INSTITUTE OF PHYSICS PUBLISHING MODELLING AND SIMULATION IN MATERIALS SCIENCE AND ENGINEERING

Modelling Simul. Mater. Sci. Eng. **14** (2006) 83–98 doi:10.1088/0965-0393/14/1/007

A numerical method for curvature driven boundary migration

M A Martorano[1], M A Fortes[2] and A F Padilha[1]

[1] Departamento de Engenharia Metalúrgica e de Materiais, Universidade de São Paulo, Av. Prof. Mello Moraes, 2463 São Paulo-SP, 05508-900, Brazil
[2] Departamento de Engenharia de Materiais, Instituto Superior Técnico, Av. Rovisco Pais, 1049-001 Lisboa, Portugal

E-mail: martoran@usp.br, padilha@usp.br and l.marcelino@ist.utl.pt

Received 9 May 2005, in final form 12 October 2005
Published 12 January 2006
Online at stacks.iop.org/MSMSE/14/83

Abstract

This paper introduces a numerical method for curvature calculation, termed least square-normal and curvature (LSNC), for simulating the migration of two-dimensional boundaries. The method is based on the existing volume-of-fluid (VOF) method with new algorithms to calculate the normal and the curvature of a boundary. The LSNC method is applied to solve a number of simple problems pertaining to boundary migration driven by curvature and a uniform 'force' associated with an energy density difference across the boundary. The accuracy of the LSNC method is assessed by comparing its results with exact solutions, when available. The method proved more accurate than the Ripple method, which is described in the literature as another variant of VOF methods.

1. Introduction

The motion of interfaces and boundaries plays a crucial role in phase transformations. Solidification of pure metals and alloys, growth of precipitates, recrystallization of work-hardened metals and grain growth (which can be regarded as a phase transformation) are only a few examples of important phenomena that depend on the motion of interfaces [1, 2]. These interfaces have associated interface energy and can be called boundaries in mathematical terms. The mathematical modelling of their motion is a complex problem referred to as a moving boundary problem [3]. In the context of grain growth, several numerical methods have been proposed to solve moving boundary problems. These methods can be defined as either deterministic or stochastic (probabilistic), such as the Monte Carlo (Potts model) or cellular automaton models [4–6]. In stochastic models, the movement of boundaries is predicted using numerical cells and a set of rules to determine the cell state, without explicitly accounting for the boundary curvature. Deterministic models, such as the finite element, vertex or phase field method, are usually based on deterministic kinetic equations to predict the movement of

each portion of the boundary. Finite element methods are rather complex and predict the time evolution of grains using a variational [7] or a weak formulation [8] to minimize the total energy associated with the grain boundaries. The vertex method [9] simplifies the description of a set of grains by assuming that boundaries (in 2D) are straight lines, neglecting the curvature effect. The time evolution of the system is predicted by a set of equations describing the displacement of the grain vertices. In phase field models [10, 11] the boundary is assumed to be diffuse and its position is determined by a transition in the value of the phase field variable across the boundary region. The numerical grid in these methods is usually very refined when compared with those in other methods, because a sufficiently large number of numerical cells must be used to guarantee a smooth transition of the phase field variable across the boundary.

A different set of methods that is hardly applied to phase transformation problems is frequently used for moving boundaries in fluid dynamics. Some of these methods can be considered as front capturing methods [12] in which the boundary position is not tracked explicitly, but recovered from some field variable. Alternatively, the boundary can be tracked explicitly by a so-called front tracking method. Both methods can be designated as front advancing methods and are reviewed in sections 1.1–1.3. A novel method (the least-square normal and curvature or LSNC method) for simulation of boundary migration is presented in section 1.4 and described in detail in section 2. The final sections 3 and 4 discuss various applications of the LSNC method.

1.1. Front advancing methods

Front advancing methods are a family of algorithms specially constructed to handle a boundary in moving boundary problems. Specific front advancing methods were developed to solve fluid dynamics problems and some of these were defined as marker methods or marker-and-cell (MAC), which are typical front tracking methods [13]. In the family of front capturing methods, the most popular are the volume-of-fluid (VOF) methods [14]. These methods use the phase volume fraction (area fraction, in two-dimensional models) within the computational cells in order to reconstruct the boundary. It has successfully been employed to predict the shape and movement of boundaries in various practical problems [15–18]. These methods are robust and relatively easy to implement, which are often quoted as the main reasons for their popularity. Nevertheless, VOF methods have had limited applications to phase transformation problems. Jacot and Rappaz adapted a VOF method to simulate the ferrite-to-austenite transformation [19] and to track the movement of columnar grains during solidification [20] without considering the effect of boundary curvature and energy. Jacot and Rappaz [21], on the other hand, considered the effects of boundary curvature and energy to simulate the two-dimensional growth of equiaxed dendrites using a VOF method.

1.2. Boundary position determination in VOF methods

In a moving boundary problem, the boundary usually separates two parts of a continuum, either of which is designated as the 'VOF phase'. VOF methods include a front capturing algorithm that reconstructs the boundary from the fraction of the VOF phase within the computational cells used to subdivide the problem domain. A function F (or 'field') that equals the VOF phase fraction in each computational cell is introduced, such that

(1) $F = 1$ for cells completely embedded in the VOF phase;
(2) $0 < F < 1$ for cells cut by the boundary (boundary cells);
(3) $F = 0$ for cells completely outside the VOF phase.

The exact shape and position of the boundary can be determined from the F field by a boundary reconstruction algorithm, which is at the core of VOF methods.

The VOF methods basically consist of (1) procedures for boundary reconstruction from the F field and (2) procedures to advance the F field [22, 23]. In most VOF methods, the boundary is approximated by a line segment within each boundary cell. The segment is determined unambiguously from F within the cell and its normal vector $\boldsymbol{n}$. Consequently, the reconstruction algorithm is reduced to determining $\boldsymbol{n}$ from the known F field. In some special algorithms, such as Ripple [24] and Surfer [12], $\boldsymbol{n}$ is calculated as

$$\boldsymbol{n} = \nabla F. \tag{1}$$

The gradient operator is approximated by an equation of finite differences using the F values in the cells. Calculations based on equation (1) were shown to have only first order accuracy, e.g. a planar boundary is not exactly reconstructed from its F field.

In a different approach, called the least-square (LS) method [25], the $\boldsymbol{n}$ adopted for a specific cell is the one that minimizes the squared error between the actual known F values in a stencil surrounding the cell and those calculated using the adopted $\boldsymbol{n}$. In two dimensions, if the cell is indexed as (i, j), the error $E(\boldsymbol{n})$ is calculated in a 3×3 cell stencil from

$$E(\boldsymbol{n}) = \left(\sum_{k=-1}^{1} \sum_{l=-1}^{1} [F'_{i+k,j+l}(\boldsymbol{n}) - F_{i+k,j+l}]^2 \right)^{1/2}, \tag{2}$$

where F' is calculated in each of the nine cells of the 3×3 stencil by extending the line segment from the central cell into the neighbour cells. F' in the central cell (i, j) is constrained to exactly match the known F. This method was shown to have second order accuracy in the sense that a planar boundary is reconstructed exactly [22].

1.3. Curvature determination in VOF methods

The excess energy associated with a boundary is frequently important in physical problems and can affect its motion through the boundary curvature. To take this effect into account in VOF methods, some special procedures were developed to calculate the boundary local curvature. In one of the first procedures, proposed by Chorin [26], the osculation circle whose intersection with each of the cells in a 3×3 stencil gives the same F was determined. The curvature is the reciprocal of the osculating circle radius. In the Ripple algorithm [24], the local curvature κ is calculated as the divergence of the unit normal with the help of equation (1). After some manipulation, the final equation is

$$\kappa = -\nabla \cdot \left(\frac{\boldsymbol{n}}{|\boldsymbol{n}|} \right) = \frac{1}{|\boldsymbol{n}|} \left(\frac{\boldsymbol{n}}{|\boldsymbol{n}|} \cdot \nabla |\boldsymbol{n}| - \nabla \cdot \boldsymbol{n} \right). \tag{3}$$

Since the normal $\boldsymbol{n}$ depends on the first order derivatives of F (equation (1)), κ depends on its second order derivatives. Finite difference approximations of an abrupt changing function, such as F, are usually very inaccurate [27], becoming worse as the order of the derivative increases. As a result, oscillations in the calculated local curvature along a circular boundary reached values larger than the local curvature itself [15]. Smoothing the F field by interpolation functions was shown to reduce oscillations [24], but to artificially decrease the curvature in some critical regions, resulting in unphysical behaviour [28].

Meyer *et al* [15] calculated the curvature from three parameters extracted from the F field with the help of a polynomial relation between these parameters and the curvature. The authors observed that the calculated average curvature was much more accurate than that calculated by the Ripple method.

Poo and Ashgriz [29] approximated the boundary within a 3×3 cell stencil by a second order polynomial. The unknown coefficients were found by requiring that the sum of F for all three columns in the stencil equals the equivalent volume fractions cut through by the polynomial curve. Very accurate curvature results were obtained for a sine wave and for a circle.

More recently, Renardy and Renardy [27] extended Poo and Ashgriz's [29] method by locally approximating the boundary by a quadratic surface equation from which the local curvature was calculated. Although the calculated curvature values were not shown, a drastic reduction was observed in some artificial effects usually caused by inaccurate curvature calculation.

To avoid the inaccuracies of curvature calculations by the VOF methods, Sussman and Puckett [30] suggested the use of a level-set method. This method, which was called 'coupled level-set VOF' (CLSVOF), gives excellent results for curvature, but a further field variable, i.e. the level-set variable, has to be stored and updated in the algorithm. Jacot and Rappaz [21] introduced a variant of this method to study the solidification of an equiaxed dendrite in two dimensions, with excellent results. Except for this work, there are no applications of the VOF method to phase transformation problems that are affected by boundary curvature.

1.4. The least-square normal and curvature method

The main objective of the present work is to propose a two-dimensional VOF method for phase transformation problems with boundaries whose motion depends on curvature. In this method, which we term the LSNC method, a boundary is reconstructed from the volume fraction function, F, and its local normal vector and curvature are determined by two minimization procedures based on the LS method. This approach, which is essentially a geometric one, differs from that in Ripple, where differential operators approximated by finite differences are applied to the F function to obtain the normal and the curvature. Therefore, the utilization of an LS method to calculate both the boundary normal and curvature is the new feature in the LSNC method in relation to existing VOF methods.

The proposed LSNC method is applied to a number of problems of moving boundaries that are independent of long-range diffusion of heat or solute, such as recrystallization and grain growth. The LSNC solutions to these problems were compared with those obtained with the Ripple method and with available analytical solutions. The Ripple method, also referred to as the continuous surface force (CSF) method, was chosen in the present work for comparison because either all or some of its procedures have been used for comparison purposes in the literature [15, 22, 23, 27, 31, 32]. Furthermore, Ripple is easily adjusted for the type of moving boundary problems studied in this work, where the curvature is calculated explicitly, rather than implicitly as in the Surfer method [12].

2. Description of the LSNC method

The two-dimensional LSNC method proposed in the present work includes three main algorithms for specific tasks: (1) calculation of boundary normal vector and boundary reconstruction, (2) calculation of boundary local curvature, and (3) boundary migration. In order to prove the merits of this method, the boundary reconstruction and curvature algorithms from the Ripple method were also implemented for comparison with the LSNC method. In the following sub-sections, the LSNC algorithm is presented first, followed by the implementation of the Ripple algorithm.

2.1. Boundary reconstruction

The two-dimensional domain where the moving boundary is analysed is first subdivided into a mesh of rectangular or square cells. The boundary reconstruction algorithm creates an approximation to the exact boundary line from the related F field, defined as the area fraction occupied by a predefined 'VOF phase' in each mesh cell. As in the Ripple algorithm [24], each portion of the boundary within one cell is represented by a line segment. This segment is determined based on two criteria: (1) it divides the cell into two regions, either of which has the same area fraction as the VOF phase fraction, i.e. the F value for that cell and (2) the segment is orthogonal to an approximation ($\boldsymbol{n}$) to the boundary normal.

The equation for the line segment within the cell can be written as [22]

$$\boldsymbol{n} \cdot \boldsymbol{x} - c = 0, \tag{4}$$

where $\boldsymbol{n}$ is the line segment normal, $\boldsymbol{x}$ is the position vector of a point in the segment and c is the segment constant, which is calculated from F once $\boldsymbol{n}$ is known.

The local normal vector $\boldsymbol{n}$ was calculated from the F field by the LS method [22,25]. The boundary normal for the central cell in a 3×3 stencil was taken as the one that minimizes the squared error between F and the area fractions $F'(\boldsymbol{n})$ added for all cells in the stencil, as given in equation (2). $F'(\boldsymbol{n})$ for one cell is defined as the area fraction cut by a line segment constructed in the central cell and extended to its neighbouring cells in the stencil. As in other VOF methods, the line segment equation is determined by $\boldsymbol{n}$, which is unknown at first, and by imposing $F'_{i,j}(\boldsymbol{n}) = F_{i,j}$.

A first estimate for $\boldsymbol{n}$ is obtained from equation (1), using a finite difference approximation to ∇F. For an inner cell, i.e. not adjacent to a domain boundary, the approximation was written as

$$n_x(i, j) = \frac{1}{4}\left[\frac{F_{i+1,j+1} - F_{i-1,j+1}}{2\Delta x} + 2\frac{F_{i+1,j} - F_{i-1,j}}{2\Delta x} + \frac{F_{i+1,j-1} - F_{i-1,j-1}}{2\Delta x}\right], \tag{5}$$

$$n_y(i, j) = \frac{1}{4}\left[\frac{F_{i-1,j+1} - F_{i-1,j-1}}{2\Delta y} + 2\frac{F_{i,j+1} - F_{i,j-1}}{2\Delta y} + \frac{F_{i+1,j+1} - F_{i+1,j-1}}{2\Delta y}\right], \tag{6}$$

where n_x and n_y (x and y axis along the cell edges) are the components of the normal vector approximation in cell (i, j) and the subscripts of F are the indexes of the cells in a 3×3 stencil, with cell (i, j) at its centre, as shown in figure 1. Equations (5) and (6) are weighted averages of three central differences approximations, which are equal to the average of four normal vectors calculated at the cell corners, as proposed in the original Ripple [24]. For cells adjacent to the domain boundaries, no details were given in the Ripple algorithm description [24]. Consequently, approximations using the same weighted average of either forward or backward finite differences were adopted in the present work.

The estimates given by equations (5) and (6) are improved by changing the normal orientation (defined by the angle θ with the x-axis, see figure 2) and searching for the one that minimizes the error calculated from equation (2). In the computational code, this one-parameter optimization problem is solved by the Golden Section Search algorithm [33]. Notice that the LS method is actually used to improve on the Ripple calculation of $\boldsymbol{n}$.

2.2. Curvature calculation

The curvature was calculated for each boundary cell using a 3×3 cell stencil (figure 1). The boundary was first reconstructed within each cell of the stencil using line segments, as described above. The mid-points of all line segments in the stencil were found and a local reference system (x', y') was fixed at the mid-point of the central cell segment with y' parallel to the segment

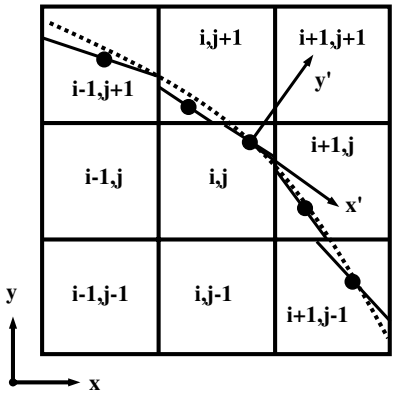

Figure 1. Stencil of 3 × 3 cells with line segments, mid-points and the quadratic curve (· · · · · ·) adjusted to the mid-points for curvature calculation.

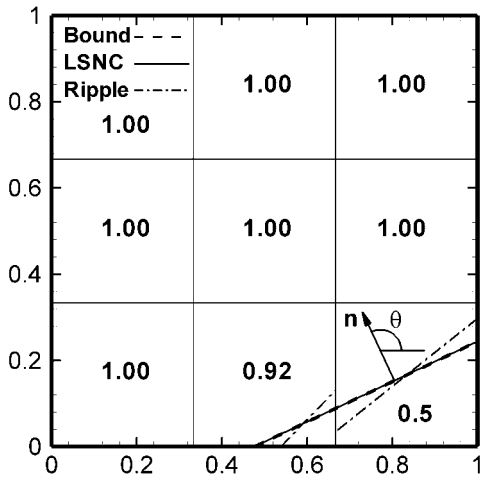

Figure 2. Straight boundary (- - - -) with normal $\boldsymbol{n}$ of orientation $\theta = 115^\circ$ and its associated F field used for reconstructions by the Ripple and LSNC methods. The LSNC reconstruction is coincident with the straight boundary line.

normal. Using this reference system, a quadratic curve of equation $y' = a \cdot (x')^2 + b \cdot (x')$ was approximately fit to all mid-points inside the stencil. Notice that this curve is constrained to go through the mid-point of the central cell. The unknown coefficients, a and b were found by minimizing the squared error between y' and the y coordinate of the mid-points. The curvature vector $\boldsymbol{\kappa} = \kappa \boldsymbol{n}$ for the centre cell is finally given by

$$\boldsymbol{\kappa} = \left(\frac{-b\boldsymbol{e}_{x'} + 2a\boldsymbol{e}_{y'}}{(1+b^2)}\right), \tag{7}$$

where $\boldsymbol{e}_{x'}$ and $\boldsymbol{e}_{y'}$ are unity vectors aligned with the x' and y' axis, respectively.

2.3. Boundary migration

In the VOF method, the boundary advances when the F field is updated. In problems of fluid dynamics, several schemes have been proposed to update F without excessively distorting the

boundary shape or creating flotsam [22,23,31]. Flotsam is a collection of boundary fragments artificially created by the numerical method, without actual physical meaning. In all these schemes, the time evolution of F is calculated from the fluid velocity at the cell edges. In phase transformation problems, however, the phases do not actually move; only the boundary between them does. Therefore, a different scheme is required.

In the present work, a scheme similar to that proposed by Jacot *et al* [20] was used to update F and predict the boundary motion. After reconstructing the boundary, the line segment within a boundary cell is displaced along its normal direction by a distance $(V_B \Delta t)$, where V_B is the boundary velocity in this direction and Δt is the time step of the numerical method. The calculation of V_B is discussed below. During displacement in the time interval $(t, t + \Delta t)$, the segment normal $\boldsymbol{n}$ is fixed, and its equation (equation (4)) is updated by calculating the segment constant c as follows:

$$c^{t+\Delta t} = c^t - V_B^{t+\Delta t} \Delta t. \tag{8}$$

After displacing the line segments within all boundary cells, the area fraction of the VOF phase, i.e. the F field, was updated and became available at time $t + \Delta t$. The numerical toolboxes suggested by Rider and Kothe [22] were used for the update.

After the segment displacement, its intersections with the cell edges are examined. When the segment intersects an edge shared with an empty ($F = 0$) or a full ($F = 1$) neighbour cell, an F fraction between 0 and 1 is attributed to this cell, i.e. the neighbour cell becomes a boundary cell, and the boundary migrates. The F value attributed to the new boundary cell is simply the area fraction cut by the extension of the line segment that intersects its edge.

To obtain the final F field, some adjustments, described as 'bookkeeping adjustments' in the original VOF method [14], are required. In the present numerical method, the F value of a cell was adjusted after updating. When $F = 1$ in the cell, its neighbours are examined and, if there is any empty cell in the 3×3 stencil around it, F is set to $(1 - \varepsilon)$, where $\varepsilon = 10^{-6}$. On the other hand, when $F = 0$ and there is at least one full cell around it, F is set to ε. These adjustments guarantee that a boundary cell always lies between a full and an empty cell. Without these adjustments, it is impossible to track the boundary between a full and an adjacent empty cell, because F for a boundary cell must be in the range $0 < F < 1$.

2.4. Boundary velocity

In section 4 the LSNC method will be applied to problems of boundaries migrating under the effect of a boundary energy and an extra driving force, as for example in the recrystallization of work-hardened metals. This type of problem is traditionally modelled using the boundary mobility coefficient M [34,35], as given below:

$$V_B = M(F_D + F_\sigma), \tag{9}$$

where F_D and F_σ are driving pressures, traditionally referred to as driving forces, associated with an energy density difference across the boundary and with curvature (i.e. boundary energy), respectively. For example, in recrystallization F_D is the excess strain energy density of the grains surrounding the recrystallized grain. Both F_D and F_σ act along the normal to the boundary. The value of F_D equals the jump in energy density across the boundary, whereas that of F_σ relates to the local curvature κ by [36,37]:

$$F_\sigma = (\sigma + \sigma'')\kappa, \tag{10}$$

where $\kappa = \boldsymbol{\kappa} \cdot \boldsymbol{n}$, σ is the boundary energy and σ'' is the second derivative of the boundary energy in relation to the boundary orientation given by $\boldsymbol{n}$. This term only appears when the boundary energy is anisotropic.

The signs of the driving forces depend on the choice of boundary normal vector. For instance, when the adopted normal $\boldsymbol{n}$ is directed out of the centre of curvature of the boundary, then $\kappa < 0$ and $F_\sigma < 0$. In this situation, the driving force F_D is positive if the centre of curvature lies inside the phase of lower energy density.

2.5. Ripple implementation

The implementation of the Ripple algorithm stems from the same methodology described above for the LSNC algorithm, except for the boundary normal and curvature calculations. In the Ripple algorithm, $\boldsymbol{n}$ is calculated from the finite difference approximations in equations (5) and (6), without the improvement proposed in the LSNC. The curvature was also calculated by a finite difference approximation to equation (3), where $\boldsymbol{n}$ is calculated from $\tilde{F}$, a smoothed F field. $\tilde{F}$ is obtained by smoothing F twice using the interpolation kernel given below [12]

$$\tilde{F}_{i,j} = \frac{1}{2}F_{i,j} + \frac{1}{8}(F_{i,j-1} + F_{i,j+1} + F_{i-1,j} + F_{i+1,j}). \tag{11}$$

Replacing F with $\tilde{F}$ in the curvature calculations decreases the sensitivity to oscillations in F.

The following sections describe applications of the LSNC method to various problems of stationary and moving boundaries. In a few cases, the LSNC results are compared with those obtained with the implemented Ripple method.

3. Stationary boundaries

The local normal vector and curvature of stationary two-dimensional boundaries were calculated by the present LSNC method and by the Ripple method. Two types of stationary boundaries were examined: straight line and circular boundaries. Normal vector calculations with the two methods were compared for straight line boundaries, while curvature calculations were compared for circular boundaries.

3.1. Straight boundaries

A straight boundary was located in a two-dimensional square domain of unit side. The domain was subdivided into a stencil of 3×3 cells and, for each cell, the exact fraction intersected by the straight boundary was calculated, i.e. the discretized F field was obtained. From this field, the boundary was reconstructed using either the Ripple method or the LSNC method, resulting in a line segment within each boundary cell. The normal vector to the line segment in one boundary cell of the stencil was determined and its inclination compared with the exact inclination of the actual straight boundary.

The procedure was carried out for cells at the centre and corner of the stencil. In each case, the actual boundary crossed the examined cell at its centre and was positioned at several inclinations relative to the cell edges. Figure 2 shows the actual and the reconstructed boundary for a corner cell. The boundary reconstructed by the Ripple method is somehow distorted, while the LSNC reconstructed boundary coincides with the actual boundary. This is consistent with the results reported by Rider and Kothe [22], who showed that the LS method has second-order accuracy. On the other hand, methods based on the gradient of the F field, which is the case with the Ripple method, are prone to large errors [38].

The orientation of the normal $\boldsymbol{n}$ calculated by the two methods as a function of the exact orientation of the actual boundary is given in figure 3(*a*). The difference between the exact

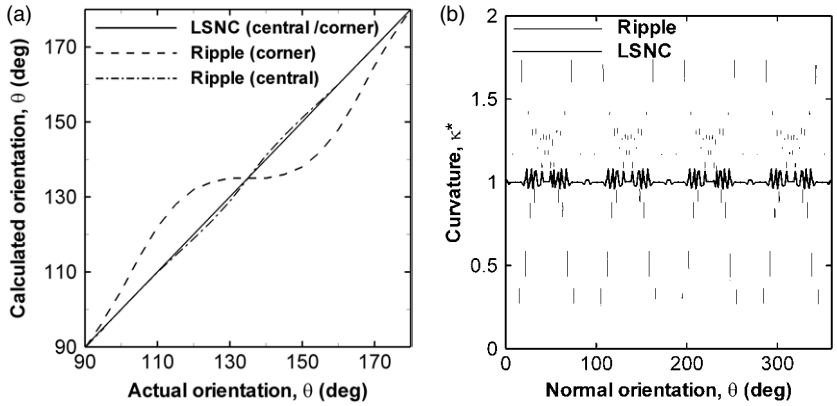

Figure 3. Calculations by Ripple and LSNC methods for (*a*) boundary normal orientation as a function of the exact orientation for central and corner cells and (*b*) local reduced curvature as a function of the normal vector orientation.

orientation and that calculated by the LSNC method is not visible in the scale of the figure. The orientation difference, however, is always larger for the Ripple method, especially for a corner cell, which is adjacent to the domain boundary. This larger error causes large distortions in simulations of a moving boundary intercepting the domain boundary, as will be shown later.

3.2. Circular boundaries

The curvature was calculated by the LSNC and the Ripple methods for a circular boundary of initial unit radius. The F field was initially obtained with the circle located at the centre of the calculation domain subdivided into a mesh of 100×100 cells and the boundary was reconstructed from it, allowing calculation of the local curvature in each boundary cell. As previously described, calculations of curvature by the Ripple method require two smoothing steps using the interpolation kernel given by equation (11).

No distortion was observed in the circular shape reconstructed by either the Ripple or the LSNC method. The local curvatures calculated by the two methods, however, differ significantly, as can be seen in figure 3(*b*), which shows the curvature as a function of the orientation of the normal vector. The local curvature calculated by both methods shows oscillations, which are the result of reconstruction errors. Nevertheless, the largest oscillations from the Ripple method are about seven times larger than those obtained with the LSNC method. In some cases they are even larger than the curvature itself.

To investigate the effect of mesh size on the accuracy of curvature calculations, the domain was subdivided into meshes of different sizes, which were characterized by the cell size, $\Delta y = \Delta x$. The results for the average local curvatures calculated in all boundary cells, κ_{av}, as a function of Δx are shown in figure 4(*a*). When the mesh is refined, κ_{av} for both methods approaches the correct unit value, especially for $\Delta x < 0.14$, although the curvature from Ripple is biased towards larger values. The magnitude of the local curvature oscillations observed in figure 3(*b*) was characterized by the standard deviation of the values around the circle and is plotted in figure 4(*b*) as a function of cell size (Δx). Initially it decreases with decreasing Δx, reaches a minimum and then increases. The standard deviation for the LSNC method, though, is at least one order of magnitude smaller than that for the Ripple method as the cell size decreases ($\Delta x < 0.14$). In conclusion, the average curvature becomes more

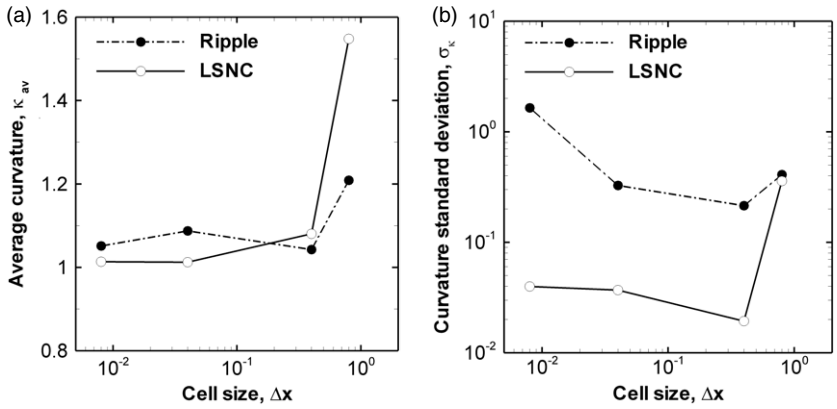

Figure 4. (*a*) Average curvature and (*b*) standard deviation of local curvature as a function of cell size calculated by the Ripple and LSNC methods.

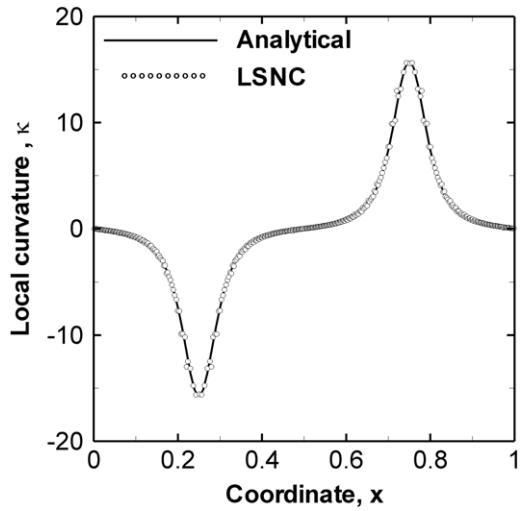

Figure 5. Curvature of a sine wave calculated analytically and with the LSNC method.

accurate as the mesh is refined, because the boundary shape is better resolved, but the local curvature oscillations reach a minimum and tend to increase for cells smaller than a certain size. This increase in curvature errors with mesh refining was also mentioned by Mosso *et al* [38].

3.3. Sinusoidal boundaries

A final test was carried out to examine the accuracy of the curvature calculated by the LSNC method. A sinusoidal boundary of unit amplitude and unit wavelength was converted into an F field in a two-dimensional square domain subdivided into a mesh of 150×150 square cells. The local curvatures of the boundary cells were calculated by the LSNC method as a function of x. Figure 5 shows that the calculated curvature is in close agreement with the analytical result. Similar good results were obtained by Jacot and Rappaz [21] using the level-set method, which is known to be very accurate for curvature calculations [30].

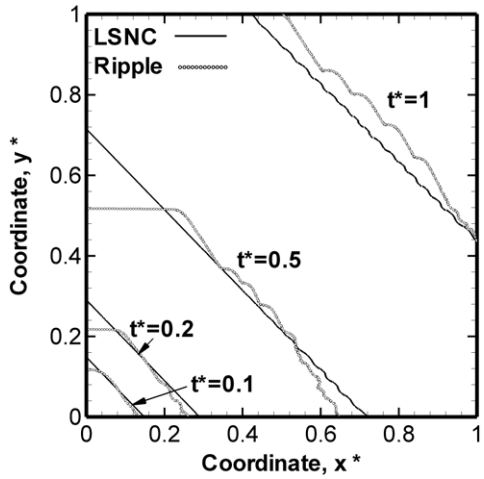

Figure 6. Evolution with reduced time (t^*) of an initially straight line boundary predicted by the Ripple and LSNC methods for a mesh of 150×150 cells and a reduced time step $\Delta t^* = 2.5 \times 10^{-4}$.

4. Moving boundaries

This section discusses the migration of initially straight and circular boundaries under the effect of curvature and a uniform, time independent driving force, F_D.

4.1. Moving straight boundaries

A straight boundary was initially located at the lower left corner of a two-dimensional square domain subdivided into a mesh of 150×150 square cells. The initial F field for this boundary was $F = 0.5$ in the lower left corner cell and $F = 1$ otherwise, so that the unit normal vector at this boundary cell is $\boldsymbol{n} = (1, 1)/\sqrt{2}$. It is convenient to introduce the following dimensionless or reduced variables: $x^* = x/L$; $y^* = y/L$; $t^* = (MF_D/L)t$; $k^* = kL$; $V_B^* = (V_B/MF_D)$; and a reduced boundary energy

$$\sigma^* = \frac{\sigma}{F_D L}, \tag{12}$$

where L is a characteristic lengthscale of the problem, which is chosen as the length of the square domain edge for this particular problem. The boundary normal is directed to the larger energy density phase, so that F_D is positive in equation (9), resulting in positive reduced times. The reduced time step used in the numerical calculations with the LSNC method was $\Delta t^* = 2.5 \times 10^{-4}$, resulting in $\Delta t^*/\Delta x^* = 0.0375$. The following equation is derived on substituting the reduced quantities in equation (9)

$$V_B^* = (\sigma^* + \sigma^{*\prime\prime})\kappa^* + 1. \tag{13}$$

For $\sigma^* = 0$, i.e. without the effect of boundary energy, figure 6 shows the evolution of the initially straight boundary calculated with the Ripple and LSNC methods. The migrating boundary predicted by Ripple becomes largely distorted, particularly at the junctions with the domain wall. Since curvature has no effect on the boundary velocity ($\sigma^* = 0$), this distortion and the oscillations are uniquely caused by errors in the normal vector calculations. As discussed in section 2.1, these errors become relatively large for cells adjacent to the domain walls, which is consistent with the distortions observed in figure 6. The boundary predicted

Table 1. Reduced parameters used in the simulations for circular boundaries.

σ^*	W^*	Δx^*	Δt^*
0.01	20	0.13	5×10^{-3}
0.94	20	0.13	5×10^{-3}
1.25	2.5	0.016	6.25×10^{-4}

by the LSNC method, however, is virtually undistorted. Some relatively low amplitude oscillations are observed, but they disappear when a negligible boundary energy, such as $\sigma^* = 10^{-4}$, is included in the calculation. This effect is probably related to the physical effect of the boundary energy in straightening a boundary to decrease its energy.

4.2. Moving circular boundaries

A circular boundary of initial radius r_0 was placed at the centre of a two-dimensional square domain and was allowed to evolve with time under the effects of surface tension and a constant driving force directed out of its centre. The evolution is described by equation (13), which for $\kappa^* = -1/r^*$ changes into the following initial value problem

$$\frac{\mathrm{d}r^*}{\mathrm{d}t^*} = 1 - \frac{\sigma^*}{r^*}, \tag{14}$$

$$r^* = 1 \quad \text{at} \quad t^* = 0, \tag{15}$$

where r^* is the reduced circle radius defined as $r^* = r/r_0$; t^* and σ^* are, respectively, the reduced time and boundary energy defined earlier, using r_0 as the characteristic lengthscale (i.e. $L = r_0$). Since initially $r^* = 1$, equation (13) shows that the circle shrinks when $\sigma^* > 1$, grows when $\sigma^* < 1$, and grows linearly when $\sigma^* = 0$ (no boundary energy effect). Notice that σ^* is analogous to a reduced or dimensionless form of the critical radius for the formation of a stable nucleus in recrystallization problems [34].

The closed form solution to equations (14) and (15) is

$$r^* + \sigma^* \ln\left(\frac{r^* - \sigma^*}{1 - \sigma^*}\right) = 1 + t^*, \tag{16}$$

where the second term on the left hand side of the equation represents the effect of boundary energy. This problem was solved by the LSNC method and by the Ripple method for three values of σ^*, namely 0.01, 0.94 and 1.25. The reduced domain size is $W^* = W/r_0$, where W is the length of the square domain; other important numerical parameters used in the simulations are given in table 1. A mesh of 150×150 cells was used in all cases; the cell size, Δx^*, was always defined equal to or smaller than 0.13, which was found to be a maximum limit for accurate curvature calculations in figure 4. The ratio between cell size and time step was constant, given as $\Delta x^*/\Delta t^* = 3.75 \times 10^{-2}$ for all simulations.

Figure 7(*a*) presents a comparison between the results from equation (16) and those obtained by the LSNC method for the instantaneous average radii of the circles. These average radii were obtained as the average distance between the segment mid-points located within boundary cells and the circle centre. The agreement between the analytical and numerical results is very good and clearly shows that, for $\sigma^* = 0.01$, i.e. much lower than 1, boundary energy effects are negligible, resulting in a linear growth behaviour, with $r^* = 1 + t^*$. For values close to unity ($\sigma^* = 0.94$), the boundary energy significantly decreases the circle growth velocity at the beginning, but its effect progressively decreases as the circle grows and the curvature decreases. Finally, for $\sigma^* = 1.25$, i.e. larger than one, the circle shrinks and finally disappears.

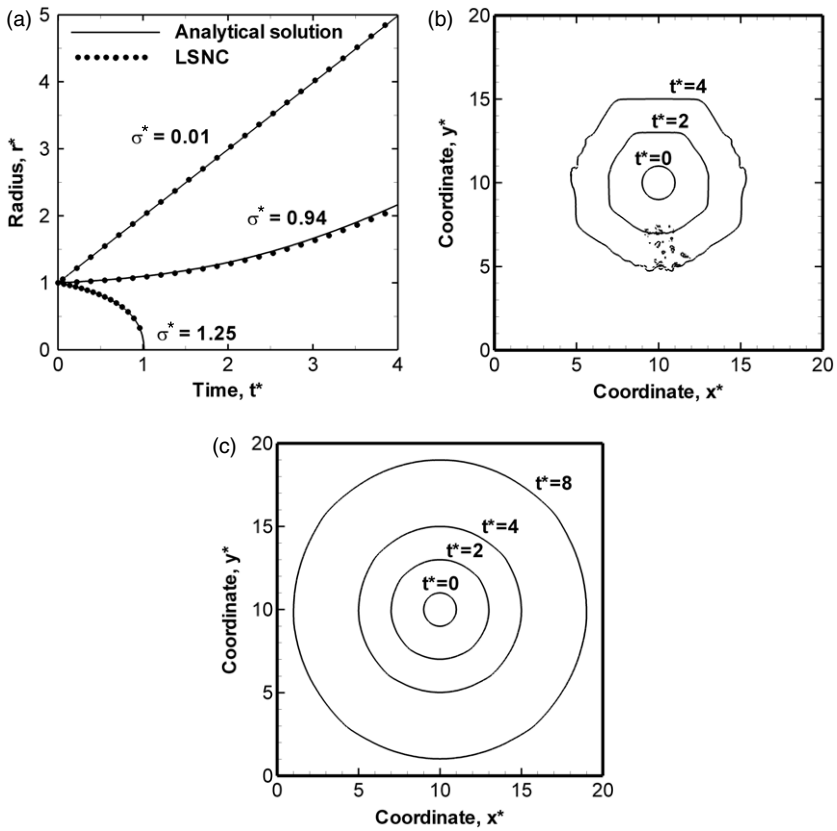

Figure 7. (*a*) Evolution of the average radius of a circular boundary predicted by the LSNC method and by the analytical solution for different reduced energies σ^*; boundary shapes for $\sigma^* = 0.01$ predicted by (*b*) the Ripple and (*c*) the LSNC methods at different times are also shown.

Figures 7(*b*) and (*c*) compare the circle evolution calculated with the Ripple and LSNC methods for $\sigma^* = 0.01$. The LSNC method preserves the boundary shape, while boundary fragmentation, i.e. flotsam, and distortions were observed with Ripple. It should be mentioned, however, that in the growth problem a non-vanishing boundary energy had to be introduced in the calculations with the LSNC method, because instabilities were observed when $\sigma^* = 0$. To avoid these instabilities, a small σ^* (for example $\sigma^* = 0.01$) may be used with a negligible effect on the growth kinetics, as shown above. In fluid dynamics problems with small surface tension values (large We number), Scardovelli and Zaleski [32] reported the break-up of the boundary and the generation of flotsam.

4.3. Moving boundaries with anisotropic boundary energy

The LSNC method was finally used to simulate the growth of an initially circular boundary of radius r_0 with anisotropic energy. This boundary is driven by curvature and a constant driving force F_D directed out of the circle. This problem has no analytical solution. Boundary energy anisotropy is relevant in problems of growth or dissolution of precipitates embedded in a solid matrix [2]. The reduced form of the boundary energy (equation (12),

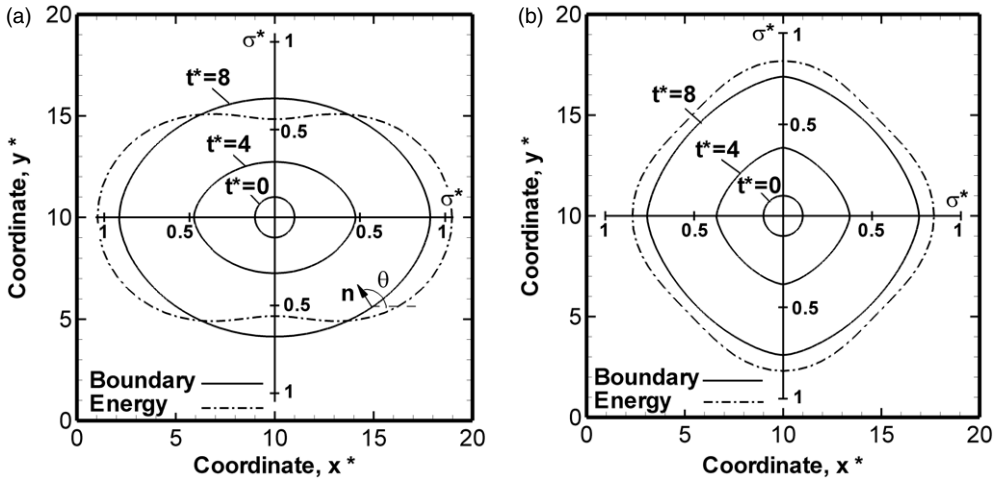

Figure 8. Time evolution predicted by the LSNC method of an initially circular boundary for (*a*) $\omega = 2$, $A = 0.3$ and (*b*) $\omega = 4$, $A = 0.06$; a polar representation of the reduced surface energy σ^* as a function of the normal vector orientation θ is also superimposed.

with $L = r_0$) was

$$\sigma^* = \sigma_0^*[1 + A\cos(\omega\theta)], \tag{17}$$

where σ_0^* is the average reduced boundary energy, A is the anisotropy amplitude, ω is the anisotropy factor; and θ is the boundary orientation, defined as the angle between its normal vector and a horizontal x-axis.

The simulations started from a circular boundary of unit reduced radius located at the centre of a two-dimensional square domain. An average boundary energy $\sigma_0^* = 0.8$ and two combinations of anisotropy amplitudes, A, and anisotropy factors, ω, were used: $A = 0.3$ with $\omega = 2$; and $A = 0.06$ with $\omega = 4$. The boundary energy as a function of orientation is shown in the polar plot superimposed on the boundary shapes of figures 8(*a*) and (*b*). Since this energy is a function of orientation, now its second derivative σ'' plays an important role in the curvature driving pressure defined by equation (10). In this problem the driving pressure is given by $F_\sigma^* = \sigma_0^*[1 + A(1 - \omega^2)\cos(\omega\theta)]\kappa$, where ω^2 is the modification introduced by $\sigma^{*\prime\prime}$. In the two problems (i.e. $\omega = 2$ and $\omega = 4$), $A(1 - \omega^2) = -0.9$, causing a minimum curvature pressure towards the centre when σ^* is at its maximum, contrary to the case in which $\sigma^{*\prime\prime}$ is not considered.

A mesh of 150×150 cells was used in the simulations and the time step was $\Delta t^* = 0.05$. The boundary evolution with time for an anisotropy factor $\omega = 2$ is given in figure 8(*a*), which shows a change in shape from the initial circle at time $t^* = 0$ into a lens shaped boundary. Initially, the left and right hand regions of the boundary move more rapidly than other regions as a result of the lower curvature pressures caused by $\sigma^{*\prime\prime}$. After some growth, the curvatures of these regions increase, increasing the curvature pressure, and the product $(\sigma^* + \sigma^{*\prime\prime})\kappa^*$ in equation (13) becomes approximately constant along the boundary; its velocity is now uniform and its shape does not change further. For an anisotropy factor $\omega = 4$, the initially circular boundary changes into a square with smoothed vertices (figure 8(*b*)). As in the previous case, the boundary appears to reach a steady-state shape after the curvature increases during growth in the regions with lower curvature pressures.

5. Summary

An algorithm termed the LSNC method was introduced to study the migration of boundaries driven by curvature (i.e. boundary energy) and an extraneous 'force' associated with an energy density difference between the two sides of the boundary. This energy may be a strain energy (as in the growth of a recrystallized grain) or one caused by applied electric or magnetic fields that give rise to a grain orientation dependence of the energy density or to a difference in chemical potential. In the LSNC algorithm the boundary is tracked using the concept of the volume fraction of phases, as in existing VOF methods. Nevertheless, the LSNC algorithm differs from these VOF methods, because both boundary normal and curvature are calculated by a LS procedure, rather than by finite difference approximations.

The results obtained with the LSNC method were compared with those obtained with the standard Ripple methods and proved more accurate. Finally the LSNC method was applied to solve a number of simple problems related to boundary migration. In these problems a uniform boundary mobility was assumed, but the method could be easily extended for anisotropic mobility, as shown for anisotropic boundary energy.

Acknowledgments

The authors thank Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) for financial support under grants 03/08576-7 and 99/10796-8, and the first author (MAM) thanks the Instituto Superior Técnico da Universidade Técnica de Lisboa for support in Lisbon and the Universidade de São Paulo for his leave permission.

References

[1] Shewmon P G 1969 *Transformation in Metal* (New York: McGraw-Hill)
[2] Martin J W, Doherty R D and Cantor B 1997 *Stability of Microstructure in Metallic Systems* (Cambridge: Cambridge University Press)
[3] Crank J 1984 *Free and Moving Boundary Problems* (Oxford: Clarendan Press)
[4] Srolovitz D J, Grest G S and Anderson M P 1986 *Acta Metall. Mater.* **34** 1833–45
[5] Hesselbarth H W and Gobel I R 1991 *Acta Metall. Mater.* **39** 2135–43
[6] Raabe D 1998 *Texture and Anisotropy of Polycrystals* **273-2** 169–74
[7] Cocks A C F and Gill S P A 1996 *Acta Mater.* **44** 4765–75
[8] Sun B, Suo Z and Yang W 1997 *Acta Mater.* **45** 1907–15
[9] Kawasaki K, Nagai T and Nakashima K 1989 *Phil. Mag.* B **60** 399–421
[10] Chen L Q 1995 *Scripta Metall. Mater.* **32** 115–20
[11] Cha P R, Kim S G, Yeon D H and Yoon J K 2002 *Acta Mater.* **50** 3817–29
[12] Lafaurie B, Nardone C, Scardovelli R, Zaleski S and Zanetti G 1994 *J. Comput. Phys.* **113** 134–47
[13] Harlow F H and Welch J E 1965 *Phys. Fluids* **8** 2182–9
[14] Hirt C W and Nichols B D 1981 *J. Comput. Phys.* **39** 201–25
[15] Meier M, Yadigaroglu G and Smith B L 2002 *Eur. J. Mech. B-Fluid.* **21** 61–73
[16] Liovic P, Rudman M and Liow J L 2002 *Appl. Math. Model.* **26** 113–40
[17] Han J W, Lee H G and Park J Y 2002 *Mater. Trans.* **43** 1816–20
[18] Cleary P, Ha J, Alguine V and Nguyen T 2002 *Appl. Math. Model.* **26** 171–90
[19] Jacot A and Rappaz M 1997 *Acta Mater.* **45** 575–85
[20] Jacot A, Maijer D and Cockcroft S 2000 *Metall. Mater. Trans.* A **31** 2059–68
[21] Jacot A and Rappaz M 2002 *Acta Mater.* **50** 1909–26
[22] Rider W J and Kothe D B 1998 *J. Comput. Phys.* **141** 112–52
[23] Scardovelli R and Zaleski S 2003 *Int. J. Numer. Methods Fluids* **41** 251–74
[24] Brackbill J U, Kothe D B and Zemach C 1992 *J. Comput. Phys.* **100** 335–54
[25] Puckett E G 1991 A volume-of-fluid interface tracking algorithm with applications to computing shock wave refraction *4th Int. Symp. on Computational Fluid Dynamics (Davis, CA)* ed H Dwyer pp 933–8

[26] Chorin A J 1985 *J. Comput. Phys.* **57** 472–90
[27] Renardy Y and Renardy M 2002 *J. Comput. Phys.* **183** 400–21
[28] Renardy Y Y, Renardy M and Cristini V 2002 *Eur. J. Mech. B-Fluid.* **21** 49–59
[29] Poo J Y and Ashgriz N 1989 *J. Comput. Phys.* **84** 483–91
[30] Sussman M and Puckett E G 2000 *J. Comput. Phys.* **162** 301–37
[31] Rudman M 1997 *Int. J. Numer. Methods Fluids* **24** 671–91
[32] Scardovelli R and Zaleski S 1999 *Annu. Rev. Fluid. Mech.* **31** 567–603
[33] Kalyanmoy D 1995 *Optimization for Engineering Design: Algorithms and Examples* (New Delhi: Prentice-Hall of India)
[34] Haessner F 1978 *Recrystallization of Metallic Materials* (Stuttgart: Riederer-Verlag)
[35] Mendelev M I and Srolovitz D J 2002 Modelling Simul. Mater. Sci. Eng. **10** R79–109
[36] Taylor J E 1992 *Acta Metall. Mater.* **40** 1475–85
[37] Taylor J E, Cahn J W and Handwerker C A 1992 *Acta Metall. Mater.* **40** 1443–74
[38] Mosso S J, Swartz B K, Kothe D B and Ferrel R C 1996 A parallel volume-tracking algorithm for unstructured meshes *Los Alamos National Laboratory Report* LA-UR-96-2420 (Los Alamos)