Applied Mathematical Modelling 77 (2020) 1704–1741

Contents lists available at ScienceDirect

Applied Mathematical Modelling

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

A smoothed particle hydrodynamics-phase field method with radial basis functions and moving least squares for meshfree simulation of dendritic solidification

Adam Yehudi Ghoneim

Department of Mechanical Engineering, University of Manitoba, Winnipeg, Manitoba R3T-5V6, Canada

ARTICLE INFO

Article history:
Received 14 April 2019
Revised 18 August 2019
Accepted 3 September 2019
Available online 10 September 2019

Keywords:
Meshfree methods
Smoothed particle hydrodynamics
Phase field method
Radial basis functions
Moving least squares
Dendritic solidification

ABSTRACT

We present a robust and efficient approach to meshfree phase-field (PF) simulation of dendritic solidification on arbitrary domain geometries using smoothed particle hydrodynamics (SPH). We use radial basis functions (RBFs) and moving least squares (MLS) as alternative approaches for constructing kernel approximation functions exhibiting a higher order of consistency than traditional kernel functions used in SPH. In the proposed smoothed particle hydrodynamics-phase field method (SPH–PFM), proper discretization of the PF order parameter at the diffuse interface region can be easily accomplished independently from the particle spacing resolution used for computing the thermal field distribution. We use an implicit geometry construction approach to automatically generate virtual boundary particles to impose Neumann-type boundary conditions at the domain boundaries. We solve the Allen–Cahn equation locally at particles constructed at a narrow band around the interface region. Additionally, only first-order derivatives of the meshfree approximation functions are needed in our implementation to solve the governing equations. Mathematical formulation and detailed analysis will be presented and discussed where we investigate the effect of the meshfree approximation scheme on the final morphology of the grown dendrite.

1. Introduction

Dendritic solidification is arguably one of the most fascinating phenomena of pattern formation occurring routinely in nature. However, aside from its natural occurrence, it directly impacts the microstructure and mechanical properties of materials subjected to a wide range of industrial processes such as additive manufacturing, welding, brazing and metal casting. The formation of dendrites during solidification involves intricate liquid–solid interfacial topologies and complex interfacial dynamics that are often difficult to study and capture in real time. This has fascinated mathematicians and scientists for decades and resulted in an extensive body of research to mathematically describe the dendritic growth phenomena. Since the developed analytical models are often limited to very simple and idealized cases, numerical techniques have become indispensable.

Dendritic solidification is a multiphase flow problem where liquid and solid phases simultaneously exist in the computational domain separated by a moving liquid–solid interface whose location at a given time is an unknown. Currently, the

E-mail address: cfd@compmath.ca

https://doi.org/10.1016/j.apm.2019.09.017
0307-904X/

two most popular numerical methods for implicit handling of propagating interfaces in numerical modelling of phase transformation are the level set and PF methods. The level set method (LSM) is a sharp-interface capturing technique developed by Osher and Sethian [1] where a level set function, Φ, implicitly representing the interface is evolved in time by solving a Hamilton–Jacobi equation. The final location of the interface is computed by determining the zero level set of the new solution. This approach can handle complex interfacial dynamics such as splitting and merging very easily. It also allows for easy computation of the interfacial normal vectors and curvature. Since its inception, the LSM has become one of the most popular methods for modelling multiphase propagating interface problems. Unfortunately, classical level set implementation often suffers from volume conservation issues which are often attributed to the reinitialization procedure required to force the level set function to be a signed distance function (SDF) typically used to implicitly define the interface. Although this reinitialization procedure is necessary to successfully obtain a level set solution, it has the side-effect of unphysically moving the interface. On the other hand, over-reinitialization tends to over-smooth the interface leading to loss in detecting minute interfacial features [2,3]. It is often very difficult to determine the frequency of reinitialization such that a well-behaved SDF is ensured while avoiding over-smoothing the interface. More sophisticated approaches proposed for resolving the conservation issues include the incorporation of a correction step once a level set solution is obtained. This can be accomplished by combining the LSM with the Volume of Fluid (VOF) method which exhibits more accurate volume conservation properties [4,5], or the use of Lagrangian interface particle methods as in [6–12]. An alternative solution to improve the accuracy of the level set solution is to eliminate the reinitialization procedure completely using a variational formulation of the level set equation such as that proposed in a number of works [13–17].

1.1. The phase field method

An alternative approach for implicit handling of a propagating interface is to treat it as a diffuse region with a finite thickness as in the phase field method (PFM) [18–20] where a phase order parameter, Ψ, is introduced such that it is 1 in phase A and 0 in phase B while it varies sharply but smoothly across an interfacial region. The PFM allows the numerical model to be treated as a continuous problem in the whole domain. Caginalp and Fife [21] adopted the PFM for interface dynamics back in the mid 80's, while Kobayashi [22] presented one of the earliest PF simulation results for dendritic solidification. Since then, the PFM has gained considerable popularity in modelling of phase transformation and multiphase flow due to its ability to directly incorporate the thermodynamics of phase transition into the formulation. It also eliminates the need to assign boundary conditions directly at the interface and avoids computation of the interface normal vectors and curvature which are automatically incorporated into the formulation. In contrast to level set methods, no reinitialization of the implicit function or computing an extended interfacial velocity field is required. In contrast to the LSM, where the sharp-interface intersects the elements [23] using a smooth and differentiable implicit function, the PFM considers the interface to have a finite thickness that is essentially smeared over a finite number of elements where Ψ is a step function that is only differentiable at the interface thickness. Therefore, it is crucial to properly discretize the interface region to successfully solve the PF equation. Caginalp and Chen [24] rigorously showed that the PF model converges to the sharp-interface limit when the interface width is much smaller than the capillary length. Wheeler et al. [25] and Wang et al. [26] have shown that the PF model requires an asymptotic expansion analysis to be performed with a small parameter proportional to the interface thickness, W. It is important to have the mesh element size to be proportional to W and only in the limit of $W \rightarrow 0$ does the PF solution converge to the sharp-interface model. To retain fidelity with the sharp-interface model, the interface width should be chosen according to the "thin-interface" analysis of Karma and Rappel [27] to be smaller than the microstructure morphology but larger than the capillary length scale. The asymptotic analysis of Karma and Rappel allowed to simulate the PF model with zero or non-zero interface kinetics without the need to make $W \rightarrow 0$ but with an interface thickness at the order of the capillary length. Unfortunately, even if the PF model is solved within the "thin-interface" limit, adequately resolving the interface thickness, which is typically many orders of magnitude smaller than the computational domain size, is paramount for obtaining a meaningful PF solution.

A large body of existing work resolves this issue by using the finite difference method (FDM) on structured grids with a small-enough grid resolution to solve the governing equations [28–34]. This is mainly due to the simplicity and convenience of implementation of finite differences. However, the solution is limited to only simple domain geometries and often influenced by grid anisotropy. Additionally, such an approach would make 3D simulations to be computationally expensive. Adaptive mesh refinement techniques within the framework of the FDM and the finite element method (FEM) has been used for solving the PF equations in an effort to sufficiently resolve the interface region, minimize unnecessary over-resolution of the mesh away from the interface, and perform the computation within manageable processing times [35–38]. In such treatments, very fine element sizes at the order of, or smaller than, the scale of the interface thickness are generally used at the interfacial regions to ensure the interface thickness is adequately resolved. The size of elements is then adaptively coarsened away from the interface region. Adaptive mesh refinement aims to sufficiently resolve the necessary features in the domain and is in fact analogous to adaptive meshing techniques used in conventional interface tracking models and suffers from the same difficulties of being tedious and time consuming where constant re-meshing is required as the interface evolves. The thermal conduction also occurs at a larger scale than the interface thickness such that the thermal field varies across the whole domain. Using the same mesh resolution to solve for both Ψ and the thermal profile may lead to an unnecessarily over-resolved thermal profile at the interface region and increases the computation time. This was addressed

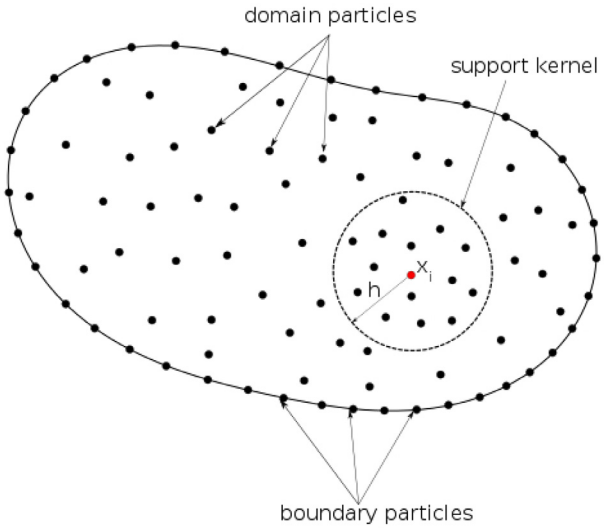

Fig. 1. Schematic of particle distribution showing the support kernel of radius h for a particle $\mathbf{x}_i$.

by Hu et al. [39] by using a multi-mesh adaptive approach where Ψ is solved on a finer mesh which adequately resolves the interface thickness while the thermal profile is solved using a secondary much coarser mesh.

In the past two decades, the development of meshfree methods has emerged as a new class of computational methods which can be used as an alternative to the FDM and the FEM such that the approximation of field variables is constructed based on scattered points without explicit mesh connectivity. However, application of meshfree methods for PF modelling is a very recent development that is still in its infancy. Recently, Rosolen et al. [40] and Peco et al. [41] presented a Lagrangian meshfree method for PF simulation of biomembranes using local maximum-entropy meshfree basis functions. Dahghan and Abbaszadeh used a linear combination of the shape functions of local RBFs collocation with moving Kriging interpolation for solving the Cahn–Hilliard equation [42]. Dahghan and Mohammadi also used differential quadrature which was coupled with RBFs for meshfree solution of the Cahn–Hilliard PF equation [43] while a pseudo-spectral RBF and a MLS method was used for meshfree solution of the PF crystal equation in [44]. Talat et al. used a meshfree diffuse approximate method for PF simulation of Rayleigh–Taylor instability [45]. Zhou and Li [46] used the meshfree reproducing kernel particle method for solving 1D problems and also applied it for a simple 2D problem of a shrinking circle. Song et al. [47] proposed a particle difference method for PF simulation of the Cahn–Hilliard equation. In this method, smoothness of the weight functions used is not required for obtaining a solution given that it is non-negative and continuous.

1.2. Smoothed particle hydrodynamics

Smoothed particle hydrodynamics (SPH) is a meshfree particle method that was first introduced in the 70's by Lucy [48] and Gingold and Monaghan [49] to simulate non-axisymmetric phenomena in astrophysics. In this approach, particles without connectivity are distributed in the computational domain as illustrated in Fig. 1. The partial differential equations (PDEs) are solved directly where field variables and their derivatives are approximated by summing the product of the field variable of neighbouring particles within a support kernel and an appropriate weighting function [50]. SPH can be classified as a strong-form point collocation method where the points are treated as particles with a specified mass and local collocation is done at the points (particles) themselves. One of the attractive properties of using the methodology of SPH is that the PDE is solved at the kernel level only while avoiding forming large systems of equations or global matrices that need to be solved. This allows the SPH to tackle very large spatial problems and allows for easier parallel processing. SPH has been used extensively in numerical simulation of "hydrodynamical" problems where the idea of SPH is to replace the fluid by a set of particles which can be advected in space in a Lagrangian manner. As such, following the motion of the particles would provide insight on the evolution of the fluid being analyzed. The SPH technique was generalized into the smoothed particle interpolation (SPI) method in [51] where SPH is viewed as a strictly interpolation scheme suitable for solving general PDEs not necessarily related to hydrodynamical systems such that the particles are treated as simply interpolation points that can be fixed (the Eulerian approach) or moving (the Lagrangian approach).

The SPH is based on approximating a field variable $f(\mathbf{x})$ at particle $\mathbf{x}$ by:

$$f(\mathbf{x}) = \int_{\Omega} f(\mathbf{x}')\delta(\mathbf{x}-\mathbf{x}')dV \tag{1}$$

where $\mathbf{x}-\mathbf{x}'$ is the distance between particle $\mathbf{x}$ and neighbouring particle $\mathbf{x}'$, Ω is the volume of the domain containing $\mathbf{x}$ and $\delta(\mathbf{x}-\mathbf{x}')$ is the Dirac delta function defined by:

$$\delta(\mathbf{x}-\mathbf{x}') = \begin{cases} 1, & \mathbf{x}=\mathbf{x}' \\ 0, & \mathbf{x}\neq\mathbf{x}' \end{cases} \tag{2}$$

To establish a meaningful discrete numerical model, the delta function $\delta(\mathbf{x}-\mathbf{x}')$ is replaced by a smooth kernel function $W(\mathbf{x}-\mathbf{x}', h)$ where h is a kernel smoothing length defining the size of the support area. The function $f(\mathbf{x})$ can then be approximated by $\hat{f}(\mathbf{x})$ by:

$$\hat{f}(\mathbf{x}) \approx \int_{\Omega} f(\mathbf{x}')W(\mathbf{x}-\mathbf{x}', h)dV \tag{3}$$

Eq. (3) can be re-written in discrete form for particle $\mathbf{x}_i$ in terms of the set of neighbouring particles $\{\mathbf{x}\}_{nbrs}$ such that:

$$\hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} f(\mathbf{x}_j)W(\mathbf{x}_i-\mathbf{x}_j, h)\Delta V_j = \sum_{j=1}^{n} f(\mathbf{x}_j)N_j(\mathbf{x}_i) \tag{4}$$

where n is the total number of particles in the kernel containing $\{\mathbf{x}\}_{nbrs}$ and the term $W(\mathbf{x}_i-\mathbf{x}_j, h)\Delta V_j$ forms an interpolant $N_j(\mathbf{x}_i)$ that weighs the influence of a neighbour particle $\mathbf{x}_j$ on the solution at $\mathbf{x}_i$ such that its influence decreases as it gets farther from $\mathbf{x}_i$. Indeed, it could be argued that this is basically the approximation approach of all strong-form meshfree methods that were developed after the SPH where they mainly differ in the manner of constructing the interpolant $N_j(\mathbf{x}_i)$. Eq. (4) can be re-expressed as:

$$\hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} f(\mathbf{x}_j)W(\mathbf{x}_i-\mathbf{x}_j, h)\Delta V_j = \sum_{j=1}^{n} f(\mathbf{x}_j)W(\mathbf{x}_i-\mathbf{x}_j, h)\frac{1}{\rho_j}(\rho_j\Delta V_j) \tag{5}$$

or alternatively:

$$\hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} f(\mathbf{x}_j)W(\mathbf{x}_i-\mathbf{x}_j, h)\frac{m_j}{\rho_j} \tag{6}$$

where ρ_j and m_j is the density and mass of particle j, respectively.

To choose a suitable kernel function $W(\mathbf{x}-\mathbf{x}', h)$ within the context of SPH, Monaghan proposed three main conditions [50]. The first is the so called normalization (or unity) condition defined as:

$$\int_{\Omega} W(\mathbf{x}-\mathbf{x}', h)dV = 1 \tag{7}$$

The second condition is the so called Delta function property defined as:

$$\lim_{h\to 0} W(\mathbf{x}-\mathbf{x}', h)\Delta V = \delta(\mathbf{x}-\mathbf{x}') \tag{8}$$

This means that as the kernel smoothing length h decreases, the delta property in Eq. (2) is recovered. The third condition is the compactness condition defined as:

$$W(\mathbf{x}-\mathbf{x}', h) = 0 \quad \forall \quad |\mathbf{x}-\mathbf{x}'| > h \tag{9}$$

A popular kernel function satisfying these conditions is the Gaussian kernel defined by:

$$W(R, h) = \frac{1}{\pi^{(d/2)}h^d}e^{-R^2} \tag{10}$$

where d is the spatial dimension of the problem $d = \{1, 2, 3\}$ and $R = \frac{|\mathbf{x}-\mathbf{x}'|}{h}$. Another kernel function is the cubic spline kernel function expressed as:

$$W(\mathbf{x}-\mathbf{x}', h) = C_h \begin{cases} (2-R)^3 - 4(1-R)^3 & if \quad 0 \le R < 1 \\ (2-R)^3 & if \quad 1 \le R < 2 \\ 0, & if \quad q \ge 2 \end{cases} \tag{11}$$

where C_h is $1/(6h)$ in one dimension, $15/(14\pi h^2)$ in two dimensions and $1/(4\pi h^3)$ in three dimensions. Wendland kernel functions have also been used in SPH. One type is the so-called quintic Wendland kernel function expressed as:

$$W(\mathbf{x}-\mathbf{x}', h) = C_h \begin{cases} (2-R)^4(2R+1) & if \quad 0 \le R \le 2 \\ 0 & if \quad R > 2 \end{cases} \tag{12}$$

where C_h is $7/(4\pi h^2)$ in 2D and C_h is $21/(16\pi h^3)$ in 3D.

Applying the same methodology to approximate the first and second derivative of a function $f(\mathbf{x}_i)$ at $\mathbf{x}_i$ as was done in Eq. (4), one can approximate the first derivative as:

$$\nabla\hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} f(\mathbf{x}_j)\nabla W(\mathbf{x}_i-\mathbf{x}_j, h)\Delta V_j \tag{13}$$

and the second derivatives is approximated as:

$$\nabla^2 \hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} f(\mathbf{x}_j) \nabla^2 W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j \tag{14}$$

where n is the total number of particles in the kernel containing $\{\mathbf{x}\}_{nbrs}$.

The main difficulty in using the kernel functions in Eqs. (10)–(12) directly into the approximation in Eq. (4) is that it fails even the zero-order consistency (reproducibility) condition where $\sum_{j=1}^{n} W(\mathbf{x}_i - \mathbf{x}_j, h)\Delta V_j \neq 1$ at the domain boundaries due to the kernel being only partially contained within the domain. In other words, the kernel functions used in classical SPH will not satisfy the partition of unity (POU) condition necessary for consistency. Exhibiting the consistency property is a necessary ingredient to achieve convergence of a numerical solution. An easy approach to resolve this issue is to re-assign the density weights to particles within the kernel such that the POU property is recovered [52]. This can be done by normalizing Eqs. (4), (13) and (14) as:

$$\hat{f}(\mathbf{x}_i) \approx \frac{1}{D} \sum_{j=1}^{n} f(\mathbf{x}_j) W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j \tag{15}$$

$$\nabla \hat{f}(\mathbf{x}_i) \approx \frac{1}{D} \sum_{j=1}^{n} f(\mathbf{x}_j) \nabla W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j \tag{16}$$

$$\nabla^2 \hat{f}(\mathbf{x}_i) \approx \frac{1}{D} \sum_{j=1}^{n} f(\mathbf{x}_j) \nabla^2 W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j \tag{17}$$

where $D = \sum_{j=1}^{n} W(\mathbf{x}_i - \mathbf{x}_j, h)\Delta V_j$ and

$$\frac{1}{D} \sum_{j=1}^{n} W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j = \sum_{j=1}^{n} \frac{W(\mathbf{x}_i - \mathbf{x}_j, h)\Delta V_j}{\sum_{j=1}^{n} W(\mathbf{x}_i - \mathbf{x}_j, h)\Delta V_j} = 1 \tag{18}$$

This approach coincides with Shepard's interpolation which is itself considered a special case of MLS approximation. However, Shepard's interpolation is only zero-order continuous. A more sophisticated approach is to use MLS approximation to ensure the satisfaction of POU condition and recover the consistency condition to a higher order [53–55]. Radial basis functions (RBF) were also later used in SPH to enhance its accuracy [54,56].

Note that if the function $f(\mathbf{x})$ is constant then the approximation in Eq. (13) will yield a non-zero answer. A better approximation of the first derivative is [57]:

$$\nabla \hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} \Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) \Big) \nabla W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j = \sum_{j=1}^{n} \Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) \Big) \nabla W(\mathbf{x}_i - \mathbf{x}_j, h) \frac{m_j}{\rho_j} \tag{19}$$

which is arrived at via the relation [57]:

$$\Phi(\nabla f) = \nabla(f\Phi) - f\nabla\Phi \tag{20}$$

Directly computing the second-order derivative by directly computing the second-order derivative of the weight function as presented in Eq. (17) is prone to instabilities and is very sensitive to particle distribution. One approach is to approximate the second-order derivative by differentiating the first-order derivatives of the weighting functions twice such that:

$$\nabla^2 \hat{f}(\mathbf{x}_i) = \nabla \cdot \nabla \hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} \Big(\nabla f(\mathbf{x}_j) - \nabla f(\mathbf{x}_i) \Big) \cdot \nabla W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j \tag{21}$$

A better approach was presented by Brookshaw [58], Cleary [59], Monaghan [57] and Cleary and Monaghan [60] which uses a Taylor series expansion in a similar manner to finite differences to achieve the approximation:

$$\nabla^2 \hat{f}(\mathbf{x}_i) \approx 2 \sum_{j=1}^{N} \frac{\Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) \Big)}{|\mathbf{r}_{ji}|^2} \mathbf{r}_{ji} \cdot \nabla W(\mathbf{x}_i - \mathbf{x}_j, h) \Delta V_j \tag{22}$$

where $\mathbf{r}_{ji} = \mathbf{r}_j - \mathbf{r}_i$ is the unit vector between interacting particles j and i. An analysis done in [61] using conventional SPH kernel functions shows that this approximation yields the most stable results without oscillations if compared with computing the second order derivatives directly using Eq. (17) or (21). A similar formulation was used by Monaghan et al. [62] for modelling solidification using SPH. It is interesting to note that in that work, all particles are fixed in space and the non-solidified particles were viewed as virtual particles. Rather than aggregating the solidified particles which would be much more computationally expensive, an enthalpy method is used to track the volume fraction change of the solidified liquid particles.

Another obstacle in SPH is that while it can indeed be used for handling multiphase problems, it is often difficult to have a well-defined continuous phase interface. As such, application of surface tension effects at the interface is often difficult.

An approach to resolve this issue is to use an interface reconstruction algorithm. This was done in [63] where RBFs were used to reconstruct an interface surface. This approach was reported to be more computationally expensive than traditional SPH since the interface surface has to be reconstructed continuously as the interface propagates. Another approach was presented in [64] where an interfacial surface mesh construction is done based on the marching cubes method. The ability of easy incorporation of surface tension effects is essential in modelling dendritic solidification as it is well known to affect the formation and growth of dendrite branches. Employing a more efficient approach which does not require constant reconstruction of the interface is therefore worthwhile.

1.3. Purpose and scope

It is well known that in PF simulations, proper discretization of the thickness of the moving interface is a requirement for obtaining a successful solution. In contrast to finite differences and finite elements, the flexibility of meshfree methods to adaptively resolve essential features of the interface with little computational effort in addition to the flexibility of modifying the order of interpolation and size of the support kernel with minimal effort makes the application of a meshfree method for PF modelling quite attractive. Coupling the PFM with a SPH formulation would allow for easy and convenient imposition of surface tension effects which are directly incorporated in the PF formulation without the need to reconstruct the interface surface as it evolves. All interfacial dynamics, such as merging and splitting, are handled easily and naturally using the PF approach. Moreover, there is currently very limited data on the effect of various meshfree approximation parameters on the morphology of the dendrite. For example, RBF interpolation requires the selection of a shape parameter which affects the shape of the basis functions. However, there is hardly any information on the effect of the selected shape parameter on the dendrite growth rate and the final morphology of the dendrite. Similarly, the impact of the choice between the different types of weight functions used in traditional SPH kernel functions, MLS and RBFs on the final solution of the dendrite is currently unavailable. Effect of the spatial distribution of the domain particles on branch growth of dendrites also needs further investigation.

Therefore, the goal of this work is to further the development of meshfree PF modelling where, to our knowledge, this is the first paper which presents a meshfree PFM for numerical simulation of dendritic solidification. We propose the SPH–PFM as an advancement of the work by Monaghan et al. [62] for predicting dendritic growth during solidification where we take advantage of the local features of SPH to solve the governing PF equations. We employ a methodology where only particles at a narrow band around the interface are modified while all other domain particles remain fixed. We use an implicit boundary construction approach to automatically generate virtual boundary particles to impose Neumann-type boundary condtions. We use MLS and RBFs for computing the interpolation functions to a higher degree of consistency than traditional SPH kernel functions. This implementation requires solving a local system of equations proportional to the number of particles present in the support kernel. We take into account several factors (outlined in Section 3) to alleviate the computation demands and ensure a robust and efficient meshfree model suitable for PF simulations.

Mathematical formulation and implementation will be presented and discussed in the following section. This will be followed by discussing the details of implementation and algorithm. We begin our analysis by presenting a detailed quantitative analysis of SPH using traditional SPH kernel functions, RBFs and MLS for approximating the spatial derivatives of a non-trivial benchmark function. We will then investigate the use of traditional SPH kernel functions and various meshfree interpolation parameters in RBFs and MLS on the resulting dendrite morphology within the framework of the SPH–PFM. Numerical simulation results of dendritic solidification obtained using the proposed SPH–PFM will be compared with that obtained using conventional finite differences. We will also demonstrate the ability of the SPH–PFM to easily capture the effect of various PF parameters on the resulting dendrite morphology. Finally, we demonstrate the use of SPH–PFM for modeling the growth of multiple dendrites on arbitrary domain geometries with internal cavities.

2. Mathematical formulation

2.1. The phase-field model for dendritic solidification

The PF model presented in this paper is based on the work by Kobayashi [22]. However, the general methodology of the SPH-PFM allows it to be used regardless of PF model employed. Consider an arbitrary domain $\Omega \subset \mathbb{R}^d$ in d-dimensional space such that $d = \{2, 3\}$ and let us assume that two sub-domains Ω_1 and Ω_2 are present in Ω. Let the interface Γ between Ω_1 and Ω_2 have a finite thickness W_0 such that $\Omega_1 \cup \Omega_2 \cup \Gamma = \Omega$. Let $\Psi(\mathbf{x}, t)$ be an implicit function constructed for all $\mathbf{x} \in \Omega$ at time t such that $\Psi(\mathbf{x}, t) = 0\ \forall \mathbf{x} \in \Omega_1$ and $\Psi(\mathbf{x}, t) = 1\ \forall \mathbf{x} \in \Omega_2$ while $0 < \Psi(\mathbf{x}, t) < 1\ \forall \mathbf{x} \in \Gamma$. The evolution of $\Gamma(t)$ is governed by a PF equation. As with most PF formulations, our starting point is the Ginzburg–Landau energy functional which is expressed as:

$$F(\Psi, m) = \int_V \frac{1}{2}\epsilon^2 |\nabla \Psi|^2 + f(\Psi, m) dV \tag{23}$$

where the first term in the integrand describes the gradient energy while the second term describe local free energy due to the interface which can be expressed as a double-well potential with local minimum at $\Psi = 0$ and $\Psi = 1$. The term ϵ is the coefficient of anisotropic gradient energy which determines the thickness of interface layer while m is the driving force

for interfacial motion. While there are different forms for the free energy $f(\Psi, m)$, the following function is adopted:

$$f(\Psi, m) = \frac{1}{4}\Psi^4 - \left(\frac{1}{2} - \frac{1}{3}m(T)\right)\Psi^3 + \left(\frac{1}{4} - \frac{1}{2}m(T)\right)\Psi^2 \tag{24}$$

The parameter m is assumed to depend on the thermal undercooling and expressed as:

$$m(T) = \left(\frac{\alpha}{\pi}\right)\tan^{-1}\left(\gamma_o(T_{eq} - T)\right) \tag{25}$$

where α and γ_o are positive constants and T_{eq} is the equilibrium temperature. The anisotropy term, ϵ, is assumed to depend on the normal vector at the interface and expressed as

$$\epsilon = d_o\left(1 + A\cos(\beta(\theta - \theta_o))\right) \tag{26}$$

where d_o is the anisotropy mean value, A is the strength of anisotropy, β controls the mode of anisotropy, θ_0 is the reference orientation angle to the x-axis. The growth angle θ is expressed as:

$$\theta = \tan^{-1}\left(\frac{\partial\Psi/\partial y}{\partial\Psi/\partial x}\right) \tag{27}$$

The Allen–Cahn equation is the time-dependent Ginzburg–Landau functional and expressed as:

$$\tau\frac{\partial\Psi}{\partial t} = -\frac{\partial F}{\partial\Psi} \tag{28}$$

taking the functional derivative of Eq. (23), the transient equation is expressed as:

$$\tau\frac{\partial\Psi}{\partial t} = \nabla.(\epsilon^2\nabla\Psi) + \Psi(1-\Psi)\left(\Psi - \frac{1}{2} + m(T)\right) - \frac{\partial}{\partial x}\left(\epsilon\frac{\partial\epsilon}{\partial\theta}\frac{\partial\Psi}{\partial y}\right) + \frac{\partial}{\partial y}\left(\epsilon\frac{\partial\epsilon}{\partial\theta}\frac{\partial\Psi}{\partial x}\right) \tag{29}$$

At the boundaries of the domain, a zero-flux Neumann boundary condition exist such that:

$$\nabla\Psi|_{\mathbf{n}} = 0 \tag{30}$$

where $\mathbf{n}$ is the normal vector to the boundary. The temperature field $T(x, t)$ is evolved by:

$$\frac{\partial T}{\partial t} = K\nabla^2 T + L\frac{\partial\Psi}{\partial t} \tag{31}$$

where L and K are the non-dimensional latent heat and thermal conductivity, respectively, where K is weighted based on the distribution of Ψ at time t. If $\Psi = 1$ expresses the solid phase while $\Psi = 0$ expresses the liquid phase then $K(x, t)$ can be expressed as:

$$K(x,t) = K_l(1-\Psi) + K_s(\Psi) \tag{32}$$

Where subscripts l and s correspond to liquid and solid phases, respectively. At the boundaries of the domain, a zero-flux Neumann boundary condition exists such that:

$$\nabla T|_{\mathbf{n}} = 0 \tag{33}$$

where $\mathbf{n}$ is the normal vector to the boundary.

2.2. The smoothed particle hydrodynamics-phase field method

While SPH typically treat the domain particles in a Lagrangian sense where they are advected in space, in this work we treat the particles in the domain in an Eulerian manner where they are fixed and only the set of particles that are around the interface region are updated. One of the most attractive features of this approach is that the interpolation functions at the domain particles can be computed only once during pre-processing and used for all time steps. New interpolation functions are computed only as necessary at the set of updated particles within the interface region. This is in contrast to the conventional treatment in SPH where interpolation functions are computed at each time step and for all particles in the domain.

To accomplish this, a set of particles $\{\mathbf{x}\}_{domain}$ are first generated in the whole domain and are kept fixed in space throughout the simulations. We take advantage of the fact that the PF order parameter Ψ is a step function that is only differentiable at the interface region. Therefore, a localized computational approach for meshfree solution of the PF equation can be adopted. First, a set of interpolation functions $\{N, \nabla N\}$ are computed at each fixed domain particle $\mathbf{x}_i$ using $\{\mathbf{x}\}_{nbrs}$ as support particles. Since the interpolation functions depend only on the spatial location of the particles, they can be computed only once during pre-processing and reused for all time steps. Also, since the interpolation functions are constructed at the kernel level only without the need for assembling a global matrix for the whole domain, easier parallelization is feasible where multiple kernels can be processed simultaneously.

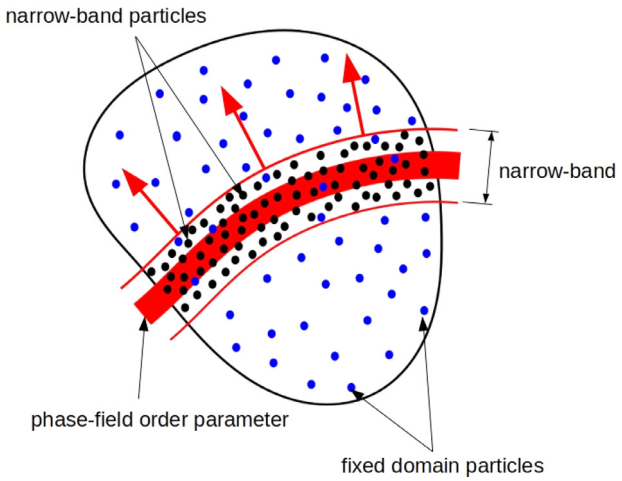

Fig. 2. Illustration of particles located within a narrow band constructed around a diffuse interface.

Since at this stage the resolution of the domain particles may not be sufficient to properly resolve the interface thickness, we therefore construct a narrow band of a user-specified width around the liquid–solid interface as illustrated in Fig. 2. Additional particles $\{\mathbf{x}\}_{nb}$, are then added locally within the narrow band at a resolution that ensures that the interface-thickness is properly resolved. As such, the nodal resolution of $\{\mathbf{x}\}_{nb}$ can be smaller than and independent from that used in $\{\mathbf{x}\}_{domain}$. The temperature at the added particles is then interpolated using the surrounding domain particles by:

$$T(\mathbf{x}_{nb})_i = \sum_{j=1}^{k} T(\mathbf{x}_d)_j N_{ij} \tag{34}$$

where $\mathbf{x}_{nb}$ corresponds to a narrow band particle, $\mathbf{x}_d$ corresponds to a neighbour domain particle, k is the total neighbouring domain particles, while N_{ij} is the interpolation function between particles $\mathbf{x}_{nb}$ and $\mathbf{x}_d$.

Once the thermal distribution at the narrow band particles is determined, the PF equation is then solved at the narrow band particles where a new Ψ at the next time step is determined. If the width of the narrow band is sufficiently larger than the interface thickness, then the newly constructed narrow band particles do not need to be reconstructed at each time step and the interpolation functions constructed at the narrow band at a time step can be reused for the next time step leading to additional computational savings. Also if the width of the narrow band is chosen to be larger than the resolution of the fixed domain particles, then a set of domain particles are guaranteed to exist within the narrow band. Therefore, the newly computed Ψ at those domain particles within the narrow band are used to solve the thermal equation everywhere in the domain using the already constructed interpolation functions. The process is repeated again for the next time step where the newly computed thermal field at the domain particles is then used to evolve Ψ at the narrow band particles. At the domain particles located outside the narrow band, Ψ is assumed to be far away from the interface and therefore hold no change in their PF solution such that:

$$\frac{\partial \Psi}{\partial t} = 0 \tag{35}$$

as such, they are assigned the same Ψ as that of the previous step. The transient thermal equation outside the narrow band then reduces to simply:

$$\frac{\partial T}{\partial t} = K\nabla^2 T \tag{36}$$

Indeed one can simply construct fixed domain particles eveywhere in the domain at a resolution that adequately resolves the interface thickness then construct the meshfree interpolation functions everywhere once during pre-processing and reuse for all time steps. While this will allow for handling arbitrary domain geometries, it unnecessarily over-resolves the thermal field and may not be practical for 3D phase-field simulations where an adaptive scheme becomes indespensible given the typically very large disparity between the interface thickness and domain size. The localized and adaptive approach taken in this work takes advantage of the ease and flexibility of meshfree methods to adequately resolve essential features in the domain (the interface) by simply adding/removing particles while avoiding un-necessary over-resolution elsewhere and takes advantage of the fact that the PF function is differentiable only at the diffuse region such that computing resources are reserved only to where they are needed at the interface region. As such it is particularly attractive for phase field modelling.

Solving the set of PF Eqs. (29) and (31) requires computing first-order and second-order derivatives of the field variable. In this work, first-order derivatives are approximated using Eq. (19) while the second-order Laplacian operators are

approximated using Eq. (22). As such, a maximum of only first-order derivatives of the kernel functions are needed for all computations in this work. Using an explicit forward time-stepping scheme, the PF Eq. (29) can then be re-expressed in its discrete particle form as:

$$\begin{aligned}\Psi_i^{t+1} = \Psi_i^t + \Delta t(\Psi_i^t(1-\Psi_i^t)(\Psi_i^t - \frac{1}{2} + m(T))) + 2\Delta t\sum_{j=1}^{n}(\epsilon_i^2+\epsilon_j^2)(\Psi_j^t-\Psi_i^t)\frac{\mathbf{r}_{ji}}{|\mathbf{r}_{ji}|^2}\cdot\nabla N_{ij}\\ -\Delta t\sum_{j=1}^{n}\left((\epsilon\frac{\partial\epsilon}{\partial\theta}\sum_{j=1}^{n}(\Psi_j^t-\Psi_i^t)\frac{\partial N_{ij}}{\partial y})_i - (\epsilon\frac{\partial\epsilon}{\partial\theta}\sum_{j=1}^{n}(\Psi_j^t-\Psi_i^t)\frac{\partial N_{ij}}{\partial y})_j\right)\frac{\partial N_{ij}}{\partial x}\\ +\Delta t\sum_{j=1}^{n}\left((\epsilon\frac{\partial\epsilon}{\partial\theta}\sum_{j=1}^{n}(\Psi_j^t-\Psi_i^t)\frac{\partial N_{ij}}{\partial x})_i - (\epsilon\frac{\partial\epsilon}{\partial\theta}\sum_{j=1}^{n}(\Psi_j^t-\Psi_i^t)\frac{\partial N_{ij}}{\partial x})_j\right)\frac{\partial N_{ij}}{\partial y}\end{aligned} \tag{37}$$

where $\mathbf{r}_{ji} = \mathbf{x}_j - \mathbf{x}_i$, $\nabla N_{ij} = \nabla W(\mathbf{x}_i - \mathbf{x}_j, h)\Delta V$ is the gradient of the constructed interpolation functions, Δt is the time step size, subscript t corresponds to the current time step while subscript $t+1$ corresponds to the future time step, and n is the total number of particles within a support kernel. Similarly, the thermal equation is expressed in its discrete particle form as:

$$T_i^{t+1} = T_i^t + \Delta t\sum_{j=1}^{n}(K_i+K_j)(T_j^t - T_i^t)\frac{\mathbf{r}_{ji}}{|\mathbf{r}_{ji}|^2}\cdot\nabla N_{ij} + L_i\frac{\Psi_i^{t+1}-\Psi_i^t}{\Delta t} \tag{38}$$

The set of Eqs. (37) and (38) describe the 2D phase-field evolution model for dendritic solidification expressed in discrete particle form. In this work, three sets of interpolation functions $\{N, \nabla N\}$ are constructed to solve these equations. The first set $\{N^{SPH}, \nabla N^{SPH}\}$ uses traditional SPH kernel weight functions. The other two sets $\{N^{RBF}, \nabla N^{RBF}\}$ and $\{N^{MLS}, \nabla N^{MLS}\}$ use RBF and MLS, respectively, as alternative approaches for constructing kernel interpolation functions as discussed in the next two sections.

2.2.1. Constructing interpolation functions using radial basis functions

To construct RBFs, the Euclidean distance r between a particle $\mathbf{x}_i$ to its neighbour $\mathbf{x}_j$ is computed such that:

$$r = |\mathbf{x}_i - \mathbf{x}_j| \tag{39}$$

In the present work, we use Gaussian exponential RBFs of the form

$$R(\mathbf{x}_i) = e^{-\gamma(\frac{r}{h})^2} \tag{40}$$

and inverse multi-quadrics RBFs expressed as:

$$R(\mathbf{x}_i) = \frac{1}{\sqrt{\left(r^2 + (\gamma h)^2\right)}} \tag{41}$$

where h is the radius of the support kernel and γ is a shape parameter. For a particle $\mathbf{x}_i$, a function $f(\mathbf{x}_i)$ is approximated using RBFs by:

$$f(\mathbf{x}_i) = \sum_{j=1}^{n} R_j(\mathbf{x}_i)a_j + \sum_{q=1}^{m} p_q(\mathbf{x}_i)b_q = \{\mathbf{R}(\mathbf{x}_i) \quad \mathbf{p}(\mathbf{x}_i)\}\{\mathbf{a} \quad \mathbf{b}\}^T \tag{42}$$

where $R_p(\mathbf{x}_i)$ is a RBF evaluated at the particle $\mathbf{x}_i$, $P_q(\mathbf{x}_i)$ is a monomial evaluated at $\mathbf{x}_i$, n is the total number of support particles, m is the number of polynomial basis functions, while a_j and b_q are unknowns to be determined. For a given point $\mathbf{x}_i$ with a given n support particles:

$$\mathbf{R}(\mathbf{x}_i) = \{R_1(\mathbf{x}_i),\ R_2(\mathbf{x}_i),\ R_3(\mathbf{x}_i),\ \ldots..,\ R_n(\mathbf{x}_i)\} \tag{43}$$

$$\mathbf{a} = \{a_1,\ a_2,\ a_3,\ \ldots..,\ a_n\} \tag{44}$$

and for a given m polynomial basis function:

$$\mathbf{p}(\mathbf{x}_i) = \left\{1, x, y, z, x^2, y^2, z^2, xy, xz, yz, \ldots\right\} \tag{45}$$

$$\mathbf{b} = \{b_1,\ b_2,\ b_3,\ \ldots,\ b_m\} \tag{46}$$

To satisfy Eq. (42) at all support particles, we express it in matrix form:

$$\mathbf{f} = [\mathbf{R} \quad \mathbf{P}]\{\mathbf{a} \quad \mathbf{b}\}^T \tag{47}$$

where

$$\mathbf{f} = \{f(\mathbf{x}_1),\ f(\mathbf{x}_2),\ f(\mathbf{x}_3),\ \ldots.,\ f(\mathbf{x}_n)\}^T \tag{48}$$

and $\mathbf{R}$ is an $n \times n$ matrix:

$$\mathbf{R} = \begin{bmatrix} R_1(\mathbf{x}_1) & R_2(\mathbf{x}_1) & R_3(\mathbf{x}_1) & \ldots & R_n(\mathbf{x}_1) \\ R_1(\mathbf{x}_2) & R_2(\mathbf{x}_2) & R_3(\mathbf{x}_2) & \ldots & R_n(\mathbf{x}_2) \\ R_1(\mathbf{x}_3) & R_2(\mathbf{x}_3) & R_3(\mathbf{x}_3) & \ldots & R_n(\mathbf{x}_3) \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ R_1(\mathbf{x}_n) & R_2(\mathbf{x}_n) & R_3(\mathbf{x}_n) & \ldots & R_n(\mathbf{x}_n) \end{bmatrix} \tag{49}$$

and $\mathbf{P}$ is an $n \times m$ matrix:

$$\mathbf{P} = \begin{bmatrix} p_1(\mathbf{x}_1) & p_2(\mathbf{x}_1) & p_3(\mathbf{x}_1) & \ldots & p_m(\mathbf{x}_1) \\ p_1(\mathbf{x}_2) & p_2(\mathbf{x}_2) & p_3(\mathbf{x}_2) & \ldots & p_m(\mathbf{x}_2) \\ p_1(\mathbf{x}_3) & p_2(\mathbf{x}_3) & p_3(\mathbf{x}_3) & \ldots & p_m(\mathbf{x}_3) \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ p_1(\mathbf{x}_n) & p_2(\mathbf{x}_n) & p_3(\mathbf{x}_n) & \ldots & p_m(\mathbf{x}_n) \end{bmatrix} \tag{50}$$

since there are $n + m$ unknowns and only n equations, we need to impose the constraint:

$$\mathbf{Pa} = 0 \tag{51}$$

which yields

$$\mathbf{f} = \begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{P} & 0 \end{bmatrix} \{\mathbf{a} \quad \mathbf{b}\}^T = \mathbf{G}\{\mathbf{a} \quad \mathbf{b}\}^T \tag{52}$$

where

$$\mathbf{G} = \begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{P} & 0 \end{bmatrix} \tag{53}$$

To determine the unknowns $\mathbf{a}$ and $\mathbf{b}$ we can re-express Eq. (52) to be

$$\{\mathbf{a} \quad \mathbf{b}\}^T = \mathbf{G}^{-1}\mathbf{f} \tag{54}$$

which can then be substituted back into Eq. (42) to yield:

$$f(\mathbf{x}_i) = \{\mathbf{R}(\mathbf{x}_i) \quad \mathbf{p}(\mathbf{x}_i)\}\mathbf{G}^{-1}\mathbf{f} = \mathbf{N}^{RBF}\mathbf{f} \tag{55}$$

where $\mathbf{N}^{RBF}$ are the set of interpolation functions $\mathbf{N}^{RBF} = \{N_1, N_2, N_3, \ldots, N_n\}$ computed by:

$$\mathbf{N}^{RBF} = \{\mathbf{R}(\mathbf{x}_i) \quad \mathbf{p}(\mathbf{x}_i)\}\mathbf{G}^{-1} \tag{56}$$

First and second order derivatives of the constructed RBF interpolants can be computed by:

$$\frac{\partial \mathbf{N}}{\partial u} = \left\{ \frac{\partial \mathbf{R}(\mathbf{x}_i)}{\partial u} \quad \frac{\partial \mathbf{p}(\mathbf{x}_i)}{\partial u} \right\} \mathbf{G}^{-1} \tag{57}$$

where $\frac{\partial}{\partial u}$ is a partial derivative with respect to $u = \{x, y, z, xx, yy, \ldots\}$.

Therefore, Eq. (4) can be re-expressed to approximate a field variable $f(\mathbf{x}_i)$ at particle i using the computed set of interpolants $\{N^{RBF},\ \nabla N^{RBF}\}$ such that:

$$\hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} f(\mathbf{x}_j) N_{ij}^{RBF} \tag{58}$$

similarly, Eq. (19) is re-expressed as

$$\nabla \hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} \Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) \Big) \nabla N_{ij}^{RBF} \tag{59}$$

and Eq. (22) is re-expressed as

$$\nabla^2 \hat{f}(\mathbf{x}_i) \approx 2 \sum_{j=1}^{n} \frac{\Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) \Big)}{|\mathbf{r}_{ji}|^2} \mathbf{r}_{ji} \cdot \nabla N_{ij}^{RBF} \tag{60}$$

Note that a global matrix system does not need to be computed. Rather, the interpolants and their derivatives are computed by solving a small matrix system whose size is directly related to the number of particles in the support kernel. This allows for easier parallel computing where multiple support kernels can be processed simultaneously and enables us to handle large number of particles easily and efficiently. Since RBF interpolation inherently satisfies the Kronecker-delta condition then applying the Direchlet boundary conditions can be done easily and directly at the required particles.

2.2.2. Constructing interpolation functions using moving least squares

For a particle $\mathbf{x}_i$, a field variable $f(\mathbf{x}_i)$ at $\mathbf{x}_i$ can be approximated using a polynomial with q monomials such that:

$$f(\mathbf{x}_i) = \sum_{q=1}^{m} p_q(\mathbf{x}_i) b_q = \mathbf{p}(\mathbf{x}_i)\mathbf{b}^T \tag{61}$$

where for a given m polynomial basis function:

$$\mathbf{p}(\mathbf{x}_i) = \left\{1, x, y, z, x^2, y^2, z^2, xy, xz, yz, \ldots\right\} \tag{62}$$

and $\mathbf{b}$ are coefficients of the polynomials to be determined where

$$\mathbf{b} = \{b_1,\ b_2,\ b_3,\ \ldots.,\ b_m\} \tag{63}$$

Generally, the number of supporting particles n in any given kernel typically exceeds the number of coefficients m, which leads to an over-determined system of equations. This kind of optimization problem can be solved by using a least squares approach [65]. For all support particles in the kernel, Eq. (61) is re-expressed as:

$$\mathbf{P}^T\mathbf{f} = \mathbf{P}^T\mathbf{P}\mathbf{b}^T \tag{64}$$

where

$$\mathbf{f} = \{f(\mathbf{x}_1),\ f(\mathbf{x}_2),\ f(\mathbf{x}_3),\ \ldots,\ f(\mathbf{x}_n)\}^T \tag{65}$$

and $\mathbf{P}$ is and $n \times m$ matrix expressed as:

$$\mathbf{P} = \begin{bmatrix} p_1(\mathbf{x}_1) & p_2(\mathbf{x}_1) & p_3(\mathbf{x}_1) & \ldots & p_m(\mathbf{x}_1) \\ p_1(\mathbf{x}_2) & p_2(\mathbf{x}_2) & p_3(\mathbf{x}_2) & \ldots & p_m(\mathbf{x}_2) \\ p_1(\mathbf{x}_3) & p_2(\mathbf{x}_3) & p_3(\mathbf{x}_3) & \ldots & p_m(\mathbf{x}_3) \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ p_1(\mathbf{x}_n) & p_2(\mathbf{x}_n) & p_3(\mathbf{x}_n) & \ldots & p_m(\mathbf{x}_n) \end{bmatrix} \tag{66}$$

Let us introduce a weighting function such as the cubic spline function of the form:

$$w_i(\mathbf{x}) = \begin{cases} 2/3 - 4r_i^2 + 4r_i^3 & r_i \le 0.5 \\ 4/3 - 4r_i + 4r_i^2 - 4/3r_i^3 & 0.5 < r_i \le 1 \\ 0 & r_i > 1 \end{cases} \tag{67}$$

or, alternatively, the quartic spline of the form:

$$w_i(\mathbf{x}) = \begin{cases} 1 - 6r_i^2 + 8r_i^3 - 3r_i^4 & r_i \le 1 \\ 0 & r_i > 1 \end{cases} \tag{68}$$

This can be expressed as an $n \times n$ matrix for all support particles in the kernel such that:

$$\mathbf{w} = \begin{bmatrix} w_1(\mathbf{x}_1) & w_1(\mathbf{x}_2) & w_1(\mathbf{x}_3) & \ldots & w_1(\mathbf{x}_n) \\ w_2(\mathbf{x}_1) & w_2(\mathbf{x}_2) & w_2(\mathbf{x}_3) & \ldots & w_2(\mathbf{x}_n) \\ w_3(\mathbf{x}_1) & w_3(\mathbf{x}_2) & w_3(\mathbf{x}_3) & \ldots & w_3(\mathbf{x}_n) \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ . & . & . & \ldots & . \\ w_n(\mathbf{x}_1) & w_n(\mathbf{x}_2) & w_n(\mathbf{x}_3) & \ldots & w_n(\mathbf{x}_n) \end{bmatrix} \tag{69}$$

We can then re-express Eq. (64) to be

$$\mathbf{P}^T\mathbf{w}\mathbf{f} = \mathbf{P}^T\mathbf{w}\mathbf{P}\mathbf{b}^T \tag{70}$$

let

$$\mathbf{A} = \mathbf{P}^T\mathbf{w}\mathbf{P} \tag{71}$$

and

$$\mathbf{B} = \mathbf{P}^T\mathbf{w} \tag{72}$$

then the set of unknown coefficients $\mathbf{b}$ can be obtained from

$$\mathbf{b}^T = \mathbf{B}\mathbf{A}^{-1}\mathbf{f} \tag{73}$$

one can then substitute it back into Eq. (61) to obtain $f(\mathbf{x}_i)$ such that

$$f(\mathbf{x}_i) = \mathbf{p}(\mathbf{x}_i)\mathbf{B}\mathbf{A}^{-1}\mathbf{f} \tag{74}$$

or simply

$$f(\mathbf{x}) = \mathbf{N}^{MLS}\mathbf{f} \tag{75}$$

where $\mathbf{N}^{MLS}$ are the MLS approximation functions $\mathbf{N}^{MLS} = \{N_1, N_2, N_3, \ldots, N_n\}$ computed by:

$$\mathbf{N}^{MLS} = \mathbf{p}(\mathbf{x}_i)\mathbf{B}\mathbf{A}^{-1} \tag{76}$$

let

$$\gamma = \mathbf{p}(\mathbf{x}_i)\mathbf{A}^{-1} \tag{77}$$

$$\mathbf{A}\gamma = \mathbf{p}(\mathbf{x}_i) \tag{78}$$

We follow the technique proposed by Belytschko et al. [66] to compute the partial derivatives of the approximants such that:

$$\mathbf{N}_\alpha = \gamma_\alpha \mathbf{B} + \gamma \mathbf{B}_\alpha \tag{79}$$

$$\mathbf{N}_{\alpha\beta} = \gamma_{\alpha\beta}\mathbf{B} + \gamma_\alpha \mathbf{B}_\beta + \gamma_\beta \mathbf{B}_\alpha + \gamma \mathbf{B}_{\alpha\beta} \tag{80}$$

where subscripts $\alpha = \{x, y, z\}$ and $\beta = \{x, y, z\}$ correspond to the partial derivative with respect to spatial co-ordinates x, y and z.

Eq. (4) can then be re-expressed to approximate a field variable $f(\mathbf{x}_i)$ at particle i using the computed set of interpolants $\{N^{MLS}, \nabla N^{MLS}\}$ such that:

$$\hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} f(\mathbf{x}_j) N_{ij}^{MLS} \tag{81}$$

similarly, Eq. (19) is re-expressed as

$$\nabla \hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} \Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) \Big) \nabla N_{ij}^{MLS} \tag{82}$$

and Eq. (22) is re-expressed as:

$$\nabla^2 \hat{f}(\mathbf{x}_i) \approx 2 \sum_{j=1}^{n} \frac{\Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) \Big)}{|\mathbf{r}_{ji}|^2} \mathbf{r}_{ji} \cdot \nabla N_{ij}^{MLS} \tag{83}$$

Due to the least squares approach, the MLS approximation does not pass through the particle values. This implies that the Kronecker-delta condition is not inherently satisfied. This condition however can be recovered locally by introducing a singular weighing function at the particles as proposed by Lancaster and Salkauskas [65]. Alternatively, a regularized weighting function can be used which recovers the Kronecker-delta condition to a high degree of accuracy [67].

2.2.3. Imposition of neumann boundary conditions

Solving the set of PF equations requires imposing zero-flux Neumann boundary conditions at the domain boundaries. This procedure is very trivial in PF models solved using finite differences and is inherently satisfied in finite element formulations but it is not as straight-forward in strong-form meshfree methods. Since the SPH-PFM is designed to handle arbitrary domain geometries which may even exhibit internal cavities, then a robust and fully automatic procedure is needed.

For a boundary particle $\mathbf{x}_B$ describing a domain boundary B, we construct a virtual particle $\mathbf{x}_v$ positioned at the outside normal vector at $\mathbf{x}_B$. We can then enforce a zero-flux condition $\nabla f(\mathbf{x}_B)|_{\mathbf{n}} = 0$ by enforcing $f(\mathbf{x}_v) = f(\mathbf{x}_B)$. To accomplish this, we recognize that despite the SPH method being a meshfree particle method, a background grid is often used for book-keeping the relative location of particles [68,69]. This background grid/mesh is often used to efficiently determine the set of neighbouring particles present in the kernel support which can reduce the computational costs considerably [70]. In this work, we utilize this background book-keeping mesh, as shown in Fig. 3 for a square with an internal circular cavity, to construct an implicit function, Φ, whose zero-isocontour coincides with $\mathbf{x}_B$. Mathematically, we can express this as:

$$B = \{\mathbf{x} \in \Omega : \Phi(\mathbf{x}) = 0\} \tag{84}$$

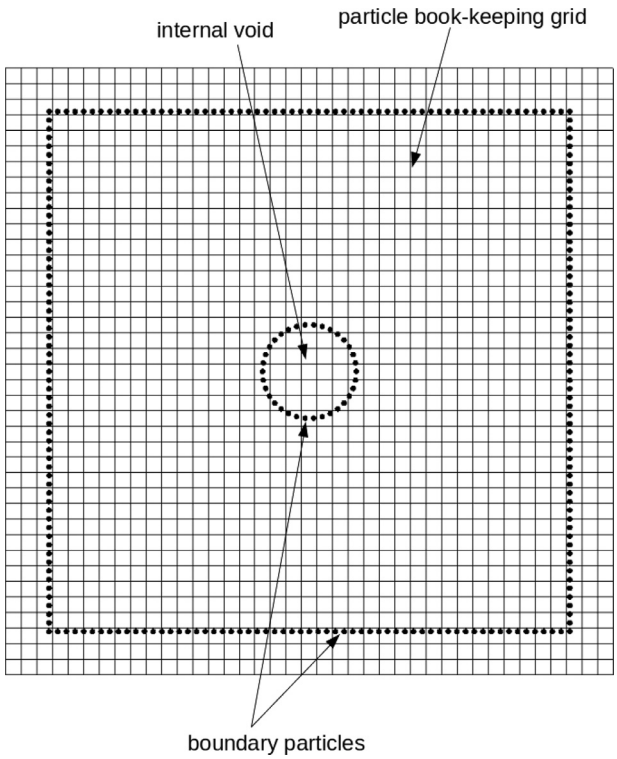

Fig. 3. Boundary particles describing a square with an internal circular cavity imposed on a background grid for particle book-keeping.

For d-dimensional space ($d = \{2, 3\}$), let $\Phi(\mathbf{x})$ be an implicit function constructed in $\mathbb{R}^{d+1}$ for all $\mathbf{x} \in \Omega$ such that it is preferably continuous, smooth and differentiable. The easiest implicit function satisfying these conditions is a signed distance function (SDF) where the distance of $\mathbf{x} \in \Omega$ to domain boundary B is expressed as:

$$\phi(\mathbf{x}) = min|\mathbf{x} - \mathbf{x}_B| \tag{85}$$

where $\mathbf{x}_B \in B$, and SDF is expressed as:

$$\Phi(\mathbf{x}) = \begin{cases} \phi(\mathbf{x}) & \forall \quad \mathbf{x} \in \Omega_{out} \\ 0 & \forall \quad \mathbf{x} \in B \\ -\phi(\mathbf{x}) & \forall \quad \mathbf{x} \in \Omega_{in} \end{cases} \tag{86}$$

This implicit function is then used to construct virtual particles positioned at the outside normal vector direction to the boundary particles, as illustrated in Fig. 4, and used to impose the Neumann boundary conditions. We note that since the domain boundary particles are fixed, this procedure is done only once during pre-processing and is fully automated. We first collect the set of background grid elements which contain at least one boundary particle. We call these elements "boundary-embedded" elements. For each vertex $\mathbf{x}_e$ of the boundary-embedded element, we determine the closest boundary particle and construct a distance function $\phi(\mathbf{x}_e)$. Note that since we do not need to define the distance function everywhere in the domain, computing $\phi(\mathbf{x}_e)$ is only required at the grid vertices located at the vicinity of the domain boundary. To satisfy (86) we need to "sign" the computed $\phi(\mathbf{x}_e)$. To do this, we first assume that the un-signed $\phi(\mathbf{x}_e)$ is equal to the signed $\Phi(\mathbf{x}_e)$ and compute $\nabla\phi(\mathbf{x}_e) = \nabla\Phi(\mathbf{x}_e)$ using RBF interpolation. We then construct two virtual particles located inside and outside of the domain boundary, respectively, and positioned at a small distance, δ_1, normal to the set of boundary particles $\mathbf{x}_B$ such that:

$$\mathbf{x}_{\Omega_{out}} = \mathbf{x}_B + \delta_1 \frac{\nabla\Phi}{|\nabla\Phi|} \tag{87}$$

$$\mathbf{x}_{\Omega_{in}} = \mathbf{x}_B - \delta_1 \frac{\nabla\Phi}{|\nabla\Phi|} \tag{88}$$

Now the distance between $\mathbf{x}_e$ and the two inside/outside normal particles ($\mathbf{x}_{\Omega_{in}}, \mathbf{x}_{\Omega_{out}}$) are determined. If the $\mathbf{x}_e$ is closer to $\mathbf{x}_{\Omega_{in}}$ then $\mathbf{x}_e$ is inside the domain and $\Phi(\mathbf{x}_e)$ is then corrected to be $\Phi(\mathbf{x}_e) = -\phi(\mathbf{x}_e)$ and it is $\Phi(\mathbf{x}_e) = \phi(\mathbf{x}_e)$ otherwise. This procedure of determining $\Phi(\mathbf{x}_e)$ locally provides information about the orientation of the boundary particles. Constructing

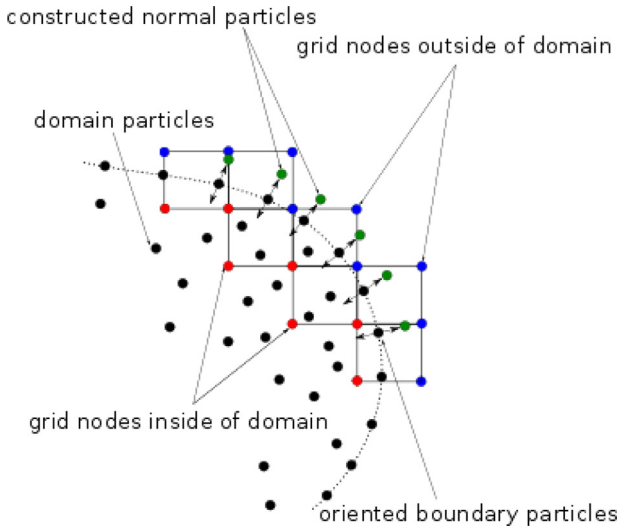

Fig. 4. Illustration of constructing inside/outside particles at domain boundary particles.

normal virtual particles outside the domain can then be done trivially using the signed $\Phi(\mathbf{x}_e)$ by:

$$\mathbf{x}_v = \mathbf{x}_B + \delta_2 \frac{\nabla \Phi}{|\nabla \Phi|} \tag{89}$$

where δ_2 is taken here to be the average Euclidean distance between the particles present in the kernel. If the domain geometry has no internal cavities, then one can alternatively use the distance function $\phi(\mathbf{x}_e)$ directly to construct the normal particles. However, if internal cavities exist, then determining a signed $\Phi(\mathbf{x}_e)$ is essential in determining the correct direction of the normal particles. To illustrate this, consider the domain geometry with an internal cavity as shown in Fig. 3. Using the computed distance function directly without signing it first, the normal particles at the boundaries of the square were correctly positioned outside the domain. However, they are incorrectly generated at the cavity where they are position inside of the domain rather than outside as shown in Fig. 5. On the other hand, signing the distance function resulted in correctly generating normal particles positioned outside of the domain regardless of the presence of the internal cavity as shown in Fig. 6 due to the correct computed orientation of the boundary particles. This shows that computing a signed distance function is essential in correctly generating outside normal virtual particles even in the presence of internal cavities. Once the set of normal particles are generated for each boundary particle during pre-processing, they are appended to the list of domain particles and are then used throughout the simulations to impose the zero-flux boundary conditions in a similar manner to that done in the finite difference method. For a zero-flux boundary condition, every constructed virtual normal particle is assigned its parent boundary particle's field variable such that:

$$f(\mathbf{x}_v)|_{\mathbf{n}} = f(\mathbf{x}_B) \tag{90}$$

yielding

$$\nabla f(\mathbf{x}_B)|_{\mathbf{n}} = 0 \tag{91}$$

3. Implementation and computational algorithm

The presented numerical analysis is based on a number of assumptions for simulating dendritic growth which can be summarized as follows:

1. the liquid–solid interface is assumed to be a diffuse region with a finite thickness;
2. the liquid phase is assumed to be of a pure melt;
3. the governing Eqs. (29) and (31) assume the liquid phase to be stationary;
4. thermal conductivity in the liquid and solid phases are assumed to be constant;
5. domain boundaries and internal cavities or obstacles are assumed to be perfectly insulated.

We first initialize the system by generating particles which describe the domain geometry. A convenient and practical approach to accomplish this during pre-processing is to utilize the nodal distribution of the meshed geometry as the initial distribution of the fixed domain particles. This mesh information is used later during post-processing for visualization of the results for convenience and clarity. The initial liquid and solid regions are assigned at the domain particles. Note that at this stage, the particle resolution may be larger than that of the interface thickness such that it is not properly resolved. For each

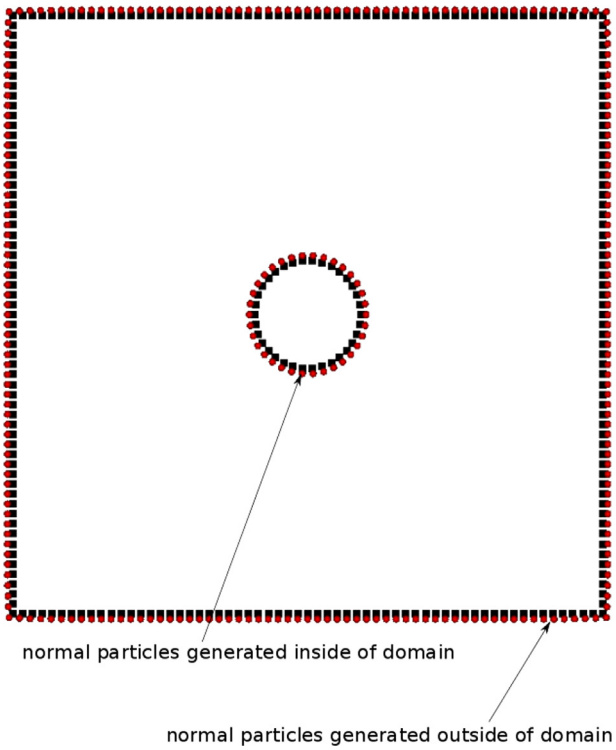

Fig. 5. Using an unsigned distance function yielded the correct normal particles (red spheres) positioned outside of the domain boundary. However, the normal particles at the internal cavity are positioned incorrectly inside the domain. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

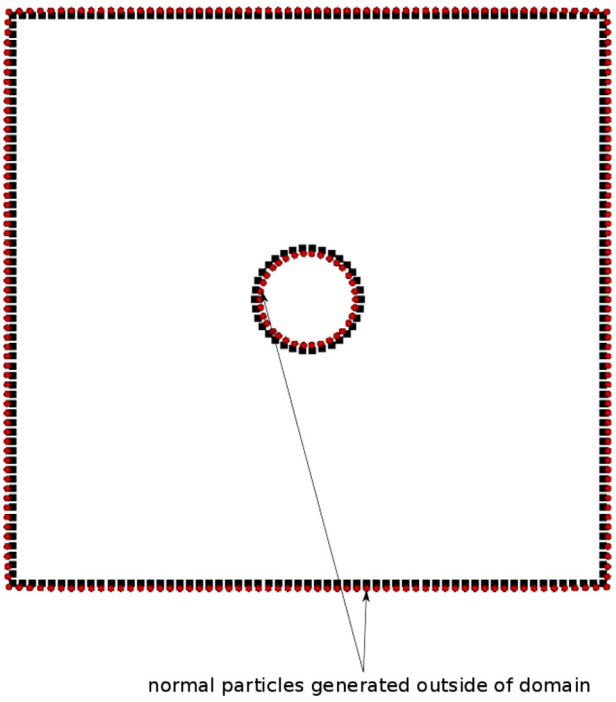

Fig. 6. Using a signed distance function yielded the correct normal particles (red spheres) positioned outside of the domain boundary and the internal cavity. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

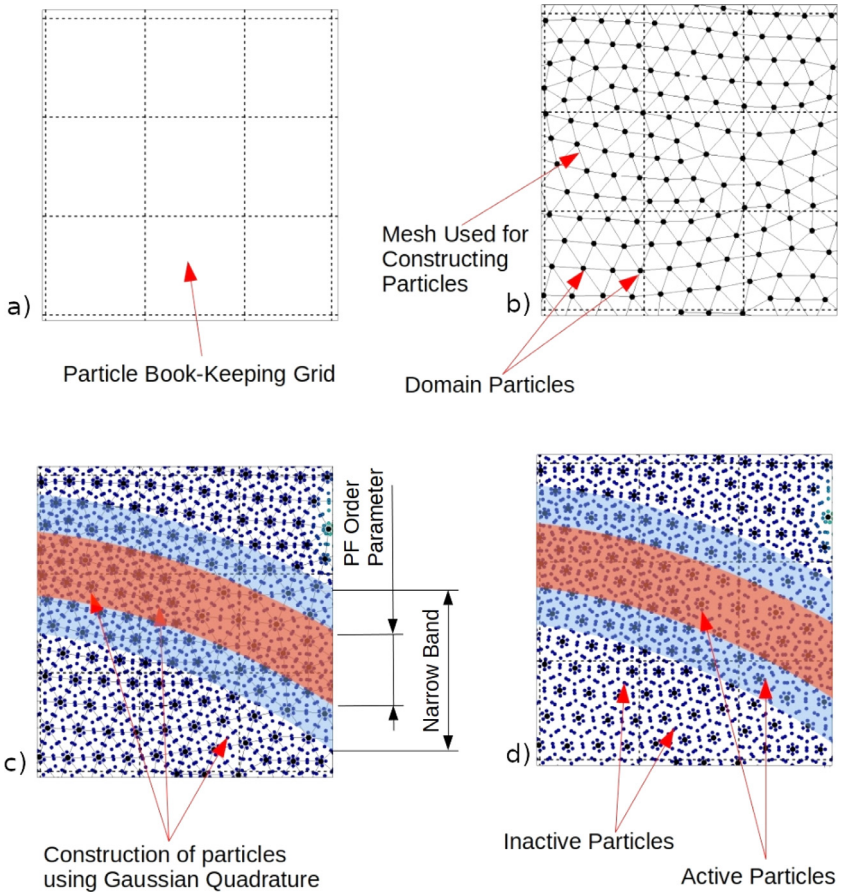

Fig. 7. Strategy for constructing domain particles and narrow band active particles.

domain boundary particle we construct a virtual particle positioned at its outside normal vector. These virtual particles are appended to the set of domain particles and each is associated with its parent domain boundary particle. We then compute the meshfree interpolation functions and their derivatives at the domain particles everywhere in the domain. Since the set of domain particles are fixed in space, this is done only once as part of the pre-processing stage.

As mentioned earlier, since the fixed domain particles may not adquately resolve the interface thickness, a narrow band of a user-specified width is constructed around the interface. In essence, this is analogous to adaptive mesh refinement around the interface region but without the need to use any meshing algorithms, such as Delaunay triangulation for example, and without the need to know the explicit connectivity between particles. Only their position in space relative to the book-keeping grid is needed to determine the neighbouring particles within a kernel radius. In this work, the narrow band is taken as 3δ where δ is the average spacing between the domain particles. Choosing the narrow band width to be larger than the resolution of the domain particles guarantees that some fixed domain particles lie within the narrow band which are used to transfer data between the domain particles and the narrow band particles. Now, there are multiple approaches to construct particles at the narrow band. For example, Feldman and Bonet [71] proposed a dynamic particle refinement algorithm for SPH by replacing an SPH particle with smaller daughter particles positioned using a square pattern centered at the position of the refined particle.

An alternative approach is used in this work. As we mentioned earlier, for practical purposes, a mesh is used to conveniently descretize the domain geometry and to determine the initial distribution of the domain particles. This mesh is also used during post-processing to render the results on a surface for convinient visualization rather than on coloured particles. We note that Gaussian quadrature is a standard procedure in finite elements [72] where quadrature points are typically generated for unsctructured meshes describing arbitrary domain geometries. Therefore, for purposes of particle refinement in this work, we take advantage of the fact that the domain particles are fixed in space and utilize the mesh used for constructing the domain particles to also construct the narrow band particles where Gaussian quadrature points are generated and are used as refinement particles within the narrow band as illustrated in Fig. 7. To clarify, during the pre-processing stage, for a relatively coarse book-keeping grid shown in Fig. 7(a), we use the vertices of an overlayed secondary mesh of triangular elements as domain particles (Fig. 7(b)). Quadrature points are generated using the triangular elements

(Fig. 7(c)) and are used as refinement particles which are all set to an initial state of "inactive". During the processing stage, a narrow band is constructed around the diffuse interface and all refinement particles that are within the narrow band are activated and appended to the set of narrow band particles (Fig. 7(d)). The main advantage of this approach is that the constructed refinement particles inherently exhibit good distribution relative to the domain particles regardless of the number of the constructed quadrature points or the structure of the bounding mesh element. It also remains faithful to the meshfree nature of the solution since knowing the explicit connectivity between the constructed particles is not required. We use the coarse particle book-keeping grid to determine the neighbouring particles within a support kernel. During the pre-processing stage, Gaussian quadrature points are constructed for all mesh elements and are used as refinement particles. As the interface and the narrow band evolves, the once active refinement particles that now lie outside the scope of the narrow band are reset to an "inactive" state and are deleted from the set of narrow band particles.

A summary of the computational algorithm during the processing stage can be summarized as follows:

1. construct a narrow band with a user specified width around the liquid–solid interface region. The width of the narrow band must be larger than the interface thickness;
2. the set of refinement particles that lie within the narrow band are now set to an active state; All other refinement particles remain inactive.
3. interpolate the thermal distribution at the set of active refinement particles using the temperature distribution at the original domain particles that are within the narrow band region;
4. solve the PF equation at the set of particles in the narrow band using the interpolated temperature distribution computed in step 3;
5. using the new PF solution computed at the domain particles that are within the narrow band, compute a new thermal field distribution at the domain particles using the original interpolation functions computed during pre-processing. The PF order parameter for domain particles that are outside the narrow band remains unchanged from the previous time step to be either 0 for liquid or 1 for solid.

As can be observed from the algorithm, the computed PF order parameter information computed at the narrow band particles need to be transferred to the lower-resolution domain particles to compute the thermal distribution. Rather than re-interpolate the computed PF parameter at the narrow band particles to the coarser domain particles to compute the thermal field which may over-smooth (smear) the PF order parameter during this process, we directly use the PF parameter values computed at domain particles that lie inside the narrow band. In other words, the domain particles that end up within the narrow band are considered as "shared particles" between the scale of narrow band particles and the scale of the domain particles and are used to transfer data between both scales. However, this implies that the thickness of the narrow band must not be smaller than the resolution of the domain particles at the interface region to ensure that sufficient domain particles are bounded within the narrow band. Since the narrow band chosen in this work is typically about 3-6 times wider than the thickness of the interface and since the interface can not jump by this magnitude within a given time-step and still yield a stable solution, then domain particles outside of the narrow band are considered particles that are far away from the interface and are assigned the same PF order parameter of either 0 or 1 as that computed at the previous time step. As the interface evolves, the narrow band is updated and steps 1 to 5 are repeated. Also, since the narrow band is wider than the interface thickness, then reconstructing it at each time step is not necessary. As such, the constructed interfacial interpolation functions can be reused again for the next time step. Only when the particles near the boundary of the narrow band have an order parameter that is $0 < \Psi < 1$ do we start to construct a new narrow band region.

The presented algorithm has a number of advantages:

1. calculations can be done on arbitrary domain geometries with or without internal cavities;
2. a narrow band is constructed at the interface region where the nodal resolution at the interface region is independent of that used throughout the domain. This is done despite using a fixed set of domain particles;
3. the PF equation is solved only at particles within the narrow band leading to more computational savings;
4. solving a large global system of equations is not required. The PDEs are solved at the kernel level only allowing for simpler parallelization and easier handling of larger domains relative to the interface thickness;
5. computing the interpolants in the majority of the domain is done only once during pre-processing. At any given time step, new interpolants are computed only at the interface region as required at particles within the narrow band;
6. computing second order derivatives of the meshfree interpolants is not required leading to a more efficient and stable solution.
7. automatic construction of normal virtual particles allows for trivial handling of Neumann-type boundary conditions in a similar manner as in conventional finite differencing but adapted to work on arbitrary domain geometries with or without internal cavities.

4. Results and discussions

4.1. Investigation of RBFs and MLS performance as implemented in the SPH–PFM for approximating exact solutions

Before we demonstrate the use of RBFs and MLS for PF modelling of dendritic solidification, we first investigate their performance as implemented in the SPH–PFM for approximating the exact non-trivial peaks function and its first-order

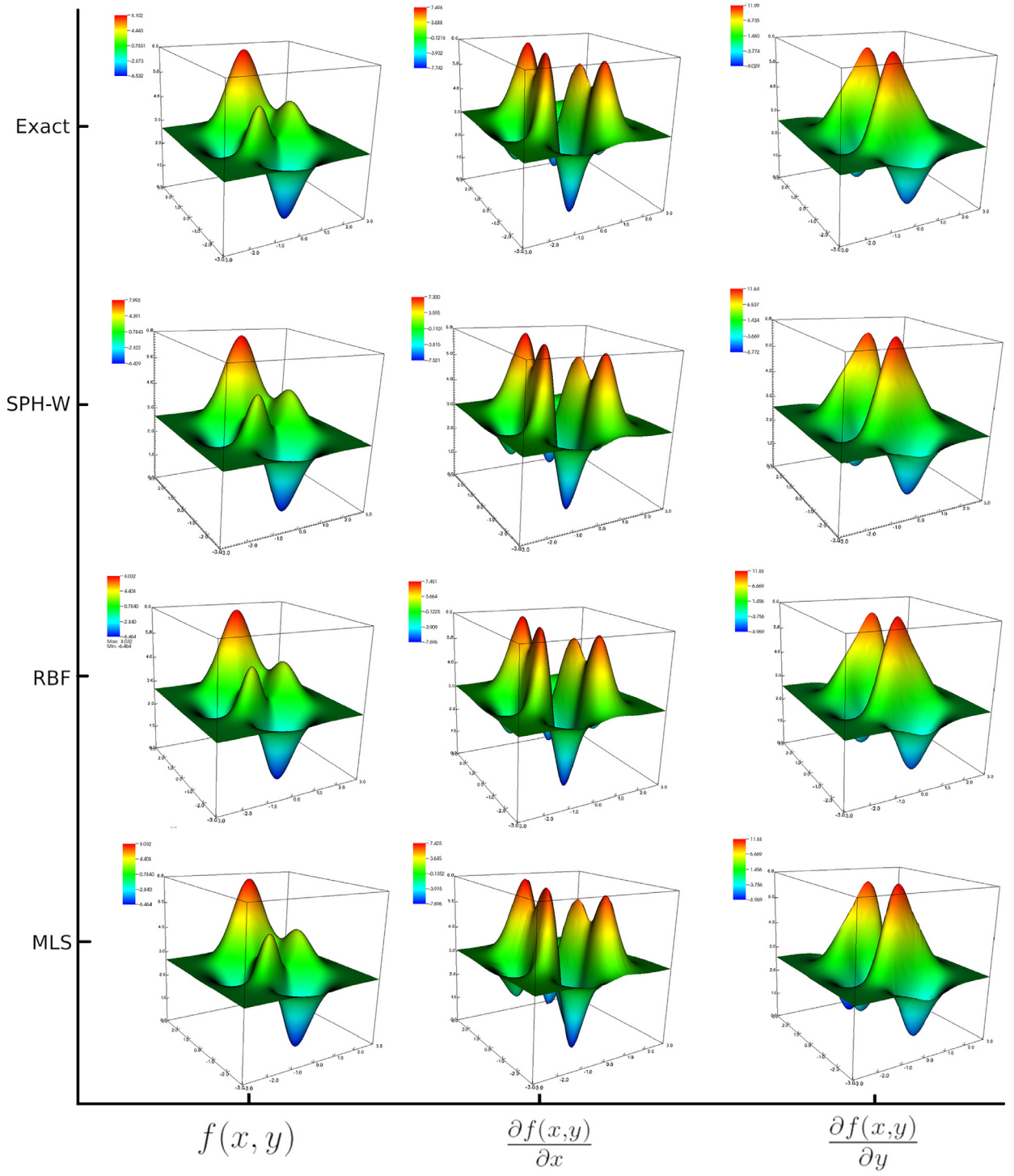

Fig. 8. Approximation of the peaks function and its first-order derivatives using SPH with Wendland's kernel weight function, RBFs and MLS.

derivatives. The peaks function is used as a benchmark because it exhibits steep gradients that are often difficult to approximate adequately. As such, it serves as a good benchmark to compute the relative error of the utilized numerical method. The peaks function is expressed as:

$$f(x,y) = 3(1-x)^2 e^{(-x^2-(y+1)^2)} - 10\left(\frac{x}{5} - x^3 - y^5\right)e^{(-x^2-y^2)} - \frac{1}{3}e^{(-(x+1)^2-y^2)} \tag{92}$$

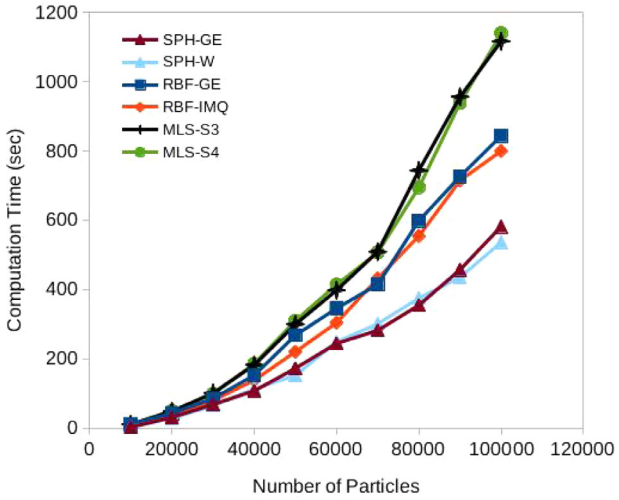

Fig. 9. Comparison of the computation times for approximating the peaks function and its derivatives using SPH with RBFs and MLS.

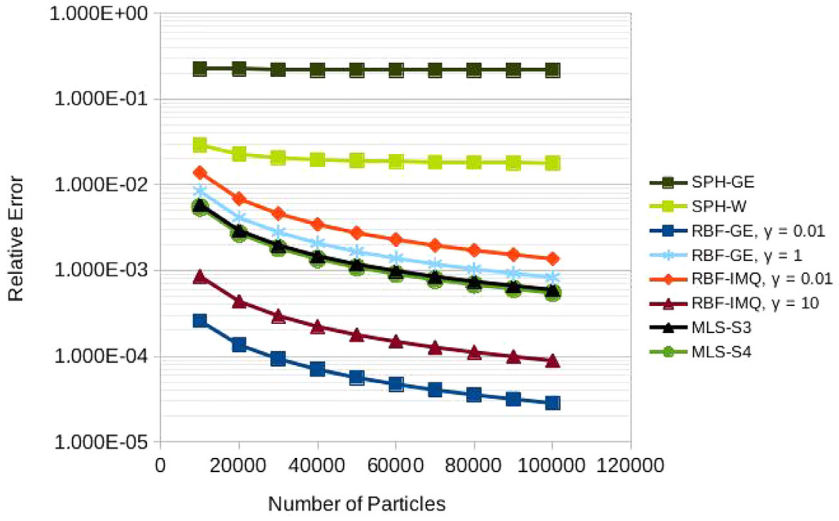

Fig. 10. Relative error for approximating $\frac{\partial f(x,y)}{\partial x}$ of the peaks function using SPH with RBFs and MLS.

Its first-order derivative relative to the x-axis is expressed as:

$$\frac{\partial f}{\partial x} = -6((1-x)^2 x e^{(-(y+1)^2-x^2)} - 6(1-x)e^{(-(y+1)^2-x^2)} + (2/3)(x+1)e^{(-y^2-(x+1)^2)} + 20x(-(y^5-(x^3+x/5)e^{(-y^2-x^2)} - 10(0.2-3x^2)e^{(-y^2-x^2)} \quad (93)$$

while its first-order derivative relative to the y-axis is expressed as:

$$\frac{\partial f}{\partial y} = -6(1-x)^2(y+1)e^{(-(y+1)^2-x^2)} + (2/3)ye^{(-y^2-(x+1)^2)} + 20y(-y^5-x^3+x/5) * e^{(-y^2-x^2)} + 50y^4 e^{(-y^2-x^2)} \quad (94)$$

We use the traditional Gaussian exponential SPH kernel function (SPH-GE) and the fourth-order Wendland SPH kernel function (SPH-W). Throughout this work, we use the normalized version of the traditional SPH kernel weight functions which, as mentioned earlier, only exhibits zero-order consistency. For RBFs, we use Gaussian exponential RBFs (RBF-GE), inverse multi-quadrics RBFs (RBF-IMQ). For MLS, we use cubic spline MLS (MLS-S3) and quartic spline MLS (MLS-S4) for constructing the interpolation functions. A 2D computational domain of $[-3,3]\times\ [-3,3]$ is selected and a structured distribution of particles is utilized. The approximated peaks function and its derivatives are shown in Fig. 8 where we see an excellent agreement between the exact and approximated solutions using the traditional SPH kernel functions, RBF and MLS. The solution appears to be smooth and free from instabilities. For a more detailed quantitative analysis, the relative error of the

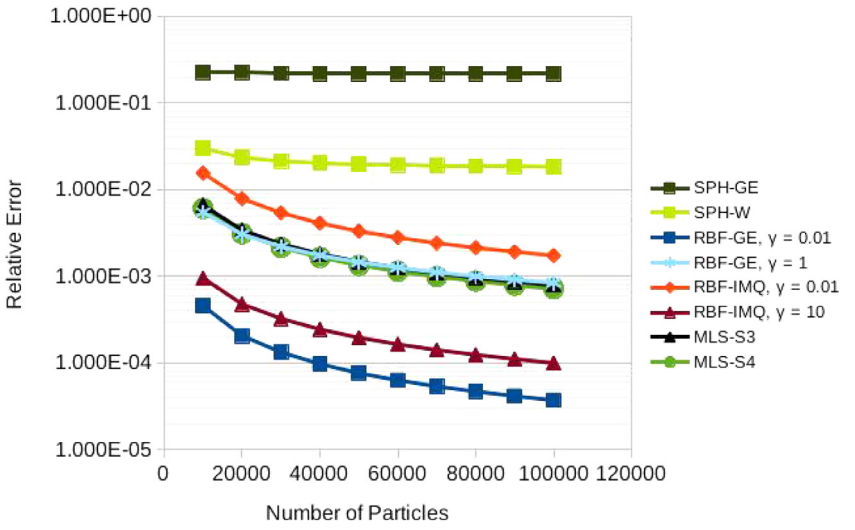

Fig. 11. Relative error for approximating $\frac{\partial f(x,y)}{\partial y}$ of the peaks function using SPH with RBFs and MLS.

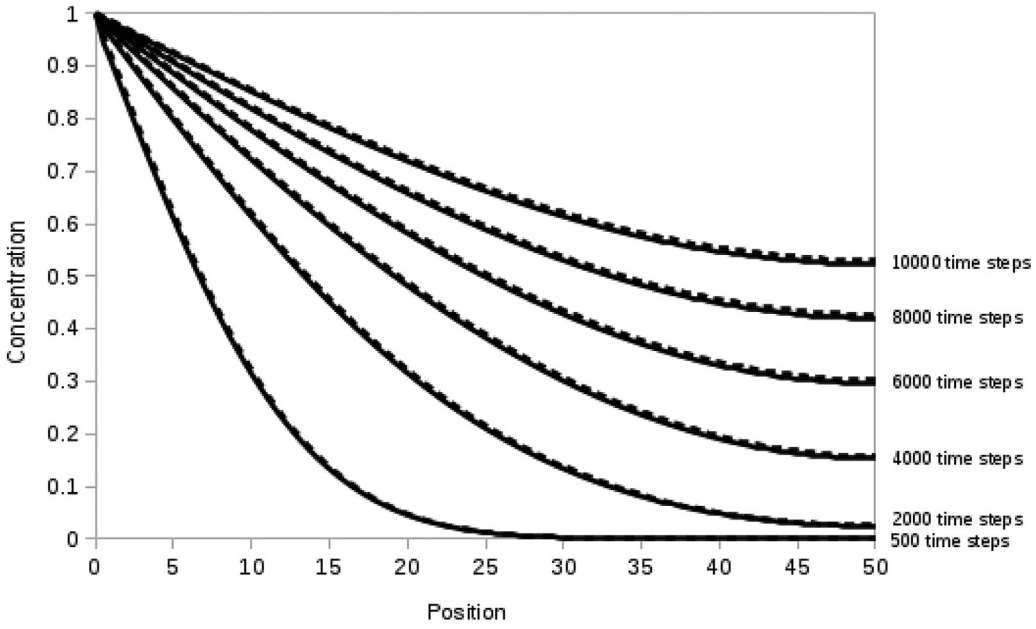

Fig. 12. Distribution of field variable f (concentration) in a 1D bar at different simulation times. Solid lines are the SPH solutions obtained using RBF, while the dashed lines are solutions obtained using conventional finite differencing.

approximated solution is computed by:

$$e_r = \sqrt{\frac{\sum_{i=1}^{N}\left(f_{exact}(\mathbf{x}_i) - f_{approx}(\mathbf{x}_i)\right)^2}{\sum_{i=1}^{N}\left(f_{exact}(\mathbf{x}_i)\right)^2}} \tag{95}$$

The computational cost for constructing the weight functions using traditional SPH kernel functions, RBFs and MLS as the number of particles is increased is shown in Fig. 9 where we see that traditional SPH kernel functions are notably cheaper than that using both RBF or MLS. This is expected since its implementation is simpler and does not require solving a local system of equations. RBF–IMQ and RBF–GE have relatively comparable processing times. Similarly, MLS-S3 and MLS-S4 also exhibited relatively comparable processing times. MLS was found to be relatively more computationally expensive than RBFs, most likely due to the more intensive procedure for computing the spatial derivatives of the approximation functions. The relative error for computing $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ approximations obtained using traditional SPH kernels, RBFs and MLS is shown in Figs. 10 and 11, respectively. We find that traditional SPH kernel functions resulted in notably poorer accuracy than RBFs

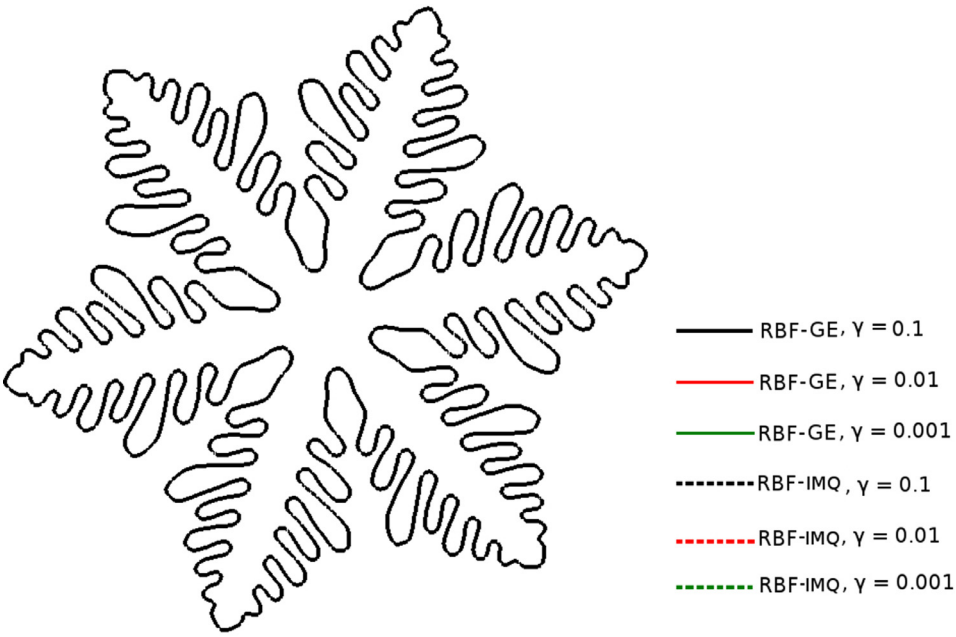

Fig. 13. The 0.5 isocontour of the PF solution for 6-branch dendritic growth using the SPH–PFM with RBF–GE and RBF–IMQ.

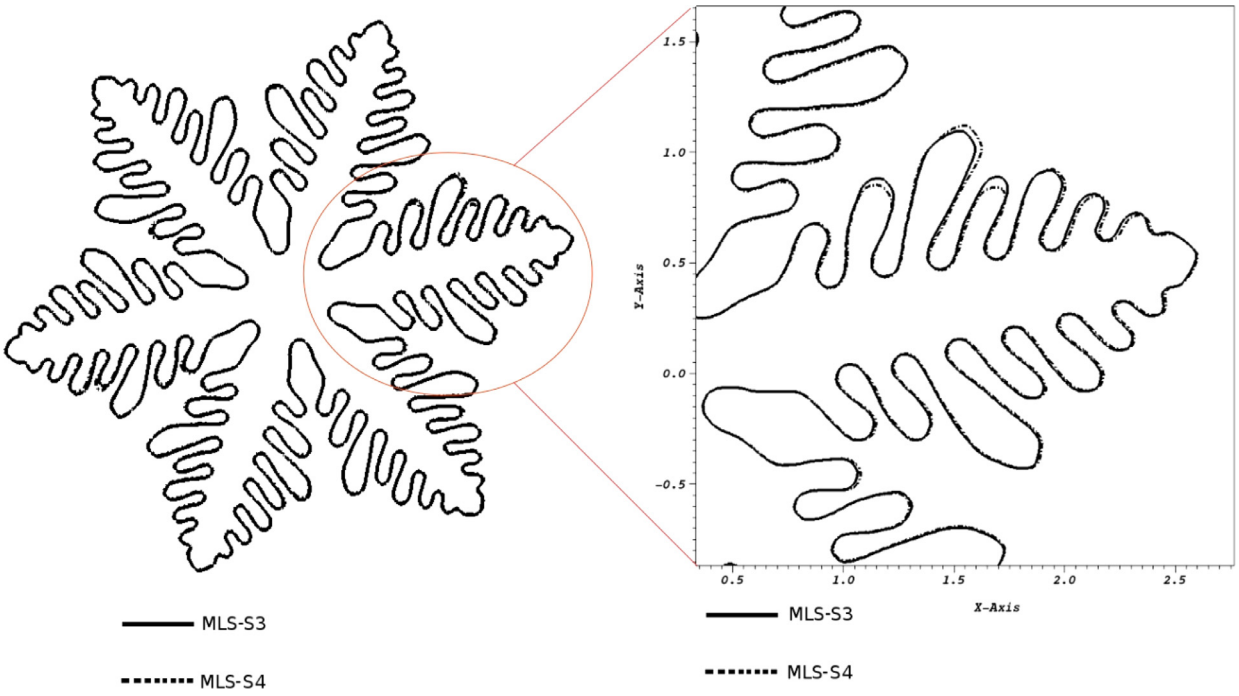

Fig. 14. The 0.5 isocontour of the PF solution for 6-branch dendritic growth using the SPH–PFM with MLS-S3 and MLS-S4.

and MLS. Also, poor convergence is observed when the Gaussian exponental kernel function is used where no significant reduction in the relative error is observed as the number of particles is increased. Using Wendland kernel functions improved both accuracy and convergence but the relative error was still higher than that obtained using RBFs and MLS. Both cubic spline MLS (MLS-S3) and Quartic spline MLS (MLS-S4) exhibited a comparable relative error where there was no significant decrease in the error when the order of spline is increased from cubic to quartic. For RBF–IMQ, increasing the shape parameter from $\gamma = 0.01$ to $\gamma = 10$ yielded a significant reduction in the relative error by a factor of about 10. Similarly, for RBF-GE, decreasing the shape parameter from $\gamma = 1$ to $\gamma = 0.01$ also resulted in notable reduction of the relative error by a

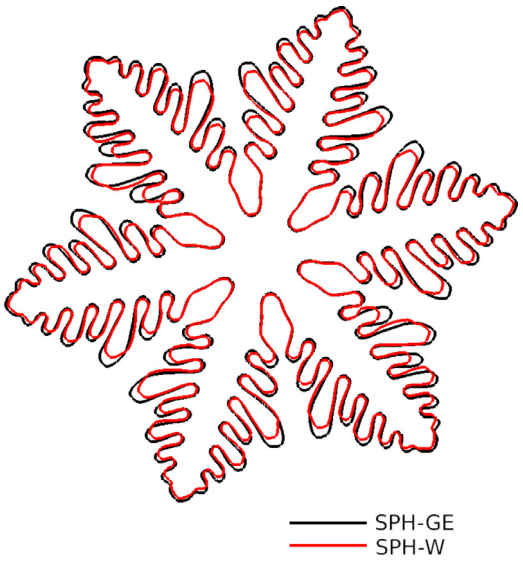

Fig. 15. The 0.5 isocontour of the PF solution for 6-branch dendritic growth using the SPH–PFM with traditional SPH kernel weight functions. Black lines correspond to the Gaussian kernel weight weight functions while the red lines correpond to the Wendland kernel weight function. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

factor of about 10. Using RBF–GE with a shape parameter of $\gamma = 0.01$ yielded the smallest relative error of all the methods examined. The results are also consistent for both $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ approximations. This behaviour is expected since decreasing the shape parameter for RBF–GE leads to a flatter shape function which has been found to increase the accuracy of approximation to a certain extent [73–75]. Wang and Liu [76] reported that the optimal shape parameter for RBF-GE is between 0.003-0.03 for most engineering type problems. This is also consistent with the shape parameter value used in [15–17].

We use the methodology of SPH for solving a parabolic equation on a 1D bar of length L. The equation is expressed as:

$$\frac{\partial f}{\partial t} = D\nabla^2 f \tag{96}$$

With the following boundary conditions:

$$f = 1 \quad x = 0, t >= 0 \tag{97}$$

$$\frac{\partial f}{\partial x} = 0 \quad x = L, t >= 0 \tag{98}$$

Where f is a field variable (such as concentration) and t is time. Let the initial concentration f in the domain to be zero, $L = 50$ descretized by $\Delta x = 0.5$ and let the time step be $\Delta t = 1$. Using RBF–GE with a shape parameter of $\gamma = 0.01$, SPH is used to approximate the distribution of f in the bar where we used a backward Euler time stepping scheme for simplicity. We also solved the parabolic equation using simple central finite differencing in space, and an explicit backward Euler in time to compare with the obtained SPH solution. As shown in Fig. 12, we see an excellent agreement between the distribution of f obtained at different time steps computed using SPH with RBFs and the finite difference method. This confirms earlier results in [57–60] which shows that using Eq. (22) for approximating diffusion processes is indeed an attractive alternative which does not require the computation of the second-order derivatives of weight functions. It has been shown by Brookshaw, that for structured distribution of particles and choosing an appropriate kernel weight function, Eq. (22) reduces to the typical central-differencing equation of the FDM [58].

4.2. Investigation of the interpolation scheme used in SPH–PFM on dendrite morphology

There is currently a significant lack of numerical data available in the literature that investigates the effect of mesh-free interpolation parameters on the resulting dendrite morphology and growth rate. For example, as mentioned earlier, constructing interpolation functions using RBF–GE and RBF–IMQ require defining a shape parameter, γ which affects the accuracy of interpolation. It is however unclear how the choice of the shape function affects the final PF solution and the morphology of the dendrite. Therefore, in this section we investigate the effect of varying γ on the final morphology of the grown dendrite. To do this we used the RBF–GE and RBF–IMQ for constructing the weight functions where the shape parameter γ is varied between 0.001 and 0.1. PF simulations are carried out at an undercooling of $\Delta = 1$ using $\tau = 0.0003$, $d_o = 0.01$, $A = 0.02$, $\beta = 6$, $\theta_o = 0^o$, $K = 1.8$, $\alpha = 0.9$, $\gamma_o = 9.0$, and $\Delta t = 0.00005$. After 6000 time steps, the

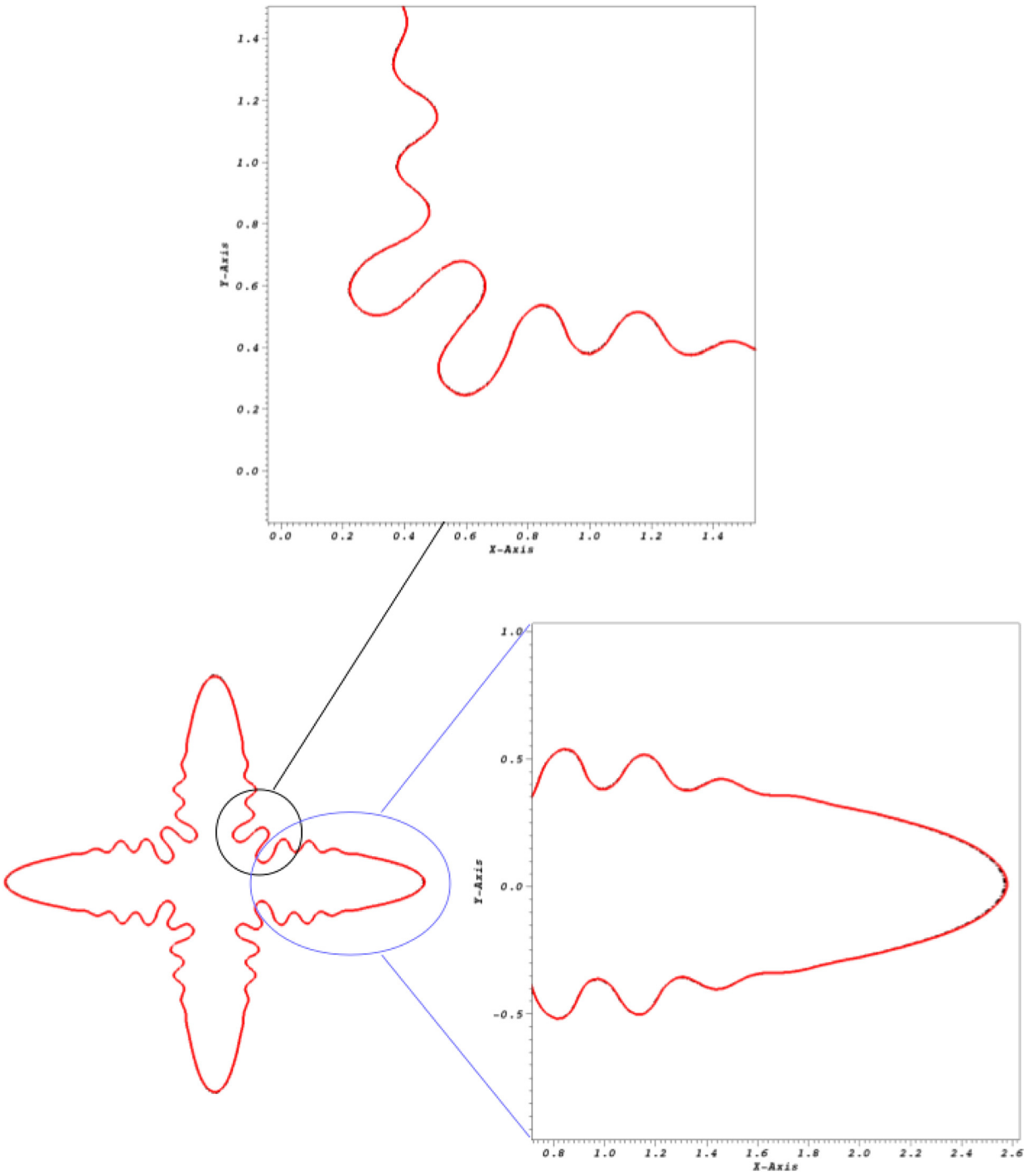

Fig. 16. The 0.5 isocontour of the PF solution for 4-branch dendritic growth using the SPH–PFM with RBF–GE (dashed black line) and MLS-S4 (solid red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

0.5 isocontour of the PF order parameter is obtained and plotted for all cases. When RBFs were used as can be seen from Fig. 13, varying the shape parameter for both RBF–GE and RBF–IMQ did not have a notable effect on the growth rate of the dendritic morphology where the results of all cases were in very close agreement. We then compared MLS-S3 and MLS-S4 for constructing the interpolation functions instead of RBFs. Note that, in contrast to RBFs, MLS does not require defining a shape parameter. Using the same simulation parameters noted earlier, the 0.5 isocontour of the PF order parameter is obtained and plotted for both cases as shown in Fig. 14 where we see again an excellent agreement between both

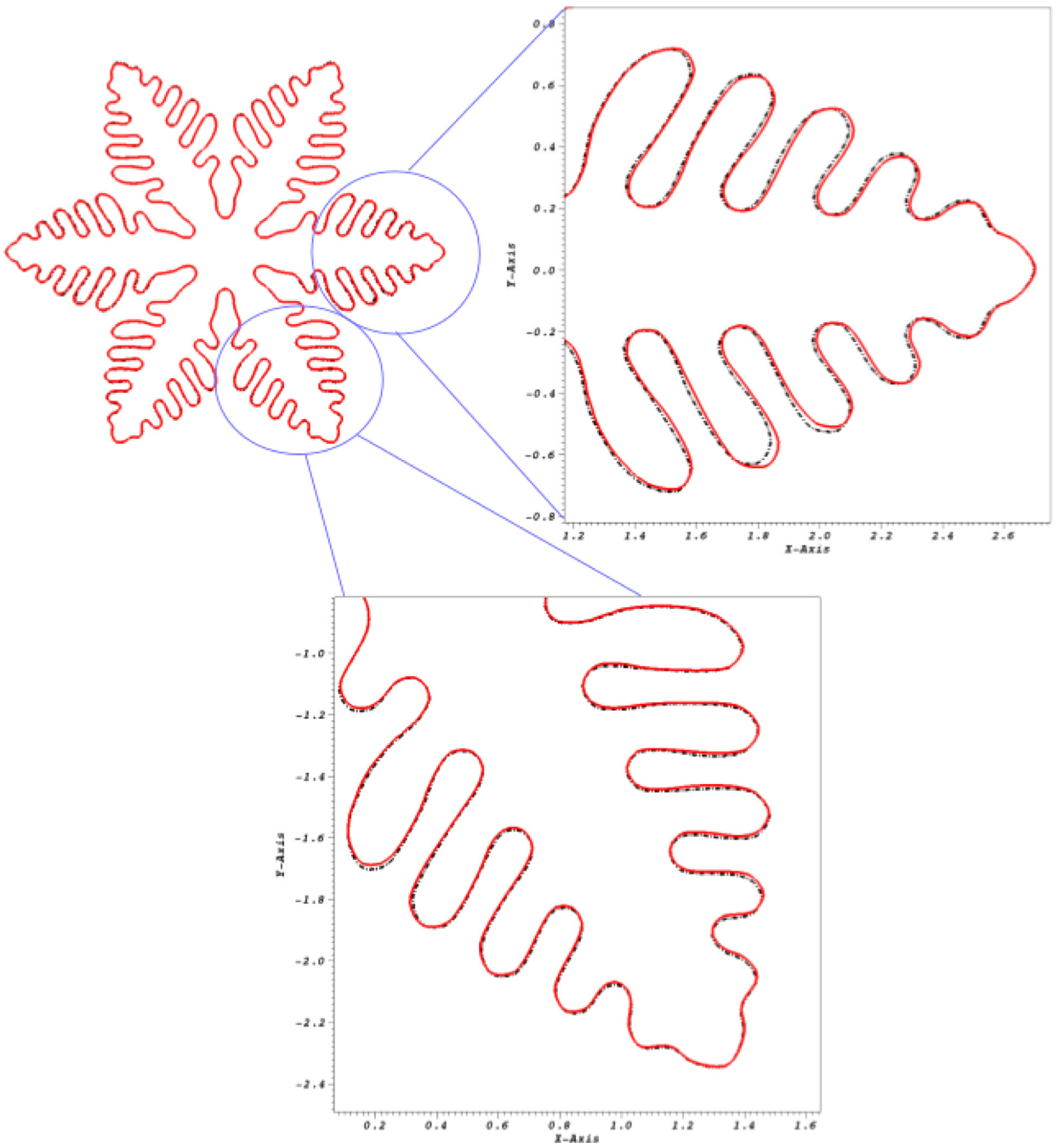

Fig. 17. The 0.5 isocontour of the PF solution 6-branch dendritic growth using the SPH–PFM with RBF–GE (dashed black line) and MLS-S4 (solid red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

dendritic solutions. A magnification of the solution at the dendrite branches shows only a very slight variation between both solutions. Investigations were then done where we compared the solution obtained using SPH-GE and SPH-W for a 6-branch dendrite where we find both solutions to exhibit good qualitative agreement as shown in Fig. 15 but with some variation in the grown dendrite branches which indicates a slight variation in the computed gradients which affects the interfacial migration rates. This is most likely due to the poorer accuracy observed using SPH-GE relative to the SPH-W kernel weight functions. When the SPH-W, RBF–GE and MLS-S4 are compared, we find that the RBF and MLS yielded solutions in excellent agreement which was found to be consistent for both 4-branch and 6-branch dendritic growth as shown in Figs. 16 and 17, respectively. In contrast, while SPH-W yielded dendritic branches in good qualitiative agreement

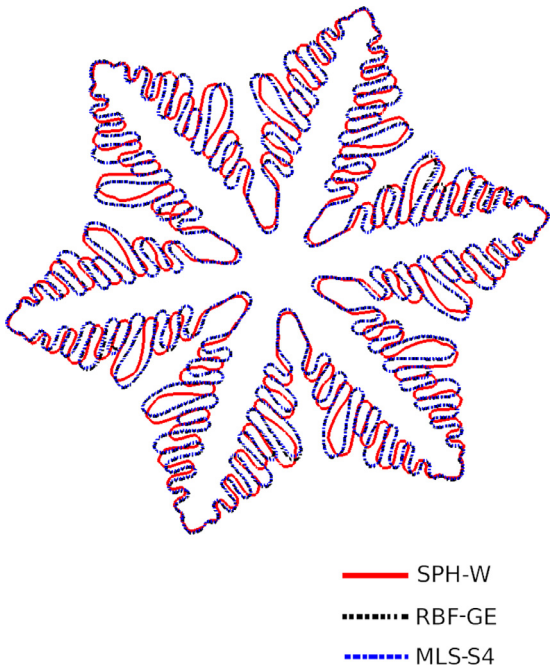

Fig. 18. The 0.5 isocontour of the PF solution for 6-branch dendritic growth using the SPH–PFM. Red lines correspond to the traditional Wendland kernel weight function (SPH-W) while the dashed black and blue lines correspond to the RBF–GE and MLS-S4, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

with both RBF and MLS as can be seen in Fig. 18, there was more pronounced variation in the obtained solution most likely due to the poorer accuracy in computing gradients relative to RBF and MLS.

Three interesting points can be deduced from the obtained results. The dependence of RBFs on selection of a shape parameter has been generally noted in the literature unfavorably as a drawback for using RBFs. This is because it is often unclear how the choice of the shape parameter affects the accuracy of final numerical solution. However, as shown in the present work, the meshfree PF solution obtained using the SPH–PFM does not appear to be sensitive to the selected shape parameter at least within the range examined in this work. Another interesting point, is that despite MLS being only approximants (not interpolants), it is found that they can be used effectively in the SPH–PFM since imposition of Direchlet-type boundary conditions at the interface was not necessary in the present PF simulations. Thirdly, despite the rather poor accuracy of traditional SPH kernel functions as observed in the previous section, a reasonable dendritic morphology that is in good agreement with that obtained using RBF and MLS is observed. Given the computational effiency in computing the traditional SPH kernel functions relative to MLS and RBFs, they may be an attractive option if a quick qualitative solution is needed and numerical accuracy is not the main concern.

4.3. Investigating the effect of particle distribution on the SPH–PFM solution of dendritic solidification

Artificial anisotropy or grid-induced anisotropy occurs when the morphology of the grown dendrite changes by merely changing the orientation of underlying grid. This phenomenon exists when structured grids are used. To demonstrate this behaviour using finite differences for solving the PF equations, we consider a 2D square domain of size $[0, 6] \times [0, 6]$ and a grid density of 300×300. A solid seed is positioned at the center of the domain. Numerical simulations are carried out at an undercooling of $\Delta = 1$ using $\tau = 0.0003$, $d_o = 0.01$, $\delta = 0.02$, $\beta = 4$, $\gamma_o = 10.0$, $K = 1.8$, $\alpha = 0.9$, and $\Delta t = 0.0001$. Two preferential growth angles are used for the simulations. The first one is aligned with the x-axis of the computational domain ($\theta_o = 0^o$), while the second is aligned with the corners of the domain ($\theta_o = 45^\circ$). After 5000 time steps, the PF order parameter is shown in Fig. 19. The 0.5 isocontours of the PF solution at both orientations are overlayed as shown in Fig. 20 where we can clearly see that simply changing the orientation of the grid changed the dendrite tip location which indicates a deviation in the dendrite tip velocity. This behaviour indicates that grid-induced anisotropy strongly influenced the final PF solution. This is often a side-effect of using structured grids and is often difficult to completely eliminate.

Using the same PF parameters, we now perform the meshfree PF simulations using the SPH–PFM on a circular domain of radius $r = 3$ where there is no specific preferential growth direction introduced by the domain geometry itself. A structured triangular mesh is used to initialize the distribution of 50,000 particles. PF simulations are carried out using RBF-GE at orientation angles of $\theta_o = 0^\circ$ and $\theta_o = 45^\circ$. As can be seen in Fig. 21, a comparable PF solution is obtained at both orientations

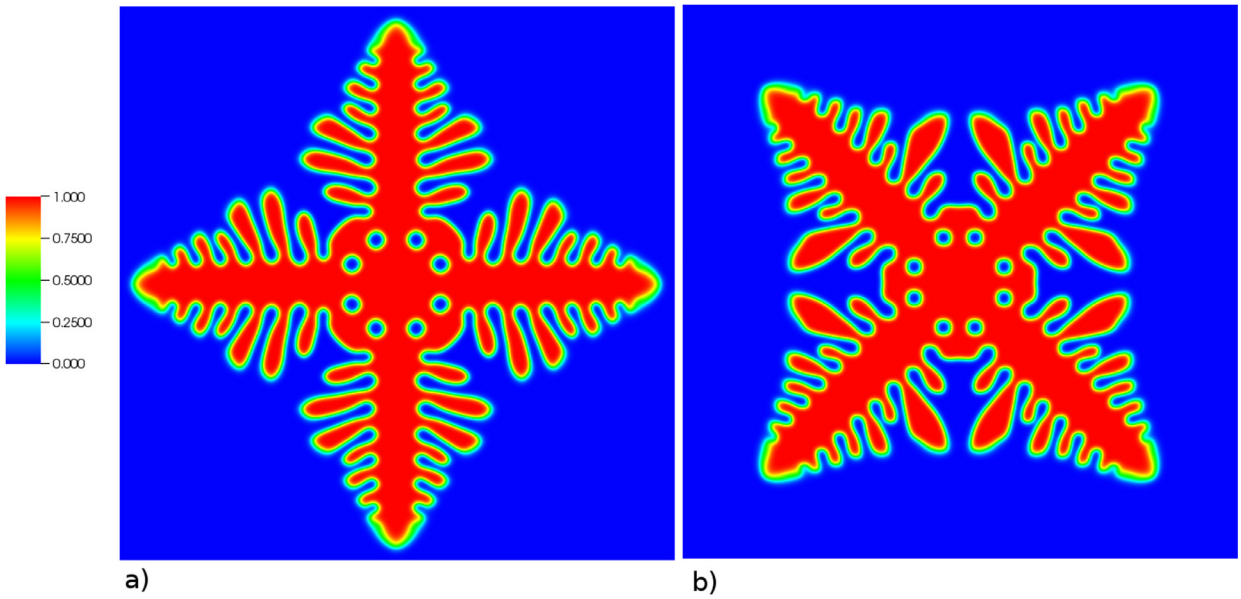

Fig. 19. PF solution using the finite difference method for a 4-branch dendrite with preferential growth angle of (a) $\theta_o = 0^\circ$ and (b) $\theta_o = 45^\circ$.

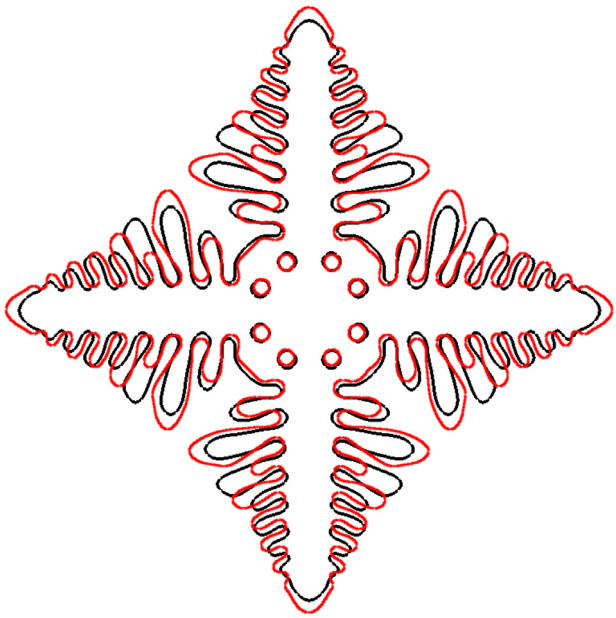

Fig. 20. The 0.5 isocontour of PF finite difference solution for a dendrite with preferential growth angle of $\theta_o = 0^\circ$ (black curve) and $\theta_o = 45^\circ$ (red curve). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

of the dendrites. This is further validated by overlaying the 0.5 isocontours of both PF solutions where we clearly see overall comparable results at both orientations especially at the dendrite tips indicating significant reduction in geometry induced anisotropy (Fig. 22).

While using a triangular mesh to initialize the distribution of particles in the domain has the advantage of easy handling of arbitrary domain geometries, it is found that a structured distribution of particles is still necessary to obtain a symmetric dendritic branch growth. To demonstrate this, the effect of unstructured distribution of the domain particles on the grown dendrites was investigated by using the same circular domain with radius $r = 3$ and using the same PF parameters noted earlier. Numerical simulations using the same meshfree parameters is shown in Fig. 23(a) where we see that while the 4 main dendrite branches are visible, the dendritic structure is without discernible symmetry as that obtained using a more structured particle distribution despite using the same PF simulation parameters. A similar observation is obtained by

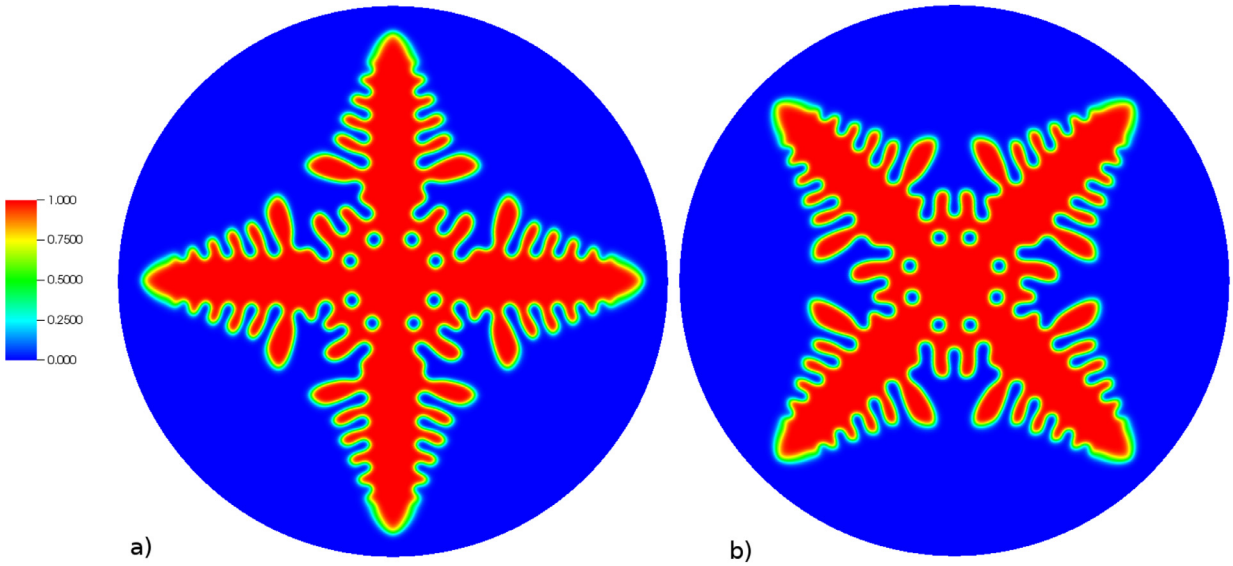

Fig. 21. PF solution using the SPH–PFM for a dendrite with preferential growth angle of (a) $\theta_o = 0^\circ$ and (b) $\theta_o = 45^\circ$.

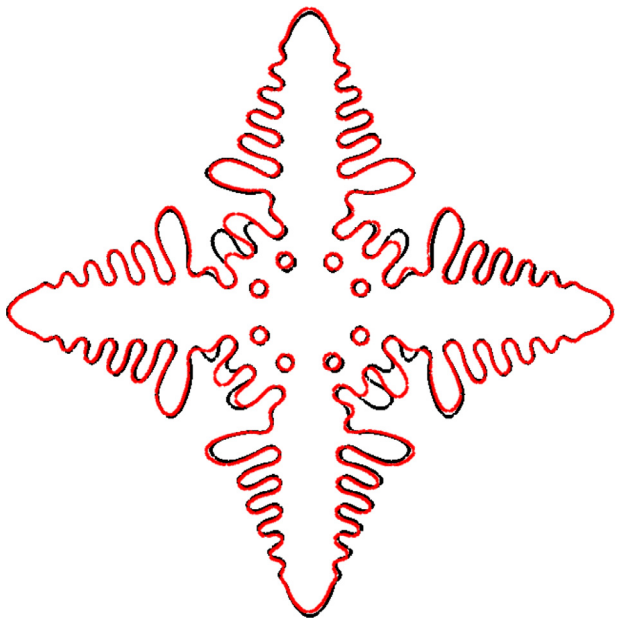

Fig. 22. The 0.5 isocontour of SPH–PFM solution for a dendrite with preferential growth angle of $\theta_o = 0^\circ$ (black curve) and $\theta_o = 45^\circ$ (red curve). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

varying the anisotropy parameter β from $\beta = 4$ to $\beta = 6$ where a 6-branch growth is correctly obtained as shown Fig. 23(b) but the growth of secondary dendrite branches are clearly not symmetric. We conclude that the regularity of the underlying particle distribution strongly influences the symmetry of the grown dendrite branches.

Further investigations are done where the thermal distribution is computed at a lower particle spacing resolution than that used for solving the PF equation. A set of fixed particles with a structured distribution is used to describe the same circular computational domain of radius $r = 3$. A solid seed is centered in the domain and the thermal distribution of particles is computed everywhere in the domain using the same simulation parameters noted above. A narrow band that is about 6 times wider than the thickness of the PF order parameter is constructed and meshfree particles are generated such that the PF order parameter is properly resolved. As the PF order parameter evolves, the set of particles within the narrow band is updated as shown in Fig. 24. The temperature distribution at particles within the narrow band is interpolated using the temperature distribution at domain particles and used for solving the PF equation at the narrow band. As shown in

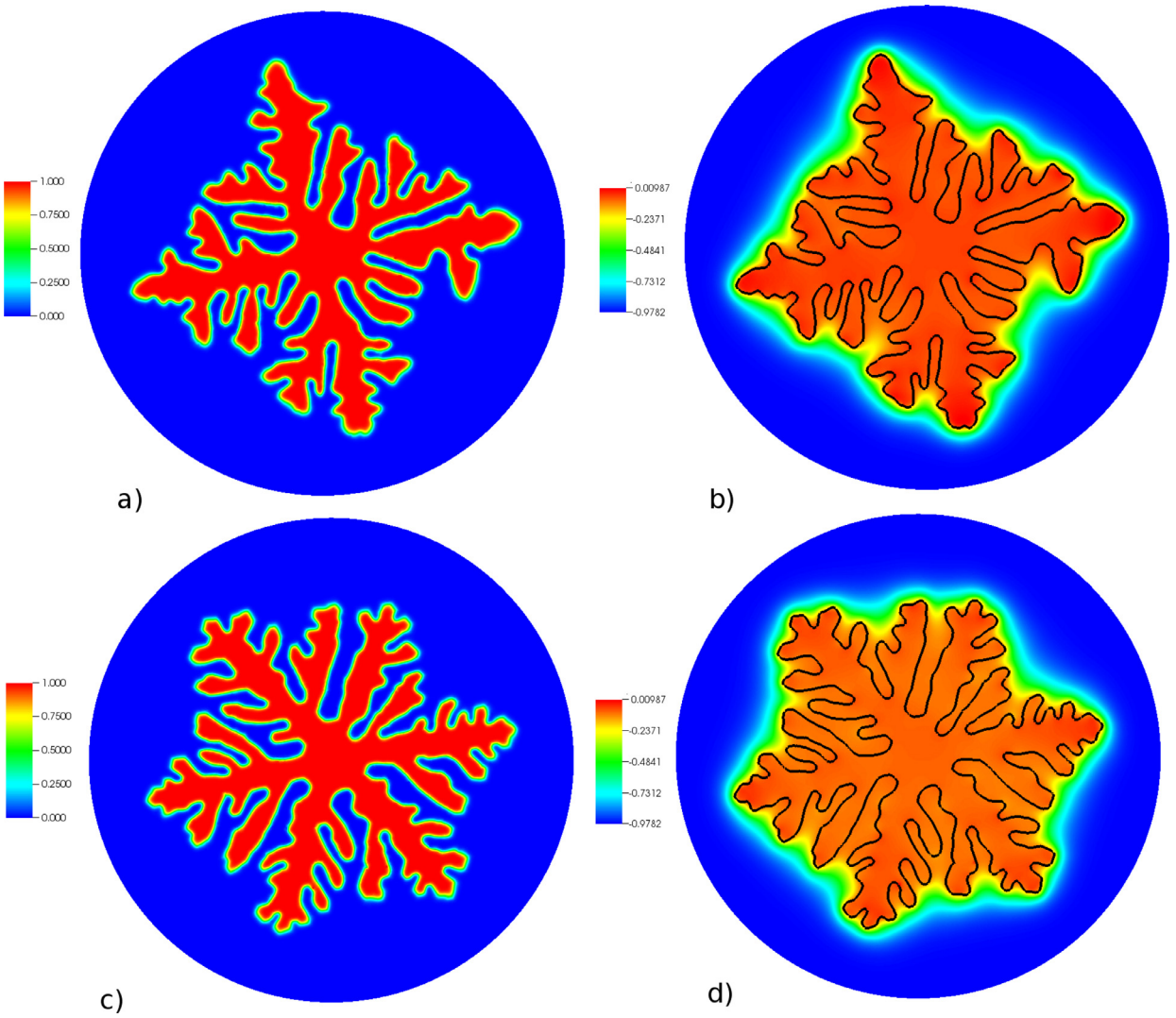

Fig. 23. Using a random particle distribution in a circular domain of radius $r = 3$, the PF and temperature distribution using the SPH–PFM are shown for a 4-branch dendrite (a and b) and for a 6-branch dendrite (c and d).

Fig. 24, a 6-branch dendrite is successfully obtained where secondary dendrite branches are clearly observed. Notice that in comparison to the result shown in Fig. 21, the secondary dendrite branches obtained is not perfectly as symmetric which is expected due to the unstructured distribution of particles locally at the narrow band.

The results obtained so far indicate that the use the SPH–PFM has the ability of correctly predicting the expected dendritic structure of solidification as that obtained using finite differences. However, unlike finite differences which is limited to simple domain geometries and often heavily influence the PF solution where a perfectly symmetric dendrite structure is obtained, the SPH–PFM has the ability of handling arbitrary domain particle distributions. Unstructured particle distribution is found to have a direct influence on the symmetry of the grown dendrite. This is particularly visible in the growth and propagation of secondary dendrite branches.

4.4. Using the SPH–PFM to investigate the effect of phase-field parameters on dendritic morphology

The PF solution for dendritic growth is dependent on a number of simulation parameters. In this section, we demonstrate the ability of SPH–PFM in predicting the effect of various PF parameters for simulating the growth a single solid seed centered in a circular domain of radius $r = 3$ discretized using 50,000 particles. Numerical simulations are carried out using RBF–GE with a shape parameter of $\gamma = 0.01$ for constructing the meshfree weighting functions in addition to using default parameters of $\tau = 0.0003$, $d_o = 0.01$, $\delta = 0.02$, $K = 1.8$, $\alpha = 0.9$, $\gamma_o = 9.0$, $\Delta t = 0.0001$ and an arbitrary orientation angle of $\theta_o = 12^\circ$. The effect of varying different parameters on the final dendrite solution after 5000 time steps is summarized

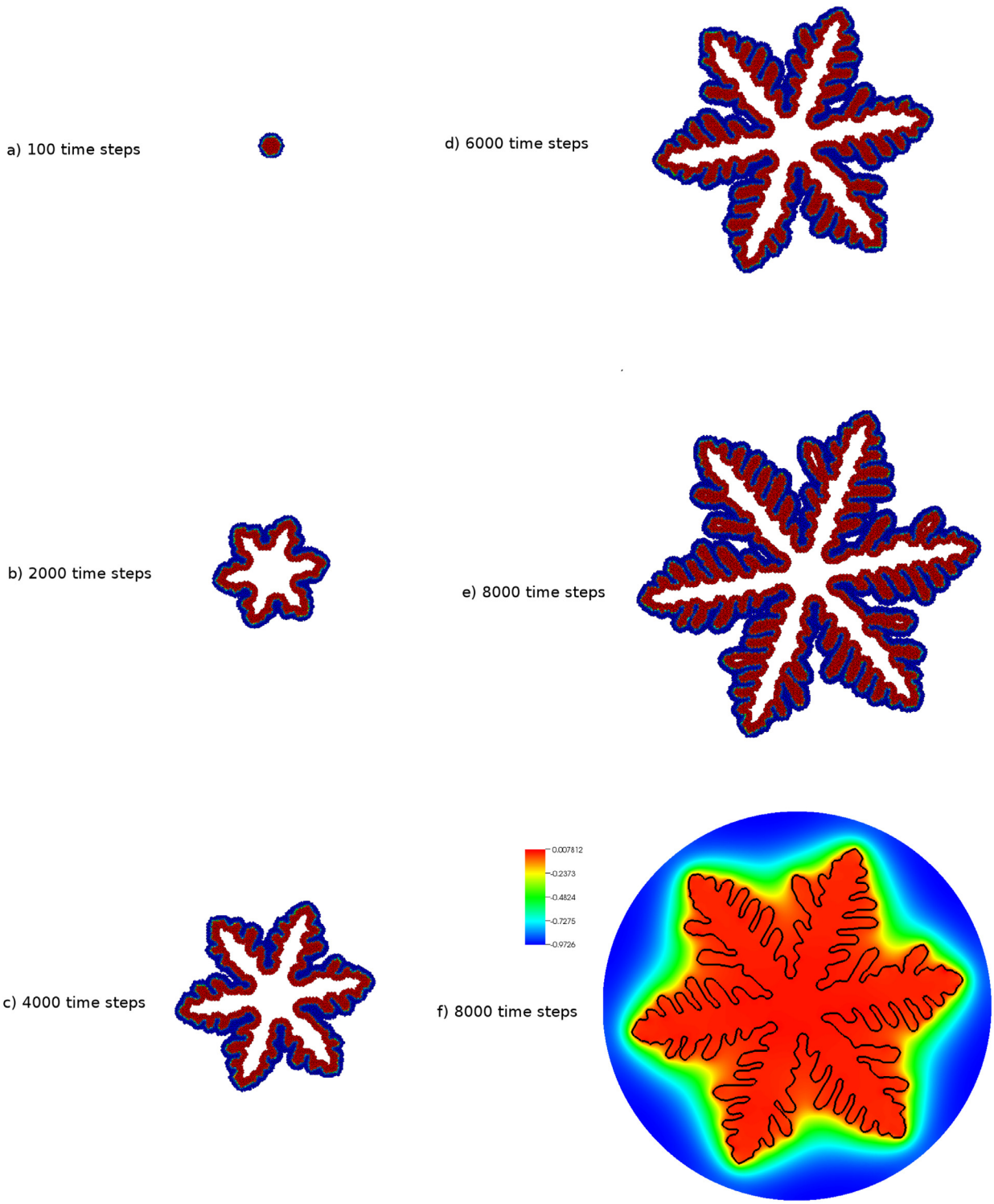

Fig. 24. Evolution of the particles at the narrow band region around the liquid–solid interface for a 6-branch dendrite. Blue particles correspond to $\Psi = 0$ while red particles correspond to $\Psi = 1$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

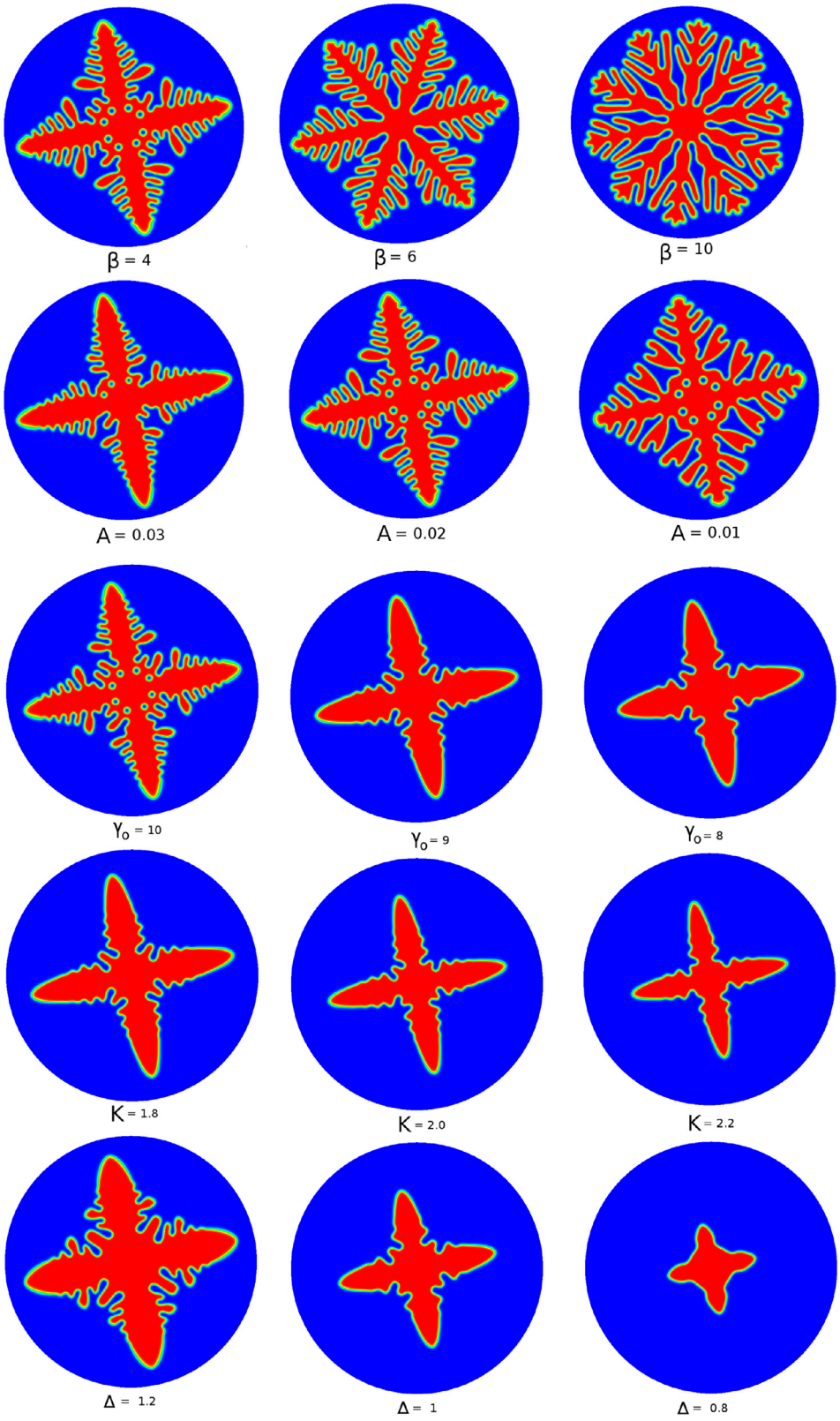

Fig. 25. Effect of varying PF parameters on the resulting dendrite morphology as predicted using the SPH–PFM with RBF–GE.

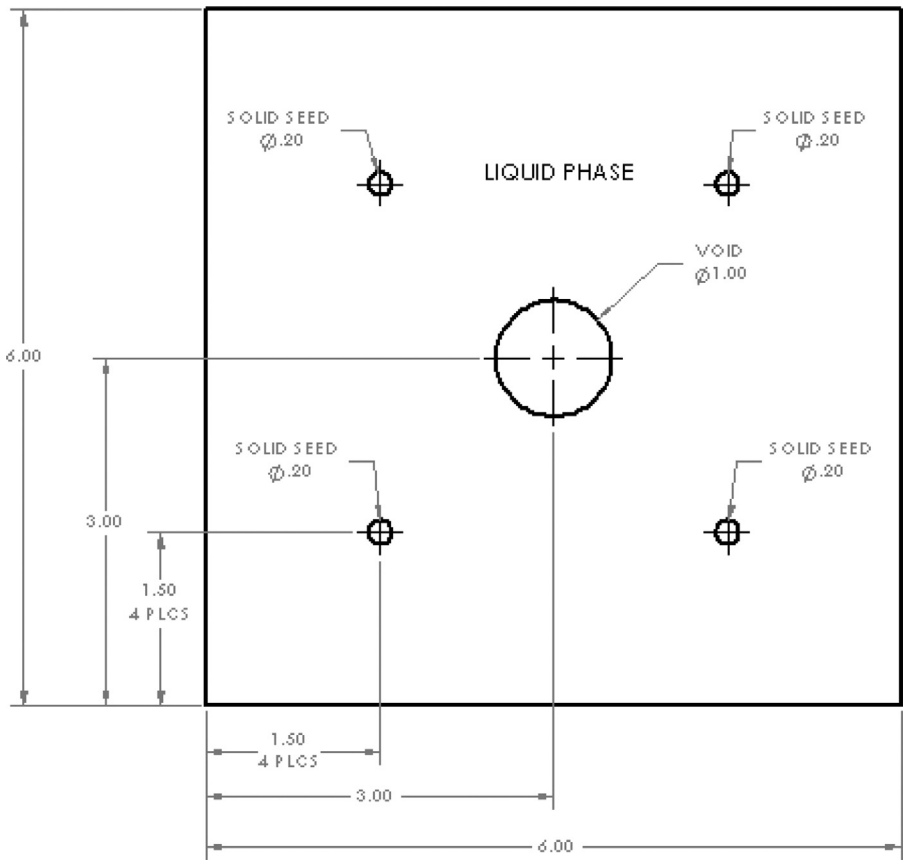

Fig. 26. Geometry of the computational domain used for PF simulation of dendritic solidification of 4 seeds.

in Fig. 25 where we observe the ability to easily capture the formation of primary and secondary dendrite branches. As expected, varying the anisotropy parameter β from 4 to 10 yields the correct number of primary dendritic branches. We have found that the parameter γ_o has significant effect on the surface tension of the liquid–solid interface. As γ_o reduces, surface tension is increased which suppresses the formation of secondary dendrite branches. The thickness of the liquid–solid interface itself is also directly affected by the choice of γ_o such that using $\gamma_o = 10$ visibly showed a wider liquid–solid transition region than that obtained using $\gamma_o = 8$. Increasing the thermal conductivity parameter K reduces the rate of solidification. This is expected since the thermal heat rejected at the liquid–solid interface is diffused further into the liquid resulting in a reduction in the thermal gradient ahead of the interface which in effects reduces the interface velocity of the liquid–solid interface. Increasing the undercooling Δ increases the rate of solidification as expected where the interfacial instability results in more aggressive growth of the secondary dendrite branches. To demonstrate the ability of the SPH–PFM in handling the growth of multiple dendrites where each dendrite has its own independent preferential growth angle we consider a 2D square computational domain with a circular obstacle as shown illustrated in Fig. 26. We assume the obstacle is perfectly insulated. We initialize the domain with 4 seeds each assigned some random growth angle θ_o between 0 and π. Numerical simulations are carried out at an undercooling of $\Delta = 1$ using $\tau = 0.0003$, $d_o = 0.01$, $A = 0.02$, $\beta = 4$, $K = 1.8$, $\alpha = 0.9$, $\gamma_o = 9.0$ and $\Delta t = 0.0001$. The evolution of the PF order parameter and temperature profile is shown in Fig. 27. where we observe the independent orientation of the growing dendrit and the formation of primary and secondary dendrite branches. Note also the competitive growth between the dendrite arms and their evolution around the obstacle. The thermal conductivity is varied between $K = 2$ and $K = 1.6$ and the PF solution is used to compute the percentage of solid present in the domain as plotted in Fig. 28. Initially, the growth rate of the dendrites is high, however it starts to plateau as expected due to the accumulation of rejected thermal heat ahead of the dendrite interface. As the grown dendrites approach each other, heat trapped at the liquid phase between the solid dendrites reduces the growth rate of the dendrites. Therefore, after 4000 time steps, decreasing the thermal conductivity from 1.8 to 1.6 is shown to increase the amount of solid present in the domain as shown in Fig. 29. Increasing the thermal conductivity to 2.0 allowed for a faster diffusion of the rejected heat in the liquid phase which resulted in reducing the growth rate of the dendrites. These observations are consistent with the results obtained by Kobayashi [22]. Further demonstration of the SPH–PFM capability is done using an irregular 2D closed spline curved geometry with an irregular internal cavity. A total of 7 seeds are randomly scattered in the domain where each seed is assigned its own preferential growth angle chosen randomly. A 6-branch dendritic growth at different time steps

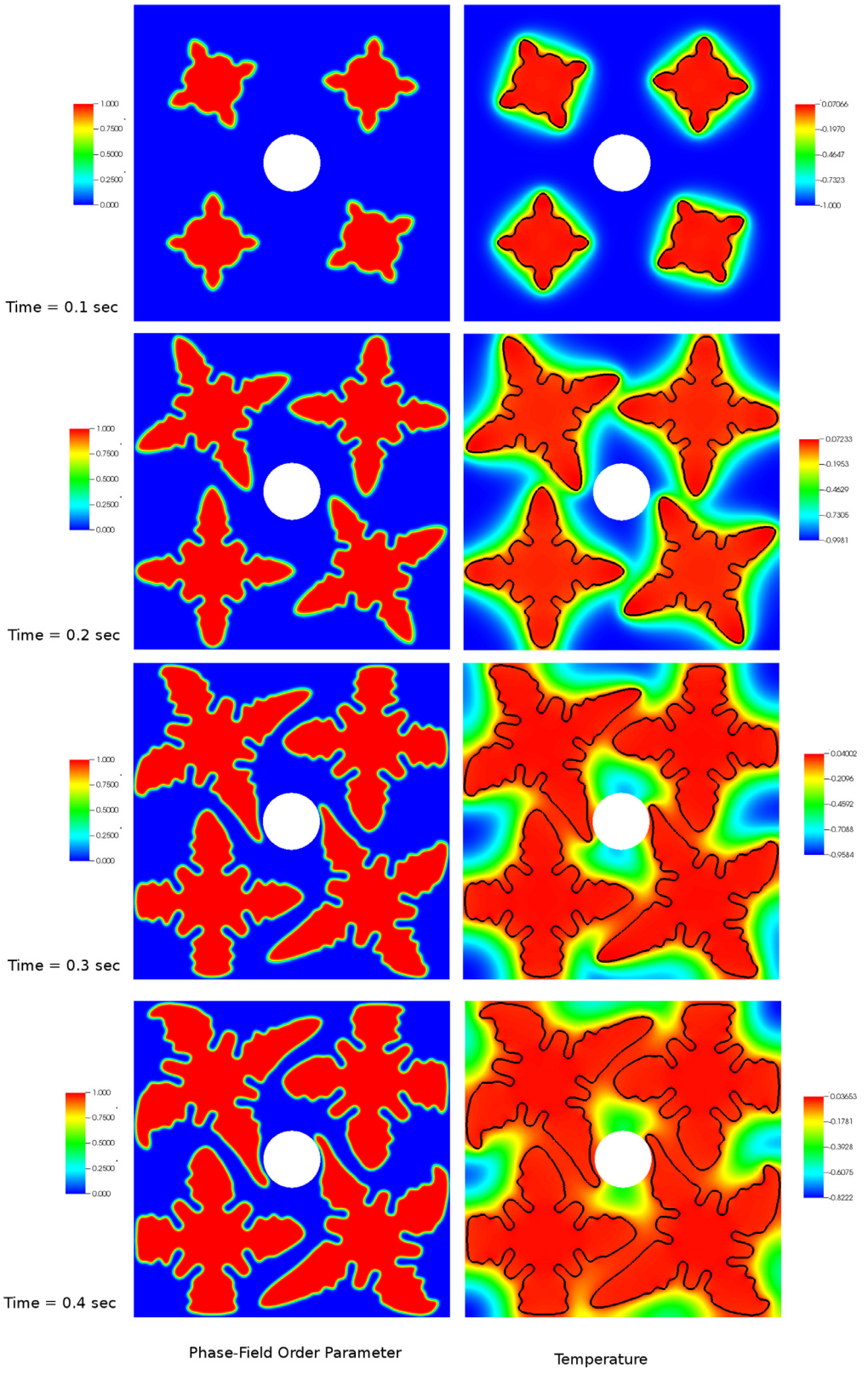

Fig. 27. SPH–PFM solution for the evolution of 4 solid seeds with random preferential growth angle θ_o in a square domain of length 6.0×6.0 and a central circular obstacle of radius $r = 0.5$ at undercooling of $\Delta = 1$.

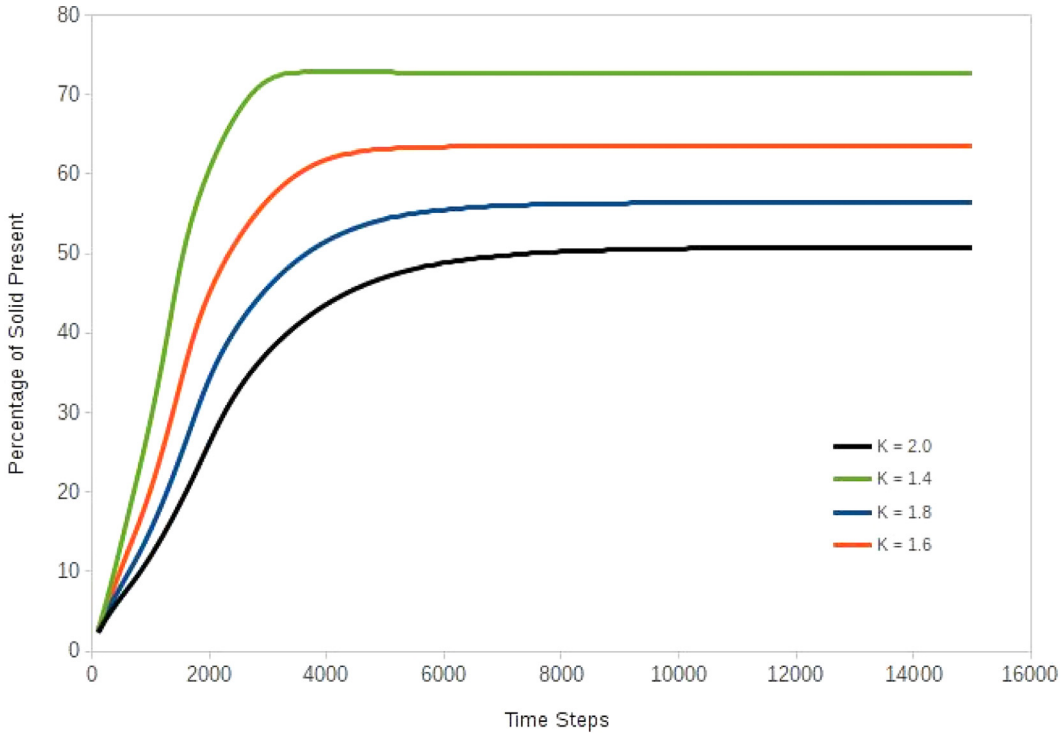

Fig. 28. Effect of varying the thermal conductivity parameter K on the rate of simultaneous solidification of 4 randomly oriented seeds.

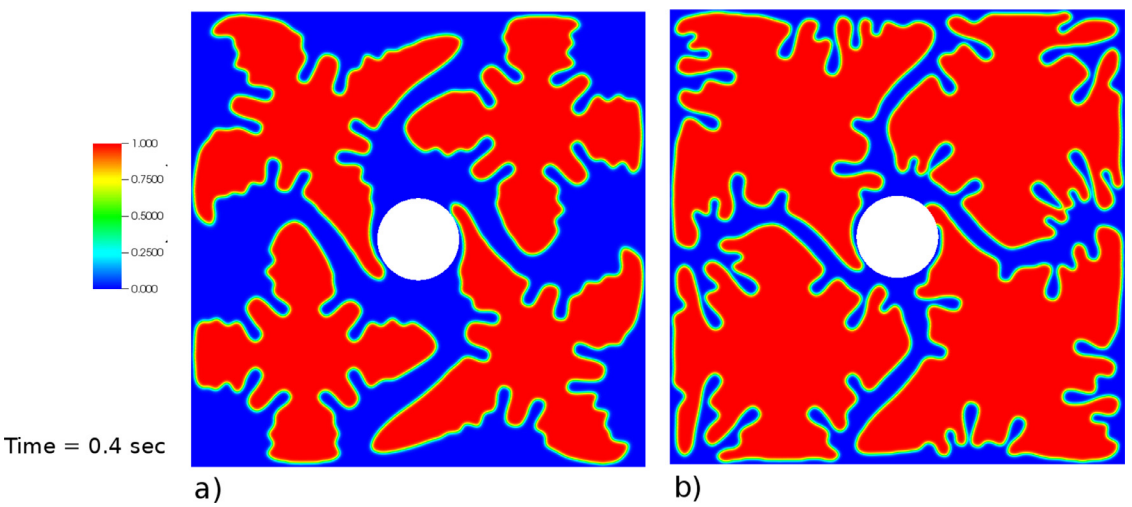

Fig. 29. Final solid distribution after 4000 time steps for 4 simultaneously solidified seeds using a thermal conductivity of (a) $K = 1.8$ and (b) $K = 1.6$.

is shown in Fig. 30 where we observe the competitive growth between the dendrites and the formation of the expected primary and secondary dendrite branches. This is further validated using the same domain geometry for an initially planar solid interface with small purturbations as shown in Fig. 31. As the liquid–solid interface propagates, the Mullins–Sekerka interface instability is observed where the planar interface forms dendritic fingers. The competitive growth between the dendrite branches and the eventual splitting and growth of secondary dendrite branches is also clearly observed. These results demonstrate the ability of the SPH-PFM in handling arbitrary domain goemetries with and without internal cavities while easily handling the complex morphologies and interfacial dynamics often involved in dendritic solidification.

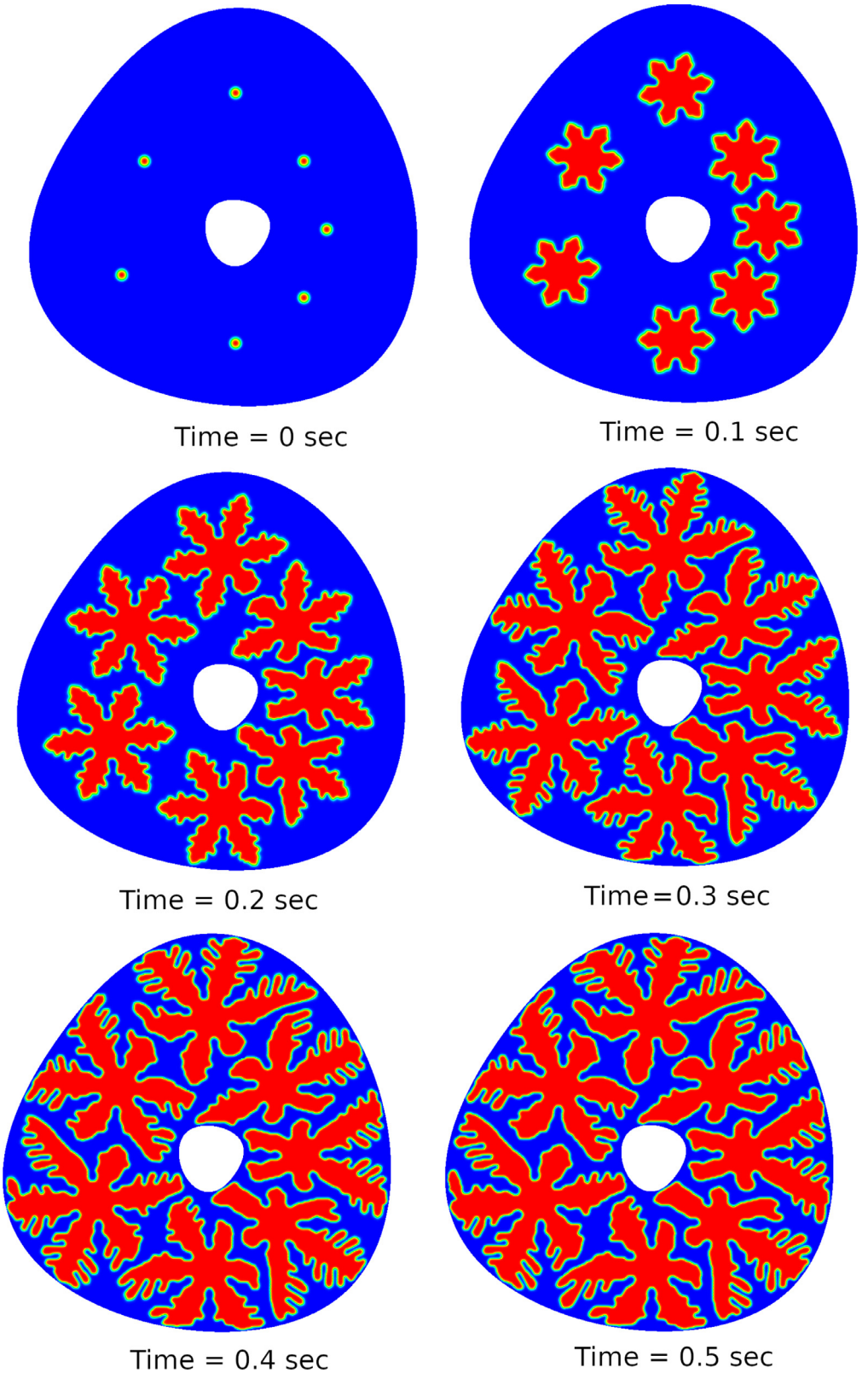

Fig. 30. SPH–PFM solution for the evolution of 7 randomly distributed and randomly oriented seeds in an irregular domain geometry with internal cavity at undercooling of $\Delta = 1$.

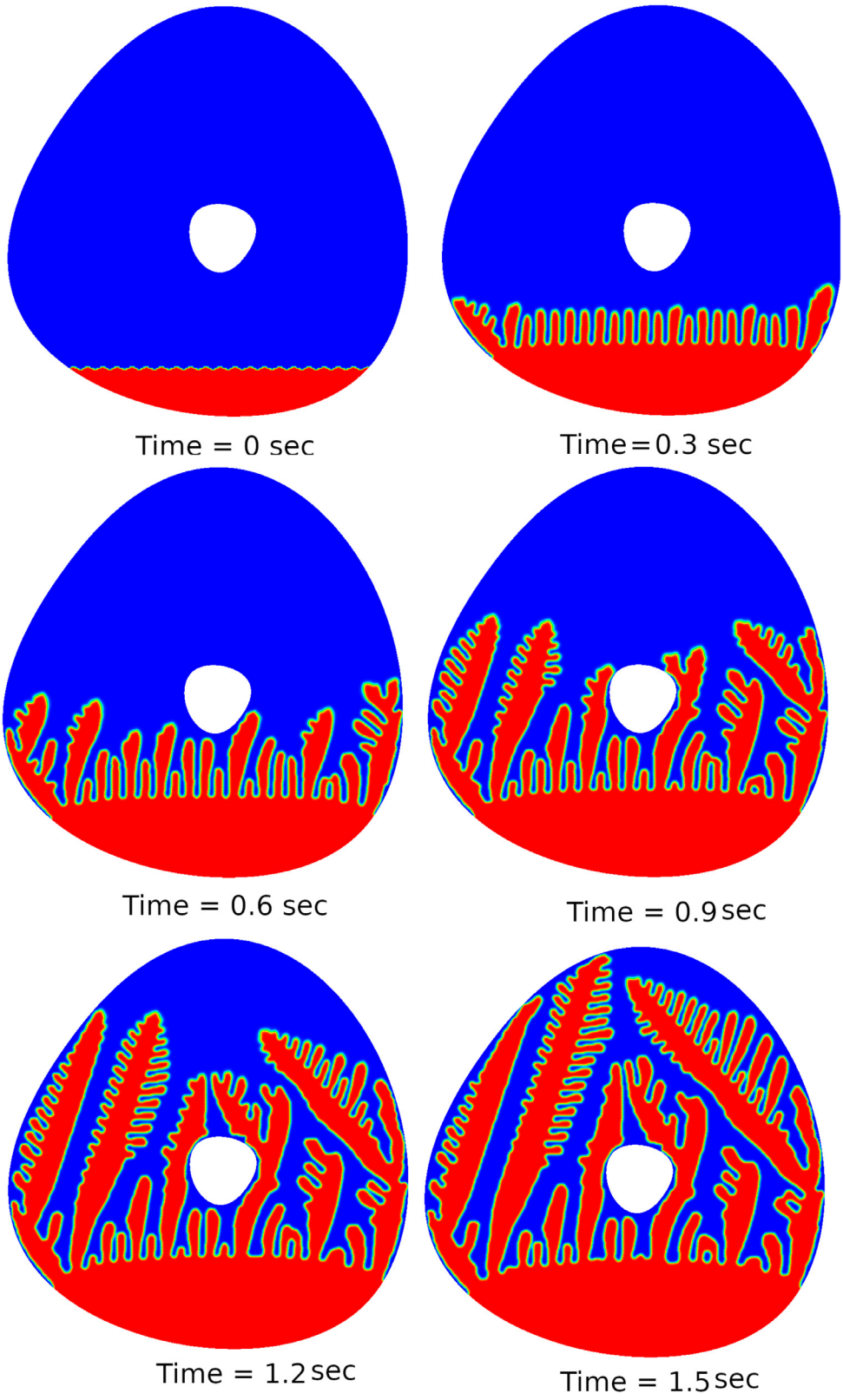

Fig. 31. SPH–PFM solution for the evolution of a planar interface with small perturbations in an irregular domain geometry with internal cavity at undercooling of $\Delta = 1$.

5. Conclusions

We presented a new approach for meshfree PF modelling of dendritic solidification using SPH. The attractive methodology of SPH in solving PDEs is utilized such that computing global matrices or solving large systems of equations is not required. In contrast to the kernel functions traditionally used in SPH, we used MLS and RBFs to construct weighting functions capable of achieving a higher order of consistency. Coupling of SPH with the PF method allowed for easy incorporation of surface tension effects without the need for interface reconstruction. It also allowed for easy handling of the growth of multiple dendrites with independent preferential growth angles on arbitrary domain geometries with or without internal cavities.

Several factors were taken into account to ensure a robust and efficient meshfree model for PF simulations. For a fixed set of meshfree domain particles with a relatively coarse spacing resolution, the set of interpolation functions are computed only once during pre-processing and reused during transient computations. As the diffuse interface evolves, a narrow band around the interface is constructed and a set of meshfree interfacial particles are constructed which resolves the interface thickness independent of the domain particle spacing resolution used to compute the thermal field distribution. Only first order derivatives of the weight functions are needed to solve the governing PF equations resulting in a more efficient and stable solution. For imposing Neumann boundary conditions at the domain boundaries, we employed a localized technique based on implicit surface reconstruction of the domain boundaries which allowed for handling arbitrary domain geometries with or without internal cavities.

The ability to handle arbitrary domain geometries was found to potentially reduce the grid-based anisotropy typically encountered in grid-based finite differences. We studied the effect of meshfree parameters used in the SPH–PFM on the resulting PF solution. It was found that while interpolants constructed using traditional SPH kernel functions, RBFs and MLS yield very comparable dendritic morphologies, MLS was found to be abit more computationally expensive than RBFs while interpolants constructed using traditional SPH kernel functions are found to be less accurate than both RBF and MLS. There was no notable difference in accuracy or in the obtained dendritic morphologies when quartic splines were used instead of cubic splines in MLS computations. Shape parameters within the range of 0.001–0.1 used for computing RBF-GE and RBF-IMQ was found to have a very limited effect on the morphology of the dendrite. Unstructured distribution of the particles was found to significantly affect the symmetry of the grown dendrites. This was particularly evident in the growth of secondary dendrite branches despite obtaining the correct overall morphology of the primary dendrite branches based on the anisotropy parameter selected. We demonstrated the ability of SPH–PFM in capturing the effects of various PF process parameters and handling the growth of multiple dendrites with independent crystallographic orientations. The work presented in this paper indicates that the SPH–PFM can be a suitable tool for PF simulations without the limitations of popular methods based on finite differencing or finite element formulations.

References

[1] S. Osher, J. Sethian, Level set method, J. Comput. Phys. 79 (1988) 12–49.
[2] N. Zabaras, B. Ganapathysubramanian, L. Tan, Modeling dendritic solidification with melt convection using the extended finite element method, J. Comput. Phys. 218 (1) (2006) 200–227.
[3] Y.T. Kim, N. Goldenfeld, J. Dantzig, Computation of dendritic structures using a level set method, Phys. Rev. E 62 (2000) 2471–2474.
[4] B. Ningegowda, B. Premachandran, A coupled level set and volume of fluid method with multi-directional advection algorithms for two-phase flows with and without phase change, Int. J. Heat Mass Transf. 79 (2014) 532–550.
[5] X. Lv, Q. Zou, Y. Zhao, D. Reeve, A novel coupled level set and volume of fluid method for sharp interface capturing on 3d tetrahedral grids, J. Comput. Phys. 229 (2010) 2573–2604.
[6] S. Osher, R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Springer, New York, 2003.
[7] D. Enright, R. Fedkiw, J. Firziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (1) (2002) 83–116.
[8] D. Enright, Use of the particle level set method for enhanced resolution of free surface flows, PhD Thesis, Stanford University, 2002.
[9] S. Ianniello, A. Mascio, A self-adaptive oriented particles level-set method for tracking interfaces, J. Comput. Phys. 229 (2010) 1353–1380.
[10] S. Hieber, P. Koumoutsakos, A lagrangian particle level set method, J. Comput. Phys. 210 (2005) 342–367.
[11] K. Reuther, M. Rettenmayr, Simulating dendritic solidification using an anisotropy-free meshless front-tracking method, J. Comput. Phys. 279 (2014) 63–66.
[12] Y. Aizawaa, J. Nishiwaki, Y. Harada, S. Muraishi, S. Kumai, Experimental and numerical analysis of the formation behavior of intermediate layers at explosive welded Al/Fe joint interfaces, J. Manufact. Process. 24 (2016) 100–106.
[13] C. Li, C. Xu, C. Gui, M. Fox, Distance regularized level set evolution and its application to image segmentation, IEEE Trans. Image Process. 19 (12) (2010) 3243–3254.
[14] K. Zhang, L. Zhang, D. Zhang, Reinitialization-free level set evolution via reaction diffusion, IEEE Trans. Image Proces. 22 (1) (2013) 258–271.
[15] A. Ghoneim, A meshfree interface-finite element method for modeling isothermal solutal melting and solidification in binary systems, Finite Elem. Anal. Des. 95 (2015) 20–41.
[16] A. Ghoneim, A new technique for numerical simulation of dendritic solidification using a meshfree interface finite element method, Int. J. Numer. Methods Eng. 107 (10) (2016) 813–852.
[17] A.Y. Ghoneim, The meshfree interface finite element method for numerical simulation of dendritic solidification with fluid flow, Int. J. Numer. Methods Eng. 15 (7) (2018) 1850057.
[18] S.M. Allen, J.W. Cahn, A macroscopic theory for antiphase boundary motion and its application to antiphase domain coarsening, Acta Metall. 27 (1979) 1085–1095.
[19] J.W. Cahn, J.E. Hilliard, Free energy of a nonuniform system. I. interfacial free energy, J. Chem. Phys. 28 (1958) 258–267.
[20] J.W. Cahn, J.E. Hilliard, Free energy of a non-uniform system. III. Nucleation in a two point compressible fluid, J. Chem. Phys. 31 (1959) 688–699.

[21] G. Caginalp, P. Fife, Phase-field methods for interfacial boundaries, Phys. Rev. B 33 (11) (1986) 7792–7794.
[22] R. Kobayashi, Modeling and numerical simulations of dendritic crystal growth, Phys. D 63 (1993) 410–423.
[23] H. Ji, D. Chopp, J.E. Dolbow, A hybrid extended finite element/level set method for modeling phase transformations, Int. J. Numer. Methods Eng. 54 (2002) 1209–1233.
[24] G. Caginalp, X. Chen, On The Evolution of Phase Boundaries, Springer-Verlag, USA, 1992.
[25] A. Wheeler, W. Boettinger, G. Mcfadden, Phase-field model for isothermal phase transitions in binary alloys, Phys. Rev. A 45 (1992) 7424–7440.
[26] S.L. Wang, R.F. Sekerka, A.A. Wheeler, B.T. Murray, S.R. Coriell, R.J. Braun, G.B. McFadden, Thermodynamically consistent phase field models for solidification, Phys. D 69 (1993) 189200.
[27] A. Karma, W. Rappel, Quantitative phase-field modeling of dendritic growth in two and three dimensions, Phys. Rev. E 57 (1998) 4323–4349.
[28] A. Shah, A. Haider, S. Shah, Numerical simulation two-dimensional dendritic growth using phase-field model, World J. Mech. 4 (2014) 128–136.
[29] P. Bollada, C. Goodyer, P. Jimack, A. Mullis, F. Yang, Three dimensional thermal-solute phase field simulation of binary alloy solidification, J. Comput. Phys. 287 (2015) 130–150.
[30] A.F. Ferreira, L.O. Ferreira, Microsegregation in Fe-C-P ternary alloys using a phase-field model, J. Braz. Soc. Mech. Sci. Eng. 31 (2009) 173–180.
[31] T. Suzuki, M. Ode, S.G. Kim, W.T. Kim, Phase-field model of dendritic growth, J. Cryst. Growth 237 (2002) 125–131.
[32] Y. Zhang, C. Wang, D. Li, Y. Li, Phase field modeling of dendritic growth, Acta Metall. Sin. 22 (3) (2009) 197–201.
[33] C. Zhu, R. Xiao, Z. Wang, L. Feng, Numerical simulation of recalescence of 3-dimensional isothermal solidification for binary alloy using phase-field approach, Trans. Nonferrous Metals Soc. China 19 (5) (2009) 1286–1293.
[34] A. Ferreira, A. Silva, J. Castro, Simulation of the solidification of pure nickel via the phase-field method, Mater. Res. 9 (4) (2006) 349–356.
[35] N. Provatas, M. Greenwood, B. Athreya, N. Goldenfeld, J. Dantzig, Multiscale modeling of solidification: phase-field methods to adaptive mesh refinement, Int. J. Modern Phys. B 19 (31) (2005) 4525–4565.
[36] T. Takaki, T. Fukuoka, Y. Tomita, Phase-field simulation during directional solidification of a binary alloy using adaptive finite element method, Journal of Crystal Growth 283 (2005) 263–278.
[37] C. Zhu, P. Lei, R. Xiao, L. Feng, Phase-field modeling of dendritic growth under forced flow based on adaptive finite element method, Trans. Nonferrous Merals Soc. China 25 (2015) 241–248.
[38] P. Yue, C. Zhou, J. Feng, C. Ollivier-Gooch, H. Hu, Phase-field simulations of interfacial dynamics in viscoelastic fluids using finite elements with adaptive meshing, J. Comput. Phys. 219 (2006) 47–67.
[39] X. Hu, R. Li, T. Tao, A multi-mesh adaptive finite element approximation to phase field models, Commun. Comput. Phys. 5 (2009) 1012–1029.
[40] A. Rosolen, C. Peco, M. Arroyo, An adaptive meshfree method for phase-field models of biomembranes. Part I: Approximation with maximum-entropy basis functions, J. Comput. Phys. 249 (2013) 303–319.
[41] C. Peco, A. Rosolen, M. Arroyo, An adaptive meshfree method for phase-field models of biomembranes. Part II: A Lagrangian approach for membranes in viscous fluids, J. Comput. Phys. 249 (2013) 320–336.
[42] M. Dehghan, M. Abbaszadeh, The meshless local collocation method for solving multi-dimensional Cahn–Hilliard, Swift–Hohenberg and phase field crystal equations, Eng. Anal. Bound. Elem. 78 (2017) 49–64.
[43] M. Dehghan, V. Mohammadi, The numerical solution of cahnhilliard (CH) equation in one, two and three-dimensions via globally radial basis functions (GRBFs) and RBFs-differential quadrature (RBFs-DQ) methods, Eng. Anal. Bound. Elem. 51 (2015) 74–100.
[44] M. Dehghan, V. Mohammadi, The numerical simulation of the phase field crystal (PFC) and modified phase field crystal (MPFC) models via global and local meshless methods, Comput. Methods Appl. Mech. Eng. 298 (2016) 453–484.
[45] N. Talat, B. Mavric, V. Hatic, S. Bajt, B. Sarler, Phase field simulation of rayleightaylor instability with a meshless method, Engineering Analysis with Boundary Elements 87 (2018) 78–89.
[46] J.X. Zhou, M.E. Li, Solving phase field equations using a meshless method, Commun. Numer. Methods Eng. 22 (2006) 1109–1115.
[47] J. Song, Y. Fu, T. Kim, Y. Yoon, J.G. Michopoulos, T. Rabczuk, Phase field simulations of coupled microstructure solidification problems via the strong form particle difference method, Int. J. Mech. Mater. Des. (2017).
[48] L.B. Lucy, A numerical approach to the testing of the fission hypothesis, Astron. J. 82 (12) (1977) 10131024.
[49] R.A. Gingold, J.J. Monaghan, Smoothed particle hydrodynamics – theory and application to non-spherical stars, Month. Notic. R. Astronom. Soc. 181 (1977) 375–389.
[50] J.J. Monaghan, Smoothed particle hydrodynamics, Ann. Rev. Astron. Astrophys. 30 (1992) 543–574.
[51] O. Laguna, Smoothed particle interpolation, Astrophys. J. 439 (1995) 814–821.
[52] A. Panizzo, Physical and Numerical Modelling of Subaerial Landslide Generated Waves, Ph. D. Thesis, 2004.
[53] G.A. Dilts, Moving least squares particle hydrodynamics i. consistency and stability, Int. J. Numer. Methods Eng. 44 (1999) 1115–1155.
[54] R. Brownleee, P. Houston, J. Levesley, S. Rosswog, Enhancing sph using moving least-squares and radial basis functions, in: Proceedings of the 5th International Conference Algorithms for Approximation, Chester, 2005, pp. 103–112.
[55] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, P. Krysl, Meshless methods: an overview and recent developments, Comput. Methods Appl. Mech. Eng. 139 (1996) 3–47.
[56] J. Ha, Numerical comparison of radial basis functions and generalized smoothed particle hydrodynamics, in: Proceedings of the Fifth International Conference on CFD in the Process Industries, CSIRO, 2006, pp. 1–6. Melbourne, Australia
[57] J.J. Monaghan, Smoothed particle hydrodynamics, Rep. Progr. Phys. 68 (2005) 1703–1759.
[58] L. Brookshaw, A method of calculating radiative heat diffusion in particle simulations, Proc. Astronom. Soc. Austr. 6 (1985) 207–210.
[59] P.W. Cleary, Modelling confined multi-material heat and mass flows using SPH, Appl. Math. Model. 22 (1998) 981–993.
[60] P.W. Cleary, J.J. Monaghan, Conduction modelling using smoothed particle hydrodynamics, J. Comput. Phys. 148 (1999) 227–264.
[61] R. Fatehi, M.A. Fayazbakhsh, M. Manzari, On discretization of second-order derivatives in smoothed particle hydrodynamics, Int. J. Mech. Mechatron. Eng. 2 (4) (2008) 428–431.
[62] J.J. Monaghan, H.E. Huppert, M.G. Worster, Solidification using smoothed particle hydrodynamics, J. Comput. Phys. 206 (2005) 684–705.
[63] B. Andersson, S. Jakobsson, A. Mark, F. Edelvik, L. Dividson, Modeling surface tension in SPH by interface reconstruction using radial basis functions, in: Proceedings of the Fifth International SPHERIC Workshop, Manchester, UK, 2010.
[64] M. Yang, X. Li, G. Yang, E. Wu, SPH-Based fluid simulation with a new surface tension formulation, in: Proceedings of the 2015 International Conference on Virtual Reality and Visualization, 2015.
[65] P. Lancaster, K. Salkauskas, Surface generated by moving least squares methods, Math. Comput. 37 (1981) 141–158.
[66] T. Belytschko, Y.Y. Lu, L. Gu, Element-free galerkin methods, Int. J. Numer. Methods Eng. 37 (1994) 229–256.
[67] T. Most, C. Bucher, A moving least squares weighting function for the element-free Galerkin method which almost fulfills essential boundary conditions, Struct. Eng. Mech. 21 (3) (2005) 315–332.
[68] A. Crespo, Application of the Smoothed Particle Hydrodynamics model SPHysics to free-surface hydrodynamics, Ph.D. Thesis, Universidade de Vigo, 2008.
[69] A. Colagrossi, A Meshless Lagrangian Method for FreeSurface and Interface Flows with Fragmentation, Ph.D. Thesis, Universita di Roma La Sapienza, 2005.
[70] G. Viccione, V. Bovolin, E.P. Carratelli, Defining and optimizing algorithms for neighbouring particle identification in SPH fluid simulations, Int. J. Numer. Methods Fluids 58 (2008) 625–638.

[71] J. Feldman, J. Bonet, Dynamic refinement and boundary contact forces in SPH with applications in fluid flow problems, Int. J. Numer. Methods Eng. 72 (2007) 295–324.
[72] J. Fish, T. Belytschko, A first Course in Finite Elements, Wiley & Sons Ltd., England, 2007.
[73] T.A. Driscoll, B. Fornberg, Interpolation in the limit of increasingly flat radial basis functions, Comput. Math. Appl. 43 (2002) 413–422.
[74] B. Fornberg, G. Wright, E. Larsson, Some observations regarding interpolants in the limit of flat radial basis functions, Comput. Math. Appl. 47 (2004) 37–55.
[75] E. Larsson, B. Fornberg, Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions, Comput. Math. Appl. 49 (2005) 103–130.
[76] J.J. Wang, G.R. Liu, On the optimal shape parameters of radial basis functions used for 2-D meshless methods, Comput. Methods Appl. Mech. Eng. 191 (2002) 2611–2630.