Dissertation for the Degree of
Doctor of Philosophy

Solving various partial differential equations on non-flat surfaces

by

Choi, Yongho

Department of Mathematics

Graduate School
Korea University

December, 2017

김준석 教授指導

博士學位論文

Solving various partial differential equations on non-flat surfaces

이 論文을 理學 博士學位 論文으로 提出함.

2017 年 12 月 日

高麗大學校大學院

數學科

崔龍鎬

崔龍鎬의 理學 博士學位 論文
審査를 完了함.

2017 年 12 月 日

委員長 김 준 석 Jun(印)Kim

委員 양 성 덕 (印)

委員 황 운 재 W.(印) Hwang

委員 김 양 진 (印)

委員 이 현 근 (印)

Contents

Abstract

The primary purpose of this dissertation is to study the various partial differential equations on non-flat surfaces. To solve the partial differential equations (the Allen–Cahn (AC) equation, conservative Allen–Cahn (CAC) equation, and Lengyel–Epstein (LE) equation) on surfaces, we first discuss the AC, CAC, LE equations on the general domain. And then, we describe the surface reconstruction algorithm by using modified AC equation which source data are cloud points, and slice data (CT, MRI, X-ray). That is, reconstruction from two-dimensional data to three-dimensional surface data. Next, we construct computational domain which is defined by narrow band domain and quasi-Neumann boundary condition which applied closest points method. We finally consider solving the AC, CAC, and LE equations on the various surfaces.

We present the fast, efficient, and robust numerical method. We can use the standard Laplacian operator instead of the Laplace–Beltrami operator by using narrow band domain and quasi-Neumann boundary condition. For overall numerical simulations, we use operator splitting method. The multigrid method and explicit method are used in some cases for fast solution. Various numerical results demonstrate that the proposed methods are fast and accurate.

Keywords: PDE on surfaces, narrow band domain, closest points method, surface reconstruction, Allen–Cahn equation, conservative Allen–Cahn equation, Lengyel–Epstein equation.

Acknowledgments

I would like to express sincere thanks and gratitude to my advisor Professor Junseok Kim. His encouragement and unfailing support have been most valuable to me during a period of graduate work, and I feel deeply privileged for the opportunity to work in the research group. I also gratefully acknowledge Professor Seong-Deog Yang, Professor Woonjae Hwang, Professor Yangjin Kim, and Professor Hyun Geun Lee, of my dissertation committee members.

Next, I am grateful to my colleagues who have been for more time than my family in graduate school and have always given me wisdom and courage. Their passion always inspire me with the whole process of my research. Also, I always appreciate all who gave me encouragement and hope. I will do my best to become a more advanced researcher in the future. Please give me wisdom, courage and hope as always.

Above all things, I sincerely thank my parents (Shinchul Choi and Yangsuk Noh) for always growing me up upright and healthy both internally and externally. I am also grateful to my sister, Miyoung Choi who has always been a strength to me, and her husband, Inkyu Lim. Whenever I see cute nephews Juhu and Jian, thank you. I am deeply grateful to my father-in-law Changdeok Kim and my mother-in-law, Gyeongsun Lee. I will do my best to be a better son-in-law. And to my beloved wife, Bokyung Kim, I would like to quote Carl

Sagan and talk. In the vastness of space and the immensity of time, it is my joy to share a planet and an epoch with Bokyung.

Chapter 1

Introduction

The main purpose of this dissertation is solving a partial differential equations (PDEs) on the non-flat surfaces. From this studies, we expect that researches will be applicable on fine dust, convection current, and weather on the earth surface. Also we can study pattern formation in nature such as animals skin, sand dunes and giant's causeway.

The previous studies that have been carried out are as follows. In the nature and applied sciences need solving the PDEs on surfaces. For instance, fluid flow and solidification of a thin film [77], brain imaging [74], diblock copolymers [98], computer graphics for texture synthesis [103], and surfactant distribution on a moving interface [107]. Since an analytic solution is not always present, accurate and efficient numerical approximation is needed. In [88], the authors introduced an embedding method which is closest point method for solving PDEs on surfaces. They use of standard diffusion in higher dimension than the original surface and the use of the closest point representation of the surface. The authors [12] represented a framework for solving PDEs on surfaces. They apply a level set of a higher dimensional function and solve the problems.

Also the pattern formation has been studied by many researchers. Pattern formation based on the Turing model has been one of the notable exceptions since the middle part of the 20th century while acceptance in bioinformatics of mathematical biology has been slower

[71]. Most mathematical models of these patterns are based on a Turing's [102] reaction-diffusion model. The model is composed of two distributed reacting and diffusing chemicals, could generate spatial patterns autonomously. The more detailed previous researches for pattern formations are introduced in Section 5.3.

To solve the PDEs on the surfaces, our detail strategies and implementations are: surface reconstruction algorithm (using modified Allen–Cahn equation, cloud points, and slice data), construct computational domain and boundary condition (narrow band domain, and closest points method), and then solving the Allen–Cahn equation, conservative Allen–Cahn equation, and Lengyel–Epstein equations on the various surfaces.

This Ph.D. dissertation contains the idea and results in the following publications, that the author devoted in the master and doctor course.

- **Y. Choi**, D. Jeong, S. Lee, M. Yoo, and J. Kim, Motion by mean curvature of curves on surfaces using the Allen–Cahn equation, International Journal of Engineering Science 97 (2015) 126–132.
- J. Kim, D. Jeong, S.D. Yang, and **Y. Choi**, A finite difference method for a conservative Allen–Cahn equation on non-flat surfaces, Journal of Computational Physics 334 (2017) 170–181.
- J. Kim, S. Lee, and **Y. Choi**, A conservative Allen–Cahn equation with a space-time dependent Lagrange multiplier, International Journal of Engineering Science 84 (2014) 11–17.
- D. Jeong, Y. Li, **Y. Choi**, M. Yoo, D. Kang, J. Park, J. Choi, and J. Kim, Numerical simulation of the zebra pattern formation on a three-dimensional model, Physica A: Statistical Mechanics and its Applications 475 (2017) 106–116.

- Y. Li, J. Shin, **Y. Choi**, and J. Kim, Three-dimensional volume reconstruction from slice data using phase-field models, Computer Vision and Image Understanding 137 (2015) 115–124.

This dissertation is organized as follows. In Chapter 2, we present the partial differential equations which are Allen–Cahn, conservative Allen–Cahn, and Lengyel–Epstein equations on the general domain. In Chapter 3, we explain surface reconstruction algorithm with modified Allen–Cahn equation and from cloud points, slice data to three-dimensional surface reconstruction. Construction of computational domain and boundary condition are presented in Chapter 4. Numerical results to solve a various partial differential equations on various non-flat surfaces are presented in Chapter 5. We present our conclusions in Chapter 6.

Chapter 2

Partial differential equations

2.1. Allen–Cahn equation

2.1.1. Governing equations. The Allen–Cahn (AC) equation [2] was introduced originally as a phenomenological model for antiphase domain coarsening in a binary alloy:

$$\begin{aligned} \frac{\partial \phi}{\partial t}(\mathbf{x},t) &= -M\left(\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2}-\Delta\phi(\mathbf{x},t)\right),\ \mathbf{x}\in\Omega,\ t>0, &(2.1)\\ \mathbf{n}\cdot\nabla\phi(\mathbf{x},t) &= 0,\ \mathbf{x}\in\partial\Omega. &(2.2)\end{aligned}$$

Here, bounded domain Ω, time t, a positive coefficient M, and unit outer normal vector $\mathbf{n}$ on the domain boundary. Double-well potential energy function $F(\phi) = 0.5\phi^2(1-\phi)^2$ and interfacial energy ϵ which is small positive value. And order parameter $\phi(\mathbf{x},t) \in [0,1]$ which is one of the concentrations of the two components in a binary mixture. That is, $\phi = 1$ in the one phase and $\phi = 0$ in the other phase. Or we can denote $\phi = 1$ in the one phase and $\phi = -1$ in the other phase. When $\phi = 1$ and $\phi = 0$, the interface is defined by $\Gamma = \{\mathbf{x} \in \Omega | \phi(\mathbf{x},t) = 0.5\}$.

On a single closed interface Γ, Allen and Cahn [2] introduced the normal velocity v is governed by its mean curvature $\kappa(\mathbf{x},t)$:

$$v(\mathbf{x},t) = \kappa(\mathbf{x},t),\ \mathbf{x}\in\Gamma. \tag{2.3}$$

The property of above equation has been researched in [11, 68, 86, 93]. Figure 2.1 represents the evolutions for the classical AC equation in two dimension. The perturbed shape (dashed lines) is the initial condition and the temporal evolutions of interfaces (solid lines). The arrow shows the directions of evolutions. Therefore, AC equation does not conserve the mass. We can check analytically:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \phi \,\mathrm{d}\mathbf{x} &= \int_\Omega \phi_t \,\mathrm{d}\mathbf{x} \\
&= \int_\Omega M\left(-\frac{F'(\phi)}{\epsilon^2} + \Delta\phi\right)\mathrm{d}\mathbf{x} \\
&= -\int_\Omega \frac{MF'(\phi)}{\epsilon^2}\,\mathrm{d}\mathbf{x} + \int_{\partial\Omega} M\mathbf{n}\cdot\nabla\phi \,\mathrm{d}s \\
&= -\int_\Omega \frac{MF'(\phi)}{\epsilon^2}\,\mathrm{d}\mathbf{x} \\
&\neq 0,
\end{aligned}$$

which is not always zero. Here, we set $M = 1$ for simplicity.

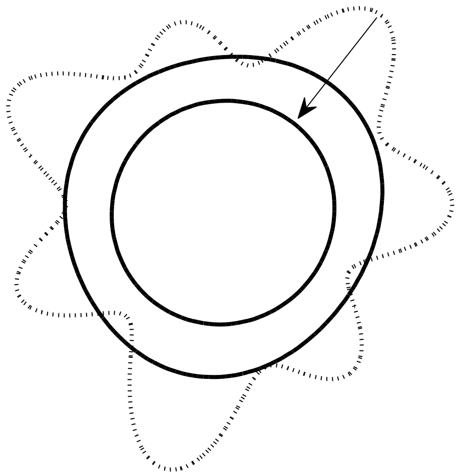


Figure 2.1. Evolutions of perturbed shape with the AC equation. The initial condition is dashed lines.

2.1.2. Discretization. We solve the AC equation. In order to simplify the description, we consider a two-dimensional space. The three-dimensional case is defined analogously. Let a computational domain $\Omega = [a, b] \times [c, d]$ be partitioned into a uniform mesh with spatial step size $h = (b - a)/N_x = (d - c)/N_y$. Here, N_x and N_y are the numbers of cells in x- and y- directions, respectively.

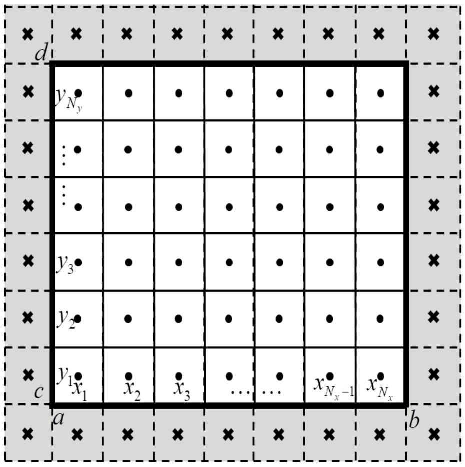


FIGURE 2.2. Schematic of computational domain $\Omega = [a, b] \times [c, d]$, cells, and boundary.

The center of each cell, Ω_{ij}, is located at $\mathbf{x}_{ij} = (x_i, y_j) = (a + (i - 0.5)h, c + (j - 0.5)h)$ for $i = 1, \ldots, N_x$ and $j = 1, \ldots, N_y$. Let ϕ_{ij}^n be approximations of $\phi(x_i, y_j, n\Delta t)$, where $\Delta t = T/N_t$ is the temporal step size, T is the final time, and N_t is the total number of time steps. For the zero Neumann boundary condition, we use

$$\nabla_d \phi_{1/2,j}^n = \nabla_d \phi_{N_x+1/2,j}^n = \nabla_d \phi_{i,1/2}^n = \nabla_d \phi_{i,N_y+1/2}^n = 0,$$

where

$$\nabla_d \phi_{i+1/2,j}^n = \frac{\phi_{i+1,j}^n - \phi_{ij}^n}{h}.$$

A discrete Laplacian operator is defined as

$$\Delta_d \phi_{ij}^n = \frac{\nabla_d \phi_{i+1/2,j}^n - \nabla_d \phi_{i-1/2,j}^n}{h}$$

and the discrete l^2 inner product by

$$(\phi, \psi)_h = h^2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \phi_{ij} \psi_{ij}.$$

We also define the discrete norm as $||\phi||^2 = (\phi, \phi)_h$.

We use an operator splitting method, in which we numerically solve the original problem Eq. (2.1) by solving successively a sequence of simpler problems:

$$\begin{aligned} \phi_t &= \Delta\phi, \qquad (2.4) \\ \phi_t &= -\frac{F'(\phi)}{\epsilon^2}, \qquad (2.5) \end{aligned}$$

First, we solve Eq. (2.4) by applying the implicit Euler's method:

$$\frac{\phi_{ij}^{n+1,1} - \phi_{ij}^n}{\Delta t} = \Delta_d \phi_{ij}^{n+1,1}. \qquad (2.6)$$

We use the multigrid method [4, 14, 101] to solve the implicit discrete Eq. (2.6). We should note that we can use the Crank–Nicolson scheme as in [57] to solve Eq. (2.4). However, although the Crank–Nicolson scheme is unconditionally stable, it is well-known that the scheme suffers from oscillatory behavior with large time steps.

Next, Eq. (2.5) is solved analytically using the method of separation of variables [95]. From $F(\phi) = 0.5\phi^2(1-\phi)^2$, we obtain $F'(\phi) = \phi - 3\phi^2 + 2\phi^3$. Then Eq. (2.5) can be rewritten as follows:

$$\begin{aligned} \frac{d\phi}{dt} &= -\frac{F'(\phi)}{\epsilon^2} \\ &= -\frac{\phi - 3\phi^2 + 2\phi^3}{\epsilon^2} \end{aligned}$$

By using the separation of variables,

$$\begin{aligned}
&-\frac{1}{F'(\phi)}d\phi = \frac{1}{\epsilon^2}dt \\
&\Leftrightarrow -\frac{1}{\phi(1-\phi)(2-\phi)}d\phi = \frac{1}{\epsilon^2}dt \\
&\Leftrightarrow \left(-\frac{1}{\phi} + \frac{1}{1-\phi} - \frac{4}{1-2\phi}\right) d\phi = \frac{1}{\epsilon^2}dt
\end{aligned}$$

And then, integrating both sides,

$$\begin{aligned}
&\int_{\phi_{ij}^{n+1,1}}^{\phi_{ij}^{n+1,2}} \left(-\frac{1}{\phi} + \frac{1}{1-\phi} - \frac{4}{1-2\phi}\right) d\phi = \int_{n\Delta t}^{(n+1)\Delta t} \frac{1}{\epsilon^2}dt, \\
&\Leftrightarrow \ln\left|\frac{(1-2\phi_{ij}^{n+1,2})^2}{\phi_{ij}^{n+1,2}(1-\phi_{ij}^{n+1,2})}\right| - \ln\left|\frac{(1-2\phi_{ij}^{n+1,1})^2}{\phi_{ij}^{n+1,1}(1-\phi_{ij}^{n+1,1})}\right| = \frac{1}{\epsilon^2}\Delta t, \\
&\Leftrightarrow \left|\frac{(1-2\phi_{ij}^{n+1,2})^2}{\phi_{ij}^{n+1,2}(1-\phi_{ij}^{n+1,2})}\right| = \left|\frac{(1-2\phi_{ij}^{n+1,1})^2}{\phi_{ij}^{n+1,1}(1-\phi_{ij}^{n+1,1})}\right| e^{\frac{\Delta t}{\epsilon^2}}.
\end{aligned}$$

It can be rewritten in the following polynomial form,

$$\begin{aligned}
&\left(\frac{\left(1-2\phi_{ij}^{n+1,1}\right)^2}{\phi_{ij}^{n+1,1}\left(1-\phi_{ij}^{n+1,1}\right)} e^{\frac{\Delta t}{\epsilon^2}} + 4\right)\left(\phi_{ij}^{n+1,2}\right)^2 \\
&- \left(\frac{\left(1-2\phi_{ij}^{n+1,1}\right)^2}{\phi_{ij}^{n+1,1}\left(1-\phi_{ij}^{n+1,1}\right)} e^{\frac{\Delta t}{\epsilon^2}} + 4\right)\phi_{ij}^{n+1,2} + 1 = 0.
\end{aligned} \tag{2.7}$$

The polynomial equation (2.7) is solved as

$$\phi_{ij}^{n+1,2} = \frac{1}{2} \pm \frac{\left|1-2\phi_{ij}^{n+1,1}\right|}{2\sqrt{\left(1-2\phi_{ij}^{n+1,1}\right)^2 + 4\phi_{ij}^{n+1,1}\left(1-\phi_{ij}^{n+1,1}\right)e^{-\frac{\Delta t}{\epsilon^2}}}}.$$

From the condition of $\phi_{ij}^{n+1,2}$ in $[0,1]$, we obtain the solution:

$$\phi_{ij}^{n+1,2} = \frac{1}{2} - \frac{1-2\phi_{ij}^{n+1,1}}{2\sqrt{\left(1-2\phi_{ij}^{n+1,1}\right)^2 + 4\phi_{ij}^{n+1,1}\left(1-\phi_{ij}^{n+1,1}\right)e^{-\frac{\Delta t}{\epsilon^2}}}}. \tag{2.8}$$

Thus our scheme [49] can be summarized as

$$\begin{aligned} \frac{\phi_{ij}^{n+1,1}-\phi_{ij}^{n}}{\Delta t} &= \Delta_d \phi_{ij}^{n+1,1}, && (2.9)\\ \phi_{ij}^{n+1,2} &= \frac{1}{2}-\frac{1-2\phi_{ij}^{n+1,1}}{2\sqrt{\left(1-2\phi_{ij}^{n+1,1}\right)^2+4\phi_{ij}^{n+1,1}\left(1-\phi_{ij}^{n+1,1}\right)e^{-\frac{\Delta t}{\epsilon^2}}}}, && (2.10) \end{aligned}$$

2.1.3. Numerical experiments. We simulate numerical experiments: the basic mechanism of the model, comparison of numerical and theoretical results, and the evolution of time in the two- and three-dimensional spaces. The equilibrium order parameter

$$\phi = \frac{1}{2}\left(1+\tanh\left[\frac{x}{2\epsilon}\right]\right)$$

varies from 0.05 to 0.95 over a distance of approximately $4\epsilon\tanh^{-1}(0.9)$ across the interfacial regions. Therefore, if we want this value to be approximately m grid points, then ϵ value is given as the following form [48]

$$\epsilon_m = \frac{hm}{4\tanh^{-1}(0.9)}.$$

2.1.3.1. *Basic mechanism of the model.* The AC Eqs. (2.9)–(2.10) have the following basic mechanism in Fig. 2.3. When we use the initial guess taken elliptic shape (dotted line in Fig. 2.3). The AC Eqs. (2.9) and (2.10) make the initial elliptic shape shrinks to circle under the motion by mean curvature [43]. The higher curvature moves faster than lower curvatures on the curve. That is, the initial ellipse condition goes to the circular shape and finally disappear.

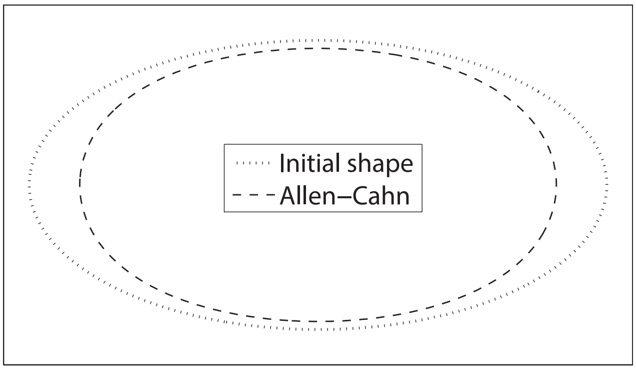
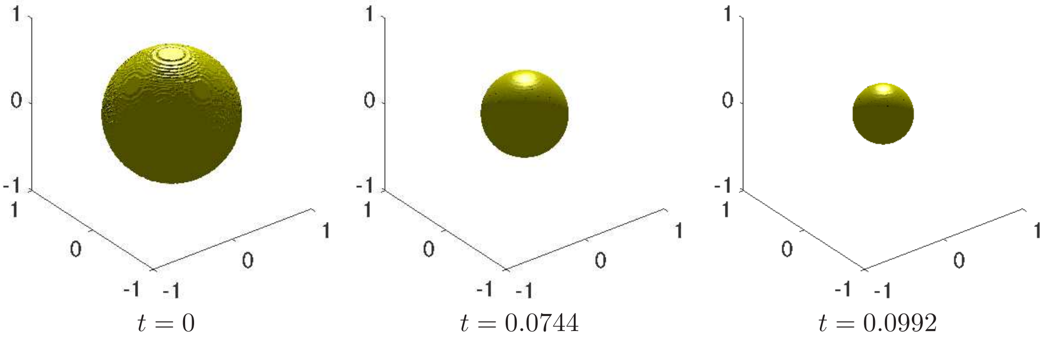


FIGURE 2.3. Basic mechanism of the AC equation.

2.1.3.2. *Comparison with theoretical radius.* We compare the changes in theoretical and numerical radius over time. The parameters are used $128 \times 128 \times 128$ mesh, positive constant value $\epsilon = \epsilon_4$, spatial step size $h = 0.0157$, time step size $\Delta t = 0.1h^2$ in the computational domain $\Omega = (-1, 1) \times (-1, 1) \times (-1, 1)$. Radius of the sphere is $r = 0.7$.



$t = 0$ $t = 0.0744$ $t = 0.0992$

FIGURE 2.4. Shrink the sphere over time. Each time is denoted under the figures.

The size of the theoretically reduced radius over time is defined as [53]

$$R_{anal} = \sqrt{r_0^2 - 4t}$$

where R_{anal} is analytic radius and r_0 is initial radius of sphere.

Figure 2.5 shows the comparison of theoretical and numerical radius over time. And we obtain that the theoretical and numerical results are matched.

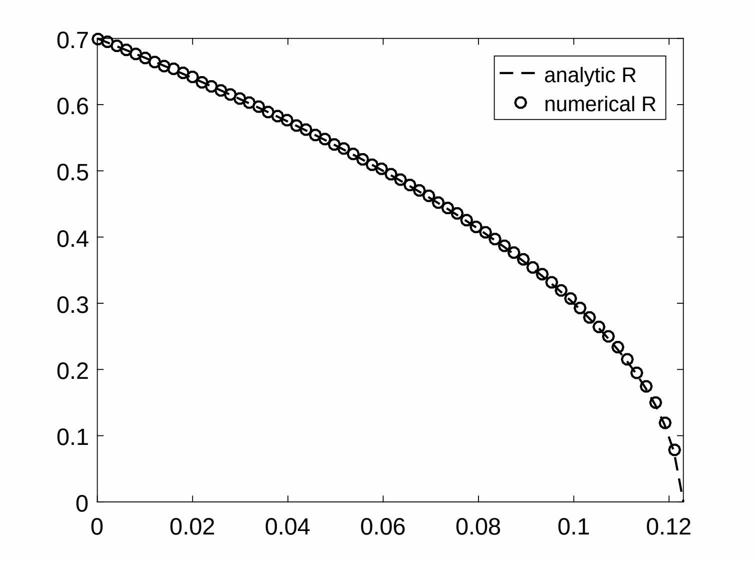


FIGURE 2.5. Comparison of the theoretical and numerical radius.

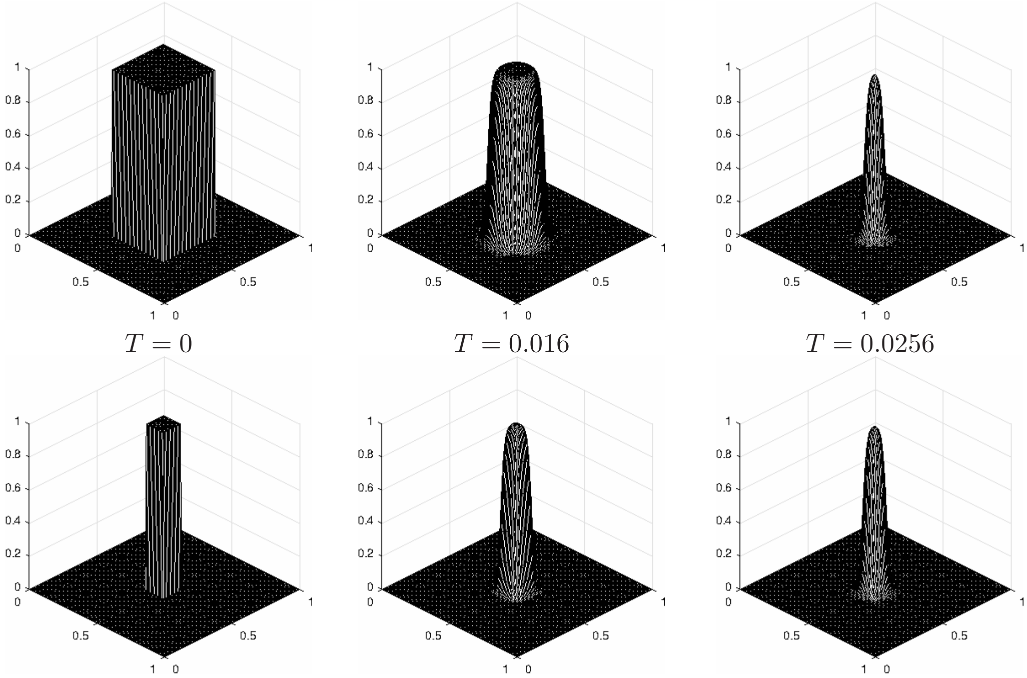
2.1.3.3. *Temporal evolution in the two- and three-dimensional spaces.* We simulate the simple test in the two- and three-dimensional spaces. On a computational domain $\Omega = [0,1] \times [0,1]$, we use 128×128 mesh grid, and the two initial conditions are given as

$$(i) \quad \phi_{ij} = 1 \quad \text{if} \quad 40 \leq i, j \leq 88,$$

$$(ii) \quad \phi_{ij} = 1 \quad \text{if} \quad 56 \leq i, j \leq 72,$$

and otherwise we take $\phi_{ij} = 0$ (see Fig. 2.6(a)). The time step size is chosen as $\Delta t =$ 1.0e-5.

Figures 2.6(b) and (c) show the numerical time evolution results for large initial object and small initial object. As time goes by, both shape will be disappeared.



$T = 0$ $T = 0.016$ $T = 0.0256$

$T = 0$ $T = 0.0003$ $T = 0.0011$

(a) (b) (c)

FIGURE 2.6. (a) Initial conditions with two different shapes. (b) and (c) are time evolution results. Each simulation time are denoted under the figures.

Next, we simulate numerical simulations of three-dimensional cubes in Fig. 2.7. The initial conditions are given on $\Omega = [0, 1] \times [0, 1] \times [0, 1]$ with $h = 1/128$ as

$$(i) \quad \phi_{ijk} = 1, \quad \text{if} \quad 40 \leq i, j, k \leq 88,$$

$$(ii) \quad \phi_{ijk} = 1, \quad \text{if} \quad 56 \leq i, j, k \leq 72,$$

and $\phi_{ijk} = 0$ otherwise (see Fig. 2.7(a) for the isosurfaces $\phi = 0.5$). The time step $\Delta t = 1.0$e-5 is used. The three-dimensional results are also the same tendency as the two-dimensional result.

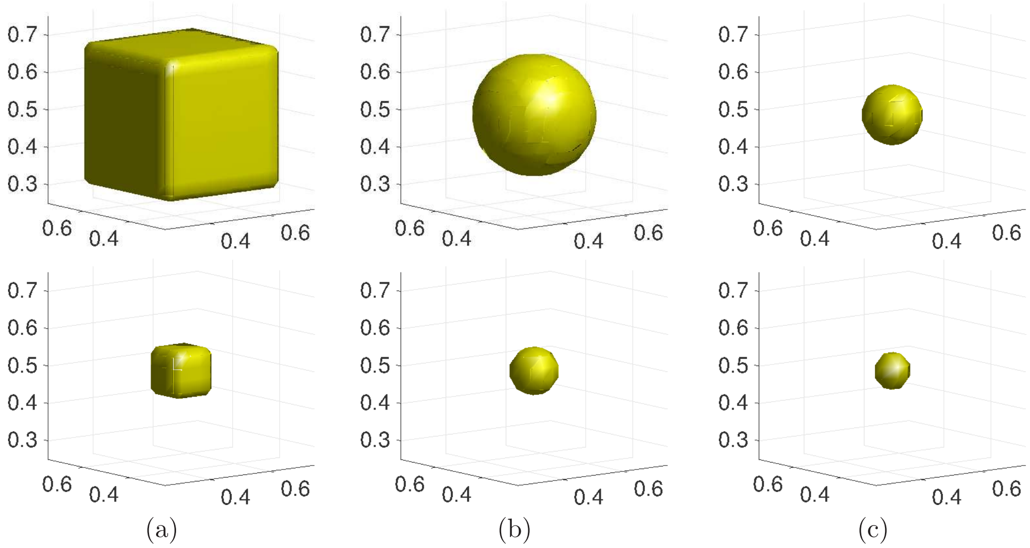


FIGURE 2.7. Temporal evolution of objects. (a) Initial condition, (b) top: $t = 0.01$, row: $t = 0.0007$, (c) top: $t = 0.015$, row: $t = 0.0011$.

2.1.4. Summary. In this section, we explain the Allen–Cahn equation and show various numerical results. The Allen–Cahn equation was originally introduced for antiphase domain coarsening in a binary alloy [2]. The original Allen–Cahn equation and modified formulae have been applied to a various problems: image analysis [13], motion by mean curvature [22, 25], fluid flows [108], phase transitions [2], and crystal growth [24]. And the Allen–Cahn equation has some properties: energy decreasing, non-mass conservation, shrink the object, and phase separation. We also show these properties on the non-flat surfaces in Chapter 5.

2.2. Conservative Allen–Cahn equation

2.2.1. Governing equations. The original AC equation does not conserve the mass that the detail explain in Section 2.1. Rubinstein and Sternberg [87] introduced a Lagrange multiplier $\beta(t)$ into the AC model to preserve the mass as follows

$$\frac{\partial \phi}{\partial t}(\mathbf{x},t) = -\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2} + \Delta\phi(\mathbf{x},t) + \beta(t). \tag{2.11}$$

To keep the mass conservation, $\beta(t)$ must satisfy

$$\beta(t) = \frac{\int_\Omega F'(\phi(\mathbf{x},t))\,\mathrm{d}\mathbf{x}}{\epsilon^2 \int_\Omega \mathrm{d}\mathbf{x}}.$$

Equation (2.11) has been widely used [6, 108, 114]. The normal velocity v on a single closed interface Γ is given by the volume-preserving mean curvature flow:

$$v(\mathbf{x},t) = \kappa(\mathbf{x},t) - \frac{1}{|\Gamma|}\int_\Gamma \kappa\,\mathrm{d}s,\ \ \mathbf{x} \in \Gamma, \tag{2.12}$$

where $|\Gamma|$ is the total length of curve in two-dimensional space and the total area in three-dimensional space. Rubinstein and Sternberg's model has been various researched [7, 11, 15, 104, 106, 112]. However, since Lagrangian multiplier is a function of the time variable, there are drawbacks to preserving small features. For example, there is a critical radius of drop which eventually disappears below the radius. This phenomenon is observed in the frame of the Cahn–Hilliard model [112].

The main purpose of this study is to propose a practically unconditionally stable numerical scheme for the conservative AC equation with a space-time dependent Lagrange multiplier. The scheme is based on the recently developed hybrid scheme for the AC equation [57] with an exact mass-conserving update at each time step.

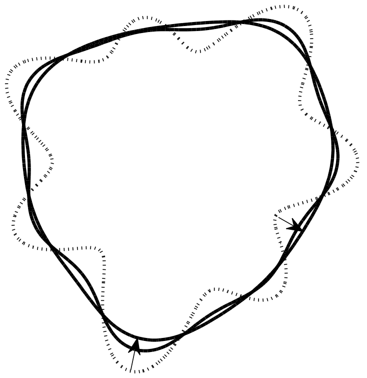


FIGURE 2.8. Temporal evolutions of arbitrary curves with the conservative AC equation. The initial curves (dashed lines) and directions of time evolutions are indicated by arrows.

The authors in [11] proposed the following conservative AC equation which guarantees to preserve small geometric features:

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = -\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2} + \Delta\phi(\mathbf{x},t) + \beta(t)\sqrt{2F(\phi(\mathbf{x},t))}, \tag{2.13}$$

where $\beta(t) = \int_\Omega F'(\phi(\mathbf{x},t))\,\mathrm{d}\mathbf{x}/[\epsilon^2 \int_\Omega \sqrt{2F(\phi(\mathbf{x},t))}\,\mathrm{d}\mathbf{x}]$. Then, the solution $\phi(\mathbf{x},t)$ of the conservative AC Eq. (2.13) possesses the total mass conservation property, i.e.,

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \phi\,\mathrm{d}\mathbf{x} &= \int_\Omega \phi_t\,\mathrm{d}\mathbf{x} \\
&= \int_\Omega \left[-\frac{F'(\phi)}{\epsilon^2} + \Delta\phi + \beta(t)\sqrt{2F(\phi)}\right]\mathrm{d}\mathbf{x} \\
&= -\frac{1}{\epsilon^2}\int_\Omega F'(\phi)\,\mathrm{d}\mathbf{x} + \int_{\partial\Omega} \mathbf{n}\cdot\nabla\phi\,\mathrm{d}s + \beta(t)\int_\Omega \sqrt{2F(\phi)}\,\mathrm{d}\mathbf{x} \\
&= -\frac{1}{\epsilon^2}\int_\Omega F'(\phi)\,\mathrm{d}\mathbf{x} + \beta(t)\int_\Omega \sqrt{2F(\phi)}\,\mathrm{d}\mathbf{x} \\
&= 0,
\end{aligned}$$

where we used the zero Neumann boundary condition (2.2). We note that to conserve mass, there is a classical model such as the Cahn–Hilliard equation [1, 17, 41, 66]. Also, see

[42] on the development and generalizations of Allen–Cahn and Stefan equations within a thermodynamic framework.

2.2.2. Discretization. In this section, we propose a hybrid numerical algorithm for solving the conservative AC equation. For simplify to explain, we explain in a two-dimensional space. The three-dimensional case is treated analogously. Computational domain $\Omega = [a, b] \times [c, d]$, spatial step size $h = (b - a)/N_x = (d - c)/N_y$ which N_x and N_y are the numbers of cells in x- and y- directions, respectively. The center of each cell, Ω_{ij}, is located at $\mathbf{x}_{ij} = (x_i, y_j) = (a + (i - 0.5)h, c + (j - 0.5)h)$ for $i = 1, \ldots, N_x$ and $j = 1, \ldots, N_y$. Let ϕ_{ij}^n be approximations of $\phi(x_i, y_j, n\Delta t)$, where $\Delta t = T/N_t$ is the temporal step size, T is the final time, and N_t is the total number of time steps.

We use zero Neumann boundary condition, $\nabla_d\phi_{1/2,j}^n = \nabla_d\phi_{N_x+1/2,j}^n = \nabla_d\phi_{i,1/2}^n = \nabla_d\phi_{i,N_y+1/2}^n = 0$, where $\nabla_d\phi_{i+1/2,j}^n = (\phi_{i+1,j}^n - \phi_{ij}^n)/h$. We define a discrete Laplacian operator by $\Delta_d\phi_{ij}^n = (\nabla_d\phi_{i+1/2,j}^n - \nabla_d\phi_{i-1/2,j}^n)/h$ and the discrete l^2 inner product by

$$(\phi, \psi)_h = h^2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \phi_{ij}\psi_{ij}.$$

We also define the discrete norm as $||\phi||^2 = (\phi, \phi)_h$. We use an operator splitting method which is numerically solve the original Eq. (2.13) by solving successively a sequence of simpler problems:

$$\phi_t = \Delta\phi, \tag{2.14}$$

$$\phi_t = -\frac{F'(\phi)}{\epsilon^2}, \tag{2.15}$$

$$\phi_t = \beta\sqrt{2F(\phi)}. \tag{2.16}$$

First, we solve Eq. (2.14) by applying the implicit Euler's method:

$$\frac{\phi_{ij}^{n+1,1} - \phi_{ij}^{n}}{\Delta t} = \Delta_d \phi_{ij}^{n+1,1}. \tag{2.17}$$

To solve the implicit discrete Eq. (2.17), we use the multigrid method [4, 14, 101].

Next, Eq. (2.15) is solved analytically using the method of separation of variables [95] that more detailed explain in section 2.1.1. We obtain the solution:

$$\phi_{ij}^{n+1,2} = \frac{1}{2} - \frac{1 - 2\phi_{ij}^{n+1,1}}{2\sqrt{\left(1 - 2\phi_{ij}^{n+1,1}\right)^2 + 4\phi_{ij}^{n+1,1}\left(1 - \phi_{ij}^{n+1,1}\right)e^{-\frac{\Delta t}{\epsilon^2}}}}. \tag{2.18}$$

Finally, we discretize Eq. (2.16) as

$$\frac{\phi_{ij}^{n+1} - \phi_{ij}^{n+1,2}}{\Delta t} = \beta^{n+1,2}\sqrt{2F(\phi_{ij}^{n+1,2})}. \tag{2.19}$$

By Eq. (2.19), we get

$$\phi_{ij}^{n+1} = \phi_{ij}^{n+1,2} + \Delta t \beta^{n+1,2}\sqrt{2F(\phi_{ij}^{n+1,2})},$$

then by the property of mass conservation

$$\begin{aligned} \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\phi_{ij}^{0} &= \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\phi_{ij}^{n+1} \\ &= \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\left(\phi_{ij}^{n+1,2} + \Delta t \beta^{n+1,2}\sqrt{2F(\phi_{ij}^{n+1,2})}\right). \end{aligned} \tag{2.20}$$

Therefore,

$$\beta^{n+1,2} = \frac{1}{\Delta t}\frac{\sum\limits_{i=1}^{N_x}\sum\limits_{j=1}^{N_y}\left(\phi_{ij}^{0} - \phi_{ij}^{n+1,2}\right)}{\sum\limits_{i=1}^{N_x}\sum\limits_{j=1}^{N_y}\sqrt{2F(\phi_{ij}^{n+1,2})}}. \tag{2.21}$$

Our proposed scheme [49] can be summarized as

$$
\begin{aligned}
\frac{\phi_{ij}^{n+1,1}-\phi_{ij}^{n}}{\Delta t} &= \Delta_d \phi_{ij}^{n+1,1}, && (2.22)\\
\phi_{ij}^{n+1,2} &= \frac{1}{2}-\frac{1-2\phi_{ij}^{n+1,1}}{2\sqrt{\left(1-2\phi_{ij}^{n+1,1}\right)^2+4\phi_{ij}^{n+1,1}\left(1-\phi_{ij}^{n+1,1}\right)e^{-\frac{\Delta t}{\epsilon^2}}}}, && (2.23)\\
\phi_{ij}^{n+1} &= \phi_{ij}^{n+1,2}+\Delta t\beta^{n+1,2}\sqrt{2F(\phi_{ij}^{n+1,2})}. && (2.24)
\end{aligned}
$$

2.2.3. Numerical experiments. We simulate various numerical experiments: the basic mechanism of the CAC equaion, a comparison with previous model, and the evolution of drops in the two- and three-dimensional spaces. The equilibrium order parameter $\phi = 0.5(1+\tanh[x/(2\epsilon)])$ varies from 0.05 to 0.95 over a distance of approximately $4\epsilon\tanh^{-1}(0.9)$ across the interfacial regions. Therefore, if we want this value to be approximately m grid points, then ϵ value is given as [48], $\epsilon_m = hm/4\tanh^{-1}(0.9)$. We shall use ϵ_8 if not otherwise specified.

2.2.3.1. *Basic mechanism of the model.* The basic mechanism of the algorithm Eqs. (2.22)–(2.24) is illustrated in Fig. 2.9. Let us consider an elliptical initial condition (dotted line in Fig. 2.9). If we take only Eqs. (2.22) and (2.23), then the initial shape shrinks (dashed line) [43]. The higher curvature moves faster than lower curvatures on the curve. However, with the mass correction step Eq. (2.24), the curve uniformly moves to the outward normal direction (solid line). The initial ellipse shape goes to the circular shape with the same mass.

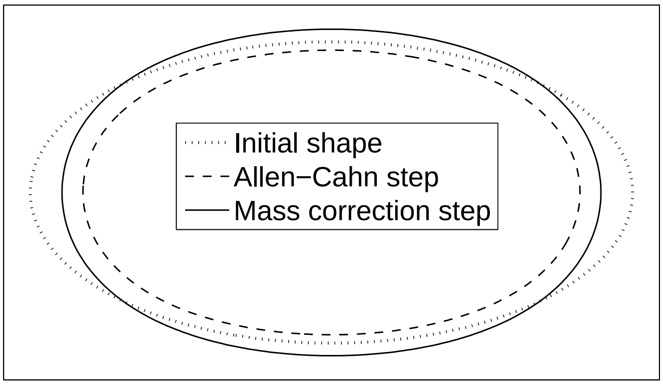


FIGURE 2.9. Basic mechanism of the proposed numerical scheme.

2.2.3.2. *Comparison of two models.* To see the difference between two models original AC Eqs. (2.11) and CAC Eqs. (2.13), we consider the following numerical experiments. Mesh grid of 128 × 128, the initial conditions are given as (i) $\phi_{ij} = 1$ if $40 \leq i, j \leq 88$, (ii) $\phi_{ij} = 1$ if $56 \leq i, j \leq 72$, and $\phi_{ij} = 0$ otherwise (see Fig. 2.10(a)) on a computational domain $\Omega = [0, 1] \times [0, 1]$. The time step size $\Delta t = 1.0$e-5.

Figures 2.10(b) and (c) show the numerical results of original AC Eqs. (2.11) and CAC Eqs. (2.13) at a steady state with two different initial conditions, respectively. The numerical steady state is defined as the state when the discrete l_2 norm of the difference between ϕ^{n+1} and ϕ^n becomes less than $tol = 1.0$e-6. Observing the numerical results in the top row of Fig. 2.10, we can see that both models work well when the initial feature is large enough. It should be noted that the order parameter in the outside phase is 0.009 for Eq. (2.11), on the other hand, the value is 0.0 for Eq. (2.13) with our proposed numerical scheme. The reason why the order parameters have different values is that our scheme corrects mass loss in the interfacial region. If the geometry is small, then the geometry disappears with Eq.

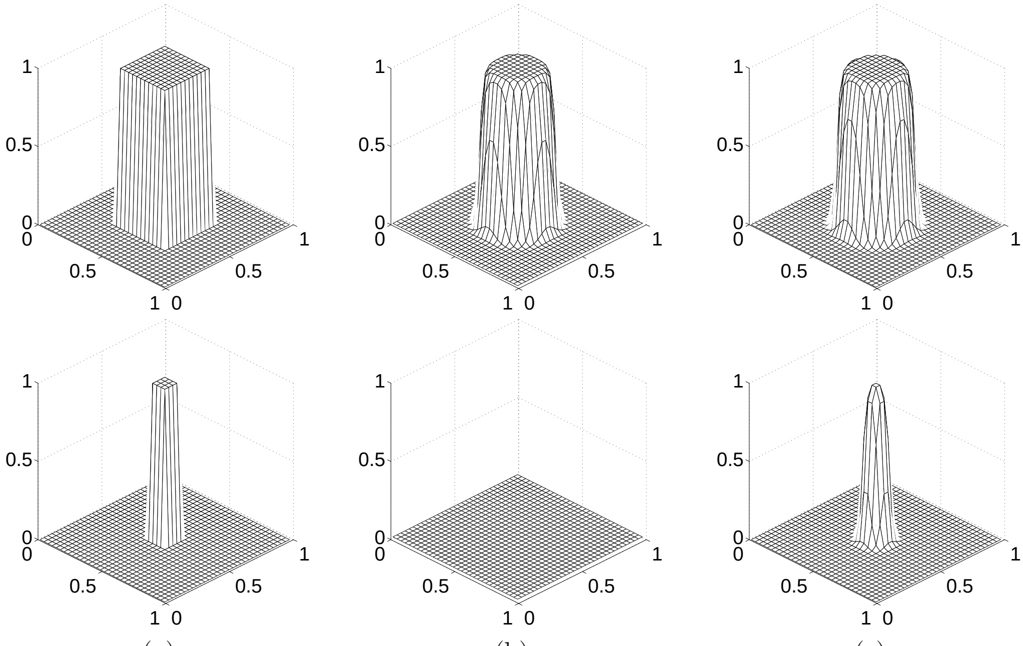
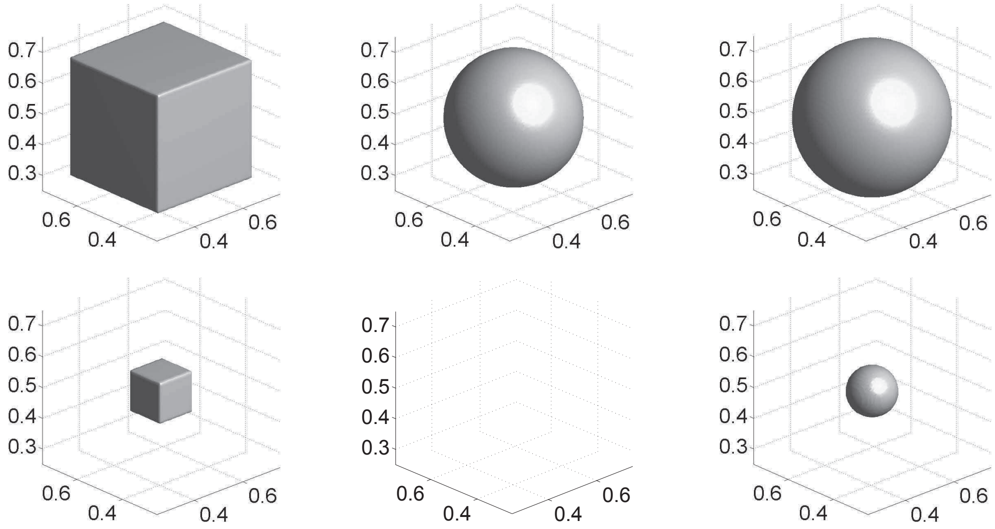


Figure 2.10. (a) Initial conditions with two different shapes. (b) and (c) are numerical results from Eqs. (2.11) and (2.13), respectively.

(2.11) (see the second row of Fig. 2.10(b)). On the other hand, with our scheme, the drop stays as shown in the second row of Fig. 2.10(c).

Next, we simulate numerical tests of three-dimensional cubes in Fig. 2.11. The initial conditions are given on $\Omega = [0,1] \times [0,1] \times [0,1]$ with $h = 1/128$ as (i) $\phi_{ijk} = 1$ if $40 \leq i,j,k \leq 88$, (ii) $\phi_{ijk} = 1,$ if $56 \leq i,j,k \leq 72$, and $\phi_{ijk} = 0$ otherwise (see Fig. 2.11(a) for the isosurfaces $\phi = 0.5$). The temporal step size $\Delta t = 1.0\text{e-}5$ is used. Figures 2.11(b) and (c) show the steady states with Eqs. (2.11) and (2.13), respectively. The three-dimensional results are almost similar to the two-dimensional ones.



(a) (b) (c)

FIGURE 2.11. (a) Two different-sized cubes for initial conditions. (b) and (c) are numerical results from Eqs. (2.11) and (2.13), respectively.

2.2.3.3. *Evolution of disks.* In [15], the authors gave the evolution law for radii of spheres in n-dimensional geometric flows. For the m interfaces of radii r_i for $i = 1, 2, \ldots, m$ with $r_j < r_{j+1}$ for $j = 1, 2, \ldots, m-1$, the equations of evolution in n-dimensional case are given by

$$\frac{\mathrm{d}r_i}{\mathrm{d}t} = (n-1)\left(\frac{\sum_{k=1}^{m} r_k^{n-2}}{\sum_{k=1}^{m} r_k^{n-1}} - \frac{1}{r_i}\right), \quad i = 1, 2, \ldots, m.$$

We consider two disjoint circular interfaces in two-dimensional space. Assume that the two interfaces have radii r and R with $r < R$, then the equations of evolution become

$$\frac{\mathrm{d}r}{\mathrm{d}t} = \frac{2}{r+R} - \frac{1}{r}, \tag{2.25}$$

$$\frac{\mathrm{d}R}{\mathrm{d}t} = \frac{2}{r+R} - \frac{1}{R}. \tag{2.26}$$

From the above equations, we can get the time t_f at which smaller circle disappears by solving a system of ordinary differential equations [11]:

$$t_f = -\frac{1}{2}r_0 R_0 + \frac{1}{4}\left(r_0^2 + R_0^2\right)\ln\left(1 + \frac{2r_0 R_0}{(R_0 - r_0)^2}\right), \tag{2.27}$$

where r_0 and R_0 are the initial radii. We present results for $r_0 = 0.1$ and $R_0 = 0.15$ using a temporal step size $\Delta t = 1.1264 \times 10^{-4}$ on $\Omega = [0, 1] \times [0, 1]$ with a mesh grid 128×128. Then $t_f = 0.0133$ by Eq. (2.27). For the reference solutions of r and R, we numerically solve the ordinary differential equations by using the fourth order Runge–Kutta method [10, 16, 113].

In Fig. 2.12, the solid lines represent the result from the Runge–Kutta method, dot and star represent the radius evolutions of R and r with Eq. (2.13), respectively, and circle and diamond also represent the radius evolutions of R_p and r_p with Eq. (2.11), respectively.

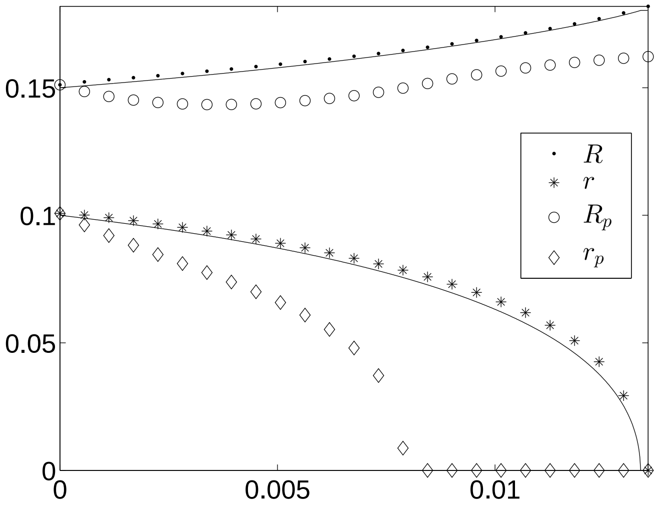


FIGURE 2.12. Evolution of the radii of two distinct circles against time. R and r are radii from Eq. (2.13) and R_p and r_p are radii from Eq. (2.11). The solid lines are the corresponding reference solutions.

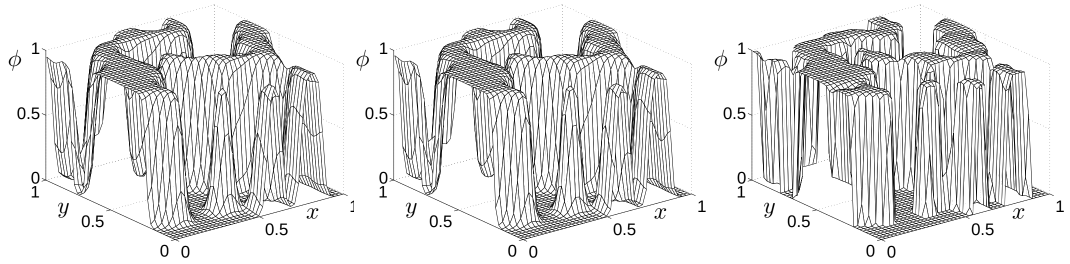
As shown in Fig. 2.12, R grows monotonically with our numerical scheme and r disappears at the similar time as predicted from the analytic calculation. Compared to Eq.

(2.13), the results from Eq. (2.11) do not predict the theoretical prediction because most mass diffuse into the bulk phase from a global mass conservative Lagrange multiplier.

2.2.3.4. *Practically unconditional stability test.* We test a numerical simulation to demonstrate the practically unconditional stability of the proposed scheme. The initial condition use random perturbation around 0.5 as follows:

$$\phi(x, y, 0) = \frac{1}{2} + 0.02\text{rand}(x, y),$$

where $\text{rand}(x, y)$ is a random number between -1 and 1. We use a mesh grid size 128×128 on the computational domain $\Omega = [0, 1] \times [0, 1]$ with three different time steps, $\Delta t = 0.1h^2$, h^2, and $10h^2$. Figures 2.13(a), (b), and (c) are snapshots at the same time $T = 50h^2$ with three different time steps.



(a) $\Delta t = 0.1h^2$ (b) $\Delta t = h^2$ (c) $\Delta t = 10h^2$

FIGURE 2.13. The time steps are shown below each figure and results are at the same time $T = 50h^2$.

From the results, proposed scheme is demonstrated the practically unconditionally stable. We should note that the value of Δt is typically smaller than $0.1h^2$ to get accurate numerical approximations. Otherwise, the numerical scheme may unnecessarily result in large discretization errors. Therefore, the fact that we can use two orders of magnitudes larger time step than $\Delta t = 0.1h^2$ suggests the proposed scheme is practically unconditionally stable.

2.2.4. Summary. In this section, we proposed a new numerical scheme for solving the conservative Allen–Cahn equation with a space-time dependent Lagrange multiplier. Rubinstein and Sternberg's nonlocal Allen–Cahn equation conserves mass. However, with their model it is difficult to keep small features since they dissolve into the bulk region because mass conservation is realized by a global correction using the time-dependent Lagrange multiplier. To resolve the problem, we used a space-time dependent Lagrange multiplier to preserve the volume of the system and proposed a practically unconditionally stable hybrid scheme to solve the model. We performed numerical experiments such as the basic mechanism of the model, a comparison with previous model, the temporal evolution of drops in two- and three-dimensional spaces, and a practically unconditional stability test of the proposed numerical scheme. The numerical results indicate a potential usefulness of our proposed numerical scheme for accurately calculating geometric features of interfaces. In particular, it is applicable to various problems with a mass conservation constraint.

2.3. Lengyel–Epstein equation

2.3.1. Governing equations. We consider the following Lengyel–Epstein equation [54]:

$$\begin{aligned}\frac{\partial u}{\partial t} &= D_u\Delta_{\mathcal{S}}u + f(u,v)\\ &= D_u\Delta_{\mathcal{S}}u + k_1\left(v - \frac{uv}{1+v^2}\right), \qquad (2.28)\\ \frac{\partial v}{\partial t} &= D_v\Delta_{\mathcal{S}}v + g(u,v)\\ &= D_v\Delta_{\mathcal{S}}v + k_2 - v - \frac{4uv}{1+v^2}, \qquad (2.29)\end{aligned}$$

where $u(\mathbf{x},t)$ and $v(\mathbf{x},t)$ are concentrations of an inhibitor and an activator at position $\mathbf{x}\in\mathcal{S}$ and time t, respectively. Here, $\Delta_{\mathcal{S}}$ denotes the Laplace–Beltrami operator, D_u and D_v are the diffusion coefficients, and k_1 and k_2 are positive constants related to the feed concentrations. In this section, we test on the two-dimensional problem, so we use the surface Laplacian operator $\Delta_{\mathcal{S}}$ by the standard Laplacian operator Δ. Then we can rewrite the Eqs. (2.28) and (2.29) as follows:

$$\begin{aligned}\frac{\partial u}{\partial t} &= D_u\Delta u + k_1\left(v - \frac{uv}{1+v^2}\right), \qquad (2.30)\\ \frac{\partial v}{\partial t} &= D_v\Delta v + k_2 - v - \frac{4uv}{1+v^2}. \qquad (2.31)\end{aligned}$$

2.3.2. Discretization. We present a numerical algorithm for the Lengyel–Epstein model. We discretize the reaction-diffusion equation in a two-dimensional domain $\Omega = (a,b)\times(c,d)$. Let N_x and N_y be positive integers, $h=(b-a)/N_x=(d-c)/N_y$ be the uniform mesh size, and

$$\Omega^h = \{\mathbf{x}_{ij} = (x_i,y_j) = (a+hi, c+hj)\big|0\le i\le N_x, 0\le j\le N_y\}$$

be the discrete domain. Let u_{ij}^n and v_{ij}^n be approximations of $u(x_i, y_j, n\Delta t)$ and $v(x_i, y_j, n\Delta t)$, where $\Delta t = T/N_t$ is the time step, T is the final time, and N_t is the total number of time steps. We consider the discretization of the reaction-diffusion system (2.30) and (2.31) using explicit scheme,

$$\begin{aligned} \frac{u_{ij}^{n+1} - u_{ij}^n}{\Delta t} &= D_u \Delta_h u_{ij}^n + k_1 \left(v_{ij}^n - \frac{u_{ij}^n v_{ij}^n}{1 + (v_{ij}^n)^2} \right), && (2.32) \\ \frac{v_{ij}^{n+1} - v_{ij}^n}{\Delta t} &= D_v \Delta_h v_{ij}^n + k_2 - v_{ij}^n - \frac{4 u_{ij}^n v_{ij}^n}{1 + (v_{ij}^n)^2}, && (2.33) \end{aligned}$$

Here, the descretization of Laplacian is given by

$$\Delta_h u_{ij} = \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}}{h^2}.$$

and

$$\Delta_h v_{ij} = \frac{v_{i+1,j} + v_{i-1,j} + v_{i,j+1} + v_{i,j-1} - 4v_{ij}}{h^2}.$$

2.3.3. Numerical experiments.

2.3.3.1. *Linear stability analysis.* In this section, we study the linear stability analysis for Eqs. (2.30) and (2.31). We seek a solution of the form,

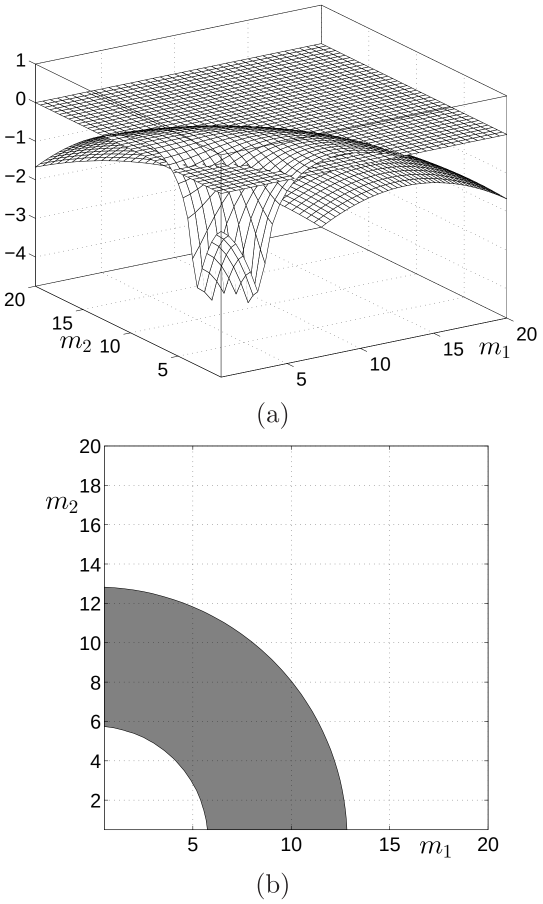
$$\begin{aligned} u(x, y, t) &= \bar{u} + \alpha_{m_1, m_2}(t) \cos\left(\frac{2\pi m_1 x}{L_x}\right) \cos\left(\frac{2\pi m_2 y}{L_y}\right), && (2.34) \\ v(x, y, t) &= \bar{v} + \beta_{m_1, m_2}(t) \cos\left(\frac{2\pi m_1 x}{L_x}\right) \cos\left(\frac{2\pi m_2 y}{L_y}\right), && (2.35) \end{aligned}$$

where $\Omega = (0, L_x) \times (0, L_y)$ and $f(\bar{u}, \bar{v}) = g(\bar{u}, \bar{v}) = 0$. Substituting Eqs. (2.34) and (2.35) into the linearized equations of Eqs. (2.30) and (2.31) yields

$$\frac{d}{dt} \begin{pmatrix} \alpha_{m_1, m_2}(t) \\ \beta_{m_1, m_2}(t) \end{pmatrix} = A \begin{pmatrix} \alpha_{m_1, m_2}(t) \\ \beta_{m_1, m_2}(t) \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \alpha_{m_1, m_2}(t) \\ \beta_{m_1, m_2}(t) \end{pmatrix}, \qquad (2.36)$$

where A is a 2×2 matrix whose components are given as

$$\begin{aligned} a &= -D_u\left[\left(\frac{2\pi m_1}{L_x}\right)^2+\left(\frac{2\pi m_2}{L_y}\right)^2\right]-\frac{k_1\bar{v}}{1+\bar{v}^2}, && (2.37)\\ b &= k_1\left[1-\frac{\bar{u}(1-\bar{v}^2)}{(1+\bar{v}^2)^2}\right], \\ c &= -\frac{4\bar{v}}{1+\bar{v}^2}, && (2.38)\\ d &= -D_v\left[\left(\frac{2\pi m_1}{L_x}\right)^2+\left(\frac{2\pi m_2}{L_y}\right)^2\right]-1-\frac{4\bar{u}(1-\bar{v}^2)}{(1+\bar{v}^2)^2}. \end{aligned}$$



(a)

(b)

FIGURE 2.14. (a) Mesh plot and (b) filled contour plot of max($Re(\lambda_1)$, $Re(\lambda_2)$). Shaded area is the positive value of max($Re(\lambda_1)$, $Re(\lambda_2)$).

Figures 2.14(a) and (b) show the mesh plot and the contour plot of $\max(Re(\lambda_1),\ Re(\lambda_2))$, respectively, where λ_1 and λ_2 are the eigenvalues of the matrix A. For a better visualization, we put the zero plane together. Here, we used the parameters: $D_u = 1$, $D_v = 0.02$, $k_1 = 9$, $k_2 = 11$, $\bar{u} = 1 + 0.04k_2^2$, $\bar{v} = 0.2k_2$, $L_x = 10$, and $L_y = 10$.

2.3.3.2. *Pattern on two-dimensional rectangular domain.* We first numerically solve Eqs. (2.32) and (2.33) on a two-dimensional rectangular domain $\Omega = [0, 10] \times [0, 10]$ using $D_u = 1$, $k_2 = 11$, a mesh grid 101×101, $h = 0.1$, and $\Delta t = 0.1h^2$. Here, periodic boundary conditions in each direction are used. Initial conditions for u and v are

$$u(x, y, 0) = \bar{u} + 0.1\text{rand}(x, y),$$

$$v(x, y, 0) = \bar{v} + 0.1\text{rand}(x, y),$$

where $\bar{u} = 1 + 0.04k_2^2$, $\bar{v} = 0.2k_2$, and $\text{rand}(x, y)$ is a random number between -1 and 1.

Figures 2.15(a) and (b) show the temporal evolutions of the activator concentration (v) when we use $D_v = 0.04$, $k_1 = 7$ and $D_v = 0.02$, $k_1 = 9$, respectively. The times are shown below each column. We can observe spots and stripes depending on the parameters as shown in [79]. Note that Othmer et al. [79] showed more patterns observed in animal skin, including spots, stripes, reticulated stripes, and inverted spots.

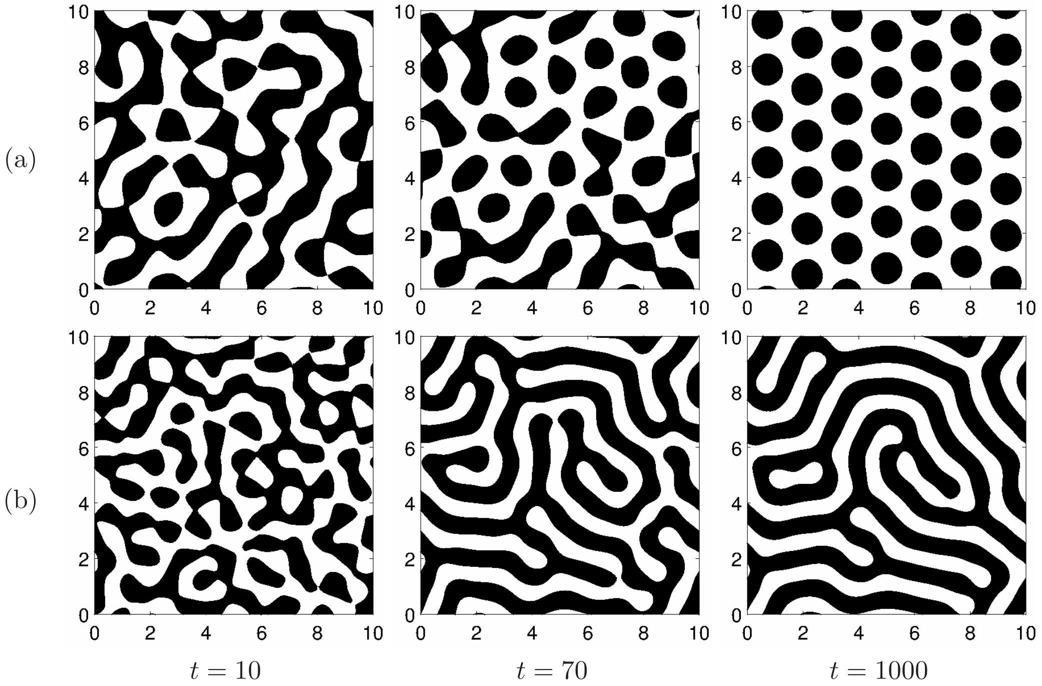


FIGURE 2.15. Temporal evolution of activator concentration v with $D_u = 1$ and $k_2 = 11$ on two-dimensional domain $\Omega = [0, 10] \times [0, 10]$. Here, (a) $D_v = 0.04$, $k_1 = 7$ and (b) $D_v = 0.02$, $k_1 = 9$ are used.

2.3.4. Summary. We numerically simple studied the Lengyel–Epstein equation which with a two-component activator-inhibitor system of reaction-diffusion equations. We discretized the governing equations using a finite difference method and show the linear stability analysis. Also we represent the pattern on two-dimensional rectangular domain. In that simulation, we obtain the hexagonal and lamella pattern. This results be applied on the surface in the Chapter 5.3.

Chapter 3

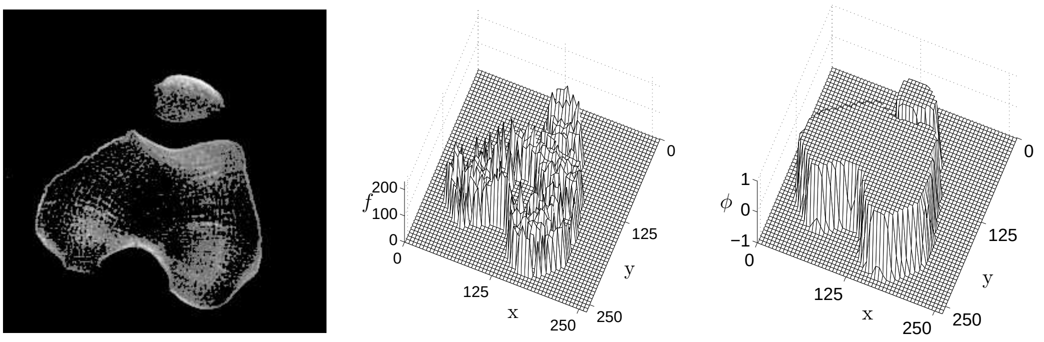
Surface reconstruction algorithm

3.1. Introduction

Developing reconstruction algorithms have been attracted and considered important research area. Three-dimensional (3D) volume reconstruction from a sequence of medical images (CT, MRI, X-ray) has various applications such as medical diagnostic, plastic and artificial limb surgery, virtual surgery system, anatomy teaching, and treatment planning [23, 27]. Various algorithms have been proposed to reconstruct a surface or volume from a set of planar cross-sections. A method which combines the elastic interpolation algorithm, spline theory, and surface consistency theorem was proposed for reconstructing a smooth 3D object from serial cross sections [65]. Guo *et al.* presented a morphology-based mathematical method to implement the interpolation by means of a combined operation of weighted dilation and erosion [37]. Jones and Chen constructed surfaces from cross sections using a field function in each slice and the marching cubes algorithm to generate a surface consisting of polygonal facets [45]. In shape-based interpolation method, the signed distance value of a voxel to the edges of a cross section is calculated. After each slice has been assigned the distance values, distances for other slices are defined using an interpolation. Then, the volume is obtained by the zero isosurface [85]. For other approaches to 3D reconstruction, refer to [33, 38, 56, 110].

3.2. Image segmentation with modified Allen–Cahn equation

Because of the original image usually has noises as in Figs. 3.1(a) and (b). Therefor, to prepare the numerical slice data ψ in Fig. 3.1(c), we need to applying the proposed method for volume reconstruction by using image segmentation technique [19, 21, 60, 61, 62, 63, 83].



(a) (b) (c)

FIGURE 3.1. Image segmentation process: (a) given original medical image, (b) mesh plot of the original image, and (c) results of after image segmentation.

The image segmentation is based on the Allen–Cahn equation which detailed descriptions in Chapter 2.1 and it enforces the diffuse interface to be the hyperbolic tangent profile. The geometric active contour model based on the mean curvature motion is given by the following evolution equation [60]:

$$\frac{\partial \psi(\mathbf{x},t)}{\partial t} = g(f_0(\mathbf{x}))\left(-\frac{F'(\psi(\mathbf{x},t))}{\epsilon^2} + \Delta\psi(\mathbf{x},t)\right) + \beta g(f_0(\mathbf{x}))F(\psi(\mathbf{x},t)), \tag{3.1}$$

where $\mathbf{x} = (x, y)$ and

$$f_0(\mathbf{x}) = \frac{f(\mathbf{x}) - f_{min}}{f_{max} - f_{min}},$$

here, f_{max} and f_{min} are the maximum and minimum values of the given slice image $f(\mathbf{x})$, respectively. Here, $\psi(\mathbf{x}, t)$ is a phase-field function which is close to 1 or -1 and $F(\psi) =$

$0.25(\psi^2-1)^2$. The function

$$g(f_0(\mathbf{x}))=\frac{1}{1+|\nabla(G_\sigma*f_0)(\mathbf{x})|^2}$$

is the edge-stopping function, which stops the evolution when the contour reaches the edge.

$$(G_\sigma*f_0)(\mathbf{x})=\int_\Omega G_\sigma(\mathbf{x}-\mathbf{y})f_0(\mathbf{y})d\mathbf{y}$$

is the convolution of the given image f_0 with the Gaussian function

$$G_\sigma(\mathbf{x})=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

ϵ is a constant that is related to the phase transition width, and β is a parameter. And we use $\sigma=1.5$ and $\beta=50000$.

We apply a hybrid method [60] to solve Eq. (3.1) and we outline the numerical solution algorithm for the sake of completeness. We discretize Eq. (3.1) in $\Omega=(a,b)\times(c,d)$. Let N_x and N_y be positive even integers, $h=(b-a)/N_x=(d-c)/N_y$ be the uniform mesh size. Let ψ_{ij}^n be approximations of $\psi(x_i,y_j,n\Delta t)$, where $x_i=a+(i-0.5)h$, $y_j=(j-0.5)h$, and Δt is the time step. The discrete edge function is defined by

$$g(f_0)_{ij}=\frac{1}{1+(G_\sigma*f_0)_{x,ij}^2+(G_\sigma*f_0)_{y,ij}^2},$$

where

$$(G_\sigma*f_0)_{x,ij}=\frac{(G_\sigma*f_0)_{i+1,j}-(G_\sigma*f_0)_{i-1,j}}{2h}$$

and

$$(G_\sigma*f_0)_{y,ij}=\frac{(G_\sigma*f_0)_{i,j+1}-(G_\sigma*f_0)_{i,j-1}}{2h}.$$

The discrete convolution is defined as

$$(G_\sigma*f_0)_{ij}=\sum_{p=i-1}^{i+1}\sum_{q=j-1}^{j+1}\frac{f_{0pq}}{2\pi\sigma^2}e^{-\frac{[(i-p)^2+(j-q)^2]h^2}{2\sigma^2}}.$$

The zero Neumann boundary condition is used. Then the following operator splitting numerical algorithm for Eq. (3.1) is as follows:

$$\frac{\psi_{ij}^* - \psi_{ij}^n}{\Delta t} = g_{ij}\Delta_d \psi_{ij}^* + \beta g_{ij} F(\psi_{ij}^n), \tag{3.2}$$

which is solved by a multigrid method [101]. Here Δ_d is the standard five point discrete Laplacian operator. Next, using the method of separation of variables we analytically solve the equation

$$\psi_t = g\frac{\psi - \psi^3}{\epsilon^2} \tag{3.3}$$

with the condition $\psi^n = \psi^*$. Then the solution of Eq. (3.3) at $t = (n+1)\Delta t$ is given as

$$\psi_{ij}^{n+1} = \frac{\psi_{ij}^*}{\sqrt{e^{-\frac{2g_{ij}\Delta t}{\epsilon^2}} + (\psi_{ij}^*)^2\left(1 - e^{-\frac{2g_{ij}\Delta t}{\epsilon^2}}\right)}}.$$

Figures 3.1 and 3.2 illustrate the process of image segmentation using Eq. (3.1). For the given medical image in Fig. 3.1(a), we define the scaled image $f_0(\mathbf{x})$ and initialize $\psi(\mathbf{x}, 0)$ as $\psi(\mathbf{x}, 0) = 1$ if $\mathbf{x}$ is inside the square contour, $\psi(\mathbf{x}, 0) = -1$ otherwise (see Fig. 3.2(a)). In Figs. 3.2(b) and (c), the initial contour evolves until it reaches the boundary of the image through the motion created by the mean curvature (the first term) and the second term on the right-hand side of Eq. (3.1). The term $\beta g(f_0(\mathbf{x}))F(\psi(\mathbf{x}, t))$ evolves the contour beyond the non-convex and disconnected regions.

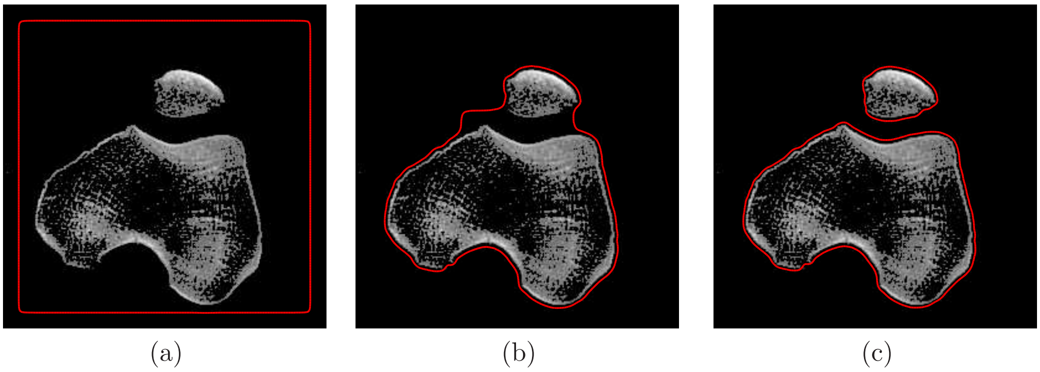


FIGURE 3.2. Image segmentation process of the given slice data: (a)–(c) evolutions of image segmentation, in which the curves are the zero contours of $\psi(x, y, t)$.

3.3. Using cloud points

Using cloud points, we can make the surface. First, we obtain a 3ds file format for a three-dimensional zebra model which is available in [99]. Next, we extract the geometry information such as vertices and triangles by using Autodesk ® 3DS MAX ®. Figure 3.3(a) shows the vertices of the zebra. However, some regions do not contain sufficient points for constructing a smooth surface. We add more points as shown in Fig. 3.3(b).

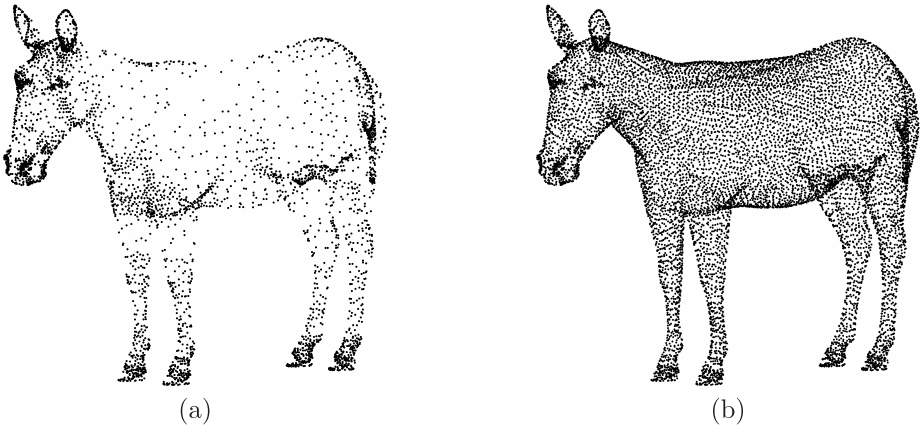


FIGURE 3.3. (a) Given unorganized data set of zebra and (b) enriched data set.

The procedure of adding points is as follows: Let *tol* be a tolerance which is a maximum size of the sides of the triangles. Loop over the triangles recursively until all the sides of the triangles are less than the given tolerance. For example, take a triangle consisting of three vertices $P_1,\ P_2,\ P_3$ and divide the triangle and add mid points until the maximum size of the sides is less than the given tolerance, see Figs. 3.4(a)–(d).

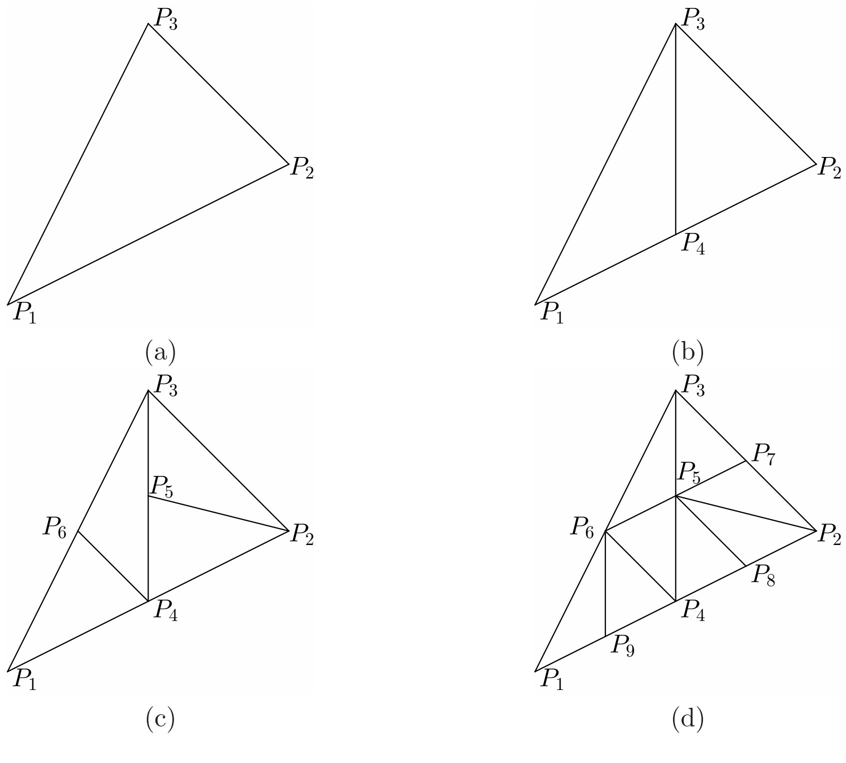


FIGURE 3.4. Recursively supplement points into polygon: (a) initial polygon, (b) first iteration, (c) second iteration, and (d) third iteration.

Now, we need to get the surface of zebra as the zero level set of a scalar function from the scattered data points. In this step, we apply an image segmentation technique which detailed description is in Chapter 3.2. For the segmentation, the governing equation is given as

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = g(\mathbf{x}) \left(-\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2} + \Delta\phi(\mathbf{x},t) + \beta F(\phi(\mathbf{x},t)) \right), \tag{3.4}$$

where $g(\mathbf{x})$ is the unsigned distance function constructed from the unorganized data set of zebra, ϕ is a phase-field function, which is close to 1 and -1 in the inside and outside domain of the reconstructed zebra.

Figure 3.5(a) represents the zero isosurface of the segmented image using Eq. (3.4) with $\epsilon = 0.0225$ and $\lambda = 500$. Here, a $64 \times 256 \times 192$ mesh grid on the domain $\Omega = (0,1) \times (0,4) \times (0,3)$ and $\Delta t = 0.001$ are used.

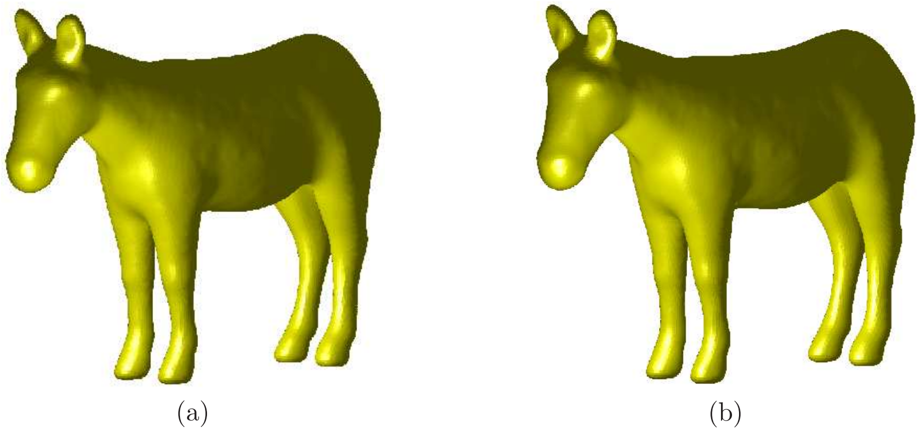

(a) (b)

FIGURE 3.5. Surface reconstruction process of the given unorganized data set of zebra: (a) after image segmentation step, and (b) after reinitialization shape on the zero isosurface.

Finally, we construct the signed distance function from the unsigned distance function which is obtained in the previous step, since $|\nabla\phi|$ is not one in the neighborhood of $\mathcal{S}$, i.e., $\phi = 0$. To obtain the signed distance function (Fig. 3.5 (b)), we apply the following equation:

$$\phi_t = S_\delta(\phi^0) \left(1 - \sqrt{\phi_x^2 + \phi_y^2 + \phi_z^2}\right), \tag{3.5}$$

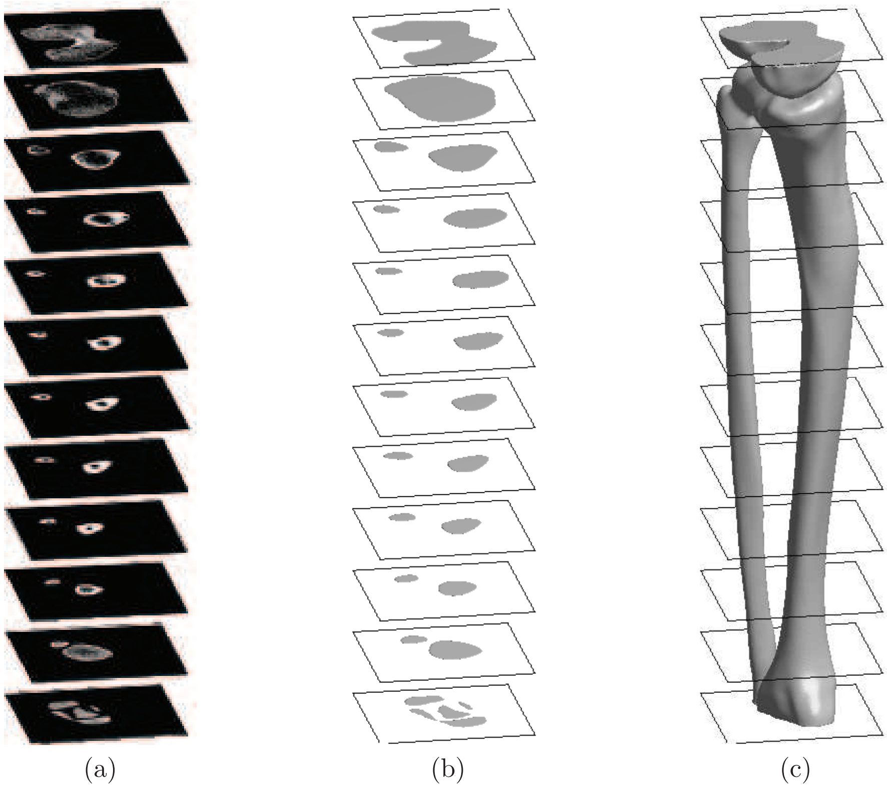
where $\phi_0 = \phi(\mathbf{x}, 0)$ and

$$S_\delta(\phi^0) = \frac{\phi^0}{\sqrt{(\phi^0)^2 + \delta^2}}$$

is a smoothed sign function [97].

3.4. Using slice data

We propose a numerical algorithm for 3D volume reconstruction from slice data. We start with an illustration of the process of the proposed algorithm when we have a set of cross-sectional slice data (Fig. 3.6(a)).



(a) (b) (c)

FIGURE 3.6. Volume reconstruction from slice data: (a) given slice data, (b) filled contour plots of segmented slice data, and (c) isosurface of the reconstructed volume.

First, by using a modified Allen–Cahn equation, the image segmentation algorithm is applied for the given slice data f to obtain the phase-field function ψ (segmented image) that detailed description in Chapter 3.2. Figure 3.6(b) shows the filled contour plots of segmented slice data ψ. For the second, by using a modified Cahn–Hilliard equation and the segmented slice data, we reconstruct the volume (Fig. 3.6(c)).

In order to reconstruct a 3D volume from a set of segmented slice data, we consider the modified Cahn–Hilliard equation which contains a fidelity term:

$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = \Delta\mu(\mathbf{x},t) + \lambda(\mathbf{x})(\psi(\mathbf{x}) - \phi(\mathbf{x},t)),\ \mathbf{x} \in \Omega,\ 0 < t \leq T, \tag{3.6}$$

$$\mu(\mathbf{x},t) = F'(\phi(\mathbf{x},t)) - \epsilon^2 \Delta\phi(\mathbf{x},t), \tag{3.7}$$

where $\mathbf{x} = (x, y, z)$, $\Omega = (0, L_x) \times (0, L_y) \times (0, L_z)$ is a domain, and

$$\lambda(\mathbf{x}) = \begin{cases} \lambda_0, & \text{if } \mathbf{x} \text{ is in the given slice data,} \\ 0, & \text{otherwise.} \end{cases}$$

Here, $\phi(\mathbf{x}, t)$ is a phase-field function which is close to 1 or -1 for the reconstructed volume's respective interior and exterior. The surface of the volume is represented by the zero-level set of ϕ. Let $\psi(x, y, z_i)$ for $i = 1, \ldots, N_s$ be the segmented slice data obtained by performing the image segmentation algorithm on $S_i := \Omega \cap \{z = z_i\}$, where $z_1 = 0$ and $z_{N_s} = L_z$. Here, N_s is the number of slice data. Figure 3.7 shows the slice data's schematics.

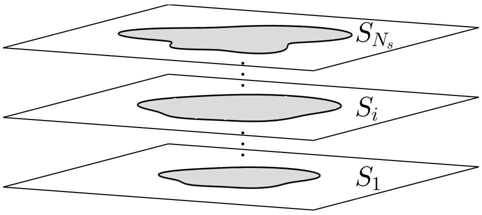


FIGURE 3.7. Schematic of slice data S_1 to S_{N_s}.

To define the initial condition $\phi(\mathbf{x}, 0)$, we use a linear interpolation between two consecutive slices: We have

$$\phi(x, y, \theta z_{i+1} + (1-\theta)z_i) = \theta\psi(x, y, z_{i+1}) + (1-\theta)\psi(x, y, z_i), \tag{3.8}$$

for $0 \leq \theta \leq 1$ and $i = 1, \ldots, N_s - 1$. Homogeneous Neumann boundary conditions for both ϕ and μ are applied: $\mathbf{n} \cdot \nabla\phi = \mathbf{n} \cdot \nabla\mu = 0$ on $\partial\Omega$ except $z = z_1$ and $z = z_{N_s}$, where Dirichlet boundary condition for ϕ is applied.

Equations (3.6) and (3.7) have also been used in image inpainting problem, which is the process of filling in missing parts of damaged images based on information from the surrounding areas [8]. The basic mechanism of the inpainting model is illustrated in Fig. 3.8.

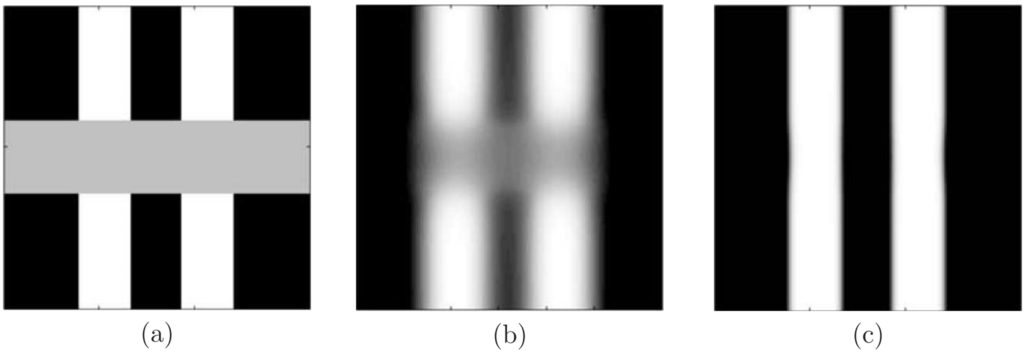


FIGURE 3.8. Process of the image inpainting: (a) Initial data, (b) intermediate state, and (c) steady state. Image domain is 128 × 128 and gap distance of the inpainting region is 30 grid points. Reprinted from A.L. Bertozzi, S. Esedoglu, and A. Gillette [9] with permission from Society for Industrial and Applied Mathematics.

The inpainting region is denoted as gray in Fig. 3.8(a). Let $\psi(\mathbf{x})$ be the image data from Fig. 3.8(a). In the figure, black, gray, and white colors represent $\psi(\mathbf{x}) \approx -1$, 0, and 1, respectively. Take the initial condition as $\phi(\mathbf{x}, 0) = \psi(\mathbf{x})$ and solve Eqs. (3.6) and

(3.7). Then outside the inpainting region, due to the fidelity term $\lambda(\mathbf{x})(\psi(\mathbf{x}) - \phi(\mathbf{x}, t))$, the temporal evolution of $\phi(\mathbf{x}, t)$ does not deviate much from the original image data $\psi(\mathbf{x})$. On the other hand, inside the inpainting region where $\lambda(\mathbf{x}) = 0$, the Cahn–Hilliard dynamics takes place. The gray value will evolve to either -1 or 1 depending on the boundary values of the inpainting region. Figures 3.8(b) and (c) show the intermediate and steady states, respectively. More details can be found in [8].

We use a nearly identical governing equation to those used for image inpainting, which is two-dimensional. However, we use the equation in a different context, meaning we reconstruct a 3D volume from a set of slice data. Our approach involves the salient application of the partial differential equation used in image inpainting.

3.4.1. Discretization. For the stable temporal discretization, we use the efficient numerical scheme based on the unconditionally gradient stable scheme [59, 109], which allows large time steps. We discretize the governing equations, Eqs. (3.6) and (3.7), in our 3D space, $\Omega = (0, L_x) \times (0, L_y) \times (0, L_z)$. Let $x_i = (i - 0.5)h,\ y_j = (j - 0.5)h,\ z_k = (k - 0.5)h,$ $1 \le i \le N_x$, $1 \le j \le N_y$, and $1 \le k \le N_z$, where N_x, N_y, and N_z are positive even integers and $h = L_x/N_x = L_y/N_y = L_z/N_z$ is the uniform mesh size. Let $\mathbf{x}_{ijk} = (x_i, y_j, z_k)$ and let ϕ_{ijk}^n be an approximation of $\phi(\mathbf{x}_{ijk}, n\Delta t)$, where $\Delta t = T/N_t$ is the time-step, T is the final time, and N_t is the total number of time-steps. Then the discrete domain is defined by $\Omega_h = \{\mathbf{x}_{ijk} \mid 1 \le i \le N_x,\ 1 \le j \le N_y,\ 1 \le k \le N_z\}$. In addition, we denote the grid function as $\phi_h = \{\phi_{ijk} \mid \mathbf{x}_{ijk} \in \Omega_h\}$ and discrete l_2-norm of ϕ_h as

$$\|\phi_h\|_2 = \sqrt{\sum_{\mathbf{x}_{ijk} \in \Omega_h} h^3 \left(\phi_{ijk}^n\right)^2}. \tag{3.9}$$

We then have the following discretization.

$$\begin{aligned}
\frac{\phi_{ijk}^{n+1}-\phi_{ijk}^{n}}{\Delta t} &= \lambda_{ijk}(\psi_{ijk}-\phi_{ijk}^{n}) + \frac{\mu_{i+1,jk}^{n+1}+\mu_{i-1,jk}^{n+1}}{h^2} \\
&\quad + \frac{\mu_{i,j+1,k}^{n+1}+\mu_{i,j-1,k}^{n+1}+\mu_{ij,k+1}^{n+1}+\mu_{ij,k-1}^{n+1}-6\mu_{ijk}^{n+1}}{h^2}, \qquad (3.10)\\
\mu_{ijk}^{n+1} &= (\phi_{ijk}^{n+1})^3-\phi_{ijk}^{n}-\epsilon^2\frac{\phi_{i+1,jk}^{n+1}+\phi_{i-1,jk}^{n+1}}{h^2} \\
&\quad -\epsilon^2\frac{\phi_{i,j+1,k}^{n+1}+\phi_{i,j-1,k}^{n+1}+\phi_{ij,k+1}^{n+1}+\phi_{ij,k-1}^{n+1}-6\phi_{ijk}^{n+1}}{h^2}. \qquad (3.11)
\end{aligned}$$

The resulting discrete equations, Eqs. (3.10) and (3.11), are solved using a multigrid method. In order to condense the discussion, we describe only the relaxation step of this method for more detailed description in Appendix. Let $\phi_{ijk}^{n+1,m}$ and $\phi_{ijk}^{n+1,m+1}$ be the respective current and updated approximations of ϕ_{ijk}^{n+1}. We set the initial guess to be the previous time-step solution as $\phi_{ijk}^{n+1,0}=\phi_{ijk}^{n}$. We linearize $(\phi_{ijk}^{n+1})^3$ as

$$(\phi_{ijk}^{n+1})^3 \approx (\phi_{ijk}^{n+1,m})^3+3(\phi_{ijk}^{n+1,m})^2(\phi_{ijk}^{n+1}-\phi_{ijk}^{n+1,m}).$$

Then we apply the Gauss–Seidel relaxation to the multigrid method:

$$\begin{aligned}
&\frac{\phi_{ijk}^{n+1,m+1}}{\Delta t}+\frac{6\mu_{ijk}^{n+1,m+1}}{h^2} = \frac{\phi_{ijk}^{n}}{\Delta t}+\lambda_{ijk}(\psi_{ijk}-\phi_{ijk}^{n}) \\
&\quad +\frac{\mu_{i+1,jk}^{n+1,m}+\mu_{i-1,jk}^{n+1,m+1}+\mu_{i,j+1,k}^{n+1,m}+\mu_{i,j-1,k}^{n+1,m+1}+\mu_{ij,k+1}^{n+1,m}+\mu_{ij,k-1}^{n+1,m+1}}{h^2}, \\
&-\left(\frac{6\epsilon^2}{h^2}+3(\phi_{ijk}^{n+1,m})^2\right)\phi_{ijk}^{n+1,m+1}+\mu_{ijk}^{n+1,m+1} = -2(\phi_{ijk}^{n+1,m})^3-\phi_{ijk}^{n} \\
&\quad -\epsilon^2\frac{\phi_{i+1,jk}^{n+1,m}+\phi_{i-1,jk}^{n+1,m+1}+\phi_{i,j+1,k}^{n+1,m}+\phi_{i,j-1,k}^{n+1,m+1}+\phi_{ij,k+1}^{n+1,m}+\phi_{ij,k-1}^{n+1,m+1}}{h^2}.
\end{aligned}$$

For more details about the multigrid algorithm of a 3D Cahn–Hilliard equation, please refer to [109]. It should be noted that without the fitting term (Eq. (3.6) with $\lambda_0=0$), the proposed scheme is an unconditionally gradient stable scheme [30].

3.4.2. Numerical experiments. In this section, we perform several numerical experiments in order to demonstrate the performance of our proposed scheme. We regard the numerical result as the steady state solution if the relative error $\|\phi^{n+1} - \phi^n\|_2^2/\|\phi^n\|_2^2$ is less than a tolerance tol. Unless otherwise specified, we use $\epsilon = \epsilon_4$, time step $\Delta t = 0.5h$, $\lambda_0 = 1000$, and $tol = 0.002$. Throughout the paper, we use isotropic grids and we apply the image segmentation step for the slice data unless we use the analytic hyperbolic tangent function.

3.4.2.1. *Basic mechanism of the algorithm.* We start with an example in the two-dimensional (2D) space $\Omega = (0,1) \times (0,0.5)$ with a 64×32 mesh grid to show the algorithm's basic mechanism in Eqs. (3.6) and (3.7).

$$\begin{aligned}
\psi(x,y_0) &= -1 + \tanh\left[\frac{x-0.15}{\sqrt{2}\epsilon_4}\right] - \tanh\left[\frac{x-0.4}{\sqrt{2}\epsilon_4}\right] + \tanh\left[\frac{x-0.6}{\sqrt{2}\epsilon_4}\right] - \tanh\left[\frac{x-0.9}{\sqrt{2}\epsilon_4}\right], \\
\psi(x,y_{11}) &= -1 + \tanh\left[\frac{x-0.3}{\sqrt{2}\epsilon_4}\right] - \tanh\left[\frac{x-0.45}{\sqrt{2}\epsilon_4}\right] + \tanh\left[\frac{x-0.55}{\sqrt{2}\epsilon_4}\right] - \tanh\left[\frac{x-0.7}{\sqrt{2}\epsilon_4}\right], \\
\psi(x,y_{21}) &= -1 + \tanh\left[\frac{x-0.4}{\sqrt{2}\epsilon_4}\right] - \tanh\left[\frac{x-0.65}{\sqrt{2}\epsilon_4}\right], \\
\psi(x,y_{33}) &= -1 + \tanh\left[\frac{x-0.25}{\sqrt{2}\epsilon_4}\right] - \tanh\left[\frac{x-0.65}{\sqrt{2}\epsilon_4}\right].
\end{aligned}$$

In Fig. 3.9(a), thick line segments are the given slices and has following hyperbolic tangent profiles, which are shown in Fig. 3.9(b). Because this is an example in 2D, we assume that the given slice data has the hyperbolic tangent profile. However, in the 3D case, we need an image segmentation algorithm as a preprocess before the volume reconstruction. We guess the initial state using the liner interpolation (see Fig. 3.9(c)), and the filled contour at level zero is shown in Fig. 3.9(d). Figure 3.9(e) displays the reconstructed image obtained by our proposed method.

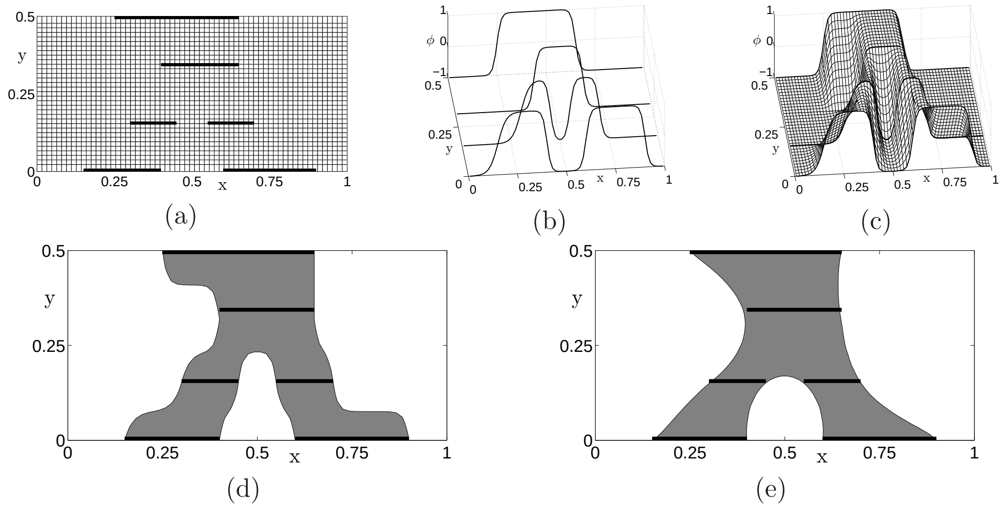
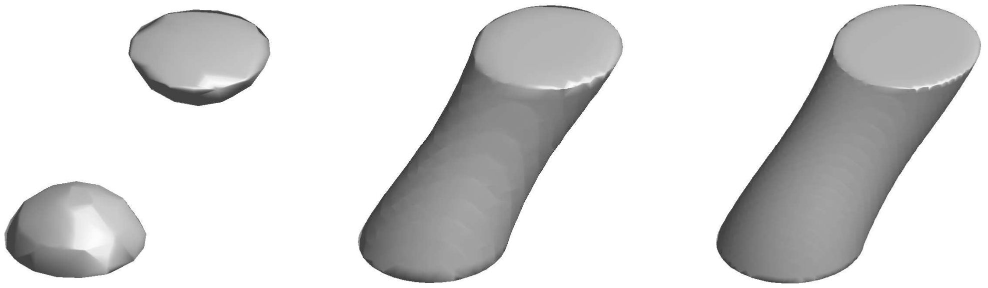


FIGURE 3.9. (a) mesh grid with initial data, (b) processed slice data, (c) initial guess obtained using a linear interpolation, (d) zero-filled contour of ϕ at $t = 0$, and (e) zero-filled contour at a steady state after 36 iterations.

3.4.2.2. *Effect of image pixel and interslice dimension.* We perform simulations to show the effect of pixel dimension on 3D volume reconstruction with increasing pixel number, $16 \times 16 \times 8$, $32 \times 32 \times 16$, and $64 \times 64 \times 32$. We set numerical parameters as $\epsilon = \epsilon_4$, $\Delta t = 0.1h$, and $tol = 0.0001$ on a domain $\Omega = (0, 1) \times (0, 1) \times (0, 0.5)$. The given slice data is a set of two circles whose radii are same as $r = 0.2$ and centers are apart by $d = 0.4$. Figure 3.10 shows the steady solutions with different pixel numbers. We can see that, with a few grid points, the reconstructed surface could not be connected, however, with increasing pixel number, it is getting smooth and connected.

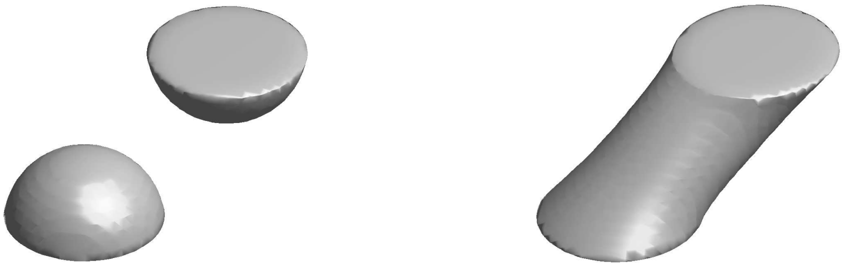
We consider the effect of interslice dimension between two consecutive slices. A $64 \times 64 \times 16$ mesh grid is used and numerical parameters are set as $\epsilon = \epsilon_4$, $\Delta t = 0.5h$, and $tol = 0.0001$ on a domain $\Omega = (0, 1) \times (0, 1) \times (0, 0.25)$. The given slice data is a set of two circles which radii are same as $r = 0.15$ and centers are apart by $d = 0.32$, see Fig.



(a) $16 \times 16 \times 8$ (b) $32 \times 32 \times 16$ (c) $64 \times 64 \times 32$

FIGURE 3.10. Steady solutions for different pixel numbers. Here, only the bottom and top slices are used.

3.11(a). The reconstructed surface is not connected because the circles are not close enough comparing with the given interslice. However, if we add a slice at $z = z_8$, which has same radius and is located in middle in two circles, then we have smoothly connected surface as shown in Fig. 3.11(b).

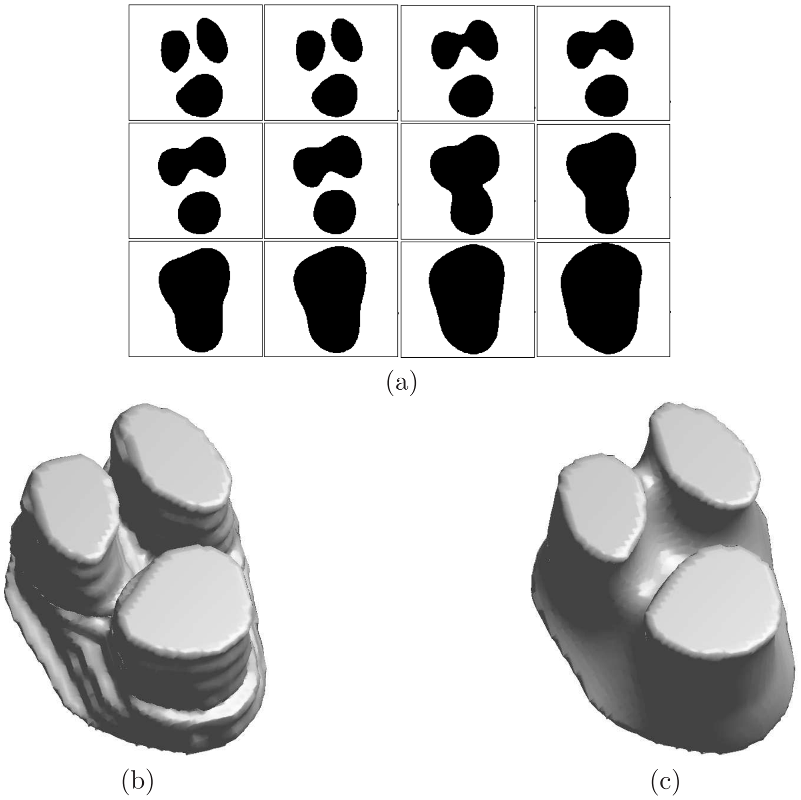


(a) (b)

FIGURE 3.11. Steady solution for different interface dimension.

3.4.2.3. *Volume reconstructions using synthetic slice data.* We reconstruct the 3D volume using the three set of more complicated synthetic slice data, consisting a spiral, two linked tori, and three branching. In this section, we use $tol = 0.001$. We consider twelve slice data are used (Fig. 3.12(a)) and there are two slices between any two consecutive slice data sets. This simulation is performed on the domain $(0, 1) \times (0, 1.125) \times (0, 0.531)$

with a 64 × 72 × 34 mesh grid. Figures 3.12(b) and 3.12(c) are the initial shape by linear interpolation and final result obtained after 44 iterations.


(a)

(b)

(c)

FIGURE 3.12. (a) Synthetic slice data of three branching (ordered left to right and top to bottom), (b) initial shape, and (c) reconstructed volume.

3.4.2.4. *Volume reconstructions using real slice data.* In this section, we simulate several volume reconstructions by using real slice data. We consider a human vertebra with a twenty-six slice data from [28] (Fig. 3.13(a)). There are two slices between any two given successive slice data sets, except for the first and last sets, between which we use only one slice. This simulation is performed on the domain $(0, 1) \times (0, 1) \times (0, 0.45)$ with a $160 \times 160 \times 72$ mesh grid. Figures 3.13(b) and (c) represent the initial state by the linear interpolation and final result obtained after 14 iterations. As can be seen, our proposed method produces visually clear results, even though the contours are not convex and multiple links between contours are exist.

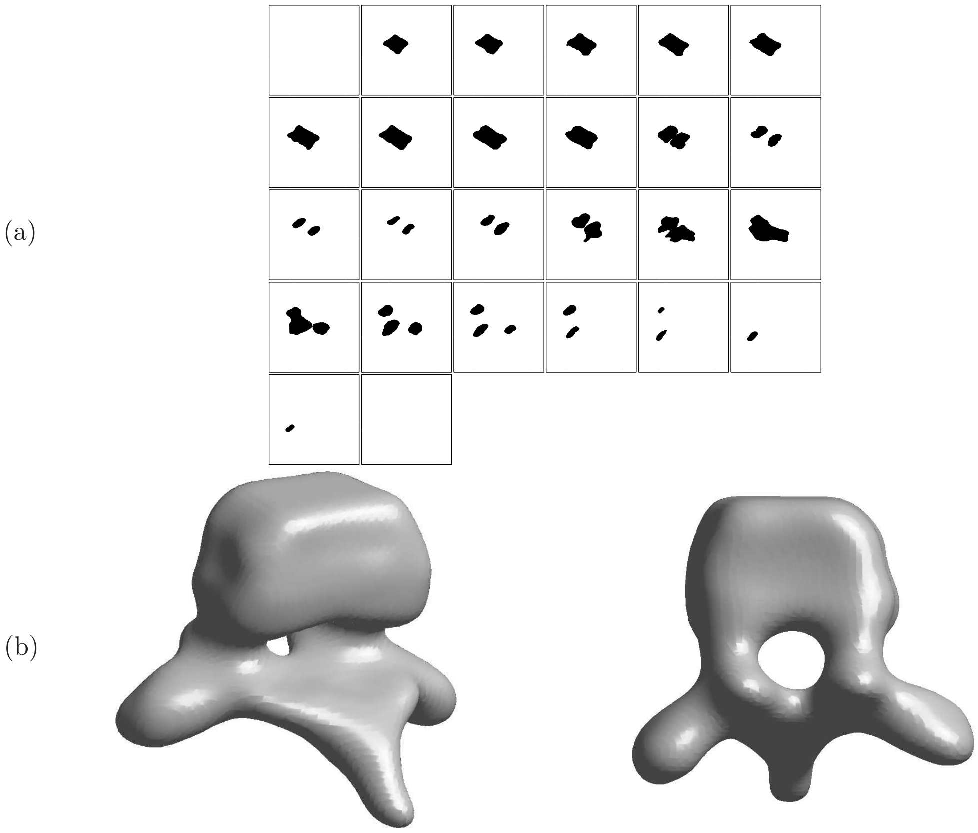
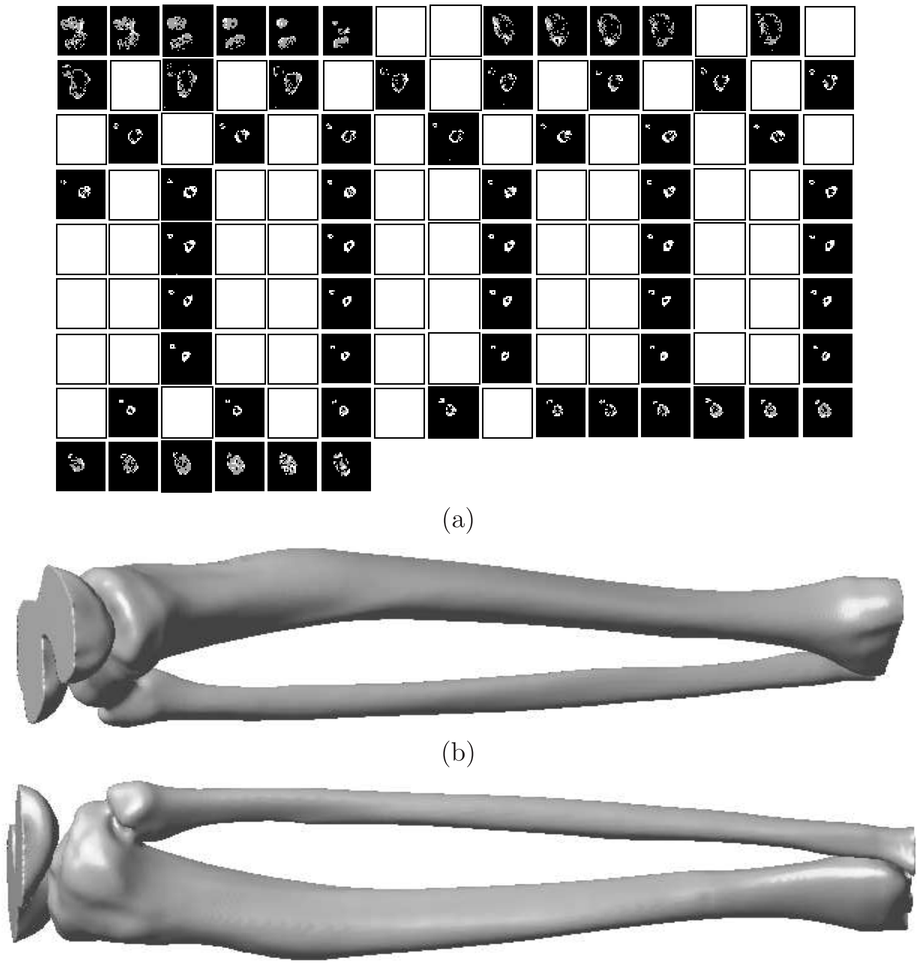


FIGURE 3.13. Volume reconstruction of human vertebra: (a) slice data (ordered left to right and top to bottom) and (b) reconstructed volume of the human vertebra from different angles.

Our last test examines the volume reconstruction from medical images of a human bone (tibia and fibula). We have a set of bone slice images with an image size of 216 × 216 (Fig 3.14(a)). The number of slices is 128 and there are seven slices between any two consecutive slices. Note that, because our method can reconstruct the volume with fewer slice data, we remove some similar slices (the empty boxes) and use 63 slices. We perform the resolution on the domain $(0,1) \times (0,1) \times (0,4.120)$ with a $216 \times 216 \times 890$ mesh grid. Figures 3.14(b) and (c) show results of volume reconstruction with different angles after only 23 iterations. As can be seen, our algorithm represents the bone image well and produces good quality.



(a)

(b)

(c)

FIGURE 3.14. Volume reconstruction from medical images of a human bone (tibia and fibula): (a) slice data (ordered left to right and top to bottom), in which empty boxes represent skipped data, and (b) and (c) reconstructed volumes obtained from two different angels.

3.5. Summary

We presented a fast, robust, and accurate numerical method for creating a mathematical model that produces 3D volume reconstruction using cloud points and slice data. The governing equations are based on the Allen–Cahn and Cahn–Hilliard equations with a fidelity term. The proposed algorithm has two steps: image segmentation for the raw given slice data or cloud points and 3D volume reconstruction using the segmented images. We applied a hybrid method and an unconditionally stable nonlinear splitting scheme, and then we solved the resulting system of discrete equations using a multigrid method. We demonstrated the accuracy, efficiency, and robustness of the method on both synthetic and real medical images, such as a zebra, three branching, vertebra, tibia, and fibula.

Chapter 4

Construction of computational domain and boundary condition

4.1. Narrow band domain

For simplicity of exposition, we shall illustrate schematics in two-dimensional space even though the actual algorithm is three-dimensional. Figure 4.1 shows the schematic illustration of surface S, the narrow band domain Ω_δ with thickness 2δ, and its boundary $\partial\Omega_\delta$.

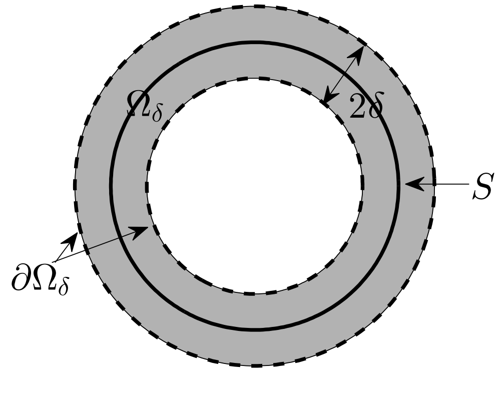


FIGURE 4.1. Schematic illustration of surface S, the narrow band domain Ω_δ with thickness 2δ, and its boundary $\partial\Omega_\delta$.

Let $\psi : \mathbb{R}^3 \rightarrow \mathbb{R}$ be the signed distance function to S. Its zero level set is given by

$$S = \{\mathbf{x} \in \mathbb{R}^3 \big| \psi(\mathbf{x}) = 0\}$$

with $\psi < 0$ inside S and $\psi > 0$ outside S.

Let us define the discrete narrow band domain as

$$\Omega_\delta^h = \{(x_i, y_j, z_k) \big| |\psi(x_i, y_j, z_k)| < \delta\}$$

(see Fig. 4.2). In other word,

$$\Omega_\delta = \{\mathbf{y}|\mathbf{x} \in S, \mathbf{y} = \mathbf{x} + \theta\mathbf{n}(\mathbf{x}) \text{ for } |\theta| < \delta\}$$

where $\mathbf{n}$ is unit normal vector on the surface S and δ is a positive constant.

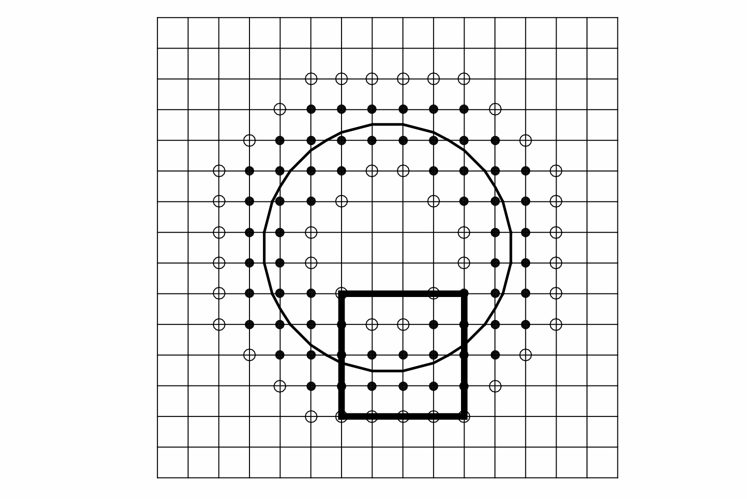


FIGURE 4.2. Schematic illustration of the discrete narrow band domain Ω_δ^h (indicated by $\bullet$) and its ghost points $\partial\Omega_\delta^h$ (indicated by $\circ$). Here, the curve S represents as a solid curve.

Note that we should choose $\delta \geq \sqrt{3}h$ since the computational narrow band domain must contain the interpolation stencil for the closest points of the ghost points.

We also define an indicator function I as

$$I_{ijk} = \begin{cases} 0 & \text{if } (x_i, y_j, z_k) \in \Omega_\delta^h, \\ 1 & \text{otherwise.} \end{cases} \tag{4.1}$$

Then, we define the ghost points as

$$\partial\Omega_\delta^h = \{(x_i, y_j, z_k) \big| I_{ijk}|\nabla_h I_{ijk}| \neq 0\}$$

(see Fig. 4.3), where

$$\nabla_h I_{ijk} = \frac{I_{i+1,jk} - I_{i-1,jk}, I_{i,j+1,k} - I_{i,j-1,k}, I_{ij,k+1} - I_{ij,k-1}}{2h}.$$

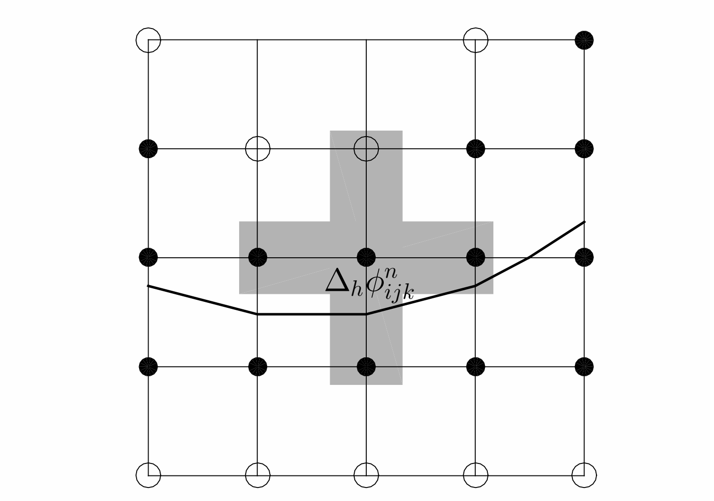


FIGURE 4.3. Expanded illustration of the boxed section in Fig. 4.2. The shaded region indicates the stencil for the discrete Laplacian operator.

4.2. Closest points method

A boundary condition is needed to calculate Laplacian operator in a given band domain. That boundary condition can be defined by using the closest point method.

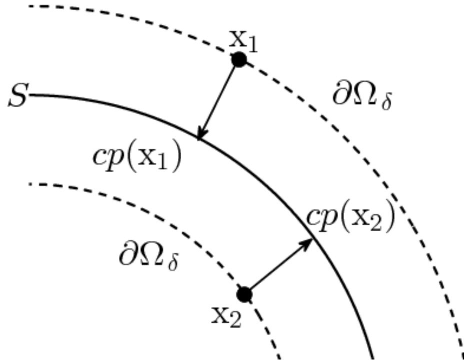


FIGURE 4.4. Closest points cp($\mathbf{x}_1$) and cp($\mathbf{x}_2$) for points $\mathbf{x}_1$ and $\mathbf{x}_2$.

Given a surface S, let $\text{cp}(\mathbf{x})$ be a (possibly non-unique) point belonging to S which is closest to $\mathbf{x}$ [69]. Figure 4.4 shows the closest points $\text{cp}(\mathbf{x}_1)$ and $\text{cp}(\mathbf{x}_2)$ for points $\mathbf{x}_1$ and $\mathbf{x}_2$. The boundary condition is

$$\phi(\mathbf{x}, t) = \phi(\text{cp}(\mathbf{x}), t) \text{ on } \partial\Omega_\delta. \tag{4.2}$$

If δ is small enough, then this boundary condition allows us to use the standard Laplacian operator in the narrow band domain because the condition results in ϕ which is constant in the normal direction to the surface.

The numerical closest point of $\mathbf{x}_{ijk}$ to the surface S is given as

$$\text{cp}(\mathbf{x}_{ijk}) = \mathbf{x}_{ijk} - \frac{\nabla_h|\psi_{ijk}|}{|\nabla_h|\psi_{ijk}||}|\psi_{ijk}|. \tag{4.3}$$

Since $\text{cp}(\mathbf{x}_{ijk})$ is generally not on a grid point, $\phi^n(\text{cp}(\mathbf{x}_{ijk}))$ is obtained using the trilinearly interpolated value (see Fig. 4.5).

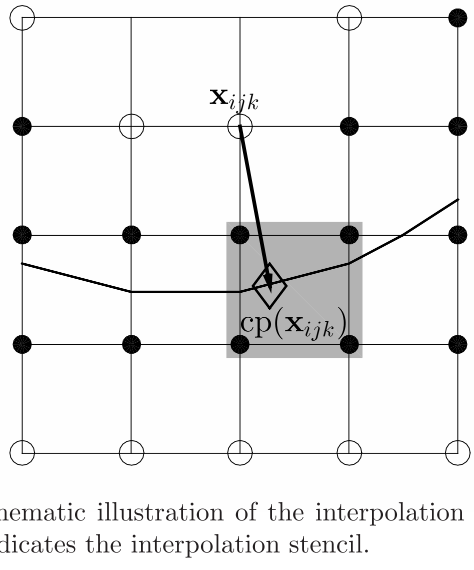


FIGURE 4.5. Schematic illustration of the interpolation of $\text{cp}(\mathbf{x}_{ijk})$. The shaded region indicates the interpolation stencil.

To compute the interpolation efficiently, we tabulated the interpolation stencil (we only save the smallest index in lexicographical order) and three fractions for each ghost point before starting time iterations.

4.3. Summary

The overall process is summarized as follows. Figure 4.6-4.9 show the surface taken from the numerical simulation results in Chapter 5.2.

Figure 4.6 shows the surface of the bunny and we define the surface S.

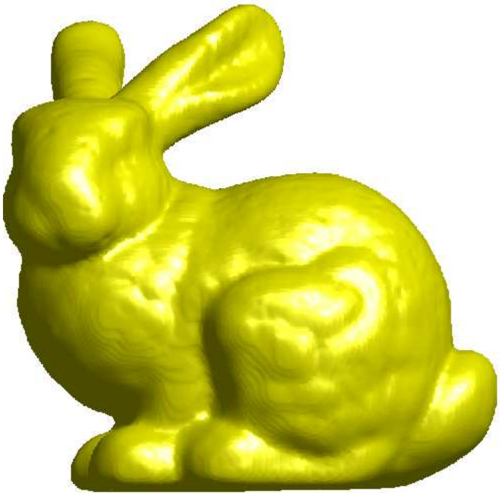


FIGURE 4.6. Bunny surface.

The boundary $\partial\Omega_\delta$, band width 2δ, and band domain Ω_δ which is denoted by the shaded region in Fig. 4.7.

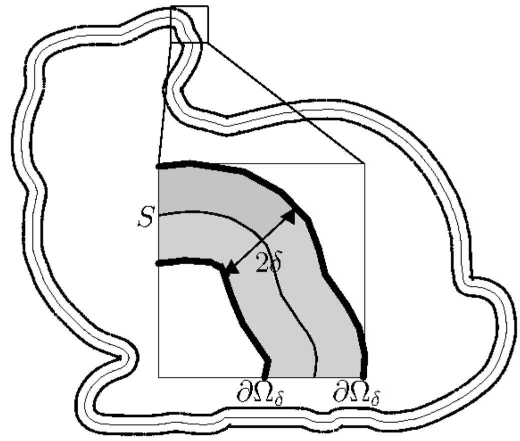


FIGURE 4.7. Slice data of the band domain.

When we calculate $\Delta_h \phi^n_{ijk}$, the boundary values at the empty circles as shown in Fig. 4.8 are needed.

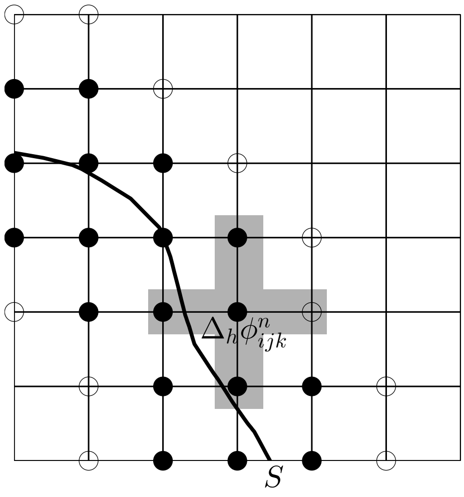


FIGURE 4.8. The part of the bunny's ear in Fig. 4.7, definition of the Laplacian using the points on the shaded region.

For each point $\mathbf{x} \in \partial\Omega_\delta$, using the trilinear interpolation we define the closest point function $\text{cp} : \partial\Omega_\delta \to S$ which assigns the value of closest point $\text{cp}(\mathbf{x}, t)$ as shown in Fig. 4.9.

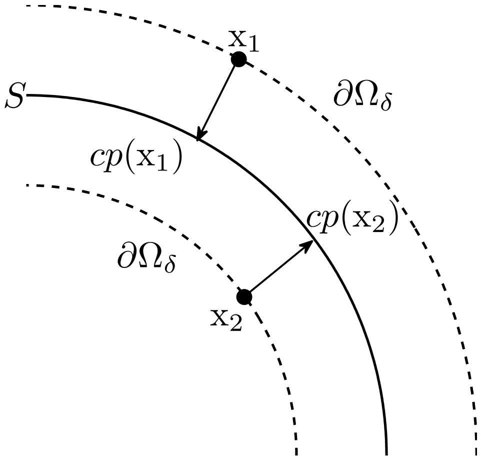


FIGURE 4.9. Closest points $cp(\mathbf{x}_1)$ and $cp(\mathbf{x}_2)$ for boundary points $\mathbf{x}_1$ and $\mathbf{x}_2$.

Therefore, we define the boundary condition as

$$\phi(\mathbf{x},t) = \phi(\mathrm{cp}(\mathbf{x}),t) \text{ on } \partial\Omega_\delta. \tag{4.4}$$

We solve the various partial differential equations on the band domain and boundary conditions by using closest point method in the next Chapter 5.

Chapter 5

Partial differential equations on non-flat surfaces

5.1. Allen–Cahn equation

5.1.1. Introduction. The previous studies that have been carried out are as follows. In the nature and applied sciences need solving the PDEs on surfaces. For instance, fluid flow and solidification of a thin film [77], brain imaging [74], diblock copolymers [98], computer graphics for texture synthesis [103], and surfactant distribution on a moving interface [107]. Analytic solutions to the problems on general surfaces are not always feasible, therefore it is important to be able to numerically approximate them accurately and efficiently.

In [88], the authors presented the closest point method, which is an embedding method for solving PDEs on surfaces. In [75], the diffusion generated motion algorithm has been extended to the moving curves on surfaces. Key aspects to their approach are the use of standard diffusion in one dimension higher than the underlying surface and the use of the closest point representation of the surface. The authors of [12] introduced a framework for solving variational problems and partial differential equations on surfaces. The main idea is to represent the surface as the level set of a higher dimensional function and to solve the surface equations in a fixed Cartesian coordinate system.

In this chapter, we propose a fast and accurate numerical method on a narrow band domain for motion by mean curvature on a surface in three-dimensional space using the

Allen–Cahn equation. The proposed hybrid explicit numerical method is based on an operator splitting method. First, we solve the heat equation by using an explicit finite difference scheme. For the domain boundary cells, we use an interpolation using the closest point method. Then, we update the solution by using a closed-form solution. The proposing numerical algorithm is simple and computationally efficient since we use a hybrid explicit numerical scheme and solve the governing equation only on the narrow domain. The implementation of the algorithm is straightforward. We perform a series of numerical experiments. The computational results are consistent with known analytic solutions.

5.1.2. Governing equations. In this section, we consider motion by mean curvature of curves on a surface in three-dimensional space. Let S be a smooth surface in $\mathbb{R}^3$ and then we define a δ-neighborhood band of S as $\Omega_\delta = \{\mathbf{y} | \mathbf{x} \in S, \mathbf{y} = \mathbf{x} + \theta \mathbf{n}(\mathbf{x}) \text{ for } |\theta| < \delta\}$, where $\mathbf{n}(\mathbf{x})$ is a unit normal vector at $\mathbf{x} \in S$. We use the following Allen–Cahn (AC) equation:

$$\frac{\partial \phi(\mathbf{x}, t)}{\partial t} = -\frac{F'(\phi(\mathbf{x}, t))}{\epsilon^2} + \Delta \phi(\mathbf{x}, t), \ \mathbf{x} \in \Omega_\delta, \ 0 < t \leq T, \tag{5.1}$$

where $\phi(\mathbf{x}, t)$ is the difference between the concentrations of the two mixtures' components and $F(\phi) = 0.25(\phi^2 - 1)^2$. The parameter ϵ is the gradient energy coefficient related to the interfacial energy. The AC equation (5.1) was originally introduced as a mathematical model for antiphase domain coarsening in a binary alloy [2].

5.1.3. Discretization. We present a numerical scheme for the AC equation on the narrow band domain, Ω_δ. We discretize the AC equation in a three-dimensional domain $\Omega = (a, b) \times (c, d) \times (e, f)$ which includes Ω_δ. Let N_x, N_y, and N_z be positive integers, $h = (b - a)/N_x = (d - c)/N_y = (f - e)/N_z$ be the uniform mesh size, and $\Omega^h = \{(x_i = a + hi, y_j = c + hj, z_k = e + hk) | 0 \leq i \leq N_x, 0 \leq j \leq N_y, 0 \leq k \leq N_z\}$ be the discrete domain. Let ϕ^n_{ijk} be approximations of $\phi(x_i, y_j, z_k, n\Delta t)$, where $\Delta t = T/N_t$ is the time step, T is the final time, and N_t is the total number of time steps. We use the standard seven point discrete Laplacian:

$$\Delta_h \phi_{ijk} = \frac{\phi_{i+1,jk} + \phi_{i-1,jk} + \phi_{i,j+1,k} + \phi_{i,j-1,k} + \phi_{ij,k+1} + \phi_{ij,k-1} - 6\phi_{ijk}}{h^2} \tag{5.2}$$

We solve the AC equation (5.1) by using splitting method. First we solve the diffusion equation in the discrete narrow band domain Ω^h_δ using the explicit Euler's method:

$$\frac{\phi^*_{ijk} - \phi^n_{ijk}}{\Delta t} = \Delta_h \phi^n_{ijk} \tag{5.3}$$

with the boundary condition on $\partial\Omega^h_\delta$;

$$\phi^n_{ijk} = \phi^n(\mathrm{cp}(\mathbf{x}_{ijk})). \tag{5.4}$$

The definition of closest point is in Chapter 4. We have a stability condition, $\Delta t \le h^2/6$, which is not that severe constraint if we are concerned with the numerical accuracy.

Next, with ϕ^*_{ijk} as the solution at $t = n\Delta t$, we solve the following equation up to $t = (n+1)\Delta t$:

$$\phi_t = \frac{\phi - \phi^3}{\epsilon^2}. \tag{5.5}$$

We have the following closed-form solution in the discrete narrow band domain Ω^h_δ for Eq. (5.5):

$$\phi^{n+1}_{ijk} = \frac{\phi^*_{ijk}}{\sqrt{e^{-\frac{2\Delta t}{\epsilon^2}} + (\phi^*_{ijk})^2 \left(1 - e^{-2\frac{\Delta t}{\epsilon^2}}\right)}}. \tag{5.6}$$

This scheme is explicit and we only solve the AC equation (5.1) on a narrow band domain, therefore it is very fast.

5.1.4. Numerical experiments. We perform a series of numerical experiments to demonstrate the accuracy and efficiency of the proposed numerical algorithm. We define the width of the transition layer by using the ϵ value. From an equilibrium profile

$$\phi(x) = \tanh\left(\frac{x}{\sqrt{2}\epsilon}\right),$$

the concentration field ϕ varies from -0.9 to 0.9 over a distance of about $2\sqrt{2}\epsilon\tanh^{-1}(0.9)$. Therefore, if we want this value to be about mh [26], then we should take ϵ as

$$\epsilon_m = \frac{mh}{2\sqrt{2}\tanh^{-1}(0.9)} \approx 0.24mh.$$

Unless otherwise stated, we use $\epsilon = \epsilon_4$.

5.1.4.1. *Algorithm and code.* In this section, we explain the algorithm and represent the MATLAB code. Algorithm is follows:

Algorithm 1 Determine the values of boundary and solve AC equation

```
for all domain do
    calculate the difference, $\partial\Omega^h_\delta = \{(x_i, y_j, z_k)|I_{ijk}|\nabla_h I_{ijk}|\}$
    if value of $\partial\Omega^h_\delta$ is positive then
        new numbering the indices which are boundary points's indices
    end if
end for
for all boundary do
    find the indices and points ($\text{cp}(\mathbf{x}_{ijk})$) on the surface $S$
    from boundary points ($\mathbf{x}_{ijk}$) to normal direction
end for
for maximum iteration do
    for all boundary do
        find the value $\phi^n(\text{cp}(\mathbf{x}_{ijk}))$ by using trilinear interpolation
    end for
    for On the band do
        Solve the diffusion equation
    end for
    for On the band do
        Solve the closed-form
    end for
end for
```

The main code of Fig. 5.3 is as follows:

```
iterm=0;
for i=1:nx
    for j=1:ny
        for k=1:nz
            d(i,j,k)=sqrt(x(i)^2+y(j)^2+z(k)^2)-R0;
            if (abs(d(i,j,k))<delta)
                u(i,j,k)=0; iterm=iterm+1;
                do(iterm,1)=i; do(iterm,2)=j; do(iterm,3)=k;
            else
                u(i,j,k)=1;
            end
        end
    end
end

for i=1:nx
    for j=1:ny
        for k=1:nz
            r=sqrt(x(i)^2+y(j)^2);
            p(i,j,k) = tanh((r-z(k))/(sqrt(2.0)*eps)); tmpp1(i,j,k)=p(i,j,k);
        end
    end
end

clear bx by bz
count=0;
for i=2:nx-1
    for j=2:ny-1
        for k=2:nz-1
            a=(abs(u(i+1,j,k)-u(i-1,j,k))+abs(u(i,j+1,k)-u(i,j-1,k)) ...
                +abs(u(i,j,k+1)-u(i,j,k-1)))*u(i,j,k);
            if (a>0.5)
                count=count+1; bx(count)=i; by(count)=j; bz(count)=k;
            end
        end
    end
end

for tm=1:count
    i=bx(tm); j=by(tm); k=bz(tm);
    v= -[d(i+1,j,k)-d(i-1,j,k) d(i,j+1,k)-d(i,j-1,k) d(i,j,k+1)-d(i,j,k-1)];
    v=v/sqrt(v*v’); w=[x(i),y(j),z(k)]+d(i,j,k)*v;

    ijk(tm,1) = floor((w(1)-x(1))/h)+1;
    ijk(tm,2) = floor((w(2)-y(1))/h)+1;
    ijk(tm,3) = floor((w(3)-z(1))/h)+1;
    xr(tm)=(w(1)-x(ijk(tm,1)))/h;
    yr(tm)=(w(2)-y(ijk(tm,2)))/h;
    zr(tm)=(w(3)-z(ijk(tm,3)))/h;
end

for iter=1:maxit
    for tm=1:count
        i=bx(tm); j=by(tm); k=bz(tm);
        p(i,j,k)= ...
           (1-xr(tm))*(1-yr(tm))*(1-zr(tm))*p(ijk(tm,1),ijk(tm,2),ijk(tm,3)) ...
            +(1-xr(tm))*(1-yr(tm))*zr(tm)*p(ijk(tm,1),ijk(tm,2),ijk(tm,3)+1) ...
            +(1-xr(tm))*yr(tm)*(1-zr(tm))*p(ijk(tm,1),ijk(tm,2)+1,ijk(tm,3)) ...
            +(1-xr(tm))*yr(tm)*zr(tm)*p(ijk(tm,1),ijk(tm,2)+1,ijk(tm,3)+1) ...
            +xr(tm)*(1-yr(tm))*(1-zr(tm))*p(ijk(tm,1)+1,ijk(tm,2),ijk(tm,3)) ...
            +xr(tm)*(1-yr(tm))*zr(tm)*p(ijk(tm,1)+1,ijk(tm,2),ijk(tm,3)+1) ...
            +xr(tm)*yr(tm)*(1-zr(tm))*p(ijk(tm,1)+1,ijk(tm,2)+1,ijk(tm,3)) ...
            +xr(tm)*yr(tm)*zr(tm)*p(ijk(tm,1)+1,ijk(tm,2)+1,ijk(tm,3)+1);
    end

    for ii=1:iterm
        i=do(ii,1); j=do(ii,2); k=do(ii,3);
        np(i,j,k)=p(i,j,k)+dt*(p(i-1,j,k)+p(i+1,j,k)+p(i,j-1,k)+p(i,j+1,k) ...
            +p(i,j,k-1)+p(i,j,k+1)-6*p(i,j,k))/h^2;
    end

    for ii=1:iterm
        i=do(ii,1); j=do(ii,2); k=do(ii,3);
        p(i,j,k)= ...
        np(i,j,k)/sqrt(exp(-2*dt/eps^2)+np(i,j,k)^2*(1-exp(-2*dt/eps^2)));
    end
    tmpp2 = p;

    ns = 1;
    if mod(iter,ns)==0
      clf; T = find(tmpp1-tmpp2==0); tmpp2(T) = NaN;
      Y = p(:); it = abs(d) < delta;
      [xx,yy,zz]=meshgrid(y,x,z);
      xt=xx(:); yt=yy(:); zt=zz(:);
      xt = xt(it(:)); yt = yt(it(:)); zt = zt(it(:)); Y = Y(it(:));

      ppS=isosurface(xx,yy,zz,d,0); hold on; ppC=isosurface(xx,yy,zz,p,0);
      paS = patch(ppS); paC = patch(ppC);
      isonormals(xx,yy,zz,d,paS); isonormals(xx,yy,zz,d,paC);
      ptS = ppS.vertices; ptC = ppC.vertices;

      X = [xt,yt,zt];
      XIS = [ptS(:,1),ptS(:,2),ptS(:,3)]; XIC = [ptC(:,1),ptC(:,2),ptC(:,3)];
```

```
   cdataS = griddatan(X,Y,XIS); cdataC = griddatan(X,Y,XIC);
   set(gca,'CLim',[-1.1 1.1]);
   set(paS,'FaceColor','interp','FaceVertexCData',cdataS,'EdgeColor','none');
   set(paC,'FaceColor','none','FaceVertexCData',cdataC,'EdgeColor','none');
   axis image; axis([x(1) x(end) y(1) y(end) z(1) z(end)])
   daspect([1 1 1]); view(-34,20); axis off camlight; lighting phong;
   colormap jet; drawnow;
  end
end
```

5.1.4.2. *Motion by mean curvature on a sphere.* We consider the motion of a circular interface on a sphere evolving according to in-surface curvature motion [88]. In Fig. 5.1(a), R and r are radii of the sphere and the circle, respectively.

$$V = -\frac{1}{r}$$

be the curvature of the circle, x be the tangential component of V, and u be the component of x toward the center of the circle. Then, we can find that

$$u = -\frac{1}{r}\cos^2\theta$$

from Fig. 5.1(b). Now, the governing equation for the motion of a circular interface on a sphere is given as:

$$\frac{dr}{dt} = u = -\frac{1}{r}\cos^2\theta = \frac{r^2 - R^2}{rR^2}. \tag{5.7}$$

By solving Eq. (5.7), we have the analytic solution for $r(t)$,

$$r(t) = \sqrt{R^2 - (R^2 - r_0^2)e^{\frac{2t}{R^2}}}. \tag{5.8}$$

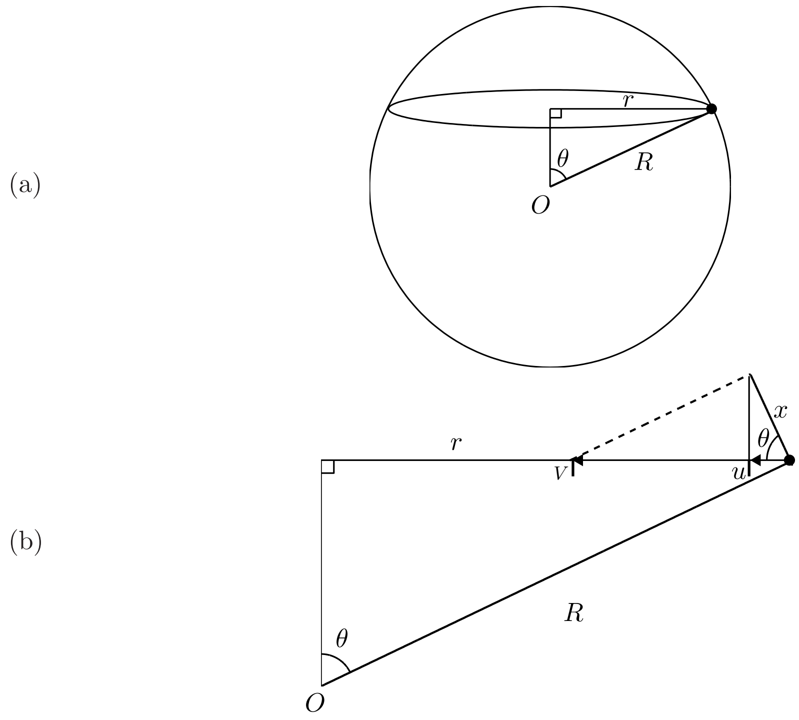


FIGURE 5.1. Schematic of (a) parameters are on the sphere and (b) detailed description for some part of (a).

To show the convergence test, we adopt difference mesh size $n = 21,\ 61$, and 101. Other conditions are same as spatial step size $h = 2/(n-1)$, sphere radius $R = 1$, circle radius $r = 1/\sqrt{2}$, total simulation time $T = 0.2$ on the computational domain $\Omega = (-1,1) \times (-1,1) \times (-1,1)$.

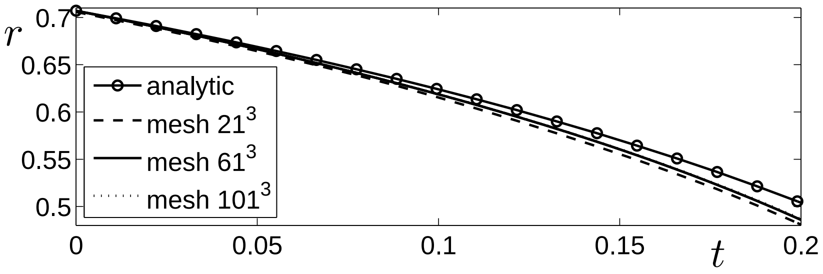


FIGURE 5.2. Convergence test for different mesh size.

Figure 5.2 shows the convergence result with respect to grid size. Each of the mesh sizes (21^3, 61^3, and 101^3) has the radius value as 0.4809, 0.4859, and 0.4869, respectively. The errors from the analytic radius (0.5041) at $t = 0.2$ are 0.0232, 0.0182, and 0.0171 on the mesh 21^3, 61^3, and 101^3, respectively. Therefore, as we refine the mesh, the numerical result is getting closer to the analytic radius. Since we have a stability condition, $\Delta t \leq h^2/6$, we choose the proper Δt as $0.1h^2$ for the simulation.

In this simulation, the initial condition surface is defined using signed distance function:

$$d(x, y, z) = \sqrt{x^2 + y^2 + z^2} - R, \tag{5.9}$$

where R is sphere radius 1. And phase-fields function:

$$\phi(x, y, z) = \tanh \frac{\sqrt{x^2 + y^2} - z}{\sqrt{2}\epsilon}, \tag{5.10}$$

where $\epsilon = 0.1$ is transition layer. We also use other parameters are: circle radius $r = 1/\sqrt{2}$, mesh size $n = 51^3$, spatial step size $h = 2/(n - 1)$, time step $\Delta t = 0.1h^2$ on the computational domain $\Omega = (-1, 1)^3$. Figures 5.3(a)–(c) represent the evolution for each time (a) $t = 0$, (b) $t = 800\Delta t$, and (c) $t = 1600\Delta t$. The first column shows the surface view and second column represents grid view for calculating region only on the sphere.

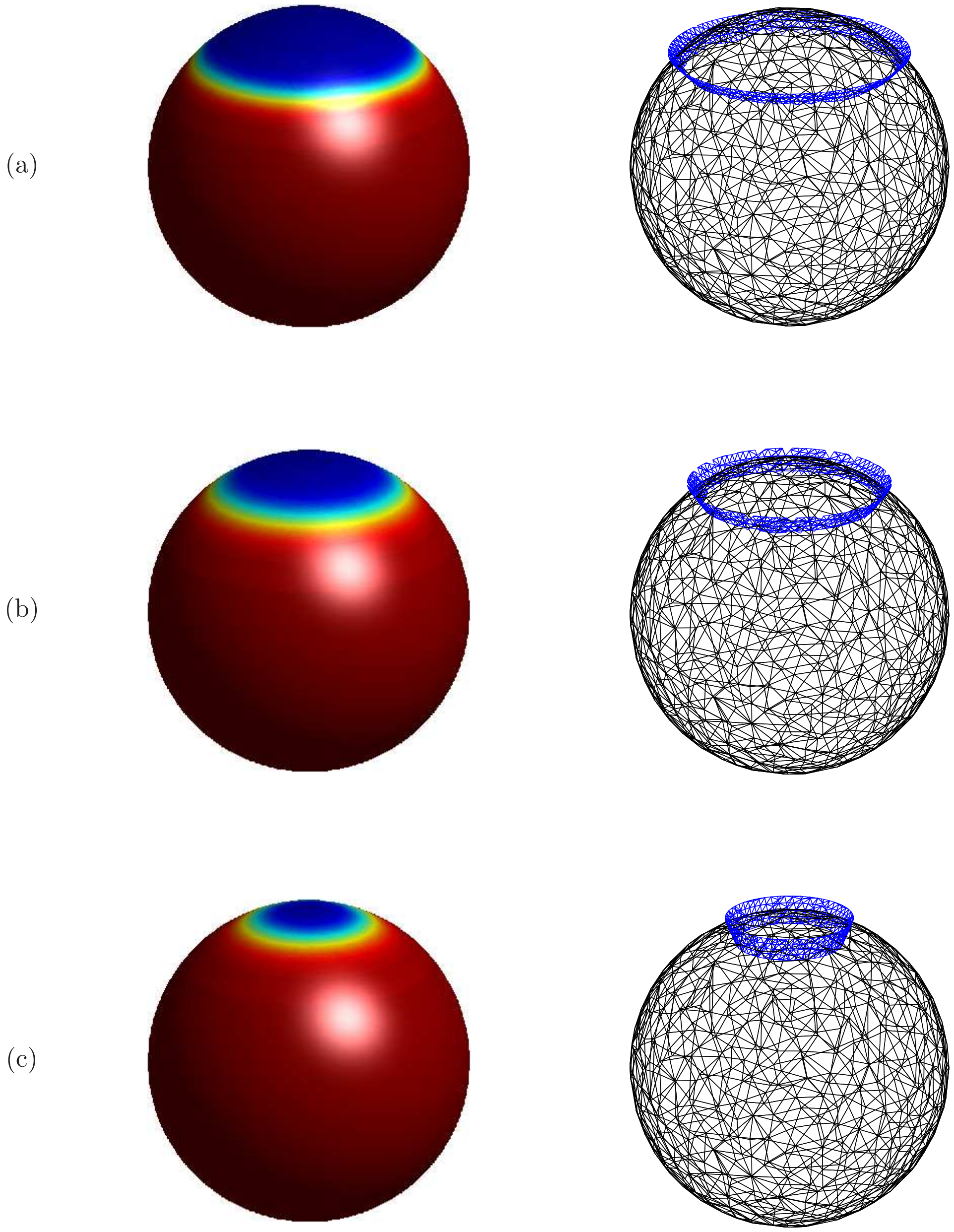


Figure 5.3. First column represents the surface view and second column shows grid view at the (a) $t = 0$, (b) $t = 800\Delta t$, and (c) $t = 1600\Delta t$.

5.1.4.3. *Motion by mean curvature on a torus.* In this section, we investigate motion by mean curvature on a torus. For all numerical simulation of torus, we use the following signed distance function as

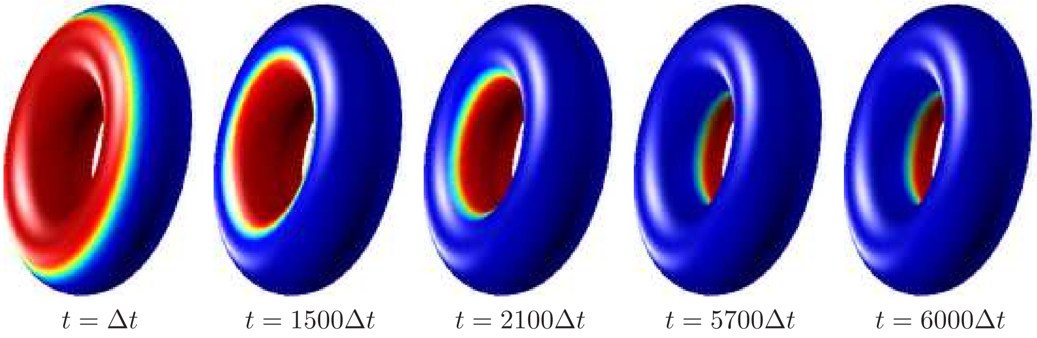
$$d(x, y, z) = \sqrt{(\sqrt{x^2 + y^2} - R)^2 + z^2} - r, \tag{5.11}$$

where R represents the circumferential radius which is the size of the circle that constitutes the center of the torus tube and r is the tube radius. We take the two radii to be $R = 0.7$ and $r = 0.3$. We also use the following parameters as $\epsilon = \epsilon_4$, $h = 2/50$, $\Delta t = 0.1h^2$, $\delta = 1.1\sqrt{3}h$, and 51×51 grids on the computational domain $\Omega = (-1, 1) \times (-1, 1) \times (-0.3, 0.3)$.

For the first example, we set the initial condition to be

$$\phi(x, y, z) = \begin{cases} 1, & \text{if } z > 0 \text{ and } \sqrt{x^2 + y^2} < 0.9 \\ -1, & \text{otherwise.} \end{cases} \tag{5.12}$$

The initial curve is taken to be the intersection of the torus with the condition (5.12) as displayed in Fig. 5.4.



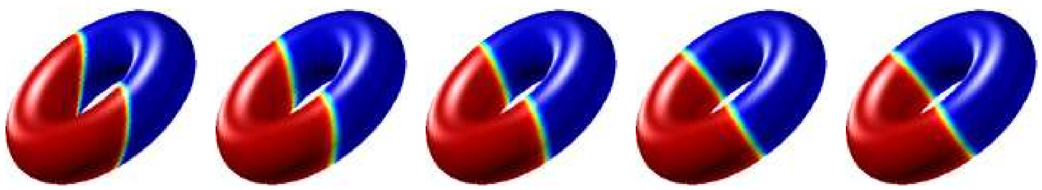
$t = \Delta t$ $t = 1500\Delta t$ $t = 2100\Delta t$ $t = 5700\Delta t$ $t = 6000\Delta t$

FIGURE 5.4. Numerical solutions on the torus with the initial condition (5.12) at each time t. Red and blue regions denote $\phi = 1$ and $\phi = -1$.

Figure 5.4 illustrates the temporal behavior of the interfaces on the surface. In this figure, computational times are written below each one. The interfaces shrink by mean curvature as time evolves. In the last frame of Fig. 5.4, the interface reaches to the the steady state.

In the second example, we use the following initial condition as

$$\phi(x,y,z) = \begin{cases} 1, & \text{if } \sqrt{x^2+z^2} < 1.2 \text{ and } y+z > 0.1 \\ -1, & \text{otherwise.} \end{cases} \tag{5.13}$$



$t = 2\Delta t$ $t = 300\Delta t$ $t = 600\Delta t$ $t = 1500\Delta t$ $t = 1800\Delta t$

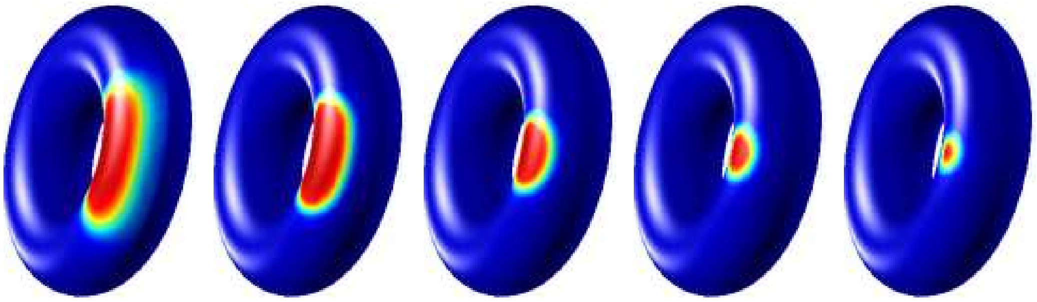
FIGURE 5.5. Numerical solutions on the torus with the initial condition (5.13) at each time t. Red and blue regions denote $\phi = 1$ and $\phi = -1$.

Figure 5.5 displays the rendered surface of the numerical solution ϕ at $t = 2\Delta t$, $300\Delta t$, $600\Delta t$, $1500\Delta t$, and $1800\Delta t$, respectively. As time goes on, the interfaces evolve according to curvature motion. In final frame which represents a steady state, the two interfaces with ring-shape have zero curvature on the torus.

As third example, we consider the following initial condition as

$$\phi(x,y,z) = \begin{cases} 1, & \text{if } \sqrt{(x-0.7)^2+0.25z^2} < 0.25 \text{ and } z > 0.0 \\ -1, & \text{otherwise.} \end{cases} \tag{5.14}$$

Figure 5.6 shows the motion by mean curvature on torus at time t. When approaching steady state, the interface approaches a circle before disappearing.



$t = \Delta t$ $t = 120\Delta t$ $t = 280\Delta t$ $t = 360\Delta t$ $t = 400\Delta t$

FIGURE 5.6. Numerical solutions on the torus with the initial condition (5.14) at each time t. Red and blue regions denote $\phi = 1$ and $\phi = -1$.

5.1.4.4. *Phase ordering on surfaces.* Although our primary goal is to simulate motion by mean curvature on surfaces, we present phase ordering on curved surfaces. Phase separation on curved membranes are intriguing physical phenomena, ranging from nonequilibrium statistical physics and hydrodynamic theories to cell biology (see [72] and references therein). One example is lipid bilayers [89].

First, we consider the process of phase separation on the surface of a sphere of radius R. The signed distance function is given as

$$d(x, y, z) = \sqrt{x^2 + y^2 + z^2} - R,$$

where $R = 1$. Then, the spherical surface is defined as the zero level set, i.e.,

$$S = \{(x, y, z) \big| d(x, y, z) = 0\}.$$

The narrow band domain is defined as

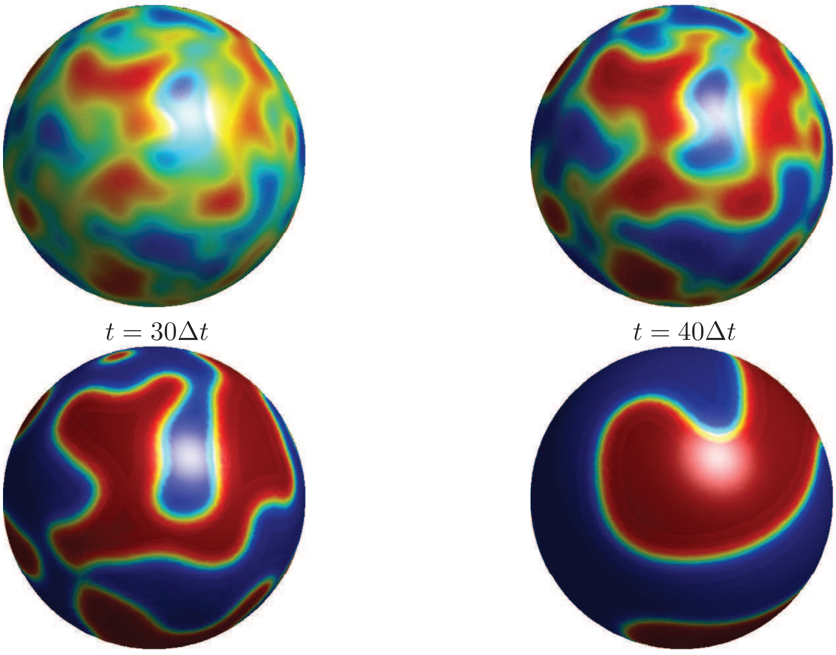
$$\Omega_\delta = \{(x, y, z) \big| |d(x, y, z)| < \delta\},$$

where $\delta = 1.1\sqrt{3}h$. The space step $h = 2R/50$ and the time step $\Delta t = 0.1h^2$ are used. The domain is $\Omega = (-1 - 4h, 1 + 4h) \times (-1 - 4h, 1 + 4h) \times (-1 - 4h, 1 + 4h)$.

The initial condition is

$$\phi(x, y, z) = 0.5\text{rand}(x, y, z),$$

where rand(x, y, z) is a uniformly distributed random number between -1 and 1. Figure 5.7 shows the time evolution of morphologies.

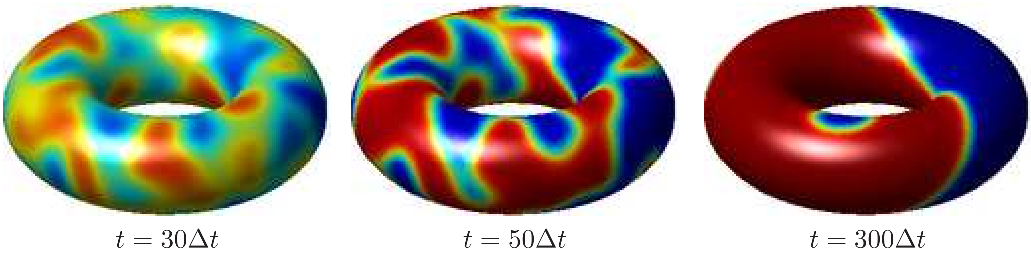


$t = 30\Delta t$ $t = 40\Delta t$

$t = 80\Delta t$ $t = 500\Delta t$

FIGURE 5.7. Temporal evolution of numerical solutions on a sphere. The computational times are shown below each figure.

Next, we consider the process of phase separation on the surface of a torus with the signed distance function (5.11). And we choose the same value for numerical parameter as in the previous tests in Section 5.1.4.3). The initial condition is $\phi(x, y, z) = 0.5\text{rand}(x, y, z)$, where rand$(x, y, z)$ is a uniformly distributed random number between -1 and 1.



$t = 30\Delta t$ $t = 50\Delta t$ $t = 300\Delta t$

FIGURE 5.8. Temporal evolution of numerical solutions on a torus. The computational times are shown below each figure.

Figure 5.8 shows the time evolution of morphologies of torus. As shown in this figure, spinodal decomposition proceeds to completion without arresting when approaching to steady state.

5.1.5. Summary. By using the Allen–Cahn equation, we developed a fast and accurate numerical method for motion by mean curvature of curves on a surface in the three-dimensional space. We use operator splitting method and solved the Allen–Cahn equation on a narrow band domain. The domain boundary are defined by an trilinear interpolation algorithm with the closest point method. To solve efficiently, we first solve the heat equation by using an explicit scheme, and then update the solution by using a closed-form solution. Through various numerical experiments such as motion by mean curvature or phase ordering on sphere and torus, we showed that the proposed numerical algorithm is computationally efficient and accurate.

5.2. Conservative Allen–Cahn equation

5.2.1. Introduction. The Allen–Cahn (AC) equation is a second-order nonlinear parabolic partial differential equation, originally proposed by Allen and Cahn [2] to describe the phase separation in binary alloys. The classical AC equation is

$$\frac{\partial \phi}{\partial t}(\mathbf{x},t) = -M\left(\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2} - \Delta\phi(\mathbf{x},t)\right). \tag{5.15}$$

Detailed description of the AC equations, please refer to Chapter 2.1 and 5.1. The AC equation has been used to model many phenomena such as crystal growth [24], image inpainting [31, 58], image segmentation [13, 46], and tumor growth [105] on flat surfaces. Also, the AC equation has been studied on non-flat surfaces [25]. Although the conservative Allen–Cahn (CAC) equation was solved and studied on flat surfaces [49, 50, 90, 104], there is no research work to solve the CAC equation on non-flat surfaces to the authors' knowledge.

Therefore, the main purpose of this article is to develop a fast and computationally efficient finite difference method for the CAC equation on non-flat surfaces in three-dimensional space. The problem for partial differential equations on the surfaces has been studied in various fields such as image processing [40, 78] and biological modeling [3, 78, 96]. Therefore, solving the CAC equation on the surfaces is geometrically and numerically an important issue. We employ a hybrid explicit numerical method, which is based on an operator splitting method and we solve the resulting discrete equations on a narrow band domain. We use an idea of the closest point method [88] to define the boundary condition. The numerical results demonstrate that the proposed algorithm is accurate and efficient.

5.2.2. Governing equations. In this section, we describe the CAC equation [49]

$$\frac{\partial \phi}{\partial t}(\mathbf{x},t) = -\frac{F'(\phi(\mathbf{x},t))}{\epsilon^2} + \Delta\phi(\mathbf{x},t) + \beta\sqrt{F(\phi(\mathbf{x},t))}, \tag{5.16}$$

where $\phi(\mathbf{x},t)$ is the order parameter, $\sqrt{F(\phi)} = 0.5|\phi^2 - 1|$, ϵ is thickness of transition layer, and β is Lagrange multiplier to conserve the total mass. Let S be a smooth surface in $\mathbb{R}^3$ and Ω_δ be a neighborhood of S which is defined as

$$\Omega_\delta = \{\mathbf{y}|\mathbf{x} \in S, \mathbf{y} = \mathbf{x} + \theta\mathbf{n}(\mathbf{x}) \text{ for } |\theta| < \delta\}$$

where $\mathbf{n}$ is unit normal vector on the surface S and δ is a positive constant.

We detailed describe how to find the boundary points by using closest point method in Chapter 4. In that chapter, we explain how to define the Laplacian and boundary condition. To help the reader understand the idea, we explain the algorithm in the two-dimensional space. We define the surface S, boundary $\partial\Omega_\delta$, band width 2δ, and band domain Ω_δ. When we calculate $\Delta_h\phi_{ijk}^n$, the boundary values are needed. For each point $\mathbf{x} \in \partial\Omega_\delta$, using the trilinear interpolation we define the closest point function cp : $\partial\Omega_\delta \rightarrow S$ which assigns the value of closest point cp$(\mathbf{x})$. Therefore, we define the boundary condition as

$$\phi(\mathbf{x},t) = \phi(\text{cp}(\mathbf{x}),t) \text{ on } \partial\Omega_\delta. \tag{5.17}$$

In this chapter, we describe how to solve the CAC equation on the narrow band domain.

5.2.3. Discretization. In this section, we represent the numerical schemes for the CAC equation on the narrow band domain, Ω_δ. The CAC equation is descretized on the three-dimensional domain $\Omega = (a,b) \times (c,d) \times (e,f)$. The uniform spatial step size is $h = (b-a)/N_x = (d-c)/N_y = (f-e)/N_z$, where N_x, N_y, and N_z are the numbers of cells in x-, y-, and z-directions, respectively. Discrete domain Ω^h is defined as

$$\Omega^h = \{\mathbf{x}_{ijk} = (x_i, y_j, z_k) = (a + hi, c + hj, e + hk)\big| 0 \le i \le N_x,\ 0 \le j \le N_y,\ 0 \le k \le N_z\}$$

and

$$\Omega^h_\delta = \{\mathbf{x}_{ijk}\big||\psi_{ijk}(\mathbf{x}_{ijk})| < \delta\}$$

is the discrete narrow band domain where ψ is a signed distance function. The narrow band domain must contain the stencil, we should take $\delta \ge \sqrt{3}h$. Let boundary points be defined as

$$\partial\Omega^h_\delta = \{\mathbf{x}_{ijk}\big|I_{ijk}|\nabla_h I_{ijk}| \ne 0\},$$

where

$$\nabla_h I_{ijk} = \frac{I_{i+1,jk} - I_{i-1,jk}, I_{i,j+1,k} - I_{i,j-1,k}, I_{ij,k+1} - I_{ij,k-1}}{2h}.$$

Here, $I_{ijk} = 0$ if $\mathbf{x}_{ijk} \in \Omega^h_\delta$; otherwise $I_{ijk} = 1$.

Let ϕ^n_{ijk} be approximations of $\phi(\mathbf{x}_{ijk}, n\Delta t)$, where $\Delta t = T/N_t$ is the time step, T is the final time, and N_t is the total number of time steps. On Ω^h_δ, we define a discrete L^2-norm error as

$$\|\phi\|_{L^2} = \sqrt{\frac{1}{\#\Omega^h_\delta} \sum_{\mathbf{x}_{ijk} \in \Omega^h_\delta} \phi^2_{ijk}},$$

where $\#\Omega^h_\delta$ is the number of points on the band. We consider the discretization of the CAC Eq. (5.16). First we solve the AC equation which is obtained by using an operator splitting method. We solve the diffusion term on the narrow band domain Ω^h_δ by using the explicit Euler's method with the boundary condition $\phi^n_{ijk} = \phi^n(\text{cp}(\mathbf{x}_{ijk}))$ on $\partial\Omega^h_\delta$:

$$\frac{\phi^*_{ijk} - \phi^n_{ijk}}{\Delta t} = \Delta_h \phi^n_{ijk}. \tag{5.18}$$

Here, we use the standard Laplacian,

$$\Delta_h \phi_{ijk} = \frac{\phi_{i+1,jk} + \phi_{i-1,jk} + \phi_{i,j+1,k} + \phi_{i,j-1,k} + \phi_{ij,k+1} + \phi_{ij,k-1} - 6\phi_{ijk}}{h^2}.$$

Note that we can use fully implicit or Crank–Nicolson type semi-implicit numerical schemes for the diffusion equation. In this work, we use the fully explicit scheme for simplicity and accuracy. We define the closest point $\text{cp}(\mathbf{x}_{ijk})$ as

$$\text{cp}(\mathbf{x}_{ijk}) = \mathbf{x}_{ijk} - \frac{\nabla_h|\psi_{ijk}|}{|\nabla_h|\psi_{ijk}||}|\psi_{ijk}|, \tag{5.19}$$

where ψ is a signed distance function. The closet point $\text{cp}(\mathbf{x}_{ijk})$ is generally not on a grid point. Therefore, we use the trilinear interpolation algorithm to obtain the values $\phi^n(\text{cp}(\mathbf{x}_{ijk}))$ in Chapter 4.

Next, we solve the reaction term from the following closed-form solution on the discrete narrow band domain,

$$\phi_{ijk}^{n+1,**} = \frac{\phi_{ijk}^*}{\sqrt{e^{-2\Delta t/\epsilon^2} + (\phi_{ijk}^*)^2(1 - e^{-2\Delta t/\epsilon^2})}}. \tag{5.20}$$

Finally, we conserve the total mass. Let

$$\phi_{ijk}^{n+1} = \phi_{ijk}^{n+1,**} + \beta\sqrt{F(\phi_{ijk}^{n+1,**})}, \tag{5.21}$$

where

$$\beta = \frac{\sum_{\mathbf{x}_{ijk}\in\Omega_\delta^h}(\phi_{ijk}^0 - \phi_{ijk}^{n+1,**})}{\sum_{\mathbf{x}_{ijk}\in\Omega_\delta^h}\sqrt{F(\phi_{ijk}^{n+1,**})}}. \tag{5.22}$$

Here, we have used the total mass conserving constraint,

$$\sum_{\mathbf{x}_{ijk}\in\Omega_\delta^h}\phi_{ijk}^0 = \sum_{\mathbf{x}_{ijk}\in\Omega_\delta^h}\phi_{ijk}^{n+1}.$$

Since we want that the initial mass to be the same, β is defined as Eq. (5.22). Therefore, we obtain the ϕ_{ijk}^{n+1} as follows. Our mass correction step is only corrected on the interfacial transition layer, however in the previous research, the mass is corrected by shifting the whole phase-field values [52]. Since we solve the CAC equation only on the narrow band domain and use the explicit scheme, computational cost is low.

As a similar method, there is a level set based Eulerian method for solving partial differential equations on general geometries [35]. There are a couple of differences between our algorithm and the one in [35]. Whenever, the stencil of the numerical schemes needs a grid point outside the narrow band, $\nabla\psi \cdot \nabla\phi = 0$ is used to prescribe the value of ϕ at that grid point. For the AC equation without the Lagrange multiplier, Eq. (5.15), the following modified equation is used:

$$\frac{\partial \phi}{\partial t} = -\frac{F'(\phi)}{\epsilon^2} + (\tilde{P}\nabla) \cdot (\tilde{P}\nabla\phi) + \nabla \cdot (\nabla\psi \otimes \nabla\psi\nabla\phi), \tag{5.23}$$

where $\tilde{P} = (I - \psi D^2\psi)^{-1}(I - \nabla\psi \otimes \nabla\psi)$ and $D^2\psi$ is the Hessian of ψ. For more details, see [35]. Compared to this level set based method, our proposed algorithm is simpler and can naturally deal with mass conservation property by using the discrete Lagrange multiplier, $\beta\sqrt{F(\phi_{ijk}^{n+1,**})}$ in Eq. (5.21).

5.2.4. Numerical experiments. We show a number of numerical experiments to demonstrate the accuracy and efficiency of the proposed numerical algorithm. In an equilibrium profile, ϕ varies from -0.9 to 0.9 over a distance of about $2\sqrt{2}\epsilon \tanh^{-1}(0.9)$. If we take ϵ as $\epsilon_m = mh/[2\sqrt{2}\tanh^{-1}(0.9)] \approx 0.24mh$, then we have approximately mh transition layer [26]. Unless otherwise stated, we use the parameters as $\epsilon = \epsilon_4$, mesh size $141 \times 141 \times 141$, spatial step size $h = 0.02$, time step size $\Delta t = 0.1h^2$, and the half of narrow band width $\delta = 1.1\sqrt{3}h$ on the computational domain $\Omega = (-1.4, 1.4) \times (-1.4, 1.4) \times (-1.4, 1.4)$ in all numerical simulations.

5.2.4.1. *Quantitative comparison.* In this section, we quantitatively compare the numerical results of our algorithm with the analytic solutions to show the accuracy. We consider two spherical caps on a sphere with radius R, see Fig. 5.9. Here, O is origin point.

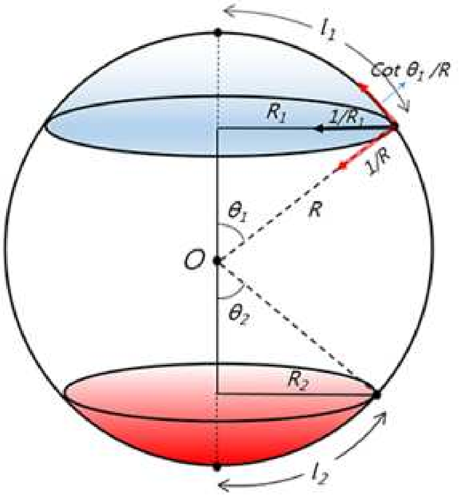


FIGURE 5.9. Two spherical caps on a sphere with radius R.

Let $R_1(t)$ and $R_2(t)$ be the radii of two spherical caps at time t. Let $l_1(t) = R\theta_1(t)$ and $l_2(t) = R\theta_2(t)$ be the arc lengths at time t, where

$$\theta_1(t) = \sin^{-1}\left(\frac{R_1(t)}{R}\right)$$

and

$$\theta_2(t) = \sin^{-1}\left(\frac{R_2(t)}{R}\right).$$

Let us assume $R_1(0) > R_2(0)$, then we can expect that $R_1(t)$ increases until $R_2(t)$ becomes zero by the dynamics of the CAC equation. Let

$$A(t) = 2\pi R^2[1 - \cos\theta_1(t)] + 2\pi R^2[1 - \cos\theta_2(t)]$$

be the sum of the areas of the two spherical caps, which is constant by the conservative property, that is, $\mathrm{d}A(t)/\mathrm{d}t = 0$. Therefore, we obtain

$$\sin\theta_1(t)\frac{\mathrm{d}\theta_1(t)}{\mathrm{d}t} + \sin\theta_2(t)\frac{\mathrm{d}\theta_2(t)}{\mathrm{d}t} = 0. \tag{5.24}$$

We extend the evolution equations [32, 87] for the case of a flat domain to the ones on a spherical domain. The curvature of the circle with the radius R_1 is $1/R_1$. The tangential component of the curvature on the sphere is $\cot\theta_1/R$. Therefore, we can write the evolution equations of the arc lengths, $l_1(t)$ and $l_2(t)$ as

$$\frac{\mathrm{d}l_1}{\mathrm{d}t} = -\frac{\cot\theta_1}{R} + \beta(\theta_1,\theta_2), \tag{5.25}$$

$$\frac{\mathrm{d}l_2}{\mathrm{d}t} = -\frac{\cot\theta_2}{R} + \beta(\theta_1,\theta_2), \tag{5.26}$$

where β is the factor for mass conservation. By substituting Eqs. (5.25)–(5.26) into Eq. (5.24), we obtain the mass conserving factor β as follows.

$$\beta(\theta_1,\theta_2) = \frac{\cos\theta_1 + \cos\theta_2}{R(\sin\theta_1 + \sin\theta_2)}.$$

We can rewrite Eqs. (5.25) and (5.26) in terms of the radian angles:

$$\frac{\mathrm{d}\theta_1}{\mathrm{d}t} = -\frac{\cot\theta_1}{R^2} + \frac{\cos\theta_1 + \cos\theta_2}{R^2(\sin\theta_1 + \sin\theta_2)}, \tag{5.27}$$

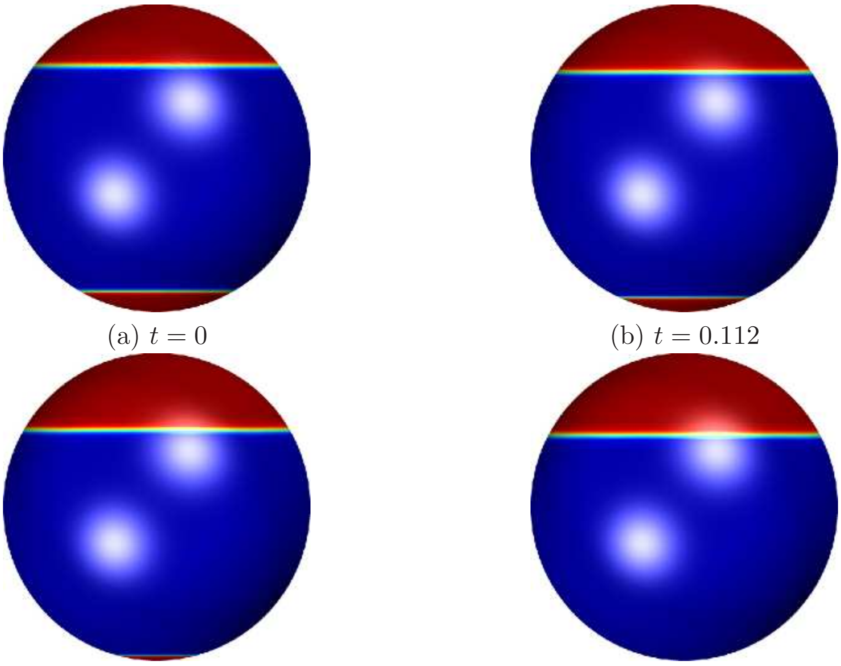
$$\frac{\mathrm{d}\theta_2}{\mathrm{d}t} = -\frac{\cot\theta_2}{R^2} + \frac{\cos\theta_1 + \cos\theta_2}{R^2(\sin\theta_1 + \sin\theta_2)}. \tag{5.28}$$

For quantitative comparison, we compute the radii $R_1(t)$ and $R_2(t)$ of numerical solution with three different space and time step sizes.

We set the initial condition as $R_1(0) = 0.8$ and $R_2(0) = 0.5$ on the unit sphere $R = 1$, i.e.,

$$\phi(x, y, z, 0) = \begin{cases} \tanh\left(\dfrac{0.8 - \sqrt{x^2 + y^2}}{\sqrt{2}\epsilon}\right), & z \geq 0 \\ \tanh\left(\dfrac{0.5 - \sqrt{x^2 + y^2}}{\sqrt{2}\epsilon}\right), & z < 0. \end{cases} \tag{5.29}$$

Note that Eq. (5.29) represents the two different spherical caps as shown in Fig. 5.10(a). Figure 5.10 shows the temporal evolution of two spherical caps on surface. As we expect, the larger one grows and the smaller one shrinks until it disappears, Fig. 5.10(d).



(a) $t = 0$

(b) $t = 0.112$

(c) $t = 0.224$

(d) $t = 0.272$

FIGURE 5.10. Temporal evolutions of two different spherical caps on the surface of the sphere with the initial condition $\phi(x, y, z, 0)$ in Eq. (5.29). The computational times are shown below each figure.

Figure 5.11 shows the temporal evolutions of the radii R_1 and R_2 with different space step size h.

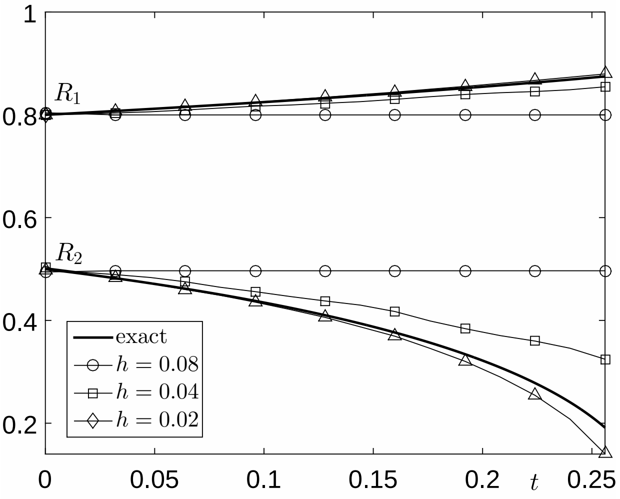


Figure 5.11. Convergence of radii R_1 and R_2 with different space step size h. Here, analytic radii are represented by the thick line.

Here, the analytic radii, which are obtained using a fourth-order Runge–Kutta method [16] for Eqs. (5.27) and (5.28), are represented by the thick line. Table 5.1 represents the errors and orders of the numerical solutions by the proposed scheme with different space step size h and time step $\Delta t = 0.1h^2$ at $T = 0.176$.

Table 5.1. Error and order of the numerical solutions by the proposed scheme with $\Delta t = 0.1h^2$ at $T = 0.176$. Note that the analytic radii are $R_1(T) = 0.84690$ and $R_2(T) = 0.35658$.

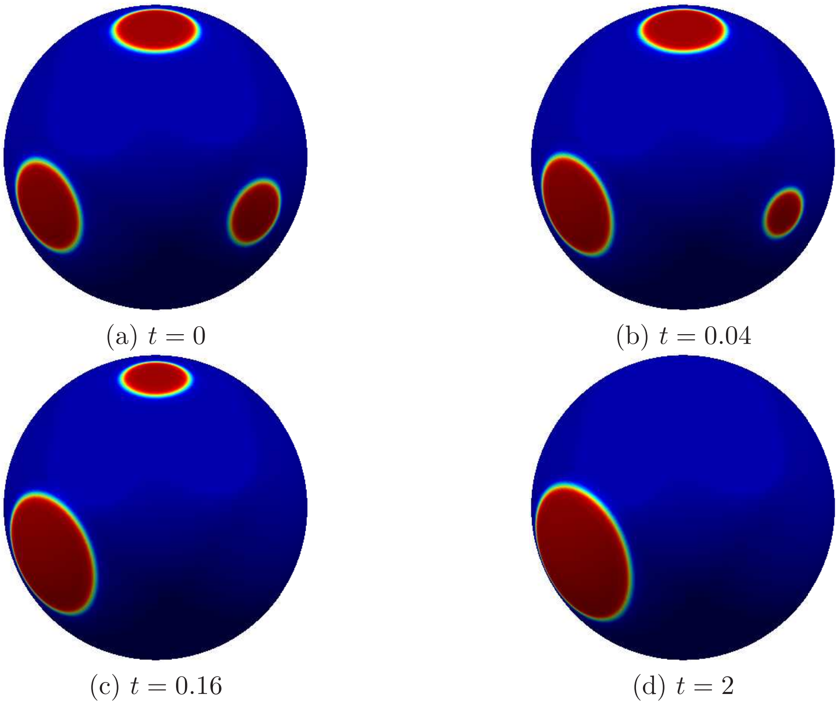
h	0.08		0.04		0.02
Radius (R_1)	0.79983		0.83522		0.84994
Error	0.04707		0.01168		0.00305
Order		2.01110		1.93827	
Radius (R_2)	0.49660		0.39890		0.34552
Error	0.14002		0.04232		0.01106
Order		1.72626		1.93601	

Table 5.1 indicates that the numerical solutions R_1 and R_2 by our scheme are almost second-order accurate in space and first-order accurate in time.

5.2.4.2. *Evolution on surfaces.* Next, we show the evolution of three spherical caps with different sizes on the unit sphere. In this test, the initial condition is set to

$$\phi(x,y,z,0)=\begin{cases}\tanh\left(\frac{R_1-\sqrt{y^2+z^2}}{\sqrt{2}\epsilon}\right), & \text{if } x>0.5,\\ \tanh\left(\frac{R_2-\sqrt{x^2+z^2}}{\sqrt{2}\epsilon}\right), & \text{if } y>0.5,\\ \tanh\left(\frac{R_3-\sqrt{x^2+y^2}}{\sqrt{2}\epsilon}\right), & \text{if } z>0.5,\end{cases} \tag{5.30}$$

where $R_1 = 0.3162$, $R_2 = 0.2236$, and $R_3 = 0.2774$. Figure 5.12 shows the temporal evolutions. We can see that smaller spherical caps shrink and the largest spherical cap survives in the long run in Fig. 5.12.


(a) $t = 0$

(b) $t = 0.04$

(c) $t = 0.16$

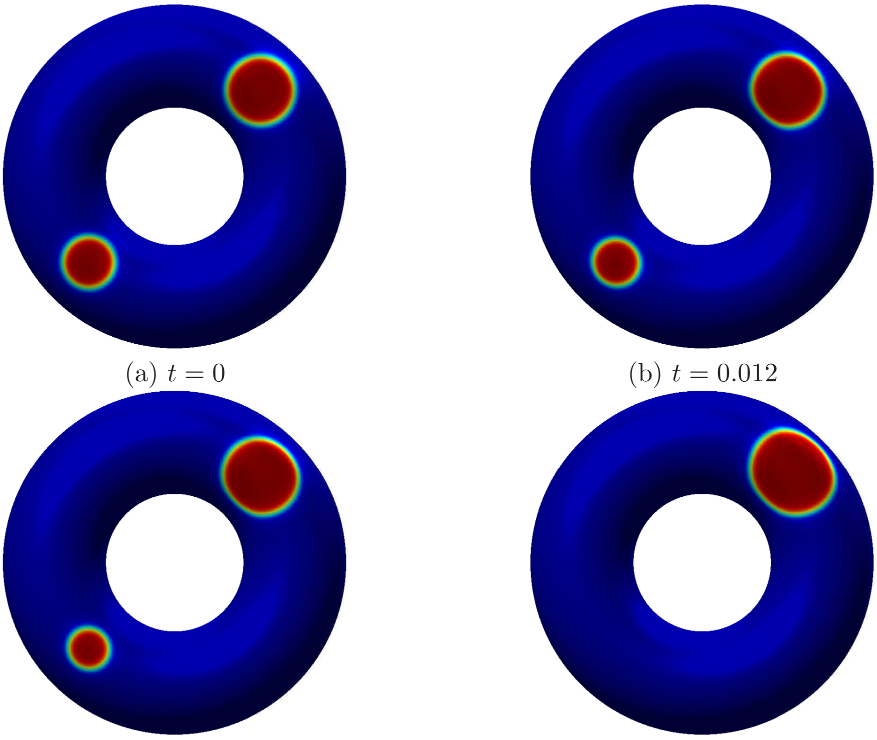
(d) $t = 2$

FIGURE 5.12. The evolutions on the surface of the sphere with initial condition $\phi(x, y, z)$, three lens shapes in Eq. (5.30). The computational times are shown below each figure.

We also simulate the evolutions on the torus surface which is embedded in the computational domain $\Omega = (-1.4, 1.4) \times (-1.4, 1.4) \times (-0.5, 0.5)$. The major and the minor radii of the torus are 0.7 and 0.3, respectively. The initial condition is defined as

$$\phi(x, y, z, 0) = \begin{cases} \tanh\left(\dfrac{0.2 - \sqrt{(x-0.5)^2 + (y-0.5)^2}}{\sqrt{2}\epsilon}\right), \ x > 0, \ y > 0, \ z > 0, \\ \tanh\left(\dfrac{0.15 - \sqrt{(x+0.5)^2 + (y+0.5)^2}}{\sqrt{2}\epsilon}\right), \ x < 0, \ y < 0, \ z > 0. \end{cases} \tag{5.31}$$

In Fig. 5.13, the smaller one shrinks and the large one grows, and then eventually the smaller one disappears.


(a) $t = 0$

(b) $t = 0.012$

(c) $t = 0.02$

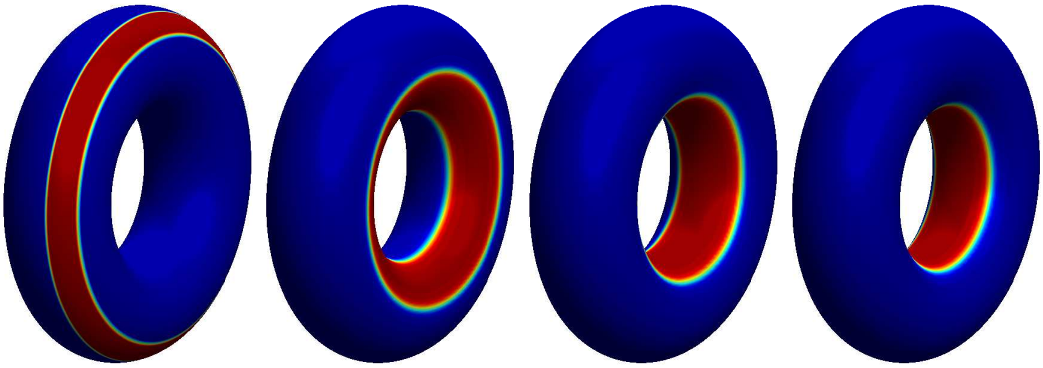
(d) $t = 2$

FIGURE 5.13. The times are shown below each figure.

Next, we consider a band on the torus surface. Figure 5.14(a) shows the initial condition on $\Omega = (-1.4, 1.4) \times (-1.4, 1.4) \times (-0.5, 0.5)$:

$$\phi(x, y, z, 0) = \begin{cases} 1 & \text{if } 0.1 < z < 0.25, \\ -1 & \text{otherwise.} \end{cases} \tag{5.32}$$

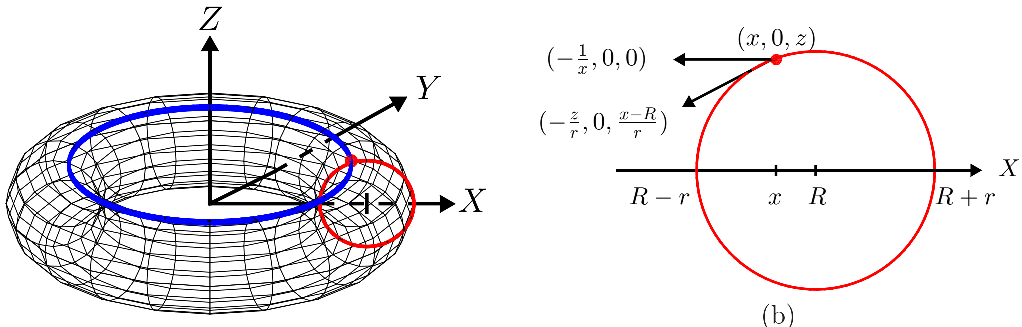
Figures 5.14(b)–(d) are the temporal evolution of the initial band on the torus.



(a) $t = 0$ (b) $t = 0.4$ (c) $t = 0.8$ (d) $t = 2$

FIGURE 5.14. The initial condition is a band on the torus. The times are shown below each figure.

To explain the evolution dynamics on the torus surface, let us consider a longitude curve as shown in Fig. 5.15(a).



(a) (b)

FIGURE 5.15. (a) shape of torus, (b) part of (a) for $y = 0$ plane.

Figure 5.15(b) represents a meridian curve on xz-plane at $y = 0$. The curvature at a point $(x, 0, z)$ at the longitude curve is

$$\left(-\frac{1}{x}, 0, 0\right).$$

The unit tangent vector at $(x, 0, z)$ is

$$\left(-\frac{z}{r}, 0, \frac{x-R}{r}\right).$$

Therefore, the tangential component of the curvature is

$$\left(-\frac{1}{x}, 0, 0\right) \cdot \left(-\frac{z}{r}, 0, \frac{x-R}{r}\right) = \frac{z}{rx}. \tag{5.33}$$

Figure 5.16 shows the magnitude of $\frac{z}{rx}$ at each point x for $R - r \leq x \leq R + r$ with $R = 0.7$ and $r = 0.3$. Here, $z = \sqrt{r^2 - (R-x)^2}$. Figure 5.16 explains the dynamic direction of the band in Fig. 5.14.

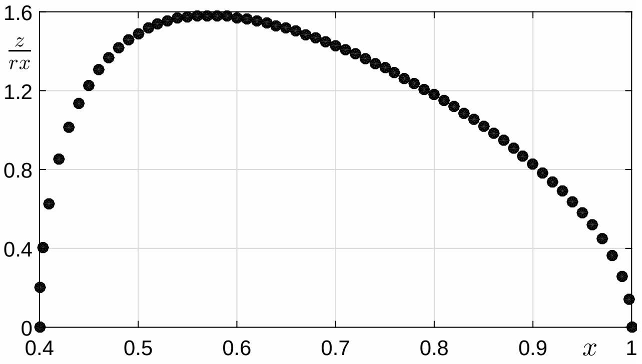


FIGURE 5.16. Magnitude of z/rx at each point x for $R - r \leq x \leq R + r$ with $R = 0.7$ and $r = 0.3$. Here, $z = \sqrt{r^2 - (R-x)^2}$.

5.2.4.3. *Effect of band width on the torus.* In this section, we investigate the effect of band width on the equilibrium state. For numerical test, we only change the band width ($2b$) and fix other parameters: $251 \times 251 \times 251$ mesh, spatial step size $h = 0.0112$ on the computational domain $(-1.4, 1.4) \times (-1.4, 1.4) \times (-1.4, 1.4)$. Initial condition is defined by

$$\phi(x, y, z, 0) = \begin{cases} 1 & \text{if } -b < x < b \text{ and } y < 0, \\ -1 & \text{otherwise.} \end{cases} \tag{5.34}$$

When the band width is 0.2, the band is snapped and becomes a lens shape as shown in Fig. 5.17(a). However, snapping does not occur in the case of band width 0.3 as shown in Fig. 5.17(b).

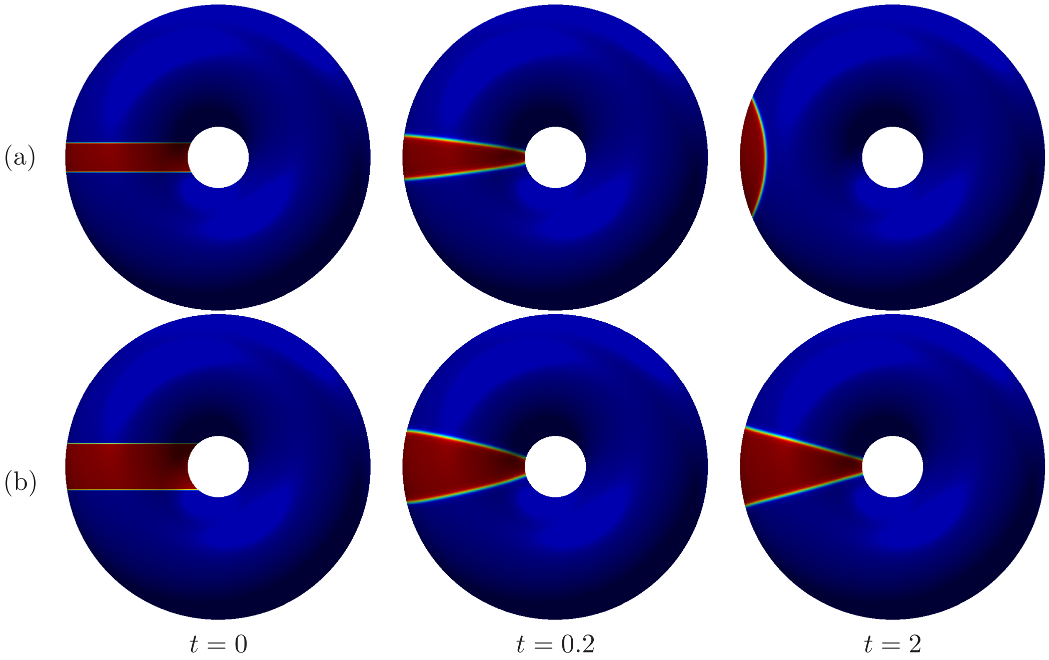


FIGURE 5.17. Comparison of the effect of band width: (a) $b = 0.1$ and (b) $b = 0.15$.

5.2.4.4. *Phase separation on sphere.* We perform the phase separation on the surface of a unit sphere. Figure 5.18 shows the time evolution of phase on the sphere.

The initial conditions are

$$\phi(x, y, z, 0) = -0.4 + 0.1\text{rand}(x, y, z)$$

and

$$\phi(x, y, z, 0) = 0.1\text{rand}(x, y, z)$$

for Fig. 5.18(a) and Fig. 5.18(b), respectively, where rand(x, y, z) is a uniformly distributed random number between -1 and 1.

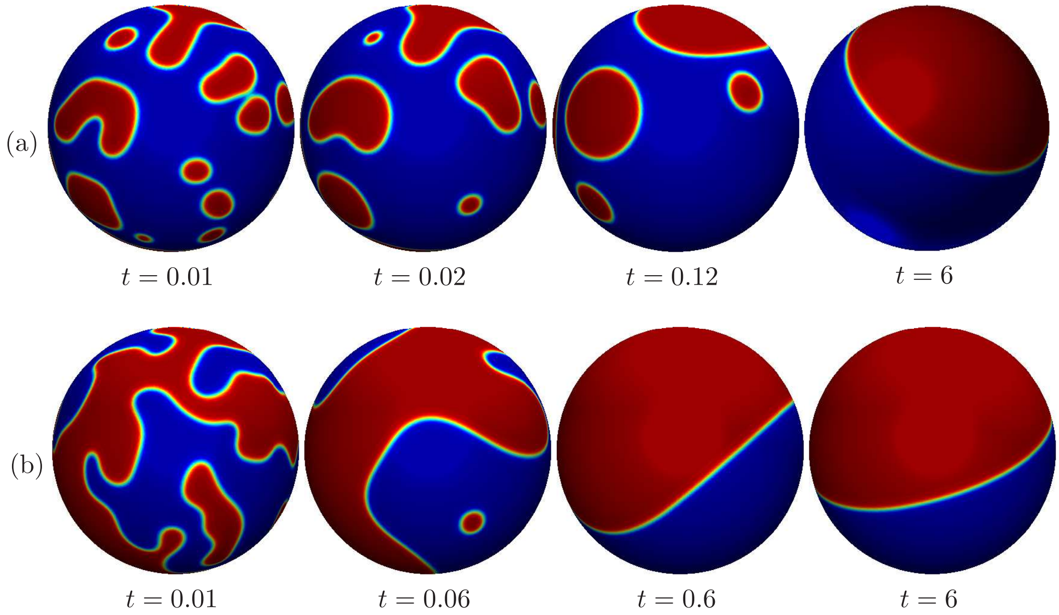


FIGURE 5.18. The temporal evolutions on the surface of the sphere with initial conditions (a) $\phi(x, y, z) = -0.4 + 0.1\text{rand}(x, y, z)$ and (b) $\phi(x, y, z) = 0.1\text{rand}(x, y, z)$. The computational times are shown below each figure.

5.2.4.5. *Phase separation on torus.* We simulate the phase separation on the torus with signed distance function as:

$$\psi(x,y,z) = \sqrt{(C-R)^2+z^2} - r, \tag{5.35}$$

where $C = \sqrt{x^2+y^2}$, R is the distance from the center of the tube to the center of the torus and r is radius of the tube. We use $R = 0.7$ and $r = 0.3$. Figure 5.19 shows the phase separation on the torus with the following initial conditions:

$$\phi(x,y,z) = -0.4 + 0.1\text{rand}(x,y,z)$$

and

$$\phi(x,y,z) = 0.1\text{rand}(x,y,z).$$

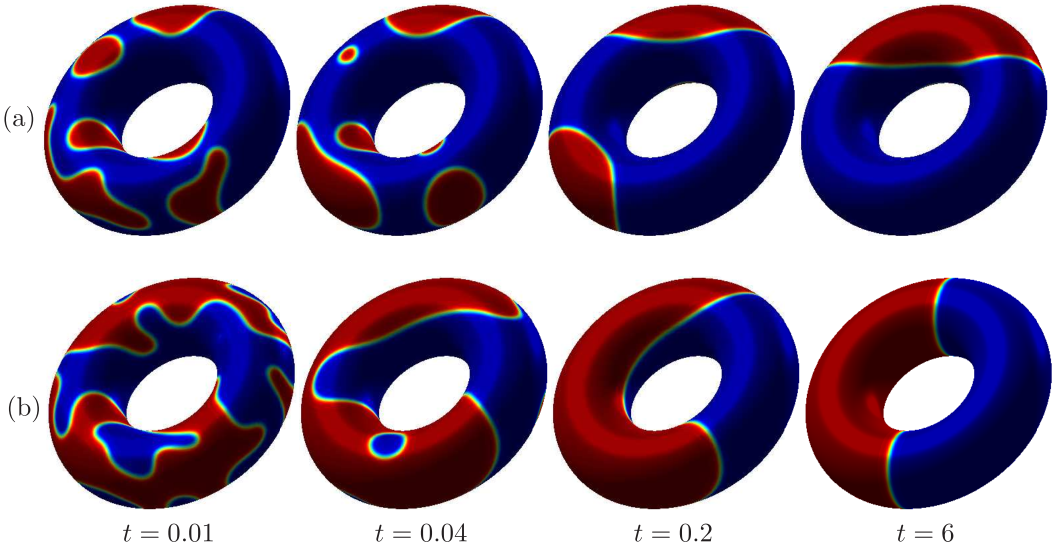


FIGURE 5.19. The evolutions on the surface of the torus with initial conditions (a) $\phi(x,y,z) = -0.4 + 0.1\text{rand}(x,y,z)$ on the top row and (b) $\phi(x,y,z) = 0.1\text{rand}(x,y,z)$ on the bottom row. The computational times are shown below each figure.

5.2.4.6. *Phase separation on spindle.* We simulate phase separation on the surface of a spindle with different average concentrations. The spindle shape is defined by parametric equations: $r = \cos x$, $x(r, \theta) = x$, $y(r, \theta) = r\cos\theta$, $z(r, \theta) = r\sin\theta$, where $|x| \leq \pi/2$ and $\theta \in [0, 2\pi)$. In this simulation, we use $h = 0.0379$ on $\Omega = (-200h, 100h) \times (-32h, 32h) \times (-32h, 32h)$. Figure 5.20 shows the time evolution of spinodal decomposition on the surface of a spindle with two different initial conditions: $\phi(x, y, z, 0) = -0.4 + 0.1\text{rand}(x, y, z)$ and $\phi(x, y, z, 0) = 0.1\text{rand}(x, y, z)$.

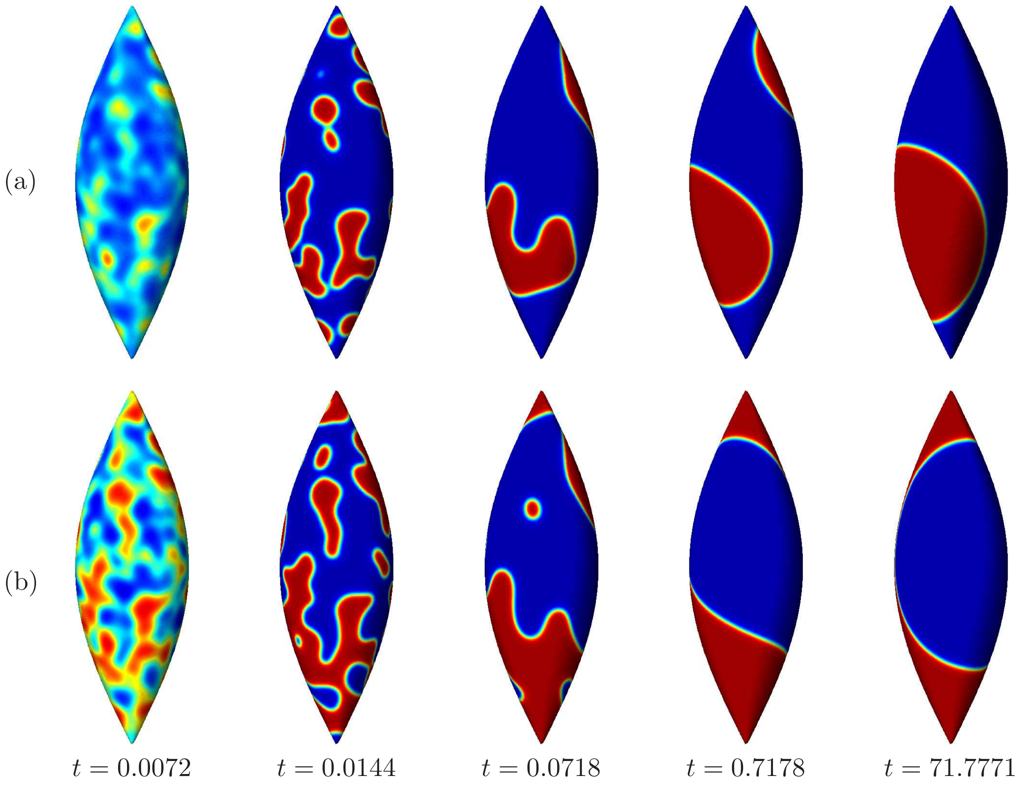


FIGURE 5.20. Temporal evolution of the phase separation on the spindle shape. The initial conditions are (a) $\phi(x, y, z) = -0.4 + 0.1\text{rand}(x, y, z)$ and (b) $\phi(x, y, z) = 0.1\text{rand}(x, y, z)$.

5.2.4.7. *Phase separation on funnel.* In this section, we simulate the phase separation on the funnel shape which is defined by parametric equations: $x(r,\theta) = r\cos\theta$, $y(r,\theta) = r\sin\theta$, $z(r,\theta) = \ln r$, where $r > 0$ and $\theta \in [0, 2\pi)$. We use $h = 0.0121$ on the computational domain $\Omega = (-1.2, 1.2) \times (-1.2, 1.2) \times (-2.4, 1.2)$. Initial conditions are

$$\phi(x, y, z, 0) = -0.4 + 0.1\text{rand}(x, y, z)$$

and

$$\phi(x, y, z, 0) = 0.1\text{rand}(x, y, z).$$

Figures 5.21(a) and (b) show the results of the temporal evolutions on the funnel shape with the average concentrations -0.4 and 0, respectively.

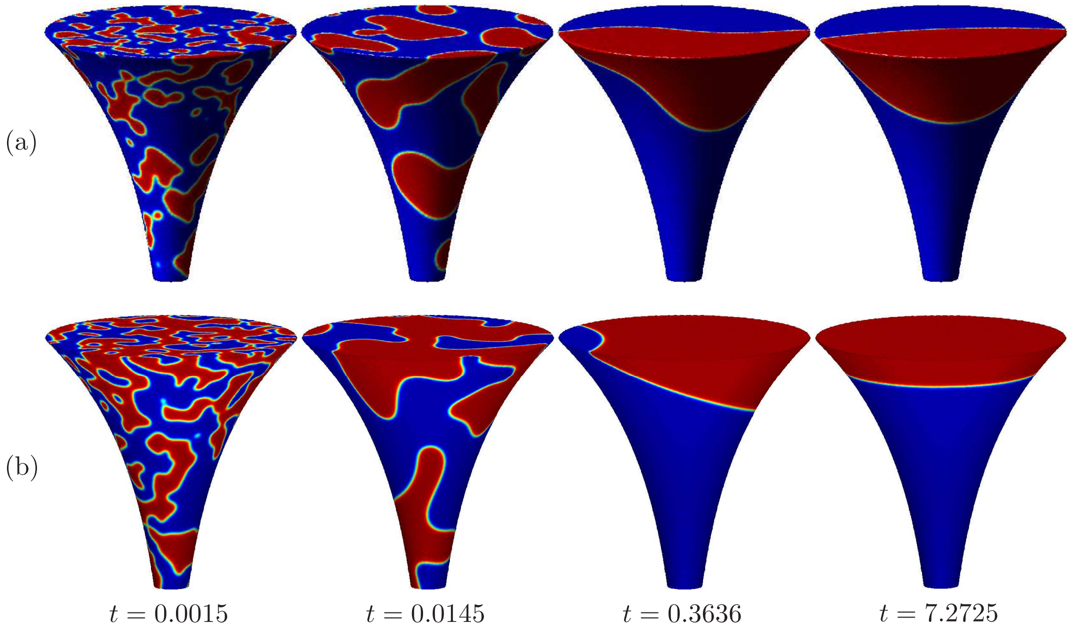


FIGURE 5.21. Temporal evolution of the phase separation on the funnel shape. The initial conditions are (a) $\phi(x, y, z) = -0.4 + 0.1\text{rand}(x, y, z)$ and (b) $\phi(x, y, z) = 0.1\text{rand}(x, y, z)$.

5.2.4.8. *Phase separation on bunny.* In this section, we perform phase separation on the surface of a bunny with different average concentrations. To perform the numerical simulation, we use $h = 0.5$ on $\Omega = (0, 68.5)^3$.

Figure 5.22 shows the temporal evolution of spinodal decomposition on the surface of the bunny with two different initial conditions:

$$\phi(x, y, z, 0) = -0.4 + 0.1\text{rand}(x, y, z)$$

and

$$\phi(x, y, z, 0) = 0.1\text{rand}(x, y, z).$$

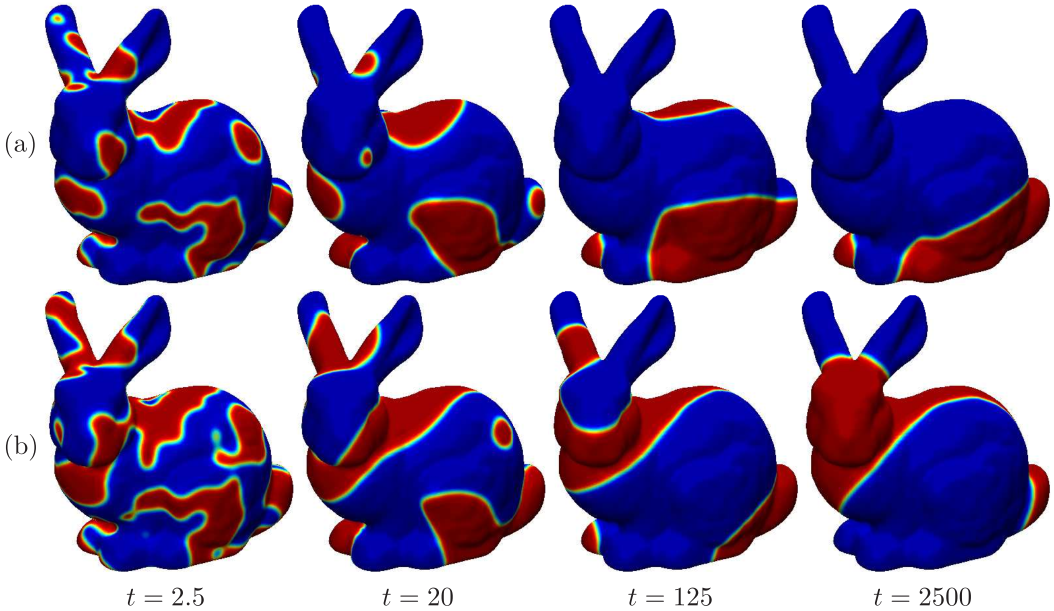


FIGURE 5.22. Morphological evolutions on the surface of the bunny with initial conditions (a) $\phi(x, y, z, 0) = -0.4 + 0.1\text{rand}(x, y, z)$ and (b) $\phi(x, y, z, 0) = 0.1\text{rand}(x, y, z)$. The computational times are shown below each figure.

5.2.4.9. *Comparison study with an implicit scheme.* In this section, we demonstrate that our proposed algorithm is fast and efficient than an implicit algorithm by comparing the computational costs. For comparison study, we consider a well-known nonlinear convex splitting scheme [30], which is unconditionally gradient stable. We take the following two steps to solve the CAC Eq. (5.16):

$$\begin{aligned} \frac{\phi_{ijk}^{n+1,*} - \phi_{ijk}^{n}}{\Delta t} &= \frac{\phi_{ijk}^{n} - (\phi_{ijk}^{n+1,*})^3}{\epsilon^2} + \Delta_h \phi_{ijk}^{n+1,*}, && (5.36) \\ \phi_{ijk}^{n+1} &= \phi_{ijk}^{n+1,*} + \beta \sqrt{F(\phi_{ijk}^{n+1,*})}, && (5.37) \end{aligned}$$

where

$$\beta = \frac{\sum_{\mathbf{x}_{ijk} \in \Omega_\delta^h} (\phi_{ijk}^{0} - \phi_{ijk}^{n+1,*})}{\sum_{\mathbf{x}_{ijk} \in \Omega_\delta^h} \sqrt{F(\phi_{ijk}^{n+1,*})}}.$$

Here, we use the Gauss–Seidel iterative method to solve Eq. (5.36). Let $\phi_{ijk}^{n+1,*,s}$ and $\phi_{ijk}^{n+1,*,s+1}$ be the approximation of $\phi_{ijk}^{n+1,*}$ before and after one Gauss–Seidel iteration. Since $(\phi_{ijk}^{n+1,*})^3$ in Eq. (5.36) is nonlinear, we linearize it as follows:

$$(\phi_{ijk}^{n+1,*,s+1})^3 = 3(\phi_{ijk}^{n+1,*,s})^2(\phi_{ijk}^{n+1,*,s+1}) - 2(\phi_{ijk}^{n+1,*,s})^3.$$

Then, we rearrange Eq. (5.36) as the Gauss–Seidel iterative form:

$$\begin{aligned} \left(\frac{1}{\Delta t} + \frac{3(\phi_{ijk}^{n+1,*,s})^2}{\epsilon^2} + \frac{6}{h^2}\right) \phi_{ijk}^{n+1,*,s+1} &= \left(\frac{1}{\Delta t} + \frac{1}{\epsilon^2}\right) \phi_{ijk}^{n} + \frac{2(\phi_{ijk}^{n+1,*,s})^3}{\epsilon^2} \\ &+ \frac{\phi_{i-1,jk}^{n+1,*,s+1} + \phi_{i+1,jk}^{n+1,*,s} + \phi_{i,j-1,k}^{n+1,*,s+1} + \phi_{i,j+1,k}^{n+1,*,s} + \phi_{ij,k-1}^{n+1,*,s+1} + \phi_{ij,k+1}^{n+1,*,s}}{h^2}. \end{aligned}$$

In this study, we update new solution ϕ_{ijk}^{n+1} when the l_2-norm of the consecutive error, $\|\phi_{ijk}^{n+1,*,s+1} - \phi_{ijk}^{n+1,*,s}\|_2$, is less than a tolerance, $tol = 10^{-6}$. Now, we compare the analytic solution with numerical results by the proposed scheme and the implicit convex splitting scheme. We use the same initial condition used in Section 5.2.4.1.

Figure 5.23 represents the time evolution of two radii with the explicit and implicit schemes. The analytic solution is shown with the solid line. Also, we put the computational times in the legend to show the efficiency of the explicit scheme.

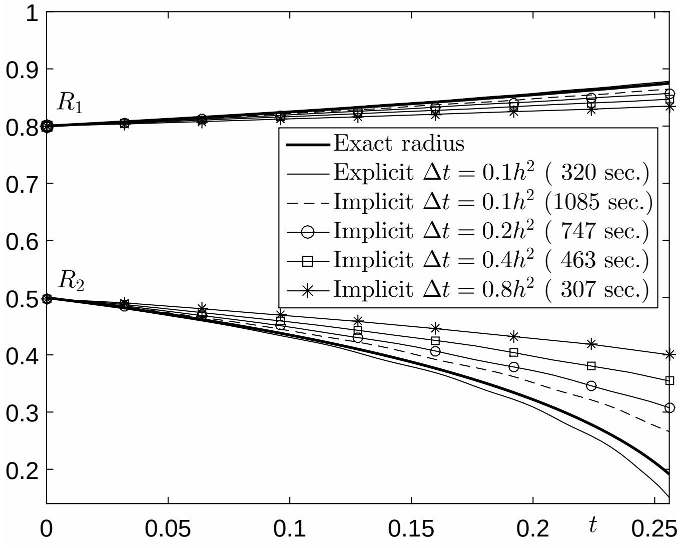


FIGURE 5.23. Temporal evolutions of two radii by the explicit and implicit schemes. The CPU times are shown in the parenthesis.

It is well known that one of the drawbacks of an explicit scheme is the severe time step restriction for stability. For example, the fourth-order nonlinear equation such as the Cahn–Hilliard equation [30] has $\Delta t = O(h^4)$ time step constraint for the explicit scheme. However, it turns out that $\Delta t \leq h^2/6$ time step constraint in three-dimensional space for the CAC equation is not severe constraint if we consider the accuracy of the numerical solutions. As shown in Fig. 5.23, it only takes $320s$ of CPU time for the explicit scheme, while it takes $1085s$ of CPU time for the implicit scheme with $\Delta t = 0.1h^2$. If we compare the results from the implicit ($\Delta t = 0.8h^2$) and the explicit ($\Delta t = 0.1h^2$), we can see that the result from the explicit is more accurate than the implicit with a comparable CPU time.

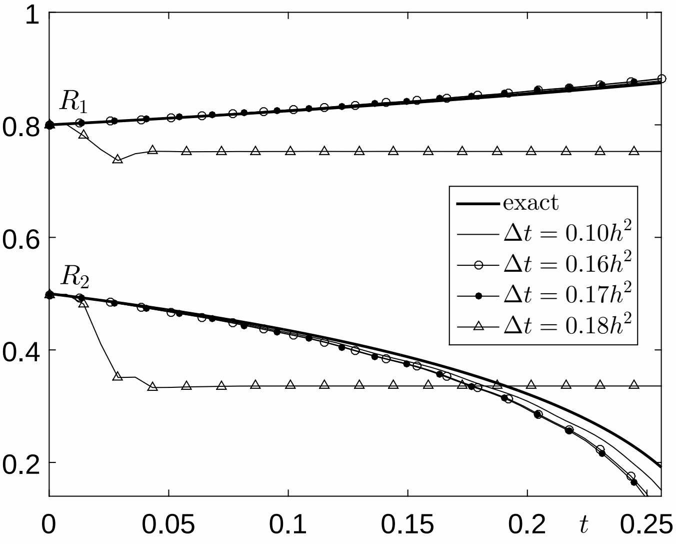


FIGURE 5.24. Stability of numerical solutions by the explicit scheme with different time step sizes.

In our algorithm, we have the time step restriction as $\Delta t \leq h^2/6 \approx 0.1667h^2$ when we solve the diffusion Eq. (5.18). Figure 5.24 shows the stability of numerical solutions by the proposed scheme with different time step sizes. To show the stability, we use four different time step sizes $\Delta t = 0.1h^2$, $0.16h^2$, $0.17h^2$, and $0.18h^2$. As shown in Fig. 5.24, we can see the stable solutions when $\Delta t \leq 0.17h^2$. One of the reasons why $\Delta t = 0.17h^2$ gives a stable solution is that the second step of the proposed algorithm damps an unstable solution and stabilize the solution.

5.2.5. Summary. We performed numerical simulations on the various three-dimensional surface shapes with the CAC equation. For efficiency of the computational cost, we employed a narrow band domain and solved the equation on that domain. On the narrow band domain boundary, we applied a quasi-Neumann boundary condition which is defined by using a closest point method. To solve the CAC equation, we used the hybrid operator splitting method which consist of three steps. First, we solved the diffusion term by using the explicit Euler method and then updated the nonlinear term by using a closed-form solution. Lastly, to conserve the total concentration, we applied the space-time dependent Lagrange multiplier. The overall scheme is fully explicit in time and does not need iterative steps, therefore it is fast. A series of numerical experiments with various initial conditions on the sphere, torus, spindle, funnel, and bunny shapes demonstrated the effectiveness and accuracy of the proposed hybrid scheme. We used uniform meshes. As future work, we plan to consider the body-fitted coordinate on non-uniform mesh and triangular surface mesh for possible improvements.

5.3. Lengyel–Epstein equation

5.3.1. Introduction. Although the pattern in living organisms is one of the classical problems of morphogenesis which explains how animals such as mammals, seashells, and marine fishes and vegetation evolve differently resulting in a consolidated and stable pattern [34], it had been generally believed that how the skin pattern in living organisms, especially mammals, is generated genetically is unclear. Nevertheless, pattern formation based on the Turing model has been one of the notable exceptions since the middle part of the 20th century while acceptance in bioinformatics of mathematical biology has been slower [71]. Most mathematical models of these patterns are based on a reaction-diffusion model, which was first proposed by Turing [102]. The reaction-diffusion model, the system of two distributed reacting and diffusing chemicals, could generate spatial patterns autonomously. Murray [76] presented the reaction-diffusion mechanism for laying down of the pre-patterns for animal marking. Young [111] performed the simulation that cells lay out on a grid with an activator and an inhibitor. Barrio et al. [5] studied numerically the spatial pattern formation with the Turing's model on the two-dimensional domain. Also, activator-inhibitor systems with non-local coupling are studied in two dimensions by Silva et al.[92]. Painter et al. [81] proposed the robustly generated patterns without parameter control which is overcome that the difficulty of applicability of classical Turing models to pattern formation is limited by the sensitivity of patterns to model parameters. Based on Turing's hypothesis, evolving morphogenetic fields in the zebra skin pattern was researched by Grávan and Lahoz-Beltra [34]. Guiu-Souto et al. [36] introduced a set of quantitative morphological measures that describe the geometrical and topological properties of Turing patterns (area, boundary length, cluster numbering, connectivity, and so on) for easily distinguish among different Turing structures.

The theoretically predicted patterns has been demonstrated under controlled experimental conditions in a Chlorite-Iodide-Malonic Acid-Starch (CIMA) reaction [20, 47, 80]. Lengyel and Epstein [54, 55, 91] derived the mathematical model from the chemical reactions and the predicted pattern by their model has a good agreement with those experimental results. The main purpose of this article is to develop a fast and computationally efficient finite difference method for the Turing pattern on curved surfaces in the three-dimensional space. We solve the resulting discrete equations on a narrow band domain. We use an interpolation using the closest point method [25, 44, 70] for the domain boundary cells. We present numerical results of the zebra patterns generated by the proposed numerical method.

5.3.2. Governing equations. We consider the following reaction-diffusion equation [54] on a smooth closed surface $\mathcal{S}$ in $\mathbb{R}^3$:

$$\begin{aligned} \frac{\partial u}{\partial t} &= D_u \Delta_{\mathcal{S}} u + f(u,v) = D_u \Delta_{\mathcal{S}} u + k_1 \left(v - \frac{uv}{1+v^2} \right), && (5.38) \\ \frac{\partial v}{\partial t} &= D_v \Delta_{\mathcal{S}} v + g(u,v) = D_v \Delta_{\mathcal{S}} v + k_2 - v - \frac{4uv}{1+v^2}, && (5.39) \end{aligned}$$

where $u(\mathbf{x},t)$ and $v(\mathbf{x},t)$ are concentrations of an inhibitor and an activator at position $\mathbf{x} \in \mathcal{S}$ and time t, respectively. Here, D_u and D_v are the diffusion coefficients, and k_1 and k_2 are positive constants related to the feed concentrations. Also, $\Delta_{\mathcal{S}}$ denotes the Laplace–Beltrami operator [69, 84, 100]. In this study, we represent a given smooth surface $\mathcal{S}$ using the signed distance function $\phi : \mathbb{R}^3 \to \mathbb{R}$. In other words, $\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^3 : \phi(\mathbf{x}) = 0\}$ with $\phi < 0$ inside of $\mathcal{S}$ and $\phi > 0$ outside of $\mathcal{S}$. The tangential gradient of u on $\mathcal{S}$ is defined as $\nabla_{\mathcal{S}} u(\mathbf{x},t) = \mathbf{P}(\mathbf{x}) \nabla u(\mathbf{x},t)$, where $\mathbf{P} = \mathbf{I} - \nabla\phi\nabla\phi^T$ is a projection operator on the tangent space. Here, $\mathbf{I}$ is the identity matrix [35]. Then, the Laplace–Beltrami operator is defined as $\Delta_{\mathcal{S}} u = \nabla_{\mathcal{S}} \cdot \nabla_{\mathcal{S}} u = (\mathbf{P}\nabla) \cdot (\mathbf{P}\nabla u) = \nabla \cdot (\mathbf{P}\nabla u)$. Next, we define a δ-neighborhood band of S as $\Omega_\delta = \{\mathbf{y} | \mathbf{x} \in S, \mathbf{y} = \mathbf{x} + \theta \mathbf{n}(\mathbf{x}) \text{ for } |\theta| < \delta\}$, where $\mathbf{n}(\mathbf{x})$ is a unit normal vector at $\mathbf{x} \in S$. Figure 5.25(a) shows the schematic illustration of the surface S, the narrow band domain Ω_δ, and the boundary $\partial\Omega_\delta$ of the narrow band domain Ω_δ. Let $\mathrm{cp}(\mathbf{x})$ be a point in the surface S which is closest to $\mathbf{x}$ [69]. Figure 5.25(b) shows the closest points $\mathrm{cp}(\mathbf{x}_1)$ and $\mathrm{cp}(\mathbf{x}_2)$ for boundary points $\mathbf{x}_1$ and $\mathbf{x}_2$ on $\partial\Omega_\delta$. Then, the boundary condition is defined as

$$u(\mathbf{x}, t) = u(\mathrm{cp}(\mathbf{x}), t) \text{ and } v(\mathbf{x}, t) = v(\mathrm{cp}(\mathbf{x}), t) \text{ on } \partial\Omega_\delta. \tag{5.40}$$

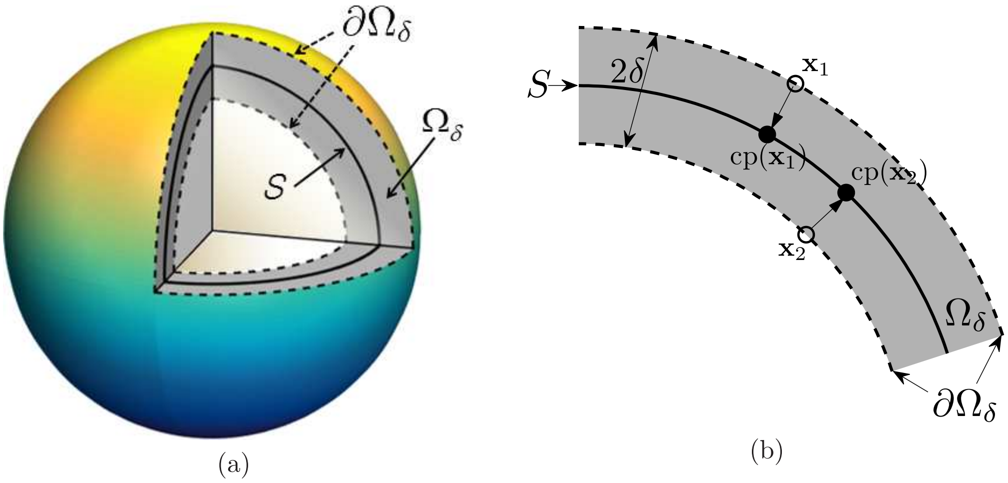


FIGURE 5.25. (a) Schematic illustration of the surface S, the narrow band domain Ω_δ with thickness 2δ, and its boundary $\partial\Omega_\delta$. (b) Closest points $\mathrm{cp}(\mathbf{x}_1)$ and $\mathrm{cp}(\mathbf{x}_2)$ for points $\mathbf{x}_1$, $\mathbf{x}_2 \in \partial\Omega_\delta$.

If δ is small enough, then the temporal evolution of Eqs. (5.38) and (5.39) with the boundary condition (5.40) results in u and v which are constant in the direction normal to the surface. Therefore, we can replace the surface Laplacian operator $\Delta_{\mathcal{S}}$ by the standard

Laplacian operator Δ in the narrow band domain Ω_δ, i.e.,

$$\frac{\partial u}{\partial t} = D_u \Delta u + k_1 \left(v - \frac{uv}{1+v^2} \right), \tag{5.41}$$

$$\frac{\partial v}{\partial t} = D_v \Delta v + k_2 - v - \frac{4uv}{1+v^2}. \tag{5.42}$$

5.3.3. Discretization. We present a numerical algorithm for the Lengyel–Epstein model on the narrow band domain, Ω_δ. We discretize the reaction-diffusion equation in a three-dimensional domain $\Omega = (a, b) \times (c, d) \times (e, f)$ embedding Ω_δ. Let N_x, N_y, and N_z be positive integers, $h = (b-a)/N_x = (d-c)/N_y = (f-e)/N_z$ be the uniform mesh size, and $\Omega^h = \{\mathbf{x}_{ijk} = (x_i, y_j, z_k) = (a+hi, c+hj, e+hk) | 0 \le i \le N_x, 0 \le j \le N_y, 0 \le k \le N_z\}$ be the discrete domain. Let u_{ijk}^n and v_{ijk}^n be approximations of $u(x_i, y_j, z_k, n\Delta t)$ and $v(x_i, y_j, z_k, n\Delta t)$, where $\Delta t = T/N_t$ is the time step, T is the final time, and N_t is the total number of time steps.

We adopt the numerical scheme used in the fast and accurate numerical method for motion by mean curvature of curves on a surface in three-dimensional space using the Allen–Cahn equation [25]. For a given smooth surface S, we define $\phi : \mathbb{R}^3 \to \mathbb{R}$ as the signed distance function to S so that $S = \{\mathbf{x} \in \mathbb{R}^3 | \phi(\mathbf{x}) = 0\}$ with $\phi < 0$ inside S and $\phi > 0$ outside S. Let $\Omega_\delta^h = \{\mathbf{x}_{ijk} || \phi_{ijk}| < \delta\}$ be the discrete narrow band domain. We take $\delta \ge \sqrt{3}h$ since Ω_δ^h must contain the interpolation stencil for the closest points of the domain boundary points. We define a discrete L^2-norm on Ω_δ^h as

$$\|\phi\|_{L^2(\Omega_\delta^h)} = \sqrt{\frac{1}{\#\Omega_\delta^h} \sum_{\mathbf{x}_{ijk} \in \Omega_\delta^h} \phi_{ijk}^2},$$

where $\#\Omega_\delta^h$ is the cardinality of the set Ω_δ^h. Let us define the domain boundary points as

$$\partial\Omega_\delta^h = \{(x_i, y_j, z_k) \big| I_{ijk} |\nabla_h I_{ijk}| \neq 0\},$$

where

$$\nabla_h I_{ijk} = \frac{I_{i+1,jk} - I_{i-1,jk}, I_{i,j+1,k} - I_{i,j-1,k}, I_{ij,k+1} - I_{ij,k-1}}{2h}.$$

Here, $I_{ijk} = 0$ if $(x_i, y_j, z_k) \in \Omega_\delta^h$; otherwise $I_{ijk} = 1$. We consider the discretization of the reaction-diffusion system (5.41) and (5.42) using explicit scheme,

$$\frac{u_{ijk}^{n+1} - u_{ijk}^n}{\Delta t} = D_u \Delta_h u_{ijk}^n + k_1 \left(v_{ijk}^n - \frac{u_{ijk}^n v_{ijk}^n}{1 + (v_{ijk}^n)^2} \right), \tag{5.43}$$

$$\frac{v_{ijk}^{n+1} - v_{ijk}^n}{\Delta t} = D_v \Delta_h v_{ijk}^n + k_2 - v_{ijk}^n - \frac{4u_{ijk}^n v_{ijk}^n}{1 + (v_{ijk}^n)^2}, \tag{5.44}$$

with the boundary condition on $\partial\Omega_\delta^h$: $u_{ijk}^n = u^n(\text{cp}(\mathbf{x}_{ijk}))$ and $v_{ijk}^n = v^n(\text{cp}(\mathbf{x}_{ijk}))$. Here, the descretization of second spatial derivative is given by

$$\Delta_h u_{ijk} = \frac{1}{h^2}(u_{i+1,jk} + u_{i-1,jk} + u_{i,j+1,k} + u_{i,j-1,k} + u_{ij,k+1} + u_{ij,k-1} - 6u_{ijk}).$$

The other one $\Delta_h v_{ijk}$ is similarly defined. We define the numerical closest point of the boundary point $\mathbf{x}_{ijk}$ to the surface S as

$$\text{cp}(\mathbf{x}_{ijk}) = \mathbf{x}_{ijk} - \frac{\nabla_h |\phi_{ijk}|}{|\nabla_h |\phi_{ijk}||} |\phi_{ijk}|. \tag{5.45}$$

If the closest point $\text{cp}(\mathbf{x}_{ijk})$ is not lying on a given computational grid, we obtain $u^n(\text{cp}(\mathbf{x}_{ijk}))$ and $v^n(\text{cp}(\mathbf{x}_{ijk}))$ by using the trilinear interpolation. For the fast computation, we tabulated the interpolation stencil and three fractions for each boundary point before starting time iterations.

5.3.4. Numerical experiments.

5.3.4.1. *Parameter study on pattern formation on spherical surface.* Next, we investigate the effect of parameters D_v and k_1 on the pattern dynamics on the surface of a sphere with radius 4.5. The spherical surface is represented by the zero level set of the signed distance function

$$\phi(x,y,z) = \sqrt{(x-5)^2+(y-5)^2+(z-5)^2} - 4.5.$$

For the numerical simulation, we use the same parameter values for k_1, k_2, D_u, and D_v as in [79]. The initial conditions are taken to be

$$\begin{aligned} u(x,y,z,0) &= 1+0.04k_2^2+0.1\text{rand}(x,y,z), & (5.46)\\ v(x,y,z,0) &= 0.2k_2+0.1\text{rand}(x,y,z), & (5.47)\end{aligned}$$

where $\text{rand}(x,y,z)$ is a random number between -1 and 1. The parameters used are $D_u = 1$, $k_2 = 11$, $h = 0.1$, $\delta = 1.1\sqrt{3}h$, $\Delta t = 0.1h^2$, and $T = 1000$ on the computational domain $\Omega = [0,10]^3$.

Figure 5.26 represents numerical solutions of activator v by Eqs. (5.43) and (5.44) with respect to D_v and k_1. Depending on the parameter sets (D_v, k_1), we obtain the numerical results such as spots, stripes, and mixtures of spots and stripes.

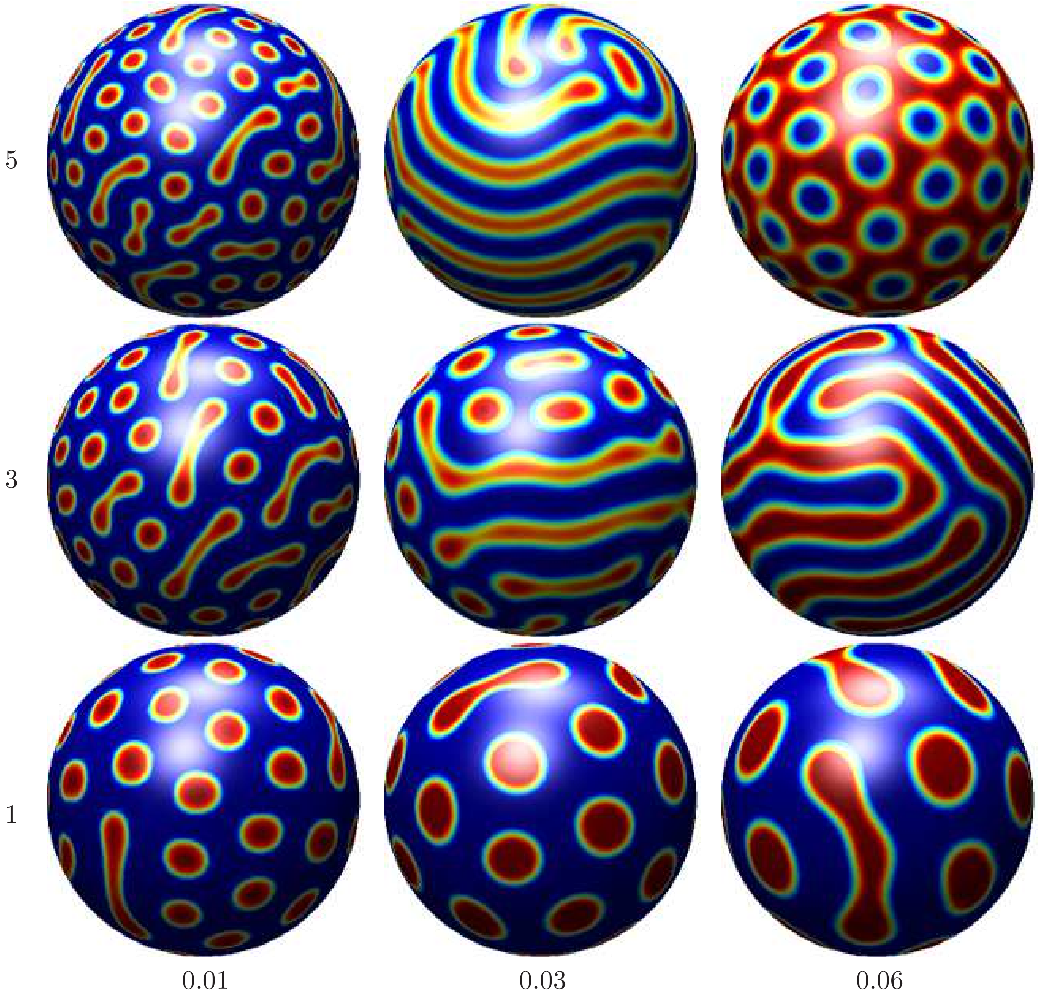


FIGURE 5.26. Overview of pattern formation on surface of sphere corresponding to concentration of activator at different values of D_v (horizontal axis) and k_1(vertical axis).

5.3.4.2. *Parameter study on pattern formation on torus surface.* Similar to the previous section, we study the pattern changes according to the parameters(D_v and k_1) on the torus with major radius 5.5 and minor radius 3.5. The torus surface is represented by the zero level set of the signed distance function

$$\phi(x, y, z) = \sqrt{(\sqrt{x^2 + y^2} - 5.5)^2 + z^2} - 3.5$$

For the numerical simulation, we use the same parameter values and initial conditions in previous section. The parameters used are $D_u = 1$, $k_2 = 11$, $h = 0.1$, $\delta = 1.1\sqrt{3}h$, $\Delta t = 0.1h^2$, and $T = 1000$ on the computational domain $\Omega = [0, 10]^3$.

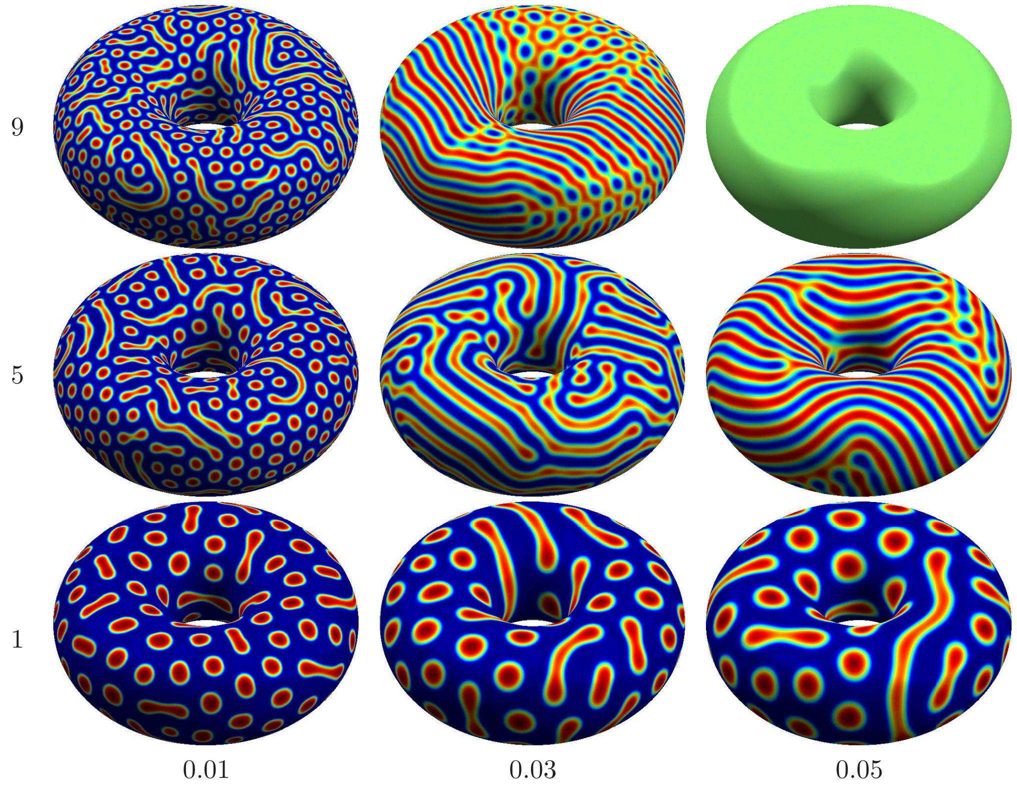


FIGURE 5.27. Overview of pattern formation on surface of torus corresponding to concentration of activator at different values of D_v (horizontal axis) and k_1(vertical axis).

Figure 5.27 represents numerical solutions of activator v. We obtain the similar numerical results in the previous section such as spots, stripes, and mixtures of spots and stripes from each cases (D_v, k_1). In the case of $k1 = 9$ and $D_v = 0.05$, Turing instability does not occur.

5.3.4.3. *Stripe patterns on zebra surface.* Depending on varying parameters, different kinds of stripe patterns on a given zebra surface are generated. To investigate this, we take the initial conditions (5.46) and (5.47). The other parameters used are as follows: $D_u = 1$, $k_2 = 11$, $h = 10/64$, $\delta = 1.1\sqrt{3}h$, $\Delta t = 0.1h^2$, $T = 2441.4$, and $\Omega = [0, 84h] \times [0, 276h] \times [0, 212h]$. For k_1 and D_v, we use the three different values as $(k_1, D_v) = (11, 0.02)$, $(k_1, D_v) = (5, 0.04)$, and $(k_1, D_v) = (3, 0.06)$. Here, for the chosen parameters, we may refer to [36].

Figure 5.28 represents the temporal evolution of morphological patterns of activator v on the zebra surface with different values of (k_1, D_v). The computational times are listed below each row. In this figure, we plot the pattern with a local grayscale corresponds to white and black with maximum and minimum concentration of activator v. As k_2 decreases and D_v increases, the thickness of the stripes are larger.

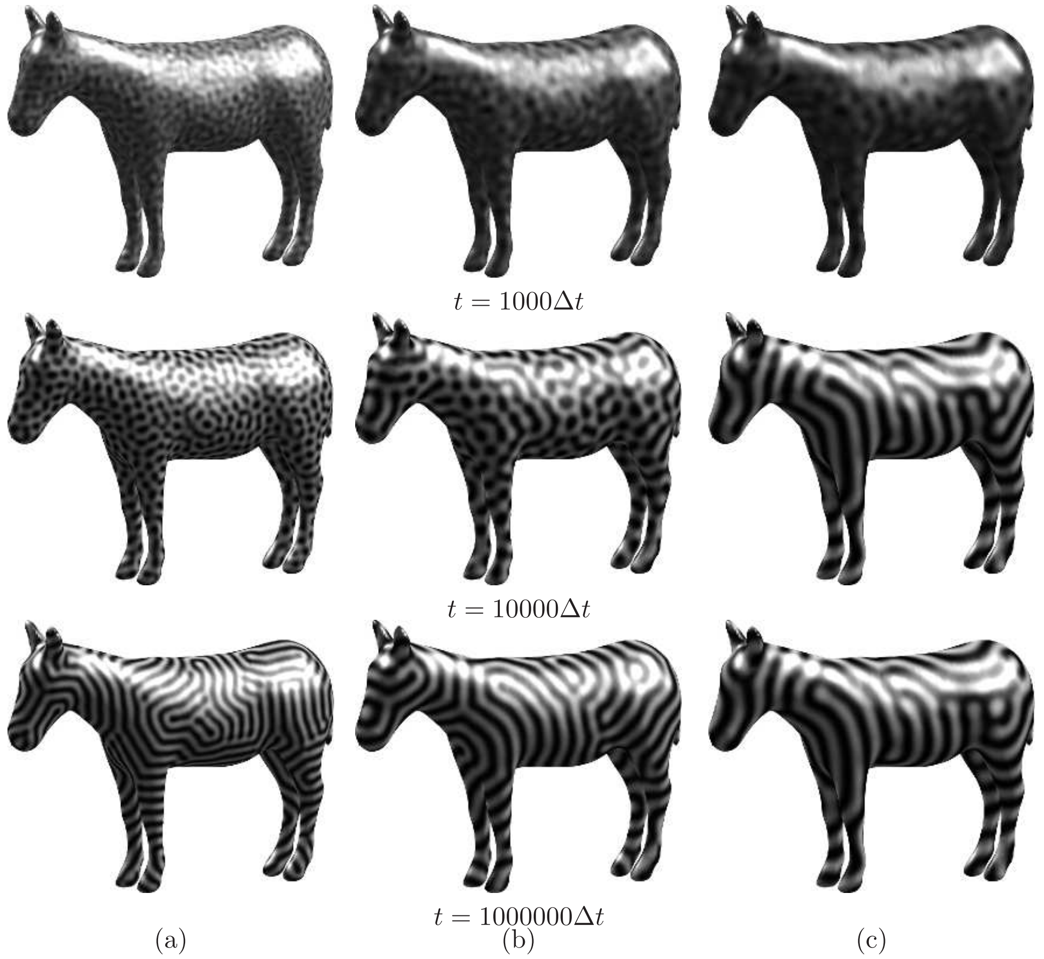


FIGURE 5.28. Pattern formation process of the Lengyel–Epstein model on a zebra surface with $k_1 = 11$ and $D_u = 1$ at $t = 1000\Delta t,\ 10000\Delta t,\ 100000\Delta t$, and $100000\Delta t$. Here, the other parameters are used (a) $k_2 = 11, D_v = 0.02$, (b) $k_2 = 5, D_v = 0.04$, and (c) $k_2 = 3, D_v = 0.06$, respectively.

5.3.5. Summary. We numerically studied the zebra skin pattern formation on the surface of a zebra model in three-dimensional space using a two-component activator-inhibitor system of reaction-diffusion equations. We discretized the governing equations using a finite difference method and solved the resulting system on a discrete narrow band domain containing the zebra skin. For the domain boundary cells, we used an interpolation using the closest point method. We presented numerical results of the zebra patterns generated by the model. Depending on parameter sets, we had different spatial pattern formations such as spots, stripes, and mixture of spots and stripes. Also, we can see this results on zebra surface, which is more complex than sphere, as well as sphere. From the numerical tests, we knew that the condition that distinguish between stripe or spots is determined by D_v in reaction-diffusion systems and k_1 plays a role in decision of the wavelength of pattern. Especially, we obtained the similar numerical pattern on given zebra surface to the real one.

Chapter 6

Conclusions

We presented that solving a partial differential equations on the non-flat surfaces. Our proposed detail strategies and implementations are followings: surface reconstruction algorithm (using modified Allen–Cahn equation, cloud points, and slice data), construct computational domain and boundary condition (narrow band domain, and closest points method), and then solving the Allen–Cahn equation, conservative Allen–Cahn equation, and Lengyel–Epstein equations on the various surfaces.

From the surface reconstruction algorithm, we obtained the three-dimensional surface data which original data are two-dimensional data such as CT, MRI, and cloud points. And by using narrow band domain and quasi-Neumann boundary condition which applied closest points method, we can use the standard Laplacian operator instead of Laplace–Beltrami operator. Overall numerical simulations, we used operator splitting method. Some tests were used multigrid method, and some simulations were applied an explicit method for fast solution. Numerical results demonstrated that the proposed methods were fast and accurate.

The future work will focus on solving the fluid-related equations and pattern formation on the surfaces, such as Navier–Stokes equation, shallow water equations (Saint–Venant equations), reaction-diffusion equations and so on. Through this, we expect that researches will be applicable on fine dust, convection current, and weather on the earth surface. Also we can study pattern formation in nature such as animals skin, sand dunes and giant's causeway.

Bibliography

[1] H. Abels, H. Garcke, and G. Grun, Thermodynamically consistent, frame indifferent diffuse interface models for incompressible two-phase flows with different densities, Mathematical Models and Methods in Applied Sciences 22 (2012) 1150013.

[2] S.M. Allen and J.W. Cahn, A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening, Acta Metallurgica 27 (1979) 1085–1095.

[3] S. Angenent, S. Haker, A. Tannenbaum, and R. Kikinis, On the Laplace–Beltrami operator and brain surface flattening, IEEE Transactions on Medical Imaging 18 (1999) 700–711.

[4] A.C. Aristotelous, O. Karakashian, and S.M. Wise, A mixed discontinuous Galerkin, convex splitting scheme for a modified Cahn–Hilliard equation and an efficient nonlinear multigrid solver, Discrete and Continuous Dynamical Systems - Series B 18 (2013) 2211–2238.

[5] R.A. Barrio, C. Varea, J.L. Aragón, and P.K. Maini, A two-dimensional numerical study of spatial pattern formation in interacting Turing systems, Bulletin of Mathematical Biology 61 (1999) 483–505.

[6] P.W. Bates and J. Jin, Global dynamics of boundary droplets, Discrete and Continuous Dynamical Systems 34 (2013) 1–17.

[7] M. Beneš, S. Yazaki, and M. Kimura, Computational studies of non-local anisotropic Allen–Cahn equation, Mathematica Bohemica 136 (2011) 429–437.

[8] A.L. Bertozzi, S. Esedoglu, and A. Gillette, Inpainting of binary images using the Cahn–Hilliard equation, IEEE Transactions on Image Processing 16 (2007) 285–291.

[9] A.L. Bertozzi, S. Esedoglu, and A. Gillette, Analysis of a two-scale Cahn–Hilliard model for image inpainting, Multiscale Modeling and Simulation 3 (2007) 913–936.

[10] R. Bhargava and H.S. Takhar, Numerical study of heat transfer characteristics of the micropolar boundary layer near a stagnation point on a moving wall, International journal of engineering science 38 (2000) 383–394.

[11] M. Brassel, E. Bretin, A modified phase field approximation for mean curvature flow with conservation of the volume, Mathematical Methods in the Applied Sciences 34 (2011) 1157–1180.

[12] M. Bertalmio, L.T. Cheng, S. Osher, and G. Sapiro, Variational problems and partial differential equations on implicit surfaces, Journal of Computational Physics 174 (2001) 759–780.

[13] M. Bene*š*, V. Chalupeck*ý*, and K. Mikula, Geometrical image segmentation by the Allen–Cahn equation, Applied Numerical Mathematics 51 (2004) 187–205.

[14] W.L. Briggs and F. Steve, A Multigrid Tutorial, SIAM, Vol. 72. Philadelphia, 1987.

[15] L. Bronsard and B. Stoth, Volume-preserving mean curvature flow as a limit of a nonlocal Ginzburg–Landau equation, SIAM Journal on Mathematical Analysis 28 (1997) 769–807.

[16] R.L. Burden and J.D. Faires, Numerical Analysis, Cengage Learning, Belmont, 2005.

[17] J.W. Cahn and J.E. Hilliard, Free energy of a nonuniform system. I. interfacial free energy, The Journal of chemical physics28 (1958) 258–267.

[18] Y. Cao and R. Erban, Stochastic Turing patterns: Analysis of compartment-based approaches, Bulletin of Mathematical Biology 76 (2014) 3051–3069.

[19] V. Caselles, F. Catte, T. Coll, and F. Dibos, A geometric model for active contours in image processing, Numerische mathematik 66 (1993) 1–31.

[20] V. Castets, E. Dulos, and P.D. Kepper, Experimental evidence of a sustained standing Turing-type nonequilibrium chemical pattern, Physical Review Letters 64 (1990) 2953–2956.

[21] T.F. Chan and L.A. Vese, Active contours without edges, IEEE Transactions on Image Processing 10 (2001) 266–277.

[22] L.Q. Chen and J. Shen, Applications of semi-implicit Fourier-spectral method to phase field equations, Computer Physics Communications 108 (1998) 147–158.

[23] J. Cheng and Y. Liu, 3-D reconstruction of medical image using wavelet transform and snake model, Journal of Multimedia 4 (2009) 427–434.

[24] M. Cheng and J.A. Warren, An efficient algorithm for solving the phase field crystal model, Journal of Computational Physics 227 (2008) 6241–6248.

[25] Y. Choi, D. Jeong, S. Lee, M. Yoo, and J. Kim, Motion by mean curvature of curves on surfaces using the Allen–Cahn equation, International Journal of Engineering Science 97 (2015) 126–132.

[26] J.W. Choi, H.G. Lee, D. Jeong, and J. Kim, An unconditionally gradient stable numerical method for solving the Allen–Cahn equation, Physica A: Statistical Mechanics and its Applications 388 (2009) 1791–1803.

[27] J. Dornheim, D.J. Lehmann, L. Dornheim, B. Preim, and G. Strauß, Reconstruction of blood vessels from neck CT datasets using stable 3D mass-spring models, VCBM (2008) 77–82.

[28] A.B. Ekoule, F.C. Peyrin, and C.L. Odet, A triangulation algorithm from arbitrary shaped multiple planar contours,ACM Transactions on Graphics 10 (1991) 182–199.

[29] B. Ermentrout, Stripes or spots? Nonlinear effects in bifurcation of reaction-diffusion equations on the square, Proceedings of the Royal Society of London A 434 (1991) 413–417.

[30] D.J. Eyre, Unconditionally gradient stable time marching the Cahn–Hilliard equation, In: Computational and mathematical models of microstructural evolution (San Francisco, CA, 1998). Materials Research Society Symposia Proceedings 529 39–46.

[31] Z. Feng, J. Yin, and J. Zhou, Inpainting algorithm for jacquared image based on phase-field model, IEEE Intelligent Systems and Knowledge Engineering ISKE (2008) 1203–1207.

[32] H. Garcke, B. Nestler, B. Stinner, and F. Wendler, Allen–Cahn systems with volume constraints, Mathematical Models and Methods in Applied Sciences 18 (2008) 1347–1381.

[33] S.F. Gibson, Constrained elastic surface nets: Generating smooth surfaces from binary segmented data, Medical Image Computing and Computer-Assisted Intervention Springer Berlin Heidelberg. (1998) 888–898.

[34] C.P. Gravan and R. Lahoz-Beltra, Evolving morphogenetic fields in the zebra skin pattern based on Turing's morphogen hypothesis, International Journal of Applied Mathematics and Computer Science 14 (2004) 351–362.

[35] J.B. Greer, An improvement of a recent Eulerian method for solving PDEs on general geometries, Journal of Scientific Computing 29 (2006) 321–352.

[36] J. Guiu-Souto, J. Carballido-Landeira, and A.P. Munuzuri, Characterizing topological transitions in a Turing-pattern-forming reaction-diffusion system, Physical Review E 85 (2012) 056205.

[37] J.F. Guo, Y.L. Cai, and Y.P. Wang, Morphology-based interpolation for 3D medical image reconstruction, Computerized Medical Imaging and Graphics 19 (1995) 267–279.

[38] A. Gutierrez, M.J. Jimenez, D. Monaghan, and N.E. ÓConnor, Topological evaluation of volume reconstructions by voxel carving, Computer Vision and Image Understanding 121 (2014) 27–35.

[39] D. Haile and Z. Xie, Long-time behavior and Turing instability induced by cross-diffusion in a three species food chain model with a Holling type-II functional response, Mathematical Biosciences 267 (2015) 134–148

[40] S. Haker, S. Angenent, A.R. Tannenbaum, and R. Kikinis, Conformal 3D visualization for virtual colonoscopy, Medical Imaging 2000: Physiology and Function from Multidimensional Images (2000) 154–164.

[41] M. Heida, J. Malek, and K.R. Rajagopal, On the development and generalizations of Cahn–Hilliard equations within a thermodynamic framework, Zeitschrift fr angewandte Mathematik und Physik 63 (2012) 145–169.

[42] M. Heida, J. Malek, and K.R. Rajagopal, On the development and generalizations of Allen–Cahn and Stefan equations within a thermodynamic framework Zeitschrift fr angewandte Mathematik und Physik 63 (2012) 759–776.

[43] G. Huisken, Flow by mean curvature of convex surfaces into spheres, Journal of Differential Geometry 20 (1984) 237–266.

[44] D. Jeong and J. Kim, Microphase separation patterns in diblock copolymers on curved surfaces using a nonlocal Cahn–Hilliard quation, The European Physical Journal E 38 (2015) 1–7.

[45] M.W. Jones and M. Chen, A new approach to the construction of surfaces from contour data, Computer Graphics Forum 13 (1994) 75–84.

[46] D.A. Kay and A. Tomasi, Color image segmentation by the vector-valued Allen–Cahn phase-field model: a multigrid solution, IEEE Transactions on Image Processing 18 (2009) 2330–2339.

[47] P. Kepper, V. Castets, E. Dulos, and J. Boissonade, Turing-type chemical patterns in the chlorite-iodide-malonic acid reaction, Physica D 49 (1991) 161–169.

[48] J. Kim, Phase-field models for multi-component fluid flows, Communications in Computational Physics 12 (2012) 613–661.

[49] J. Kim, S. Lee, and Y. Choi, A conservative Allen–Cahn equation with a space-time dependent Lagrange multiplier, International Journal of Engineering Science 84 (2014) 11–17.

[50] M. Kowalczyk, On the existence and Morse index of solutions to the Allen–Cahn equation in two dimensions, Annali di Matematica Pura ed Applicata 184 (2005) 17–52.

[51] D. Lee, J.Y. Huh, D. Jeong, J. Shin, A. Yun, and J. Kim, Physical, mathematical, and numerical derivations of the Cahn–Hilliard equation, Computational Materials Science 81 (2014) 216–225.

[52] H.G. Lee and J. Kim, A simple and efficient finite difference method for the phase-field crystal equation on curved surfaces, Computer Methods in Applied Mechanics and Engineering 307 (2016) 32–43.

[53] D.S. Lee and J.S. Kim, Mean curvature flow by the Allen-Cahn equation, European Journal of Applied Mathematics 26 (2015) 535–559.

[54] I. Lengyel and I.R. Epstein, Modeling of Turing structure in the Chlorite–Iodide–Malonic acid–Starch reaction system, Science 251 (1991) 650–652.

[55] I. Lengyel and I.R. Epstein, A chemical approach to designing Turing patterns in reaction-diffusion system, Proceedings of the National Academy of Sciences 89 (1992) 3977–3979.

[56] M. Lhuillier and S. Yu, Manifold surface reconstruction of an environment from sparse Structure-from-Motion data, Computer Vision and Image Understanding 117 (2013) 1628–1644.

[57] Y. Li, H.G. Lee, D. Jeong, and J. Kim, An unconditionally stable hybrid numerical method for solving the Allen–Cahn equation, Computers & Mathematics with Applications 60 (2010) 1591–1606.

[58] Y. Li, D. Jeong, J. Choi, S. Lee, and J. Kim Fast local image inpainting based on the Allen–Cahn model, Digital Signal Processing 37 (2015) 65–74.

[59] Y. Li, D. Jeong, J. Shin, and J. Kim, A conservative numerical method for the Cahn–Hilliard equation with Dirichlet boundary conditions in complex domains, Computers & Mathematics with Applications 65 (2013) 102–115.

[60] Y. Li and J. Kim, A fast and accurate numerical method for medical image segmentation, The Journal of the Korean Society for Industrial and Applied Mathematics 14 (2010) 201–210.

[61] Y. Li and J. Kim, Multiphase image segmentation using a phase-field model, Computers & Mathematics with Applications 62 (2011) 737–745.

[62] Y. Li and J. Kim, An unconditionally stable numerical method for bimodal image segmentation, Applied Mathematics and Computation 219 (2012) 3083–3090.

[63] Y. Li and J. Kim, An unconditionally stable hybrid method for images segmentation, Applied Numerical Mathematics 82 (2014) 32–43.

[64] Y. Li, D. Lee, C. Lee, J. Lee, S. Lee, J. Kim, S. Ahn, and J. Kim, Surface embedding narrow volume reconstruction from unorganized points, Computer Vision and Image Understanding 121 (2014) 100–107.

[65] W.C. Lin, S.Y. Chen, and C.T. Chen, A new surface interpolation technique for reconstructing 3D objects from serial cross-sections, Computer Vision Graphics and Image Processing 48 (1989) 124–143.

[66] J. Lowengrub and L. Truskinovsky, Quasi-incompressible Cahn–Hilliard fluids and topological transitions, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 454 (1998) 2617–2654.

[67] M.J. Lyons and L.G. Harrison, Stripe selection: An intrinsic property of some pattern-forming models with nonlinear dynamics, Developmental Dynamics 195 (1992) 201–215.

[68] J. Ma, Y. Jiang, and K. Xiang, Numerical simulation of blowup in nonlocal reaction-diffusion equations using a moving mesh method, Journal of Computational and Applied Mathematics 230 (2009) 8–21.

[69] C.B. Macdonald, J. Brandman, and S.J. Ruuth, Solving eigenvalue problems on curved surfaces using the Closest Point Method, Journal of Computational Physics 230 (2011) 7944–7956.

[70] C.B. Macdonald and J.R. Steven, Level set equations on surfaces via the closest point method, Journal of Scientific Computing 35 (2008) 219–240.

[71] M.C. Mackey and P.K. Maini, What has mathematics done for biology?, Bulletin of Mathematical Biology 77 (2015) 735–738.

[72] D. Marenduzzo and E. Orlandini, Phase separation dynamics on curved surfaces, Soft Matter 9 (2013) 1178–1187.

[73] MathWorks, Inc. *MATLAB: the language of technical computing*. Natick, MA.: http://www.mathworks.com/, The MathWorks. 2015.

[74] F. Memoli, G. Sapiro, and P. Thompson, Implicit brain imaging, NeuroImage 23 (2004) S179–S188.

[75] B. Merriman and S.J. Ruuth, Diffusion generated motion of curves on surfaces, Journal of Computational Physics 225 (2007) 2267–2282.

[76] J.D. Murray, A pre-pattern formation mechanism for animal coat markings, Journal of Theoretical Biology 88 (1981) 161–199.

[77] T.G. Myers, J.P.F. Charpin, and S.J. Chapman, The flow and solidification of a thin fluid film on an arbitrary three-dimensional surface, Physics of Fluids 14 (2002) 2788–2803.

[78] E. Orlandini, D. Marenduzzo, and A.B. Goryachev, Domain formation on curved membranes: phase separation or Turing patterns?, Soft Matter 9 (2013) 9311–9318.

[79] H.G. Othmer, K. Painter, D. Umulis, and C. Xue, The intersection of theory and application in Elucidating pattern formation in developmental biology, Mathematical Modelling of Natural Phenomena 4 (2009) 3–82.

[80] Q. Ouyang and H.L. Swinney, Transition from a uniform state to hexagonal and striped turing patterns, Nature 352 (1991) 610–612.

[81] K.J. Painter, P.K. Maini, and H.G. Othmer, Stripe formation in juvenile Pomacanthus explained by a generalized Turing mechanism with chemotaxis, Proceedings of the National

Academy of Sciences 96 (1999) 5549–5554.

[82] R.D. Parshad, N. Kumari, A.R. Kasimov, and H.A. Abderrahmane, Turing patterns and long-time behavior in a three-species food-chain model, Mathematical Biosciences 254 (2014) 83–102.

[83] G. Paul, J. Cardinale, and I.F. Sbalzarini, Coupling image restoration and segmentation: a generalized linear model/Bregman perspective, International Journal of Computer Vision 104 (2013) 69–93.

[84] R.G. Plaza, F. Sanchez-Garduno, P. Padilla, R.A. Barrio, and P.K. Maini, The effect of growth and curvature on pattern formation, Journal of Dynamics and Differential Equations 16 (2004) 1093–1121.

[85] S.P. Raya and J.K. Udupa, Shape-based interpolation of multidimensional objects, IEEE Transactions on Medical Imaging 9 (1990) 32–42.

[86] X. Ren and J. Wei, On a phase field problem driven by interface area and interface curvature, European Journal of Applied Mathematics 20 (2009) 531–556.

[87] J. Rubinstein and P. Sternberg, Nonlocal reaction-diffusion equations and nucleation, IMA Journal of Applied Mathematics 48 (1992) 249–264.

[88] S.J. Ruuth and B. Merriman, A simple embedding method for solving partial differential equations on surfaces, Journal of Computational Physics 227 (2008) 1943–1961.

[89] O. Schonborn, Phase-ordering kinetics on curved surfaces, Physica A 239 (1997) 412–419.

[90] J. Shen and X. Yang, A phase-field model and its numerical approximation for two-phase incompressible flows with different densities and viscosities, SIAM Journal of Scientific Computing 32 (2010) 1159–1179.

[91] H. Shoji and T. Ohta, Computer simulations of three-dimensional Turing patterns in the Lengyel–Epstein model, Physical Review E 91 (2015) 032913.

[92] F.A.S. Silva, R.L. Viana, and S.R. Lopes, Pattern formation and Turing instability in an activatorinhibitor system with power-law coupling, Physica A: Statistical Mechanics and its Applications 419 (2015) 487–497.

[93] D.Stafford, M.J. Ward, and B. Wetton, The dynamics of drops and attached interfaces for the constrained Allen–Cahn equation, European Journal of Applied Mathematics 12 (2001) 1–24.

[94] The Stanford volume data archive, Copyright ©2000, 2001 Marc Levoy, http://www-graphics.stanford.edu/data/voldata/

[95] A. Stuart and A.R. Humphries, Dynamical Systems And Numerical Analysis, Cambridge University Press, Vol. 2. Cambridge, 1998.

[96] M. Sun, X. Yan, and R.J. Sclabassi, Solving partial differential equations in real-time using artificial neural network signal processing as an alternative to finite-element analysis, IEEE Neural Networks and Signal Processing (2003).

[97] M. Sussman, P. Smereka, and S. Osher, A level set approach for computing solutions to incompressible two-phase flow, Journal of Computational Physics 114 (1994) 146–159.

[98] P. Tang, F. Qiu, H. Zhang, and Y. Yang, Phase separation patterns for diblock copolymers on spherical surfaces: A finite volume method, Physical Review E 72 (2005) 016710.

[99] G. Terragni, URL: http://open3dmodel.com/download/zebra-3d-model_3220.html 2014.

[100] G. Toole and M.K. Hurdal, Turing models of cortical folding on exponentially and logistically growing domains, Computers & Mathematics with Applications 66 (2013) 1627–1642.

[101] U. Trottenberg and C. Oosterlee, A. Schuller, Multigrid, Academic Press, London, 2001.

[102] A.M. Turing, The chemical basis of morphogenesis, Philosophical Transactions of the Royal Society B 237 (1952) 37–72.

[103] G. Turk, Generating textures on arbitrary surfaces using reaction-diffusion, ACM SIGGRAPH Computer Graphics 25 (1991) 289–298.

[104] M.J. Ward, Metastable bubble solutions for the Allen–Cahn equation with mass conservation, SIAM Journal on Applied Mathematics 56 (1996) 1247–1279.

[105] X. Wu, G.J. Zwieten, and K.G. Zee, Stabilized second-order convex splitting schemes for Cahn–Hilliard models with application to diffuse-interface tumor-growth models. International Journal for Numerical Methods in Biomedical Engineering 30 (2014) 180–203.

[106] Y. Xia, Y. Xu, and C.W. Shu, Application of the Local Discontinuous Galerkin Method for the Allen–Cahn/Cahn–Hilliard System, Communications in Computational Physics 5 (2009) 821–835.

[107] J.J. Xu and H.K. Zhao, An Eulerian formulation for solving partial differential equations along a moving interface, Journal of Scientific Computing 19 (2003) 573–594.

[108] X. Yang, J.J. Feng, C. Liu, and J. Shen, Numerical simulations of jet pinching-off and drop formation using an energetic variational phase-field method, Journal of Computational Physics 218 (2006) 417–428.

[109] S.D. Yang, H.G. Lee, and J. Kim, A phase-field approach for minimizing the area of triply periodic surfaces with volume constraint, Computer Physics Communications 181 (2010) 1037–1046.

[110] D.-J. Yoo, Three-dimensional surface reconstruction of human bone using a B-spline based interpolation approach, Computer-Aided Design 43 (2011) 934–947.

[111] D.A. Young, A local activator-inhibitor model of vertebrate skin patterns, Mathematical Biosciences 72 (1984) 51–58.

[112] P. Yue, C. Zhou, and J.J. Feng, Spontaneous shrinkage of drops and mass conservation in phase-field simulations, Journal of Computational Physics 223 (2007) 1–9.

[113] S. Zhao and G.W. Wei, A unified discontinuous Galerkin framework for time integration, Mathematical methods in the applied sciences 37 (2013) 1042–1071.

[114] Z. Zhang and H. Tang, An adaptive phase field method for the mixture of two incompressible fluids, Computers & fluids 36 (2007) 1307–1318.