces using the vertex can be obtained. Figure 3 lists the types considered in this work. Some configurations have been excluded because they naturally do not arise due to the way elements are inserted; for instance, configurations where two hexes only share one edge or where there is a hexahedral through-hole or void.

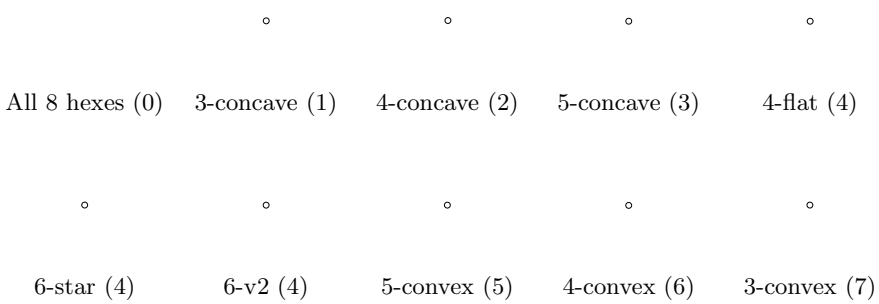


Fig. 3. Skin vertex types

The number of elements removed from the original eight is indicated in parentheses next to each vertex type name. In order, these vertex types will be referred to as: **3c**, **4c**, **5c**, **4f**, **6***, **6v2**, **5v**, **4v**, and **3v** from here on.

We will now present the topological insertion operators which process IFGs. The set of insertion operators and their IFGs is given in the figure below. Each possible insertion (left) is shown with its corresponding IFG (right) in Figure 4. Sets of required vertex types that are used to identify IFGs are enumerated in Table 1.

Table 1. Vertex type requirements for IFGs

IFG type	Vertex type requirements
Side-1	$\text{Type}(\{v_0, v_1, v_2, v_3\}) \in \{\mathbf{4f}, \mathbf{5v}, \mathbf{4c}, \mathbf{3v}\}$
Bracket-2	$\text{Type}(\{v_2, v_5\}) \in \left\{ \begin{matrix} \mathbf{4c}, \mathbf{5c}, \mathbf{6}^\star, \\ \mathbf{6v2}, \mathbf{5v} \end{matrix} \right\}, \text{Type}(\{v_0, v_1, v_3, v_4\}) \in \{\mathbf{4f}, \mathbf{5v}, \mathbf{4v}, \mathbf{3v}\}$
Corner-3	$\text{Type}(v_6) = \mathbf{3c}, \text{Type}(\{v_0, v_2, v_4\}) \in \{\mathbf{4c}, \mathbf{5c}, \mathbf{6}^\star, \mathbf{6v2}, \mathbf{5v}\},$
	$\text{Type}(\{v_1, v_3, v_5\}) \in \{\mathbf{3v}, \mathbf{4f}, \mathbf{4v}, \mathbf{4c}, \mathbf{5v}\}$
Cup-3	$\text{Type}(\{v_1, v_2, v_5, v_6\}) \in \{\mathbf{4c}, \mathbf{5c}, \mathbf{6}^\star, \mathbf{6v2}, \mathbf{5v}\},$
	$\text{Type}(\{v_0, v_3, v_4, v_7\}) \in \{\mathbf{5v}, \mathbf{4v}, \mathbf{3c}\}$
Scoop-4	$\text{Type}(\{v_0, v_1\}) = \mathbf{3c}, \text{Type}(\{v_2, v_3, v_4, v_5\}) \in \{\mathbf{4c}, \mathbf{5v}, \mathbf{5c}, \mathbf{6v2}, \mathbf{6}^\star\},$
	$\text{Type}(\{v_6, v_7\}) \in \{\mathbf{3v}, \mathbf{4f}, \mathbf{4v}, \mathbf{5v}\}$
Bucket-5	$\text{Type}(\{v_0, v_1, v_2, v_3\}) = \mathbf{3c}, \text{Type}(\{v_4, v_5, v_6, v_7\}) \in \{\mathbf{4c}, \mathbf{5c}, \mathbf{6}^\star, \mathbf{6v2}, \mathbf{5v}\}$

*Bracket-2**: There is an ambiguous case when both and are of type *5-convex*. To resolve this, an additional requirement is enforced: the edge between them should be used by three hexes

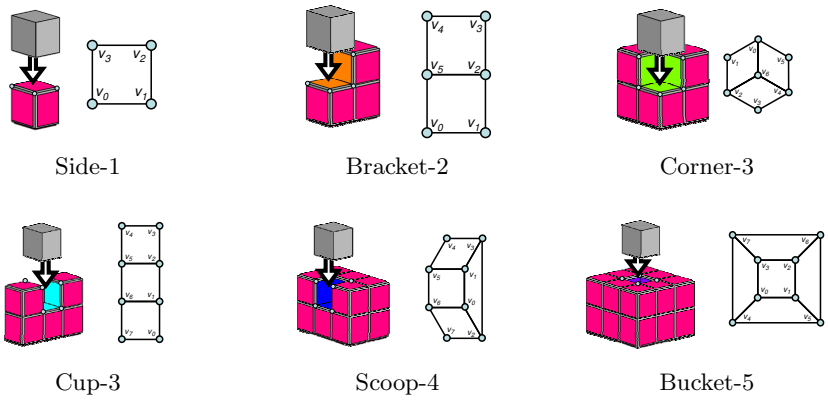


Fig. 4. Topological insertion operators

5.2 Planning and Scheduling Insertions

In this section, we discuss the precedence among the operators at a particular spot and how potential insertions over the entire skin are processed.

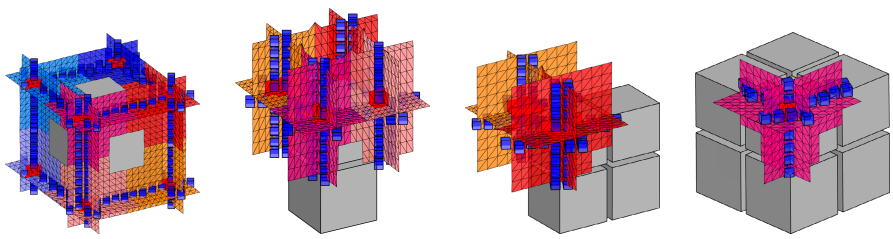
Due to the nature of the vertex type requirements, more complex insertions automatically precede simpler ones over an applicable skin region. This prevents some cases in which the mesh propagates into itself or forms sharp cracks. If vertices are instead classified by geometric criteria such as dihedral angles between the faces containing a vertex, then insertions should be searched for starting from the most complex one (one that is not contained in any other). One possible ordering is: *Bucket-5*, *Scoop-4*, *Cup-3*, *Corner-3*, *Bracket-2*, and finally, *Side-1*. This has been adopted as the current insertion precedence.

In order of precedence, IFGs are placed on unclaimed portions of the skin and also pushed onto an insertion queue. Once again, in order of precedence, the top IFG in the queue is popped and then used to place an element. If this fails, the IFG is maintained but marked as *failed* and pushed back onto the queue. Additionally, if an IFG has vertices that fall outside the volume, it is marked as *do_not_insert*. When an insertion is successfully made from an IFG, the skin and vertex types are locally updated and the IFG is deleted. The affected IFGs (e.g. adjacent ones) are deleted and removed from the queue, then local re-planning is performed. One benefit is that a failed insertion (e.g. due to field behavior) may eventually succeed if neighboring insertions succeed and trigger a reevaluation. Also, this scheme naturally prefers uniform front growth. As long as umbilics and highly distorted field regions are avoided, both features lead to avoidance of self-intersection.

Tiling proceeds until no IFGs remaining in the queue can be processed. The goals of avoiding umbilics, achieving boundary conformity, and staying within a ROI use the *do_not_insert* flag to appropriately terminate tiling in particular directions.

5.3 Element Shaping

Consider a single hexahedron: If its faces are aligned with the metric eigenvectors, as should approximately be the case, then the faces are discrete analogues to similarly placed, continuous streamsurfaces. Mesh edges are then analogous to curves of intersections between any two adjacent streamsurfaces, and mesh nodes are analogous to the intersection of any three adjacent streamsurfaces. This is the guiding principle behind the formation of elements in the proposed method. When an element is to be placed, it may use existing skin faces (IFGs) and add new faces. Streamsurfaces are generated to be parallel to the new faces. The intersections of triplets of these streamsurfaces are used to locate newly introduced nodes needed to form a new hex. Figure 5 illustrates the collections of streamsurfaces used to define the insertion types: *Seed*, *Side-1*, *Bracket-2*, and *Corner-3*. The remaining types shown in Figure 4 only use existing nodes and do not require this procedure.



Seed – Six hex faces are formed using 24 quarter-bands | *Side-1* – Fix hex faces are formed using twelve quarter-bands | *Bracket-2* – Four hex faces are formed using six quarter-bands | *Corner-3* – Three faces are formed using three quarter-bands

Fig. 5. Element shape definition using quarter-bands

To generate a seed element, half unit-length streamlines are integrated along all six eigenvector sign-direction pairs. From the endpoint of each streamline along direction (σ_i, e_i), four quarter-band surfaces are generated from the eigen-directions normal to the local (σ_i, e_i) direction. Triplets of these quarter-bands intersect at a potential node location of the new seed hex. The intersection process is accelerated using a uniform lattice and Möller's presentation of the separating axis theorem [23] to cull the set of triangle-triangle intersection tests that are performed.

For *Side-1*, *Bracket-2*, and *Corner-3* insertions, the eigen-directions for the quarter-bands are determined by looking at the eigen-directions of the metric tensors at the local skin nodes. The eigen-directions which align with skin edges or normals are selected to form the surfaces.

By design, this element formation method primarily aligns newly inserted skin faces with the metric field. Meeting the size requirement is secondary, and thus is only approximately satisfied. Otherwise the mesh would progressively

deviate from local field directions, and local eigen-direction selection could not be used.

5.4 Boundary Conformity

Because the proposed method is designed to produce a boundary-aligned field and hence elements that approach boundary-alignment as they reach the surface, a relatively simple boundary capture method suffices for many models.

When elements are inserted in proximity to the boundary, a series of local and greedy snapping moves are attempted to enable capture of feature vertices, feature curves, and surfaces. For a newly inserted hex, nodes are considered for snapping to candidate feature vertices, then edges to candidate feature curves, and finally, faces to boundary surfaces. Non-dimensional fitness scores have been developed for each class to facilitate prioritizing and qualification of moves. To exclude moves that will result in poor quality elements, the shape metric is evaluated for the hexes incident on a moved node. If the minimum metric is below the threshold, then the move is reverted; 0.3 is used in this work. Additionally, to prevent the mesh from spilling out of the volume, the IFGs containing projected faces are appropriately marked.

5.5 Hex-Dominant Mesh Finalization

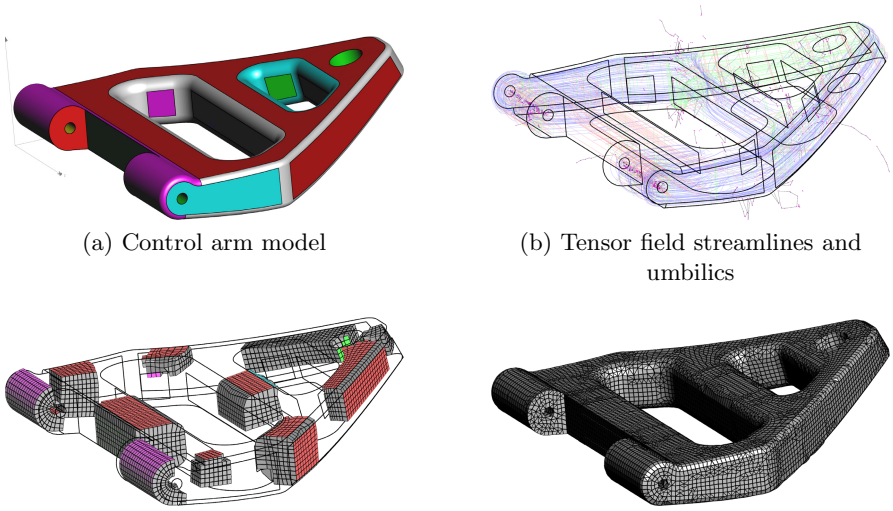
The packing approach to hex-dominant mesh conversion from an initial constrained Delaunay tet mesh is only slightly modified for the purpose of this work. The following overview includes the necessary alterations, in bold, for incorporating the hex fronts into the process:

1. **Hex front nodes are packed as fixed bubbles.**
2. The rest of curves, surfaces, and interior are then packed.
3. **Trimming is performed on regular bubbles that are within one metric unit of the front node bubbles.**
4. **After the surface is remeshed using the surface bubbles, edge swaps are performed to recover hex edges that lie on the boundary.**
5. A constrained Delaunay tet mesh is obtained from the bubble centers.
6. Topological and geometric quality improvement is performed on the tet elements.
7. Tet to hex conversion is attempted and followed by further quality improvement.

6 Results

To demonstrate the efficacy of the proposed method, a large deformation, elasto-plastic analysis of a lower control arm has been performed using

ABAQUS [26]. Figure 6 shows the control arm model, computed volume tensor field along with umbilic structures, hexahedral mesh fronts, and the resulting hex-dominant mesh. The oriented bounding box of the model has dimensions of $16.77 \times 4.61 \times 17.60$ units and pseudo-isotropy is imposed on the field with a nominal mesh size of 0.2 units. The current implementation is not yet fully optimized for performance, but the overall time for field generation, tiling, and generation of the final hex-dominant mesh is under 15 minutes.



(a) Control arm model

(b) Tensor field streamlines and umbilics

(c) Hexahedral fronts

(d) Final hex-dominant mesh

Fig. 6. Control arm results

A total of 8,046 hexes with an average length of 0.20 were generated during tiling. The final hex-dominant mesh consists of 62,300 elements, with an average edge length of 0.23. By quantity, 20,181 (32%) of the elements are hexes and 42,119 (67%) are tets. The hex elements constitute 73.52% of the total volume, whereas only 26.48% is occupied by tets. The minimum and maximum scaled Jacobian metrics of the hexes are 0.4 and 1.0, and the average is 0.93. The radius-ratio metric for the tet elements ranges from 3.0 (ideal) to 697.69, with an average value of 4.12. The distributions of hex and tet element quality according to these metrics are presented in Figure 7.

For the analysis, an elasto-plastic material model based on isotropic hardening was specified with generic parameters for steel: E=210 GPa, ν=0.3, σ_y = 480 MPa, ultimate strength σ_u = 550 MPa, and tangent modulus $E_{p,t}$ = 349.42 MPa. The interior cylindrical surfaces on the left portion of the model were displacement constrained, and lateral and vertical loads of 1,080 kN and 540 kN, respectively, were transmitted to the cylindrical surface

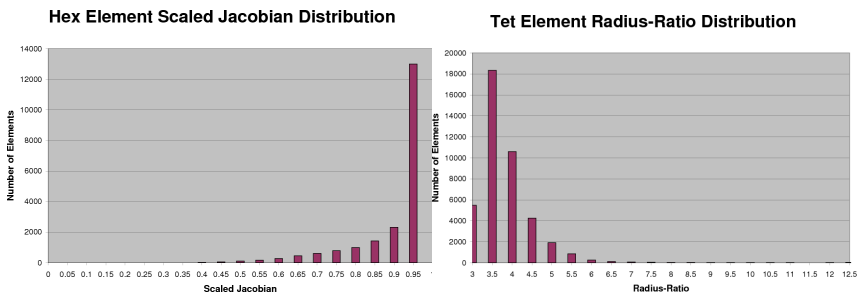
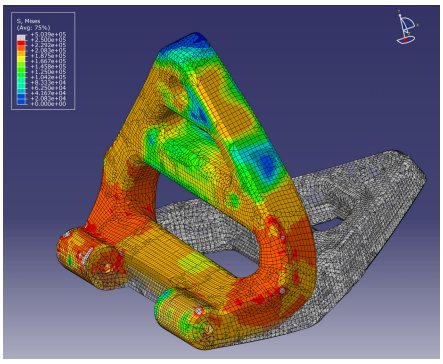
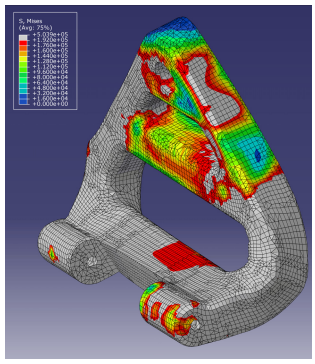


Fig. 7. Hex (Scaled Jacobian) and tet (Radius-Ratio) quality distributions



(a) Von-Mises plot on deformed configuration along with undeformed configuration



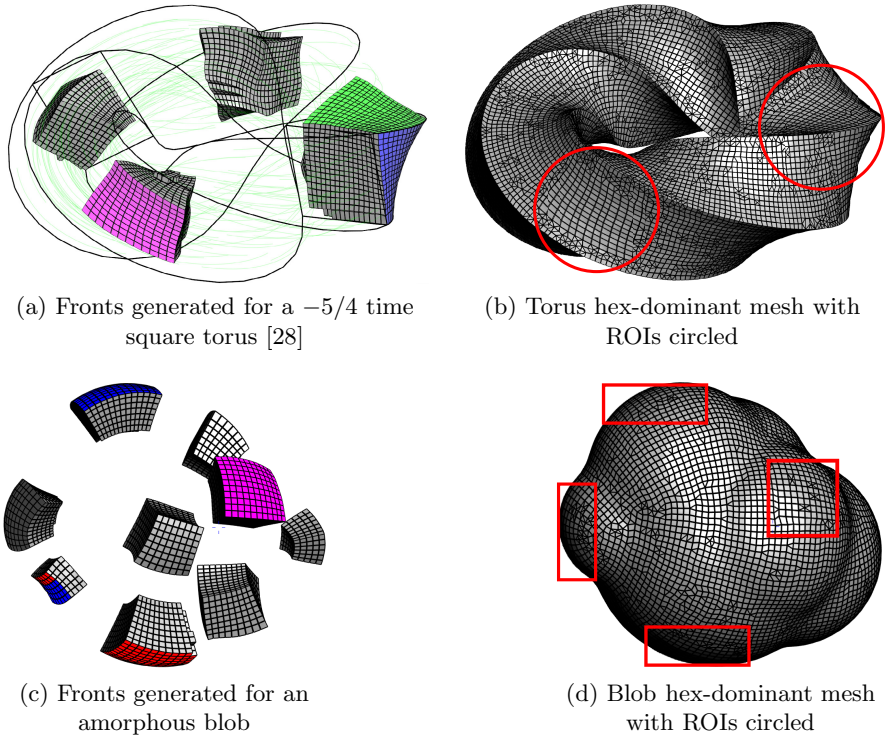
(b) Deformed configuration with plastic zones shown in gray

Fig. 8. Von-Mises stress distribution with fully plastic zones in deformed configuration

at the apex. The deformed configuration is shown in Figure 8, in which the gray regions denote fully plastic zones.

More examples of this method are shown in Figure 9. Each example shows the hexahedral fronts followed by the resulting hex-dominant mesh with the exterior targeted ROIs circled. In all examples, fronts are seeded at interior positions away from umbilics. Colored faces indicate that a similarly colored surface has been captured by the mesh.

The data for these models are summarized in Table 2. Inspection of the resulting hex-dominant meshes reveals that a significant portion of the hex fronts are recovered at the end of meshing process, and with reasonable quality overall. However, the results also warrant further work on improvement of the worst quality elements (primarily tets) and perhaps a more aggressive strategy for hex recovery.



(a) Fronts generated for a $-5/4$ time square torus [28]

(b) Torus hex-dominant mesh with ROIs circled

(c) Fronts generated for an amorphous blob

(d) Blob hex-dominant mesh with ROIs circled

Fig. 9. Additional examples

Table 2. Statistics for additional examples

Model	Square Torus	Blob
Oriented bounding box dims:	$290.11 \times 294.90 \times 100.00$	$25.80 \times 20.51 \times 20.36$
Specified anisotropy:	4.0 pseudo-isotropic	0.5 pseudo-isotropic
# of tiled hexes (avg. edge length):	9,811 (4.11)	5,704 (.46)
# of hex-dom. elements (avg. edge length):	105,052 (4.47)	127,157 (.52)
% hex by volume (by number):	76.90% (36%)	75.32% (33%)
% tet by volume (by number):	23.10% (63%)	24.68% (66%)
(Min / Avg. / Max) Hex Scaled Jacobian:	0.40 / 0.93 / 1.00	0.40 / 0.92 / 1.00
(Min / Avg. / Max) Tet Radius ratio:	3.00 / 3.99 / 87.91	3.00 / 3.82 / 129.26

7 Conclusion

A method for generating quality hex-dominant meshes with targeted all-hex regions has been presented. This is achieved by tiling fronts of hexahedral elements throughout the volume and at ROIs, and then incorporating the front nodes in the rectangular bubble-packing method. To guide element shaping and marching directions during tiling, and bubble directionality and

anisotropy during packing, a metric tensor field is required. To this end, a new approach has been introduced for constructing piecewise-linear metric tensor fields over volumes. This approach produces boundary-aligned fields of uniform anisotropy by incrementally "form-fitting" an initially Cartesian field to the directions specified by boundary curves and surfaces. Provided with the field, the tiling process is able to generate topologically valid hex fronts using insertion operators and boundary conformity operations. The nodes of the fronts are set as fixed bubbles, and through other pre- and post-processing steps, the method is able to generate good quality hex-dominant meshes. Finally, the ability of the method to produce meshes of sufficient quality for difficult, non-linear analyses has been demonstrated.

Plans for future extensions to this work include refining the techniques for field generation and interpolation as well as improving the robustness of the tiling process (including boundary conformity), especially with variations in anisotropy. Furthermore, although this work is motivated by *a priori* anisotropy and directionality requirements, the incorporation of solution-based metrics is currently being investigated.

Acknowledgements

The authors would like to thank Dr. Soji Yamakawa for the use of and help with his geometry, meshing, and graphics class library. The authors would also like to thank Mr. Iacopo Gentilini for his guidance and expertise with ABAQUS.

References

1. Lu, Y., Gadh, R., Tautges, T.J.: Volume Decomposition and Feature Recognition for Hexahedral Mesh Generation. In: Proceedings of the 8th International Meshing Roundtable, pp. 269–280 (1999)
2. Schneiders, R., Schindler, R., Weiler, F.: Octree-based Generation of Hexahedral Element Meshes. In: Proceedings of the 5th International Meshing Roundtable, pp. 205–216 (1996)
3. Zhang, Y., Hughes, T.J.R., Bajaj, C.L.: Automatic 3D Mesh Generation for a Domain With Multiple Materials. In: Proceedings of the 16th International Meshing Roundtable, pp. 367–386 (2007)
4. Blacker, T.D., Meyers, R.J.: Seams and Wedges in Plastering: A 3D Hexahedral Mesh Generation Algorithm. Engineering with Computers 2(9), 83–93 (1993)
5. Staten, M.L., Kerr, R.A., Owen, S.J., Blacker, T.D.: Unconstrained Paving and Plastering: Progress Update. In: Proceedings of the 15th International Meshing Roundtable, pp. 469–496 (2006)
6. Blacker, T.: Meeting the Challenge for Automated Conformal Hexahedral Meshing. In: Proceedings of the 9th International Meshing Roundtable, pp. 11–19 (2000)
7. Yamakawa, S., Shimada, K.: High Quality Anisotropic Tetrahedral Mesh Generation Via Ellipsoidal Bubble Packing. In: Proceedings of the 9th International Meshing Roundtable (2000)

8. Shimada, K., Liao, J.-H., Itoh, T.: Quadrilateral Meshing With Directionality Control through the Packing of Square Cells. In: Proceedings of the 7th International Meshing Roundtable, pp. 61–76 (1998)
9. Viswanath, N., Shimada, K., Itoh, T.: Quadrilateral Meshing With Anisotropy and Directionality Control Via Close Packing Of Rectangular Cells. In: Proceedings of the 9th International Meshing Roundtable, pp. 217–225 (2000)
10. Bossen, F.J., Heckbert, P.S.: A Pliant Method for Anisotropic Mesh Generation. In: Proceedings of the 5th International Meshing Roundtable, pp. 63–76 (1996)
11. Li, X.Y., Teng, S.H., Üngör, A.: Biting Ellipses to Generate Anisotropic Mesh. In: Proceedings of the 8th International Meshing Roundtable, pp. 97–108 (1999)
12. Strang, G.: Linear Algebra and Its Applications, 3rd edn. Brooks-Cole (1988)
13. Sirois, Y., Dompierre, J., Vallet, M.G., Guibault, F.: Mesh Smoothing Based On Riemannian Metric Non-Conformity Minimization. In: Proceedings of the 15th International Meshing Roundtable, pp. 271–288 (2006)
14. Zhang, E., Hays, J., Turk, G.: Interactive Tensor Field Design and Visualization on Surfaces. IEEE Trans. Visual Comput. Graphics 13(1), 94–107 (2007)
15. Palacios, J., Zhang, E.: Rotational Symmetry Field Design on Surfaces. In: Proceedings of ACM SIGGRAPH, p. 55 (2007)
16. Alliez, P., Cohen-Steiner, D., Devillers, O., Levy, B., Desbrun, M.: Anisotropic Polygonal Remeshing. In: Proceedings of ACM SIGGRAPH, pp. 485–493 (2003)
17. Tchon, K.F., Dompierre, J., Vallet, M.G., Camarero, R.: Visualizing Mesh Adaptation Metric Tensors. In: Proceedings of the 13th International Meshing Roundtable, pp. 353–363 (2004)
18. Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. JHU Press, Baltimore (1996)
19. Sundareswara, R., Schrater, P.: Extensible Point Location Algorithm. In: Proceedings of the 2003 International Conference on Geometric Modeling and Graphics, pp. 84–89 (2003)
20. Pennec, X., Fillard, P., Ayache, N.: A Riemannian Framework for Tensor Computing. Int. J. Comput. Vision 66(1), 41–66 (2006)
21. Hesselink, L., Levy, Y., Lavin, Y.: The Topology of Symmetric, Second-Order 3D Tensor Fields. IEEE Trans. Visual Comput. Graphics 3(1) (1997)
22. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C: The Art of Scientific Computing, 2nd edn. Cambridge Univ. Press, UK (1992)
23. Akenine-Möller, T.: Fast 3D triangle-box overlap testing. In: SIGGRAPH 2005: ACM SIGGRAPH 2005 Courses, Article 8 (2005)
24. Yamakawa, S., Shimada, K.: Fully-automated hex-dominant mesh generation with directionality control via packing rectangular solid cells. International Journal for Numerical Methods in Engineering 57(15), 2099–2129 (2003)
25. Zheng, X., Parlett, B.N., Pang, A.: Topological lines in 3D tensor fields and discriminant Hessian factorization. IEEE Trans. Vis. and Comput. Graphics 11(4) (2005)
26. Simulia Corp., ABAQUS Analysis User's Manual, Version 6.8 (2008)
27. Meyers, R.J., Tautges, T.J., Tuchinsky, P.M.: The 'Hex-Tet' Hex-Dominant Meshing Algorithm as Implemented in CUBIT. In: Proceedings of the 7th International Meshing Roundtable, pp. 151–158 (1998)

28. Edelsbrunner, H.: Square Tori: -5/4 time. 180 Wrapped Tubes, `http://www.cs.duke.edu/~edels/Tubes/tori/quad/n5.stl`
29. Yamakawa, S., Shimada, K.: Subdivision Templates for Converting a Non-conformal Hex-Dominant Mesh to a Conformal Hex-Dominant Mesh without Pyramid Elements. In: Proceedings of the 17th International Meshing Roundtable, pp. 497–512 (2008)