ELSEVIER

Computational Materials Science 28 (2003) 199–208

COMPUTATIONAL MATERIALS SCIENCE

www.elsevier.com/locate/commatsci

Modeling microstructure evolution in three dimensions with Grain3D and LaGriT

Andrew Kuprat *, Denise George, Galen Straub, Melik C. Demirel [1]

Theoretical Division, Los Alamos National Laboratory, P.O. Box 1663, Los Alamos, NM 87545, USA

Abstract

This paper will describe modeling microstructure evolution using a combination of our gradient-weighted moving finite elements code, Grain3D and our 3-D unstructured grid generation and optimization code, LaGriT. Grain boundaries evolve by mean curvature motion, and Grain3D allows for the incorporation of grain boundary orientation dependence modeled as anisotropic mobility and energy. We also describe the process of generating an initial computational grid from images obtained from electron backscatter diffraction. We present the grid optimization operations developed to respond to changes in the physical topology such as the collapse of grains and to maintain uniform computational grid quality. For 3-D columnar microstructures, validation of the method is demonstrated by comparison with experiments. For large systems of fully 3-D microstructures, simulations compare favorably to the parabolic law of normal grain growth.

PACS: 07.05.Tp; 68.35.−p; 68.37.Hk; 68.55.Jk; 81.40.−z
Keywords: Microstructure evolution; Motion by mean curvature; Gradient-weighted moving finite elements; Unstructured tetrahedral meshes; Deforming grids; Changing grid topology

1. Introduction

Metals important to semiconductor manufacture such as aluminum and copper possess a microstructure consisting of polycrystals. The interfaces between the crystals have excess free energy σ per unit area that represents the work that must be done to create unit area of interface. (σ may also arise from material lying in or close to the interface.) As the metal is heated, the grain boundaries move to minimize surface energy in such a way that the normal velocity of a point on the grain boundary is proportional to the mean curvature at that point [15,17].

In this paper in Section 2 we describe a moving finite element code, Grain3D, that has been developed to model grain boundary evolution. In Section 3 we describe the set of grid maintenance operations implemented in LaGriT (Los Alamos Grid Toolbox) [8]. This set keeps the computational mesh from becoming tangled and assures the quality measure of the elements of the mesh does not degrade. In Section 4 the authors provide an experimental validation of the modeling technique

* Corresponding author. Tel.: +1-505-665-5746; fax: +1-505-665-4055.
E-mail address: kuprat@lanl.gov (A. Kuprat).
[1] Present address: School of Engineering, Pennsylvania State University, University Park, PA 16802, USA.

0927-0256/$ - see front matter
doi:10.1016/S0927-0256(03)00107-1

in which we compare at the level of individual grains the experimental and computational evolution. In Section 5 we present the statistical results of simulations on large systems of 5000 grains. Sections 6 gives implementation and availability information for Grain3D and LaGriT.

2. Summary of method

In our simulation of grain growth as implemented in the code Grain3D, we discretize individual grains by tetrahedra and we discretize the interfaces between the grains by the linear triangles that are the facets shared by adjacent tetrahedra of differing grains. Typically, there are on the order of a hundred tetrahedra used to discretize single grains and on the order of tens of triangles discretizing an interface between two adjacent grains.

We use gradient-weighted moving finite elements (GWMFE) [2,3,14] to move the network of triangles that corresponds to the grain interfaces. In our mean curvature model of grain growth, the interface surface's motion is described by

$$v_{\mathrm{n}} = \mu\sigma K, \tag{1}$$

where v_{n} is the normal velocity of the interface, K is the curvature, μ is the mobility, and σ is the free interfacial energy per unit area.

As in [11], in the GWMFE method, interfaces are represented as parametrized piecewise linear surfaces:

$$\mathbf{x}(s_1, s_2) = \sum_{\text{nodes } j} \alpha_j(s_1, s_2)\mathbf{x}_j.$$

Here (s_1, s_2) is the surface parametrization, the sum is over the N interface nodes, $\alpha_j(s_1, s_2)$ is the piecewise linear basis function (hat function) which is unity at node j and zero at neighboring nodes, and $\mathbf{x}_j = (x_j^1, x_j^2, x_j^3) \in \mathbb{R}^3$ is the vector position of node j.

We have that

$$\dot{\mathbf{x}}(s_1, s_2) = \sum_j \alpha_j(s_1, s_2)\dot{\mathbf{x}}_j$$

is the velocity of the surface at the point $\mathbf{x}(s_1, s_2)$ (based upon linear interpolation of node velocities) and

$$v_{\mathrm{n}} = \dot{\mathbf{x}}(s_1, s_2) \cdot \hat{\mathbf{n}} \quad (\hat{\mathbf{n}} \text{ is local surface normal}).$$

So

$$v_{\mathrm{n}} = \sum_j (\hat{\mathbf{n}}\alpha_j) \cdot \dot{\mathbf{x}}_j. \tag{2}$$

In effect, we have that the $3N$ basis functions for v_{n} are $n_k\alpha_j$, where $\hat{\mathbf{n}} = (n_1, n_2, n_3)$. These basis functions are discontinuous piecewise linear, since the n_k are piecewise constant.

The GWMFE method is to minimize

$$\int (v_{\mathrm{n}} - \mu\sigma K)^2 \,\mathrm{d}S \tag{3}$$

over all possible values for the velocities $\dot{\mathbf{x}}_i$. (The integral is over the surface area of the interfaces.) We thus obtain

$$\begin{aligned} 0 &= \frac{1}{2}\frac{\partial}{\partial \dot{x}_i^k}\int (v_{\mathrm{n}} - \mu\sigma K)^2 \,\mathrm{d}S, \\ &= \int (v_{\mathrm{n}} - \mu\sigma K) n_k \alpha_i \,\mathrm{d}S \quad 1 \leqslant k \leqslant 3, \quad 1 \leqslant i \leqslant N. \end{aligned}$$

Using (2), we obtain a system of $3N$ ordinary differential equations (ODEs):

$$\left[\int \hat{\mathbf{n}}\hat{\mathbf{n}}^{\mathrm{T}}\alpha_i\alpha_j \,\mathrm{d}S\right]\dot{\mathbf{x}}_j = \int \mu\sigma K\hat{\mathbf{n}}\alpha_i \,\mathrm{d}S$$

or

$$\mathbf{C}(\mathbf{x})\dot{\mathbf{x}} = \mathbf{g}(\mathbf{x}), \tag{4}$$

where $\mathbf{x} = (x_1^1, x_1^2, x_1^3, x_2^1, \ldots, x_N^3)^{\mathrm{T}} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)^{\mathrm{T}}$ is the $3N$-vector containing the x, y, and z coordinates of all N interface nodes, $\mathbf{C}(\mathbf{x})$ is the matrix of inner products of basis functions, and $\mathbf{g}(\mathbf{x})$ is the right-hand side of inner products involving surface curvature. Since $\hat{\mathbf{n}}\hat{\mathbf{n}}^{\mathrm{T}}$ is a 3×3 matrix, it is clear that $\mathbf{C}(\mathbf{x})$ has a 3×3 block structure. The system of ODEs (4) along with boundary conditions is solved with an implicit second order backwards difference variable time-step ODE solver [2]. We use generalized minimal residual (GMRES) iteration [18] with block-diagonal preconditioner to solve the linear equations arising from the Newton's method. The ODEs are uniformly scaled so that the computational domain is $\mathcal{O}(1)$ and the truncation error η (i.e., the maximum acceptable

estimated error made in the x_j^k every time step) is usually set to 10^{-3}.

Our computational domain is rectangular $\left(\Omega = \prod_{k=1}^{3} [a_k, b_k]\right)$ and we take as boundary condition that nodes on the six planar bounding surfaces of Ω are constrained to frictionlessly slide on those surfaces. Since the Herring equation is satisfied where the computational interfaces intersect the external boundary (see below), this implies that our interfaces perpendicularly intersect $\partial\Omega$.

2.1. Computation of curvature forces

Since the curvature K is zero on the interior of triangle faces and is infinite at the edges, the inner products involving curvature must be computed using a mollification argument which is presented in [11]. The result of the derivation is that the right-hand side of (4) corresponding to the ith node is given by

$$\int \mu\sigma K\hat{\mathbf{n}}\alpha_i \,\mathrm{d}S = -\mu \sum_{\text{triangles } k \text{ with } i\in k} \frac{1}{2}\sigma L_k^{\text{opp}}\hat{\mathbf{t}}_k^{\text{opp}} = -\mu\nabla_{\mathbf{x}_i}\left(\sum_{\text{triangles } k \text{ with } i\in k} \sigma A_k\right), \qquad (5)$$

where the sum runs over the triangles k which have i as a vertex. A_k is the area of triangle k; L_k^{opp} is the length of the edge opposite node i on triangle k; $\hat{\mathbf{t}}_k^{\text{opp}}$ is the "inward edge normal" which is in the plane of triangle k, is normal to the edge opposite node i and points towards node i.

Eq. (5) shows that the node-concentrated "right-hand side force" on the ith node is equal to the negative gradient, with respect to the node position variables $\mathbf{x}_i = (x_i, y_i, z_i)$, of the surface integral of the surface energy density σ on our GWMFE piecewise linear surface. This mirrors the physical model where mean curvature motion is derived under the assumption that the mechanism of grain growth is minimization of excess surface energy proportional to the surface integral of σ. This negative gradient force on the ith node is exactly that which would be given by a "surface tension" of magnitude σ in the planar triangular cells of our GWMFE piecewise linear surface, as pointed out in [3,14].

We note here that the form (5) for the right-hand side PDE driving terms makes conceivable computations where $\sigma = \sigma(\hat{\mathbf{n}})$ that is, where σ depends on the direction of the normal at the surface. In this case, one need only loop through triangles in the mesh and evaluate the quantities (5) using the inclination-dependent σ. If σ depends on $\hat{\mathbf{n}}$, then triangles k incident on i will impart torque forces as well as surface tension forces. (In the computational examples in this paper, σ is not inclination-dependent.) If σ depends strongly on $\hat{\mathbf{n}}$, faceting of the surface can occur [7].

Adding up all the triangle contributions (5) from each of the three incident surfaces for each node i that exists on a triple line—a line where three grains meet—imposes in a natural fashion a balance of surface tensions and surface torques at the triple line. The means we satisfy the Herring equation [9] at triple lines which simply requires that surface tensions and torques of incident interfaces balance. In the constant σ case, this implies the interfaces make 120°–120°–120° angles at the triple line.

2.2. Regularization forces

Although the GWMFE minimization principle (3) determines grid point motion normal to the interface, it does not determine grid point motion tangential to the interface. In non-planar regions this is acceptable since the normals of the triangles incident on each node i span the three-dimensional space and motion of node i will be uniquely determined. However, if node i is coplanar with its neighbors (to within the local truncation error accuracy η), minimization of (3) will not address motions in the plane of the interface, leading to singularity or near-singularity of the matrix $\mathbf{C}(\mathbf{x})$. As detailed in [11], we solve this problem by adding tiny grid viscosity forces that only dominate in the cases where $\mathbf{C}(\mathbf{x})$ is nearly singular. These forces depend on the discretization of the grid and the local truncation error η, and they do not appreciably effect the accuracy of the solution with the value of η used in practice.

In addition, we add triangle quality forces that are negligible until surface triangles are of such poor shape that their inscribed radius is less than η.

In this case, the regularization forces take over and effectively "lock up" the triangle until it can be removed by grid maintenance operations described in Section 3.

See [11] for a full discussion of regularization forces and convergence studies where the solution of (1) is known exactly.

2.3. Enlargement of system to move non-interface nodes

System (4) gives velocities of the interface nodes only $(\dot{\mathbf{x}} = (\dot{x}_j^k)_{,1\leqslant j\leqslant N}^{,1\leqslant k\leqslant 3})$, and so the system must be enlarged to include velocities for M interior nodes that are not part of the interface. That is, we extend $\mathbf{x}$ to

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_{\text{interface}} \\ \mathbf{x}_{\text{interior}} \end{pmatrix},$$

where $\mathbf{x}_{\text{interface}} = (x_j^k)_{,1\leqslant j\leqslant N}^{,1\leqslant k\leqslant 3}$, and $\mathbf{x}_{\text{interior}} = (x_j^k)_{,N+1\leqslant j\leqslant N+M}^{,1\leqslant k\leqslant 3}$. With this extension, we enlarge system (4) to be order $N+M$.

Since interface physics only tells us how to evolve the N interface nodes, we must "artificially" construct the extra elements in the enlarged $\mathbf{C}(\mathbf{x})$, $\mathbf{g}(\mathbf{x})$ to allow for orderly (tetrahedra orientation preserving) evolution of the mesh. That is, given a physically meaningful method of evolving the triangular interfaces, we are free to develop auxiliary equations for moving the tetrahedra (some of which are conformally attached to the triangular interfaces) with the only requirement being that these equations lead to efficient solution of the system (4), and that they maintain positive orientation of tetrahedra. As detailed in [11], we add "quality" forces to maintain good tetrahedral aspect ratios. These forces are chosen to be tiny, so that where they affect a tetrahedral vertex that happens to be on an interface, the effect on the physically justified curvature forces is negligible until the tetrahedron has an inscribed radius of order of the truncation error η of the integrator. For large grains the effect is negligible, but for small grains it will effectively stop the collapse of grains when they reach a diameter of order η; then grid manipulation operations must explicitly remove the tiny grains. These grid operations are discussed in the next section.

One may ask why not just move interfacial triangles and entirely dispense with the tetrahedral discretization of the grain interiors. One reason is that we now have a moving volume mesh available for computation of ambient quantities that could affect grain growth. For example, computations have been done with Grain3D where heat diffusion was computed on the tetrahedral mesh and used in conjunction with a temperature-dependent interface mobility [1]. A second reason is that the presence of the tetrahedral mesh provides extra information for handling topological issues of grain collapse and grain neighbor switching.

3. Grid maintenance operations

Two tool sets that optimize and maintain the computational mesh are required to allow our simulations to run to completion. The first set combines node smoothing, node merging, edge refinement, and face swapping into a LaGriT tool called "graph massage" which is designed to prevent the mesh from tangling and to insure well-shaped elements. The second set, called "popcomponents", detects changes in interface topology and modifies the mesh to respond to these changes.

As the grain interfaces move, the tetrahedra attached to the interface triangles become stretched or compressed. To maintain good element quality and to prevent fatal mesh tangling, "graph massage" is invoked periodically. Graph massage accepts as input three lengths: a maximum edge length $L_{\max}$, a minimum edge length $L_{\min}$, and a damage tolerance. The first two parameters control edge bisection and node merging. If the edge is longer than $L_{\max}$, the edge will be bisected. If an edge is shorter than $L_{\min}$, then one of the two endpoints of the edge will be merged into the other endpoint. The choice of which endpoint to merge is based on which results in the best shaped elements and the least "damage" as defined below. Graph massage characterizes a node by the number and identity of the grains containing the node. Merges are allowed between nodes having exactly like characters. Additionally, a node whose char-

acter is a subset of a neighboring node may be merged into that node but the inverse merge is not allowed. For example, an interior node may be merged into an interface node, or an interface node into a triple line node, however the inverse merges are forbidden. This restriction guarantees that quadruple points at the ends of triple lines are not merged away and that the grain interfaces are preserved. After the merge/bisection step, reconnection will swap faces, so as to maximize the elements' inscribed radii.

The final parameter, damage tolerance, controls how much interface and boundary "damage" is allowed, where "damage" is a measure of the distortion that results from one of the optimization operations and is only a factor for nodes on interface surfaces. The distortion is the perpendicular distance from the original node position on an interface surface to the new position. In Fig. 1 assume that edges AB, BC exist on a triple line separating 3 grains. Then merging node B into either node A or node C would result in a change in the shape of the triple line. It would be pulled straight. We define the "damage" to be the height "h" from the original position of node B to the new edge AC. An analogous definition handles the case where the node to be merged away lies on an interface surface but not on an triple line. In this case, the "damage" is the perpendicular distance from the node's original position to the new surface. We disallow any operation that would cause damage greater than the input damage tolerance. If the damage tolerance is small, one can expect only small node movements, few node annihilations, and few edge swaps involving elements at curved portions of material interfaces or boundaries. Conversely, a large value can result in significant deformation. We choose the value large enough so that sufficient grid changes are allowed to keep the simulation going, but not so large that grid damage dominates other sources of error, such as allowed local truncation error in the ODE integrator.

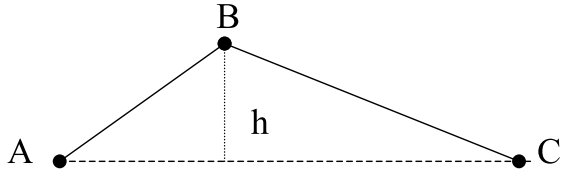

Fig. 1. h is the "damage" that would result from merging node B into either node A or node C.

Graph massage forbids any operation that would cause element inversion or near-inversion. Periodically during the grain evolution, LaGriT graph massage is invoked by Grain3D to improve mesh quality. All operations are performed in a way that guarantees that the mesh connectivity will not become corrupted and that ensures the integrity of grain interfaces. In this context LaGriT is linked into Grain3D as a toolbox and its operations are controlled by the grain evolution code.

The second category of mesh optimization tool, "popcomponents", captures changes in interface topology [12]. We keep track of the various connected topological components and determine whether they are on the verge of disappearing or changing as the simulation evolves. The connected topological components are the tetrahedral sets comprising each grain, the interface triangle sets between each pair of grains, and the edge sets that bound the triangle sets (these sets constitute the triple lines of intersection between triplets of grains). For example, for all pairs of grains that touch each other, we monitor the topological component consisting of connected interface triangles between the two grains. If the total surface area of one such component is about to go to zero, a physical topological change is imminent; Grain3D detects this change and responds by invoking the LaGriT toolbox set to perform topological operations on the computational tetrahedral mesh. Similar strategies are required to detect other forms of topological change. We identify the neighborhood surrounding the collapsing feature and refine this neighborhood to provide a thin buffer region. The encroaching grains are examined to identify a "winner"—usually that grain which subtends the largest angle, and the collapsing feature and its associated neighborhood are assigned to the winning grain. This reassignment effectively destroys the collapsing feature and thus repairs the topology. For details see [11,12].

4. Experimental validation

Grain3D is an appropriate code to simulate thermal annealing. To validate our computational model, we experimentally evolved columnar microstructures of a total of 5170 grains in 19 thin aluminum foil samples. We then compared the computed evolved microstructure with the final experimental state at the level of individual grains. The 1×1 cm^2 samples were 120 μm thick, cut from 99.98% pure Al foil [6]. The starting polycrystalline microstructure was annealed for 9 h at 550 °C and then electro-polished and measured using standard electron backscatter diffraction (EBSD) to give the initial microstructure for the simulation. The samples were then annealed a second time for 20 min at 550 °C to generate the final microstructure. The average interface velocity of typical grains of radius 100 μm was observed to be 0.1 μm/s [4]. (Note: It is reasonable to expect that at the large grain sizes (1–1000 μm) observed in this experiment, curvature-driven migration dominates, whereas rotation should dominate in nanoscale grain sizes [10,13].)

In order to create the computational meshes, we used the grain interfaces and triple junctions from the EBSD images and invoked the image processing algorithms detailed in [19]. The results of the image processing step are fed into LaGriT for grid generation; LaGriT is run in stand-alone mode to prepare an initial grid for input to Grain3D.

The first step in grid generation for our application is to separately triangulate each grain from the grain boundaries. The grain boundaries are given by the contour extraction algorithms [19] as sets of ordered pairs of coordinates. The result of the first step is a set of two-dimensional planar triangular meshes, one for each grain. Since the experimental structure is columnar, the second step is to extrude each grain's two-dimensional mesh in a direction perpendicular to the plane, resulting in a three-dimensional prism mesh for each grain. Then each prism is divided into three tetrahedra. Note that the extrusion step doubles the number of nodes, but that converting the prisms to tetrahedra adds no additional nodes to the mesh. Third, the exterior boundary is extracted from each grain's three-dimensional volume mesh; each boundary mesh is a triangular mesh that is topologically two-dimensional but geometrically three-dimensional. The first three grid generation steps are shown in Fig. 2.

The fourth step results in a single conformal three-dimensional mesh. The individual boundary meshes created in step three are used as surfaces, and the area inside each boundary mesh is identified as a separate geometric region. When meshing, LaGriT creates conforming interfaces where regions meet, and this capability in this context results in each grain being meshed into a separate region. All nodes from the boundary meshes are copied into the combined mesh. At this point additional nodes may be added to the mesh. In the

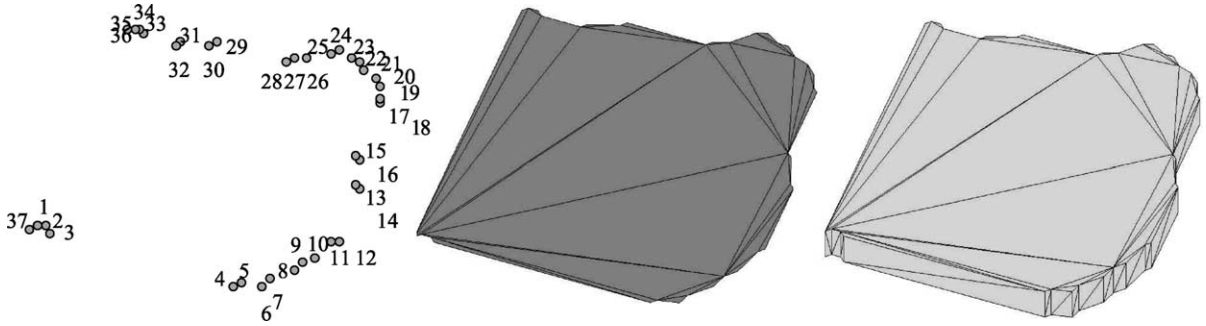

Fig. 2. Three steps in mesh generation. The left figure shows the ordered set of nodes defining a typical grain boundary. The center figure shows the triangulated single planar grain defined by the nodes in the preceding step. The right figure shows the surface mesh of a single grain generated by extruding the mesh created in the preceding step and then extracting the surface.

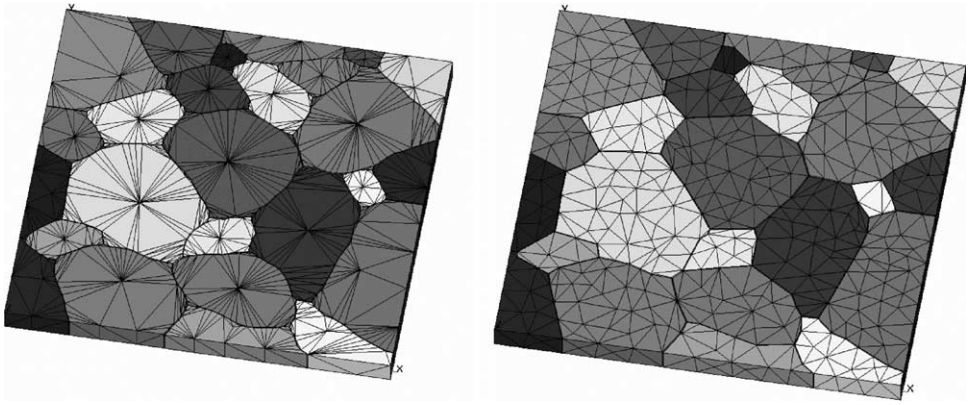

Fig. 3. Computational mesh for a 26 grain example before and after graph massage.

simple example given in Fig. 3, a node is added near the center of the top and bottom surfaces of each grain. Tetrahedral elements are formed in the combined mesh using a Watson point insertion algorithm [20]. The boundaries of the individual grains are preserved.

At this point, the quality of the initial three-dimensional mesh is quite poor and graph massage is invoked to improve quality. Fig. 3 shows a simple 3-D computational mesh before and after graph massage. The right-hand figure is a small initial computational mesh to be input to Grain3D. The actual initial computational meshes contained on average 250 grains; the smaller grid is shown to clearly illustrate the process used.

In Grain3D grain boundary motion is proportional to local mean curvature of the interface. To obtain values for the grain boundary mobility μ and the interface energy σ, we used the EBSD measured grain misorientation and previously determined statistical analysis of triple junction geometry and crystallography in aluminum [21]. Note that this process results in a single interface misorientation for each pair of grains, giving an average σ for each grain boundary. Sometimes the different average σ's from different grain boundaries made it impossible to set up a consistent satisfaction of force balance at triple lines. (After all, the *average* σ along a grain boundary is not necessarily the limiting value of σ for that boundary as it approaches a triple line.) Since this was unacceptable for computation, we limited our simulations to the case where σ was constant and mobility was either isotropic (μ constant and equal to 1) or anisotropic (μ varies according to the crystallographic orientation of the grains).

For a quantitative comparison of simulation and experiment, we introduce a normalized area match function NAMF [5] over a dense regular grid of sampling points. One may think of this comparison method as overlaying the final experimental state onto the successive simulation snapshots and measuring at each simulation snapshot the percentage of area where the grain orientations match. The higher the value, the better the match.

Because of the short annealing time, the value of NAMF at the initial time is relatively high. In a representative run, we find a 62% match between initial and final experimental states. Examining the amount of match shown in successive simulation snapshots, we find that using isotropic properties the match improves to a maximum of only 68%, whereas using anisotropic mobility the match improves to 82%. Comparing simulation to experimental results for all 19 runs bore out the pattern that isotropic simulation was of little predictive value, while anisotropic simulation did a good job of predicting the experimentally evolved microstructure as measured by NAMF. The failure of the anisotropic simulations to be perfectly predictive (i.e., produce NAMF $= 1$) may be due to the failure to incorporate energy anisotropy and may also be due to sample error, dislocation density, or stored deformation energy. It may also be due to

computational indeterminacy due to topology changes. An example of this is where an unstable "quadruple" line may have two equally favorable ways of splitting into pairs of triple lines, and our simulation chooses the "wrong" way. For a full discussion of these results, see [5].

5. Fully 3-D grain evolution simulations

For a fully 3-D non-columnar simulation, we evolve the grain boundary surfaces on large systems consisting of on the order of 5000 grains at the initial simulation time. Currently there is no fully three-dimensional non-destructively measured experimental data to compare to. Here, the purpose of these simulations is to test the self-similarity hypothesis of Mullins [16]. This hypothesis requires that the same equation for grain boundary motion be taken to be valid over a particular range of length scales and that the grain size distribution evolve in a self-similar fashion. In the previous section, we noted that Eq. (1) should be valid over the length scale of grain sizes between 1 and 1000 μm. If grain evolution occurs in a self-similar fashion, the mean grain size D should evolve as

$$D^2(t) - D^2(0) = Bt,$$

with B a constant [16]. This is known as the parabolic prediction of normal grain growth. Consequently, the evolution of the mean grain volume V should be well-fit by a curve of the form

$$V(t) = (A + Bt)^{3/2}, \quad A, B \text{ constant}.$$

In our 3-D simulations we use isotropic mobility and energy and disregard grain orientation. The starting configurations for the simulations are obtained by using a pseudo-random process of choosing the desired number (5000) of nucleation sites. We then grow the grains using a constant velocity until grains collide at which time they stop growing. This initial configuration is input to Grain3D which evolves the grain boundaries according to (1) but with the simple case of isotropic mobility and energy.

We computed on the dimensionless domain $\Omega = [-1, 1]^3$, using unit dimensionless reduced mobility ($\mu\sigma = 1$), and plotted data for four computational runs in Fig. 4 (showing the evolution of dimensionless average grain volume versus dimensionless time). To put this in concrete terms, if we assume that the domain corresponds to [−1 mm, 1 mm]^3, then the initial average grain volume is (2 mm)^3/5000 = 0.0016 mm^3, and grows in the simulations to 0.016 mm^3, corresponding to an approximate doubling in grain size. The typically observed reduced mobility for Al grains that were simulated previously in [5] is about 10^{-5} mm^2/s, so the x-axis units would correspond to units of 10^5 s, i.e. the x-axis runs from 0 to 4000 s.

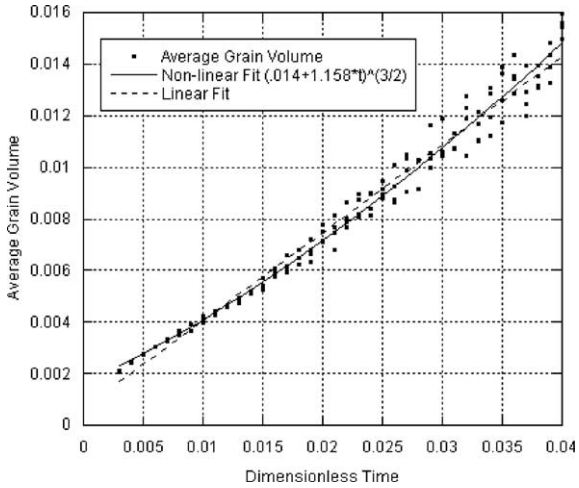

Fig. 4. Best fit of function of form $(A + Bt)^{3/2}$ through computed mean grain volume versus time data for 3-D simulation, along with best linear fit.

The exterior surfaces of grains that lie on the surface of the computational cube are required to remain planar. Because of this boundary effect, we gather statistics only from grains all of whose surfaces are strictly interior to the computational domain.

We ran four simulations varying the initial conditions by changing the seed of the grain placement algorithm and by changing the initial node density. We look at the results for 40 output time steps at which time there were still over 100 interior grains. We ignored the first couple of time steps (where we hypothesize the average grain volume size distribution was unduly influenced by the initial grain placement algorithm) and fit the

remaining data with a linear and non-linear fit. The data and fits are shown in Fig. 4. RMS fit error for the simple linear fit is 5.81e−4 whereas the RMS fit error for the non-linear fit is 4.88e−4. The superiority of the non-linear fit to a 3/2 power agrees with predictions of [16]. However, it must be stated that larger data sets of >5000 grains would provide a more convincing demonstration of grain growth self-similarity. The current 5000 grain data sets involve computational runs of approximately one-half million tetrahedra and represent approximately the largest problems we can feasibly run on single-processor workstations. Larger data sets await parallelization of the code.

6. Grain3D, LaGriT specifications

In Grain3D computations, the major portion of the computational time (≈90%) is used to move the interfaces and the remaining 10% to evaluate and update the mesh. Computational time varies with the size of the problem and hardware used. For example, each of the 19 simulations of Section 4 took on the order of 2 h on a DEC alpha workstation.

Grain3D is written entirely in FORTRAN 77. LaGriT is written primarily in FORTRAN 77, but contains a few FORTRAN 90 modules and an I/O package and memory manager written in C. LaGriT is available for UNIX and LINUX platforms. The user manual, examples, and information on obtaining LaGriT are found at http://www.t12.lanl.gov/home/lagrit. To obtain Grain3D, contact the authors.

7. Conclusions

We have used a combination of a 3-D grain evolution code Grain3D and a 3-D grid generation and optimization code LaGriT to simulate microstructure evolution in Al and have demonstrated that our simulation results agree well with experiment in columnar microstructures and with the parabolic law of normal grain growth in fully 3-D microstructures.

Acknowledgements

This research is supported by the Department of Energy under contract W-7405-ENG-36.

References

[1] N. Carlson, Unpublished work. Available at <http://www.t12.lanl.gov/home/lagrit/picturesweek.html>.
[2] N. Carlson, K. Miller, Design and application of a gradient-weighted moving finite element code I: in one dimension, SIAM J. Sci. Comput. 19 (1998) 728–765.
[3] N. Carlson, K. Miller, Design and application of a gradient-weighted moving finite element code II: in two dimensions, SIAM J. Sci. Comput. 19 (1998) 766–798.
[4] M.C. Demirel, Linking experimental characterization and computational modeling in microstructural evolution, Ph.D. Thesis, Materials Science and Engineering Department, Carnegie Mellon University, Pittsburgh, 2002, p. 80ff.
[5] M.C. Demirel, A.P. Kuprat, D.C. George, A.D. Rollett, Bridging simulations and experiments in microstructure evolution, Phys. Rev. Lett. 90 (2003) 016106.
[6] M.C. Demirel, A.P. Kuprat, D.C. George, G.K. Straub, A.D. Rollett, Linking experimental characterization and computational modeling of grain growth in Al-foil, Interf. Sci. 10 (2–3) (2002) 137–141.
[7] J.T. Gammel, A. Kuprat, Modeling metallic microstructure: incorporating grain boundary orientation dependence, in the special features supplement to the theoretical division self-assessment, LA-UR-98-1150, Supplement, Los Alamos National Laboratory Unclassified Report, Spring 1998.
[8] D.C. George, LaGriT User's Manual. Available at <http://www.t12.lanl.gov/home/lagrit> and is documented in Los Alamos National Laboratory Unclassified Report LAUR-95-3608, 1995.
[9] C. Herring, Some theorems on the free energies of crystal surfaces, Phys. Rev. 82 (1) (1951) 87–93.
[10] H. Hu, in: L. Himmel (Ed.), Recovery and Recrystallization of Metals, Interscience, New York, 1963, p. 362.
[11] A. Kuprat, Modeling microstructure evolution using gradient-weighted moving finite elements, SIAM J. Sci. Comput. 22 (2) (2000) 535–560.
[12] A. Kuprat, Inspecting and repairing physical topology in a moving grid grain growth simulation, in: Proceedings of the 7th International Conference on Numerical Grid Generation in Computational Field Simulations, Whistler, Canada, International Society of Grid Generation, Mississippi State University, 2000, pp. 541–550.
[13] G. Martin, Driving force and mobility for microstructural evolutions—the rate of grain rotation across a

grain-boundary, Phys. Status Solidi B 172 (1) (1992) 121–131.
[14] K. Miller, A geometrical–mechanical interpretation of gradient-weighted moving finite elements, SIAM J. Numer. Anal. 34 (1997) 67–90.
[15] W.W. Mullins, 2-Dimensional motion of idealized grain boundaries, J. Appl. Phys. 27 (1956) 900.
[16] W.W. Mullins, The statistical self-similarity hypothesis in grain growth and particle coarsening, J. Appl. Phys. 59 (1986) 1341–1349.
[17] D.A. Porter, K.E. Easterling, Phase Transformations in Metals and Alloys, second ed., Chapman & Hall, London, 1992, pp. 130–136.
[18] Y. Saad, M.H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 7 (1986) 856–869.
[19] B.R. Schlei, Region enclosing contours from edge pixels, in: SPIE Conference Proceedings—Vision Geometry XI, Seattle, vol. 4794, 2002.
[20] D.F. Watson, Computing the N-dimensional Delaunay tessellation with application to Voronoi polytopes, Comput. J. 24 (2) (1981) 167–172.
[21] C.-C. Yang, A.D. Rollett, W.W. Mullins, Measuring relative grain boundary energies and mobilities in an aluminum foil from triple junction geometry, Scr. Mater. 44 (12) (2001) 2735–2740.