Computational Materials Science 197 (2021) 110622

Contents lists available at ScienceDirect

Computational Materials Science

journal homepage: www.elsevier.com/locate/commatsci

# A surface-mesh gradation tool for generating gradated tetrahedral meshes of microstructures with defects

Brian R. Phung [1], Junyan He [1], Ashley D. Spear [*]

*Department of Mechanical Engineering, University of Utah, Salt Lake City, Utah, USA*

ARTICLE INFO

*Keywords:*
Numerical simulation
Polycrystal
Finite element modeling
Short crack
Voids

ABSTRACT

A framework for generating tetrahedral meshes of polycrystalline microstructures with optimal element-size distribution, as determined by the underlying defects (e.g., cracks and voids), is proposed. The framework utilizes a conformal surface mesh of each grain boundary and a point-cloud file to describe, respectively, the polycrystal and its underlying defects. The input meshes are first preprocessed to remove undesired mesh artifacts. To create a suitable element-size distribution informed by the defect structure, elements near the defects are locally refined via edge splitting, while elements far from the defects are coarsened by edge collapsing. The quality of the gradated mesh is improved via edge swapping. The presented framework is robust and flexible, and its ability to selectively refine and coarsen a given mesh is demonstrated by three proof-of-concept models: (1) a polycrystalline open-cell foam with intra-ligament voids, (2) an experimentally measured microstructure with an observed fatigue crack, and (3) an additively manufactured polycrystalline microstructure with an evolving crack. The framework is shown to generate gradated meshes that are converged with respect to both global (e.g., macroscopic stress–strain curve) and local (e.g., crack-front kink angles) metrics, with a significant reduction in both element count and computation time and a minimal reduction in element quality as compared to a uniform mesh. The gradated meshes generated using this framework also provide similar simulation results as produced by their highly refined and uniform mesh counterparts, with the advantage of being more computationally efficient.

## 1. Introduction

Modeling microstructural defects has potential applications in structural prognosis, failure prevention, materials design, and improving the understanding of underlying failure mechanisms. For instance, crack nucleation has been known to be influenced by both the material microstructure [1–4] and geometric defects such as voids [5–8]. Furthermore, crack propagation at the microstructural length scale has been shown to be influenced by the local material microstructure [9–12]. However, the influence of a defect is local, and its presence generally does not influence regions far away from the defect, in accordance with Saint–Venant's principle. In numerical modeling, while the regions near defects require highly refined discretization to resolve steep numerical gradients, the restriction of a uniform mesh leads to excessive (sometimes intractable) computational cost [13–15]. Therefore, a gradated mesh capable of adapting to the defect structure, while conforming to the local microstructure, is desired.

Recent works in the simulation of three-dimensional metallic microstructures from image data have primarily employed a mesh generation method that relies on converting a voxel-based representation into a surface mesh, and converting the surface mesh into a volume mesh (e.g., see Refs. [14–17]). It is noted that a software by Simmetrix Inc. has also been used to generate microstructural meshes [18,19]; however the software is proprietary, so the details of the implementation are unknown. When using the former method, cracks have been inserted into the mesh of the microstructure using a software called Z-Cracks [20,21] or by directly modifying the voxel metadata [15,14]. Particularly, the voxel-based method by Phung and Spear [15] relies on a surface mesh representation for the boundaries of both the microstructure and the defect. This method was shown to be robust for the representation and propagation of a complex crack surface through a three-dimensional microstructure, even when the crack surface grows near or impinges on a grain boundary.

While the previous work by Phung and Spear [15] demonstrated the

* Corresponding author.
*E-mail address:* ashley.spear@utah.edu (A.D. Spear).
[1] Authors contributed equally to this work.

https://doi.org/10.1016/j.commatsci.2021.110622
Received 12 March 2021; Received in revised form 27 May 2021; Accepted 28 May 2021

capability to represent and propagate complex three-dimensional crack surfaces in a realistic microstructure, the framework was limited in the sense that no mesh-gradation support was implemented. Volume-mesh gradation was investigated by utilizing a volume mesher capable of volume-mesh gradation, but it was found that the effectiveness of the gradation was limited by the discretization of the surface mesh. That is, when a desired volume-mesh element size was specified, the volume mesher was required to conform to the existing surface-mesh element size. Therefore, in addition to utilizing a more capable volume mesher, there is a need to directly modify the surface mesh to ease the requirements placed on the volume mesher, thereby enabling substantial mesh coarsening in the far-field while retaining sufficiently refined elements near the microstructural defects.

In general, two types of mesh-gradation methods for a pre-existing mesh are commonly used: one is based on reconstruction and parametrization of the input mesh, while the other is based on direct mesh operations. In the first method, a local or global parametrization is applied to the discrete model. In a parametrization, the discrete geometric facets contained in the input mesh are approximated with a few continuous, analytical surface patches, thereby providing closed-form expressions of model surfaces that can be used to generate an approximated computer-aided design (CAD) model of the domain. A gradated mesh can then be generated from the constructed CAD model. For instance, Bhandari et al. [22] utilized CAD operations to generate meshes of polycrystalline microstructures. By using a parametrization of the mesh, locations of the existing nodes can be modified to improve mesh quality [22–24], or new node locations can be computed to generate a gradated mesh with an optimal element size distribution [25–27]. However, the topology of a realistic microstructure is complex due to the internal grain boundaries and grain-boundary junctions, especially when a complex and three-dimensional defect is involved. Therefore, there is no guarantee that the parametrization algorithm can always approximate the domain with valid analytical surfaces, while retaining the geometric fidelity required to simulate a microstructure and its underlying defects.

In the second approach, mesh gradation is achieved by performing mesh-modification operations directly on the initial mesh. Without the need to generate a parameterization, this method can be more robust and suited for domains with complex topologies and interface structures. Common mesh-modification operations include short-edge contraction and long-edge splitting for coarsening and refinement, respectively. Mesh quality can be subsequently improved by edge-swapping operations. Examples in which mesh-modification operations have been used to generate gradated meshes can be seen in Refs. [28–31]. However, such examples in the literature either do not involve multi-material interfaces, which exist in polycrystals, or do not provide enough control to concentrate the gradation at regions of interest (e.g., near defects). A more involved variant of this approach recomputes nodal locations on material interfaces based on grain volume fraction information, while maximizing the quality of the resulting finite elements [32,33]. Since the nodal locations are recomputed at the interfaces, a locally refined mesh can be generated near interfaces of interest, allowing the user to perform localized mesh refinements. The examples in Owen et al. [32] demonstrate high-quality meshes of complex polycrystals, even with embedded voids, showing significant potential in future applications.

The objective of this work is to propose a robust extension to the previous framework by Phung and Spear [15] that adaptively modifies a mesh of a microstructure containing complex defects, while maintaining a suitable element size distribution that balances computational cost and simulation accuracy. The proposed framework extends the edge-collapsing algorithm to account for the complex multi-material interfaces (e.g., grain boundaries and their junctions) that are present in microstructures and proposes a general mesh-size control function that can account for defects like cracks and voids. Such a framework allows computational effort to be focused on regions on interest, thereby improving computational efficiency and tractability relative to a uniform mesh. Section 2 describes the algorithms underpinning this framework, and Section 3 presents three proof-of-concept (POC) models and the corresponding results to demonstrate the capabilities of this framework. The first POC is of a polycrystalline open-cell foam containing intra-ligament voids. The second POC is of an experimentally characterized aluminum-alloy polycrystal containing a complex three-dimensional crack surface. The third POC is of an evolving three-dimensional crack within a complex microstructure representation of an additively manufactured metal. A discussion of the results and the framework is provided in Section 4.

## 2. Method

The overall objective of the mesh-gradation framework is to generate a conformal mesh for a multi-material domain (e.g., a polycrystal) that contains geometrically explicit defects, with a suitable element-size distribution informed by the underlying defects. The framework uses a set of user-supplied surface-mesh files to define grain boundaries in the polycrystal, as well as a voxel-based representation of the microstructure to identify any voids or secondary-material regions (e.g., precipitates). The user also provides an array of parameters that defines an element-sizing function used to calculate the desired, spatially varying, element size. To ease subsequent mesh operations, the input meshes are preprocessed to remove any undesired artifacts. Then, the framework iterates through all edges in the model. Edges that are deemed too long by the sizing function are refined via edge splitting, and those that are too short are collapsed. During the mesh modification process, surface mesh quality is simultaneously improved via edge swapping. After surface-mesh modifications, a volume mesh is generated based on the element sizes determined by the gradated surface mesh. The following subsections describe each step in detail.

### 2.1. Framework input

The user should first provide the framework with a voxel-based representation of the microstructure and a set of triangular surface-mesh files (e.g., in stereolithography, or STL, file format) describing the boundaries of the grains in the polycrystal. Both file formats are commonly generated when reconstructing microstructures from image-based experimental data. The surface meshes are required to be conformal between adjacent grains, such that the surface mesh of coincident grain-boundary surface pairs are matching. Furthermore, the surface meshes can be smoothed via Laplacian smoothing [34] or any other smoothing operations to minimize unrealistic stair-stepped grain boundaries stemming from a voxel-based representation. This operation is recommended, as the stair-stepped artifacts were shown to induce spurious stress concentration in simulations [33]. For the POC models presented in Section 3, all STL files were generated using DREAM.3D [35] in conjunction with a voxel-based remeshing framework by Phung and Spear [15].

### 2.2. Input-mesh preprocessing

The first step in the mesh-gradation framework is to preprocess the input surface meshes. To begin, meshes of the individual constituent grains are stitched together into a single mesh of the microstructure, on which all the subsequent modifications are performed. While a set of surface meshes can be generated directly from raw-image or voxel-based data, smoothing operations are often employed to mitigate stair-stepped boundaries. A smoothed input surface mesh, such as one generated from DREAM.3D with Laplacian smoothing, can contain mesh artifacts due to the smoothing operation, which can complicate the volume-meshing process but have little effect on the simulated quantities of interest. It is therefore desired to remove such artifacts to simplify subsequent surface-mesh modifications and volume-mesh generation. Three mesh-

preprocessing options are available to handle such undesirable artifacts: unnecessary node removal, sawtooth junction-edge smoothing, and tied-nodes release.

The first available option for mesh preprocessing is *unnecessary node removal*. Each unnecessary node, which is a node with exactly three neighboring nodes and three connected facets, is identified by iterating through all nodes in the mesh. Once an unnecessary node is found, the algorithm removes the unnecessary node along with the three facets to which it is connected. A new triangular facet, which consists of the three neighboring nodes of the removed node, is added to the mesh. See Fig. 1 (a) as an example of this process. This approach is similar to an approach employed by Ito et al. [36] and can help reduce the total number of facets in the mesh as well as improve mesh quality.

The second mesh preprocessing option is *sawtooth junction-edge smoothing*. In the process of applying a smoothing operation using third-party software like DREAM.3D, the trace formed by the intersection of oblique grain boundaries and external surfaces, as well as the trace of intersecting grain boundaries, may be forced to follow angles of 0°, 90°(edges of a voxel), or 45°(face diagonal of a voxel), as shown in Fig. 1(b). This limitation can cause sawtooth-like junctions, making the subsequent junction-edge coarsening operations ineffective due to significant edge-angle mismatch. To create a smooth representation of the grain-boundary intersections, the algorithm iterates through all nodes on grain-boundary intersections and modifies the coordinates of each $i^{th}$ node based on a weighted average of the node's coordinates and those of its immediate neighbors (also on the grain-boundary intersection), calculated as:

$$\mathbf{n}_{\mathrm{i}}^{\mathrm{modified}} = \frac{1}{4}\left(\mathbf{n}_{\mathrm{i-1}}^{\mathrm{original}} + \mathbf{n}_{\mathrm{i+1}}^{\mathrm{original}}\right) + \frac{1}{2}\,\mathbf{n}_{\mathrm{i}}^{\mathrm{original}}, \tag{1}$$

where $\mathbf{n}$ denotes the nodal coordinates. The updated nodal coordinates computed using Eq. (1) are stored in a list, and all relevant nodes are moved in a single operation after all new coordinates are computed. The user also has control over the number of smoothing iterations applied. An example of sawtooth junction-edge smoothing with one iteration of smoothing applied is illustrated in Fig. 1(b). This operation is necessary for the effective coarsening of junction edges, as it minimizes the edge-angle mismatch between adjacent junction edges.

The third mesh preprocessing option is *tied-node release*. In the initial smoothing process (viz., the Laplacian smoothing process in DREAM.3D), if two geometric surfaces are in close proximity and nearly parallel to each other, some nodes on the two surfaces might inadvertently become tied together as shown in Fig. 2(a); the extent to which this happens depends on the parameters used in the smoothing process. The tied nodes create an hourglass-like artifact that deviates from the original geometric surfaces. To detect the hourglass-like artifacts, the algorithm inspects all the nodes that are not on grain-boundary intersections. Nodes that have more than one set of facets connected via a network of shared edges are considered tied nodes. For instance, in Fig. 2(a), the lower set of facets is only connected to the upper set via the highlighted node, which is identified as a tied node. Once identified, the tied node is duplicated, and the two facet sets are reconnected to different copies of the duplicated node. Each of the two coincident nodes is then assigned new coordinates, determined by the centroid of the closed polygon formed by its immediately adjacent nodes. This operation is valuable as it restores geometric information from the input mesh and helps simplify subsequent volume meshing.

### 2.3. Element-sizing function

For maximum computational efficiency, a mesh should be refined near regions of interest with high stress gradients (viz., near defects) and coarsened far from the regions of interest. To achieve this type of element configuration, this work uses a geometry-dependent element sizing function to determine local element sizes. The user controls the element sizing function via a set of user-provided parameters. The function is specially designed for two types of common defects: cracks and voids.

If the defect is a crack, the user provides the crack front geometry by supplying a collection of points $f_j$ that lie on the crack front. For a point $P_i$ in the mesh, the minimal spherical radius $r_i$ is first calculated as:

$$r_i = \min_{\forall j} |P_i, f_j|, \tag{2}$$

where $|\cdot,\cdot|$ denotes the Euclidean distance between two points. Within the framework, this is found via a KD-tree built using the Nanoflann library [37]. The desired edge length $l_i$ at $P_i$ can be related to $r_i$ using:

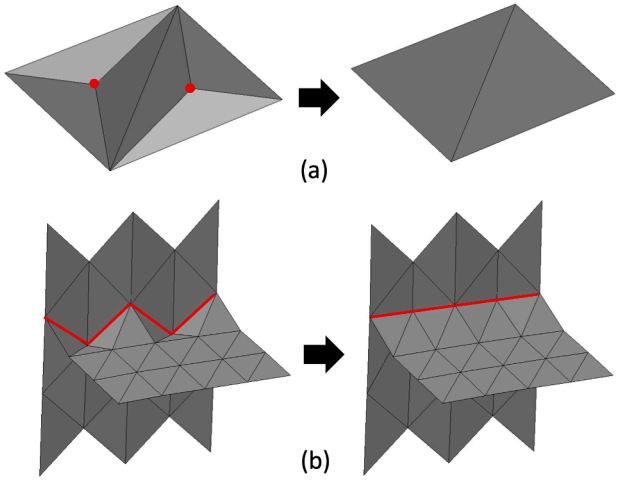


**Fig. 1.** Surface-mesh preprocessing: (a) two unnecessary nodes marked in red, which are removed such that three small facets are replaced by one, (b) a sawtooth trace formed by the edges at a grain-boundary intersection (marked in red). The trace becomes straight after one iteration of smoothing.

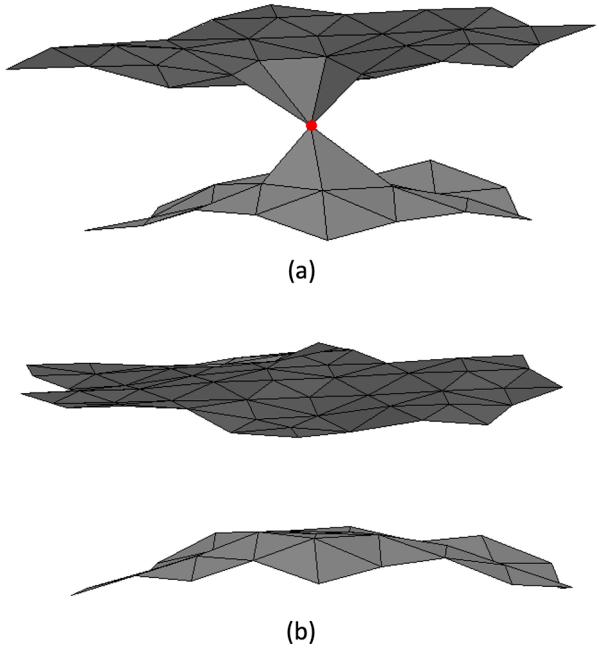


**Fig. 2.** Surface-mesh preprocessing to release tied nodes: (a) input mesh with tied node marked in red, (b) tied node released and its position adjusted. Note the hourglass-like pattern formed near the tied node.

$$l_i = \begin{cases} l_{min} + (l_0 - l_{min})(\frac{r_i}{R_0})^a, & r_i \leqslant R_0 \\ l_0 + (l_{max} - l_0)\frac{r_i - R_0}{R_1 - R_0}, & r_i \in (R_0, R_1] \\ l_{max}, & \text{otherwise} \end{cases} \quad (3)$$

where $l_{min}, l_{max}, l_0$ denote the minimum, maximum, and original edge lengths, respectively, and $R_0, R_1$ and $a$ denote the refinement-zone radius, transition-zone radius and a power-law exponent, respectively. All parameters, except for $l_0$, can be tuned by the user to achieve different element-size distributions. This function allows a smooth and gradual transition from the minimum to maximum edge length. The power-law exponent $a$ is used to control how rapidly the edge lengths gradate within the refinement zone. A two-dimensional example of the sizing function on a model with two penny-shaped cracks is presented in Fig. 3.

If the defect is a void, the user can specify the void geometry via a collection of points $v_j$ on the surface of the void. Points $v_j$ are used in Eq. (2) in place of $f_j$ to calculate $r_i$. Then, Eq. (3) is used as the element sizing function.

### 2.4. Surface-mesh coarsening

The coarsening of the surface mesh is achieved via edge-collapsing operations. An edge-collapsing algorithm, similar to that described in Refs. [36,38–40], was implemented to coarsen the surface mesh. A modified version of the edge-collapsing algorithm was developed to handle the complex connectivity near grain-boundary junctions. As a first step, all edges in the mesh are classified into two categories: regular edges (shared only by two facets) or grain-boundary junction edges (shared by more than two facets). All edges are then sorted in descending order based on edge length. The algorithm then iterates through all the grain-boundary junction edges in the mesh. The coarsening operation is initiated if the length of the current (target) edge is less than the objective edge length determined by the sizing function. Before permanently applying the edge-collapsing operation (described in the subsequent paragraphs), the algorithm temporarily collapses the target edge to its midpoint and checks if this results in a significant change in facet normals of the surrounding facets. The edge is only permanently collapsed if the change in normal angles falls within a user-defined tolerance, thereby minimizing the loss of geometric information. After the junction-edge iteration is completed, the algorithm then iterates through the remaining edges.

If the target edge is a regular edge, a mesh patch is extracted via facets that contain an edge directly connected to the target edge. The vertices that form the boundaries of the mesh patch are also collected. Fig. 4(a) illustrates an example of a mesh patch, whose external edges are highlighted in orange, with its external vertices marked in blue. While the subfigures in Fig. 4 are planar for clear illustration, the algorithm operates in three-dimensional space. The internal angles of the mesh-patch boundary are inspected for any angle greater than 180°; a node whose corresponding internal angle is greater than 180° is marked as a concave corner; two internal angles that are marked as concave are illustrated in blue in Fig. 4(a). Once the mesh patch is extracted and processed, the target edge is collapsed into its midpoint. An initial triangulation is formed by connecting edges between the midpoint and each vertex of the mesh patch; see Fig. 4(b). To maximize the minimum angle in a facet (i.e., to satisfy the constrained Delaunay property), facets in the initial triangulation are grouped into facet pairs. Edge-swapping operations [41] are performed on each facet pair until the minimum angle of each facet is maximized; see Fig. 4(c). To avoid the creation of invalid facets during edge swapping, edges in the initial triangulation directly connected to a concave corner of the polygon (e.g., two white edges in Fig. 4(b)) are not swapped. Lastly, the algorithm checks if adding the new facets results in duplicate, inverted, or otherwise invalid facets – according to user-defined quality limits – before finalizing the modification. The pseudo-code for this process is provided in Fig. 5.

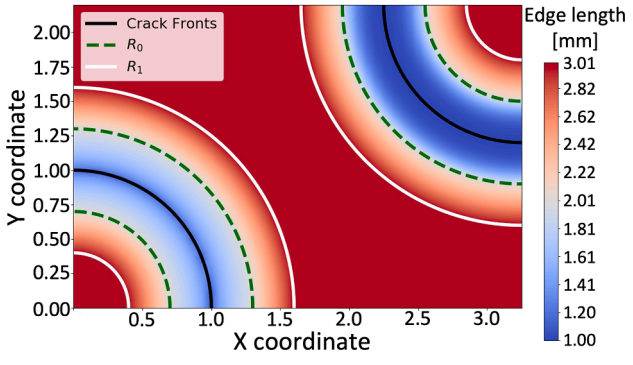


**Fig. 3.** Visualization of element edge length to illustrate (in two dimensions) the element sizing function. Crack fronts highlighted in black. $R_0$ and $R_1$ denote the refinement-zone radius and transition-zone radius (same for both crack fronts). The power-law exponent, $a$, for the left and right crack fronts are 0.5 and 3, respectively. Note the difference in gradation behavior within the refinement zones.

If the target edge is a grain-boundary junction edge, a special step is taken prior to edge collapsing. All facets that share a node with the target edge are collected; see Fig. 6(a). These facets are grouped via connectivity information, based on the grain boundary to which they belong; see Fig. 6(b). The number of groups equals the number of grains sharing the target edge. For each group of facets, all external edges that immediately enclose the target edge are identified. The edge-collapsing algorithm introduced above is then applied to each group of facets; see Fig. 6(c). The pseudo-code for this process is provided in Fig. 7.

### 2.5. Surface-mesh refinement

Element edges whose lengths exceed the specified value given by the sizing function are refined by means of edge splitting. For such an edge, facets that share the edge are collected and stored. A new node is placed at the midpoint of the target edge, with new element edges formed between this new node and the nodes on shared facets (that are not nodes of the target edge). This operation effectively splits each shared facet into two smaller facets. Each pair of split facets are assigned to the same grains as their parent facets. This process is depicted in Fig. 8.

### 2.6. Volume-mesh gradation

Once the stitched surface mesh is modified as described above, volume meshing is performed. The modified and stitched surface mesh of the microstructure is divided into separate STL files for each grain by duplicating nodes at the grain boundaries. The surface mesh for each grain is volume-meshed individually such that the framework can keep track of which elements belong to each grain. This information is used later to generate an element set for each grain in the finite-element input file, which is useful for defining grain-specific material parameters.

An additional enhancement to the current simulation framework is the incorporation of a volume-meshing code that allows gradation control. This is in contrast to previous work by the authors [14,15] that utilized a volume mesher in Abaqus [42], which, when provided a surface mesh as input, does not allow precise control over gradation of the volume mesh. A free and publicly available tetrahedral mesh generator, TetGen (V1.5) [43], is used in the current framework to generate volume meshes. To achieve volume-mesh element sizing control, an initial background volume mesh is first generated for each grain using TetGen. For each node in the initial background volume mesh, the framework assigns a desired volume mesh edge length according to the sizing function given in Eq. 3. The desired edge lengths are written into a .mtr file [44], which can be interpreted by TetGen to specify the element

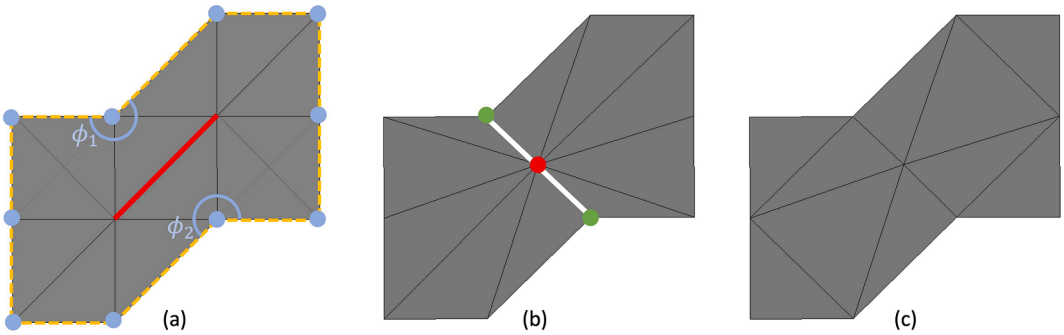


**Fig. 4.** Example edge-collapse procedure for a generic surface mesh patch: (a) Original mesh patch, highlighted in orange, formed by facets directly adjacent to the target edge, highlighted in red. Internal angles measuring greater than 180° (e.g., $\phi_1$ and $\phi_2$) are marked. (b) Target edge is collapsed into its midpoint and the initial triangulation is formed. Two nodes whose corresponding internal angle is greater than 180° are marked in green and their corresponding edges marked in white, which are retained in the final mesh. (c) Final facets after edge swapping.

```
for edge in non-boundary-junction edges:
    checkIfShouldCoarsen();
    checkNormalTolerance();
    getExternalEdges();
    checkConvexity();
    formInitialFacets();
    for facet_pair in initial_facets:
        flipEdge();
    checkIfValid();
    updateMeshInfo();
```

**Fig. 5.** Pseudocode for edge collapsing.

size in the final volume mesh. Mesh-quality control is made available by built-in TetGen command switches such as $-q$ and $-a$ [44].

A consequence of utilizing TetGen as the volume mesher is that special care is taken during volume meshing when a grain contains voids or secondary-material regions (e.g., precipitates) wholly embedded in it. TetGen, by default, fills these voids or embedded regions with volume elements when volume meshing the parent grain. This leads to the erroneous filling of voids or double filling of the embedded regions. Therefore, these voids or embedded regions must be identified first and passed into TetGen to ensure they are properly handled in the volume mesh of the parent grain. To handle this special case, the voxel-based remeshing framework described by Phung and Spear [15] was extended to detect voids and embedded grains. As such, a voxel-based representation of the polycrystal is required as input, as mentioned above, which allows for quick and simple identification of voids or secondary-material regions. If it is detected that a group of voxels sharing an identical feature (grain) ID is wholly surrounded by another group of voxels with a different feature (grain) ID, the former group is identified as a void or secondary-material region. The centroids for these voids or secondary-material regions are stored and passed into TetGen, such that TetGen can correctly generate a valid volume mesh.

## 3. Proof-of-concept models

Three POC polycrystal models are presented to demonstrate the capabilities of the mesh-gradation tool. The effectiveness of mesh gradation, mesh quality after gradation, and the reduction in simulation run time are examined. The details of the POC models are provided in the following subsections. All simulations presented in this section were run in parallel at the University of Utah Center for High Performance Computing, using Intel XeonSP Skylake processors with 32 cores/node and 2.1 GHz clock speeds.

### 3.1. POC1: Intra-ligament voids in an open-cell polycrystalline foam

The goal of POC1 is to demonstrate the framework's capability to selectively refine the mesh near voids, and to coarsen regions that are

```
for edge in boundary-junction edges:
    checkIfShouldCoarsen();
    checkNormalTolerance();
    groupFacets();
    for facet_group in all_groups:
        getExternalEdges();
        checkConvexity();
        formInitialFacets();
        for facet_pair in initial_facets:
            flipEdge();
    checkIfValid();
    updateMeshInfo();
```

**Fig. 7.** Pseudocode for grain-boundary junction edge collapsing.

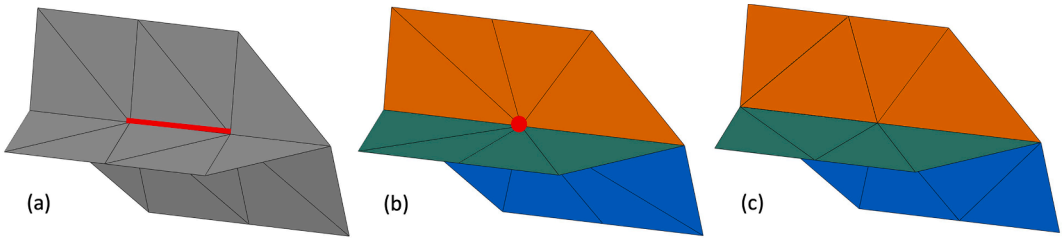


**Fig. 6.** Example edge-collapse procedure for grain-boundary junction edges. (a) Original mesh with target edge highlighted in red. (b) Facets are grouped into three groups, corresponding to the number of intersecting grains. The target edge is collapsed and initial triangulations are formed. (c) Final facets after edge swapping.

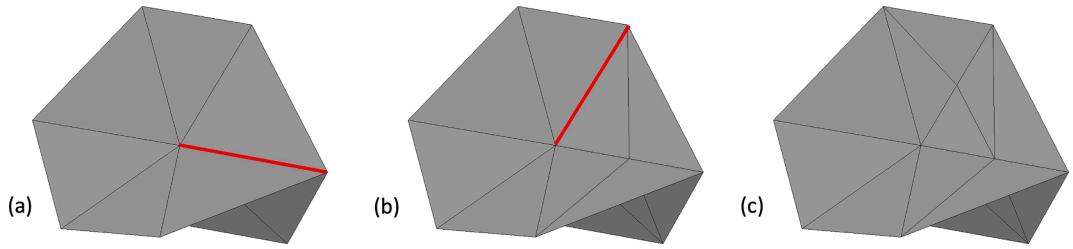


**Fig. 8.** Example edge-splitting procedure: (a) original mesh with first target edge highlighted in red, (b) target edge is split and new edges are added, (c) final facets after two steps of edge splitting.

not of interest, in a relatively complex non-cuboid geometry. In this POC, an open-cell foam with a nominal volume of $10 \times 10 \times 10$ mm$^3$ and 20 grains was synthetically generated in DREAM.3D using a plugin developed by Tucker and Spear [45]. A total of 80 voids with randomly assigned radii ranging from 0.17 to 0.42 mm were inserted at random locations within the foam volume. The geometry of the foam and the inserted voids are depicted in Fig. 9.

Target element sizes for the gradated mesh were determined based on the convergence behavior of the mechanical response with respect to mesh discretization. Specifically, the global (far-field) element size was first determined by examining the load–displacement response of the model. Using a convergence tolerance of 5%, global convergence was achieved at an element size of 0.127 mm. To demonstrate the functionalities of the framework, the target local (viz., near voids) element size was defined to be half the far-field element size (0.064 mm). The input to the mesh-gradation framework was a uniform surface mesh with an element size of 0.084 mm, which was chosen such that the framework would simultaneously coarsen and refine the mesh in specific regions to meet the target element sizes indicated above. Average element edge lengths at each node are calculated and plotted in Fig. 10 to visualize the mesh gradation. To benchmark the size, quality, and computational efficiency of the gradated mesh relative to the previous implementation [15], a uniform volume mesh with an element size of 0.064 mm (equivalent to the local element size in the gradated mesh) was generated using Abaqus 2019 [42], with all other simulation parameters held constant. Basic mesh statistics for both the gradated and uniform mesh models are presented in Table 1. A high-fidelity crystal-plasticity framework, described in work by Zhao et al. [46], was used to simulate the compressive response of the foam model. The calibrated material properties for an aluminum-alloy foam from Zhao et al. [46] were used. The foam model was compressed along the Z-axis to 2% strain via two rigid plates. The simulation of the uniform and gradated meshes took 1793 and 814 CPU hours, respectively.

### 3.2. POC2: Crack in an experimentally characterized aluminum microstructure

The goal of POC2 is to demonstrate the framework's capability to improve upon a previously generated mesh [14] of an experimentally characterized Al-Mg-Si alloy microstructure containing a complex, three-dimensional crack geometry [13]. In particular, the crack geometry imaged at 240,000 load cycles is modeled in this POC. Based on a previous mesh convergence study conducted by Spear [13], element sizes of 12 μm and 6 μm were deemed to be sufficient for global and local convergence, respectively. However, due to the inability to selectively gradate the surface mesh in the previous work, the model described in Spear et al. [14] was restricted to have a nearly uniform element size of 6 μm. Thus, a direct comparison between the uniform mesh and a gradated mesh that transitions from an element size of 6 μm near the crack front to 12 μm in the far-field is provided. First, a uniform, conformal surface mesh was generated for the cracked aluminum microstructure with an element size of 6 μm. From the input STL file, a (nearly) uniform volume mesh was then generated using Abaqus 2019, and a second, gradated mesh was generated using the framework described in Section 2. The gradated mesh was coarsened so that the regions far from the crack have a nominal element size of 12 μm. Basic mesh statistics are presented in Table 2, and the two meshes are depicted in Fig. 11. The average element length is plotted in Fig. 12.

An Abaqus User Element subroutine based on the crystal-plasticity implementation by Matouš and Maniatty [48] was used. The same crystal-plasticity parameters used by Spear et al. [14] were used in these simulations. The two models were loaded uniaxially along the Z-axis to 0.02% strain. The simulation of the uniform and gradated meshes took 594 and 128 CPU hours, respectively.

### 3.3. POC3: Crack propagation in an additively manufactured (AM) microstructure

The goal of POC3 is to demonstrate the use of the surface-mesh

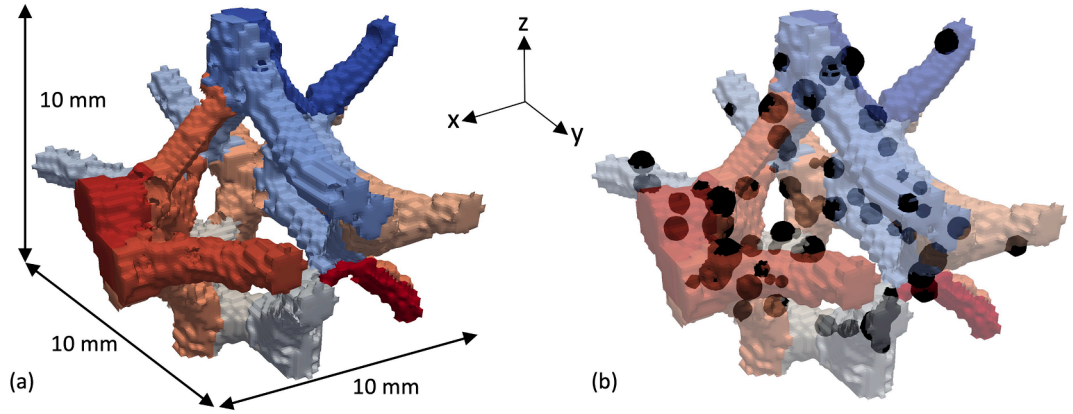


**Fig. 9.** Foam model, colored by grain ID: (a) input surface mesh with uniform element size of 0.084 mm (to aid visualization, element edges are not depicted), (b) voids overlaid on the foam ligaments.

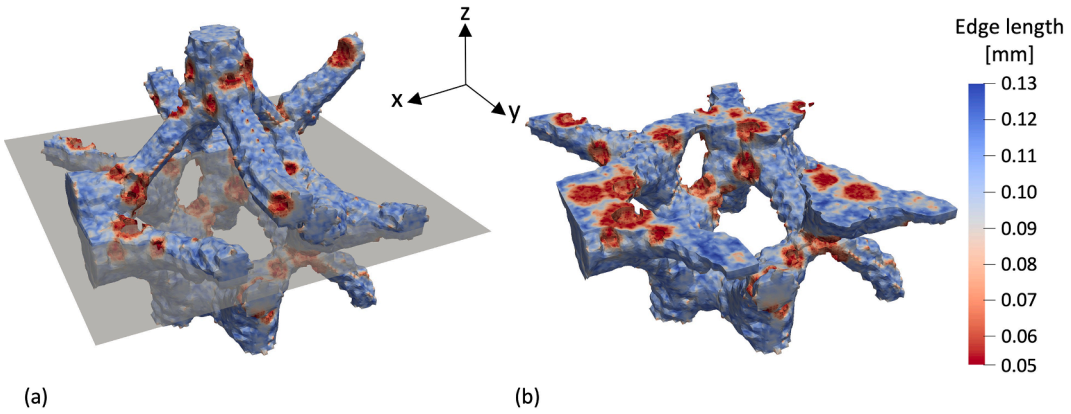


**Fig. 10.** Visualization of mesh gradation applied to an open-cell polycrystalline foam containing intra-ligament voids: (a) distribution of element edge lengths in the entire model, (b) horizontal section view, cut at the plane shown in (a). Note the localized refinement zones surrounding voids.

**Table 1**
Mesh size and quality metrics for the foam model (Proof-of-concept 1).

| | Number of elements $10^6$ | Degrees of freedom $10^6$ | Average aspect ratio[1] (AR) | Elements with AR < 5 [%] | Poor elements[2] [%] |
|---|---|---|---|---|---|
| Uniform | 6.87 | 3.77 | 1.73 | 99.99 | 0.0061 |
| Gradated | 2.36 | 1.24 | 2.02 | 98.89 | 0.28 |

[1] Computed using equations in the TetGen manual [47].
[2] As dictated by analysis check warnings in Abaqus [42].

**Table 2**
Volume mesh statistics for the cracked aluminum microstructure (Proof-of-concept 2).

| | Number of elements $10^6$ | Degrees of freedom $10^7$ | Average aspect ratio[1] (AR) | Elements with AR < 5 [%] | Poor elements[2] [%] |
|---|---|---|---|---|---|
| Uniform | 8.40 | 3.39 | 1.76 | 99.93 | 0.045 |
| Gradated | 3.35 | 1.35 | 1.99 | 98.85 | 0.36 |

[1] Computed using equations in the TetGen manual [47].
[2] As dictated by analysis check warnings in Abaqus [42].

gradation framework, in concert with adaptive remeshing, to model an evolving microstructurally small crack. In this POC, a synthetically generated AM microstructure of stainless steel SS316L from work by Herriott et al. [49] was modeled. The microstructure contains a total of 26 grains. The crystal-plasticity model used in POC2 was used in this study. The microstructure was loaded in uniaxial tension along the Z-axis to 0.2% strain. Two simulations were performed at each crack-growth increment: one in which the surface-mesh was uniform and one in which the surface-mesh was gradated according to the current crack geometry.

A convergence study was conducted to determine the target element sizes for the gradated mesh. A global (far-field) element size of 3.6 μm was found to provide a sufficiently converged (within 2% tolerance) engineering stress–strain response of the uncracked microstructure. To determine target element size based on local convergence, a semi-circular initial crack with a radius of 18 μm was inserted into the model, and the predicted kink angles were calculated point-wise along the crack front according to the maximum tangential stress (MTS) criterion, which asserts that crack propagation will occur in the direction that maximizes tangential stress (i.e., $\sigma_{\theta\theta,max}$) [50]. An element size of 1.5 μm was found to achieve convergence of the calculated kink angles (within 5% tolerance). Thus, the target global (far-field) and local (near-crack) element sizes specified in the gradated mesh were 3.6 μm and 1.5 μm, respectively.

Starting from the initial crack, a total of five crack-growth increments was simulated. For each crack increment, crack-growth criteria were evaluated point-wise along the crack front, and a new mesh was generated to conform to the predicted crack surface. For each crack-front node, the MTS criterion was used to predict the kink angle for crack extension, and a uniform crack extension of 25% of the current

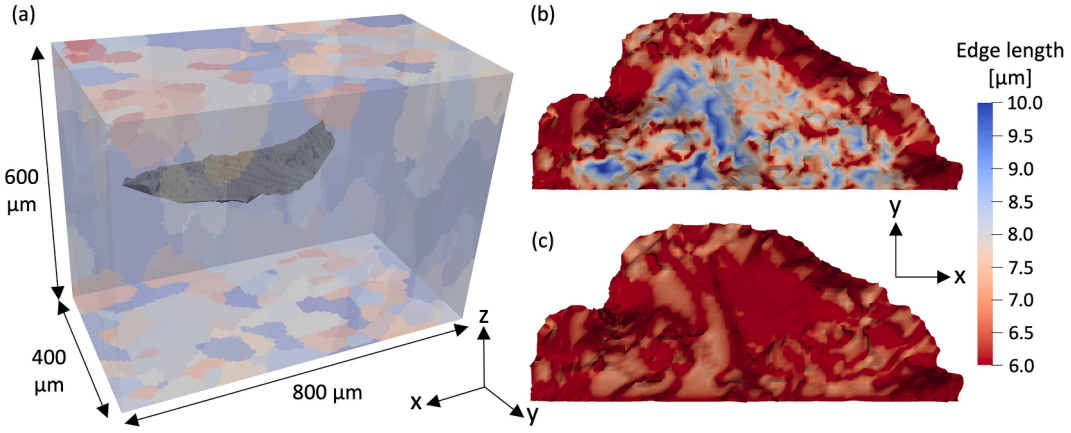


**Fig. 11.** Impact of surface-mesh gradation applied to an experimentally characterized aluminum microstructure [13] containing an observed fatigue crack: (a) model of crack surface embedded within the microstructural volume, (b) top-down view of the crack-face mesh generated using the current framework, (c) uniform crack-face mesh generated without surface-mesh gradation [14].

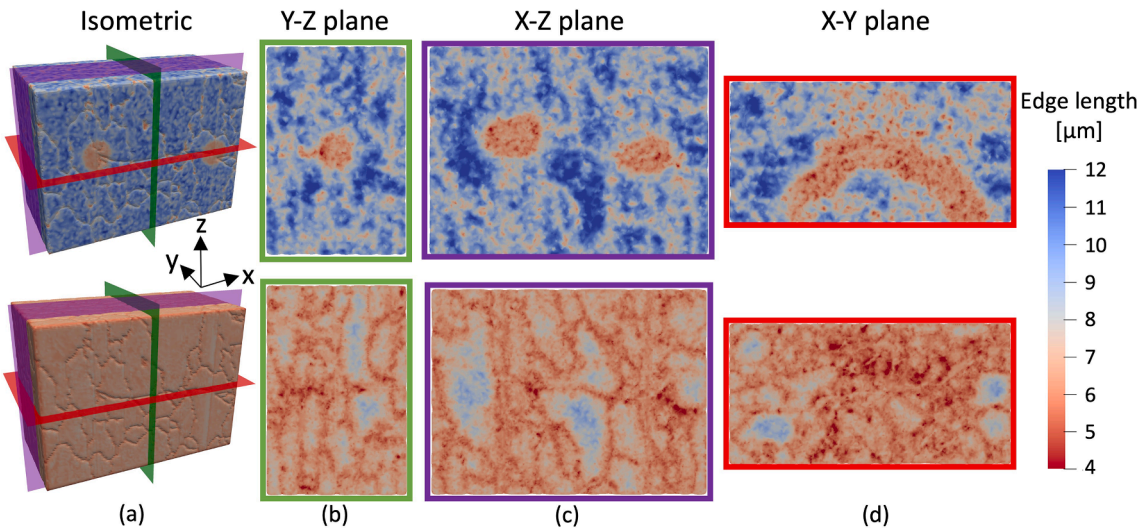


**Fig. 12.** Distribution of element-edge length for the cracked aluminum microstructure models, with (top row) and without (bottom row) surface-mesh gradation: (a) isometric view, (b) Y-Z plane view, (c) X-Z plane view and (d) X-Y plane view. Note the refinement zone surrounding the crack front in the gradated mesh.

crack radius was applied. To ensure that the uniform and gradated meshes correspond to the same crack geometry, the kink angles predicted from the uniform mesh were used to update the crack geometry as the starting configuration for the next increment. To account for noise in the predicted kink angles, a Savitzky–Golay filter [51] was used to generate a smoothed version of the updated crack front. After the crack geometry was updated, a uniform surface mesh (STL file) containing the explicitly resolved crack surface was generated using the voxel-based remeshing framework by Phung and Spear [15]. Prior to volume meshing, the surface mesh was gradated based on the target element sizes described above. For comparison, a uniform volume mesh (i.e., no surface-mesh gradation applied) with element size of 1.5 μm was also generated.

The basic mesh statistics for all meshes are presented in Table 3. The crack surfaces at selected increments, and the element sizes at the crack surfaces, are depicted in Fig. 13. The stress distributions visualized within the Y-Z plane through the center of the volume are shown in Fig. 14. The raw (unsmoothed) kink-angle predictions for the uniform and gradated meshes for the fifth (last) crack-growth increment are shown in Fig. 15(a). Fig. 15(b) provides a map of the absolute difference in maximum principal stress plotted on a plane containing both the crack-origin point and an arbitrary point along the crack front (which is circled in Fig. 15(a)). In Fig. 15(b), a magnified view of the region near the crack-front point is shown, along with the probe path (A-B) used to identify the direction of $\sigma_{\theta\theta,max}$. The tangential stress ($\sigma_{\theta\theta}$) along this probe path is shown in Fig. 15(c), with the maximum value indicated. Finally, the percent reduction in element count relative to the uniform meshes and a comparison of total simulation CPU time are presented in Fig. 16.

**Table 3**
Volume-mesh statistics for five crack-growth increments in an additively manufactured microstructure (Proof-of-concept 3)

| Crack-growth increment | Number of elements $10^5$ | Degrees of freedom $10^6$ | Average aspect ratio[1] (AR) | Elements with AR < 5 [%] | Poor elements[2] [%] |
|---|---|---|---|---|---|
| Uniform, 0 | 22.5 | 9.15 | 1.73 | 99.99 | 0.0028 |
| Gradated, 0 | 7.37 | 2.98 | 2.01 | 98.83 | 0.28 |
| Uniform, 1 | 21.6 | 8.79 | 1.73 | 99.99 | 0.0030 |
| Gradated, 1 | 7.56 | 3.05 | 2.02 | 98.82 | 0.29 |
| Uniform, 2 | 21.4 | 8.69 | 1.73 | 99.99 | 0.0034 |
| Gradated, 2 | 7.94 | 3.21 | 2.03 | 98.81 | 0.29 |
| Uniform, 3 | 21.3 | 8.68 | 1.73 | 99.98 | 0.0044 |
| Gradated, 3 | 8.42 | 3.40 | 2.02 | 98.82 | 0.29 |
| Uniform, 4 | 21.2 | 8.62 | 1.73 | 99.98 | 0.0052 |
| Gradated, 4 | 8.87 | 3.58 | 2.04 | 98.76 | 0.30 |
| Uniform, 5 | 21.0 | 8.57 | 1.73 | 99.99 | 0.0058 |
| Gradated, 5 | 8.73 | 3.53 | 2.04 | 98.77 | 0.30 |

[1] Computed using equations in the TetGen manual [47].
[2] As dictated by analysis check warnings in Abaqus [42].

## 4. Discussion

### 4.1. Framework robustness and computation time

The robustness and capabilities of the mesh-gradation framework have been demonstrated for the relatively complex configurations of microstructures and defects presented above. In the first POC, the robustness of the framework is demonstrated by the simultaneous coarsening and refinement of a complex foam geometry with multiple ligaments, grains, and intra-ligament voids. From Fig. 9(b), it can be seen that the void configuration is complex, featuring internal voids of varying sizes and voids that intersect each other. Despite such complexities, the final gradated mesh provides a faithful representation of the foam geometry, while offering improved simulation efficiency relative to a uniform mesh via selective mesh refinement surrounding the voids – which, by extension, efficiently captures the high stress gradients.

The framework robustness is further demonstrated in POC2, in which a large microstructure (more than 640,000 triangular facets in the input surface mesh and about 430 grains) was coarsened multiple times while maintaining a refinement zone near the highly complex, non-planar crack surface. This POC highlights the framework's capability of capturing arbitrarily shaped geometrical discontinuities (i.e., traction-free cracks) with simple user inputs (crack-front node coordinates), while maintaining a high-fidelity representation of the underlying microstructure.

Finally, POC3 demonstrated the application of the surface-mesh gradation framework to a problem involving adaptive remeshing to

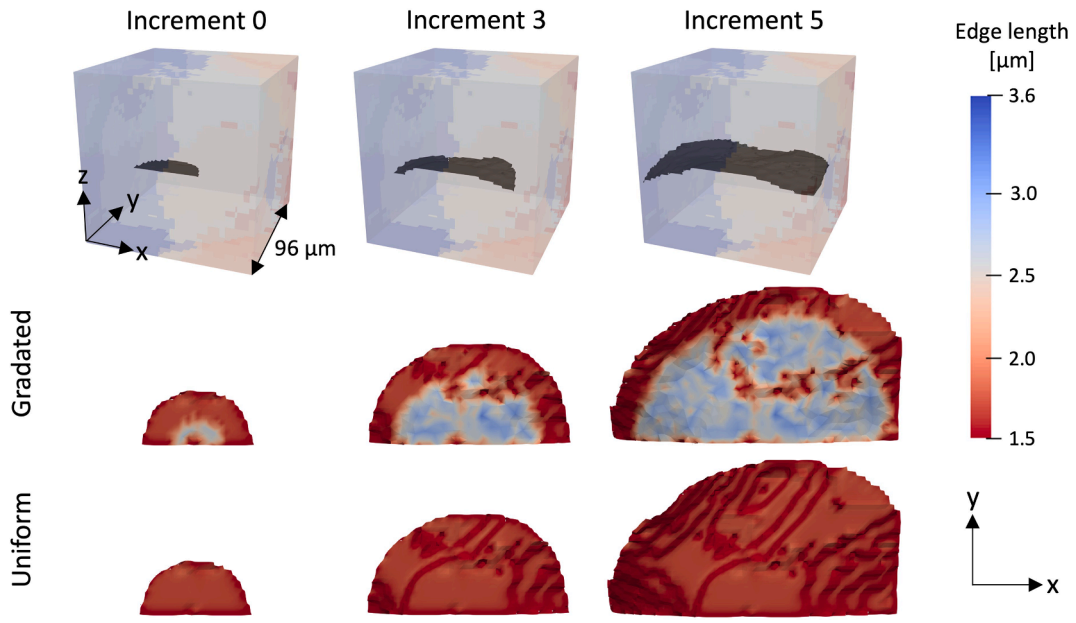


**Fig. 13.** Impact of surface-mesh gradation applied to a crack-growth simulation. Trimetric views of the crack surfaces embedded within the microstructural volume are shown in the first row. Distributions of element-edge lengths are depicted in top-down views of the evolving crack surface, with (middle row) and without (bottom row) surface-mesh gradation.

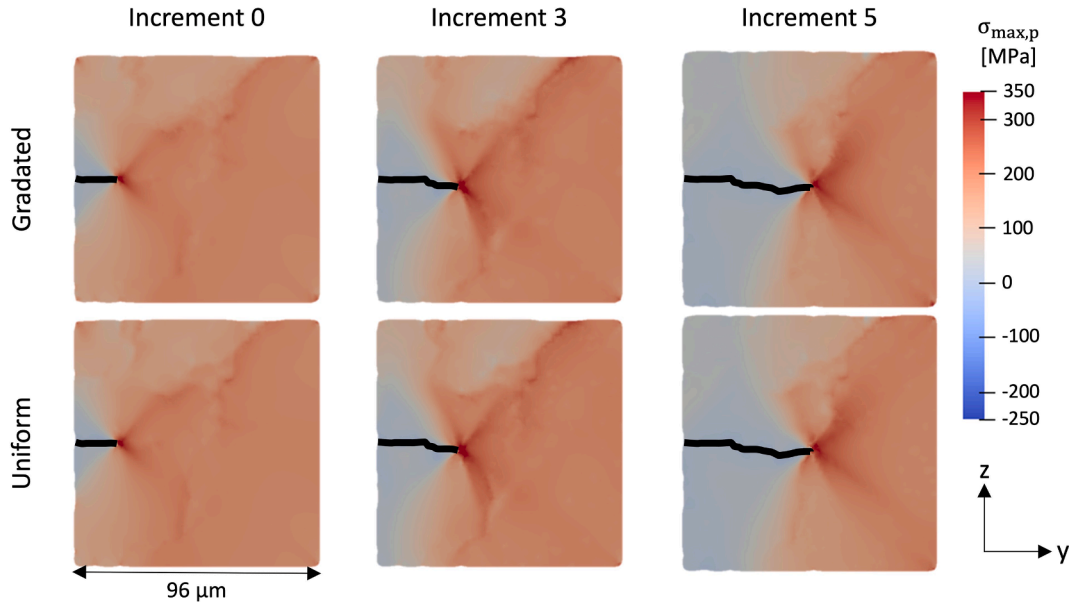


**Fig. 14.** Stress distribution in a Y-Z plane view for crack-growth simulations (POC3). The top and bottom rows depict the maximum principal stress for the gradated and uniform surface meshes, respectively. The black line depicts the trace of the three-dimensional crack surface. Note the similar stress distributions in both sets of meshes. For visual clarity, the mesh edges are not shown.

simulate an evolving crack surface in a three-dimensional, heterogeneous microstructure. This POC showed that the current framework can reliably produce meshes in multiple crack-growth increments. It also demonstrated that the refinement zone can track the movement of the evolving crack front (see Fig. 13), rather than maintaining a refined mesh throughout the entire simulation domain, which dramatically improves computational efficiency compared to previous meshing approaches. Fig. 15(b) shows that very slight differences exist in the computed stress fields between the uniform and gradated meshes. However, the differences are small enough that the fracture parameter of interest (viz., raw kink angle based on the MTS criterion) is not significantly impacted by the mesh gradation, as shown in Fig. 15(a). In fact, the distribution of $\sigma_{\theta\theta}$ along the probe path A-B is nearly identical between the two meshes, as depicted in Fig. 15(c).

It should be noted that the mesh-gradation parameters for the POCs are based on a few simplifying assumptions for the purpose of demonstrating the capabilities of the framework. Specifically, the locations of refinement are assumed to correspond to the locations of defects within each model (i.e., near the crack front or at the surfaces of internal voids). Furthermore, for POC3, it is assumed that the level of refinement determined for the initial crack geometry is sufficient for subsequent crack-growth increments. In practice, this might not be sufficient, as the initial convergence study might not capture every interaction between the defect(s), grain orientations, and grain boundaries. For applications

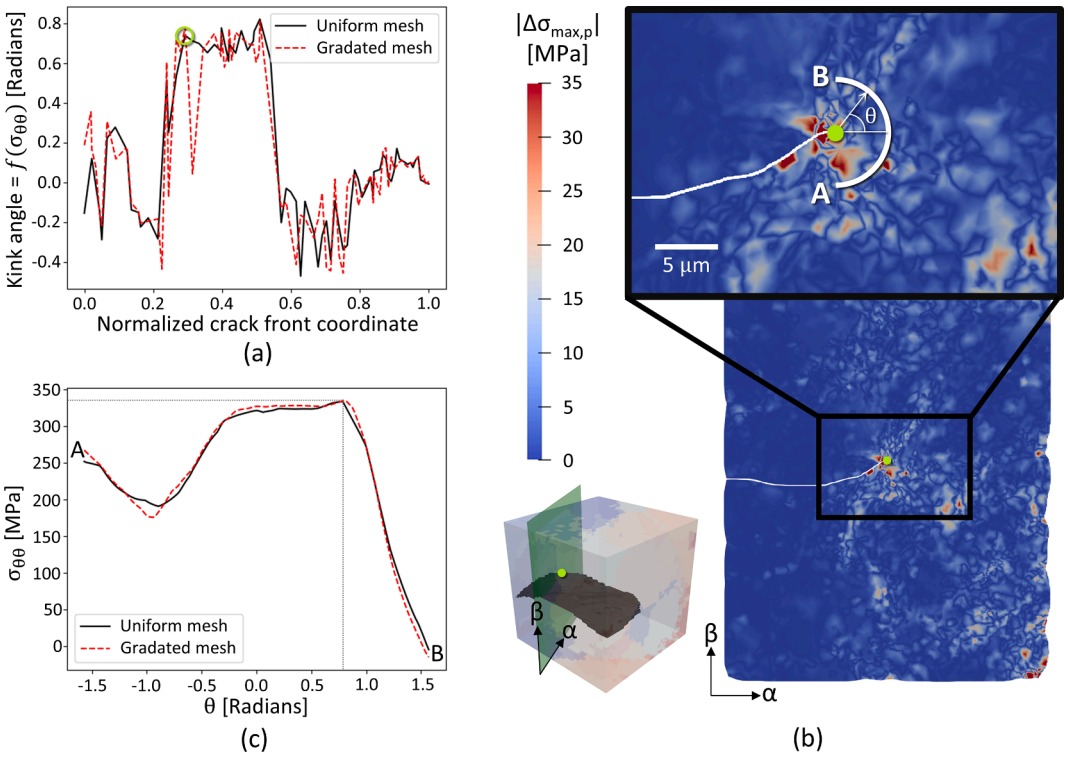


**Fig. 15.** Difference between uniform and gradated meshes in terms of relevant solution parameters for POC3. (a) Raw (unsmoothed) kink-angle predictions along the crack front for the fifth (last) increment of crack growth based on the Maximum Tangential Stress criterion, with an arbitrary point along the crack front circled in green. (b) Absolute difference in maximum principal stress in plane $\alpha - \beta$ that contains both the crack-front point circled in (a) and the crack origin point. Tangential stress ($\sigma_{\theta\theta}$) is probed along the path A-B. (c) $\sigma_{\theta\theta}$ plotted along the probe path A-B, with the angle corresponding to $\sigma_{\theta\theta,max}$ indicated.

requiring more nuanced mesh convergence, a comprehensive convergence study can be performed, or a generous factor of safety can be applied. Ultimately, the criteria for mesh gradation and mesh convergence are problem-specific, and the user is responsible for determining appropriate mesh parameters.

Significant reductions in simulation times relative to uniform meshes were observed in all three POC models due to the reduced element count in the gradated meshes. On average, element count was reduced by more than 60% for all POCs (see Tables 1–3) in the gradated meshes compared to uniform meshes at the locally convergent element size, which resulted in an average of 68.5% reduction in computation time. For simulations involving high-fidelity, geometrically explicit representations of microstructures containing voids or cracks, the large reduction in element count and simulation time provided by surface-mesh gradation offers a significant advantage over meshing strategies that are limited to uniform surface meshes (e.g., see Refs. [14,15]). The ability to control the gradation of the surface mesh also allows users to include higher mesh density at preferred regions of interest or to simulate a larger volume of microstructure that would otherwise be intractable with a uniform mesh requirement.

### 4.2. Framework limitations and future work

Despite the considerable reduction in computational time and retained computational accuracy, the framework still has its limitations when handling grain-boundary junction edges. In some cases, collapsing grain-boundary junction edges may not be possible due to the complexity of some of these junctions. This complexity could cause the algorithm to be unsuccessful in yielding a valid collapsed grain-boundary junction, as described in Section 2.4. While input mesh preprocessing such as sawtooth junction-edge smoothing, as described in Section 2.2, reduces the frequency of uncollapsible grain-boundary junctions, the junction-edge smoothing algorithm in this work is unable to completely remove all sawtooth junction edges in the models. As a result, element sizes near some grain-boundary junctions are smaller, as fewer edges are coarsened, and have lower quality overall. An example of this can be seen in Fig. 12, where the location of small-element sizes outside the region-of-interest are correlated with the location of grain-boundary junctions. Future work will focus on implementing more robust and effective smoothing operations to mitigate the effect of uncollapsible grain-boundary junctions.

Furthermore, in the current implementation of the edge-collapse and edge-splitting algorithm, the target edge is collapsed or split into its midpoint as a node, which is a reasonable first approximation. However, this does not always yield optimal surface-mesh element quality. The location of this node can be adjusted to maximize element quality while preserving local geometric information. Such mesh-quality improvement algorithms are described in Refs. [23,52]. The inclusion of these algorithms into the framework can further increase the quality of the gradated surface mesh, leading to better overall element quality.

Ultimately, the reduction in simulation time comes with a slight trade-off in the overall volume-mesh quality. As seen in Tables 1–3, the gradated surface meshes lead to slightly larger average aspect ratios in the corresponding volume meshes as compared to the volume meshes generated from the uniform surface meshes. The gradated meshes also have a higher percentage of poor elements (albeit, less than 0.4% in all cases). The reduction in element quality is primarily attributed to two factors. First, the quality of the volume mesh is tied to the quality of the gradated surface mesh. Simply collapsing edges into their midpoints or adding additional nodes on edge midpoints during coarsening and refinement are likely not sufficient to maximize element quality, when compared to other methods that recompute nodal coordinates to maximize element quality [32]. Second, the current framework uses TetGen (V1.5) [43] to generate a conformal, boundary-retained volume mesh, which offers mesh quality control through controlling the face angles and dihedral angles [43]. Controlling these mesh quality measures,

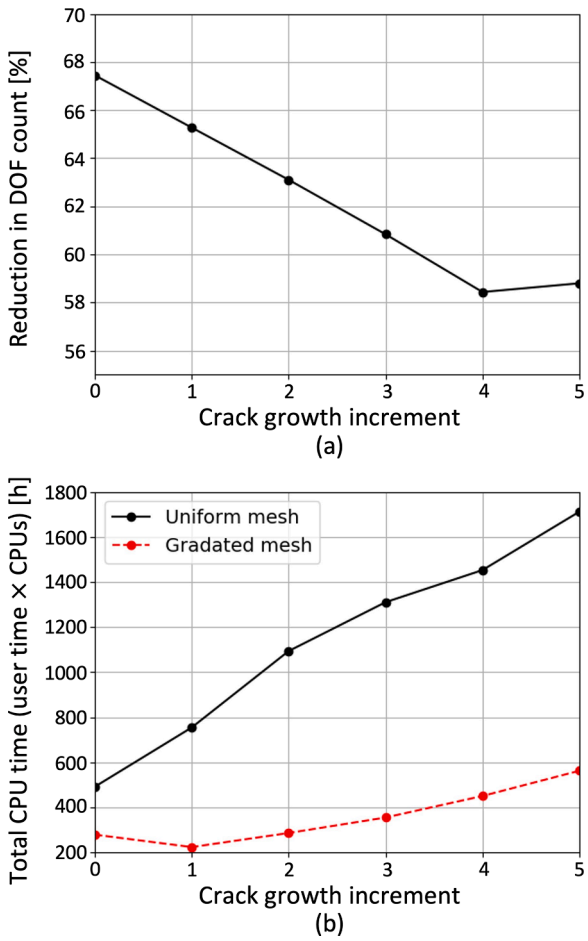


**Fig. 16.** Comparison between uniform and gradated meshes for crack-growth simulations (POC3): (a) percent reduction in degree-of-freedom counts of the gradated meshes as compared to the uniform meshes, (b) CPU time for each simulation.

however, does not effectively prevent the generation of sliver elements. In fact, most of the poor elements in the gradated meshes are sliver elements. The lack of user options to identify sliver elements, is another contributing factor to the lower (albeit, not substantial) overall quality of the gradated meshes. However, as evidenced by the comparisons in Fig. 14, the slightly degraded element quality does not have a significant effect on the overall stress distributions, nor does it have a significant effect on the crack-growth metrics studied in this work, as seen in Fig. 15 (c). This comparison provides confidence in the simulation results, even though the mesh quality has slightly deteriorated after mesh modification. The framework's capability to considerably reduce simulation time without significantly impacting simulation results justifies its use in the analysis of microstructurally small cracks, including multi-step crack propagation in complex microstructures.

## 5. Conclusions

A framework for surface-mesh gradation of three-dimensional microstructures with defects (e.g., voids or cracks) is presented. The required inputs are a microstructure representation defined via a set of user-supplied STL files and the defect(s) representation defined via a set of points on the defect surfaces. Element-size distribution can be defined by supplying an array of parameters (e.g., minimum/maximum edge lengths, refinement-zone radius, etc.) to the global sizing function. Selective mesh coarsening and refinement are performed to the input mesh via direct mesh operations: edge splitting when the edge is too long and edge collapsing when the edge is too short. Mesh quality is simultaneously improved via the edge-swapping algorithm. A volume mesh is generated from the gradated surface meshes using TetGen, with element sizes determined again by the sizing function. The flexibility and capabilities of the framework are demonstrated through three proof-of-concept models:

1. An open-cell, polycrystalline foam model with 80 voids inserted randomly throughout the ligaments. This demonstrates the ability to refine selectively the mesh near complex voids in a complex geometry.
2. An experimentally characterized aluminum microstructure with a complex crack geometry. This demonstrates the ability to handle a complex crack embedded in a large, experimentally derived microstructure.
3. Five crack propagation increments in an additively manufactured microstructure. This demonstrates the ability of the crack-front refinement zone to track the evolving three-dimensional crack front throughout a polycrystal.

The three proof-of-concept models show that the gradated mesh significantly outperforms the uniform mesh in terms of computation time while maintaining a similar level of solution accuracy. These observations render the current framework useful in studies involving defects, such as high-fidelity crack growth simulations, in complex microstructures.

## Data availability

The source code used to generate the models presented in this work can be found at: https://github.com/craftmesher/craftmesher.

## CRediT authorship contribution statement

**Brian R. Phung:** Methodology, Software, Data curation, Writing - original draft. **Junyan He:** Methodology, Software, Investigation, Writing - original draft. **Ashley D. Spear:** Supervision, Resources, Writing - review & editing, Funding acquisition.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant Nos. CMMI-1629660 and CMMI-1752400. All simulations presented in this work were conducted using the computational resources provided by the University of Utah Center for High-Performance Computing. The authors would like to acknowledge Carter Cocke, Vignesh Babu Rao, Lisa Wiesent, and Dongfang Zhao for their valuable support and fruitful discussions. The authors would also like to thank Dr. Wenda Tan and Carl Herriott for allowing the use of the microstructure presented in POC3.

## References

[1] J.D. Hochhalter, D.J. Littlewood, R.J. Christ, M.G. Veilleux, J.E. Bozek, A. R. Ingraffea, A.M. Maniatty, A geometric approach to modeling microstructurally small fatigue crack formation: II. physically based modeling of microstructure-dependent slip localization and actuation of the crack nucleation mechanism in AA 7075–T651, Model. Simul. Mater. Sci. Eng. 18 (2010) 045004, https://doi.org/10.1088/0965-0393/18/4/045004.
[2] D. McDowell, F. Dunne, Microstructure-sensitive computational modeling of fatigue crack formation, Int. J. Fatigue 32 (2010) 1521–1542.
[3] T.T. Nguyen, J. Yvonnet, Q.Z. Zhu, M. Bornert, C. Chateau, A phase field method to simulate crack nucleation and propagation in strongly heterogeneous materials

from direct imaging of their microstructure, Eng. Fracture Mech. 139 (2015) 18–39.
[4] V. Wan, J. Jiang, D. MacLachlan, F. Dunne, Microstructure-sensitive fatigue crack nucleation in a polycrystalline Ni superalloy, Int. J. Fatigue 90 (2016) 181–190.
[5] N. Shahidzadeh-Bonn, P. Vié, X. Chateau, J.N. Roux, D. Bonn, Delayed fracture in porous media, Phys. Rev. Lett. 95 (2005), 175501.
[6] E. Ferrie, J.Y. Buffiere, W. Ludwig, 3D characterisation of the nucleation of a short fatigue crack at a pore in a cast al alloy using high resolution synchrotron microtomography, Int. J. Fatigue 27 (2005) 1215–1220.
[7] D.S. Watring, K.C. Carter, D. Crouse, B. Raeymaekers, A.D. Spear, Mechanisms driving high-cycle fatigue life of as-built inconel 718 processed by laser powder bed fusion, Mater. Sci. Eng.: A 761 (2019), 137993, https://doi.org/10.1016/j.msea.2019.06.003.
[8] J.M. Erickson, A. Rahman, A.D. Spear, A void descriptor function to uniquely characterize pore networks and predict ductile-metal failure properties, Int. J. Fracture 225 (2020) 47–67, https://doi.org/10.1007/s10704-020-00463-1.
[9] D. Taylor, J. Knott, Fatigue crack propagation behaviour of short cracks; the effect of microstructure, Fatigue Fract. Eng. Mater. Struct. 4 (1981) 147–155.
[10] S. Kumar, W.A. Curtin, Crack interaction with microstructure, Mater. Today 10 (2007) 34–44.
[11] A.D. Spear, S.F. Li, J.F. Lind, R.M. Suter, A.R. Ingraffea, Three-dimensional characterization of microstructurally small fatigue-crack evolution using quantitative fractography combined with post-mortem X-ray tomography and high-energy x-ray diffraction microscopy, Acta Mater. 76 (2014) 413–424.
[12] M. Herbig, A. King, P. Reischig, H. Proudhon, E.M. Lauridsen, J. Marrow, J.-Y. Buffière, W. Ludwig, 3-D growth of a short fatigue crack within a polycrystalline microstructure studied using combined diffraction and phase-contrast X-ray tomography, Acta Mater. 59 (2011) 590–601.
[13] A.D. Spear, Numerical and experimental studies of three-dimensional crack evolution in aluminum alloys: Macroscale to microscale, Cornell University, 2014.
[14] A.D. Spear, J.D. Hochhalter, A.R. Cerrone, S.F. Li, J.F. Lind, R.M. Suter, A. R. Ingraffea, A method to generate conformal finite-element meshes from 3D measurements of microstructurally small fatigue-crack propagation, Fatigue Fract. Eng. Mater. Struct. 39 (2016) 737–751.
[15] B.R. Phung, A.D. Spear, A voxel-based remeshing framework for the simulation of arbitrary three-dimensional crack growth in heterogeneous materials, Eng. Fracture Mech. 209 (2019) 404–422.
[16] J.C. Tucker, A.R. Cerrone III, A.R. Ingraffea, A.D. Rollett, Crystal plasticity finite element analysis for René88DT statistical volume element generation, Model. Simul. Mater. Sci. Eng. 23 (2015), 035003.
[17] H. Proudhon, J. Li, P. Reischig, N. Guéninchault, S. Forest, W. Ludwig, Coupling diffraction contrast tomography with the finite element method, Adv. Eng. Mater. 18 (2016) 903–912, https://doi.org/10.1002/adem.201500414.
[18] O. Klaas, M.W. Beall, M.S. Shephard, Construction of models and meshes of heterogeneous material microstructures from image datas, in: Image-Based Geometric Modeling and Mesh Generation, Springer, 2013, pp. 171–193.
[19] M.P. Garcia, C. Luo, A. Noshadravan, A. Keck, R. Teale, A. Chattopadhyay, P. Peralta, Microstructure representation and material characterization for multiscale finite element simulations of local mechanical behavior in damaged metallic structures, in: D.K. Lindner (Ed.), Modeling, Signal Processing, and Control for Smart Structures 2008, vol. 6926, International Society for Optics and Photonics, SPIE, 2008, pp. 131–138. doi:10.1117/12.776580.
[20] V. Chiaruttini, V. Riolo, F. Feyel, Advanced remeshing techniques for complex 3D crack propagation, vol. 1, 2013.
[21] H. Proudhon, J. Li, W. Ludwig, A. Roos, S. Forest, Simulation of short fatigue crack propagation in a 3D experimental microstructure, Adv. Eng. Mater. 19 (2017) 1600721, https://doi.org/10.1002/adem.201600721.
[22] Y. Bhandari, S. Sarkar, M. Groeber, M. Uchic, D. Dimiduk, S. Ghosh, 3D polycrystalline microstructure reconstruction from FIB generated serial sections for fe analysis, Comput. Mater. Sci. 41 (2007) 222–235.
[23] R.V. Garimella, M.J. Shashkov, P.M. Knupp, Optimization of surface mesh quality using local parametrization, IMR (2002) 41–52.
[24] R.V. Garimella, M.J. Shashkov, P.M. Knupp, Triangular and quadrilateral surface mesh quality optimization using local parametrization, Comput. Methods Appl. Mech. Eng. 193 (2004) 913–928.
[25] J.-F. Remacle, C. Geuzaine, G. Compere, E. Marchandise, High-quality surface remeshing using harmonic maps, Int. J. Numer. Methods Eng. 83 (2010) 403–425.
[26] E. Marchandise, C.C. de Wiart, W. Vos, C. Geuzaine, J.F. Remacle, High-quality surface remeshing using harmonic maps-part ii: Surfaces with high genus and of large aspect ratio, Int. J. Numer. Methods Eng. 86 (2011) 1303–1321.
[27] E. Marchandise, J.F. Remacle, C. Geuzaine, Optimal parametrizations for surface remeshing, Eng. Comput. 30 (2014) 383–402.
[28] G.L. Miller, D. Talmor, S.H. Teng, Optimal coarsening of unstructured meshes, J. Algorithms 31 (1999) 29–65.
[29] X. Li, M.S. Shephard, M.W. Beall, 3D anisotropic mesh adaptation by mesh modification, Comput. Methods Appl. Mech. Eng. 194 (2005) 4915–4950.
[30] C. Ollivier-Gooch, Coarsening unstructured meshes by edge contraction, Int. J. Numer. Methods Eng. 57 (2003) 391–414.
[31] Y. Lu, E.J. Garboczi, Bridging the gap between random microstructure and 3D meshing, J. Comput. Civ. Eng. 28 (2014) 04014007.
[32] S.J. Owen, J.A. Brown, C.D. Ernst, H. Lim, K.N. Long, Hexahedral mesh generation for computational materials modeling, Procedia Eng. 203 (2017) 167–179.
[33] H. Lim, F. Abdeljawad, S.J. Owen, B.W. Hanks, J.W. Foulk, C.C. Battaile, Incorporating physically-based microstructures in materials modeling: Bridging phase field and crystal plasticity frameworks, Model. Simul. Mater. Sci. Eng. 24 (2016), 045016.
[34] D. Field, Laplacian smoothing and delaunay triangulations, Commun. Appl. Numer. Methods 4 (1988) 709–712, https://doi.org/10.1002/cnm.1630040603.
[35] M.A. Groeber, M.A. Jackson, Dream. 3D: a digital representation environment for the analysis of microstructure in 3D, Integrating materials and manufacturing innovation 3 (2014) 5.
[36] Y. Ito, P. Corey Shum, A.M. Shih, B.K. Soni, K. Nakahashi, Robust generation of high-quality unstructured meshes on realistic biomedical geometry, Int. J. Numer. Methods Eng. 65 (2006) 943–973.
[37] J.L. Blanco, P.K. Rai, nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with kd-trees, https://github.com/jlblancoc/nanoflann, 2014.
[38] X. Shi, P. Koehl, Adaptive skin meshes coarsening for biomolecular simulation, Comput. Aid. Geometr. Des. 28 (2011) 307–320.
[39] H.H. Chen, X.N. Luo, R.T. Ling. Surface simplification using multi-edge mesh collapse, IEEE, 2007, pp. 954–959.
[40] P. Cignoni, D. Costanza, C. Montani, C. Rocchini, R. Scopigno, Simplification of tetrahedral meshes with accurate error evaluation, in: Proceedings Visualization 2000. VIS 2000 (Cat. No. 00CH37145), IEEE, 2000, pp. 85–92.
[41] H.L. de Cougny, Refinement and coarsening of surface meshes, Eng. Comput. 14 (1998) 214–222.
[42] SIMULIA, Abaqus, 2019.
[43] H. Si, Tetgen, a delaunay-based quality tetrahedral mesh generator, ACM Trans. Math. Software (TOMS) 41 (2015) 1–36.
[44] H. Si, A. TetGen, A quality tetrahedral mesh generator and a 3D delaunay triangulator, Cited on (2009) 61.
[45] J.C. Tucker, A.D. Spear, A tool to generate grain-resolved open-cell metal foam models, Integr. Mater. Manuf. Innovation 8 (2019) 247–256.
[46] D. Zhao, K.E. Matheson, B.R. Phung, S. Petruzza, M.W. Czabaj, A.D. Spear, Investigating the effect of grain structure on compressive response of open-cell metal foam using high-fidelity crystal-plasticity modeling, Mater. Sci. Eng.: A 140847 (2021).
[47] H. Si, Tetgen: A quality tetrahedral mesh generator and a 3d delaunay triangulator (version 1.5-user's manual) (2013).
[48] K. Matouš, A.M. Maniatty, Finite element formulation for modelling large deformations in elasto-viscoplastic polycrystals, Int. J. Numer. Methods Eng. 60 (2004) 2313–2333.
[49] C. Herriott, X. Li, N. Kouraytem, V. Tari, W. Tan, B. Anglin, A.D. Rollett, A. D. Spear, A multi-scale, multi-physics modeling framework to predict spatial variation of properties in additive-manufactured metals, MSMSE 27 (2019), 025009.
[50] F. Erdogan, G. Sih, On the crack extension in plates under plane loading and transverse shear, ASME J. Basic Eng. 85 (1963) 519–525.
[51] A. Savitzky, M.J.E. Golay, Smoothing and differentiation of data by simplified least squares procedures, Anal. Chem. 36 (1964) 1627–1639, https://doi.org/10.1021/ac60214a047. DOI: 10.1021/ac60214a047.
[52] B. Clark, N. Ray, X. Jiao, Surface mesh optimization, adaption, and untangling with high-order accuracy, in: Proceedings of the 21st international meshing roundtable, Springer, 2013, pp. 385–402.