Bounding box algorithm for three-dimensional phase-field simulations of microstructural evolution in polycrystalline materials

Liesbeth Vanherpe,[1,*] Nele Moelans,[2,†] Bart Blanpain,[2] and Stefan Vandewalle[1]
[1]*Department of Computer Science, Faculty of Engineering, Katholieke Universiteit Leuven, 3001 Leuven, Belgium*
[2]*Department of Metallurgy and Materials Engineering, Faculty of Engineering, Katholieke Universiteit Leuven, 3001 Leuven, Belgium*
(Received 24 May 2007; published 7 November 2007)

Phase-field modeling has proven to be a versatile tool for simulating microstructural evolution phenomena, such as grain growth in polycrystalline materials. However, the computing time and computing memory requirements of a phase-field model pose severe limitations on the number of phase-field variables that can be taken into account in a practical implementation. In this paper, a sparse bounding box algorithm is proposed that allows the use of a large number of phase-field variables without excessive memory usage or computational requirements. The algorithm is applied to a three-dimensional model for grain growth in the presence of second-phase particles.

DOI: 10.1103/PhysRevE.76.056702 PACS number(s): 02.60.Cb, 81.10.Aj, 64.10.+h, 02.70.−c

I. INTRODUCTION

The microstructure of materials often consists of multiple grains with different crystallographic orientations. The study of the evolution of these grains is of great technological importance, because many material properties, such as strength and toughness, depend on the mean grain size and grain size distribution. A large number of theoretical, experimental, and computational studies have been performed on this subject (see, for example, Refs. [1–5]). Although it is well understood that grain growth behavior results from the interplay between curvature-driven grain boundary movement and the geometrical requirements at triple junctions, there are still many controversies, especially with respect to the shape and evolution of the grain size distribution. Computer simulations based on mesoscale models such as Monte Carlo, vertex and phase-field models, and cellular automata, are essential for a better understanding of grain growth. They allow one to study separately the roles of different parameters, which is impossible in experimental studies on real materials. Moreover, images of three-dimensional simulations provide more insight in the shape and size of grains than two-dimensional microscopic images of cross sections of a material.

Phase-field modeling has shown to be a versatile tool for simulating microstructural evolution phenomena, such as solidification [6–8], precipitation [9,10], and grain growth [11–13]. It allows to predict the evolution of complex morphologies considering different thermodynamic driving forces, such as interfacial energy, bulk energy, and elastic energy, and different transport processes, such as heat and mass diffusion. Grain growth in single-phase polycrystalline materials has been simulated with the phase-field method by different authors [14–16]. The model of Chen and Yang was extensively discussed and results were compared to experimental observations and analytical theories in [11,17]. An extension of the model towards two-phase materials was proposed in [18–20], and applied for simulating Zener pinning by growing second-phase particles in [21], and for particles which are constant in time in [22,23]. In [14], the phase-field model represents the microstructure of a single-phase material by a set of p nonconserved phase-field variables $\eta_1(\mathbf{r},t)$, $\eta_2(\mathbf{r},t),\ldots,\eta_p(\mathbf{r},t)$. These phase-field variables are used to distinguish the different crystallographic orientations of the grains. Inside a grain, one phase-field variable η_i takes a nonzero constant value, mostly 1 or −1, while the other phase-field variables assume values close to zero. Across the grain boundaries, the corresponding field variables vary continuously from their equilibrium value in the grain to their equilibrium value in the neighboring grains. The spatial and temporal evolution of the phase-field variables is governed by the time-dependent Ginzburg-Landau equations. In principle, the number of phase-field variables p should equal or exceed the total number of grains, as in reality the number of possible grain orientations is infinite. When not enough crystallographic orientations are involved in a grain growth simulation, growth can occur by coalescence of two neighboring grains with the same orientation. This leads to incorrect growth kinetics and unphysical grain shapes. Especially in three-dimensional simulations, where grains have on average more neighbors than in two dimensions, a very large number of crystallographic orientations are required to minimize the effect of grain coalescence [12]. Moreover, for anisotropic materials, it is important that the orientation dependence of material properties is resolved well. Furthermore, when the pinning effect of particles is modeled, the spatial resolution of the employed numerical technique has to be fine enough in order to represent the particles, which are much smaller than the grains, and reproduce the shape of grain boundaries at particle-grain boundary intersections correctly. To conclude, since one is mostly interested in the evolution of the grain size distribution, a large amount of grains (and particles) must be considered in grain growth simulations to achieve reliable statistics. As a consequence, realistic three-dimensional computer simulations for grain growth with a phase-field model demand significant amounts of computation power.

Several algorithms have been designed to overcome the computational limitations of the phase-field method. In

*liesbeth.vanherpe@cs.kuleuven.be
†nele.moelans@mtm.kuleuven.be

 ©2007 The American Physical Society

[12,24], the grain orientations are dynamically reassigned to reduce memory requirements and to avoid frequent grain coalescence. This approach is limited to systems where the only use of the phase-field variables is to distinguish unique domains. Incorporating anisotropy or any property depending on the relative or absolute position of a grain or the orientation difference between neighboring grains into this technique is impossible. Adaptive meshing [25–27], and moving mesh techniques [28,29], have been used in phase-field simulations to increase computational efficiency. However, so far these techniques were always applied for simulations with a small number of phase-field variables. For polycrystalline structures their benefits are drastically reduced, because of the amount of interface involved.

In [30–32], efficient algorithms using a sparse data structure are proposed. These algorithms are based on the observation that only a few crystallographic orientations are active at a given point in a microstructure. In [30], the microstructural evolution is only computed for nonzero phase-field values. A simple data structure is proposed which maintains a dynamic vector of the indices and the values of the nonzero phase-field variables at each grid point. In [31], a similar data structure is employed. Furthermore, a simple test condition is used, which distinguishes the evolving phase-field values at grain boundaries from the constant phase-field values interior to grains. The model equations are then only solved for the evolving phase-field values. Both sparse algorithms are designed for the model of [14]. In [32], an efficient algorithm is constructed in the context of the models of [33,34]. The algorithm also uses a similar data structure, but puts a restriction N_p on the maximum number of nonzero phase-field variables at every grid point and only stores the indices and the values of the N_p phase-field variables with the largest magnitudes. These sparse algorithms show significant improvements over conventional algorithms as they scale with the size of the microstructure instead of with the number of crystallographic orientations involved. However, all three algorithms are designed for explicit time integration. Explicit time-integration schemes do not involve the solution of coupled equations, but can require large amounts of computing time due to impractical small time steps, imposed by strong stability conditions. In contrast, implicit and semi-implicit schemes require more computations due to involved coupled equations, but have better stability properties and therefore, larger time steps can be used. In [35], a semi-implicit scheme is shown to allow much larger time-step sizes than explicit schemes for the model of [14]. Unfortunately, it is difficult to apply this semi-implicit scheme to the data structure of [30–32]. Furthermore, each of the described sparse algorithms employs a data structure that maintains, besides the values of the active phase-field variables at every grid point, also their indices. For models which include anisotropic boundary energies, a logical extension to this data structure is to store additional material parameters at each grid point. However, there is inherent overhead associated with this storage scheme, since the same values are stored at multiple locations. Finally, one of the main goals of grain growth simulations is to study the evolution of the mean grain size and the grain size distribution. For this purpose, the grains of a microstructure have to be located, and their volume has to be computed. However, this data is not directly available in the sparse data structures of [30–32]. Therefore, it can take a relatively large amount of time to obtain the necessary information, especially when this post-processing is repeated for multiple microstructures at different moments in their evolution.

In this paper, a bounding box algorithm, based on the same observations that led to [30–32], is presented, however now for semi-implicit time integration. The algorithm is designed in the context of the study of grain growth in the presence of second-phase particles according to the model of [14,22]. However, it can be applied to other cases as well. Thanks to the object-oriented design, the algorithm can easily be extended to more complex phase-field models, for example, for grain growth in materials which show orientation dependence. Moreover, the object-oriented approach has definite advantages in postprocessing.

The structure of the paper is as follows. The phase-field model for which we developed the bounding box algorithm is presented in Sec. II. In Sec. III, the bounding box algorithm itself is explained. Then, the simulation conditions are specified in Sec. IV. The bounding box algorithm is validated for three-dimensional simulations in Secs. V and VI. In Sec. VII, the algorithm is applied to grain growth in the presence of second-phase particles and simulation results are discussed. Some concluding remarks are formulated in Sec. VIII.

II. PHASE-FIELD MODEL AND DISCRETIZATION

The bounding box algorithm has been implemented for phase-field models for grain growth as described in [14]. According to [14], the microstructure of a single-phase material is represented by a set of p phase-field variables

$$\eta_1(\mathbf{r},t), \eta_2(\mathbf{r},t), \ldots, \eta_p(\mathbf{r},t). \tag{1}$$

The phase-field variables are used to distinguish the different crystallographic orientations of the grains and are continuous functions of the spatial coordinates and time. The spatial and temporal evolution of the phase-field variables is governed by the time-dependent Ginzburg-Landau equations,

$$\frac{\partial \eta_i(\mathbf{r},t)}{\partial t} = -L\frac{\partial F}{\partial \eta_i(\mathbf{r},t)}, \quad i = 1, \ldots, p. \tag{2}$$

The kinetic coefficient L is related to the grain boundary mobility. The free energy F of the system is described by

$$F = \int_V \left[\sum_{i=1}^{p} \left(\frac{\eta_i^4}{4} - \frac{\eta_i^2}{2} \right) + \sum_{i=1}^{p} \sum_{j \neq i}^{p} \eta_i^2 \eta_j^2 + \frac{\kappa}{2} \sum_{i=1}^{p} (\nabla \eta_i)^2 \right] d^3\mathbf{r}, \tag{3}$$

with κ the gradient energy coefficient.

In [22,23], this phase-field model is extended for simulating grain growth in materials containing small incoherent second-phase particles with constant properties. To include such particles in the model, a spatially dependent parameter ϕ is added. This parameter ϕ equals 1 inside a particle and 0 elsewhere and remains constant in time. The free energy F of the system is now described by

$$F = \int_V \left[\sum_{i=1}^{p} \left(\frac{\eta_i^4}{4} - \frac{\eta_i^2}{2} \right) + \sum_{i=1}^{p} \sum_{j \neq i}^{p} \eta_i^2 \eta_j^2 + \phi^2 \sum_{i=1}^{p} \eta_i^2 + \frac{\kappa}{2} \sum_{i=1}^{p} (\nabla \eta_i)^2 \right] d^3\mathbf{r}. \quad (4)$$

Substituting Eq. (4) into Eq. (2) results in a set of reaction-diffusion partial differential equations

$$\frac{\partial \eta_i}{\partial t} = L \left[\kappa \nabla^2 \eta_i - \eta_i^3 + \eta_i - 2\eta_i \left(\sum_{j \neq i}^{p} \eta_j^2 + \phi^2 \right) \right], \quad i = 1, \ldots, p. \quad (5)$$

For isotropic grain boundary energy and mobility the coefficients κ and L are constants. Periodic boundary conditions are applied, as in [36], p. 106.

Equations (5) are solved by using a finite difference scheme. The spatial derivative is discretized with second-order central differences. For $\mathbf{r}=(x,y,z)$, we have

$$\nabla^2 \eta_i(\mathbf{r}) = \sum_{v=x,y,z} \frac{\eta_i(v+\Delta v) - 2\eta_i(v) + \eta_i(v-\Delta v)}{(\Delta v)^2}, \quad (6)$$

where Δx, Δy, and Δz denote the mesh widths in the finite difference grid. The time derivative is discretized using a first-order semi-implicit scheme [35]. The diffusion part is treated implicitly, the reaction part explicitly:

$$\frac{\eta_i^{n+1} - \eta_i^n}{\Delta t} = L \left\{ (\kappa \nabla^2 \eta_i)^{n+1} + \left[\eta_i^3 + \eta_i - 2\eta_i \left(\sum_{j}^{p} \eta_j^2 + \phi^2 \right) \right]^n \right\}, \quad i = 1, \ldots, p, \quad (7)$$

where the superscript n indicates the solutions at time-step n. The implicit treatment of the Laplacian, combined with the explicit treatment of the nonlinear coupling of the phase-field variables, allows the use of a large time step without the need to solve one very large coupled system of equations. It effectively decouples the system into p scalar diffusion equations to be solved at every time step.

III. BOUNDING BOX ALGORITHM

A phase-field simulation is both memory consuming and computationally intensive. There are several ways to speed up the phase-field computations. Because of the explicit treatment of the coupling of the phase-field variables, the equations can be assigned to different processors and solutions can be computed in parallel [24]. A closer look at a grain growth simulation reveals that the solutions of the model display small regions of high activity surrounded by large regions of inactivity. This property, combined with the semi-implicit discretization, is exploited in this paper by only solving the equations locally. This technique lowers both storage requirements, computing time, and computing memory.

To initialize the bounding box algorithm, a polycrystalline microstructure is required. The initial microstructure can be obtained, for example, from microscopic images, Voronoi calculation, or by simulation.

A phase-field variable is defined to be active at a given grid point when its absolute value exceeds a small positive threshold value ϵ. At every grid point, only a few phase-field variables are active and thus contribute to the evolution of the microstructure; inside each grain, away from its boundaries, one phase-field variable, η_i, is active and near grain boundaries, only those phase-field variables corresponding to the neighboring grains, are active. Based on these properties, a grain region $\hat{G}_{i,k}$ is now identified by its type i, its sequence number k, and a set of ordered couples $(\mathbf{r}, \eta_i(\mathbf{r}))$ with grid points $\mathbf{r}$ that are connected, and with $|\eta_i(\mathbf{r})| > \epsilon$:

$$\hat{G}_{i,k} = \{(\mathbf{r}, \eta_i(\mathbf{r})) : |\eta_i(\mathbf{r})| > \epsilon \text{ and } \mathbf{r} \text{ connected}\}. \quad (8)$$

The type i corresponds to a crystallographic orientation. The sequence number k ranges from 1 to n_i, with n_i the number of grain regions associated with η_i. Grain regions are allowed to wrap around the grid boundaries so that periodic boundary conditions are automatically taken into account. Note that the grain regions overlap, which allows the regions to interact with each other.

For every grain region $\hat{G}_{i,k}$, a bounding box is determined as the smallest cuboid grid part containing the grid points $\mathbf{r}$ of $\hat{G}_{i,k}$. The bounding box is completely characterized by the coordinates of two opposite delimiting grid points. The set $\hat{B}_{i,k}$ is then defined as

$$\hat{B}_{i,k} = \{(\mathbf{r}, \eta_i(\mathbf{r})) : \mathbf{r} \text{ lies inside the bounding box of } \hat{G}_{i,k}\}. \quad (9)$$

Having defined the active phase-field variables, grain regions $\hat{G}_{i,k}$ and sets $\hat{B}_{i,k}$, Eqs. (5) only have to be solved for the values contained in the sets $\hat{B}_{i,k}$.

The bounding box algorithm proceeds as in Fig. 1. The *preprocessing step* of the algorithm is illustrated in Fig. 2. First, for every phase-field variable η_i, all corresponding grain regions $\hat{G}_{i,k}$, $k=1,\ldots,n_i$ are located. Second, the corresponding bounding boxes and the sets $\hat{B}_{i,k}$ are established. Finally, a grain region numbering takes place and a new set of phase-field variables is introduced, such that there is a one-to-one mapping between grain regions and phase-field variables. More precisely, every grain region $\hat{G}_{i,k}$ for $i=1,\ldots,p$ and $k=1,\ldots,n_i$ is given a unique index l, ranging from 1 to q, the total number of grain regions, with $q=\Sigma_{i=1}^{p} n_i$. That is, grain region $\hat{G}_{i,k}$ is rewritten asG_l, a grain region corresponding to one of the new phase-field variables, η_l, for $l=1,\ldots,q$, with bounding box information copied from $\hat{B}_{i,k}$. This renumbering of grain regions and phase-field variables is allowed, since the actual value of the grain orientation does not play a role in Eqs. (5). In the situation that grains can divide into subgrains (e.g., recovery after deformation), the renumbering procedure can be reexecuted to guarantee a one-to-one phase-field variable to grain relation. The assignment of unique phase-field variables to every

```
<preprocessing step>
for phase field variables i = 1,...,p
   find grain regions Ĝ_{i,k}
   for k = 1,...,n_i
      determine bounding box delimiters and B̂_{i,k}
grain region numbering procedure
<computation step>
for time steps t = 1,...,n
   for l = 1,...,q
      solve equation (5) for B_l
      update bounding box delimiters and B_l
```

FIG. 1. Bounding box algorithm.

grain region ensures coalescence-free simulations with the bounding box algorithm.

In the bounding box data structure, the microstructure is treated as an object with a number of attributes: the grid size and, if present in the model, second-phase particles or other features of the polycrystalline microstructure. A grain region in turn is also treated as an object. Its attributes are the material properties κ and L, the bounding box delimiters, the type and the set B_l. Figure 3 shows a unified modeling language (UML) diagram of the object-oriented data structure. The line connecting the elements of the data structure indicates their association relationship. The notation at each end of the line indicates the multiplicity, which is the number of objects that participate in the association.

After the data structure is determined, the evolution of the microstructure is computed in the *computation step*. From the starting point of the bounding box algorithm on, the phase-field values not exceeding ϵ in absolute value are assumed to be zero. Therefore, a margin of one grid point, filled with zeros, can be added to the bounding boxes and sets $\hat{B}_{i,k}$ in accordance with this assumption. Equations (5) can thus be solved locally using homogeneous Dirichlet conditions and allow for the grain regions to grow inside the bounding boxes. The resulting systems can be solved with iterative methods, for example, with multigrid methods, the successive over-relaxation (SOR) method or the Gauss-Seidel method. The latter is the method used in this paper. Within each time step, for every set B_l in turn, the solution of Eq. (5) is computed using the discretization described in Sec. II. Depending on whether grain growth is simulated in the presence of second-phase particles, the additional term ϕ^2 is included in the equations. After the computation, the algo-

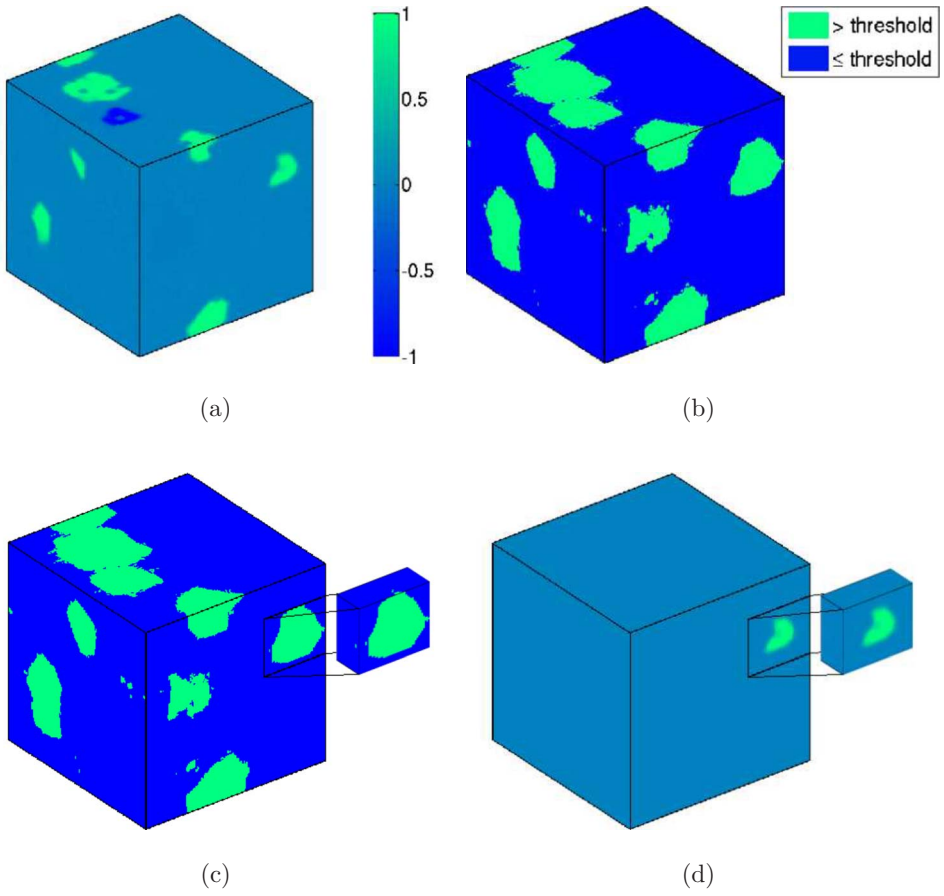


FIG. 2. (Color online) Locating the grain regions and determining the bounding boxes in the preprocessing step, when the initial microstructure is obtained from a random structure by simulation. (a) The values of one phase-field variable, η_i, vary between the two equilibrium values −1 and 1. (b) The threshold value ϵ is applied and the grain regions $\hat{G}_{i,k}$ are located. (c) The bounding boxes and the sets $\hat{B}_{i,k}$ are established. (d) A new phase-field variable η_l is assigned to every $\hat{G}_{i,k}$ and the values inside the corresponding $\hat{B}_{i,k}$ are isolated.

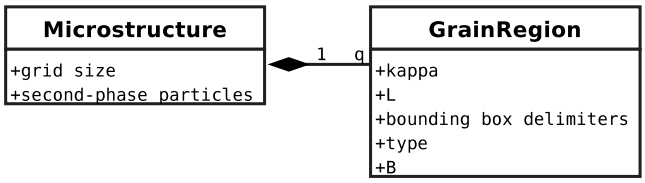


FIG. 3. UML diagram of the object-oriented data structure composed in the preprocessing step. The microstructure and grain regions are treated as objects with a number of attributes.

rithm checks whether the grain region G_l has shrunk or grown and adjusts the bounding box delimiters and the set B_l accordingly, taking into account the margin of zeros.

The object-oriented data structure allows for easy extension to more complex phase-field models. Misorientation or inclination dependence of the grain boundary energy and mobility can be inserted into the structure by adding attributes and objects (see Fig. 4). During the preprocessing step, it is then possible to apply orientation-dependent parameters to be used in the further simulations. As mentioned, a grain region is treated as an object with a number of properties. A property is stored once per grain region, and not for every grid point of the grain region. In the same way, grain boundaries can be treated as objects with specific properties.

The data structure of the bounding box algorithm can save significant time in postprocessing, e.g., when the number of grains or the mean grain size, by which we mean the mean grain volume, have to be determined. The number of grains and their location is known throughout the simulation and this information can be used immediately.

IV. SIMULATION TEST CASE

To obtain an initial polycrystalline microstructure for simulations with the bounding box algorithm, a parallel implementation was made of an algorithm solving Eqs. (5) globally, based on the semi-implicit finite-difference scheme proposed in [37]. This algorithm and its characteristics will be referred to as the conventional algorithm and the conventional characteristics. Simulations with the conventional algorithm were run on a $256\times256\times256$ grid. Because of memory limitations, only 100 phase-field variables were employed, whereas according to [12], more than 200 phase-field variables are required to prevent grain coalescence in a three-dimensional simulation. The parameter values were set to $\kappa=0.5$ and $L=1$ and no second-phase particles were included. The discretization spacings were $\Delta x=1$ and $\Delta t=0.2$. At the start of a simulation, small random values between -0.001 and 0.001 were assigned to the phase-field variables at all grid points. All computations were performed on 20 nodes of a computer cluster, which are interconnected with an Infiniband network. At simulation time t_s, the conventional algorithm was stopped and the preprocessing step of the bounding box algorithm was applied.

To visualize the microstructural evolution, the function ψ is defined,

$$\psi(\mathbf{r},t)=\sum_{l=1}^{q}\eta_l^2(\mathbf{r},t). \tag{10}$$

The function is calculated during a simulation and only the values inside the bounding boxes are taken into account for the computation of Eq. (10). The function ψ takes the value 1 inside grains and considerably smaller values on grain boundaries.

V. VALIDATION OF THE BOUNDING BOX ALGORITHM

The bounding box algorithm has two important parameters: the applied threshold value ϵ and the initial mean grain size. When the initial microstructure is obtained through simulation, its mean grain size is determined by the time point t_s at which the conventional simulation is stopped (see Sec. IV). The parameters ϵ and the mean grain size influence the accuracy, computing memory, and computing time of the bounding box algorithm. In this section, implementation independent characteristics of the algorithm are discussed. Section VI treats the characteristics which are implementation dependent.

A. Preprocessing step

In the preprocessing step, the active regions of the microstructure are identified and isolated. As a result, the storage requirements of the bounding box data structure are significantly lower than those of the conventional grid-based data structure. Whereas the requirements of the conventional data structure equal the number of phase-field variables multiplied by the grid size, the requirements of the bounding box data structure are determined by the number of active phase-field variables per grid point and equal the number of phase-field values included in the sets B_l. This number depends on ϵ and the initial mean grain size and, as will be shown below,

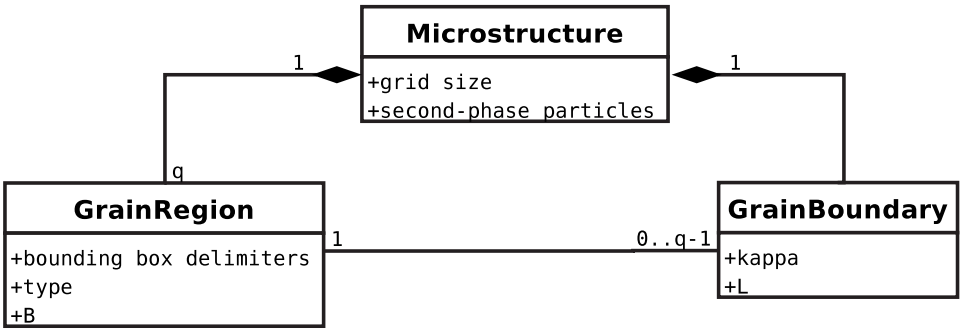


FIG. 4. UML diagram of the object-oriented data structure, extended to simulate anisotropic grain growth. The microstructure, grain regions, and grain boundaries are treated as objects with a number of attributes.

TABLE I. Number of phase-field values required per grid point for the bounding box algorithm and the conventional algorithm as a function of ϵ and the initial mean grain size.

Starting t_s	Mean grain size	Threshold value ϵ 10^{-3}	10^{-4}	10^{-5}	10^{-6}	Conventional
$t_s=200$	3.5×10^3	6.15	9.35	14.19	31.12	100
$t_s=400$	7.8×10^3	4.98	7.11	9.74	13.33	100
$t_s=600$	1.4×10^4	4.29	5.84	7.73	10.16	100
$t_s=800$	2.0×10^4	3.85	5.00	6.40	7.94	100

is typically a small fraction of what is required for the conventional implementation.

The preprocessing step was performed on the simulation results specified in Sec. IV at time points $t_s=200$, 400, 600, and 800. Different values of ϵ were applied: 10^{-3}, 10^{-4}, 10^{-5}, and 10^{-6}. Table I shows the required number of phase-field values per grid point for the bounding box algorithm and the conventional algorithm as a function of ϵ and t_s and thus the initial mean grain size. The fine-grained topology of the microstructure at the initial evolution stages results in smaller initial mean grain sizes and larger numbers of required phase-field values. It can be seen that the bounding box algorithm is far more efficient than the conventional algorithm.

B. Computation step

In the computation step of the bounding box algorithm, Eqs. (5) are only solved for the sets B_l. A large amount of computing time and computing memory is thus saved. During a simulation, the topology of a microstructure becomes more and more coarse grained. While the memory demand of the conventional algorithm is independent of the topology, the bounding box algorithm is more efficient for coarser grained topologies. To study the efficiency of the computation step, a $256\times256\times256$ microstructure was generated containing 67 grains with a mean grain size of 2.3×10^5 grid points and a different crystallographic orientation for every grain. Table II shows the time evolution of the number of phase-field values per grid point for the bounding box algorithm and the conventional algorithm for different threshold values ϵ. It can be seen that the memory efficiency of the bounding box algorithm increases with the simulation time. Furthermore, in the course of the simulation, the memory efficiencies for the different threshold values converge. This indicates that a higher threshold value ϵ only increases the computational requirements considerable at the beginning.

Next, we compare the simulation accuracy of the bounding box algorithm with that of the conventional algorithm. Since the microstructure, described in the previous paragraph, has a different crystallographic orientation for every grain, it allows for coalescence-free simulations with the conventional algorithm. The simulation accuracy is now studied by looking at the differences between the results of the conventional algorithm and those of the bounding box algorithm for different threshold values $\epsilon=10^{-3}$, 10^{-4}, 10^{-5}, and 10^{-6}, after 1, 1000, and 7000 time steps. In each case, the differences between the computed results for ψ [see Eq. (10)] are determined for every grid point and the mean grain size is obtained. After one time step, the point-wise differences between results for the threshold values 10^{-3}, 10^{-4}, 10^{-5}, and 10^{-6} are of order 10^{-5}, 10^{-6}, 10^{-6}, and 10^{-6}, respectively. Furthermore, the mean grain size shows no influence of the use of different threshold values. After 1000 time steps, the pointwise differences are only of order 10^{-3}. The mean grain size obtained by the bounding box algorithm deviates less than 10^{-7}, relatively, from the size obtained by the conventional algorithm. The differences between the results obtained for the different threshold values ϵ are even smaller. After 7000 time steps, the pointwise differences of all computed results are only of order 10^{-2} and located at the boundaries of grains which are slightly larger or smaller when compared. The mean grain sizes computed by the bounding box algorithm for the different threshold values differ by less than 10^{-7}, relatively. The relative differences between the mean grain sizes computed by the bounding box algorithm and the conventional algorithm are larger but still only of order 10^{-5}. This shows that the bounding box algorithm is highly accurate. Furthermore, the accuracy of the

TABLE II. Number of phase-field values per grid point for the bounding box algorithm and the conventional algorithm as a function of ϵ and the simulation time.

Simulation time t	Threshold value ϵ 10^{-3}	10^{-4}	10^{-5}	10^{-6}	Conventional
t_s	5.75	6.48	7.26	8.10	100
t_s+200	6.71	6.74	6.90	6.90	100
t_s+400	6.72	6.73	6.81	6.81	100
t_s+600	6.68	6.69	6.73	6.73	100
t_s+800	6.64	6.65	6.68	6.68	100

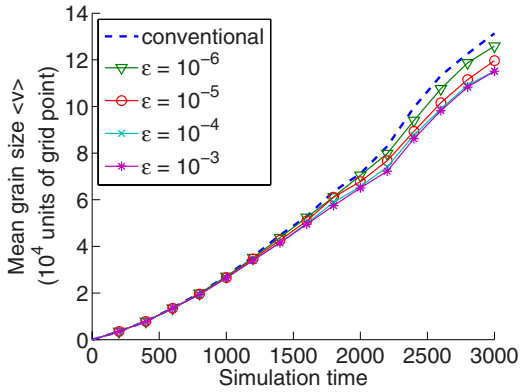


FIG. 5. (Color online) Effect of ϵ on the initial configuration: evolution of the mean grain size from $t_s=200$ to $t=3000$, computed by the conventional algorithm and by the bounding box algorithm for different threshold values ϵ.

bounding box computations is little influenced by the magnitude of the threshold value ϵ.

Finally, we study the effect of the threshold value ϵ and the time point t_s on the initial configuration. When the initial microstructure is obtained by simulation, the value of ϵ can influence the initial configuration. If not enough phase-field variables are involved in a conventional simulation, the small grains which nucleate and grow during the simulation can undergo significant coalescing. Consider a small positive threshold value ϵ. In the time steps before two neighboring grains with the same crystallographic orientation coalesce, the values of the involved phase-field variable start to increase along the contact boundary and eventually exceed ϵ in magnitude. At that point, the grains have become one according to the definition in Sec. III and will also be considered as one grain by the bounding box algorithm. As a result fewer different grains are taken into account and the initial mean grain size is larger. By using high values of ϵ, the neighboring grains will not be treated as one, but as separate grains. The use of too high threshold values will however disturb the accuracy of the simulation results. To study the influence of ϵ on the initial configuration and further, on the long-term kinetics, the bounding box is applied to the simulation results specified in Sec. IV. Figure 5 shows the evolution of the mean grain size from $t_s=200$ to $t=3000$, computed by the conventional algorithm and by the bounding box algorithm for several ϵ values. It can be seen that the mean grain size is higher for lower threshold values and highest for the conventional algorithm.

The time point t_s also has an influence on the initial configuration: when t_s is chosen earlier, some amount of the coalescence of the small grains which nucleate during the initial simulation is prevented. To study this effect, the bounding box algorithm was started at $t_s=200$, 400, 600, and 800 and run until $t=3000$. Figure 6 illustrates the computed evolution of the mean grain size from the different t_s to $t=3000$ for $\epsilon=10^{-4}$ and $\epsilon=10^{-5}$ and the conventional algorithm. As anticipated, the mean grain size is smaller when t_s is chosen earlier. For smaller values of ϵ, this effect is not clear: this can be explained by the effect discussed in the previous paragraph. Both the influence of ϵ and t_s on the initial configuration are a consequence of creating an initial microstructure with the conventional algorithm. Nevertheless, we feel that this conventional method, possibly executed on a coarser mesh, is a convenient way to obtain an initial large polycrystalline structure since in the case of ideal grain growth, the typical grain structure with stationary grain size distribution is recovered after a short transition time (see also [38]).

Based on the observations concerning the memory reduction, accuracy and initial configuration for the different threshold values, we advocate to use $\epsilon=10^{-5}$ or 10^{-6}.

The simulations further showed that at every grid point, approximately seven phase-field variables are active. Since the bounding box algorithm only takes into account the active phase-field values, this means that the computational requirements of the algorithm depend on the system size and not on the total number of phase-field variables.

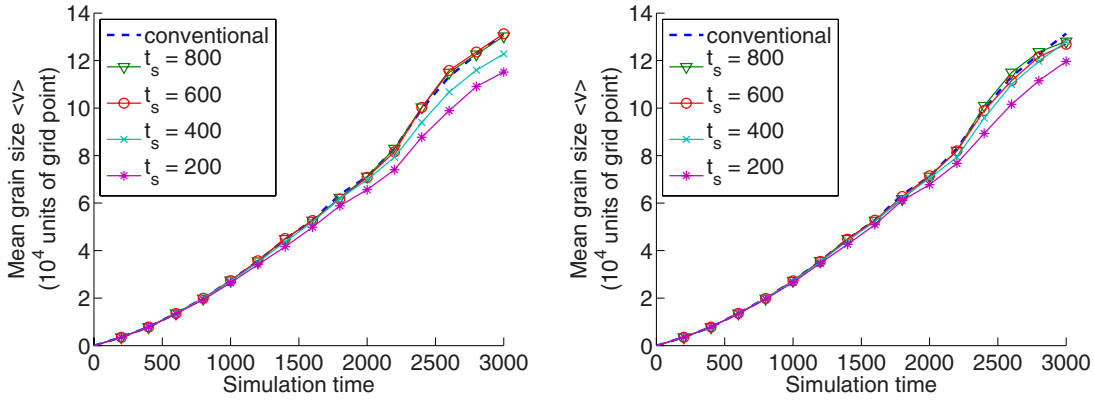


FIG. 6. (Color online) Effect of t_s on the initial configuration: evolution of the mean grain size from t_s to $t=3000$, computed by the conventional algorithm and by the bounding box algorithm for different starting time points t_s and threshold values (a) $\epsilon=10^{-4}$ and (b) $\epsilon=10^{-5}$.

TABLE III. Bounding box storage requirements (gigabyte) as a function of ϵ and the initial mean grain size (grid points), compared to conventional storage requirements.

		Threshold value ϵ				
Starting t_s	Mean grain size	10^{-3}	10^{-4}	10^{-5}	10^{-6}	Conventional
$t_s=200$	3.5×10^3	0.84	1.27	1.93	4.18	13.4
$t_s=400$	7.8×10^3	0.70	1.00	1.37	1.90	13.4
$t_s=600$	1.4×10^4	0.63	0.86	1.14	1.50	13.4
$t_s=800$	2.0×10^4	0.59	0.77	0.98	1.24	13.4

VI. COMPUTATIONAL REQUIREMENTS

In this section, implementation-dependent characteristics of the bounding box algorithm are discussed.

A. Preprocessing step

From the preprocessing step of the bounding box algorithm on, all phase-field values not exceeding ϵ in absolute value are assumed to be zero. They are not included in the sets B_l and therefore excluded from further computations. As a consequence, these values do not have to be stored and the data resulting from a bounding box simulation require less storage space than the data from a conventional simulation. Table III displays the bounding box storage requirements for $t_s=200$, 400, 600, and 800, and thus different initial mean grain sizes, and $\epsilon=10^{-3}$, 10^{-4}, 10^{-5}, and 10^{-6} and the conventional storage requirements. For smaller mean grain size, the storage requirements are higher. Furthermore, the threshold value ϵ has a larger influence on the storage amount when the mean grain size is smaller.

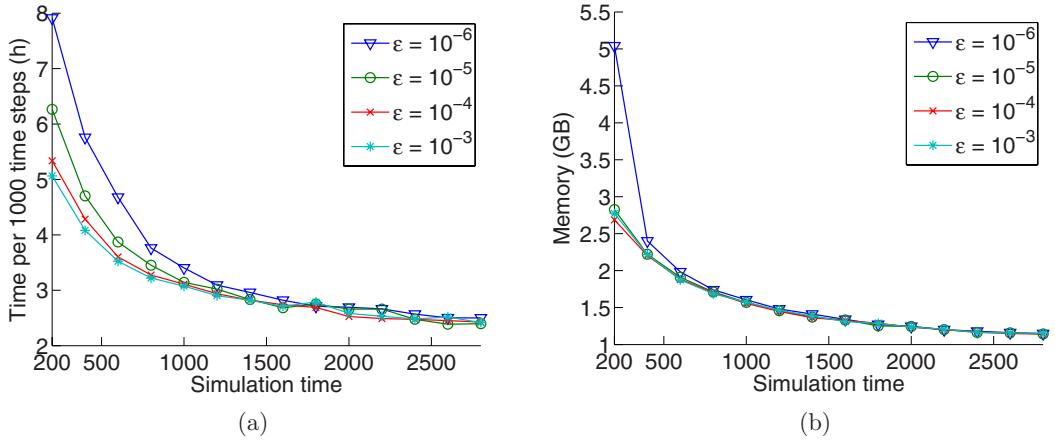
B. Computation step

If the initial microstructure is obtained by simulation, the total amount of computing resources spent on a grain growth simulation depends on the starting time point t_s. When t_s is chosen relatively small, little effort is spent on a time and memory consuming conventional simulation. The computational requirements are also influenced by the threshold value ϵ. Figure 7 shows the evolution of the computing time and computing memory per 1000 time steps of the bounding box algorithm for $t_s=200$ and different threshold values ϵ. At the start of the bounding box algorithm, considerably more computing time and computing memory is required, because of the smaller mean grain size. Furthermore, for lower threshold values ϵ, the computational requirements are higher. Later on in the simulations, the requirements are approximately the same for the different threshold values.

The conventional computational requirements are constant in time, whereas the bounding box algorithm requires less resources as simulation time progresses. In Table IV, the computational requirements for the first 5000 time steps of a conventional simulation and a bounding box simulation are shown for $\epsilon=10^{-5}$ and $t_s=200$. The conventional algorithm was run on 20 processors (see Sec. IV), in contrast to the bounding box algorithm which could be executed on a single processor.

VII. APPLICATION

To illustrate the applicability and efficiency of the bounding box algorithm, we applied the algorithm to grain growth in the presence of second-phase particles. The addition of alloying elements, which leads to the formation of finely dispersed second-phase particles, is a common technique to control the grain size of a microstructure. The particles pin



(a) (b)

FIG. 7. (Color online) Evolution of (a) computing time and (b) computing memory per 1000 time steps ($\Delta t=0.2$) of the bounding box algorithm for $t_s=200$ and different threshold values ϵ.

TABLE IV. Computational requirements of the first 5000 time steps ($\Delta t=0.2$) of the bounding box algorithm with $\epsilon=10^{-5}$ as a function of the simulation time and the mean grain size (grid points), compared to the conventional algorithm.

		Bounding box (on 1 processor)		Conventional (on 20 processors)	
Simulation time	Mean grain size	Time (h)	Memory (gigabyte)	Time (h)	Memory (gigabyte)
$200 \rightarrow 400$	$3.52\times10^3 \rightarrow 7.82\times10^3$	6.3	2.8	6.8	30.3
$400 \rightarrow 600$	$7.82\times10^3 \rightarrow 1.35\times10^4$	4.7	2.2	6.8	30.3
$600 \rightarrow 800$	$1.35\times10^4 \rightarrow 1.95\times10^4$	3.8	1.9	6.8	30.3
$800 \rightarrow 1000$	$1.95\times10^4 \rightarrow 2.66\times10^4$	3.5	1.7	6.8	30.3
$1000 \rightarrow 1200$	$2.66\times10^4 \rightarrow 3.46\times10^4$	3.1	1.6	6.8	30.3

the grain boundaries and when a limiting grain size is reached, grain growth stops. The limiting grain size depends on the number, size, shape, and spatial distribution of the particles. In [22,23], a phase-field model is presented for simulating grain growth in materials containing small incoherent second-phase particles which are constant in time. The interaction between a single particle and a grain boundary is investigated and the results of two-dimensional simulations of the pinning effect of the particles on grain growth are discussed. Unfortunately, the computational requirements of three-dimensional simulations with the conventional algorithm are memory consuming and computationally intensive. With the bounding box algorithm, three-dimensional simulation results can be obtained for relevant comparison with experimental data without excessive computational requirements.

Simulations were run on a $256\times256\times256$ grid with 100 phase-field variables and volume fractions of $f_V=5\%$, 8%, and 12% of second-phase particles with a radius r of 3 grid points. The parameter values were set to $\kappa=0.5$ and $L=1$. The discretization spacings were $\Delta x=1$ and $\Delta t=0.2$. First, the conventional algorithm was executed on 20 processors and applied until $t_s=1800$. Second, the bounding box algorithm was applied and run on only one processor until no further growth was observed, for $\epsilon=10^{-5}$. The evolution of the microstructure is illustrated for $f_V=8\%$ in Fig. 8. Figure 9 shows the evolution of the mean grain radius for the different volume fractions f_V. For larger values of f_V, the grain growth stops earlier and the final grain size is smaller.

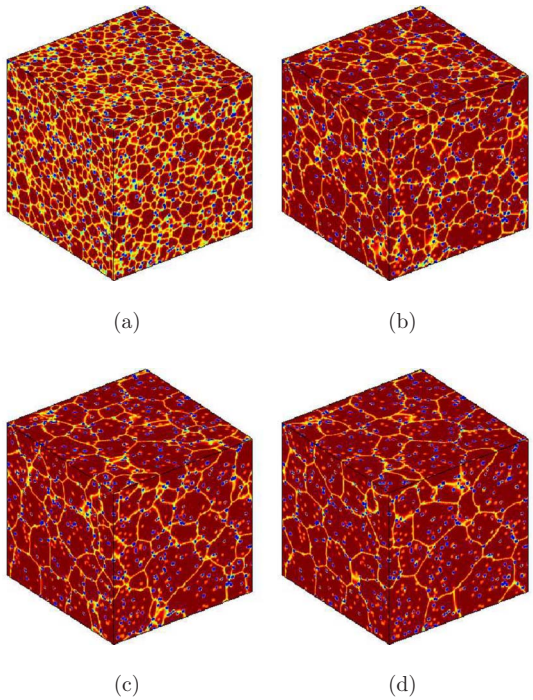


FIG. 8. (Color online) Evolution of a grain structure containing second-phase particles ($f_V=8\%$, $r=3$), obtained from a phase-field simulation on a $256\times256\times256$ grid. Images are shown at (a) $t=200$, (b) $t=2000$, (c) $t=5000$, and (d) $t=37\,200$.

In Fig. 10, the evolution of the normalized grain size distribution is shown for $f_V=8\%$, with approximately 100 grains at the end of the simulation. The normalized grain size distribution for a single-phase system is shown on one time point, since for single-phase systems, the distribution is constant in time [1]. As predicted by mean-field theories [3,39,40], the simulation results indicate that, when second-phase particles are present, the peak of the distribution shifts towards smaller grain sizes.

To validate the computed results, we compare the obtained limiting grain sizes to existing theories and observations. Figure 11 shows the ratio of the limiting mean grain size to the particle radius as a function of the volume fraction as computed by our phase-field simulations, the classical Zener relation, the relation obtained through three-dimensional Monte Carlo simulation in [41], and the relation obtained by phase-field simulation in [24]. The relation from [42], obtained by fitting the parameters in the Zener relation with

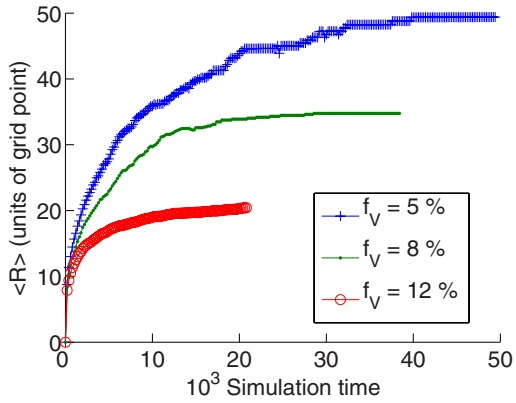


FIG. 9. (Color online) Evolution of the mean grain radius for volume fractions of $f_V=5\%$, 8%, and 12% obtained from phase-field simulations.

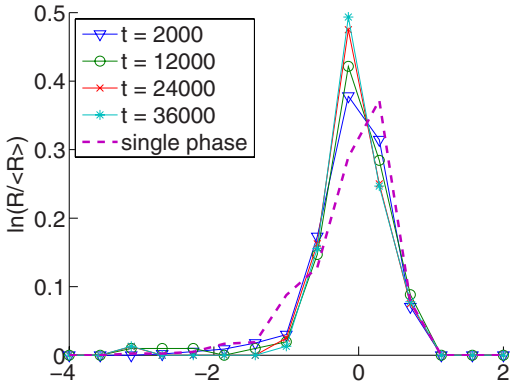


FIG. 10. (Color online) Evolution of the normalized grain size distribution for f_V=8%, compared to the normalized grain size distribution for a single-phase system, obtained from phase-field simulations.

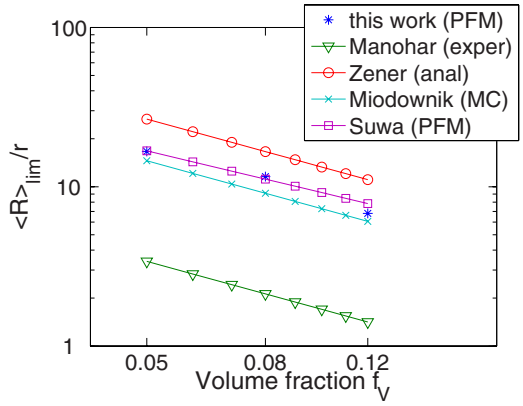


FIG. 11. (Color online) Ratio of the radius of the limiting mean grain size to the particle radius as a function of the volume fraction f_V, compared for different theories.

experimental grain growth data for a variety of material systems, obtained under different experimental conditions, is also added. The data obtained with our simulations correspond well to the Monte Carlo results and the results of [24], but deviate strongly from the experimental data. This indicates that the Monte Carlo model, the phase-field model, and the Zener relation are suitable for round-shaped particles and isotropic grain boundary energies. However, to simulate real materials, which contain plate- or lens-shaped particles or anisotropic boundary energies, the phase-field model has to be extended. The difference between the results from the Monte Carlo simulations and those of the phase-field simulations is probably due to the fact that the particles had a slightly different shape in both simulation experiments, as after discretization the small particles are no longer completely spherical. Due to the diffuse interfaces, the discretization effect is smaller in the phase-field simulations.

In [24], the results were obtained using a dynamic grain-orientation reassignment algorithm. However, the computational requirements of the latter algorithm (approximately 48 gigabyte, on 16 processors) are more than ten times larger than the requirements of the bounding box algorithm (2 gigabyte, on 1 processor).

VIII. CONCLUSION

In this paper, a sparse bounding box algorithm is presented to perform efficient phase-field simulations of microstructural evolution in polycrystalline materials. The algorithm only solves the phase-field equations locally, inside bounding boxes which delimit regions of active phase-field variables. In contrast to other sparse algorithms, the bounding box data structure allows for semi-implicit time integration. Furthermore, because of the object-oriented design, the bounding box algorithm is extendible to more complex models and has advantages in postprocessing.

The computational requirements of the bounding box algorithm depend on the system size and not on the total number of involved phase-field variables. In combination with the one-to-one mapping between grains and phase-field variables, this allows to perform coalescence-free simulations of grain growth without the excessive memory usage or computing time associated with existing methods.

To illustrate the applicability of the bounding box algorithm, three-dimensional phase-field simulations of grain growth in the presence of second-phase particles were performed to study the effect of the particles on the growth kinetics. The simulation results correspond well to other simulation results, but deviate from experimental data. To perform simulations of realistic materials, the phase-field model has to be extended. We believe that the bounding box algorithm will enable such simulations and provide better insight in microstructural evolution.

ACKNOWLEDGMENTS

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. N.M. was supported by the Research Foundation - Flanders (FWO). The simulations were performed on the computer cluster of the HPC Computing Initiative, funded by the Katholieke Universiteit Leuven Research Council.

[1] F. J. Humphreys and M. Hatherly, *Recrystallization and Related Annealing Phenomena* (Pergamon, Oxford, 1996).
[2] M. A. Miodownik, Scr. Mater. **54**, 993 (2006).
[3] M. Hillert, Acta Metall. **13**, 227 (1965).
[4] M. Guo and H. Suito, ISIJ Int. **39**, 1297 (1999).
[5] J. Gruber, D. George, A. Kuprat, G. Rohrer, and A. Rollett, in *Recrystallization and Grain Growth*, edited by B. Bacroix, J. Driver, R. L. Gall, C. Maurice, R. Penelle, H. Réglé, and L.

Tabourot, Materials Science Forum Vols. 467–470 (Trans. Tech, Stafa-Zurich, 2004), pp. 733–738.
[6] D. Danilov and B. Nestler, Acta Mater. **54**, 4659 (2006).
[7] M. Plapp, J. Cryst. Growth **303**, 49 (2007).
[8] B. Böttger, J. Eiken, and I. Steinbach, Acta Mater. **54**, 2697 (2006).
[9] Q. Bronchard, Y. L. Bouar, and A. Finel, Adv. Eng. Mater. **8**, 1245 (2006).
[10] Y. Wen, B. Wang, J. Simmons, and Y. Wang, Acta Mater. **54**, 2087 (2006).
[11] D. Fan and L.-Q. Chen, Acta Mater. **45**, 611 (1997).
[12] C. E. Krill III and L.-Q. Chen, Acta Mater. **50**, 3057 (2002).
[13] N. Ma, A. Kazaryan, S. A. Dregia, and Y. Wang, Acta Mater. **52**, 3869 (2004).
[14] L.-Q. Chen and W. Yang, Phys. Rev. B **50**, 15752 (1994).
[15] R. Kobayashi, J. A. Warren, and W. C. Carter, Physica D **119**, 415 (1998).
[16] I. Steinbach, F. Pezzolla, B. Nestler, M. Seeßelberg, R. Prieler, G. J. Schmitz, and J. L. L. Rezende, Physica D **94**, 135 (1996).
[17] D. Fan and L.-Q. Chen, Acta Mater. **45**, 1115 (1997).
[18] L.-Q. Chen and D. Fan, J. Am. Ceram. Soc. **79**, 1163 (1996).
[19] D. N. Fan and L.-Q. Chen, Acta Mater. **45**, 3297 (1997).
[20] D. Fan, S. P. Chen, L.-Q. Chen, and P. W. Voorhees, Acta Mater. **50**, 1895 (2002).
[21] D. Fan, L.-Q. Chen, and S. P. Chen, J. Am. Ceram. Soc. **81**, 526 (1998).
[22] N. Moelans, B. Blanpain, and P. Wollants, Acta Mater. **53**, 1771 (2005).
[23] N. Moelans, B. Blanpain, and P. Wollants, Acta Mater. **54**, 1175 (2006).
[24] Y. Suwa, Y. Saito, and H. Onodera, Scr. Mater. **55**, 407 (2006).
[25] N. Provatas, N. Goldenfeld, and J. Dantzig, Phys. Rev. Lett. **80**, 3308 (1998).
[26] N. Provatas, N. Goldenfeld, and J. Dantzig, J. Comput. Phys. **148**, 265 (1999).
[27] R. J. Braun and B. T. Murray, J. Cryst. Growth **174**, 41 (1997).
[28] G. Beckett, J. A. Mackenzie, and M. L. Robertson, Comm. Comp. Phys. **1**, 805 (2006).
[29] W. M. Feng, P. Yu, Z. K. Liu, Q. Du, and L.-Q. Chen, J. Comput. Phys. **220**, 498 (2006).
[30] J. Gruber, N. Ma, Y. Wang, A. D. Rollett, and G. S. Rohrer, Modell. Simul. Mater. Sci. Eng. **14**, 1189 (2006).
[31] S. Vedantam and B. S. V. Patnaik, Phys. Rev. E **73**, 016703 (2006).
[32] S. G. Kim, D. I. Kim, W. T. Kim, and Y. B. Park, Phys. Rev. E **74**, 061605 (2006).
[33] I. Steinbach and F. Pezzolla, Physica D **134**, 385 (1999).
[34] S. G. Kim, W. T. Kim, T. Suzuki, and M. Ode, J. Cryst. Growth **261**, 135 (2004).
[35] L.-Q. Chen and J. Shen, Comput. Phys. Commun. **108**, 148 (1998).
[36] N. Moelans, Ph.D. thesis, Katholieke Universiteit Leuven, 2006.
[37] M. I. M. Copetti and C. M. Elliot, Mater. Sci. Technol. **6**, 273 (1990).
[38] D. Zöllner and P. Streitenberger, Scr. Mater. **54**, 1697 (2006).
[39] O. Hunderi and N. Ryum, Acta Metall. **30**, 739 (1982).
[40] G. Abbruzzese, Acta Metall. **33**, 1329 (1985).
[41] M. Miodownik, E. A. Holm, and G. N. Hassold, Scr. Mater. **42**, 1173 (2000).
[42] P. A. Manohar, M. Ferry, and T. Chandra, ISIJ Int. **38**, 913 (1998).