FRONT-TRACKING FINITE ELEMENT METHODS FOR A VOID ELECTRO-STRESS MIGRATION PROBLEM

A thesis presented for the degree of
Doctor of Philosophy of Imperial College London
and the Diploma of Imperial College
by

ANDREA SACCONI
Department of Mathematics
Imperial College London
South Kensington Campus – SW7 2AZ, London

DECLARATION OF ORIGINALITY

I certify that this thesis, and the research to which it refers, are the product of my own work, and that any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline.

London, December 2015

COPYRIGHT DECLARATION

ACKNOWLEDGEMENTS

Tunc demum sensit cordis requiesse tumultum
spemque aluit, prorae similem maria alta secanti,
tempora mox vitae fore prosperiora futurae. [...]
et genibus flexis sponte haec sibi fervidus optat
non tam oris sonitu quam ardente cupidine cordis:
«O socium, quaeso, tibi me coniunge tuaeque
da mihi participem vitam traducere mortis».

IOANNES BAPTISTA PIGATUS c.r.s.,
Sacerdos moriens, 87–89, 124–127

As I did for my BSc. and MSc. theses, I would like to start my acknowledgements with some dactylic hexameters of Fr. Giovanni Battista Pigato (1910-1976), Catholic priest of the Order of the Somaschan Fathers, renowned Latinist, chaplain of the *Alpini* during World War II, and distinguished teacher of Latin and Greek for hundreds of high-school students. These verses too reflect perfectly the journey I embarked upon in 2011, when I started my PhD here at Imperial.

Yeah, it is almost four years since I left Como. Much water has flowed under the bridge since then! I have improved my English, I have travelled to quite a few places, I have met so many people! I would like to thank everyone who has contributed to this adventure, by both sharing with me the good moments and supporting me during troubled periods. I am sure that my list will not be exhaustive, so I ask forgiveness in advance if, inadvertently, I have forgotten someone.

First and foremost, I offer my sincerest gratitude to my family, which I managed to see only three or four times per year. Going back even for few days, driving on the motorway to arrive home, and seeing the most beautiful lake in the world have always been an opportunity I tried to enjoy to the fullest. All their messages and Skype chats have always helped me throughout these years of my PhD.

I would like to thank my supervisor, Dr. Robert Nürnberg, for the thorough explanation of materials and methods, for his constant availability, and for the numerous corrections that have improved my thesis. To Marco Agnese, fellow PhD student, my gratitude for all the time spent together solving our problems with DUNE. Had he arrived at Imperial one or two years before, I am sure I would have proceeded faster! I wish him the best of luck for the second half of his PhD. A great thank you also to my office mates in 6M30 and 617: I have spent nice years with you guys!

An immense thank you to the community of More House, where I have resided all these years. My gratitude goes to the Sisters, Canonesses of St. Augustine, who asked me to become one of the evening sub-wardens. I thank them for their trust in me and their continual appreciation of my work for the house. To the members of the Chemin Neuf community, who were with us during my first two years, a deep gratitude as well for the dedication to the students they always showed. And now, a great thank to all, students and non-students (from More House and Imperial) I have met in these four years: María José, Annemarie, Matt, Chris, Joshua, Magda, Giada, Mate, Alphonsus, Grace, Sheng, Louisa, Caroline, Ingrid, Edoardo, Charlotte; Miriam, Diego, Maria, Neha, Jan, Monika, Katie, Tatiana, Qing, Plinio, Fr. Pietro; Naomi, Giulia, Krystian, Emanuele, Pedro, Michael, Fr. Thomas, Ewa, Irina, Christiana, Sofia, Pilar, Sr. Francesca; Hélène, Matteo, Alessio, Maristella, Chiara, Priscille, Marie, Joana, Marika, Angela, Nelya, Vianney; and many others that have made my stay at More House and Imperial nice.

I would like to thank all the people of the *Pure in Heart* group I have been attending on Wednesdays. Their joy, happiness and deep beliefs have been an example and a source of true friendships.

My deepest gratitude goes also to all my friends back home: Nicholas, Matteo, Caterina, Franca, Davide, Luca, Vincenzo, Maria, Enrica, Irene, and many more. To all of the them my best wishes for their future and their families!

Last but not least, I cannot help but mention the Divine Providence, which has always watched over me and always will. My full joy has not arrived yet, but there are surely more clues than in the past. Patience will do the rest!

Andrea

ABSTRACT

Continued research in electronic engineering technology has led to a miniaturisation of integrated circuits. Further reduction in the dimensions of the interconnects is impeded by the presence of small cracks or voids. Subject to high current and elastic stress, voids tend to drift and change shape in the interconnect, leading to a potential mechanical failure of the system.

This thesis investigates the temporal evolution of voids moving along conductors, in the presence of surface diffusion, electric loading and elastic stress. We simulate a bulk-interface coupled system, with a moving interface governed by a fourth-order geometric evolution equation and a bulk where the electric potential and the displacement field are computed.

We first give a general overview about geometric evolution equations, which define the motion of a hypersurface by prescribing its normal velocity in terms of geometric quantities. We briefly describe the three main approaches that have been proposed in the literature to solve numerically this class of equations, namely *parametric approach*, *level set approach* and *phase field approach*.

We then present in detail two methods from the *parametric approach* category for the void electro-stress migration problem. We first introduce an *unfitted* method, where bulk and interface grids are totally independent, i.e. no topological compatibility between the two grids has to be enforced over time. We then discuss a *fitted* method, where the interface grid is at all times part of the boundary of the bulk grid.

A detailed analysis, in terms of existence and uniqueness of the finite element solutions, experimental order of convergence (when the exact solution to the free boundary problem is known) and coupling operations (e.g., smoothing/remeshing of the grids, intersection between elements of the two grids), is carried out for both approaches. Several numerical simulations, both two- and three-dimensional, are performed in order to test the accuracy of the methods.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

A *geometric evolution equation* defines the motion of a hypersurface by prescribing its normal velocity in terms of geometric quantities. Geometric evolution equations and, more generally, time-dependent interface evolution problems, where the normal velocity depends also on field quantities evaluated on the analysed hypersurface, are ubiquitous in physics and engineering. The overall goal of this chapter is to give the reader a brief overview of these time-dependent interface problems, in terms of areas of application, useful definitions from differential geometry, and possible solution approaches.

The chapter is organised as follows: in Section 1.1 we draw a non-exhaustive list of areas of modern science where interface evolution problems are investigated and applied. In addition, we briefly introduce the void electro-stress migration problem that constitutes the object of investigation of this thesis. In Section 1.2 we present some definitions and known results from differential geometry, which will be employed at a later stage in the discussion of the electro-stress migration problem of our interest. In Section 1.3 we discuss the three main approaches presented in the literature for the solution of the aforementioned interface problems. In Section 1.4 we introduce the coupling between interface equations and bulk equations, where the latter are solved for field variables which are forcing terms in the interface equations. Finally, in Section 1.5 we compare the solution strategies introduced in § 1.3 for the coupled problems in § 1.4, highlighting their advantages and disadvantages.

1.1 INTRODUCTORY REMARKS

Geometric evolution equations and, more generally, time-dependent interface evolution problems are investigated and applied in many fields of modern science. We draw here a non-exhaustive list of such areas. Materials Science represents a traditional field of application. For instance, an appropriate mathematical modelling of the morphology of microstructure is necessary for a correct evaluation of the mechanical properties of materials. Another application is given by void electro-stress migration, which will be analysed in this thesis. More specifically, small voids or cracks contained in metallic wires can change their shape and location, due to the presence of surface diffusion and electro-stress loading. The analysis of this phenomenon aims at understanding how the migration of the voids affects the reliability of microelectronic circuits and their likelihood of failure. Other interesting applications where the evolution of surfaces is analysed have arisen in recent years. First, the motion of grain boundaries, which separate differing orientations of the same crystalline phase, or solid-liquid interfaces exhibiting dendritic structures in under-cooled solidification, can be modelled as time-dependent interface evolution problems. Other fields of research are given by image processing and multiphase flows. In image processing, a photograph or a video frame is analysed; separation of dark regions from a brighter background and identification of separating contours are implemented to detect and correctly cluster the objects in the image. A multiphase flow problem is defined as a fluid flow where more than one phase occurs. The moving interface separates the phases.

In order to simulate numerically the temporal evolution of surfaces, different methods can be employed. Our literature review closely follows the structure and the nomenclature adopted in [46], unless stated otherwise.

The overall goal is to find a family $\{\Gamma(t)\}_{t\in[0,T]}$ of closed compact and orientable hypersurfaces in $\mathbb{R}^d$ ($d = 2$ for curves, $d = 3$ for surfaces), whose evolution is defined by prescribing the velocity V of $\Gamma(t)$ in the normal direction $\vec{\nu}$. Evolution problems for surfaces can be expressed without loss of generality as

$$V = f(\vec{x}, \vec{\nu}, \kappa) \qquad \text{on } \Gamma(t)\,, \tag{1.1}$$

where κ is the sum of the $d-1$ principal curvatures of $\Gamma(t)$. We call κ the mean curvature, even though it is not the arithmetic mean of the principal curvatures. The function f depends on the problem considered. It might be necessary to evaluate on the surface $\Gamma(t)$ field variables which satisfy their own system of partial differential equations in $\mathbb{R}^d$ away from the surface. Therefore, a dependence on $\vec{x}$ in the right-hand side of (1.1) is introduced. We note that it suffices to define the normal velocity of the surface in order to specify its temporal evolution.

Two prototype problems are introduced here. For a more detailed introduction on the geometric quantities involved, see Section 1.2.

The first problem is *motion by mean curvature*, for which

$$V = \kappa \qquad \text{on } \Gamma(t)\,. \tag{1.2}$$

It is well known that, starting from an initial surface Γ_0, equation (1.2) is a gradient flow for the area functional

$$E(\Gamma) = \int_\Gamma 1 \, d\mathcal{H}^{d-1}\,, \tag{1.3}$$

where $\mathcal{H}^{d-1}$ denotes the $(d-1)$-dimensional Hausdorff measure. In practical applications the area functional is an *interfacial energy* with a constant energy density 1. Equation (1.2) can be interpreted as an analogue for surfaces of the parabolic heat equation

$$u_t - \triangle u = 0\,.$$

A second prototype equation that will be extensively investigated and used throughout the thesis is *motion by surface diffusion*

$$V = -\triangle_s \kappa \qquad \text{on } \Gamma(t)\,, \tag{1.4}$$

where $\triangle_s$ is the Laplace-Beltrami operator, or surface Laplacian, on $\Gamma(t)$. Eq. (1.4) can be interpreted as an analogue of the spatially fourth order parabolic equation

$$u_t + \triangle^2 u = 0\,.$$

We note that we slightly changed the notation used in [46, § 1], regarding the sign of the right-hand side term, for both (1.2) and (1.4), as we have a different sign convention for the curvature. Further details will be given in Section 1.2.2.

1.2 SOME GEOMETRIC ANALYSIS

The aim of this section is to introduce some definitions and known results from differential geometry, which will be used at a later stage in the presentation of the problem of our interest. Here we follow again the review article [46]. The references [76, 74] contain a more detailed exposition of this material.

1.2.1 *Hypersurfaces*

A subset $\Gamma \subset \mathbb{R}^d$ is called a C^2-hypersurface if for each point $\vec{x}_0 \in \Gamma$ there exists an open set $U \subset \mathbb{R}^d$ containing $\vec{x}_0$ and a function $u \in C^2(U)$ such that

$$U \cap \Gamma = \{\vec{x} \in U \,|\, u(\vec{x}) = 0\} \qquad \text{and} \qquad \nabla u(\vec{x}) \neq 0 \quad \forall\, \vec{x} \in U \cap \Gamma\,.$$

The tangent space $T_{\vec{x}}\Gamma$ is then the $(d-1)$-dimensional linear subspace of $\mathbb{R}^d$ that is orthogonal to $\nabla u(\vec{x})$. It does not depend on the particular function u which is chosen to describe Γ. A C^2-hypersurface $\Gamma \in \mathbb{R}^d$ is called orientable if there exists a vector-valued function $\vec{\nu} \in C^1(\Gamma, \mathbb{R}^d)$, i.e. $\vec{\nu} \in C^1$ in an open neighbourhood of Γ, such that $\vec{\nu}(\vec{x}) \perp T_{\vec{x}}\Gamma$ and $|\vec{\nu}(\vec{x})| = 1$ for all $\vec{x} \in \Gamma$. In what follows, we shall assume that $\Gamma \subset \mathbb{R}^d$ is an orientable C^2-hypersurface.

We define the tangential gradient of a function f, which is differentiable in an open neighbourhood of Γ, by

$$\nabla_s f(\vec{x}) = \nabla f(\vec{x}) - \nabla f(\vec{x}) \cdot \vec{\nu}(\vec{x})\, \vec{\nu}(\vec{x})\,, \qquad \vec{x} \in \Gamma\,. \tag{1.5}$$

Here ∇ denotes the usual gradient in $\mathbb{R}^d$. It can be noted that $\nabla_s f(\vec{x})$ is the orthogonal projection of $\nabla f(\vec{x})$ onto $T_{\vec{x}}\Gamma$. $\nabla_s f(\vec{x})$ depends only on the values of f on Γ. The following notation is employed throughout this chapter:

$$\nabla_s f(\vec{x}) = (\underline{D}_1 f(\vec{x}), \ldots, \underline{D}_d f(\vec{x}))$$

for the d components of the tangential gradient. It follows from the definition (1.5) that

$$\nabla_s f(\vec{x}) \cdot \vec{\nu}(\vec{x}) = 0 \qquad \vec{x} \in \Gamma\,.$$

If f is twice differentiable in an open neighbourhood of Γ, then we define the *Laplace-Beltrami operator* of f as

$$\triangle_s f(\vec{x}) = \nabla_s \cdot \nabla_s f(\vec{x}) = \sum_{i=1}^{d} \underline{D}_i \underline{D}_i f(\vec{x}) \,, \qquad \vec{x} \in \Gamma \,. \tag{1.6}$$

1.2.2 *Mean curvature*

We assume $\vec{\nu} \in C^1$ in a neighbourhood of Γ so that we may introduce the matrix

$$H_{jk}(\vec{x}) = -\underline{D}_j \nu_k(\vec{x}) \qquad j, k = 1, \ldots, d \qquad \vec{x} \in \Gamma \,. \tag{1.7}$$

It can be shown that $(H_{jk}(\vec{x}))$ is symmetric. Note that we changed the sign of the right-hand side of (1.7), compared to [46, § 2.3]. Furthermore,

$$\sum_{k=1}^{d} H_{jk}(\vec{x})\nu_k(\vec{x}) = \sum_{k=1}^{d} -\underline{D}_j \nu_k(\vec{x})\nu_k(\vec{x}) = -\tfrac{1}{2}\underline{D}_j |\vec{\nu}|^2(\vec{x}) = 0 \,,$$

since $|\vec{\nu}| = 1$ on Γ. Thus, $(H_{jk}(\vec{x}))$ has one eigenvalue which is equal to zero with corresponding eigenvector $\vec{\nu}(\vec{x})$. The remaining $d-1$ eigenvalues $\kappa_1(\vec{x}), \ldots, \kappa_{d-1}(\vec{x})$ are called the principal curvatures of Γ at the point $\vec{x}$. The mean curvature of Γ at $\vec{x}$ can then be defined as the trace of the matrix $(H_{jk}(\vec{x}))$, that is

$$\kappa(\vec{x}) = \sum_{j=1}^{d} H_{jj}(\vec{x}) = \sum_{j=1}^{d-1} \kappa_j(\vec{x}) \,. \tag{1.8}$$

Note that (1.8) differs from the more common definition of mean curvature, $\kappa = \frac{1}{d-1}\sum_{j=1}^{d-1} \kappa_j$. From (1.7) we derive the following expression for mean curvature:

$$\kappa(\vec{x}) = -\nabla_s \cdot \vec{\nu}(\vec{x}) \qquad \vec{x} \in \Gamma \,, \tag{1.9}$$

where $\nabla_s \cdot \vec{f} = \sum_{j=1}^{d} \underline{D}_j f_j$ denotes the tangential divergence of a vector field $\vec{f}$. In particular, if $\Gamma = \mathcal{S}^{d-1}$ and the unit normal field is chosen to point away from Γ, i.e. $\vec{\nu}(\vec{x}) = \vec{x}$, we obtain that $\kappa = -(d-1)$, on considering the particular function $f(\vec{x}) = x_j$, $j \in \{1, \dots, d\}$ and observing that $\underline{D}_i x_j = \delta_{ij} - \nu_j \nu_i$.

Moreover, while the sign of κ depends on the choice of the normal $\vec{\nu}$, the mean curvature vector $\kappa\vec{\nu}$ is an invariant. By choosing again the particular function $f(\vec{x}) = x_j$, $j \in \{1, \dots, d\}$ in (1.6) and recalling the application of the Laplace-Beltrami operator to each independent variable x_j, we deduce that

$$\triangle_s x_j = -\sum_{i=1}^{d} \underline{D}_i(\nu_j \nu_i) = -(\nabla_s \cdot \vec{\nu})\nu_j - \nabla_s \nu_j \cdot \vec{\nu} = \kappa \nu_j \,,$$

so that

$$\triangle_s \vec{x} = \kappa \vec{\nu} \qquad \text{on } \Gamma \,. \tag{1.10}$$

This identity of differential geometry will be useful at a later stage for the presentation of finite element approximations for *surface diffusion*, see Section 2.2.

1.3 POSSIBLE APPROACHES

An appropriate mathematical description of $\Gamma(t)$ is required for solving a geometric evolution equation analytically or numerically. Each choice leads to a particular nonlinear partial differential equation defining the evolution. According to the approach chosen to describe the surface, different solution strategies are possible. To this end, following [46, § 1.1], we recall here the three main approaches presented in the literature.

1.3.1 *Parametric approach*

The first approach we discuss is the *parametric approach*. We look for hypersurfaces $\Gamma(t)$ given as

$$\Gamma(t) = \vec{X}(\cdot, t)(\Upsilon) \,,$$

where Υ is a suitable reference manifold and $\vec{X} : \Upsilon \times [0, T] \to \mathbb{R}^d$ is a parameterisation to be determined. It is worth noting that the particular

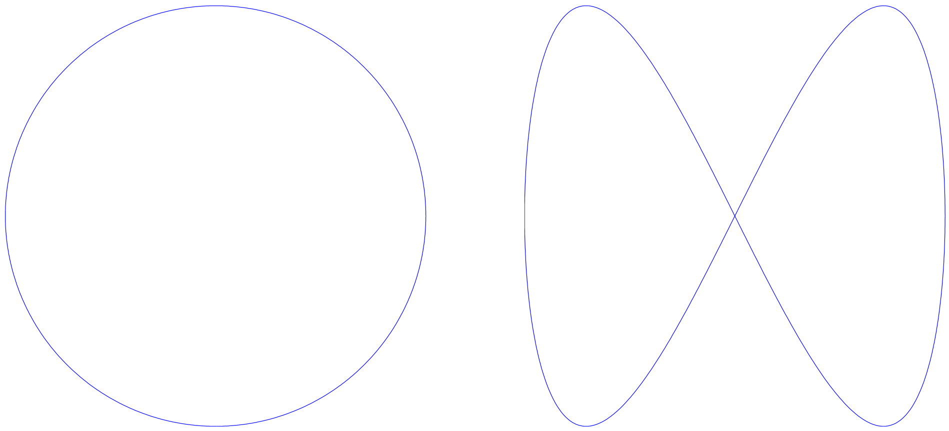

Figure 1.1: The lemniscate $\{\cos(t), \sin(2t)\}$ can be parameterised over the unit circle $\{\cos(t), \sin(t)\}$, $t \in [0, 2\pi]$.

choice of the manifold Υ fixes the topological type of $\Gamma(t)$. Here $X(\vec{p}, t)$, for $\vec{p} \in \Upsilon$, represents the position vector at time t of a point on $\Gamma(t)$. If closed curves in the plane are the object of investigation, then a suitable choice of Υ can be the unit circle $\mathcal{S}^1$, whereas if $\Gamma(t)$ is a two-dimensional surface, then Υ could be the unit sphere $\mathcal{S}^2$. Since evolution laws such as the prototypes (1.2) and (1.4) involve geometrical quantities, such quantities have to be expressed in terms of the derivatives of the parameterisation $\vec{X}$. Nonlinear parabolic systems of PDEs for the vector-valued function $\vec{X}$ are then obtained. With this approach, the surface is not the boundary of any open set and does not possess an inside and an outside. *Self-intersections* are natural for smooth parameterisations and are not necessarily associated with singularities.

We recall two interesting examples. The first one is for $d = 2$ and refers to a figure-eight curve smoothly mapped over the unit circle, see Figure 1.1. At the crossing point the curve has two smooth normals and curvatures, which depend on the parameterisation. A parameterised curve evolving by *mean curvature* can evolve smoothly from this configuration. The second example is for $d = 3$ and refers to a dumbbell-shaped surface, parameterised over the unit sphere, see [46, Fig. 1.1].

Examples of usage of the parametric approach for the geometric evolution equations (1.2) and (1.4) can be found in [50, 51, 6, 10, 12].

1.3.2 *Level set approach*

A second strategy for the manipulation of the interface is the *level set approach*. In this approach, the hypersurface $\Gamma(t)$ is given by the zero level set of an auxiliary function $u : \mathbb{R}^d \times [0, \infty) \to \mathbb{R}$, that is

$$\Gamma(t) = \{\vec{x} \in \mathbb{R}^d \,|\, u(\vec{x}, t) = 0\}.$$

The laws (1.2), (1.4) now translate into nonlinear, degenerate and singular PDEs for u. Contrary to the *parametric approach* described in § 1.3.1, the *level set approach* clearly exhibits the notion of $\Gamma(t)$ being a dividing surface between the two regions where the level set function is positive and negative, respectively. The notions of inside and outside are in this case well defined.

We recall here an example, where a figure-eight curve is described by a level set function, see [46, Fig. 1.3]. In this case, it is necessary to identify where the level set function is positive or negative, see [46, Fig. 1.4].

Examples of the usage of the level set method for the geometric evolution equations (1.2) and (1.4) can be found in [61, 62, 63, 64, 44, 45, 67, 65], while more details on the level set method in general are given in [115, 74, 108].

1.3.3 *Phase field approach*

A third way of describing the evolution of the surface $\Gamma(t)$ is the so-called *phase field approach*. The sharp interface $\Gamma(t)$ is approximated by a diffuse interface

$$\Gamma_\varepsilon(t) = \{\vec{x} \in \mathbb{R}^d \,|\, -1 + C\varepsilon \leqslant u_\varepsilon(\vec{x}, t) \leqslant 1 - C\varepsilon\},$$

of width $\mathcal{O}(\varepsilon)$, across which the phase field function u_ε varies from approximately the negative value -1 to approximately the positive value $+1$. The zero level set of the phase field function u_ε is used to approximate the surface $\Gamma(t)$. As discussed for the *level set approach*, we can identify inside and outside regions; interface self-intersection and topological changes are handled automatically. The bulk values of the phase field function correspond to the minima of a homogeneous energy function with two equal double wells. Using the gradient of the phase field function u_ε we can assign interfacial energy to the diffuse interface $\Gamma_\varepsilon(t)$.

Examples of the usage of the phase field approach for the geometric evolution equations (1.2) and (1.4) can be found in [60, 113, 43, 33, 66].

1.4 APPLICATIONS OF GEOMETRIC EVOLUTION EQUATIONS

In (1.1) we gave an example of a general geometric evolution equation. Recall that the motion of the interface is defined by prescribing the velocity V of $\Gamma(t)$ in the normal direction $\vec{\nu}$. The $\vec{x}$ dependence on the right-hand side of (1.1) might arise from evaluating on the surface $\Gamma(t)$ field variables which satisfy their own set of PDEs in a bulk domain, whose boundary (or part thereof) is precisely $\Gamma(t)$. In other words, it may become necessary to couple the interface equations on $\Gamma(t)$ with bulk equations involving unknowns which take values also away from the interface. Following and expanding on the overview in [46], we list some industrial applications where geometric evolution equations are involved. In Section 1.4.1 we present examples where the coupling between interface and bulk equations occurs. In Section 1.4.2 two additional applications, where no coupling to the bulk is involved, are introduced and discussed.

1.4.1 *Coupling interface equations to bulk equations*

In the following we list several examples where interface equations are coupled with bulk equations for field variables which appear on the right-hand side of (1.1).

- *Electro-stress migration*: microelectronic circuits usually contain small voids or cracks, and if those defects are large enough to sever the line, they cause an open circuit. The interface surface represents the voids, which can migrate in the conductor due to the presence of both surface diffusion and electro-stress loading. In the literature this problem has been treated numerically in the context of parametric methods ([27, 94, 123, 104]), level set methods ([2, 95] and [115, § 18.5]) and phase field methods ([100, 101, 25, 20, 9, 11]).

- *Stefan problem*: when a container is filled with an undercooled liquid, solidification of the liquid follows the nucleation of an initial solid seed with characteristic diameter larger than the critical radius. The seed will then grow into the liquid. An appropriate mathematical representation of this situation is the Stefan problem with kinetic undercooling, in which the solid-liquid interface is described by a surface $\Gamma(t)$ that has to be determined together with the temperature field. In the literature this problem has been treated numerically in the context of parametric methods ([14, 17]), level set methods ([67]) and phase field methods ([26, 35, 17, 19]).

- *Image processing*: this is a branch of imaging science, and deals with signal processing techniques where an image, a photograph or a video frame is received as input and analysed. Interface curves represent the contours of objects, which need to be detected and correctly clustered. In the literature this problem has been treated numerically in the context of parametric methods ([24]), level set methods ([96, 118]) and phase field methods ([111, 59]).

- *Multiphase flow*: this is defined as a fluid flow problem in which more than one phase occurs. The phases are separated by an interface. Such flows are ubiquitous in industrial applications, which range from bubble column reactors to ink-jet printing to fuel injection in engines and to biomedical engineering. In the literature this problem has been treated numerically in the context of front-tracking methods ([121, 18]), level set methods ([106, 107]) and phase field methods ([84, 98]).

1.4.2 *Additional applications*

It is worthwhile to mention two additional applications which are technologically important. In this case, no coupling to the bulk is involved.

- *Grain boundary motion*: grain boundaries in crystalline metallic materials are interfaces which separate two adjacent crystallites of the same crystal structure and chemical composition, but of different orientation. Grain boundaries are the fundamental defect in polycrystalline materials; associated with the grain boundary there is a surface energy which gives rise to a thermodynamic restoring force. For a constant surface energy density, this is the surface tension proportional to the mean curvature, and the resulting evolution law is (1.2). The grain boundary can move over time if an external loading is present. In the literature this problem has been treated numerically in the context of parametric methods ([56, 57]), level set methods ([56, 124]) and phase field methods ([47, 56, 71, 11, 93]).

- *Surface growth*: this is a process taking place between two media separated by an interface, which is the growing surface. A technological application is epitaxy, which refers to the deposition of a crystalline overlayer of atoms and molecules onto a substrate. Numerous physical mechanisms are present, and their different time and length scales affect the growth process. A mathematical description of the phenomenon involves a driving force representing the deposition flux of atoms onto the surface. In the literature this problem has been treated numerically in the context of parametric methods ([4, 5]), level set methods ([37]) and phase field methods ([86, 112]).

1.5 COMPARISON BETWEEN METHODS

As we have seen in Sections 1.3 and 1.4.1, the numerical solution of partial differential equations for systems with moving boundaries can be approached in different ways. Once again, three possible strategies can be employed in order to handle the interface: *parametric approach*, *level set approach* and *phase field approach*. In particular, in the *parametric approach* the user can make use of two different techniques: a *fitted approach* and an *unfitted approach*, depending on whether the topological compatibility between the bulk mesh and the interface mesh is preserved. In what follows, we will group the three aforementioned methods into either *implicit* or *explicit* methods, depending on how they represent the free boundary.

- *Implicit* surface methods do not track directly the coordinates of the points on the surface, but rather utilise fixed data points located at regular intervals throughout the computational domain to reconstruct the surface when needed. This class of methods is sometimes called "front-capturing", since they do not store the points on the surface explicitly, but rather carry the information needed to reconstruct the surface. The *level let approach* (described in Section 1.3.2) and the *phase field approach* (described in Section 1.3.3) belong to this category.
- *Explicit* surface methods, by contrast, discretise the surface using a set of connected points. This is sometimes called "front-tracking", as points on the surface are directly followed as the surface evolves, and these points define the surface itself; for example, the surface can be represented by a simplex mesh, i.e. a collection of line segments or triangles in two or three dimensions, respectively. The *parametric approach* described in Section 1.3.1 belongs to this category.

As discussed in [104], implicit and explicit methods differ in the way they handle topological changes, such as pinching-off and merging. The fact that such topological changes occur naturally within the framework of implicit methods is often seen as their main advantage over explicit methods. However, with modern tools available to incorporate topological changes into explicit methods, see e.g. [30], this advantage is reduced. In fact, within explicit methods the user can apply heuristic criteria aiming to detect if a topological change is imminent. In contrast to implicit methods, where topological changes take place automatically, this feature gives the user an active control over the topological changes, see also [85]. Heuristic strategies can also be applied to delay or prevent topological changes from happening. In addition, explicit methods have the advantage that the partial differential equation that governs the evolution of the interface can be solved with numerical methods in one dimension lower than is the case for implicit methods, where the interface is captured as the zero level set of an auxiliary function, as we have seen before in the case of *level set methods* and *phase field methods*. Finally, in applications where geometric quantities of the interface are of interest, implicit methods face the difficulty of extracting an explicit representation of the interface from the implicit definition. This is in general is a nontrivial operation, especially in higher space dimensions.

In this thesis we consider the *parametric approach* applied to *electro-stress migration* problems. *Fitted* and *unfitted* methods will be thoroughly analysed and compared.

2

FRONT-TRACKING METHODS FOR GEOMETRIC EVOLUTION EQUATIONS

Recall that in Chapter 1 we introduced the notion of a *geometric evolution equation*, which defines the motion of a hypersurface by prescribing its normal velocity in terms of geometric quantities. The overall goal of this chapter is to investigate efficient numerical tools for simulating the evolution of closed curves/surfaces in the setting of the *parametric approach*. As previously introduced in Section 1.3.1, the hypersurfaces $\Gamma(t)$ are given as

$$\Gamma(t) = \vec{X}(\cdot, t)(\Upsilon)\,,$$

where Υ is a suitable reference manifold (fixing the topological type of $\Gamma(t)$) and $\vec{X} : \Upsilon \times [0, T] \to \mathbb{R}^d$ has to be determined. Here $\vec{X}(\vec{p}, t)$, for $\vec{p} \in \Upsilon$, is the position vector at time t of a point on $\Gamma(t)$. We will confine our attention to two specific problems: *mean curvature flow* (eq. (1.2)) and *surface diffusion* (eq. (1.4)).

The chapter is organised as follows: in Section 2.1 we present some fundamental properties of geometric flows, recalling theoretical results already proven in the literature, for both *mean curvature flow* (§ 2.1.1) and *surface diffusion* (§ 2.1.2). In Section 2.2 we discuss the fully practical finite element approximation for the interface equations, which will be used throughout this thesis. We recall from the literature the main results about existence, uniqueness, stability, and some properties of the semidiscrete, continuous-in-time counterpart of the aforementioned approximation. We also compare it with other finite element approximations from the literature. In § 2.2.3 we analyse how the choice of two different quadrature rules (namely, mass lumping and exact integration) affects the evolution of moving interfaces, in terms of the quality of the final mesh. In § 2.2.4 we investigate the temporal evolution of surfaces when very small time steps are considered, and discuss a finite element scheme that aims at overcoming the appearance of spurious elongated elements. In Section 2.3 we extend our analysis allowing moving surfaces to have multiple components. In particular, § 2.3.1 discusses the software implementation for such interfaces, for which topological changes are predicted and performed, for both $d = 2$ and $d = 3$. Section 2.4 investigates several methods for the solution of the linear equations arising from the finite element approximation discussed in Section 2.2. We compare them in terms of CPU time and number of iterations (for iterative methods) for a number of sample problems. Finally, in 2.5 we summarise the most relevant points discussed in the chapter.

2.1 FUNDAMENTAL PROPERTIES OF GEOMETRIC FLOWS

For the sake of presentation, let us recall some fundamental properties associated to geometric flows.

Consider a family $\{\Gamma(t)\}_{t\in[0,T]}$ of hypersurfaces that evolve over time. Such a family is called a $C^{2,1}$-family of hypersurfaces if, for each point $(\vec{x}_0, t_0) \in \mathbb{R}^d \times (0,T)$ with $\vec{x}_0 \in \Gamma(t_0)$, there exists an open set $U \subset \mathbb{R}^d$, $\delta > 0$ and a function $u \in C^{2,1}(U \times (t_0 - \delta, t_0 + \delta))$ such that

$$U \cap \Gamma(t) = \{\vec{x} \in U \,|\, u(\vec{x},t) = 0\} \qquad \text{and} \qquad \nabla u(\vec{x},t) \neq 0 \quad \forall\, \vec{x} \in U \cap \Gamma(t)\,.$$

Consider now a family $\{\Gamma(t)\}_{t\in[0,T]}$ of evolving hypersurfaces which satisfies the above assumptions and suppose in addition that each surface $\Gamma(t)$ is compact. We are interested in the time derivative of certain volume and area integrals.

LEMMA 2.1. *Let $g \in C^1(Q)$, where Q is an open set containing*

$$\bigcup_{0<t<T} \Gamma(t) \times \{t\}\,.$$

Suppose in addition that each surface $\Gamma(t)$ is the boundary of an open bounded subset $\Omega(t) \subset \mathbb{R}^d$. Then

$$\frac{d}{dt}\int_{\Omega(t)} g \, d\mathcal{L}^d = \int_{\Omega(t)} \frac{\partial g}{\partial t}\, d\mathcal{L}^d + \int_{\Gamma(t)} gV \, d\mathcal{H}^{d-1}\,, \tag{2.1}$$

$$\begin{aligned} \frac{d}{dt}\int_{\Gamma(t)} g \, d\mathcal{H}^{d-1} &= \int_{\Gamma(t)} \frac{\partial g}{\partial t}\, d\mathcal{H}^{d-1} - \int_{\Gamma(t)} gV\kappa \, d\mathcal{H}^{d-1} \\ &\quad + \int_{\Gamma(t)} \frac{\partial g}{\partial \vec{\nu}} V \, d\mathcal{H}^{d-1}\,. \end{aligned} \tag{2.2}$$

Proof. For a proof, see [46, § 2.6, Lemma 2.1]. ✓

We are now ready to present the main theoretical results concerning *mean curvature flow* (§ 2.1.1) and *surface diffusion* (§ 2.1.2).

2.1.1 *Properties of Mean Curvature Flow*

In this section we investigate motion by *mean curvature* and describe some of its main features. Our literature review follows [46, § 3].

Consider a $C^{2,1}$-family of hypersurfaces $\{\Gamma(t)\}_{t\in[0,T]} \subset \mathbb{R}^d$ together with a choice $\vec{\nu}$ of a unit normal. Recall (1.2), which describes motion by *mean curvature*:

$$V = \kappa \qquad \text{on } \Gamma(t)\,,$$

where V denotes the velocity of $\Gamma(t)$ and κ is the sum of the $d-1$ principal curvatures of $\Gamma(t)$. The above equation gives rise to a parabolic equation for the function describing the surface $\Gamma(t)$, to which an initial condition

$$\Gamma(0) = \Gamma_0 \tag{2.3}$$

has to be provided.

In order to understand the time-dependent behaviour of a hypersurface subject to this flow, we consider the case of a shrinking sphere, where the solution to the geometric evolution equation is known in closed form. Let $\Gamma(t) = \partial B_{r(t)}(\vec{x}_0) \subset \mathbb{R}^d$, oriented by the outer normal $\vec{\nu}(\vec{x}) = \frac{\vec{x}-\vec{x}_0}{r(t)}$. It is straightforward to derive that $V = r'(t), \kappa = -\frac{d-1}{r(t)}$ on $\Gamma(t)$, so that $\Gamma(t)$ moves by *mean curvature* provided that $r'(t) = -\frac{d-1}{r(t)}$. The solution of this separable variable ODE is easily given by $r(t) = \sqrt{r_0^2 - 2(d-1)t}$, $0 \leqslant t < \frac{r_0^2}{2(d-1)}$, where $\Gamma_0 = \partial B_{r_0}(\vec{x}_0)$. It is worth noting that the surface $\Gamma(t)$ shrinks to a point as $t \nearrow \frac{r_0^2}{2(d-1)}$.

Motion by *mean curvature* exhibits an interesting *area-decreasing* property, which is obtained from the following lemma.

LEMMA 2.2. *Let* $\Gamma(t)$ *be a family of evolving hypersurfaces satisfying* (1.2) *and assume that each* $\Gamma(t)$ *is compact. Then*

$$\int_{\Gamma(t)} V^2 \, d\mathcal{H}^{d-1} + \frac{d}{dt}\mathcal{H}^{d-1}(\Gamma(t)) = 0\,. \tag{2.4}$$

Proof. Choosing $g \equiv 1$ in (2.2) and the evolution law (1.2) yields the desired result. ✓

For the benefit of the reader, we now recall from [46] some theoretical results about *mean curvature flow*. The law (1.2) gives rise to a second order parabolic problem, therefore the existence of a smooth solution locally in time for a smooth initial hypersurface Γ_0 can be expected. Furthermore, [53] showed that two smooth compact solutions which are initially disjoint will stay disjoint. Using the shrinking sphere as a comparison solution, it follows in particular that if $\Gamma(t)$, $0 \leqslant t < T$ is a smooth solution with $\Gamma_0 \subset B_{r_0}(\vec{x}_0)$, then $\Gamma(t) \subset B_{\sqrt{r_0^2 - 2(d-1)t}}(\vec{x}_0)$ for $0 \leqslant t < \min(T, \frac{r_0^2}{2(d-1)})$. Solutions will in general develop singularities in finite time before disappearing. However, it has been shown that certain initial configurations guarantee that solutions stay smooth until they shrink to a point. This result is expressed in the following theorem.

Theorem 2.3. *Let* $d \geqslant 3$ *and assume that* $\Gamma_0 \subset \mathbb{R}^d$ *is a smooth, compact and uniformly convex hypersurface. Then* (1.2)*, equipped with the initial condition* (2.3)*, has a smooth solution on a finite time interval* $[0, T)$ *and the* $\Gamma(t)$ *converge to a point as* $t \nearrow T$*. If one rescales the surfaces in such a way that the enclosed volume remains fixed, one has convergence against a sphere as* $t \nearrow T$*.*

Proof. A complete proof can be found in [83]. ✓

The case $d = 2$ is usually referred to as *curve shortening flow*.

Theorem 2.4. *Assume that* $\Gamma_0 \subset \mathbb{R}^2$ *is a smooth embedded closed curve, and so in particular it has no self-intersections. Then* (1.2)-(2.3) *have a smooth embedded solution on a finite time interval* $[0, T)$*, which shrinks to a 'round' point as* $t \nearrow T$*.*

Proof. The authors in [68] proved this result for convex Γ_0; subsequently in [77] it was shown that a smooth embedded closed curve remains smooth and embedded and becomes convex in finite time. ✓

Cusp-like singularities may develop if the initial curve is not embedded (see Figures 2.1 and 2.2). The analogue of Theorem 2.4 for surfaces does not hold, as can be seen by choosing a suitable dumbbell-shaped initial surface which develops a pinch-off singularity before it shrinks to a point (see

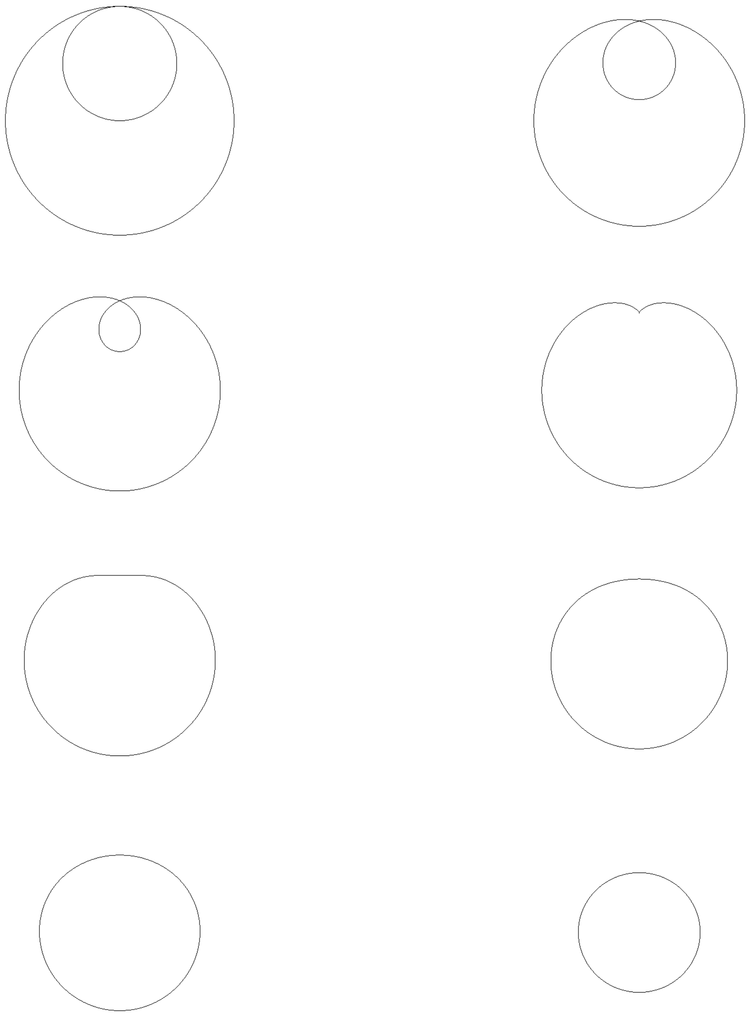

Figure 2.1: Motion by *mean curvature* (according to the discrete scheme that will be presented in (2.16a)-(2.16b)), applied to a curve with a self-intersection. A singularity (cusp) appears. The effect is that the algorithm jumps across the singularity. See Figure 2.2 for a magnified image.

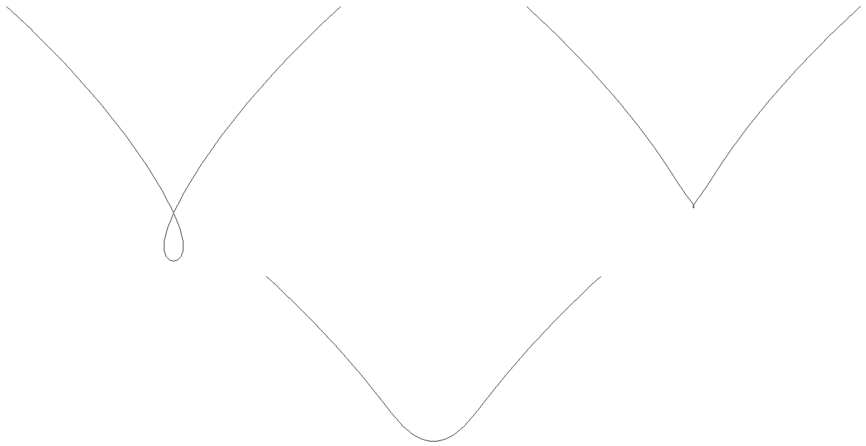

Figure 2.2: Close-up of Figure 2.1. The parametric theory breaks down.

[46, Fig. 4.5] and [78]). This pinch-off leads to a change of the topological type of $\Gamma(t)$, so that the *parametric approach* – in which the topological type is fixed – will develop a singularity that is difficult to handle. It follows that the question whether it is possible to introduce a notion of solution that is capable of following the flow through a singularity is of extreme importance. It is possible to apply some heuristic algorithms for the detection of topological changes (pinch-off, coalescence, etc.), which can lead to a modified interface mesh that can be handled again with the *parametric approach*. More details will be given in § 2.3.1.

2.1.2 *Properties of Surface Diffusion*

In this section we investigate motion by *surface diffusion* and describe some of its fundamental features. This particular geometric flow was introduced in (1.4), which we restate here for the sake of readability:

$$V = -\bigtriangleup_s \kappa \qquad \text{on } \Gamma(t)\,,$$

where $\bigtriangleup_s$ is the Laplace-Beltrami operator, or surface Laplacian, on $\Gamma(t)$.

Our discussion follows the introductory remarks given in [6]. Let us recall two fundamental properties of motion by *surface diffusion*. The first one is *conservation of volume* for closed hypersurfaces, while the second one is *area decrease*:

$$\begin{aligned}\frac{d}{dt}\mathcal{L}^d(\Omega(t)) &= -\int_{\Gamma(t)} V\, d\mathcal{H}^{d-1} = \int_{\Gamma(t)} \triangle_s\, \kappa\, d\mathcal{H}^{d-1} \\ &= -\int_{\Gamma(t)} \nabla_s\, \kappa \cdot \nabla_s\, 1\, d\mathcal{H}^{d-1} = 0\,, \qquad (2.5)\\ \frac{d}{dt}\mathcal{H}^{d-1}(\Gamma(t)) &= -\int_{\Gamma(t)} V\kappa\, d\mathcal{H}^{d-1} = -\int_{\Gamma(t)} |\nabla_s\, \kappa|^2\, d\mathcal{H}^{d-1} \leqslant 0\,, \qquad (2.6)\end{aligned}$$

where $\mathcal{L}^d(\Omega(t))$ is the Lebesgue measure of $\Omega(t)$ and $|\cdot|$ is the usual norm in $\mathbb{R}^d$. In fact, motion by *surface diffusion* is formally the H^{-1} gradient flow for the area functional, see [34]. A number of issues may arise, from existence, well posedness and regularity to the design of robust algorithms for the numerical simulation of (1.4). In [58] the authors showed local existence, regularity, and uniqueness of solutions when the initial surface is sufficiently smooth. They also proved that if the initial surface is embedded and close to a sphere, the solution exists globally and converges exponentially fast to a sphere. We refer to [54] for related results for curves in $\mathbb{R}^2$. In addition, it is worth mentioning two other features, which can occur for *surface diffusion* in finite time: a surface which starts as a graph may cease to be so (see [55] and [6, Fig. 1]), and a closed embedded hypersurface may self-intersect (see [75] and [6, Fig. 2]).

In designing efficient and robust algorithms for the numerical simulation of (1.4), it would be desirable to enforce (2.5) and (2.6) also in the discretisation schemes. This is not always theoretically possible, even if in some cases experimental expertise guarantees that conservation properties are observed in practice up to a very small tolerance.

2.2 FINITE ELEMENT APPROXIMATION OF GEOMETRIC FLOWS

In this section we analyse different parametric finite element approximations for the evolution of closed hypersurfaces $\Gamma \subset \mathbb{R}^d$, $d = 2, 3$, moving under given geometric flows such as motion by *mean curvature* and motion by *surface diffusion*, which have been introduced in previous sections. Particular attention will be paid to some crucial aspects, such as conservation properties, solvability, mesh smoothing, time-space adaptivity and topological changes.

In his seminal work [50] for the design of a finite element method for *mean curvature flow* (see also [51]), Dziuk employed the fundamental identity of differential geometry:

$$\kappa\vec{\nu} = \triangle_s \vec{X} = \triangle_s \vec{\mathrm{id}} \tag{2.7}$$

to obtain a variational discretisation of κ involving only first derivatives of the identity function $\vec{\mathrm{id}}$ on Γ, or equivalently, of the parameterisation $\vec{X} : \Upsilon \times [0,T] \to \mathbb{R}^d$ of Γ, where Υ is a suitable compact reference manifold without boundary in $\mathbb{R}^d$. Observe that in (2.7) we use a slight abuse of notation, so that the equation can be interpreted to hold either on Γ or on the reference manifold Υ. For the aforementioned parameterisation $\vec{X}$, (1.2) can be rewritten as

$$V := \vec{X}_t \cdot \vec{\nu} = \kappa\,, \qquad \kappa\vec{\nu} = \triangle_s \vec{X}\,, \tag{2.8}$$

while (1.4) can be rewritten as

$$V := \vec{X}_t \cdot \vec{\nu} = -\triangle_s \kappa\,, \qquad \kappa\vec{\nu} = \triangle_s \vec{X}\,. \tag{2.9}$$

We now discuss the fully practical finite element approximation for the variational formulation of (2.8) and (2.9) that will be used throughout the thesis. The schemes described below have been introduced in [10] for the case $d = 2$, and then extended to the case $d = 3$ in [12]. We closely follow the latter reference in the presentation of all the necessary quantities. Let $0 = t_0 < t_1 < \ldots < t_{M-1} < t_M = T$ be a partitioning of $[0,T]$ into possibly variable time steps $\tau_m := t_{m+1} - t_m$, $m = 0 \to M-1$. We set $\tau := \max_{m=0\to M-1} \tau_m$. Let $\Gamma^m \subset \mathbb{R}^d$ be a $(d-1)$-dimensional *polyhedral surface*, i.e. a union of non-degenerate $(d-1)$-simplices with no hanging vertices, which approximates the closed surface $\Gamma(t_m)$, $m = 0 \to M$. In particular, let $\Gamma^m = \bigcup_{j=1}^{J_\Gamma^m} \overline{\sigma_j^m}$, where $\{\sigma_j^m\}_{j=1}^{J_\Gamma^m}$ is a family of mutually disjoint open $(d-1)$-simplices with vertices $\{\vec{q}_k^m\}_{k=1}^{K_\Gamma^m}$. In order to approximate the position vector $\vec{X}$ and the curvature κ in (2.8) and (2.9), we need to introduce the following parametric finite element spaces: For $m = 0 \to M-1$, define

$$\begin{aligned}\underline{V}(\Gamma^m) &:= \{\vec{\chi} \in C(\Gamma^m, \mathbb{R}^d) : \vec{\chi}\,|_{\sigma_j^m} \text{ is linear } \forall\, j = 1 \to J_\Gamma^m\} \\ &=: [W(\Gamma^m)]^d \subset H^1(\Gamma^m, \mathbb{R}^d)\,, \end{aligned} \tag{2.10}$$

where $W(\Gamma^m) \subset H^1(\Gamma^m, \mathbb{R})$ is then the space of scalar continuous piecewise linear functions on Γ^m. The standard Lagrangian basis of $W(\Gamma^m)$ is denoted by $\{\phi_k^m\}_{k=1}^{K_\Gamma^m}$. For later purposes, we also introduce $\pi^m : C(\Gamma^m, \mathbb{R}) \to W(\Gamma^m)$, the standard interpolation operator at the nodes $\{\vec{q}_k^m\}_{k=1}^{K_\Gamma^m}$. This operator will be used when the finite element approximation of the coupled bulk-interface problem of our interest is presented. We will use π^m in order to evaluate on Γ^m bulk quantities that satisfy their own set of equations away from the interface. Throughout this thesis, the new closed surface Γ^{m+1} will be parameterised over Γ^m, with the help of a parameterisation $\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$, i.e. $\Gamma^{m+1} = \vec{X}^{m+1}(\Gamma^m)$. Moreover, for $m \geqslant 0$, $\vec{X}^m$ will be identified with $\vec{id} \in \underline{V}(\Gamma^m)$, i.e. the identity function on Γ^m.

For scalar and vector functions $v, w \in L^2(\Gamma^m, \mathbb{R}^{(d)})$ we introduce the L^2 inner product $\langle \cdot, \cdot \rangle_{\Gamma^m}$ over the current polyhedral surface Γ^m as follows

$$\langle v, w \rangle_{\Gamma^m} := \int_{\Gamma^m} v \cdot w \; d\mathcal{H}^{d-1}\,. \tag{2.11}$$

If v, w are piecewise continuous, we define the mass lumped inner product $\langle \cdot, \cdot \rangle_{\Gamma^m}^h$ as

$$\langle v, w \rangle_{\Gamma^m}^h := \frac{1}{d} \sum_{j=1}^{J_\Gamma^m} \mathcal{H}^{d-1}(\sigma_j^m) \sum_{k=1}^{d} (v \cdot w)((\vec{q}_{j_k}^m)^-)\,, \tag{2.12}$$

where $\{\vec{q}_{j_k}^m\}_{k=1}^d$ are the vertices of σ_j^m, in combination with the definition $v((\vec{q}_{j_k}^m)^-) := \lim_{\sigma_j^m \ni \vec{p} \to \vec{q}_{j_k}^m} v(\vec{p})$. Here $\mathcal{H}^{d-1}(\sigma_j^m) = \frac{1}{(d-1)!} |(\vec{q}_{j_2}^m - \vec{q}_{j_1}^m) \wedge \cdots \wedge (\vec{q}_{j_d}^m - \vec{q}_{j_1}^m)|$ is the measure of σ_j^m, and $\wedge$ indicates the standard cross product on $\mathbb{R}^d$. We will also denote $|\cdot|_{\Gamma^m}^{(h)}$ as

$$|v|_{\Gamma^m}^{(h)} := (\langle v, v \rangle_{\Gamma^m}^{(h)})^{\frac{1}{2}}\,.$$

Here and throughout this thesis, $\cdot^{(*)}$ denotes an expression with or without the superscript $*$, and similarly for subscripts. We note that the inner product (2.12) corresponds to the trapezium rule for numerical integration, which can integrate exactly polynomials of degree 1. An exact quadrature formula will be discussed at a later stage, in § 2.2.3. In addition, the unit normal $\vec{\nu}^m$ to Γ^m is defined by

$$\vec{\nu}_j^m := \vec{\nu}^m|_{\sigma_j^m} := \frac{(\vec{q}_{j_2}^m - \vec{q}_{j_1}^m) \wedge \cdots \wedge (\vec{q}_{j_d}^m - \vec{q}_{j_1}^m)}{|(\vec{q}_{j_2}^m - \vec{q}_{j_1}^m) \wedge \cdots \wedge (\vec{q}_{j_d}^m - \vec{q}_{j_1}^m)|}, \tag{2.13}$$

where we have assumed that the vertices $\{\vec{q}_{j_k}^m\}_{k=1}^d$ of σ_j^m are ordered such that $\vec{\nu}^m : \Gamma^m \to \mathbb{R}^d$ induces an orientation on Γ^m.

Before we describe fully practical finite element approximations to (2.8) and (2.9), it is necessary to introduce the notion of a vertex normal on Γ^m. We will combine this definition with a natural assumption that is needed in order to show existence and uniqueness for the discrete schemes.

$(\mathcal{A})$ We assume for $m = 0 \to M-1$ that $\mathcal{H}^{d-1}(\sigma_j^m) > 0$ for all $j = 1 \to J_\Gamma^m$. For $k = 1 \to K_\Gamma^m$, let $\mathcal{T}_k^m := \{\sigma_j^m : \vec{q}_k^m \in \overline{\sigma_j^m}\}$ and set

$$\Lambda_k^m := \bigcup_{\sigma_j^m \in \mathcal{T}_k^m} \overline{\sigma_j^m},$$

$$\vec{\omega}_k^m := \frac{1}{\mathcal{H}^{d-1}(\Lambda_k^m)} \sum_{\sigma_j^m \in \mathcal{T}_k^m} \mathcal{H}^{d-1}(\sigma_j^m)\, \vec{\nu}_j^m. \tag{2.14}$$

Then we further assume that $\vec{\omega}_k^m \neq \vec{0}$, $k = 1 \to K_\Gamma^m$, and that dim span $\{\vec{\omega}_k^m\}_{k=1}^{K_\Gamma^m} = d$, $m = 0 \to M-1$.

Given the above definitions, the piecewise linear vertex normal function can be defined as

$$\vec{\omega}^m := \sum_{k=1}^{K_\Gamma^m} \chi_k^m\, \vec{\omega}_k^m \in \underline{V}(\Gamma^m).$$

Moreover, combining (2.14) with the definition of the mass lumped inner product (2.12), we note that

$$\langle \vec{z}, w\, \vec{\nu}^m \rangle_{\Gamma^m}^h = \langle \vec{z}, w\, \vec{\omega}^m \rangle_{\Gamma^m}^h \qquad \forall\, \vec{z} \in \underline{V}(\Gamma^m),\ w \in W(\Gamma^m). \tag{2.15}$$

Remark 2.5. *The aforementioned assumption $(\mathcal{A})$ requires the interface mesh to include non-degenerate $(d-1)$-simplices only. Moreover, the set $\mathcal{T}_k^m$, defined for* *every vertex of the grid, represents the set of elements which have $\vec{q}_k^m$ as one of their vertices. For instance, when $d = 2$, Γ^m is a curve, so $\mathcal{T}_k^m$ is composed of those two segments sharing the corner $\vec{q}_k^m$. For the case $d = 3$, the cardinality of $\mathcal{T}_k^m$ varies with the vertex k, depending on the connectivity of the initial triangulation. By the help of these remarks, one can interpret $\vec{\omega}_k^m$ as a weighted normal defined at the node $\vec{q}_k^m$ of the surface Γ^m, where in general $|\vec{\omega}_k^m| < 1$. In addition, the authors in [10, 12] noted that $(\mathcal{A})$ is only violated in very rare occasions. We report here their analysis for the benefit of the reader.*

$(\mathcal{A})$ always holds for surfaces without self-intersections. For $d = 2$, since $\vec{\omega}_k^m$ points in the direction $\left[\vec{q}_{k+1}^m - \vec{q}_{k-1}^m\right]^\perp$, $(\mathcal{A})$ can be equivalently reformulated to exclude the following situation: All points $\{\vec{q}_k^m : k$ is even$\}$ lie on one straight line and simultaneously all points $\{\vec{q}_k^m : k$ is odd$\}$ lie on another parallel line. For a closed curve, we obtain that in the case that K_Γ^m is odd, then $(\mathcal{A})$ immediately holds provided all points do not lie on one straight line. If K_Γ^m is even, then $(\mathcal{A})$ is only violated on very rare occasions, see e.g. Figure 2.3.

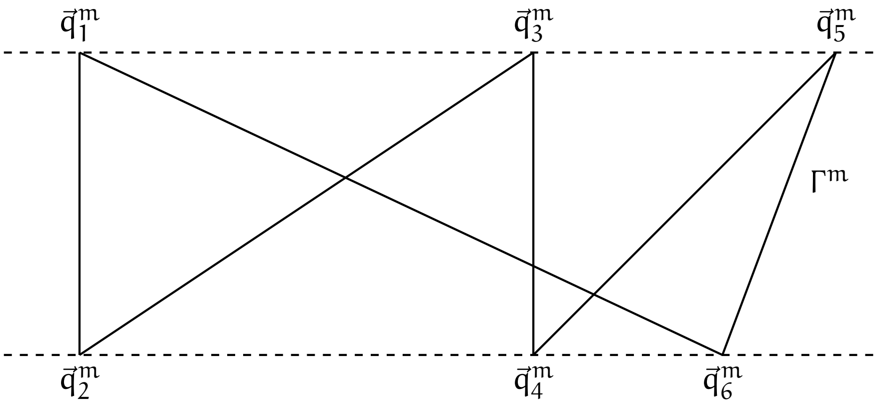

Figure 2.3: The assumption $(\mathcal{A})$ is violated in this case, for $d = 2$ (adapted from [10, Fig. 3]). Since $\vec{\omega}_k^m$ points in the direction $\left[\vec{q}_{k+1}^m - \vec{q}_{k-1}^m\right]^\perp$, by analysing all the pairs $\{\vec{q}_{k+1}^m, \vec{q}_{k-1}^m\}$ we immediately deduce that dim span $\{\vec{\omega}_k^m\}_{k=1}^{K_\Gamma^m} = 1 \neq 2$, which violates the assumption $(\mathcal{A})$.

We are now ready to recall the following approximation to (2.8): Given Γ^0 and the identity function $\vec{X}^0 \in \underline{V}(\Gamma^0)$ on Γ^0, then for $m = 0 \to M-1$ find $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ such that $\forall\, \chi \in W(\Gamma^m)$, $\forall\, \vec{\eta} \in \underline{V}(\Gamma^m)$

$$\left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi\, \vec{\nu}^m \right\rangle_{\Gamma^m}^h - \langle \kappa^{m+1}, \chi \rangle_{\Gamma^m}^h = 0\,, \tag{2.16a}$$

$$\langle \kappa^{m+1}\, \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m}^h + \langle \nabla_s\, \vec{X}^{m+1}, \nabla_s\, \vec{\eta} \rangle_{\Gamma^m} = 0\,, \tag{2.16b}$$

where, as noted above, the inner products $\langle \cdot, \cdot \rangle_{\Gamma^m}^{(h)}$ as well as ∇_s depend on m.

Moreover, (2.9) will be approximated with the help of the following scheme: Given Γ^0 and the identity function $\vec{X}^0 \in \underline{V}(\Gamma^0)$ on Γ^0, then for $m = 0 \to M-1$ find $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ such that $\forall\, \chi \in W(\Gamma^m)$, $\forall\, \vec{\eta} \in \underline{V}(\Gamma^m)$

$$\left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi\, \vec{\nu}^m \right\rangle_{\Gamma^m}^h - \langle \nabla_s\, \kappa^{m+1}, \nabla_s\, \chi \rangle_{\Gamma^m} = 0\,, \tag{2.17a}$$

$$\langle \kappa^{m+1}\, \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m}^h + \langle \nabla_s\, \vec{X}^{m+1}, \nabla_s\, \vec{\eta} \rangle_{\Gamma^m} = 0\,. \tag{2.17b}$$

We now investigate existence and uniqueness of solutions to the schemes (2.16a)-(2.16b) and (2.17a)-(2.17b).

THEOREM 2.6. *Let the assumption* $(\mathcal{A})$ *hold. Then there exist unique solutions* $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ *to the systems* (2.16a)-(2.16b) *and* (2.17a)-(2.17b).

Proof. The complete proof can be found in [12, Thm. 2.1]. ✓

Furthermore, the schemes (2.16a)-(2.16b) and (2.17a)-(2.17b) are unconditionally stable. This feature is proven in the following theorem.

THEOREM 2.7. *Let the assumption* $(\mathcal{A})$ *hold, and* $\{\vec{X}^m, \kappa^m\}_{m=1}^M$ *be the unique solution to the system* (2.16a)-(2.16b). *Then for* $k = 1 \to M$ *we have that*

$$|\Gamma^k| + \sum_{m=0}^{k-1} \tau_m\, (|\kappa^{m+1}|_{\Gamma^m}^h)^2 \leqslant |\Gamma^0|\,. \tag{2.18}$$

Moreover, for the unique solution $\{\vec{X}^m, \kappa^m\}_{m=1}^M$ *to the system* (2.17a)-(2.17b), *for* $k = 1 \to M$ *we have that*

$$|\Gamma^k| + \sum_{m=0}^{k-1} \tau_m\, |\nabla_s\, \kappa^{m+1}|_{\Gamma^m}^2 \leqslant |\Gamma^0|\,. \tag{2.19}$$

Proof. We refer to [10, Thm. 2.3] for the case $d = 2$ and to [12, Thm. 2.2] for $d = 3$. ✓

2.2.1 *Semidiscrete continuous-in-time approximation*

As shown in [10, Rem. 2.3] and [12, § 4], it is worthwhile to consider continuous-in-time semidiscrete versions of the aforementioned finite element schemes. For example, (2.17a)-(2.17b) can be replaced by: For a.e. $t \in (0, T)$ find $\{\vec{X}^h, \kappa^h\} \in \underline{V}(\Gamma^h(t)) \times W(\Gamma^h(t))$ such that $\forall\ \chi \in W(\Gamma^h(t))$, $\forall\ \vec{\eta} \in \underline{V}(\Gamma^h(t))$

$$\langle \vec{X}^h_t, \chi\,\vec{\nu}^h\rangle^h_{\Gamma^h(t)} - \langle \nabla_s\,\kappa^h, \nabla_s\,\chi\rangle_{\Gamma^h(t)} = 0\,, \tag{2.20a}$$

$$\langle \kappa^h\,\vec{\nu}^h, \vec{\eta}\rangle^h_{\Gamma^h(t)} + \langle \nabla_s\,\vec{X}^h, \nabla_s\,\vec{\eta}\rangle_{\Gamma^h(t)} = 0\,, \tag{2.20b}$$

where we always integrate over the current surface $\Gamma^h(t)$ (with normal $\vec{\nu}^h(t)$) described by the identity function $\vec{X}^h(t) \in \underline{V}(\Gamma^h(t))$. In addition, $\langle\cdot,\cdot\rangle^{(h)}_{\Gamma^h(t)}$ is the continuous-in-time counterpart of $\langle\cdot,\cdot\rangle^{(h)}_{\Gamma^m}$. where Γ^m and $\vec{X}^m$ are replaced by $\Gamma^h(t)$ and $\vec{X}^h(t)$, respectively. It is straightforward to show that (2.20a)-(2.20b) conserves the enclosed volume exactly: by choosing $\chi \equiv 1$ in (2.20a) and recalling the mass lumped inner product (2.12), we can show that

$$0 = \langle \vec{X}^h_t, \vec{\nu}^h\rangle^h_{\Gamma^h(t)} = \int_{\Gamma^h(t)} \vec{X}^h_t \cdot \vec{\nu}^h \, d\mathcal{H}^{d-1} = \frac{d}{dt}\,\mathcal{L}^d(\Omega^h(t))\,, \tag{2.21}$$

where $\Omega^h(t) \subset \mathbb{R}^d$ is open bounded set whose boundary is given by $\Gamma^h(t)$. It does not appear possible to prove the analogue of (2.21) for the fully discrete scheme (2.17a)-(2.17b). However, from all the numerical simulations performed in this thesis, we experimentally observe that the enclosed volume is approximately preserved, and that the volume loss tends to zero as $\tau \to 0$.

We now investigate an equidistribution property for the scheme (2.17a)-(2.17b).

THEOREM 2.8. *Let* $d = 2$ *and let* $\{\vec{q}^h_k\}_{k=1}^{K^h_\Gamma}$ *denote the sequentially ordered vertices on* $\Gamma^h(t)$. *Let* $\{\vec{X}^h, \kappa^h\}$ *be a solution to the system* (2.20a)-(2.20b), *and set* $\vec{h}_k(t) := \vec{q}^h_k(t) - \vec{q}^h_{k-1}(t)$. *Then it holds that*

$$|\vec{h}_k(t)| = |\vec{h}_{k-1}(t)| \quad \textit{if} \quad \vec{h}_k(t) \nparallel \vec{h}_{k-1}(t) \quad k = 1, \ldots, K^h_\Gamma\,. \tag{2.22}$$

Proof. We need to prove that the scheme (2.20a)-(2.20b) will always equidistribute the vertices along $\Gamma^h(t)$ for $t > 0$, provided that they are not locally parallel. A proof of (2.22) can be found in [10, Rem. 2.4]. We repeat here the modified approach followed in [104, Thm. 3.2] for the benefit of the reader. Choosing $\vec{\eta} = (\vec{h}_{k+1} + \vec{h}_k)\,\phi_k \in \underline{V}(\Gamma^h(t))$ in (2.20b), where $\{\phi_l\}_{l=1}^{K_\Gamma^h}$ denote the basis functions of $W(\Gamma^h(t))$, and recalling (2.12) and the equality (2.15), we obtain that for $k = 1 \to K_\Gamma^h$

$$0 = \langle \nabla_s \vec{X}^h, \nabla_s \,[(\vec{h}_{k+1} + \vec{h}_k)\,\phi_k]\rangle_{\Gamma^h(t)}\,. \tag{2.23}$$

The integral in (2.23) can now be easily computed by hand, since the support of ϕ_k is given by $[\vec{q}_{k-1}, \vec{q}_k] \cup [\vec{q}_k, \vec{q}_{k+1}]$. On noting that $\nabla_s\,\phi_k = \frac{1}{|\vec{h}_k|}$ on $[\vec{q}_{k-1}, \vec{q}_k]$ and $\nabla_s\,\phi_k = -\frac{1}{|\vec{h}_{k+1}|}$ on $[\vec{q}_k, \vec{q}_{k+1}]$, we then deduce that

$$\left[\frac{\vec{h}_{k+1}}{|\vec{h}_{k+1}|} - \frac{\vec{h}_k}{|\vec{h}_k|}\right] \cdot (\vec{h}_{k+1} + \vec{h}_k) = 0\,. \tag{2.24}$$

It immediately follows from (2.24) that

$$(|\vec{h}_{k+1}| - |\vec{h}_k|)(|\vec{h}_{k+1}|\,|\vec{h}_k| - \vec{h}_{k+1}\cdot\vec{h}_k) = 0\,, \tag{2.25}$$

which means that the desired result (2.22) follows directly from the Cauchy-Schwarz inequality. ✓

REMARK 2.9. *Of course, the analysis in* (2.25) *immediately carries over to a fully discrete scheme that is fully implicit; that is,* $\vec{v}^m$ *and* $\langle\cdot,\cdot\rangle_{\Gamma^m}^{(h)}$ *in* (2.17a)-(2.17b) *are replaced by* $\vec{v}^{m+1}$ *and* $\langle\cdot,\cdot\rangle_{\Gamma^{m+1}}^{(h)}$*. These approximations have been discussed in [16], where the authors proved that they are unconditionally stable and equidistribute a given parameterisation after one time step. However, the highly nonlinear nature of these schemes makes them less practical than the scheme* (2.17a)-(2.17b)*. It does not appear possible to prove an analogue of* (2.25) *for* (2.17a)-(2.17b)*. Nevertheless, in practice we see that over a number of time steps, the vertices are moved tangentially so that they will eventually be equidistributed.*

REMARK 2.10. *The case* $d = 3$ *has been thoroughly investigated in [12, § 4]. We recall here the main results for the benefit of the reader. Surfaces satisfying* (2.20b) *are called conformal polyhedral surfaces. The authors showed that infinitesimally small "tangential" changes to a conformal polyhedral surface* $\Gamma^h(t)$ *do not decrease* *its surface area and maintain its volume. Moreover, if only one vertex is moved, the enclosed volume is conserved if and only if this movement is "tangential". Hence no individual vertex in* $\Gamma^h(t)$ *can be moved so as to maintain the volume, and decrease the surface area to leading order. In addition, the authors showed that it is possible to characterise conformal polyhedral surfaces with a stronger condition. The local criticality condition to "tangential" variations can be strengthened to a global minimisation condition, if the surface area is replaced by a Dirichlet integral. Precisely, it is clear from* (2.20b) *that*

$$\langle \nabla_s \vec{X}^h, \nabla_s \vec{\eta} \rangle_{\Gamma^h(t)} = 0 \tag{2.26}$$

for all $\vec{\eta} \in \underline{V}_{\vec{\tau}}(\Gamma^h(t)) := \{\vec{\xi} \in \underline{V}^h(\Gamma^h(t)) \, : \, \vec{\xi}(\vec{q}^h_k(t)) \cdot \vec{\omega}^h(\vec{q}^h_k(t), t) = 0,\, k = 1, \ldots, K^h_\Gamma\}$*., where* $\vec{\omega}^h(\cdot, t)$ *denotes the continuous-in-time counterpart of the vertex normal function* $\vec{\omega}^m$*. The space* $\underline{V}_{\vec{\tau}}(\Gamma^h(t))$ *contains therefore all the polynomial, piecewise linear parameterisations of* $\Gamma^h(t)$ *that are orthogonal to* $\vec{\omega}^h(\cdot, t)$ *at every vertex of the interface* $\Gamma^h(t)$*. It can be shown that* (2.26) *leads to*

$$\begin{aligned} \mathcal{H}^{d-1}(\Gamma^h(t)) &= \frac{1}{2} \int_{\Gamma^h(t)} |\nabla_s \vec{X}^h|^2 \, d\mathcal{H}^{d-1} \\ &= \min_{\vec{\eta} \in \underline{V}_{\vec{\tau}}(\Gamma^h(t))} \frac{1}{2} \int_{\Gamma^h(t)} |\nabla_s (\vec{X}^h + \vec{\eta})|^2 \, d\mathcal{H}^{d-1} \,, \end{aligned} \tag{2.27}$$

which guarantees good mesh properties, since the parameterisation $\vec{X}^h(t)$ *for all* $t \in (0, T]$ *cannot move any vertex as to preserve the volume and at the same time decrease the Dirichlet integral in* (2.27)*.*

2.2.2 *Comparison with other discretisation schemes*

It is worthwhile to mention two alternative finite element approximations that have been introduced in the literature.

The first scheme approximates *mean curvature flow* (eq. (1.2)). It was proposed in [50] and can be formulated as follows: Find $\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$ such that

$$\left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \vec{\eta} \right\rangle^h_{\Gamma^m} + \langle \nabla_s \vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_{\Gamma^m} = 0 \qquad \forall\, \vec{\eta} \in \underline{V}(\Gamma^m)\,. \tag{2.28}$$

The system (2.28) is a discretisation of the variational formulation of

$$\vec{X}_t = \Delta_s \vec{X},$$

as opposed to (2.8). It can be shown that the scheme (2.16a)-(2.16b) can be rewritten as: Find $\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$ such that, for all $\vec{\eta} \in \underline{V}(\Gamma^m)$

$$\left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m} \cdot \vec{\omega}^m, \vec{\eta} \cdot \vec{\omega}^m \right\rangle_{\Gamma^m}^h + \langle \nabla_s \vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_{\Gamma^m} = 0\,, \tag{2.29}$$

which clearly highlights the key difference between the two schemes.

The second scheme approximates *surface diffusion* (eq. (1.4)). It was introduced in [6] and can be stated as follows. Let $\vec{X}^{m+1} := \vec{X}^m + \tau_m \vec{V}^{m+1}$, where $\vec{V}^{m+1} \in \underline{V}(\Gamma^m)$ is part of the solution of: Find $\{\vec{V}^{m+1}, \kappa^{m+1}, \vec{\kappa}^{m+1}, V^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m) \times \underline{V}(\Gamma^m) \times W(\Gamma^m)$ such that, for all $\chi \in W(\Gamma^m)$ and for all $\vec{\eta} \in \underline{V}(\Gamma^m)$

$$\langle \vec{\kappa}^{m+1}, \vec{\eta} \rangle_{\Gamma^m} + \tau_m \langle \nabla_s \vec{V}^{m+1}, \nabla_s \vec{\eta} \rangle_{\Gamma^m} = -\langle \nabla_s \vec{X}^m, \nabla_s \vec{\eta} \rangle_{\Gamma^m}\,, \tag{2.30a}$$

$$\langle \kappa^{m+1}, \chi \rangle_{\Gamma^m} - \langle \vec{\kappa}^{m+1}, \chi \vec{\nu}^m \rangle_{\Gamma^m} = 0\,, \tag{2.30b}$$

$$\langle V^{m+1}, \chi \rangle_{\Gamma^m} - \langle \nabla_s \kappa^{m+1}, \nabla_s \chi \rangle_{\Gamma^m} = 0\,, \tag{2.30c}$$

$$\langle \vec{V}^{m+1}, \vec{\eta} \rangle_{\Gamma^m} - \langle V^{m+1} \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m} = 0\,. \tag{2.30d}$$

The system (2.30a)-(2.30d) is a discretisation of the variational formulation of

$$\vec{\kappa} = \Delta_s \vec{X}\,, \qquad \kappa = \vec{\kappa} \cdot \vec{\nu}\,, \qquad \mathcal{V} = -\Delta_s \kappa\,, \qquad \vec{X}_t = \vec{\mathcal{V}} = \mathcal{V} \vec{\nu}\,,$$

as opposed to (2.9). We note that both schemes (2.28) and (2.30a)-(2.30d) only change the approximation of $\vec{X}$ in the normal direction, whereas the schemes used throughout this thesis also induce tangential changes. This is a crucial difference, which we address here for the benefit of the reader by recalling some simulations presented in the literature. Before comparing (2.28) with (2.16a)-(2.16b) and (2.30a)-(2.30d) with (2.17a)-(2.17b), let us introduce two indicators of mesh quality. Similarly to [12, § 5], and recalling (2.14), we set

$$h_{\Gamma^m} = \max_{k=1\to K_\Gamma^m} \left\{ \max_{\vec{p}_l \in \partial \Lambda_k^m} |\vec{q}_k^m - \vec{p}_l| \right\}, \tag{2.31}$$

$$l_{\Gamma^m} = \min_{k=1\to K_\Gamma^m} \left\{ \min_{\vec{p}_l \in \partial \Lambda_k^m} |\vec{q}_k^m - \vec{p}_l| \right\}, \tag{2.32}$$

where h_{Γ^m} and l_{Γ^m} represent the maximum and minimum segment of the hypersurface Γ^m, respectively.

The first example we recall is from [10, Fig. 4]. The experiment is for a mild ellipse. The parameters are chosen as follows: $J_\Gamma^m = K_\Gamma^m = 64$, $\tau = 10^{-6}$, $T = 10^{-3}$ and the initial curve is a $3:1$ ellipse with semiminor axis $R = 0.075$. The scheme (2.30a)-(2.30d) breaks down at $t = 3.2 \times 10^{-4}$, due to a coalescence of grid points. On the other hand, the scheme (2.17a)-(2.17b) intrinsically moves the vertices such that the problem can be computed until time $t = T$, when the solution has reached the shape of a circle. The second example we recall is from [10, Fig. 5]. Starting with an initial curve that consists of a semi-circle and a single additional node on the periphery of the circle, we investigate the ratio

$$r_h := h_{\Gamma^m} / l_{\Gamma^m} \tag{2.33}$$

over time. The parameters are $J_\Gamma^m = K_\Gamma^m = 128$, $\tau = 10^{-7}$, $T = 5 \times 10^{-4}$ and $R = 0.075$ as the radius of the circle. The scheme (2.30a)-(2.30d) could compute only up to time $t = 8.4 \times 10^{-5}$. From [10, Fig. 5] one can clearly see that although the true solution (a circle) is reached very quickly (at around time $t = 2 \times 10^{-5}$), in the remaining time the vertices are continually moved tangentially, which results in a further decrease in the ratio r_h, which approaches the optimal value 1; see [10, Fig. 6]. The coalescence of vertices for the scheme (2.30a)-(2.30d) can be prevented by heuristically redistributing all the mesh points tangentially after each time step, as described in [6]. However, such mesh operations can have a dramatic effect on the size of the final circular solution.

The final example we recall is from [12, Figs. 2 and 3]. The geometry is a $2 \times 1 \times 1$ cuboid, with the initial triangulation given by $K_\Gamma^m = 1282$ vertices and $J_\Gamma^m = 2560$ triangles. The remaining parameters are chosen as $\tau = 10^{-3}$ and $T = 0.14$; see [12, Fig. 2]. In order to highlight one difference between the two schemes, for each of them the ratios r_h and

$$r_a := \max_{j=1\to J_\Gamma^m} \mathcal{H}^{d-1}(\sigma_j^m) \,/ \min_{j=1\to J_\Gamma^m} \mathcal{H}^{d-1}(\sigma_j^m) \tag{2.34}$$

can be plotted over time, see [12, Fig. 3]. One can clearly see that the ratios increase substantially for the scheme (2.28), while the tangential movement of vertices induced by (2.16a)-(2.16b), as discussed in Section 2.2.1, results in only a moderate increase in the ratios r_h and r_a. In all our simulations for the case $d = 3$, r_h and r_a qualitatively behave very similarly. Therefore, in this particular case, from now on we will only present plots of r_a, in order to visualise the behaviour of the mesh quality over time.

2.2.3 *Comparison between mass lumping and true integration*

It is worthwhile to investigate a variant of the schemes (2.16a)-(2.16b) and (2.17a)-(2.17b), where the mass lumped quadrature rule (2.12), which integrates exactly polynomials of degree 1, is replaced by exact integration (2.11). The choice of an appropriate set of Gaussian quadrature nodes allows then the exact calculation of the integrals where the integrand functions are polynomials of degree 2. For *mean curvature flow*, we now solve the following system: Find $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ such that for all $\chi \in W(\Gamma^m)$, $\vec{\eta} \in \underline{V}(\Gamma^m)$

$$\left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi \, \vec{\nu}^m \right\rangle_{\Gamma^m} - \langle \kappa^{m+1}, \chi \rangle_{\Gamma^m} = 0 \,, \tag{2.35a}$$

$$\langle \kappa^{m+1} \, \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m} + \langle \nabla_s \vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_{\Gamma^m} = 0 \,, \tag{2.35b}$$

while for *surface diffusion* we analyse the system: Find $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m)$ $\times$ $W(\Gamma^m)$ such that for all $\chi \in W(\Gamma^m)$, $\vec{\eta} \in \underline{V}(\Gamma^m)$

$$\left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi \, \vec{\nu}^m \right\rangle_{\Gamma^m} - \langle \nabla_s \, \kappa^{m+1}, \nabla_s \, \chi \rangle_{\Gamma^m} = 0 \,, \tag{2.36a}$$

$$\langle \kappa^{m+1} \, \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m} + \langle \nabla_s \vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_{\Gamma^m} = 0 \,. \tag{2.36b}$$

We establish existence and uniqueness for the schemes (2.35a)-(2.35b) and (2.36a)-(2.36b) with the help of the following theorem.

THEOREM 2.11. *Let the assumption* ($\mathcal{A}$) *hold. Then there exist unique solutions* $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ *to the systems* (2.35a)-(2.35b) *and* (2.36a)-(2.36b).

Proof. We discuss (2.36a)-(2.36b), since existence and uniqueness for (2.35a)-(2.35b) can be shown in a similar way. The proof is a straightforward adaption of the proof of [12, Thm. 2.1]. Existence and uniqueness of $\{\vec{X}^{m+1}, \kappa^{m+1}\}$ can be shown by proving that the linear system associated at each time level to (2.36a)-(2.36b) is invertible. To this end, we investigate the homogeneous system: Find $\{\vec{\Xi}, \kappa\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ such that

$$\langle \vec{\Xi}, \chi\, \vec{\nu}^m \rangle_{\Gamma^m} - \tau_m \,\langle \nabla_s\, \kappa, \nabla_s\, \chi \rangle_{\Gamma^m} = 0 \qquad \forall\, \chi \in W(\Gamma^m)\,, \tag{2.37a}$$

$$\langle \kappa\, \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m} + \langle \nabla_s\, \vec{\Xi}, \nabla_s\, \vec{\eta} \rangle_{\Gamma^m} = 0 \qquad \forall\, \vec{\eta} \in \underline{V}(\Gamma^m)\,, \tag{2.37b}$$

and prove that the trivial solution is the unique solution to the system. Choosing $\chi \equiv \kappa \in W(\Gamma^m)$ in (2.37a) and $\vec{\eta} \equiv \vec{\Xi} \in \underline{V}(\Gamma^m)$ in (2.37b) we obtain that

$$|\nabla_s\, \vec{\Xi}|^2_{\Gamma^m} + \tau_m\, |\nabla_s\, \kappa|^2_{\Gamma^m} = 0\,. \tag{2.38}$$

It follows from (2.38) that $\kappa \equiv \kappa^c \in \mathbb{R}$ and $\vec{\Xi} \equiv \vec{\Xi}^c \in \mathbb{R}^d$, which yields

$$\langle \vec{\Xi}^c, \chi\, \vec{\nu}^m \rangle_{\Gamma^m} = 0 \quad \forall\, \chi \in W(\Gamma^m), \qquad \kappa^c\, \langle \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m} = 0 \quad \forall\, \vec{\eta} \in \underline{V}(\Gamma^m)\,. \tag{2.39}$$

Since for every element σ_j^m the vectors $\vec{X}^c$ and $\vec{\nu}^m$ are constant functions and χ is linear, the integrand functions in (2.39) are polynomials of degree 1. Therefore it follows that, for all $\chi \in W(\Gamma^m), \vec{\eta} \in \underline{V}(\Gamma^m)$

$$\langle \vec{\Xi}^c, \chi\, \vec{\nu}^m \rangle^h_{\Gamma^m} = \langle \vec{\Xi}^c, \chi\, \vec{\nu}^m \rangle_{\Gamma^m} = 0\,, \quad \kappa^c\, \langle \vec{\nu}^m, \vec{\eta} \rangle^h_{\Gamma^m} = \kappa^c\, \langle \vec{\nu}^m, \vec{\eta} \rangle_{\Gamma^m} = 0\,. \tag{2.40}$$

We now conclude the proof with the same argument used in [12, Thm. 2.1]. By choosing $\vec{\eta} \equiv \vec{z}\, \phi_k^m \in \underline{V}(\Gamma^m)$ in the left-hand side of (2.40) and recalling the definitions of the mass lumped inner product (2.12) and the vertex normal (2.14), we derive, on assuming $\kappa^c \neq 0$, that for $k = 1 \to K_\Gamma^m$

$$\vec{\omega}_k^m \cdot \vec{z} = 0 \quad \forall\, \vec{z} \in \mathbb{R}^d \qquad \Longleftrightarrow \qquad \vec{\omega}_k^m = \vec{0}\,. \tag{2.41}$$

However, this contradicts assumption $(\mathcal{A})$ and hence $\kappa^c = 0$. Similarly, by choosing $\chi \equiv \phi_k^m$ in (2.40) we derive that $\vec{\Xi}^c \cdot \vec{\omega}_k^m = 0$ for $k = 1 \to K_\Gamma^m$. It follows from assumption $(\mathcal{A})$ that $\vec{\Xi}^c \equiv \vec{0}$. Hence we have shown that there exists a unique solution $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ to the linear system (2.36a)-(2.36b). ✓

Furthermore, the schemes (2.35a)-(2.35b) and (2.36a)-(2.36b) are unconditionally stable. This feature is proven in the following theorem.

THEOREM 2.12. *Let the assumption* $(\mathcal{A})$ *hold, and* $\{\vec{X}^m, \kappa^m\}_{m=1}^M$ *be the unique solution to the system* (2.35a)-(2.35b). *Then for* $k = 1 \to M$ *we have that*

$$|\Gamma^k| + \sum_{m=0}^{k-1} \tau_m \, |\kappa^{m+1}|^2_{\Gamma^m} \leqslant |\Gamma^0|\,. \tag{2.42}$$

Moreover, for the unique solution $\{\vec{X}^m, \kappa^m\}_{m=1}^M$ *to the system* (2.36a)-(2.36b), *for* $k = 1 \to M$ *we have that*

$$|\Gamma^k| + \sum_{m=0}^{k-1} \tau_m \, |\nabla_s \, \kappa^{m+1}|^2_{\Gamma^m} \leqslant |\Gamma^0|\,. \tag{2.43}$$

Proof. We prove here only the stability result (2.43), since the result for (2.42) can be shown in a similar way. The proof is a straightforward adaption of the proofs of [10, Thm. 2.3] and [12, Thm. 2.2]. Choosing $\chi \equiv \kappa^{m+1} \in W(\Gamma^m)$ in (2.36a) and $\vec{\eta} \equiv \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m} \in \underline{V}(\Gamma^m)$ in (2.36b) yields that

$$\langle \nabla_s \, \vec{X}^{m+1}, \nabla_s \, (\vec{X}^{m+1} - \vec{X}^m) \rangle_{\Gamma^m} + \tau_m \, |\nabla_s \, \kappa^{m+1}|^2_{\Gamma^m} = 0\,. \tag{2.44}$$

For the case $d = 2$, with the help of [10, Thm. 2.3] and [52], and for the case $d = 3$, with the help of [12, Thm. 2.2], we can show that

$$\langle \nabla_s \, \vec{X}^{m+1}, \nabla_s \, (\vec{X}^{m+1} - \vec{X}^m) \rangle_{\Gamma^m} \geqslant |\Gamma^{m+1}| - |\Gamma^m|\,. \tag{2.45}$$

Combining (2.44) and (2.45) yields that

$$|\Gamma^{m+1}| - |\Gamma^m| + \tau_m \, |\nabla_s \, \kappa^{m+1}|^2_{\Gamma^m} \leqslant 0\,. \tag{2.46}$$

Summing (2.46) for $m = 0 \to k - 1$ yields the desired result (2.43). ✓

REMARK 2.13. *The choice of the exact quadrature formula* (2.11) *instead of the mass lumped inner product* (2.12) *means that the analogue of the equidistribution property investigated in Theorem 2.8 and Remarks 2.9 and 2.10 cannot be established. We will therefore conduct a set of numerical experiments, for both* $d = 2$ *and* $d = 3$, *in order to investigate the behaviour of mesh quality indicators over time, when* (2.11) *is employed.*

Our first experiment for $d = 2$ is inspired by [10, Fig. 4]. The initial geometry is given by the same $3 : 1$ ellipse, with $J^m_\Gamma = K^m_\Gamma = 100$. The

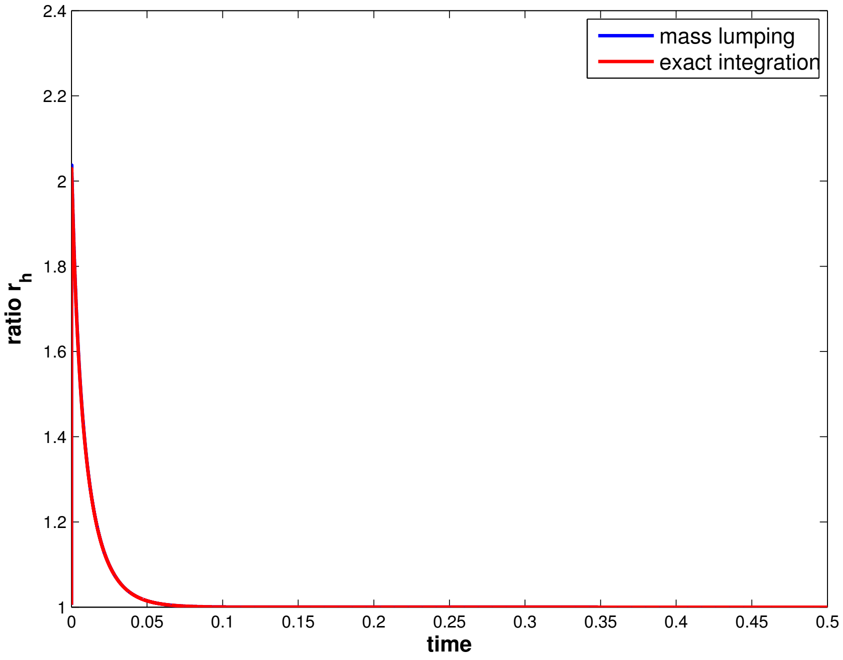

Figure 2.4: Plots of the ratio r_h for both mass lumping (blue) and exact integration (red), with initial domain given by a 3 : 1 ellipse. The two curves are indistinguishable.

simulation parameters are chosen as $\tau = 10^{-5}$ and $T = 0.5$. In Figure 2.4 we plot the behaviour of the ratio r_h over time. We note that both mass lumping (blue) and exact integration (read) lead to an equidistribution of mesh vertices. The plots for the two choices are almost indistinguishable.

The next experiment corresponds to [10, Fig. 5], with $\tau = 10^{-4}$ and $T = 5$. In Figure 2.5 we plot the behaviour of the ratio r_h over time. We note that also in this case the equidistribution of mesh points is reached in practice, and the temporal behaviours of r_h for the two inner products are almost indistinguishable.

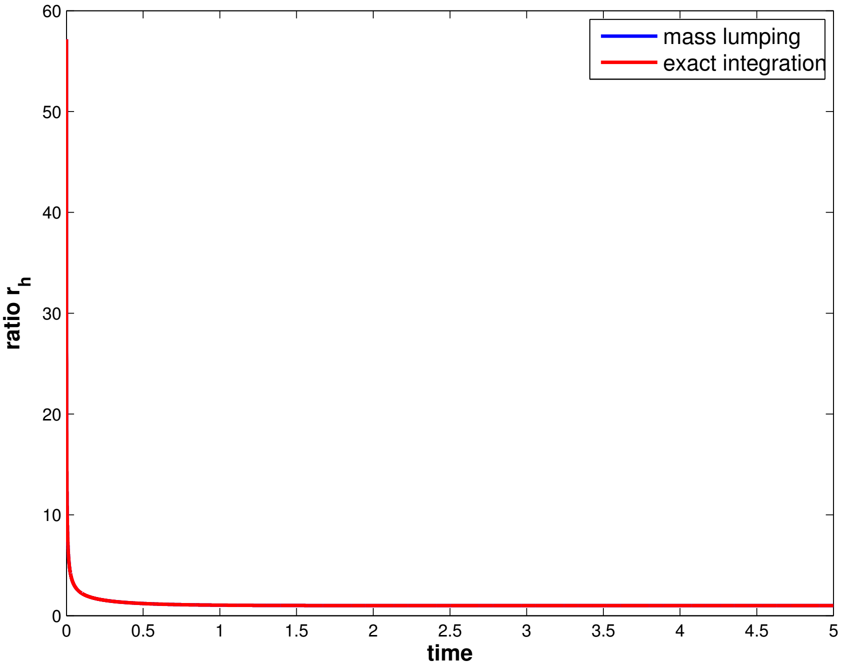

Figure 2.5: Plots of the ratio r_h for both mass lumping (blue) and exact integration (red), with initial domain given by a semi-circle and a single additional node on the periphery of the circle. The two curves are indistinguishable.

Our last example for $d = 2$ analyses the unit circle, with $J_\Gamma^m = K_\Gamma^m = 128$. At $t = 0$ the points on either semi-circle are placed such that the arcs connecting pairs of consecutive vertices constitute a geometric progression with ratio 1.25. The initial geometry is shown in Figure 2.6. From Figure 2.7 we note that the mesh points eventually equidistribute for both mass lumping and exact integration.

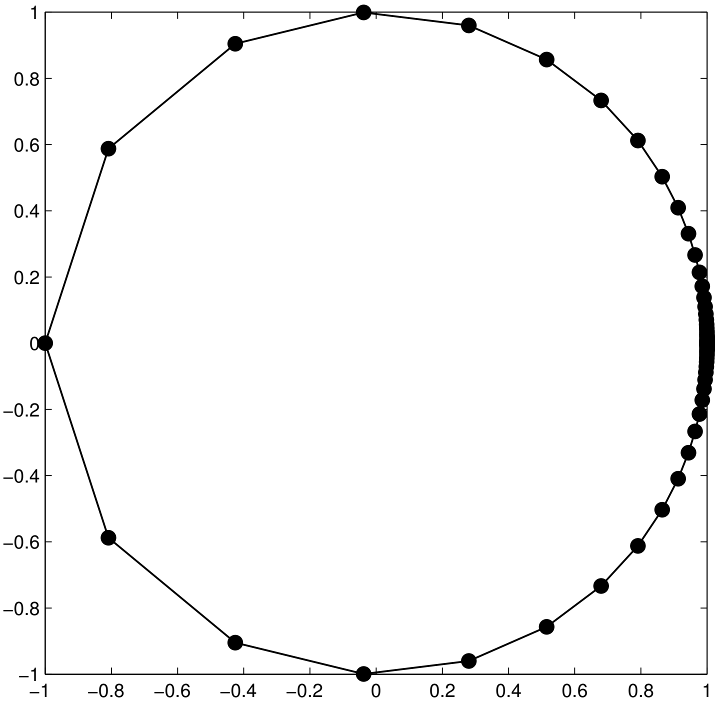

Figure 2.6: Plot at time $t = 0$ of the unit circle with 128 points, placed such that on either semi-circle the arcs connecting pairs of consecutive vertices constitute a geometric progression with ratio 1.25.

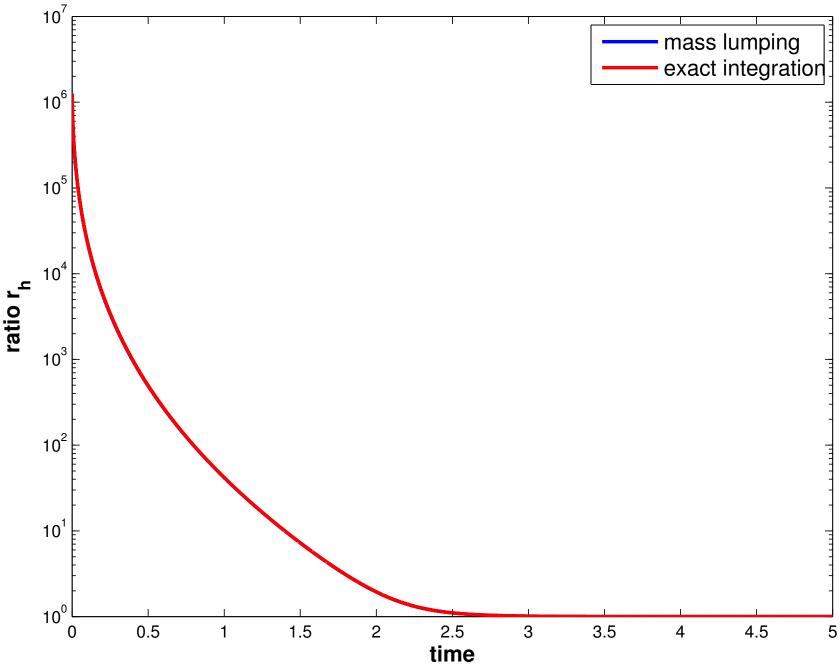

Figure 2.7: Plots of the ratio r_h for both mass lumping (blue) and exact integration (red), with initial domain given by a circle with 128 points, placed such that the arcs connecting pairs of consecutive vertices constitute a geometric series with ratio 1.25. The two curves are indistinguishable.

Our first experiment for $d = 3$ corresponds to the one recalled from [12, Fig. 2]. We note that the initial $2 \times 1 \times 1$ cuboid is triangulated with right-angled, isosceles triangles, with $K_\Gamma^m = 1282$ and $J_\Gamma^m = 2560$. We plot its temporal evolution under *mean curvature flow* with mass lumping ((2.16a)-(2.16b)) in Figure 2.8. In Figure 2.9 we plot the ratio r_a, defined in (2.34), for mass lumping ((2.16a)-(2.16b), blue) and exact integration ((2.35a)-(2.35b), red). We observe that mass lumping contributes to keeping a more regular mesh. Moreover, the ratio r_a is higher when exact integration is used. We also analyse the case of a different triangulation of the same cuboidal domain. To this end, we employ the mesh generator GMSH (see [73] for details), which produces high-quality Delaunay triangulations. We obtain an initial mesh with size comparable to the previous case, precisely with $K_\Gamma^m = 1284$ and $J_\Gamma^m = 2564$. We plot this Delaunay mesh in Figure 2.10. The plots for the ratio r_a for both mass lumping and true integration are shown in Figure 2.11. We again observe that the choice of the inner product (2.12) contributes to keeping a more regular mesh. In addition, the ratio r_a tends to be higher for the geometry produced by the package GMSH than in the case of right-angled, isosceles triangles.

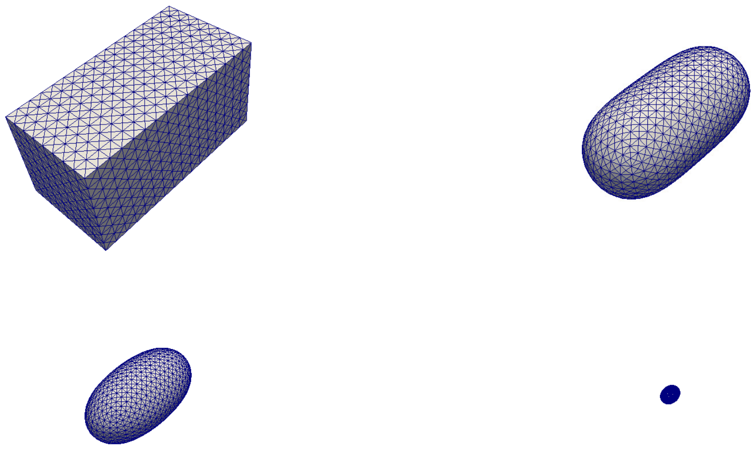

Figure 2.8: Evolution of a $2 \times 1 \times 1$ cuboid under *mean curvature flow* with mass lumping ((2.16a)-(2.16b)), at times $t = 0, 0.05, 0.1, T = 0.14$. At $t = T$ the enclosed volume is reduced by 99.74%.

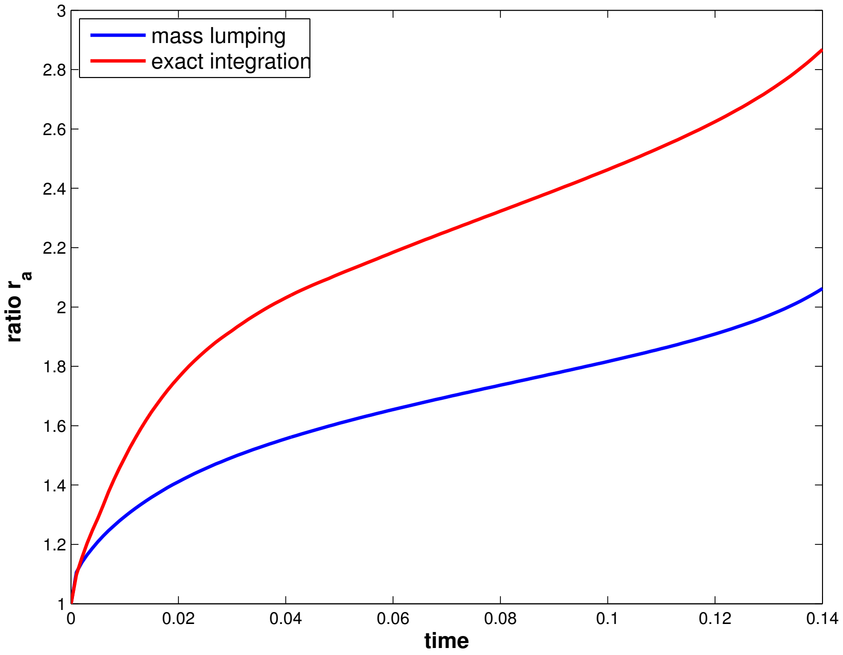

Figure 2.9: Plots of the ratio r_a (eq. (2.34)) for both mass lumping (blue) and exact integration (red), with initial domain given by a $2 \times 1 \times 1$ cuboid triangulated with right-angled, isosceles triangles.

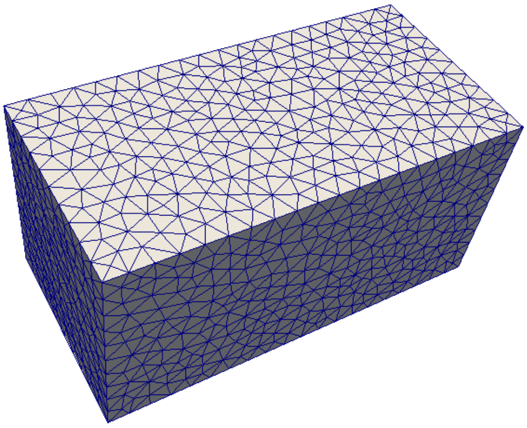

Figure 2.10: Plot at time $t = 0$ of the $2 \times 1 \times 1$ cuboid triangulated by the package GMSH.

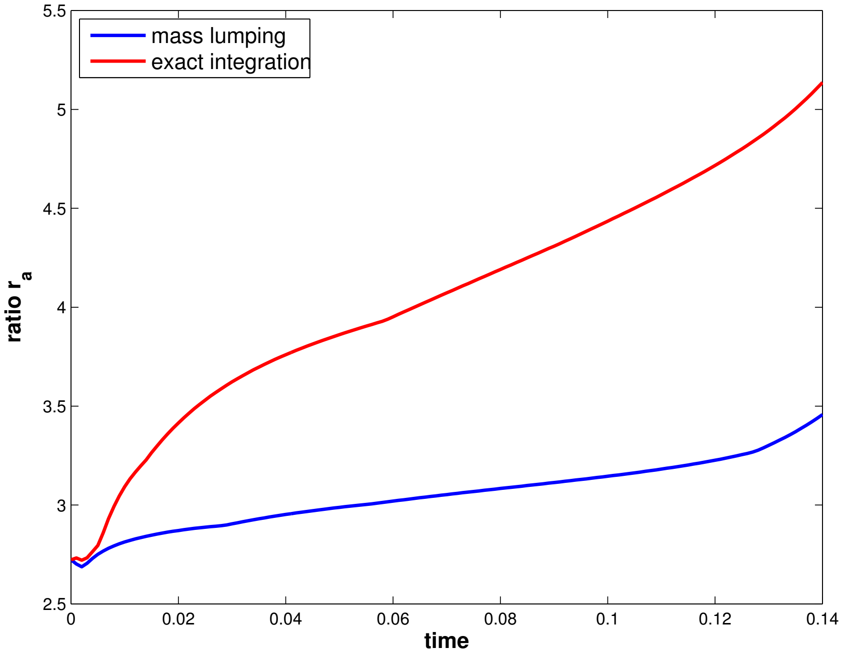

Figure 2.11: Plots of the ratio r_a (eq. (2.34)) for both mass lumping (blue) and exact integration (red), with initial domain given by a $2 \times 1 \times 1$ cuboid triangulated by the package GMSH.

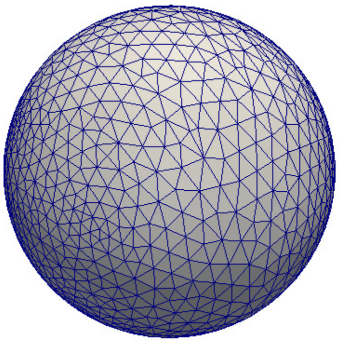

Figure 2.12: Plot at time $t = 10$ of the $2 \times 1 \times 1$ cuboid, evolving under *surface diffusion* with mass lumping ((2.17a)-(2.17b)) and triangulation given by the package GMSH. The cuboid has asymptotically converged to a sphere.

Our next experiment is for *surface diffusion* and involves the same $2 \times 1 \times 1$ cuboid analysed in the first experiment. We choose the parameters $\tau = 10^{-3}$ and $T = 10$. We note that the cuboid converges asymptotically to a sphere, see Figure 2.12 for the case where the initial mesh is triangulated by the package GMSH and mass lumping ((2.17a)-(2.17b)) is used. We do not report the plot for the case of an initial mesh with right-angled, isosceles triangles since it looks very similar. We plot the ratio r_a for the mesh with right-angled, isosceles triangles in Figure 2.13 and for the mesh produced by GMSH in Figure 2.14, respectively. The mesh quality indicators for the two cases are almost indistinguishable.

Our last set of experiments is inspired by the test in [12, Fig. 15]. We investigate the evolution of a "cage" under *surface diffusion*. The dimensions of the initial surface are $4 \times 4 \times 4$, with the region enclosed by Γ^0 given as the union of 12 cuboids of dimension $4 \times 1 \times 1$. Here a topological change is encountered when the six holes of the surface are about to close to form a hollow ball. We choose $\tau = 10^{-4}$ and $T = 0.48$. We compare two different cases. In the first one, an initial mesh with right-angled and isosceles triangles is considered. The discretisation parameters are $K_\Gamma^m = 1912$ and $J_\Gamma^m = 3840$. In the second case, we consider the same initial domain tri-

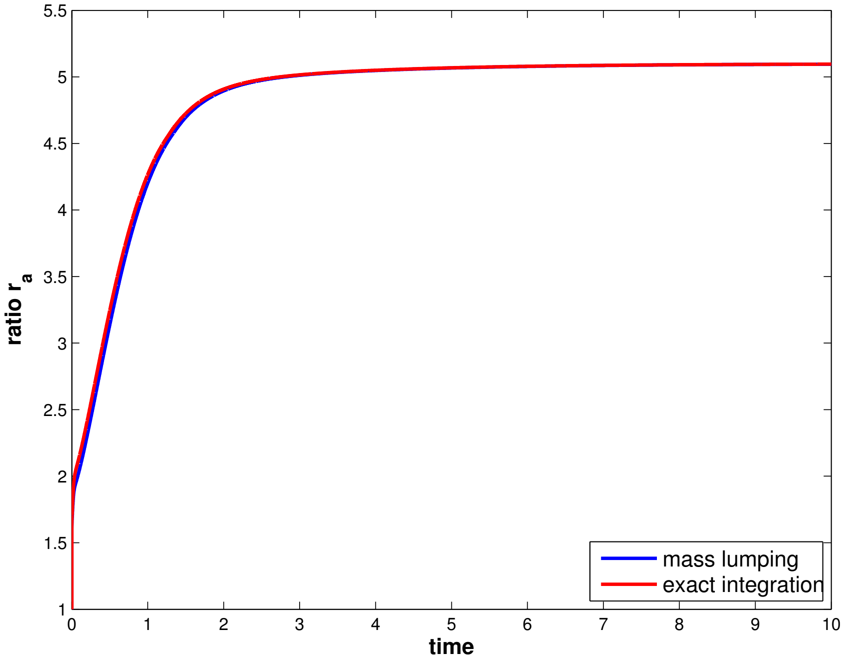

Figure 2.13: Plots of the ratio r_a (eq. (2.34)) for both mass lumping (blue) and exact integration (red), with initial domain given by a $2 \times 1 \times 1$ cuboid triangulated with right-angled, isosceles triangles.

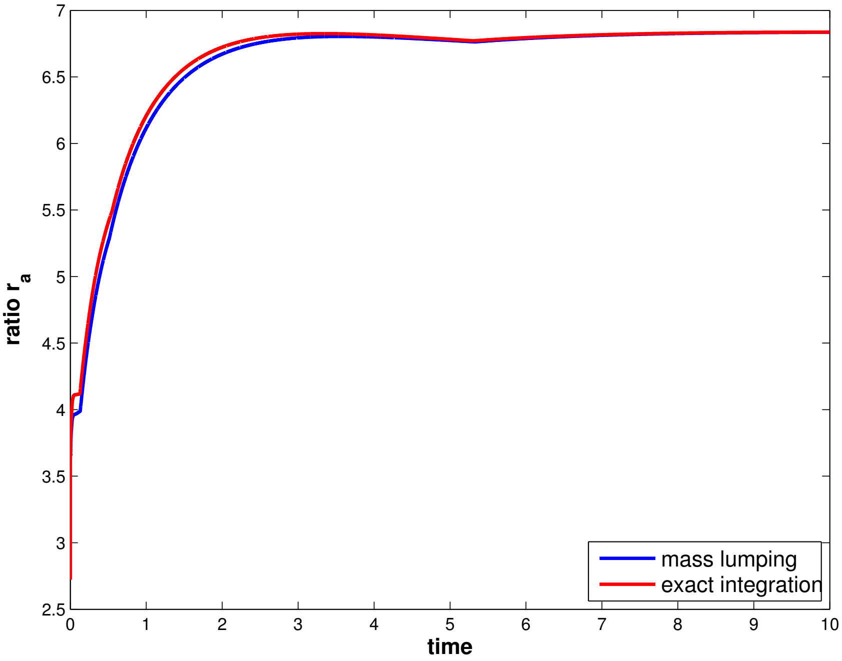

Figure 2.14: Plots of the ratio r_a (eq. (2.34)) for both mass lumping (blue) and exact integration (red), with initial domain given by a $2 \times 1 \times 1$ cuboid triangulated by the package GMSH.

angulated with GMSH. The discretisation parameters are $K_\Gamma^m = 1900$ and $J_\Gamma^m = 3816$. The temporal evolution for the latter set of parameters is shown in Figure 2.15. We do not report the evolution for the former case, since it looks very similar. The plot for the ratio r_a for the first case is shown in Figure 2.16, both for mass lumping (blue) and exact integration (red). The results for the second case are shown in Figure 2.17. We notice that for the "cage" surface exact integration keeps the evolving mesh smoother. Only

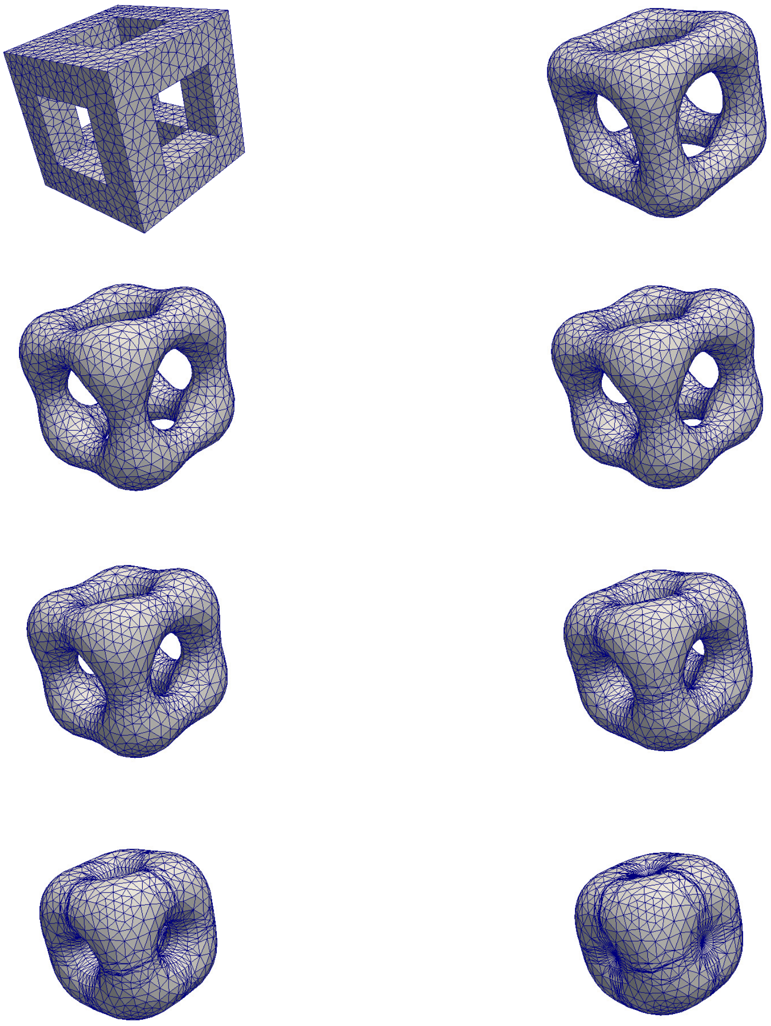

Figure 2.15: Evolution of a $4 \times 4 \times 4$ "cage" under *surface diffusion* with mass lumping ((2.17a)-(2.17b)), at times $t = 0, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, T = 0.48$. The initial mesh is produced by the package GMSH.

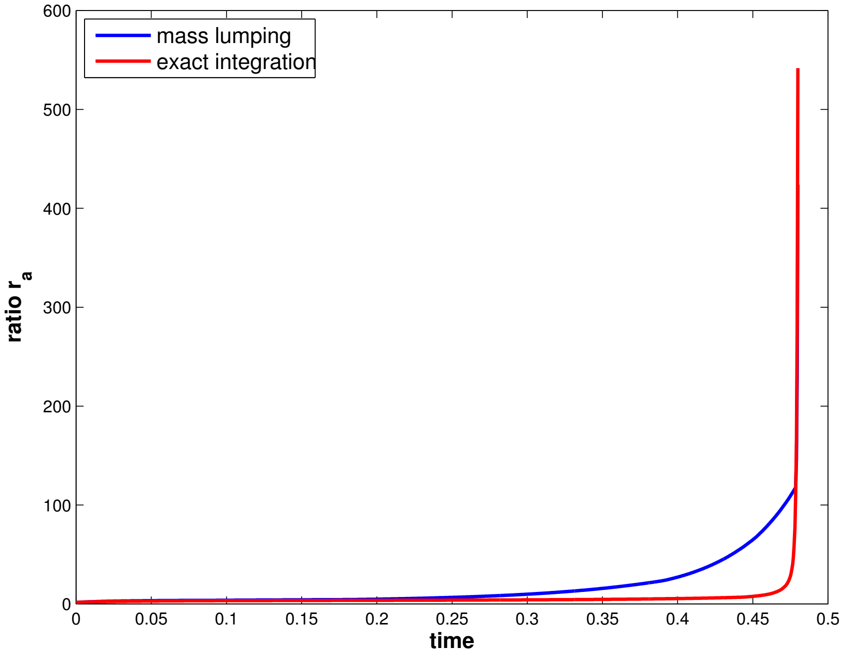

Figure 2.16: Plots of the ratio r_a (eq. (2.34)) for both mass lumping (blue) and exact integration (red), with initial domain given by a $4 \times 4 \times 4$ "cage" triangulated with right-angled, isosceles triangles.

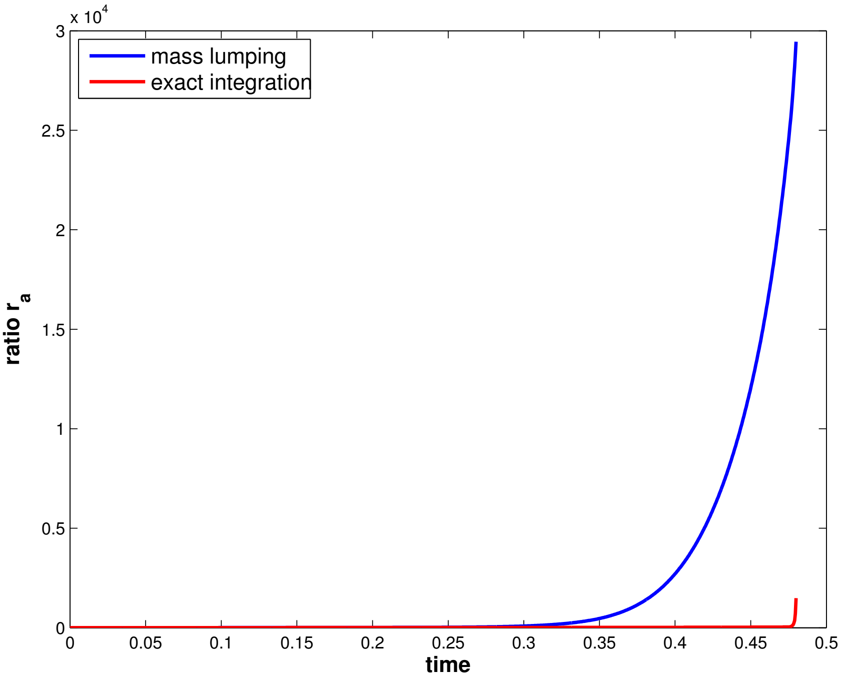

Figure 2.17: Plots of the ratio r_a (eq. (2.34)) for both mass lumping (blue) and exact integration (red), with initial domain given by a $4 \times 4 \times 4$ "cage" triangulated by the package GMSH.

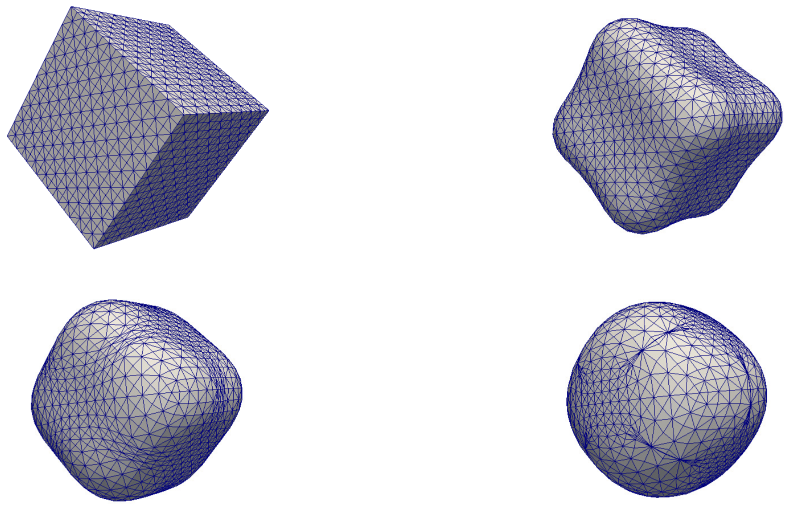

Figure 2.18: Evolution of the unit cube under *surface diffusion* with mass lumping ((2.17a)-(2.17b)), at times $t = 0$, 10^{-4}, 5×10^{-4}, 10^{-3}. The initial mesh is composed of right-angled, isosceles triangles, and the time step is $\tau = 10^{-7}$.

very close to $t = T$, when the surface is about to undergo a topological change, does the ratio r_a increase for exact integration as well.

2.2.4 *Evolution under very small time steps: the α-scheme*

We now go back to the scheme (2.17a)-(2.17b) and investigate the temporal evolution of surfaces when very small time steps are considered. We take as initial geometry the unit cube, where the interface mesh is composed of right-angled, isosceles triangles. The discretisation parameters are the following: $\tau = 10^{-7}$, $T = 10^{-3}$, $K_\Gamma^m = 1538$ and $J_\Gamma^m = 3072$. The evolution is plotted in Figure 2.18. We can see that distorted elements appear already for $t = 5 \times 10^{-4}$, and the mesh keeps deteriorating with time. To further investigate the quality of the mesh, we plot the temporal evolution of the quality indicator r_a in Figure 2.19. We note that the increase for r_a is caused by the "rings" of elongated elements that we already noticed for the "cage" in Figure 2.15.

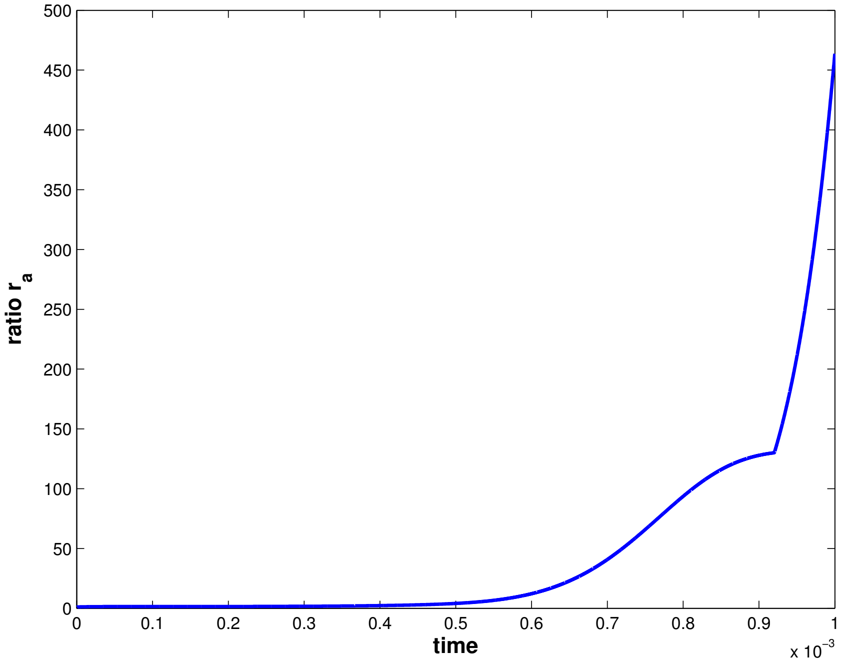

Figure 2.19: Plots of the ratio r_a (eq. (2.34)) for mass lumping, with initial domain given by the unit cube triangulated with right-angled, isosceles triangles, and time step $\tau = 10^{-7}$.

In order to eliminate the spurious "rings" appearing on the interface mesh, we now apply a different strategy, which aims at reducing the tangential movement for the discrete parameterisation. Precisely, we solve for an alternative of (2.17b), where the idea is to represent also the tangential components of the discrete surface Laplacian of $\vec{X}^{m+1}$. The scheme was first introduced in [13], and we recall it here for the benefit of the reader. First, we define the function $\vec{z}^m \in \underline{V}(\Gamma^m)$ such that

$$\vec{z}^m(\vec{q}_k^m) := \frac{\vec{\omega}_k^m}{|\vec{\omega}_k^m|}, \qquad k = 1 \to K_\Gamma^m, \tag{2.47}$$

where we recall the definition of $\vec{\omega}_k^m$ given in (2.14). The vector-valued function $\vec{z}^m$ is, vertex-wise, a unit normal vector.

Given the above definition, we now introduce the vector-valued functions $\vec{\tau}_i^m := \sum_{k=1}^{K_\Gamma^m} \vec{\tau}_{i,k}^m \phi_k^m \in \underline{V}(\Gamma^m)$, where $\{\vec{z}^m(\vec{q}_k^m), \vec{\tau}_{1,k}^m, \dots, \vec{\tau}_{d-1,k}^m\}$ for each vertex $k = 1, \dots, K_\Gamma^m$ form an orthonormal basis of $\mathbb{R}^d$. We now illustrate a variant of (2.17b) that reduces tangential motion. Let the coefficient $\alpha \in \mathbb{R}_{\geqslant 0}$ be given. In addition to the position vector $\vec{X}^{m+1}$ and to the curvature κ^{m+1}, find $\beta_i^{m+1} \in W(\Gamma^m), i = 1 \to d-1$, such that $\forall\, \chi \in W(\Gamma^m)$, $\forall\, \vec{\eta} \in \underline{V}(\Gamma^m)$

$$\alpha \left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi\, \vec{\tau}_i^m \right\rangle_{\Gamma^m}^h = \alpha \, \langle \beta_i^{m+1}, \chi \rangle_{\Gamma^m}^h, \quad i = 1 \to d-1, \tag{2.48a}$$

$$\left\langle \kappa^{m+1}\, \vec{\omega}^m + \alpha \sum_{i=1}^{d-1} \beta_i^{m+1} \vec{\tau}_i^m, \vec{\eta} \right\rangle_{\Gamma^m}^h + \langle \nabla_s \vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_{\Gamma^m} = 0, \tag{2.48b}$$

and (2.17a) hold. It is straightforward to note that, choosing $\alpha \equiv 0$, the scheme (2.17a), (2.48a)-(2.48b) collapses to the scheme (2.17a)-(2.17b), on recalling (2.15). We now establish existence and uniqueness for the scheme (2.17a), (2.48a)-(2.48b) with the help of the following theorem.

THEOREM 2.14. *Let the assumption* $(\mathcal{A})$ *hold. Then there exist unique solutions* $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ *and* $\beta_i^{m+1} \in W(\Gamma^m)$, $i = 1 \to d-1$, *to the system* (2.17a), (2.48a)-(2.48b).

Proof. The proof is a straightforward adaption of the one given in [13, Thm. 3.1]. Since (2.17a), (2.48a)-(2.48b) is a linear system of equations, it suffices to analyse it for the unknowns $\{\vec{\Xi}, \kappa\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$, $\beta_i \in W(\Gamma^m)$, $i = 1 \to d-1$, with homogeneous right-hand side and deduce that the trivial solution is the only solution of such a system. With the particular choice of test functions $\chi \equiv \kappa \in W(\Gamma^m)$ in the analogue of (2.17a), $\vec{\eta} \equiv \vec{\Xi} \in \underline{V}(\Gamma^m)$ in the analogue of (2.48a) and $\chi \equiv \beta_i \in W(\Gamma^m)$, $i = 1 \to d-1$, in the analogue of (2.48b), we obtain

$$|\nabla_s \vec{\Xi}|^2_{\Gamma^m} + \tau_m |\nabla_s \kappa|^2_{\Gamma^m} + \alpha \sum_{i=1}^{d-1} \langle \beta_i, \beta_i \rangle^h_{\Gamma^m} = 0 \,. \tag{2.49}$$

It immediately follows from (2.49) that $\beta_i = 0$, $i = 1 \to d-1$. We can now continue the proof similarly to the proof of Theorem 2.11 to obtain $\vec{\Xi} = \vec{0}$ and $\kappa = 0$. Hence there exist unique solutions $\{\vec{X}^{m+1}, \kappa^{m+1}\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ and $\beta_i^{m+1} \in W(\Gamma^m)$, $i = 1 \to d-1$, to the system (2.17a), (2.48a)-(2.48b). ✓

We now perform several numerical simulations in order to test whether the appearance of elongated elements can be eliminated. We keep the same parameters as before, i.e. $\tau = 10^{-7}$ and the initial mesh composed of right-angled, isosceles triangles, with $K^m_\Gamma = 1538$ and $J^m_\Gamma = 3072$. Results for the mesh quality indicator r_a are plotted in Figures 2.20 and 2.21 for different values of α.

We can observe that the quality of the mesh strongly depends on the choice of the parameter α. With $\alpha = 0.1$, tangential movement is reduced too much, which causes the ratio r_a to increase higher than in the case shown in Figure 2.19. Decreasing the value of α improves the quality of the mesh, which reaches a steady state. In particular, it appears that the choice of $\alpha = 10^{-4}$ best overcomes the "ringing" effect shown in Figure 2.18. The complete evolution of the cube according to the scheme (2.17a), (2.48a)-(2.48b) with $\alpha = 10^{-4}$ is reported in Figure 2.22. Finally, we note that, taking $\alpha \to 0$, the quality of the mesh deteriorates again, and the ratio r_a exhibits a behaviour similar to the one in Figure 2.19.

2.3 MULTI-COMPONENT INTERFACES

For ease of presentation, so far we have restricted ourselves to the case of Γ^m being a single closed surface. It is straightforward to extend the approximations (2.16a)-(2.16b) and (2.17a)-(2.17b) to the case where Γ^m is given by

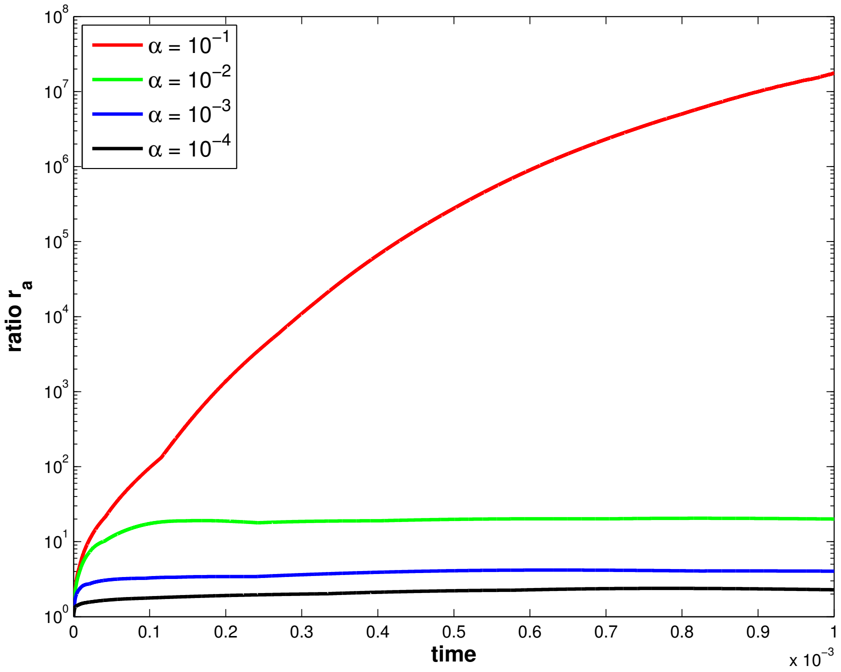

Figure 2.20: Plots of the ratio r_a (eq. (2.34)) for the unit cube under the scheme (2.17a), (2.48a)-(2.48b), with $\tau = 10^{-7}$ and $\alpha = 10^{-1}$ (red), 10^{-2} (green), 10^{-3} (blue), 10^{-4} (black).

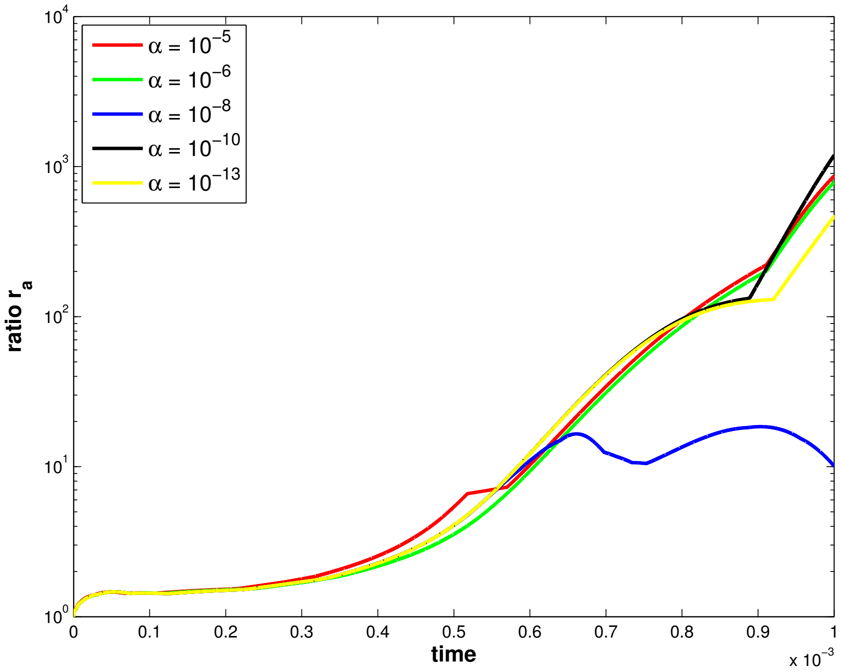

Figure 2.21: Plots of the ratio r_a (eq. (2.34)) for the unit cube under the scheme (2.17a), (2.48a)-(2.48b), with $\tau = 10^{-7}$ and $\alpha = 10^{-5}$ (red), 10^{-6} (green), 10^{-8} (blue), 10^{-10} (black), 10^{-13} (yellow).

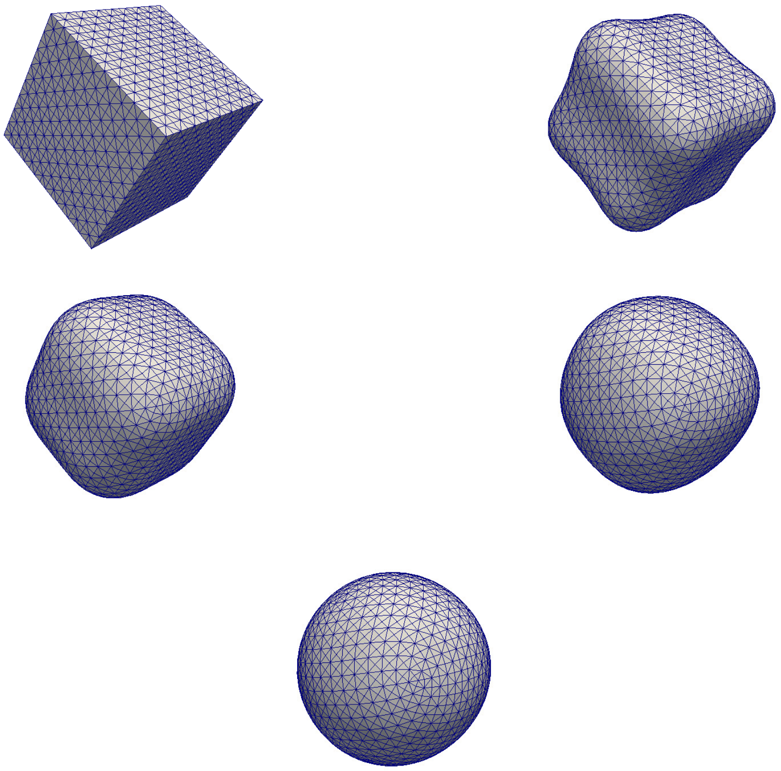

Figure 2.22: Evolution of the unit cube under the schemes (2.17a), (2.48a)-(2.48b), with $\alpha = 10^{-4}$ and $\tau = 10^{-7}$. The evolving surface is shown at times $t = 0$, 10^{-4}, 5×10^{-4}, 10^{-3}, 2×10^{-3}.

a family of closed surfaces, say $\Gamma^m = \bigcup_{l=1}^{L} \Gamma_l^m$, with $\Gamma_l^m = \bigcup_{j=1}^{J_l^m} \overline{\sigma_j^{m,l}}$, where $\{\sigma_j^{m,l}\}_{j=1}^{J_l^m}$ is a family of mutually disjoint open $(d-1)$-simplices with vertices $\{\vec{q}_k^{m,l}\}_{k=1}^{K_l^m}$, $l = 1, \ldots, L$. We also define the necessary parametric finite element spaces as follows. For $m = 0 \to M-1$, let

$$\underline{V}^l(\Gamma^m) := \{\vec{\chi} \in C(\Gamma_l^m, \mathbb{R}^d) : \vec{\chi}\,|_{\sigma_j^{m,l}} \text{ is linear } \forall\, j = 1 \to J_l^m\} =: \left[W^l(\Gamma^m)\right]^d . \tag{2.50}$$

The natural extensions of (2.16a)-(2.16b) to the multi-component case can be presented as follows: Given Γ_l^0 and the identity function $\vec{X}_l^0 \in \underline{V}^l(\Gamma^0)$ on Γ_l^0, then for $m = 0 \to M-1$ find $\{\vec{X}_l^{m+1}, \kappa_l^{m+1}\} \in \underline{V}^l(\Gamma_l^m) \times W^l(\Gamma_l^m)$ such that, for $l = 1 \to L$

$$\left\langle \frac{\vec{X}_l^{m+1} - \vec{X}_l^m}{\tau_m}, \chi\,\vec{\nu}_l^m \right\rangle_{\Gamma_l^m}^h - \langle \kappa_l^{m+1}, \chi \rangle_{\Gamma_l^m}^h = 0 \qquad \forall\, \chi \in W^l(\Gamma_l^m)\,, \tag{2.51a}$$

$$\langle \kappa_l^{m+1}\,\vec{\nu}_l^m, \vec{\eta} \rangle_{\Gamma_l^m}^h + \langle \nabla_s\,\vec{X}_l^{m+1}, \nabla_s\,\vec{\eta} \rangle_{\Gamma_l^m} = 0 \qquad \forall\, \vec{\eta} \in \underline{V}^l(\Gamma_l^m)\,. \tag{2.51b}$$

Moreover, the natural extensions of (2.17a)-(2.17b) to the multi-component case can be presented as follows: Given Γ_l^0 and the identity function $\vec{X}_l^0 \in \underline{V}^l(\Gamma^0)$ on Γ_l^0 for $l = 1, \ldots, L$, then for $m = 0 \to M-1$ find $\{\vec{X}_l^{m+1}, \kappa_l^{m+1}\} \in \underline{V}^l(\Gamma_l^m) \times W^l(\Gamma_l^m)$ such that, for $l = 1 \to L$

$$\left\langle \frac{\vec{X}_l^{m+1} - \vec{X}_l^m}{\tau_m}, \chi\,\vec{\nu}_l^m \right\rangle_{\Gamma_l^m}^h - \langle \nabla_s\,\kappa_l^{m+1}, \nabla_s\,\chi \rangle_{\Gamma_l^m} = 0 \quad \forall\, \chi \in W^l(\Gamma_l^m)\,, \tag{2.52a}$$

$$\langle \kappa_l^{m+1}\,\vec{\nu}_l^m, \vec{\eta} \rangle_{\Gamma_l^m}^h + \langle \nabla_s\,\vec{X}_l^{m+1}, \nabla_s\,\vec{\eta} \rangle_{\Gamma_l^m} = 0 \quad \forall\, \vec{\eta} \in \underline{V}^l(\Gamma_l^m)\,. \tag{2.52b}$$

Let $(\mathcal{A}_l)$ be the analogue of assumption $(\mathcal{A})$ for the closed surface Γ_l^m, $l = 1, \ldots, L$. The we can generalise Theorems 2.6 and 2.7 as follows.

Theorem 2.15. *Let the assumptions* $(\mathcal{A}_l)$, $l = 1, \ldots, L$ *hold. Then there exist unique solutions* $\{\vec{X}_l^{m+1}, \kappa_l^{m+1}\} \in \underline{V}^l(\Gamma^m) \times W^l(\Gamma^m)$, $l = 1 \to L$, *to the systems* (2.51a)-(2.51b) *and* (2.52a)-(2.52b). *Moreover, for mean curvature flow it holds that*

$$|\Gamma^k| + \sum_{m=0}^{k-1} \tau_m \sum_{l=1}^{L} (|\kappa_l^{m+1}|_{\Gamma_l^m}^h)^2 \leqslant |\Gamma^0|\,, \tag{2.53}$$

while for surface diffusion

$$|\Gamma^k| + \sum_{m=0}^{k-1} \tau_m \sum_{l=1}^{L} |\nabla_s \, \kappa_l^{m+1}|^2_{\Gamma_l^m} \leqslant |\Gamma^0| \,. \tag{2.54}$$

Proof. As the systems for $\{\vec{X}_l^{m+1}, \kappa_l^{m+1}\}$ decouple for each $l = 1, \ldots, L$, we immediately obtain the desired existence and uniqueness results from Theorem 2.6. Moreover, in order to prove the stability result (2.54), we note that

$$|\Gamma_l^{m+1}| + \tau_m \, |\nabla_s \, \kappa_l^{m+1}|^2_{\Gamma_l^m} \leqslant |\Gamma_l^m| \,,$$

for $l = 1, \ldots, L$, which on summing from $m = 0 \to k-1$ and for $l = 1 \to L$ yields the desired result (2.54). (2.53) can be proven with a straightforward adaption. ✓

Remark 2.16. *For a semi-discrete variant of* (2.52a)-(2.52b)*, as for* (2.20a)-(2.20b)*, we can show that* (2.21) *holds for each* Γ_l, $l = 1, \ldots, L$. *Moreover, for the case* $d = 2$*, the equidistribution property now holds for each curve* $\Gamma_l^h(t)$ *separately. In fact, on letting* $\vec{h}_k^l(t) := \vec{q}_k^{h,l} - \vec{q}_{k-1}^{h,l}$ *for* $k = 1, \ldots, K_{\Gamma_l^h}^h$ *and* $l = 1, \ldots, L$*, where* $\{\vec{q}_k^{h,l}\}_{k=1}^{K_{\Gamma_l}^h}$ *are the vertices along* $\Gamma_l^h(t)$*, we obtain*

$$|\vec{h}_k^l(t)| = |\vec{h}_{k-1}^l(t)| \quad \textit{if} \quad \vec{h}_k^l(t) \nparallel \vec{h}_{k-1}^l(t) \,. \tag{2.55}$$

2.3.1 *Topological changes*

By extending the schemes (2.16a)-(2.16b) and (2.17a)-(2.17b) to the multi-component cases (2.51a)-(2.51b) and (2.52a)-(2.52b), we have obtained the possibility of simulating the temporal evolution of a family of hypersurfaces. This feature also allows us to perform topological changes, such as the merging of two curves into one, or the pinching-off of one surface from another. In practice we monitor the need for topological changes, and then implement the changes, with the help of the package El-Topo, see [30]. In that paper, the authors require every operation to leave the mesh in a consistent, nonintersecting state – as opposed to attempting to recover such a state after an intersection has occurred. To this end, the authors present a combination of different routines, which can be classified into three main types of geometric interference detection: *intersection* detection, *proximity* detection, and *collision* detection. The first operation aims at checking whether a mesh intersects itself at any given time. The second operation detects when mesh elements are closer than a specified tolerance (in particular, when a vertex is close to a triangle or when two edges are close to each other). The third operation tests whether a collision between a moving vertex and a moving triangle or between two moving edges will occur in a specified time span, and resolves such a collision guaranteeing intersection-free meshes. The same library has successfully been applied in several contexts: coupling of an explicit surface tracker to a Voronoi simulation mesh ([29]), linear-time smoke animations with vortex sheet meshes ([31]), evolution of multi-material interfaces ([39]), and ocean-based animations using boundary integral equations ([89]).

Our usage of this library can be summarised as follows. Let $\mathbf{T}^m$ be the mesh configuration at time $t = t_m$, consisting of (i) the current positions of the vertices of Γ^m and (ii) the connectivity of the grid in terms of pairs (for $d = 2$) or triplets ($d = 3$) of vertex indices. As input parameters El-Topo expects the current mesh configuration $\mathbf{T}^m$ as well as a user-defined velocity function, which we naturally choose to be $\frac{1}{\tau_m}\,\delta\vec{X}^{m+1}$, and the time step size τ_m. This allows El-Topo to define a predicted mesh configuration $\widehat{\mathbf{T}}^{m+1}$, where each vertex is transported with its own velocity, and where no topological changes are performed. If $\widehat{\mathbf{T}}^{m+1}$ satisfies certain criteria, then topological changes are performed to produce a final mesh configuration $\mathbf{T}^{m+1}$. For example, if a component of $\widehat{\mathbf{T}}^{m+1}$ features self-intersections, then $\widehat{\mathbf{T}}^{m+1}$ is split up such that the self-intersection is removed, and $\mathbf{T}^{m+1}$ still remains as close as possible to $\widehat{\mathbf{T}}^{m+1}$. Moreover, the user can specify tolerances that indicate that different components of $\widehat{\mathbf{T}}^{m+1}$ have come too close to each other, which then results in them being merged. Finally, one can also specify a minimum grid size, which leads to elements below a certain tolerance being removed altogether. Due to the heuristic nature of the employed criteria, it is always necessary to experimentally change the various tolerances according to the analysed hypersurface, depending on the number of its points and elements, time step, and preservation of the enclosed volume. For more details on how El-Topo proceeds to determine the necessary topological changes, we refer to the description in [30, § 3.3].

We now show some examples of the application of the El-Topo package for surfaces evolving under *surface diffusion*, for both $d = 2$ and $d = 3$. The first example is for $d = 2$, with the initial geometry given by two ellipses with horizontal semiaxis $a = 2.4$ and vertical semiaxis $b = 0.3$, centred at $(0, \pm 0.33)$ and with 316 elements on each ellipse. The simulation is divided into three parts. The first one has time step $\tau = 10^{-6}$ and $T_1 = 8 \times 10^{-3}$, where no topological changes are applied. At $t = T_1$, the two ellipses, which now present a dumbbell-shaped profile, coalesce. The two points of coalescence exhibit a very high value for the curvature. In order to capture the evolution correctly and avoid volume loss, we continue the simulation with a very small time step, namely $\tau = 10^{-10}$, until $T_2 = 8.0031 \times 10^{-3}$. At time $t = T_2$, the shape of the geometry has been smoothed, so we can reuse the time step $\tau = 10^{-6}$ until $T_3 = 6$. The complete evolution is shown in Figure 2.23, where we note that the initial elliptical profiles evolve first into a dumbbell-shaped one, and then undergo a topological change which results in a final configuration of two circles.

Our second example is for $d = 3$, and investigates the evolution of the "cage" already presented in Figure 2.15. The simulation is again divided into three parts. For the first part we set $\tau = 10^{-4}$ and $T_1 = 0.48$. At this point, El-Topo first applies a *mesh smoothing* routine that removes the "rings" of distorted elements, and then performs a topological change leading to a hollow sphere, as plotted in Figure 2.24. We experimentally note that the smoothing routine is necessary in order to obtain the correct topological change. To capture the evolution correctly, in the second part of the simulation we reduce the time step to $\tau = 10^{-7}$ until $T_2 = 0.4802$, from which we reuse $\tau = 1 \times 10^{-4}$ until $T_3 = 0.8$.

Our next example is inspired by the tests in [12, Figs. 21 and 22]. The initial configuration is a "cigar like" rounded cylinder of dimension $8 \times 1 \times 1$, which is reported in Figure 2.25. The simulation is again divided into three parts. For the first part we set $\tau = 10^{-4}$ and $T_1 = 0.237$. At this point,

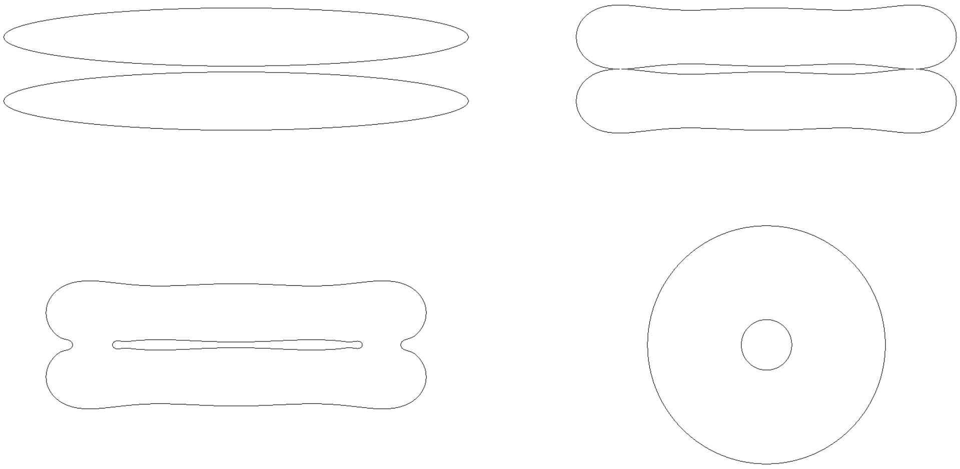

Figure 2.23: Evolution of two 8 : 1 ellipses under *surface diffusion*, at times $t = 0$, 8×10^{-3}, 8.0031×10^{-3}, 6.

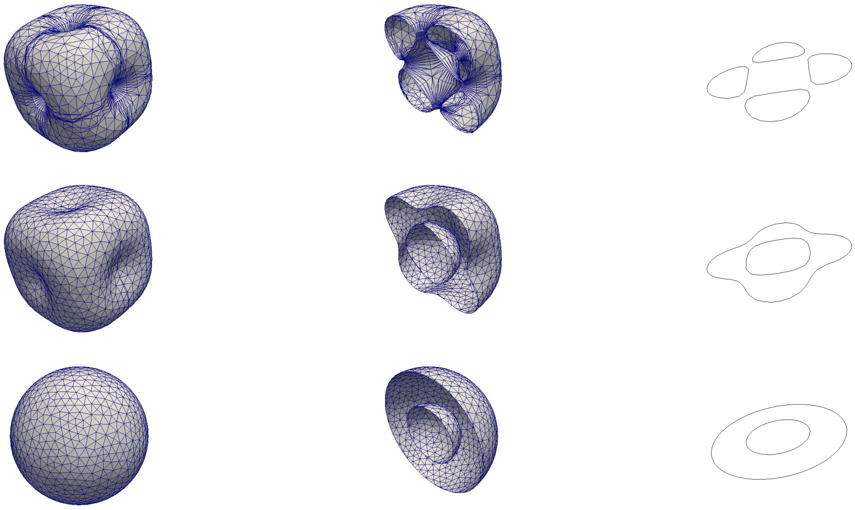

Figure 2.24: Temporal evolution of a $4 \times 4 \times 4$ "cage" under *surface diffusion*, at times $t = 0.48, 0.4802, 0.8$. On the left the interface, on the centre half-cuts for $x_2 = 0$, on the right the cross section for $x_3 = 0$.

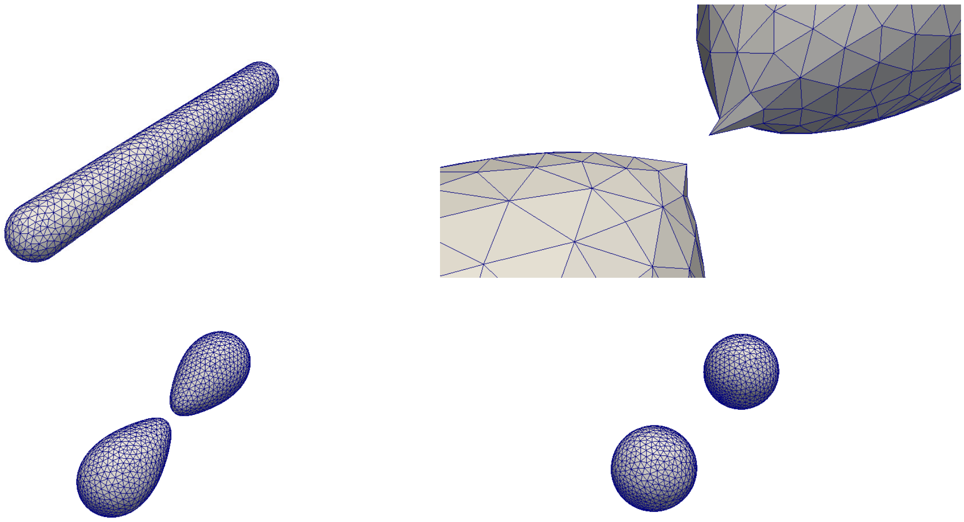

Figure 2.25: Temporal evolution for a $8 \times 1 \times 1$ "cigar like" rounded cylinder under *surface diffusion*, at times $t = 0, 0.237, 0.2372, 0.5$. The plots are scaled for the sake of visualisation. The loss of volume at the end of the simulations is 0.2%.

El-Topo performs a topological change leading to a pinch-off, as plotted in Figure 2.25. In the second part of the simulation we reduce the time step to $\tau = 10^{-7}$ until $T_2 = 0.2372$, from which we reuse $\tau = 10^{-4}$ until $T_3 = 0.5$. We can observe that after the topological change the two newly-formed surfaces evolve to a sphere-shaped stationary state.

Our last example is inspired by the test in [12, Fig. 14]. The initial geometry is given by a torus with radii $R = 1$ and $r = 0.25$. We divide the simulation into three parts. In the first part, we set $\tau = 10^{-4}$ and $T_1 = 0.0232$. At this point, El-Topo performs a topological change that modifies the torus into a genus-0 surface, as plotted in Figure 2.26. In order to capture the evolution correctly, we reduce the time step to $\tau = 10^{-7}$ until $T_2 = 0.0234$, from which we reuse $\tau = 10^{-4}$ until $T_3 = 0.052$. The complete evolution is plotted in Figure 2.26. We can see that after undergoing the topological change, the newly-formed genus-0 surface asymptotically tends to a sphere.

Additional examples of the usage of the El-Topo library will be presented at a later stage, in the framework of bulk-interface coupling.

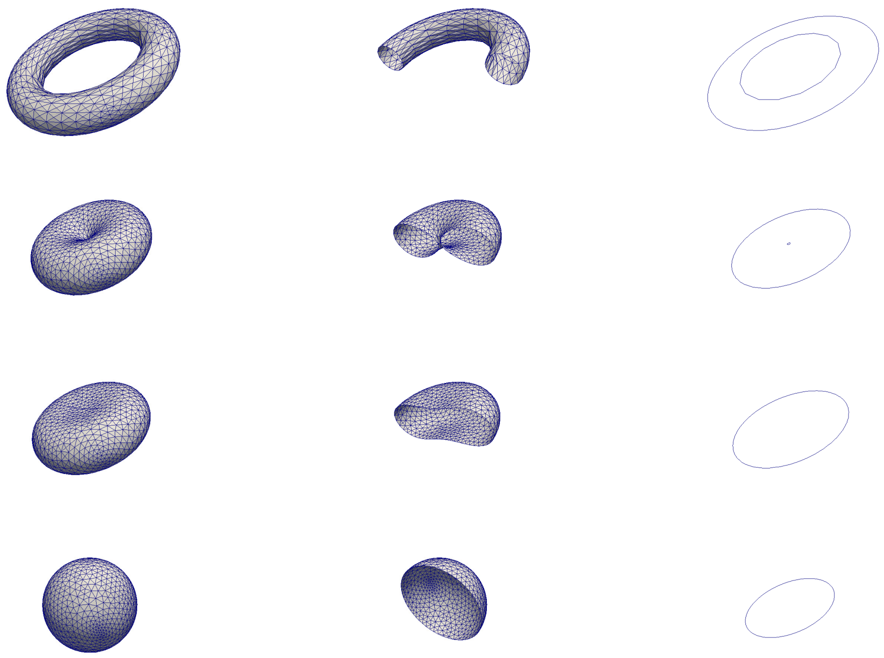

Figure 2.26: Temporal evolution of a torus with radii $R = 1$ and $r = 0.25$ under *surface diffusion*, at times $t = 0, 0.0232,\ 0.0234, 0.052$. On the left the interface, on the centre half-cuts for $x_2 = 0$, on the right the cross section for $x_3 = 0$.

2.4 SOLUTION METHODS

In this section, we aim at investigating various methods for the solution of the linear systems of equations arising from the finite element approximations discussed in Section 2.1. For ease of presentation, we define $\delta\vec{X}^{m+1} = \vec{X}^{m+1} - \vec{X}^m$.

In the case of *mean curvature flow*, equations (2.16a)-(2.16b) give rise to the following linear system: Find $\delta\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$ and $\kappa^{m+1} \in W(\Gamma^m)$ such that

$$\begin{pmatrix} M_m & -\frac{1}{\tau_m}\vec{N}_m^T \\ \vec{N}_m & \vec{A}_m \end{pmatrix} \begin{pmatrix} \kappa^{m+1} \\ \delta\vec{X}^{m+1} \end{pmatrix} = \begin{pmatrix} 0 \\ -\vec{A}_m\,\vec{X}^m \end{pmatrix}. \tag{2.56}$$

In the case of *surface diffusion*, equations (2.17a)-(2.17b) give rise to the following linear system: Find $\delta\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$ and $\kappa^{m+1} \in W(\Gamma^m)$ such that

$$\begin{pmatrix} A_m & -\frac{1}{\tau_m}\vec{N}_m^T \\ \vec{N}_m & \vec{A}_m \end{pmatrix} \begin{pmatrix} \kappa^{m+1} \\ \delta\vec{X}^{m+1} \end{pmatrix} = \begin{pmatrix} 0 \\ -\vec{A}_m\,\vec{X}^m \end{pmatrix}. \tag{2.57}$$

In the above, we have introduced the matrices $\vec{N}_m \in (\mathbb{R}^d)^{K_\Gamma^m \times K_\Gamma^m}$, $M_m \in \mathbb{R}^{K_\Gamma^m \times K_\Gamma^m}$, $A_m \in \mathbb{R}^{K_\Gamma^m \times K_\Gamma^m}$ and $\vec{A}_m \in (\mathbb{R}^{d\times d})^{K_\Gamma^m \times K_\Gamma^m}$, with entries

$$\left[\vec{N}_m\right]_{kl} := \langle \phi_k^m, \phi_l^m\,\vec{\nu}^m\rangle_{\Gamma^m}^h\,, \qquad [M_m]_{kl} := \langle \phi_k^m, \phi_l^m\rangle_{\Gamma^m}\,,$$
$$[A_m]_{kl} := \langle \nabla_s\,\phi_k^m, \nabla_s\,\phi_l^m\rangle_{\Gamma^m}\,,$$

where we recall that $\{\phi_j^m\}_{j=1}^{K_\Gamma^m}$ are the basis functions of $W(\Gamma^m)$. In addition, $[\vec{A}_m]_{kl} := [A_m]_{kl}\vec{Id}$, where $\vec{Id} \in \mathbb{R}^{d\times d}$ is the identity matrix.

In order to discuss efficient solution methods for linear systems as in (2.56) and (2.57), let us rewrite them in a more compact way, where superscripts and subscripts are dropped for ease of presentation. We are interested in solving a block linear system of the form

$$\begin{pmatrix} -\tau B & \vec{N}^T \\ \vec{N} & \vec{A} \end{pmatrix} \begin{pmatrix} \kappa \\ \vec{X} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{b} \end{pmatrix}, \tag{2.58}$$

where B can be either a stiffness matrix or a mass matrix. Before we proceed to discuss possible solution methods of (2.58), we note that an equivalent formulation, which is nonsymmetric but positive definite, is given by:

$$\begin{pmatrix} \tau B & -\vec{N}^T \\ \vec{N} & \vec{A} \end{pmatrix} \begin{pmatrix} \kappa \\ \vec{X} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{b} \end{pmatrix} . \tag{2.59}$$

In addition to the system (2.59), the following two equivalent reformulations can also be considered in practice:

$$\begin{pmatrix} -B & \frac{1}{\tau}\vec{N}^T \\ \vec{N} & \vec{A} \end{pmatrix} \begin{pmatrix} \kappa \\ \vec{X} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{b} \end{pmatrix} \tag{2.60}$$

and

$$\begin{pmatrix} B & -\frac{1}{\tau}\vec{N}^T \\ \vec{N} & \vec{A} \end{pmatrix} \begin{pmatrix} \kappa \\ \vec{X} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{b} \end{pmatrix} . \tag{2.61}$$

We are now ready to investigate in detail different methods of solution. In § 2.4.1 we discuss the Schur complement approach, where the two unknowns are obtained at separate stages, when possible. In Sections 2.4.2 and 2.4.3, instead, we compute both unknowns at once. In the former case, we apply iterative solvers, such as a BiCGSTAB algorithm. In the latter case, we make use of sparse solvers which allow permutations in the numberings of the unknowns. These solvers minimise the *fill-in* and guarantee the well posedness of the factorisation algorithm. In practice, we consider the packages LDL (see [41] for details), together with the sparse matrix ordering package AMD ([1]), and UMFPACK ([40]).

2.4.1 *Schur complement approach*

The linear system (2.58) can be solved with the help of a Schur complement approach. For a general introduction to this approach, see [69] and the references therein. If B is nonsingular, the system can be equivalently formulated as

$$\vec{D}\,\vec{X} := (\vec{A} + \tfrac{1}{\tau}\vec{N}B^{-1}\vec{N}^T)\vec{X} = \vec{b}\,, \qquad \kappa = \tfrac{1}{\tau}B^{-1}\vec{N}^T\vec{X}\,. \tag{2.62}$$

As the matrix $\vec{D}$ is positive definite, the system (2.62) can be solved with a Conjugate Gradient solver. This is the case for the motion by *mean curvature*, where B is the mass matrix, therefore invertible.

Unfortunately, in the case of *surface diffusion*, the matrix B is singular. Here we closely follow the discussion in [10, § 2]. Introducing the inverse S of B restricted on the set $(\ker B)^{\perp} \equiv (\operatorname{span}\{1\})^{\perp}$ and noting that the first equation in (2.58) implies $1^T\vec{N}^T\vec{X}$, one can transform (2.58) to

$$\kappa = \tfrac{1}{\tau} S\vec{N}^T\vec{X} + \mu 1\,, \tag{2.63}$$

$$(\vec{A} + \tfrac{1}{\tau}\vec{N}S\vec{N}^T)\vec{X} = \vec{b} - \mu\vec{N}1\,, \qquad (\vec{X})^T\vec{N}1 = 0\,, \tag{2.64}$$

where $\mu = \frac{1^T\kappa}{1^T1} \in \mathbb{R}$ is unknown. We introduce also the orthogonal projection $\vec{\Pi}$ onto $\mathcal{R}^{\perp} := \{\vec{X} \in (\mathbb{R}^d)^K : \vec{X}^T\vec{N}1 = 0\}$ by $\vec{\Pi} := \vec{\mathrm{id}}_K - \frac{\vec{w}\vec{w}^T}{\vec{w}^T\vec{w}}$ where $\vec{w} := \vec{N}1$. Then (2.64), on noting that $\vec{\Pi}\vec{X} = \vec{X}$, is replaced by

$$\vec{\Pi}(\vec{A} + \tfrac{1}{\tau}\vec{N}S\vec{N}^T)\vec{\Pi}\vec{X} = \vec{\Pi}\vec{b}\,. \tag{2.65}$$

The system (2.65) is positive semidefinite, so it can be solved with a Conjugate Gradient solver. The inverse S could be computed with the help of an iterative solution method, such as a CG iteration. However in practice it is better to solve for the pseudoinverse S directly. This can be done as follows, recalling that for motion by *surface diffusion* we have the case $\ker B = \operatorname{span}\{1\}$. We first obtain the factorisation $B = LDL^T$ employing the package LDL. Assuming without loss of generality that $D_{KK} = 0$, then the solution to the linear system $Bx = f$, for $f \in \mathbb{R}^K$ with $f^T\,1 = 0$, is given by:

$$x = z - \frac{z^T\,1}{1^T\,1}\,1\,, \qquad z = L^{-T}\widehat{D}^{-1}L^{-1}f\,, \tag{2.66}$$

where $\widehat{D}$ is the same as D, apart from $\widehat{D}_{KK} := 1$. It is worth noting that, solving for (2.65), each (outer) iteration of the CG method requires the solution of the system (2.66), where the factorisation $B = LDL^T$ does not change. The matrices L and D can therefore be factorised only once and employed without additional computational cost at each iteration of the outer CG. In addition, at every iteration of the outer CG, the right-hand side f for the sub-system (2.66) is given by $\vec{N}^T\vec{\Pi}\vec{X}^*$, where $\vec{X}^*$ is the current iterate. Given the definition of the projection $\vec{\Pi}$, f is therefore automatically guaranteed to satisfy the condition $f^T\, 1 = 0$.

The presence of the projection operator $\vec{\Pi}$ makes it difficult to find efficient preconditioners for the outer CG iteration for the system (2.65). Usually, only simple diagonal preconditioners are applied. In particular, in the case that $\operatorname{diag}(B)$ is nonsingular (which always holds for both *mean curvature flow* and *surface diffusion*), a possible choice of diagonal preconditioner is given by

$$\vec{\Pi}\vec{\mathcal{P}}^{-1}\vec{\Pi}\,, \qquad \vec{\mathcal{P}} := \operatorname{diag}(\vec{A}) + \tfrac{1}{\tau}\operatorname{diag}(\vec{N}\,[\operatorname{diag}(B)]^{-1}\,\vec{N}^T)\,. \tag{2.67}$$

Alternatively, one could use the following preconditioner:

$$\vec{\Pi}\vec{\mathcal{P}}^{-1}\vec{\Pi}\,, \qquad \vec{\mathcal{P}} := \vec{A} + \tfrac{1}{\tau}\vec{N}\,[\operatorname{diag}(B)]^{-1}\,\vec{N}^T\,. \tag{2.68}$$

Of course, this preconditioner will only be practical if it is solved for exactly with a direct (sparse) solver rather than an iterative solution method. As the operator $\vec{\mathcal{P}}$ is clearly symmetric positive definite, we can employ the package LDL.

The factorisation of (2.68) could be computationally expensive in practice, outweighing the achieved reduction in number of iterations. Hence, as an alternative, one could consider

$$\vec{\Pi}\vec{\mathcal{P}}^{-1}\vec{\Pi}\,, \qquad \vec{\mathcal{P}} := \operatorname{diag}(\vec{A}) + \tfrac{1}{\tau}\vec{N}\,[\operatorname{diag}(B)]^{-1}\,\vec{N}^T\,. \tag{2.69}$$

We will compare the three preconditioning techniques (2.67), (2.68) and (2.69) in terms of CPU time and number of iterations. As initial data for (2.62) and (2.65) we always choose $\vec{X} = \vec{0}$.

2.4.2 *Preconditioned BiCGSTAB*

Alternatively, one can solve (2.58) with a BiCGSTAB solver. As the system is in general not very well conditioned, one needs to employ a good preconditioner in practice. In [38] several preconditioners are suggested for the system (2.58), see also [116]. We consider here the choice [38, eq. (4)]

and present the details for the nonsymmetric system (2.61). The preconditioner immediately carries over to the remaining equivalent formulations discussed before. The authors in [38, (4)] proposed the operator:

$$P_c := \begin{pmatrix} B & -\frac{1}{\tau}\vec{N}^T \\ \vec{N} & \operatorname{diag}(\vec{A}) \end{pmatrix}^{-1} . \tag{2.70}$$

In order to apply the preconditioner P_c, we need to solve a system of the form

$$\begin{pmatrix} B & -\frac{1}{\tau}\vec{N}^T \\ \vec{N} & \operatorname{diag}(\vec{A}) \end{pmatrix} \begin{pmatrix} \kappa \\ \vec{X} \end{pmatrix} = \begin{pmatrix} c \\ \vec{g} \end{pmatrix} . \tag{2.71}$$

Now (2.71) can be solved with a Schur complement approach, leading to

$$(B + \tfrac{1}{\tau}\vec{N}^T \left[\operatorname{diag}(\vec{A})\right]^{-1} \vec{N})\kappa = c + \tfrac{1}{\tau}\vec{N}^T \left[\operatorname{diag}(\vec{A})\right]^{-1} \vec{g} \tag{2.72}$$

and

$$\vec{X} = \left[\operatorname{diag}(\vec{A})\right]^{-1} (\vec{g} - \vec{N}\kappa) . \tag{2.73}$$

We note that (2.72) is a symmetric positive definite system, which can be solved with a direct solver, like the package LDL. We note that also in this case the factorisation of (2.72) needs to be computed only once, since the preconditioner (2.70) is fixed for the whole iteration. As initial data for the BiCGSTAB solver we always choose $\vec{X} = \vec{0}$ and $\kappa = 0$.

2.4.3 *Direct factorisation*

As the system (2.58) is symmetric and nonsingular, it is also possible to solve it with a sparse factorisation solver. In practice, we employ the packages LDL and UMFPACK.

2.4.4 *Comparison between solution methods*

In this section we compare the three solution strategies presented in § 2.4.1, § 2.4.2 and § 2.4.3 for several geometries, in $d = 2$ and $d = 3$.

For the case $d = 2$, we first consider the same ellipse analysed in [10, Fig. 4]. We refine its mesh width, since $J_\Gamma^m = K_\Gamma^m = 64$ would give rise to negligible CPU times. We consider two cases: a coarser one with $J_\Gamma^m = K_\Gamma^m = 1004$ and a finer one with $J_\Gamma^m = K_\Gamma^m = 10024$. Our third example for $d = 2$ corresponds to the test in [10, Fig. 5], with a refined geometry given by $J_\Gamma^m = K_\Gamma^m = 10001$.

For the case $d = 3$, we first consider a "cage" domain of the same shape as in Figure 2.15, with different mesh sizes: a coarser one with $K_\Gamma^m = 3816$ and a finer one with $K_\Gamma^m = 124732$. Our next examples are inspired by the tests in [12, Figs. 21 and 22], which we will reconsider at a later stage in the case of topological changes. The initial configuration is a "cigar like" rounded cylinder of dimension $8 \times 1 \times 1$, with two different mesh sizes: $K_\Gamma^m = 1322$ and $K_\Gamma^m = 139550$, respectively. The last examples are inspired by the test in [12, Fig. 4]. The initial surface consists of a cube, where the parameterisation is such that it is very coarse for the lower part, and more refined for the upper part. Two different mesh sizes have been considered: $K_\Gamma^m = 1658$ and $K_\Gamma^m = 6394$, respectively.

A comparison between different solution strategies is reported in Tables 2.1 (for the formulation (2.58)), 2.2 (for the formulation (2.59)), and 2.3 (for the formulation (2.61)). In Table 2.4 we compare the three different preconditioning choices (2.67), (2.68) and (2.69) for the Schur complement approach applied to (2.59). In all the cases we solve one single time step for *surface diffusion*, with $\tau = 10^{-6}$. The stopping criterion for all iterative algorithms has been a reduction by a factor of 10^{-10} with respect to the initial residuum.

For all the formulations (2.58), (2.59) and (2.61), direct factorisation strategies perform best, both in terms of CPU time and residuum. We do not report the latter, in order to keep Tables 2.1, 2.2 and 2.3 more readable. For the symmetric formulation (2.58), LDL outperforms UMFPACK in the case $d = 2$, while UMFPACK proves to be the fastest solver with $d = 3$. In addition, it is worthwhile to mention that the preconditioned BiCGSTAB algorithm outperforms the Schur complement in all the cases where convergence is reached. We also note that both direct solvers and BiCGSTAB return the value of the unknowns $\vec{X}^{m+1}$ and κ^{m+1} at once, while in the

K_Γ^m	Schur [s]	BiCGSTAB [s]	UMFPACK [s]	LDL [s]
ellipse_{1004}	0.0934972 (474)	0.316954 (421)	0.0066117	0.00171481
ellipse_{10024}	8.69014 (4745)	–	0.0658188	0.019992
usp_{10001}	27.7106 (11346)	–	0.0534503	0.01277
cage_{3816}	2.0741 (1508)	0.1724 (47)	0.0880664	0.148899
cage_{124732}	480.413 (5608)	63.2434 (243)	20.1991	57.436
cigar_{1322}	0.814498 (1970)	0.0821715 (65)	0.0344753	0.0155069
cigar_{139559}	467.818 (5021)	193.955 (687)	15.9091	41.6407
usp_{1658}	0.249725 (327)	0.378896 (64)	0.0803338	0.070555
usp_{6394}	4.51197 (484)	1.93529 (126)	0.483679	0.768679

Table 2.1: Comparison of performance between the three solution strategies, namely Schur complement approach, preconditioned BiCGSTAB and direct factorisation, for the formulation (2.58). In brackets is the number of iterations needed for convergence. The symbol '–' indicates that no convergence was reached after 50000 iterations.

case of *surface diffusion* the Schur complement approach only computes the positon vector $\vec{X}^{m+1}$. Finally, we can see from Table 2.4 that the best preconditioning choice for the Schur complement approach is given by (2.69), in terms of both CPU times and number of iterations. The more sophisticated strategy (2.68) generally reduces the number of iterations necessary to reach convergence, but this feature is achieved at the cost of a dramatic increase in CPU time, which makes the strategy unfeasible in practice.

All the simulations have been performed with the help of the C++-based toolbox DUNE (see [22, 21]) and the discretisation module `dune-fem` (see [48]). We recently contributed to the current trunk version of `dune-fem`, providing a set of inverse operators based on the sparse factorisation packages UMFPACK, LDL and SPQR ([42]); see A.3 for more details.

REMARK 2.17. *Our analysis about different solution methods immediately carries over to the multi-component extensions* (2.51a)-(2.51b) *and* (2.52a)-(2.52b). *We note that, in order to apply the Schur complement approach presented in § 2.4.1 to*

K_{Γ}^{m}	Schur [s]	BiCGSTAB [s]	UMFPACK [s]
ellipse_{1004}	0.15465 (474)	0.326215 (421)	0.00661912
ellipse_{10024}	11.0413 (4745)	–	0.0618152
usp_{10001}	29.329 (11346)	–	0.0659916
cage_{3816}	2.26495 (1506)	0.206006 (47)	0.179766
cage_{124732}	559.517 (5608)	50.1937 (243)	20.5345
cigar_{1322}	1.17244 (1970)	0.09944 (65)	0.0339386
cigar_{139559}	528.717 (5021)	155.904 (687)	17.4718
usp_{1658}	0.59104 (327)	0.208178 (64)	0.077629
usp_{6394}	3.14164 (484)	2.18038 (126)	0.519092

Table 2.2: Comparison of performance between the three solution strategies, namely Schur complement approach, preconditioned BiCGSTAB and direct factorisation, for the formulation (2.59). In brackets is the number of iterations needed for convergence. The symbol '–' indicates that no convergence was reached after 50000 iterations.

the multiple-component case, the user would need to know explicitly the ordering of the degrees of freedom associated to the different surfaces. In fact, when surface diffusion is considered, the projection operator (2.65) *would have to be applied component-wise. This feature makes the application of the Schur complement strategy not feasible in practice. Therefore, also for the solution of the linear systems arising from* (2.51a)-(2.51b) *and* (2.52a)-(2.52b)*, the best strategy is to use the sparse factorisation solvers UMFPACK and LDL.*

K_Γ^m	Schur [s]	BiCGSTAB [s]	UMFPACK [s]
ellipse$_{1004}$	0.132566 (474)	0.407295 (423)	0.00745771
ellipse$_{10024}$	11.7392 (4745)	–	0.0751189
usp$_{10001}$	26.9042 (11346)	–	0.0686211
cage$_{3816}$	2.98698 (1508)	0.323536 (47)	0.152187
cage$_{124732}$	482.017 (5608)	59.7989 (255)	21.536
cigar$_{1322}$	0.814333 (1970)	0.148154 (65)	0.026165
cigar$_{139559}$	461.317 (5021)	198.527 (611)	17.3792
usp$_{1658}$	0.456745 (327)	0.32329 (64)	0.0732525
usp$_{6394}$	4.71077 (484)	2.5761 (126)	0.459524

Table 2.3: Comparison of performance between the three solution strategies, namely Schur complement approach, preconditioned BiCGSTAB and direct factorisation, for the formulation (2.61). In brackets is the number of iterations needed for convergence. The symbol '–' indicates that no convergence was reached after 50000 iterations.

2.4.5 *Solution methods for the α-scheme with very small time steps*

We now go back to the α-scheme (2.17a), (2.48a)-(2.48b) presented in § 2.2.4. We need to solve the following linear system: Find $\delta\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$, $\kappa^{m+1} \in W(\Gamma^m)$ and $\beta_i^{m+1} \in W(\Gamma^m)$, $i = 1 \to d-1$, such that

$$\begin{pmatrix} A_m & -\frac{1}{\tau_m}\vec{N}_m^T & & & \\ \vec{N}_m & \vec{A}_m & \alpha\,\vec{Z}_{1,m} & \cdots & \alpha\,\vec{Z}_{d-1,m} \\ & -\frac{\alpha}{\tau_m}\,\vec{Z}_{1,m}^T & \alpha\,M_m & & \\ & \vdots & & \ddots & \\ & -\frac{\alpha}{\tau_m}\,\vec{Z}_{d-1,m}^T & & & \alpha\,M_m \end{pmatrix} \begin{pmatrix} \kappa^{m+1} \\ \delta\vec{X}^{m+1} \\ \beta_1^{m+1} \\ \vdots \\ \beta_{d-1}^{m+1} \end{pmatrix} = \begin{pmatrix} 0 \\ -\vec{A}_m\,\vec{X}^m \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{2.74}$$

In the above, we have introduced the matrices $\vec{Z}_{i,m} \in (\mathbb{R}^d)^{K_\Gamma^m \times K_\Gamma^m}$, $i = 1 \to d-1$, with entries $\left[\vec{Z}_{i,m}\right]_{kl} := \langle \phi_k^m, \phi_l^m\,\vec{\tau}_i^m\rangle_{\Gamma^m}^h$. We always solve the

	Schur complement (2.65) with outer CG		
K_Γ^m	(2.67)	(2.68)	(2.69)
ellipse$_{1004}$	0.15465 (474)	–	0.130424 (474)
ellipse$_{10024}$	11.0413 (4745)	–	12.6034 (4745)
usp$_{10001}$	29.329 (11346)	1.20865 (3)	29.3774 (11131)
cage$_{3816}$	2.26495 (1506)	32.6047 (118)	0.648456 (482)
cage$_{124732}$	559.517 (5608)	7947.05 (552)	523.16 (5597)
cigar$_{1322}$	1.17244 (1970)	7.85051 (96)	0.157636 (442)
cigar$_{139559}$	467.818 (5021)	–	334.419 (4745)
usp$_{1658}$	0.59104 (327)	13.2231 (99)	0.269663 (273)
usp$_{6394}$	3.14164 (484)	350.96 (241)	2.40654 (412)

Table 2.4: Comparison of performance between the three preconditioning strategies (2.67), (2.68) and (2.69) for the Schur complement approach applied to the formulation (2.59). The symbol '–' indicates that no convergence was reached after 50000 iterations.

linear system (2.74) with the help of the sparse factorisation package UMFPACK.

2.5 CONCLUSIONS

In this chapter we have analysed the fully practical finite element approximation for the interface equations which will be used throughout the thesis. In particular, we have discussed the choice of two different quadrature rules (namely, mass lumping and exact integration), the introduction of an additional term to overcome the appearance of spurious elongated elements, and several solution strategies for the algebraic counterparts of the aforementioned finite element approximation. We summarise here the main results of our investigation for the benefit of the reader.

In § 2.2.3 we have analysed two different integration rules: the mass lumping scheme (2.12), which corresponds to the trapezium rule for numerical integration, and the exact integration scheme (2.11). In order to test how the choice of the quadrature rule affects the temporal evolution of the quality of the mesh, we have performed several numerical experiments. For $d = 2$, in all our tests we observe that both schemes lead to the equidistribution of mesh points, and the plots for mesh quality indicators are almost indistinguishable. For $d = 3$, mass lumping contributes to keeping smoother meshes; in only one experiment, namely the "cage" under *surface diffusion*, does exact integration perform better than mass lumping. However, when the "cage" is about to undergo a topological change, the quality of the mesh deteriorates more quickly when exact integration is employed. In light of these results, for all the numerical simulations conducted in Chapters 4 and 5, the mass lumping scheme (2.12) has been used in practice.

In § 2.2.4 we have investigated the temporal evolution of surfaces when very small time steps are considered. In particular, we observe the appearance of spurious elongated elements. In order to eliminate such elements, we apply a different strategy, where also the tangential components of the discrete surface Laplacian of $\vec{X}^{m+1}$ are represented. We observe that the quality of the evolving mesh strongly depends on the choice of the parameter α, which multiplies the additional term in the finite element approximation. Moreover, by taking the limit $\alpha \to 0$, we notice that the temporal behaviour of the quality mesh indicator is very similar to the one exhibited by the main scheme (2.17a)-(2.17b).

Finally, in Section 2.4 we have investigated several methods for the solution of the linear equations arising from the finite element approximation discussed in Section 2.2, namely Schur complement approach, preconditioned BiCGSTAB and direct solvers. From a comparison for a number of sample problems, in terms of CPU time and number of iterations (for iterative methods), we observe that direct factorisation strategies perform best. In addition, they can be easily applied with no additional knowledge on the numbering of the degrees of freedom when multi-component interfaces are considered.

3 MODELS OF ELECTRO-STRESS MIGRATION

This chapter is dedicated to the presentation of the electro-stress migration problem which will be investigated from here on in the thesis. In Section 3.1 we give an overview of the phenomenon, highlighting the challenges faced by interconnect designers. Section 3.2 introduces the mathematical formulation of the electro-stress migration problem, defining all the necessary bulk and interface quantities, which will be computed with the help of the finite element approximations described in Chapters 4 and 5.

3.1 THE PHYSICS OF ELECTRO-STRESS VOID MIGRATION

The term *electro-stress migration* refers to a mass transport process that operates in solid state metals stressed under high electrical current; see [123, 28, 25]. When the current density is high enough to cause the drift of metal ions, the dimension and the shape of the conductor may vary, thereby causing the creation of voids, hillocks or whiskers in the affected regions. These defects are responsible for open circuits or short circuits.

The aggressive miniaturisation of interconnect dimensions and the increasing current densities at operation have aggravated the problem. Therefore, electro-stress migration represents a strong challenge to the development of advanced interconnects for integrated circuits. Manual current-density considerations within complex circuits are not sufficient any longer. The application of mathematical modelling and efficient numerical techniques can guarantee a better understanding of the phenomenon and help electronic engineers improve the physical design of integrated circuits, see [119]. In particular, special attention is to be paid to predict the lifetime of chips, which depends on many factors, like the nature of the conductor, crystal size, interface and grain-boundary chemistry, and the magnitude of the forces involved. We refer to [32, § 1.1] for a historical overview of *mean* *time to failure* (MTF) models, and to [72] for the analysis of the dependency of failure mechanisms on the direction of the ion flux and the geometry (i.e. width and length) of metallic wires. [97] discusses various methodologies to address the electro-stress migration problem directly during physical design and verification, for both analogue and digital circuits. More details on process-related advances in electro-migration reliability and lifetime extrapolation can be found in [80].

3.2 THE MATHEMATICAL MODELLING OF ELECTRO-STRESS VOID MIGRATION

This section is dedicated to the discussion of the mathematical model that will be used from here on in this thesis. We follow [123] and report here the main assumptions behind the model.

An idealised interconnect is represented in [123, Fig. 1]. It is assumed to be a linear elastic solid, which conducts electric current according to Ohm's law. The solid is subject to two different external loadings: an electric field, which is induced in the line by prescribing the voltage on its boundaries, and an elastic stress, which is imposed as a specific traction or displacement. The displacements are considered infinitesimal. One or more voids can be present in the line, and their initial shape and location are known.

We have three driving contributions to the growth and the movement of the voids: the surface free energy of the material, the electric field in the line, and the elastic strain energy. The overall goal of the numerical approximation of our coupled bulk-interface problem is therefore to compute the elastic stress and the intensity of the electric field in the metallic wires, in order to monitor the change in shape of the voids.

For the formulation of the governing equations we closely follow the presentation in [104], see also [123, 9]. Let $\Omega = \times_{i=1}^{d}[-L_i, L_i]$, where $L_i > 0$ for all $i = 1, \ldots, d$, be the domain that contains the conductor. We denote the boundary of Ω with $\partial\Omega$. At any time $t \in [0, T]$, let $\Gamma(t) \subset \Omega$ be the boundary of the void $\Omega_-(t)$ inside the conductor Ω. Then $\Gamma(t) = \partial\Omega_-(t)$ and $\Omega_+(t) := \Omega \setminus \overline{\Omega_-(t)}$ denotes the conducting region (see Figure 3.1 for an example with $d = 2$). At this stage, we assume that $\Gamma(t)$ is a closed hypersurface without boundary and does not intersect $\partial\Omega$. At a later stage we will generalise our discussion to the case of a hypersurface $\Gamma(t)$ with boundary, where appropriate conditions have to be prescribed. Now the evolution of the interface $\Gamma(t)$, which represents the void boundary, is given by

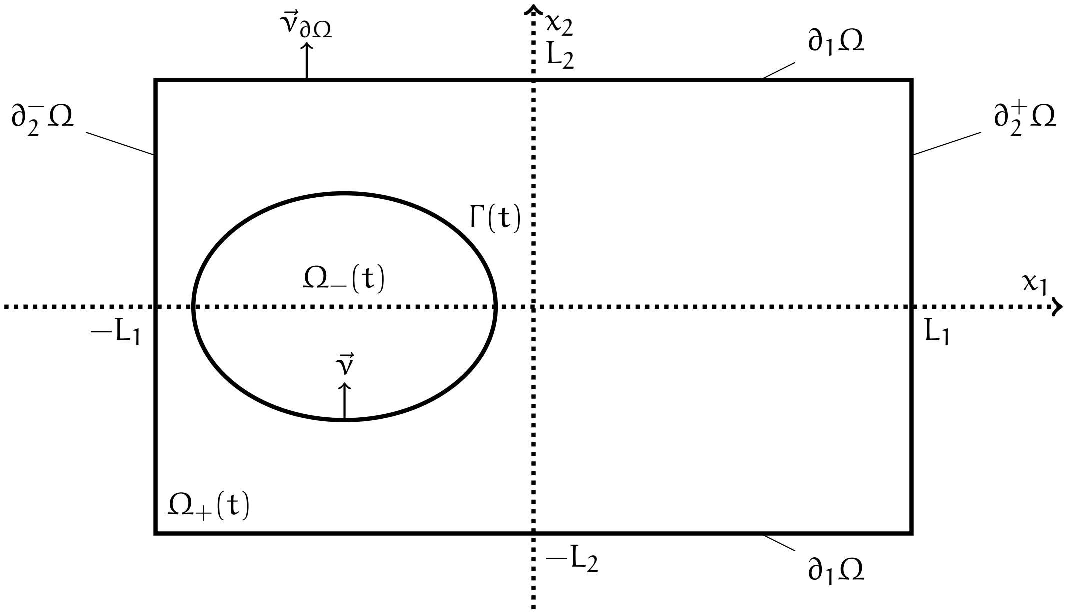

Figure 3.1: The domain Ω and the void with its boundary $\Gamma(t)$, for the case $d = 2$ (from [105, Fig. 1]).

$$V = -\,\alpha_1 \,\triangle_s\, \kappa + \alpha_2 \,\triangle_s\, \phi + \alpha_3 \,\triangle_s\, (E(\vec{u}))\,, \tag{3.1}$$

where V represents the velocity of $\Gamma(t)$ in the direction $\vec{\nu}$ (the unit normal to $\Gamma(t)$ pointing into $\Omega_-(t)$), $\triangle_s$ is the Laplace-Beltrami operator on $\Gamma(t)$, and κ is the curvature of $\Gamma(t)$ (positive when $\Omega_-(t)$ is convex). In particular, it holds that

$$\triangle_s \vec{X} = \kappa\, \vec{\nu}\,, \tag{3.2}$$

where $\vec{X}$ is a suitable parameterisation of $\Gamma(t)$, recall (2.7).

The second contribution on the right-hand side of (3.1) is given by the electric potential $\phi(t)$, which satisfies a Laplace equation in $\Omega_+(t)$:

$$\triangle\, \phi = 0 \quad \text{in } \Omega_+(t)\,, \qquad \frac{\partial \phi}{\partial \vec{\nu}} = 0 \quad \text{on } \Gamma(t)\,, \tag{3.3a}$$

$$\frac{\partial \phi}{\partial \vec{\nu}_{\partial\Omega}} = 0 \text{ on } \partial_1\Omega\,, \qquad \phi = g^{\pm} \quad \text{on } \partial_2^{\pm}\Omega\,, \tag{3.3b}$$

where $\vec{\nu}_{\partial\Omega}$ is the outer normal to $\partial\Omega$. In (3.3b), $g^{\pm} := \pm L_1$ denotes the Dirichlet boundary condition on parts of $\partial\Omega$, where $\partial\Omega = \partial_1\Omega \cup \partial_2\Omega$, with $\partial_1\Omega \cap \partial_2\Omega = \emptyset$ and

$$\partial_2\Omega = \partial_2^-\Omega \cup \partial_2^+\Omega \;\text{ with }\; \partial_2^{\pm}\Omega := \{\pm L_1\} \times \left(\times_{i=2}^{d}[-L_i, L_i]\right) .$$

The Dirichlet boundary conditions in (3.3b) model a uniform parallel electric field, $\phi \approx x_1$ as $L_1 \to \infty$.

The third contribution on the right-hand side of (3.1) is given by the elastic energy density $E(\vec{u})$, where $\vec{u}$ is the displacement field. Precisely, $E(\vec{u}) := \frac{1}{2}\,\mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{u}) : \underline{\underline{\mathcal{E}}}(\vec{u})$, where $\underline{\underline{\mathcal{E}}}(\vec{u}) := \frac{1}{2}\,(\nabla\,\vec{u} + (\nabla\,\vec{u})^T)$ is the symmetric strain tensor and $\mathcal{C}$ is the possibly anisotropic elasticity tensor, which we assume to be symmetric and positive definite. In particular, we recall that $\underline{\underline{\mathcal{E}}}(\vec{u})$ is a $d \times d$ matrix, while $\mathcal{C}$ is a tensor that maps $d \times d$ matrices to $d \times d$ matrices. For the sake of notation, recall that the inner product $\underline{\underline{\mathcal{A}}} : \underline{\underline{\mathcal{B}}}$ of two matrices $\underline{\underline{\mathcal{A}}}, \underline{\underline{\mathcal{B}}} \in \mathbb{R}^{d\times d}$ is defined as $\sum_{i,j=1}^{d} \mathcal{A}_{ij}\,\mathcal{B}_{ij}$. In addition, for a matrix $\underline{\underline{\mathcal{A}}}$, we denote its trace by $\mathrm{Tr}(\underline{\underline{\mathcal{A}}}) := \sum_{i=1}^{d} \mathcal{A}_{ii}$, while its divergence is a vector-valued function defined component-wise as $(\nabla\cdot\underline{\underline{\mathcal{A}}})_i = \sum_{j=1}^{d} \frac{\partial \mathcal{A}_{ij}}{\partial x_j}$. We will assume throughout that for all $i, j, k, l \in \{1, \ldots, d\}$ the following symmetry conditions hold:

$$\mathcal{C}_{ijkl} = \mathcal{C}_{ijlk} = \mathcal{C}_{jikl} = \mathcal{C}_{lkij} \,. \tag{3.4}$$

$\mathcal{C}$ then maps symmetric matrices to symmetric matrices. Moreover, it holds that $\mathcal{C}\,\underline{\underline{\mathcal{A}}} : \underline{\underline{\mathcal{B}}} = \underline{\underline{\mathcal{A}}} : \mathcal{C}\,\underline{\underline{\mathcal{B}}}$. We also assume throughout that $\mathcal{C}$ is positive definite; that is, there exist positive constants $m_{\mathcal{C}}$, $M_{\mathcal{C}}$ such that

$$0 < m_{\mathcal{C}}\,(\underline{\underline{\mathcal{A}}} : \underline{\underline{\mathcal{A}}}) \leqslant \mathcal{C}\,\underline{\underline{\mathcal{A}}} : \underline{\underline{\mathcal{A}}} \leqslant M_{\mathcal{C}}\,(\underline{\underline{\mathcal{A}}} : \underline{\underline{\mathcal{A}}}) \qquad \forall\,\underline{\underline{\mathcal{A}}} \in \mathbb{R}^{d\times d} \setminus \{\underline{\underline{0}}\}. \tag{3.5}$$

For an isotropic material we obtain that

$$\mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{u}) = 2\mu\,\underline{\underline{\mathcal{E}}}(\vec{u}) + \lambda\,\mathrm{Tr}(\underline{\underline{\mathcal{E}}}(\vec{u}))\,\underline{\underline{\mathcal{I}}}, \tag{3.6}$$

where $\underline{\underline{\mathcal{I}}}$ is the identity matrix, and $\mu \in \mathbb{R}_{>0}$ and $\lambda \in \mathbb{R}_{\geqslant 0}$ are the Lamé moduli. The unknown $\vec{u}$ is the displacement field which is the solution to the following problem:

$$\nabla\cdot(\mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{u})) = \vec{0} \quad \text{in } \Omega_+(t)\,, \tag{3.7a}$$

$$\mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{u})\,\vec{\nu} = \vec{0} \quad \text{on } \Gamma(t)\,, \tag{3.7b}$$

$$\mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{u})\,\vec{\nu}_{\partial\Omega} = \vec{g} \quad \text{on } \partial\Omega\,. \tag{3.7c}$$

For simplicity, we will consider

$$\vec{g} = \underline{\underline{\mathcal{S}}}\,\vec{\nu}_{\partial\Omega} = \mathcal{C}\,\underline{\underline{\mathcal{S}}}^*\,\vec{\nu}_{\partial\Omega}\,, \tag{3.8}$$

where $\underline{\underline{\mathcal{S}}} \in \mathbb{R}^{d\times d}$ is a symmetric matrix and $\underline{\underline{\mathcal{S}}}^* := \mathcal{C}^{-1}\,\underline{\underline{\mathcal{S}}}$. Alternatively, one could prescribe displacement boundary conditions, $\vec{u} = \vec{f}$ on $\partial\Omega$ or on parts thereof.

We should note that the solution $\vec{u}$ to (3.7a)-(3.7c) is not unique. This is simply because

$$\underline{\underline{\mathcal{E}}}(\vec{v}) = \vec{0} \qquad \forall\,\vec{v} \in \underline{\mathrm{RM}}\,, \tag{3.9}$$

where $\underline{\mathrm{RM}}$ is the space of rigid motions and characterised by

$$\begin{aligned}\underline{\mathrm{RM}} := \{\vec{h} \in \underline{H}^1(\Omega)\ :\ &\text{there exist } \vec{b} \in \mathbb{R}^n \text{ and } \underline{\underline{\mathcal{A}}} = -\underline{\underline{\mathcal{A}}}^T \in \mathbb{R}^{n\times n} \\ &\text{s.t. } \vec{h}(\vec{x}) = \vec{b} + \underline{\underline{\mathcal{A}}}\,\vec{x}\}\,.\end{aligned} \tag{3.10}$$

Hence one can impose uniqueness for (3.7a)-(3.7c) by seeking $\vec{u}$ such that, for all $\vec{h} \in \underline{\mathrm{RM}}$, $\int_{\Omega_+(t)} \vec{u}\cdot\vec{h}\,d\mathcal{L}^d = 0$. However, $\vec{u}$ enters the right-hand side of the evolution equation (3.1) only via $\underline{\underline{\mathcal{E}}}(\vec{u})$. Therefore, the non-uniqueness of $\vec{u}$ up to a rigid motion, which is characterised by (3.9), does not play any role in the temporal evolution of the void boundary $\Gamma(t)$.

Finally, $\alpha_1 \in \mathbb{R}_{>0}$ and α_2, $\alpha_3 \in \mathbb{R}_{\geqslant 0}$ are given parameters depending on the conductor, on the strength of the electric field and on the magnitude of the elastic stress; for a more detailed explanation of the dependency of α_1, α_2, and α_3 on various physical constants, see [123, § 2] and [95, § 2]. The first term on the right-hand side of (3.1) is *surface diffusion* due to interfacial tension, which models atoms moving around the boundary of the void to positions of large curvature, whereas the second and third term are *surface diffusion* due to the electric field and the elastic energy, respectively. The void electro-stress migration model is then the coupled system of equations (3.1), (3.3a)-(3.3b), and (3.7a)-(3.7c). In the case $\alpha_2 = \alpha_3 = 0$, the evolution (3.1) reduces to pure *surface diffusion*. For α_2, $\alpha_3 \geqslant 0$, the motion (3.1) preserves the volume enclosed by the closed hypersurface $\Gamma(t)$ since

$$\frac{d}{dt}\mathcal{L}^d(\Omega_-(t)) = -\int_{\Gamma(t)} V\,d\mathcal{H}^{d-1} = 0\,, \tag{3.11}$$

recall (2.5). In addition, for $\alpha_2 = 0$ the system has a Lyapunov structure. In fact, it can be shown that

$$\begin{aligned}
\frac{d}{dt}&\left[\alpha_1\,\mathcal{H}^{d-1}(\Gamma(t)) + \alpha_3\left(\int_{\Omega_+(t)} E(\vec{u})\,d\mathcal{L}^d - \int_{\partial\Omega} \vec{g}\cdot\vec{u}\,d\mathcal{H}^{d-1}\right)\right] \\
&= -\,\alpha_1\int_{\Gamma(t)} V\,\kappa\,d\mathcal{H}^{d-1} + \alpha_3\int_{\Omega_+(t)} \mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{u}) : \underline{\underline{\mathcal{E}}}(\vec{u}_t)\,d\mathcal{L}^d \\
&\quad + \alpha_3\int_{\Gamma(t)} V\,E(\vec{u})\,d\mathcal{H}^{d-1} - \alpha_3\int_{\partial\Omega} \vec{g}\cdot\vec{u}_t\,d\mathcal{H}^{d-1} \\
&= -\int_{\Gamma(t)} V\,(\alpha_1\,\kappa - \alpha_3\,E(\vec{u}))\,d\mathcal{H}^{d-1} \\
&= -\int_{\Gamma(t)} |\alpha_1\,\nabla_s\,\kappa - \alpha_3\,\nabla_s\,(E(\vec{u}))|^2\,d\mathcal{H}^{d-1} \leqslant 0\,. \qquad (3.12)
\end{aligned}$$

As a special case, we recover for $\alpha_2 = \alpha_3 = 0$ the well-known result that *surface diffusion* decreases the length of the interface, since

$$\frac{d}{dt}\mathcal{H}^{d-1}(\Gamma(t)) = -\int_{\Gamma(t)} V\,\kappa\,d\mathcal{H}^{d-1} = -\,\alpha_1\int_{\Gamma(t)} |\nabla_s\,\kappa|^2\,d\mathcal{H}^{d-1} \leqslant 0\,, \quad (3.13)$$

recall (2.6).

3.2.1 *Weak formulation of the coupled problem*

We can now derive a weak formulation of the electro-stress migration problem of our interest, which consists of the sets of equations (3.3a)-(3.3b), (3.7a)-(3.7c) and (3.1)-(3.2). It will be used in Chapters 4 and 5 for the introduction of appropriate finite element approximations of the coupled problem. Given Γ_0, the initial boundary of the void, find for a.e. $t \in [0, T]$ $\Gamma(t)$, $\vec{X} \in [H^1(\Gamma(t))]^d$, $\kappa \in H^1(\Gamma(t))$, $\phi(\cdot, t) \in H^1(\Omega_+(t))$, and $\vec{u}(\cdot, t) \in \underline{\widehat{H}}^1$, such that, for all $\psi \in H^1_{0,\partial_2}(\Omega_+(t))$, $\vec{\zeta} \in [H^1(\Omega_+(t))]^d$, $\chi \in H^1(\Gamma(t))$ and $\vec{\eta} \in [H^1(\Gamma(t))]^d$, it holds that

$$\int_{\Omega_+(t)} \nabla\,\phi \,\cdot\, \nabla\,\psi \;d\mathcal{L}^d = 0\,, \tag{3.14a}$$

$$\int_{\Omega_+(t)} \mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{u}) \,:\, \underline{\underline{\mathcal{E}}}(\vec{\zeta})\;d\mathcal{L}^d = \int_{\partial\Omega} \vec{g} \,\cdot\, \vec{\zeta}\;d\mathcal{H}^{d-1}\,, \tag{3.14b}$$

$$\begin{aligned}\int_{\Gamma(t)} \vec{X}_t \cdot \chi\,\vec{\nu}\;d\mathcal{H}^{d-1} - \int_{\Gamma(t)} \alpha_1\,\nabla_s\,\kappa \cdot \nabla_s\,\chi\;d\mathcal{H}^{d-1} &= \\ -\int_{\Gamma(t)} \nabla_s\,(\alpha_2\,\phi + \alpha_3\,E(\vec{u})) \cdot \nabla_s\,\chi\;d\mathcal{H}^{d-1}&\,,\end{aligned} \tag{3.14c}$$

$$\int_{\Gamma(t)} \kappa\,\vec{\nu}\cdot\vec{\eta}\;d\mathcal{H}^{d-1} + \int_{\Gamma(t)} \nabla_s\,\vec{X}\cdot\nabla_s\,\vec{\eta}\;d\mathcal{H}^{d-1} = 0\,, \tag{3.14d}$$

where $\phi(\cdot,t) \in H^1(\Omega_+(t))$ is such that $\phi = g^{\pm}$ on $\partial_2^{\pm}\Omega$. In the above we have defined $H^1_{0,\partial_2}(\Omega_+(t)) := \{\psi \in H^1(\Omega_+(t)) \,:\, \psi\,|_{\partial_2\Omega} = 0\}$ and $\underline{\widehat{H}}^1 := \{\vec{h} \in [H^1(\Omega_+(t))]^d \,:\, \int_{\Omega_+(t)} \vec{h}\cdot\vec{z}\;d\mathcal{L}^d = 0 \quad \forall\,\vec{z} \in \underline{\mathrm{RM}}\}$. The weak formulation of the void electro-stress migration problem of our interest is therefore given by (3.14a), (3.14b) and (3.14c)-(3.14d).

3.2.2 *Test case with exact solution*

For later use we recall from [20] an example for $d = 2$, with $\alpha_3 = 0$, where the exact solution is known. The solution describes a circular void, moving at constant speed through an infinite conductor. That is, for any α_1, $\alpha_2 \in \mathbb{R}_{\geqslant 0}$, $\alpha_3 = 0$, $R \in \mathbb{R}_{>0}$, and $\vec{z} = (z_1, z_2) \in \mathbb{R}^2$,

$$\begin{aligned}&\Gamma(t) := \{\vec{x} \in \mathbb{R}^2 : (x_1 - z_1(t))^2 + (x_2 - z_2)^2 = R^2\}\,,\\ &z_1(t) := z_1 + \tfrac{2\,\alpha_2}{R}\,t\,,\end{aligned} \tag{3.15}$$

where the corresponding electric potential

$$\phi(\vec{x},t) = [x_1 - z_1(t)]\left(1 + \tfrac{R^2}{(x_1-z_1(t))^2+(x_2-z_2)^2}\right) \tag{3.16}$$

solves (3.1) and (3.3a)-(3.3b) with $\Omega_+(t)$ in (3.3a) replaced by $\mathbb{R}^2 \setminus \overline{\Omega_-(t)}$ and (3.3b) replaced by the Dirichlet-type condition $\nabla\,\phi \to (1,0)^T$ as $|\vec{x}| \to \infty$. Observe that (3.1) reduces to $V = -\frac{2\,\alpha_2}{R^2}\,[x_1 - z_1(t)]$ on $\Gamma(t)$. The explicit solution (3.15)-(3.16) was first noted in [82].

4 THE PARAMETRIC APPROACH: THE *UNFITTED* METHOD

Let us recall what we described in Section 1.5. The numerical solution of partial differential equations for systems with moving boundaries can be approached in different ways. Three possible strategies for handling the interface have been introduced in the literature, namely the *parametric approach* (§ 1.3.1), the *level set approach* (§ 1.3.2) and the *phase field approach* (§ 1.3.3).

In this thesis we consider the *parametric approach*. The user can employ two different techniques: an *unfitted approach* and a *fitted approach*, depending on whether the topological compatibility between bulk and interface meshes is preserved. In this chapter we present the *unfitted approach* in detail. Our discussion expands on the exposition given in [104], where we analysed the *unfitted* approach with $d = 2$ and electro-migration only.

The chapter is organised as follows: in Section 4.1 we give a definition of the *unfitted approach*, highlighting its most important features. In Section 4.2 we introduce an *unfitted* finite element approximation for the electro-stress migration problem described in Chapter 3. We investigate existence and uniqueness of our finite element approximation, and discuss some properties of its semidiscrete, continuous-in-time counterpart. Section 4.3 presents the solution methods applied to the linear systems arising from our fully discrete schemes. In Section 4.4 we describe in detail all the mesh operations needed in our coupled bulk-interface framework. Special attention is paid to the routines for the adaptation of the bulk mesh and the identification of the intersections between bulk and interface grid elements. Finally, in Section 4.5 we present several numerical simulations, for both $d = 2$ and $d = 3$, including a convergence experiment for a test case where the exact solution is known.

4.1 FEATURES OF THE *unfitted* METHOD

Recall that the approach proposed in this thesis belongs to the *parametric* category. The main difference to existing methods, see [27, 123, 25], is that our approach is based on the weak formulation (3.14a)-(3.14d), that includes the treatment of the curvature presented in Section 2.2, eqns. (2.17a)-(2.17b). We will use in this chapter an *unfitted* finite element grid for the approximation of (3.1); see Figure 4.1 for an example with $d = 2$. Existing front-tracking methods for the approximation of void electro-stress migration require a new grid for the approximation of $\Omega_+(t)$ to be created at each time step. We avoid the drawback of this re-meshing, and the associated numerical effort, by maintaining a triangulation of the whole domain Ω throughout. Altogether, the main features of the *unfitted* approach can be briefly summarised as follows:

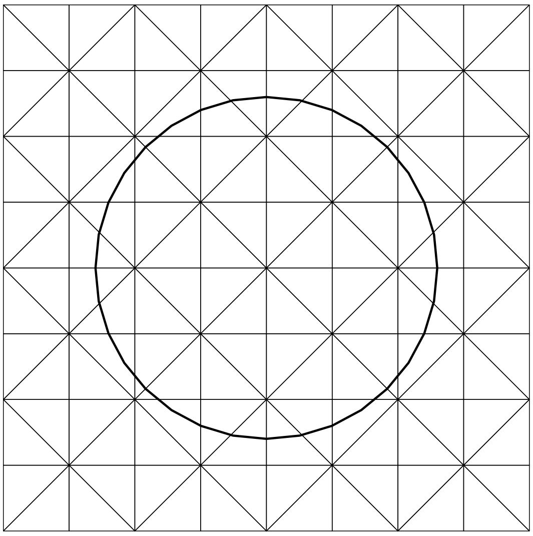

Figure 4.1: Example of an *unfitted* interface mesh, with $d = 2$ (from [104, Fig. 1]).

- the method is *unfitted*: the bulk grid and the interface grid are totally independent. It means that there is no need to re-mesh or deform the bulk mesh in order to preserve the correspondence with the interface. Standard strategies for refinement and coarsening can be employed for the bulk mesh. Furthermore, from the software implementation point of view, it follows that the two grids can be stored and manipulated as independent objects from two different classes;

- the method shows good properties for the interface mesh, whose temporal evolution is based on the scheme (2.17a)-(2.17b). In particular, for the case $d = 2$, the parametric approximation of the interface motion leads to an asymptotic equidistribution of the vertices, with no necessity of any refinement or re-meshing procedures;
- the volume enclosed by the surface is preserved by the fully discrete scheme up to a very small tolerance, while for a semidiscrete, continuous-in-time variant it is preserved exactly;
- the method can handle multi-component interfaces, presented in § 2.3, as well as topological changes that occur during the evolution. In combination with the package El-Topo, recall § 2.3.1, our method can deal with more complex domains, which are often found in electrical engineering applications.

4.2 FINITE ELEMENT APPROXIMATION

As already pointed out earlier, our approach is based on a coupled bulk-interface algorithm. For ease of presentation, we first consider the case where the interface is given by a single closed surface.

We begin with the finite element spaces needed for the approximation of the moving boundary $\Gamma(t)$. To this end, let us recall the notation already introduced in Section 2.2. Let Γ^m be a $(d-1)$-dimensional *polyhedral* surface, defined as $\Gamma^m = \bigcup_{j=1}^{J_\Gamma^m} \overline{\sigma_j^m}$, where $\{\sigma_j^m\}_{j=1}^{J_\Gamma^m}$ is a family of mutually disjoint open $(d-1)$-simplices with vertices $\{\vec{q}_k^m\}_{k=1}^{K_\Gamma^m}$. Moreover, let Ω_-^m and Ω_+^m be the interior and the exterior of Γ^m, respectively. The parametric finite element spaces for the approximation of the position vector $\vec{x}$ and the curvature κ in (3.1)-(3.2) have been defined in (2.10). Moreover, recall (2.12) and (2.13) for the definition of the mass lumped inner product $\langle\cdot,\cdot\rangle_{\Gamma^m}^h$ and the unit normal $\vec{\nu}^m$ to Γ^m, respectively. We assume that the local ordering of the vertices used in (2.13) is such that $\vec{\nu}^m$ is the normal to Γ^m pointing into Ω^m_-.

We introduce now the finite element approximation for quantities defined over the bulk mesh. Let $\mathcal{T}^m$ be a partitioning of Ω into disjoint open d-dimensional simplices o with $h_o := \operatorname{diam}(o)$ and $h := \max_{o \in \mathcal{T}^m} h_o$ so that $\overline{\Omega} = \bigcup_{o \in \mathcal{T}^m} \overline{o}$. We can then define

$$\mathcal{T}^m_+ := \{o \in \mathcal{T}^m : o \cap \Omega^m_+ \neq \emptyset\} \quad \text{and} \quad \Omega^{m,h}_+ := \bigcup_{o \in \mathcal{T}^m_+} \overline{o}\,. \tag{4.1}$$

$\Omega^{m,h}_+$ is hence given by the union of those bulk elements having a non-empty intersection with the exterior of Γ^m. It represents a suitable approximation of Ω^m_+, on which the electric potential and the elastic stress are going to be computed. Associated with $\mathcal{T}^m_+$ we can define the standard finite element space of piecewise linear functions

$$S^{m,h} := \{\chi \in C(\Omega^{m,h}_+) : \chi\,|_o \text{ is linear } \forall\, o \in \mathcal{T}^m_+\},$$

as well as

$$S^{m,h}_g := \{\chi \in S^{m,h} : \chi\,|_{\partial^\pm_2\Omega} = g^\pm\} \qquad \text{and} \qquad S^{m,h}_0 := \{\chi \in S^{m,h} : \chi\,|_{\partial^\pm_2\Omega} = 0\}.$$

In addition, we can define the standard finite element space of vector-valued, piecewise linear functions:

$$\underline{Q}^{m,h} := [S^{m,h}]^d,$$

as well as

$$\underline{\widehat{Q}}^{m,h} := \{\vec{q} \in \underline{Q}^{m,h} : \int_{\Omega^{m,h}_+} \vec{q} \cdot \vec{z}\, d\mathcal{L}^d = 0 \quad \forall\, \vec{z} \in \underline{\text{RM}}\}.$$

We then propose the following finite element approximation of (3.14a), (3.14b) and (3.14c)-(3.14d): Given Γ^0, a polyhedral approximation of Γ_0, for $m = 0, \ldots, M-1$ find functions $\{\Phi^{m+1}, \vec{U}^{m+1}, E^{m+1}, \vec{X}^{m+1}, \kappa^{m+1}\} \in$

$S^{m,h}_g \times \underline{\widehat{Q}}^{m,h} \times S^{m,h} \times \underline{V}(\Gamma^m) \times W(\Gamma^m)$ such that for all $\psi \in S^{m,h}_0$, $\vec{\zeta} \in \underline{Q}^{m,h}$, $\xi \in S^{m,h}$, $\vec{\eta} \in \underline{V}(\Gamma^m)$, $\chi \in W(\Gamma^m)$

$$\int_{\Omega^{m,h}_+} \nabla \Phi^{m+1} \cdot \nabla \psi \, d\mathcal{L}^d = 0\,, \tag{4.2a}$$

$$\int_{\Omega^{m,h}_+} \mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{U}^{m+1}) : \underline{\underline{\mathcal{E}}}(\vec{\zeta}) \, d\mathcal{L}^d = \int_{\partial\Omega} \vec{g} \cdot \vec{\zeta} \, d\mathcal{H}^{d-1}\,, \tag{4.2b}$$

$$\int_{\Omega^{m,h}_+} E^{m+1}\, \xi \, d\mathcal{L}^d = \int_{\Omega^{m,h}_+} E(\vec{U}^{m+1})\, \xi \, d\mathcal{L}^d\,, \tag{4.2c}$$

$$\begin{aligned} &\left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi\, \vec{\nu}^m \right\rangle^h_{\Gamma^m} - \alpha_1 \, \langle \nabla_s \, \kappa^{m+1}, \nabla_s \, \chi \rangle_{\Gamma^m} \\ &\qquad = - \langle \nabla_s \, (\pi^m \, [\alpha_2 \, \Phi^{m+1} + \alpha_3 \, E^{m+1}]), \nabla_s \, \chi \rangle_{\Gamma^m}\,, \end{aligned} \tag{4.2d}$$

$$\langle \kappa^{m+1} \, \vec{\nu}^m, \vec{\eta} \rangle^h_{\Gamma^m} + \langle \nabla_s \, \vec{X}^{m+1}, \nabla_s \, \vec{\eta} \rangle_{\Gamma^m} = 0\,, \tag{4.2e}$$

where $\pi^m : C(\Gamma^m, \mathbb{R}) \to W(\Gamma^m)$ is the standard interpolation operator at the nodes $\{\vec{q}^m_k\}_{k=1}^{K^m_\Gamma}$. We see from (4.2c) that E^{m+1} is defined as the $L^2(\Omega^{m,h}_+)$-projection of $E(\vec{U}^{m+1})$, which is a discontinuous function, onto the space of piecewise linear functions $S^{m,h}$. In addition, the vector-valued function $\vec{g}$ in the right-hand side of (4.2b) denotes a traction boundary condition. For simplicity we consider $\vec{g} = \underline{\underline{S}}\, \vec{\nu}_{\partial\Omega} = \mathcal{C}\, \underline{\underline{S}}^* \, \vec{\nu}_{\partial\Omega}$, recall (3.8). We note that (4.2a) (resp., (4.2b)) is a standard finite element approximation of (3.14a) (resp., (3.14b)), while (4.2d)-(4.2e) for the case $\alpha_2 = \alpha_3 = 0$ collapse to the scheme (2.17a)-(2.17b) for the geometric evolution law of *surface diffusion*. Here we note that the approach used in (4.2a) has also been considered in [7], where an *unfitted* finite element approximation for dendritic crystal growth with thermal convection was studied.

An alternative approximation of (3.14a) would replace $\Omega^{m,h}_+$ in (4.2a) by Ω^m_+, the exterior of Γ^m. However, the implementation of this variant would need the computation of $\mathcal{L}^d(\Omega^m_+ \cap o)$ for all $o \in \mathcal{T}^m_+$, which means that (4.2a) is far more practical. Finally, we observe that a more sophisticated *unfitted* approach has been studied in [8] for solving elliptic partial differential equations with Neumann data on a curved boundary. In that paper, Neumann boundary conditions are prescribed on a fixed curved boundary, which is assumed to be sufficiently smooth, and by considering integrals over partial bulk elements the authors are able to prove optimal convergence rates. Once again, our approach in (4.2a) is more practical, and by using local mesh refinement close to Γ^m we are able to observe good convergence properties in practice; see § 4.5 for details.

Let us recall the assumption $(\mathcal{A})$ from § 2.2. In particular, we note that Γ^m, which approximates the closed surface $\Gamma(t_m)$, $m = 0, \dots, M$, easily verifies $(\mathcal{A})$, since it is a surface without self-intersections. We are now ready to prove existence and uniqueness of the discrete solution to the system (4.2a)-(4.2e).

Theorem 4.1. *Let the assumption* $(\mathcal{A})$ *from § 2.2 hold. Then there exists a unique solution* $\{\Phi^{m+1}, \vec{U}^{m+1}, E^{m+1}, \vec{X}^{m+1}, \kappa^{m+1}\} \in S^{m,h}_g \times \underline{\widehat{Q}}^{m,h} \times S^{m,h} \times \underline{V}(\Gamma^m) \times W(\Gamma^m)$ *to the system* (4.2a)-(4.2e).

Proof. We first notice that the equations for Φ^{m+1}, $\vec{U}^{m+1}$, E^{m+1} and $\{\vec{X}^{m+1}, \kappa^{m+1}\}$ decouple. The existence of a unique solution for (4.2a) is trivial. As (4.2b) is a linear finite dimensional system, existence of $\vec{U}^{m+1}$ follows from uniqueness. It follows from (3.5) and a Korn's inequality, see e.g. [102, p. 79], that

$$\begin{aligned}\int_{\Omega_+^{m,h}} \mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{U}) : \underline{\underline{\mathcal{E}}}(\vec{U})\, d\mathcal{L}^d &\geqslant m_{\mathcal{C}} \int_{\Omega_+^{m,h}} \underline{\underline{\mathcal{E}}}(\vec{U}) : \underline{\underline{\mathcal{E}}}(\vec{U})\, d\mathcal{L}^d \\ &\geqslant C\, \|\vec{U}\|^2_{\underline{H}^1(\Omega_+^{m,h})} \quad \forall\, \vec{U} \in \underline{\widehat{Q}}^{m,h}.\end{aligned}$$

Hence we have existence and uniqueness of $\vec{U}^{m+1} \in \underline{\widehat{Q}}^{m,h}$ solving (4.2b). The existence of a unique solution for (4.2c) is trivial, since E^{m+1} is the $L^2(\Omega_+^{m,h})$-projection of $E(\vec{U}^{m+1})$ onto $S^{m,h}$. Given Φ^{m+1} and E^{m+1}, the existence of a solution $\{\vec{X}^{m+1}, \kappa^{m+1}\}$ for (4.2d)-(4.2e) is guaranteed by Theorem 2.6, since (4.2d)-(4.2e) is an invertible linear system completely analogous to (2.17a)-(2.17b), with an additional right-hand side term depending only on Φ^{m+1} and E^{m+1}. ✓

4.2.1 *Extension to the case of a surface with boundary intersections*

So far we have considered the case of Γ^m being a $(d-1)$-dimensional *polyhedral* surface without boundary. It is straightforward to generalise the scheme (4.2a)-(4.2e) to the case of a surface with boundary intersections, where we need to apply appropriate boundary conditions. In our simulations we will always prescribe *free-slip* conditions for the displacement $\delta\vec{X}^{m+1}$. On introducing the space

$$\underline{V}_\partial(\Gamma^m) := \{\vec{\chi} \in \underline{V}(\Gamma^m) : \vec{\chi} \cdot \vec{\nu}_{\partial\Omega} = 0 \text{ on } \partial\Gamma^m\}, \tag{4.3}$$

the generalisation of the system (4.2a)-(4.2e) to the case of a surface with boundary intersections can be stated as follows: Given Γ^0, a polyhedral approximation of Γ_0, for $m = 0, \ldots, M-1$ find functions $\{\Phi^{m+1}, \vec{U}^{m+1}, E^{m+1}, \delta\vec{X}^{m+1}, \kappa^{m+1}\} \in S_g^{m,h} \times \widehat{\underline{Q}}^{m,h} \times S^{m,h} \times \underline{V}_\partial(\Gamma^m) \times W(\Gamma^m)$, where $\vec{X}^{m+1} := \vec{X}^m + \delta\vec{X}^{m+1}$, such that for all $\psi \in S_0^{m,h}$, $\vec{\zeta} \in \underline{Q}^{m,h}$, $\xi \in S^{m,h}$ equations (4.2a)-(4.2c) hold and for all $\vec{\eta} \in \underline{V}_\partial(\Gamma^m)$, $\chi \in W(\Gamma^m)$

$$\left\langle \frac{\delta\vec{X}^{m+1}}{\tau_m}, \chi\,\vec{\nu}^m \right\rangle_{\Gamma^m}^h - \alpha_1 \left\langle \nabla_s\, \kappa^{m+1}, \nabla_s\, \chi\right\rangle_{\Gamma^m}$$
$$= -\left\langle \nabla_s \left(\pi^m \left[\alpha_2\, \Phi^{m+1} + \alpha_3\, E^{m+1}\right]\right), \nabla_s\, \chi\right\rangle_{\Gamma^m}, \tag{4.4a}$$
$$\left\langle \kappa^{m+1}\, \vec{\nu}^m, \vec{\eta}\right\rangle_{\Gamma^m}^h + \left\langle \nabla_s\, \vec{X}^{m+1}, \nabla_s\, \vec{\eta}\right\rangle_{\Gamma^m} = 0\,. \tag{4.4b}$$

It is worth mentioning that in (4.4a)-(4.4b) a boundary condition for $\partial\Gamma^m$ is formulated weakly. Precisely, on recalling the integration by parts

$$-\int_{\Gamma^m} \triangle_s \vec{x} \cdot \vec{\eta}\; d\mathcal{H}^{d-1} = \int_{\Gamma^m} \nabla_s\, \vec{x} \cdot \nabla_s\, \vec{\eta}\; d\mathcal{H}^{d-1} - \int_{\partial\Gamma^m} \vec{\mu}^m \cdot \vec{\eta}\; d\mathcal{H}^{d-2}, \tag{4.5}$$

we note that (4.4b) weakly enforces $\vec{\mu}^m$ to be orthogonal to any test function $\vec{\eta}$ on $\partial\Gamma^m$. Given the definition of the space $\underline{V}_\partial(\Gamma^m)$ in (4.3), which includes all the test functions which are orthogonal to $\vec{\nu}_{\partial\Omega}$ on $\partial\Gamma^m$, we then conclude that (4.5) enforces the conormal $\vec{\mu}^m$ to be in the direction of $\vec{\nu}_{\partial\Omega}$. Therefore, a 90°-angle condition is weakly enforced for Γ^m at the outer boundary $\partial\Omega$.

The scheme (4.4a)-(4.4b), for $\alpha_2 = \alpha_3 = 0$, was first introduced in [15], where the authors presented a variational formulation for the evolution of surface clusters in $\mathbb{R}^3$ by *mean curvature flow*, *surface diffusion* and their anisotropic variants. The proof of existence, uniqueness and stability, in the case of $\alpha_2 = \alpha_3 = 0$, for a solution to (4.4a)-(4.4b), is almost identical to the one for Theorems 2.11 and 2.12, see [15, Thms. 4.3 and 4.4] for more details.

4.2.2 *Semidiscrete continuous-in-time approximation*

It is worthwhile to consider a continuous-in-time semidiscrete version of our fully discrete scheme (4.2a)-(4.2e). To this end, we introduce the following definitions, where we assume that $\Gamma^h(t)$ is a polyhedral approximation of $\Gamma(t)$. Let $\Omega^h_-(t)$ and $\Omega^h_+(t)$ be the interior and the exterior of $\Gamma^h(t)$, respectively. We can then define $\mathcal{T}^h_+(t) := \{o \in \mathcal{T}^h : o \cap \Omega^h_+(t) \neq \emptyset\}$, and $\Omega^{h,h}_+(t) := \bigcup_{o \in \mathcal{T}^h_+(t)} \overline{o}$. In addition, we define

$$\begin{aligned}
S^{h,h}_g(t) &:= \{\chi \in C(\Omega^{h,h}_+(t)) : \chi\,|_o \text{ is linear } \forall\, o \in \mathcal{T}^h_+(t),\ \chi\,|_{\partial^\pm_2\Omega} = g^\pm\},\\
S^{h,h}_0(t) &:= \{\chi \in C(\Omega^{h,h}_+(t)) : \chi\,|_o \text{ is linear } \forall\, o \in \mathcal{T}^h_+(t),\ \chi\,|_{\partial^\pm_2\Omega} = 0\},\\
\underline{Q}^{h,h}(t) &:= \{\vec{q} \in C(\Omega^{h,h}_+(t)), \mathbb{R}^d) : \vec{q}\,|_o \text{ is linear } \forall\, o \in \mathcal{T}^m_+\},\\
\underline{\widehat{Q}}^{h,h}(t) &:= \{\vec{q} \in \underline{Q}^{h,h}(t) : \textstyle\int_{\Omega^{h,h}_+} \vec{q}\cdot\vec{z}\,d\mathcal{L}^d = 0 \quad \forall\, \vec{z} \in \underline{\mathrm{RM}}\}.
\end{aligned}$$

as well as the inner products $\langle\cdot,\cdot\rangle_{\Gamma^h(t)}$ and $\langle\cdot,\cdot\rangle^h_{\Gamma^h(t)}$ on $\Gamma^h(t)$, analogously to (2.11) and (2.12).

The semidiscrete variant of (4.2a)-(4.2e) can then be formulated as follows. Let $\Gamma^h(0)$ be given; then for $t \in (0,T)$ find $\Gamma^h(t)$, described by the identity function $\vec{X}^h(t) \in \underline{V}(\Gamma^h(t))$, $\Phi^h(t) \in S^{h,h}_g(t)$, $\vec{U}^h(t) \in \underline{\widehat{Q}}^{h,h}(t)$, $E^h(t) \in S^{h,h}(t)$ and $\kappa^h(t) \in W(\Gamma^h(t))$, such that for almost all times $t \in (0,T)$ it holds that for all $\psi \in S^{h,h}_0(t)$, $\vec{\zeta} \in \underline{Q}^{h,h}$, $\xi \in S^{h,h}(t)$, $\chi \in W(\Gamma^h(t))$, $\vec{\eta} \in \underline{V}(\Gamma^h(t))$

$$\int_{\Omega^{h,h}_+(t)} \nabla\Phi^h \cdot \nabla\psi\, d\mathcal{L}^d = 0\,, \tag{4.6a}$$

$$\int_{\Omega^{h,h}_+(t)} \mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{U}^h) : \underline{\underline{\mathcal{E}}}(\vec{\zeta})\, d\mathcal{L}^d = \int_{\partial\Omega} \vec{g}\cdot\vec{\zeta}\, d\mathcal{H}^{d-1}\,, \tag{4.6b}$$

$$\int_{\Omega^{h,h}_+} E^h\,\xi\, d\mathcal{L}^d = \int_{\Omega^{h,h}_+} E(\vec{U}^h)\,\xi\, d\mathcal{L}^d\,, \tag{4.6c}$$

$$\begin{aligned}
\langle \vec{X}^h_t, \chi\,\vec{\nu}^h\rangle^h_{\Gamma^h(t)} - \alpha_1\langle\nabla_s\,\kappa^h, \nabla_s\,\chi\rangle_{\Gamma^h(t)} &=\\
&- \langle\nabla_s\,\pi^h[\alpha_2\,\Phi^h + \alpha_3\,E^h]), \nabla_s\,\chi\rangle_{\Gamma^h(t)}\,,
\end{aligned} \tag{4.6d}$$

$$\langle\kappa^h\,\vec{\nu}^h, \vec{\eta}\rangle^h_{\Gamma^h(t)} + \langle\nabla_s\,\vec{X}^h, \nabla_s\,\vec{\eta}\rangle_{\Gamma^h(t)} = 0\,, \tag{4.6e}$$

where $\vec{\nu}^h(t)$ is the unit normal to $\Gamma^h(t)$ and $\pi^h : C(\Gamma^h, \mathbb{R}) \to W(\Gamma^h)$ is the standard interpolation operator at the nodes $\{\vec{q}^h_k\}^{K^h_\Gamma}_{k=1}$. Note that E^h is the continuous-in-time semidiscrete counterpart of E^{m+1}, recall (4.2c).

Let us define $\vec{h}_k(t)$ as in Theorem 2.8. We are now in a position to prove an exact volume conservation property for (4.6a)-(4.6e), as well as an equidistribution property.

THEOREM 4.2. *Let* $\{\Phi^h,\ \vec{U}^h,\ E^h,\ \vec{X}^h,\ \kappa^h\}(t) \in S^{h,h}_g(t) \times \underline{\widehat{Q}}^{h,h} \times S^{h,h}(t) \times \underline{V}(\Gamma^h) \times W(\Gamma^h)$ *be a solution of* (4.6a)-(4.6e). *Then it holds that*

$$\frac{d}{dt}\,\mathcal{L}(\Omega^h_-(t)) = 0\,. \tag{4.7}$$

Moreover, for the case $d = 2$ *it holds that*

$$|\vec{h}_k(t)| = |\vec{h}_{k-1}(t)| \quad \textit{if} \quad \vec{h}_k(t) \not\parallel \vec{h}_{k-1}(t) \quad k = 1, \ldots, K^h_\Gamma\,. \tag{4.8}$$

Proof. Similarly to what we discussed in § 2.2.1, the exact conservation of the enclosed volume can be shown by choosing $\chi \equiv 1$ in (4.6d) and taking into account (2.12). Then it holds that

$$0 = \langle \vec{X}^h_t, \vec{\nu}^h \rangle^h_{\Gamma^h(t)} = \int_{\Gamma^h(t)} \vec{X}^h_t \cdot \vec{\nu}^h \, d\mathcal{H}^{d-1} = \frac{d}{dt}\,\mathcal{L}^d(\Omega^h_-(t))\,, \tag{4.9}$$

which is the desired result (4.7). (4.8) can be proven by applying Theorem 2.8 to the moving interface $\Gamma^h(t)$. ✓

REMARK 4.3. *For the case* $d = 3$, *we can apply Remark 2.10 to the polyhedral surface* $\Gamma^h(t)$, *which therefore exhibits good mesh properties. We note that it is not possible to extend the conservation property* (4.9) *to the fully discrete scheme* (4.2a)-(4.2e). *However, in all our numerical experiments the volume of* Ω^m_- *is preserved up to a very small tolerance. In fact, when the maximal time step size* τ *converges to zero, we observe that the relative volume loss for simulations with* (4.2a)-(4.2e) *tends to zero.*

4.2.3 *Multi-component interfaces*

For ease of presentation, so far we have restricted ourselves to the case of Γ^m being a single $(d-1)$-dimensional *polyhedral* surface. As we discussed in § 2.3, we can extend our approximation (4.2a)-(4.2e) to the case where Γ^m is given by a family of closed surfaces, say $\Gamma^m = \bigcup_{l=1}^{L} \Gamma^m_l$, with $\Gamma^m_l = \bigcup_{j=1}^{J^m_l} \overline{\sigma^{m,l}_j}$, where $\{\sigma^{m,l}_j\}_{j=1}^{J^m_l}$ is a family of mutually disjoint open $(d-1)$-simplices with vertices $\{\vec{q}^{m,l}_k\}_{k=1}^{K^m_l}$, $l = 1, \ldots, L$. The necessary finite element spaces for the approximation of the position vector $\vec{X}^{m+1}$ and curvature κ^{m+1} have been introduced in (2.50). The natural extension of the system (4.2a)-(4.2e) to the multiple-component case is then given by: Find $\{\Phi^{m+1}$, $\vec{U}^{m+1}$ $E^{m+1}\} \in S^{m,h}_g \times \underline{\widehat{Q}}^{m,h} \times S^{m,h}$ such that (4.2a)-(4.2c) hold and then, for $l = 1,\ldots,L$, find $\{\vec{X}^{m+1}_l, \kappa^{m+1}_l\} \in \underline{V}^l(\Gamma^m_l) \times W^l(\Gamma^m_l)$ such that for all $\vec{\eta} \in \underline{V}^l(\Gamma^m_l)$, $\chi \in W^l(\Gamma^m_l)$

$$\left\langle \frac{\vec{X}^{m+1}_l - \vec{X}^m_l}{\tau_m}, \chi\, \vec{\nu}^m_l \right\rangle^h_{\Gamma^m_l} - \alpha_1 \left\langle \nabla_s\, \kappa^{m+1}_l, \nabla_s\, \chi \right\rangle_{\Gamma^m_l}$$
$$= - \left\langle \nabla_s\, (\pi^m_l\, [\alpha_2\, \Phi^{m+1} + \alpha_3\, E^{m+1}]), \nabla_s\, \chi \right\rangle_{\Gamma^m_l}\,, \tag{4.10a}$$

$$\langle \kappa^{m+1}_l\, \vec{\nu}^m_l, \vec{\eta} \rangle^h_{\Gamma^m_l} + \langle \nabla_s\, \vec{X}^{m+1}_l, \nabla_s\, \vec{\eta} \rangle_{\Gamma^m_l} = 0\,, \tag{4.10b}$$

where recall that $\pi^m_l : C(\Gamma^m_l, \mathbb{R}) \to W^l(\Gamma^m)$ is the standard interpolation operator at the nodes $\{\vec{q}^{m,l}_k\}_{k=1}^{K^m_l}$.

Let $(\mathcal{A}_l)$ be the analogue of assumption $(\mathcal{A})$ for the closed surface Γ^m_l, $l = 1,\ldots,L$. The we can generalise Theorem 4.1 as follows.

Theorem 4.4. *Let the assumptions* $(\mathcal{A}_l)$, $l = 1,\ldots,L$ *hold. Then there exist unique solutions* $\Phi^{m+1} \in S^{m,h}_g$ *to* (4.2a), $\vec{U}^{m+1} \in \underline{\widehat{Q}}^{m,h}$ *to* (4.2b), $E^{m+1} \in S^{m,h}$ *to* (4.2c), *and* $\{\vec{X}^{m+1}_l, \kappa^{m+1}_l\} \in \underline{V}^l(\Gamma^m) \times W^l(\Gamma^m)$ *to* (4.10a)-(4.10b).

Proof. As the systems for $\{\vec{X}^{m+1}_l, \kappa^{m+1}_l\}$ decouple for each $l = 1,\ldots,L$, we immediately obtain the desired existence and uniqueness from Theorems 4.1 and 2.15. ✓

Remark 4.5. *For a semidiscrete variant of* (4.2a), (4.2b) *and* (4.10a)-(4.10b), *similarly to* (4.6a)-(4.6e) *we can show that* (4.7) *holds. Moreover, for the case* $d = 2$, *the equidistribution property holds for each* $\Gamma^h_l(t)$ *separately, recall Remark 2.16.*

4.3 solution methods

Due to the special structure of the system (4.2a)-(4.2e), the equations for Φ^{m+1}, $\vec{U}^{m+1}$ and $\{\vec{X}^{m+1}, \kappa^{m+1}\}$ decouple. In practice, we can find the unique solution to (4.2a)-(4.2e) as follows. First we find $\Phi^{m+1} \in S^{m,h}_g$ such that

$$\Theta_m\, \Phi^{m+1} = 0\,, \tag{4.11}$$

where $\Theta_m \in \mathbb{R}^{Z\times Z}$ is the standard stiffness matrix for the Laplacian on $\Omega_+^{m,h}$, i.e.

$$[\Theta_m]_{kl} := \int_{\Omega_+^{m,h}} \nabla\psi_k^m \cdot \nabla\psi_l^m \, d\mathcal{L}^d\,,$$

where $\{\psi_k^m\}_{k=1}^Z$ are the basis functions of the unconstrained finite element space $S^{m,h}$. In the above we have ignored the effect of the Dirichlet boundary conditions. Hence, in practice, Θ_m and the corresponding right-hand side in (4.11) need to be adjusted appropriately in order to include the Dirichlet boundary conditions. For the solution of (4.11) we use the sparse factorisation package UMFPACK.

We proceed with finding $\vec{U}^{m+1}$ such that

$$\Xi_m \, \vec{U}^{m+1} = G^m\,, \tag{4.12}$$

where $\Xi_m \in \mathbb{R}^{L\times L}$ is the matrix of the elasticity operator on $\Omega_+^{m,h}$, i.e.

$$[\Xi_m]_{kl} := \int_{\Omega_+^{m,h}} \mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{\xi}_k^m) : \underline{\underline{\mathcal{E}}}(\vec{\xi}_l^m) \, d\mathcal{L}^d\,,$$

and G^m is the right-hand side involving the boundary forcing term, i.e.

$$[G^m]_k := \int_{\partial\Omega} \vec{g} \cdot \vec{\xi}_k^m \, d\mathcal{H}^{d-1}\,.$$

Here $\{\vec{\xi}_k^m\}_{k=1}^L$ are the basis functions of the finite element space $\underline{Q}^{m,h}$. Therefore, in practice we assemble and solve (4.12) over $\underline{Q}^{m,h}$ and not over $\widehat{\underline{Q}}^{m,h}$. Recalling (3.9), such a square system is not invertible, with the kernel of Ξ_m corresponding to the finite element functions in $\underline{\mathrm{RM}}$. Therefore we make use of the sparse factorisation package SuiteSparseQR, which performs a QR factorisation of the noninvertible system (4.12) and returns a least-square solution; see [42] for details. The solution vector $\vec{U}^{m+1}$ is not guaranteed, in general, to belong to $\widehat{\underline{Q}}^{m,h}$. However, recalling that in the right-hand side of (4.2d) we require only the elastic energy density $\mathrm{E}(\vec{U}^{m+1})$ and not the displacement field $\vec{U}^{m+1}$ itself, there is no need to project $\vec{U}^{m+1}$ onto $\widehat{\underline{Q}}^{m,h}$.

Having obtained Φ^{m+1} from (4.11) and $\vec{U}^{m+1}$ from (4.12), we proceed with solving the equations (4.2d)-(4.2e), which give rise to the following linear system of equations, where we employ the same notation introduced in Section 2.4. Find $\delta\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$ and $\kappa^{m+1} \in W(\Gamma^m)$ such that

$$\begin{pmatrix} \alpha_1\, A_m & -\frac{1}{\tau_m}\vec{N}_m^T \\ \vec{N}_m & \vec{A}_m \end{pmatrix} \begin{pmatrix} \kappa^{m+1} \\ \delta\vec{X}^{m+1} \end{pmatrix} = \begin{pmatrix} f^{m+1} \\ -\,\vec{A}_m\,\vec{X}^m \end{pmatrix}. \tag{4.13}$$

In the above, we have introduced the vector f^{m+1} with entries

$$[f^{m+1}]_k := \langle \nabla_s\,(\pi^h\,[\alpha_2\,\Phi^{m+1} + \alpha_3\,E^{m+1}]), \nabla_s\,\phi_k^m\rangle_{\Gamma^m}\,. \tag{4.14}$$

We refer to Section 2.4 for the discussion about solution strategies for (4.13) and choice of software packages.

REMARK 4.6. *Our analysis immediately carries over to the case where the interface* Γ^m *is a polyhedral surface with boundary intersections, recall § 4.2.1. Specifically, the user needs to adjust the matrix in* (4.13) *and the right-hand side term* (4.14) *in order to include the free-slip boundary conditions, recall* (4.3).

4.4 MESH OPERATIONS

We implemented our finite element approximation (4.2a)-(4.2e) within the framework of the C++-based software DUNE, see [22, 21], and the discretisation module `dune-fem`, see [48]. The computations presented in this chapter have been performed with the help of the ALBERTA grid manager, see [114]. In what follows, we first recall some useful definitions about *affine combinations* in $\mathbb{R}^d$, paying special attention to certain operations required to identify potential intersections between elements of bulk and interface grids. We then describe the mesh refinement strategies used for both bulk (§ 4.4.4) and interface (§ 4.4.5) meshes. Finally, in § 4.4.6 we discuss how to identify an appropriate partition of the bulk triangulation $\mathcal{T}^m$ over which the electric potential and the elastic displacement are to be computed.

4.4.1 *Remarks on affine combinations*

Let as recall some simple, yet useful definitions about *affine combinations* that will be employed at a later stage for the detection of potential intersections between d-dimensional simplices of the bulk grid and $(d-1)$-dimensional simplices of the interface grid.

Given n points $\{\vec{p}_i\}_{i=1}^n \in \mathbb{R}^d$, we define an *affine combination* of these points a point $\vec{p}$ s.t.

$$\vec{p} = \alpha_1 \, \vec{p}_1 + \ldots + \alpha_n \, \vec{p}_n \,,$$

where $\alpha_1, \ldots, \alpha_n$ are scalar coefficients such that $\sum_{i=1}^n \alpha_i = 1$. For example, assuming that the n points are linearly independent, in the case $n = 2$, $\vec{p}$ can be any point on the line passing through $\vec{p}_1$ and $\vec{p}_2$. When $n = 3$, $\vec{p}$ can be any point on the plane passing through $\vec{p}_1$, $\vec{p}_2$ and $\vec{p}_3$. If each α_i is such that $0 \leqslant \alpha_i \leqslant 1$, then the point $\vec{p}$ is a *convex combination* of the points $\vec{p}_1, \ldots, \vec{p}_n$. We can now easily test if a given point $\vec{p}$ belongs to a d-dimensional simplex S in $\mathbb{R}^d$ represented by its $d+1$ extreme points $\{\vec{p}_i\}_{i=0}^d$. We first write the linear system

$$\alpha_0 \, (\vec{p}_0 - \vec{p}_d) + \ldots + \alpha_{d-1} \, (\vec{p}_{d-1} - \vec{p}_d) = \vec{p} - \vec{p}_d \,, \qquad (4.15)$$

which is always solvable, provided S is nondegenerate. Defining $\alpha_d := 1 - \sum_{i=0}^{d-1} \alpha_i$, we call the coefficients $\{\alpha_i\}_{i=0}^d$ the *barycentric coordinates* of the point $\vec{p}$ with respect to the $d+1$ points identifying the d-dimensional simplex S. The point $\vec{p}$ lies within S if and only if $0 \leqslant \alpha_i \leqslant 1$ for all $i = 0, \ldots, d$.

For later use, we now discuss how *barycentric coordinates* can be employed in order to detect whether a $(d-1)$-dimensional simplex of the interface grid Γ^m and a d-dimensional simplex from the bulk triangulation $\mathcal{T}^m$ intersect or not. We investigate the case $d = 2$ in § 4.4.2 and the case $d = 3$ in § 4.4.3.

4.4.2 *Intersections between triangles and segments*

We now analyse whether a triangle $\vec{P}_0\vec{P}_1\vec{P}_2$ is intersected by a segment $\overrightarrow{AB}$, see Figure 4.2.

We first check if either of the vertices of the segment lies within the triangle. To this end, we solve the linear system (4.15) with $\vec{p}_i = \vec{P}_i$, $i = 0, \ldots, 2$, and $\vec{p} = \vec{A}$ (resp., $\vec{B}$) and test if $\{\alpha_i\}_{i=0}^2$ form a convex combination. If neither of the vertices of the segment lies within the triangle, we need to perform further operations. We make use of the barycentric coordinates $\vec{\lambda} = (\alpha_0, \dots, \alpha_2)^T$ of the points $\vec{A}, \vec{B}$ that we just computed at the first step of the algorithm. We now enumerate the three edges of the triangle $\vec{P}_0\vec{P}_1\vec{P}_2$ and check if at least one of them has non-empty intersection with the segment $\overrightarrow{AB}$. Without loss of generality, let us consider the edge $\overrightarrow{P_0P_1}$. We investigate if there is a point of the segment $\overrightarrow{AB}$ that lies on the line passing through $\vec{P}_0$ and $\vec{P}_1$. To this end, we solve the following equation:

$$0 = [\vec{\lambda}_{\vec{B}}]_2 + \alpha\left([\vec{\lambda}_{\vec{A}}]_2 - [\vec{\lambda}_{\vec{B}}]_2\right), \tag{4.16}$$

which guarantees that the barycentric coordinate of the unknown point on the line passing through $\vec{A}$ and $\vec{B}$ with respect to the point $\vec{P}_2$ vanishes. We now solve (4.16) for the scalar unknown α. If the difference $([\vec{\lambda}_{\vec{A}}]_2 - [\vec{\lambda}_{\vec{B}}]_2)$ is not 0 (i.e. $\vec{A}$ and $\vec{B}$ neither are collinear with $\vec{P}_0$ and $\vec{P}_1$ nor lie on a line parallel to $\overrightarrow{P_0P_1}$), there exists only one α. Now check if $0 \leqslant \alpha \leqslant 1$: in such a case, the candidate intersection is a point in the segment $\overrightarrow{AB}$, precisely the point $\vec{Q} := \alpha\,\vec{A} + (1-\alpha)\,\vec{B}$. We now have to test whether this point is inside the triangle. To this end, we compute the barycentric coordinates $\vec{\lambda}_{\vec{Q}} = \alpha\,\vec{\lambda}_{\vec{A}} + (1-\alpha)\vec{\lambda}_{\vec{B}}$ and check if they form a convex combination, i.e. $0 \leqslant [\vec{\lambda}_{\vec{Q}}]_i \leqslant 1$ for $i = 0, \dots, 2$. If $[\vec{\lambda}_{\vec{A}}]_2 = [\vec{\lambda}_{\vec{B}}]_2$, two special cases could occur:

- $[\vec{\lambda}_{\vec{B}}]_2 \neq 0$, so there is no α satisfying (4.16). This happens when $\vec{A}$ and $\vec{B}$ lie on a line parallel to $\overrightarrow{P_0P_1}$, but the four points are not collinear.
- $\vec{A}$ and $\vec{B}$ are collinear with $\vec{P}_0$ and $\vec{P}_1$. Then $[\vec{\lambda}_{\vec{A}}]_2 = [\vec{\lambda}_{\vec{B}}]_2 \equiv 0$, so (4.16) is satisfied for any real α. This does not pose a problem, since we can identify the intersection (if any) analysing the intersection between the segment $\overrightarrow{AB}$ and one of the other edges of the triangle.

If we find no intersection between $\overrightarrow{AB}$ and $\overrightarrow{P_0P_1}$, we consider another edge of the triangle, say $\overrightarrow{P_0P_2}$, and repeat the algorithm. The complete set of routines is summarised in Algorithm 1.

4.4.3 *Intersections between tetrahedra and triangles*

We now analyse how to detect whether a tetrahedron $\vec{P}_0\vec{P}_1\vec{P}_2\vec{P}_3$ is intersected by a triangle $\vec{A}\vec{B}\vec{C}$, see [14, Fig. 4].

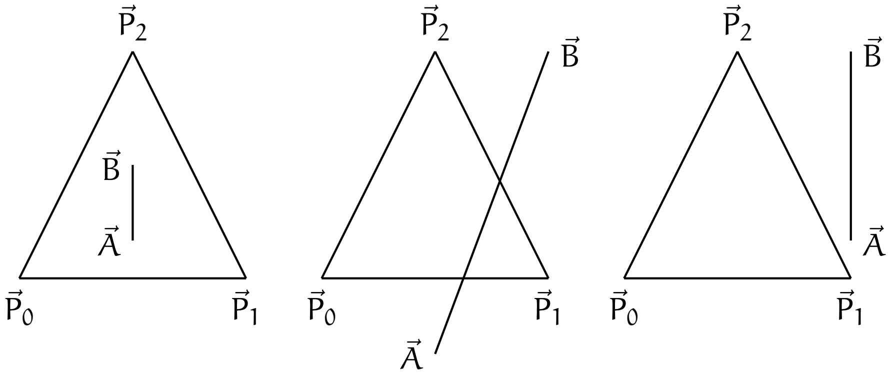

Figure 4.2: Possible cases of intersection between a triangle and a segment.

Algorithm 1: Detecting if a segment $\overrightarrow{Q_1Q_2}$ intersects a triangle $\vec{P}_0\vec{P}_1\vec{P}_2$.

```
Data: Bulk triangle $o = \vec{P}_0\vec{P}_1\vec{P}_2$ and interface segment $\sigma = \overrightarrow{Q_1Q_2}$
Result: Identification of a possible intersection between $o$ and $\sigma$
begin
    for $i = 1, 2$ do
        if $Q_i \in o$ then
            return TRUE;
        end
    end
    for $i = 1, 2, 3$ do
        $e_i \longleftarrow$ edge $i$ of triangle $o$;
        if $\sigma$ intersects $e_i$ then
            return TRUE;
        end
    end
    return FALSE;
end
```

We first check if any of the vertices of the triangle lie within the tetrahedron. To this end, we solve the linear system (4.15) with $\vec{p}_i = \vec{P}_i$, $i = 0, \dots, 3$, and $\vec{p} = \vec{A}$ (resp., $\vec{B}, \vec{C}$) and test if $\{\alpha_i\}_{i=0}^{3}$ form a convex combination. If none of the vertices of the triangle lies within the tetrahedron, we need to perform further operations.

We first analyse the case of an intersection between an edge of the triangle and a face of the tetrahedron. In the following, we make use of the barycentric coordinates $\vec{\lambda} = (\alpha_0, \dots, \alpha_3)^T$ of the points $\vec{A}, \vec{B}, \vec{C}$ that we previously computed at the first step of the algorithm. Let us now investigate, without loss of generality, the case of the segment $\overrightarrow{AB}$ and the face $\vec{P}_0\vec{P}_1\vec{P}_2$. We want to see if the segment intersects the triangular face (the intersection could be empty, a point or infinitely many points). We first need to check if there is a point of the segment $\overrightarrow{AB}$ that lies on the plane of the triangle $\vec{P}_0\vec{P}_1\vec{P}_2$. To this end, we solve the following equation:

$$0 = [\vec{\lambda}_{\vec{B}}]_3 + \alpha \left([\vec{\lambda}_{\vec{A}}]_3 - [\vec{\lambda}_{\vec{B}}]_3 \right), \tag{4.17}$$

which guarantees that the barycentric coordinate of the unknown point on the line passing through $\vec{A}$ and $\vec{B}$ with respect to the point $\vec{P}_3$ vanishes. We now solve (4.17) for the scalar unknown α. If the difference $([\vec{\lambda}_{\vec{A}}]_3 - [\vec{\lambda}_{\vec{B}}]_3)$ is not 0 (i.e. $\vec{A}$ and $\vec{B}$ lie neither in a plane parallel to $\vec{P}_0\vec{P}_1\vec{P}_2$ nor in the plane of $\vec{P}_0\vec{P}_1\vec{P}_2$ itself), there exists only one α. Now check if $0 \leqslant \alpha \leqslant 1$: in such a case, the intersection of the line and the plane is a point in the segment $\overrightarrow{AB}$, precisely the point $\vec{Q} := \alpha \vec{A} + (1 - \alpha) \vec{B}$. We now have to test whether this point is inside the triangle. To this end, we compute the barycentric coordinates $\vec{\lambda}_{\vec{Q}} = \alpha \vec{\lambda}_{\vec{A}} + (1 - \alpha) \vec{\lambda}_{\vec{B}}$ and check if they form a convex combination, i.e. $0 \leqslant [\vec{\lambda}_{\vec{Q}}]_i \leqslant 1$ for $i = 0, \dots, d$, see Figure 4.3 (left). If $[\vec{\lambda}_{\vec{A}}]_3 = [\vec{\lambda}_{\vec{B}}]_3$, two special cases could occur:

- $[\vec{\lambda}_{\vec{B}}]_3 \neq 0$, so there is no α satisfying (4.17). This happens when $\vec{A}$ and $\vec{B}$ lie on a plane parallel to the triangle $\vec{P}_0\vec{P}_1\vec{P}_2$, but *not* containing the triangle itself, see Figure 4.3 (centre).

- $\vec{A}$ and $\vec{B}$ lie on the plane containing the triangle $\vec{P}_0\vec{P}_1\vec{P}_2$. Then $[\vec{\lambda}_{\vec{A}}]_3 = [\vec{\lambda}_{\vec{B}}]_3 \equiv 0$, so (4.17) is satisfied for any real α, see Figure 4.3 (right). This does not pose a problem, since we can identify the intersection (if any) analysing the intersection between the segment $\overrightarrow{AB}$ and one of the other triangular faces of the tetrahedron.

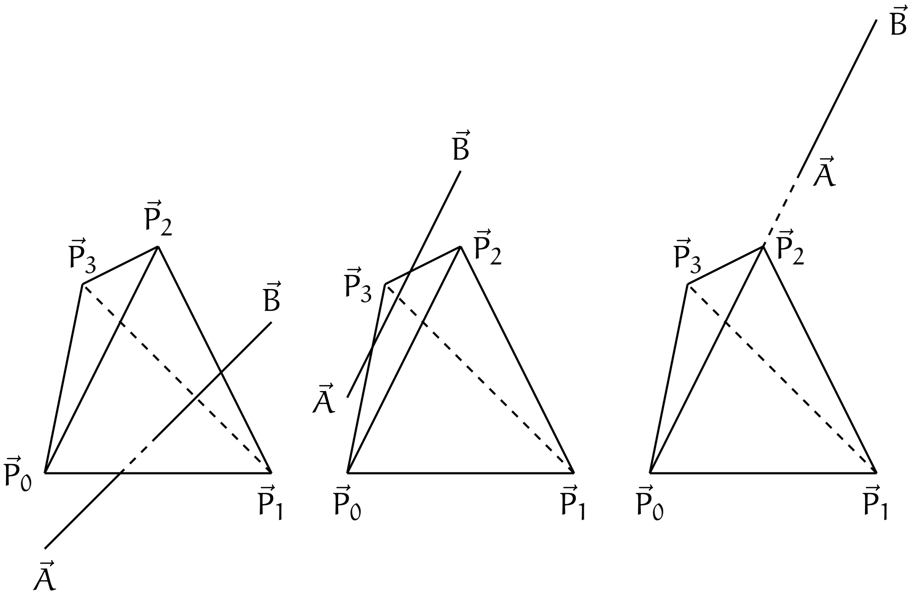

Figure 4.3: Possible cases of intersection between an edge of a triangle and a triangular face of a tetrahedron.

We now investigate the case of an intersection between an edge of the tetrahedron and the triangle. Without loss of generality, we analyse the case of the segment $\overrightarrow{P_0P_1}$ and the triangle $\vec{A}\vec{B}\vec{C}$. Recall that we have already computed $\vec{\lambda}_{\vec{A}}, \vec{\lambda}_{\vec{B}}$ and $\vec{\lambda}_{\vec{C}} \in \mathbb{R}^4$. If $\overrightarrow{P_0P_1}$ intersects $\vec{A}\vec{B}\vec{C}$, the point of intersection will be a convex combination of $\vec{P}_0$ and $\vec{P}_1$, so the barycentric coordinates w.r.t. $\vec{P}_2$ and $\vec{P}_3$ will be exactly 0. This yields

$$\alpha\,[\vec{\lambda}_{\vec{A}}]_2 + \beta\,[\vec{\lambda}_{\vec{B}}]_2 + (1-\alpha-\beta)\,[\vec{\lambda}_{\vec{C}}]_2 = 0\,, \tag{4.18}$$

$$\alpha\,[\vec{\lambda}_{\vec{A}}]_3 + \beta\,[\vec{\lambda}_{\vec{B}}]_3 + (1-\alpha-\beta)\,[\vec{\lambda}_{\vec{C}}]_3 = 0\,. \tag{4.19}$$

The system (4.18)-(4.19) is a 2×2 linear system, whose solution (if any) represents a point on the line on which the segment $\overrightarrow{P_0P_1}$ lies. If the system is invertible, we can compute the unique α, β. If it holds that $0 \leqslant \alpha, \beta, 1 - \alpha - \beta \leqslant 1$, then the candidate intersection point is inside the triangle, but *not* necessarily on the segment $\overrightarrow{P_0P_1}$. Then, on denoting $\vec{\lambda}_Q := \alpha\vec{\lambda}_A + \beta\,\vec{\lambda}_B + (1-\alpha-\beta)\vec{\lambda}_C$, we further check if $0 \leqslant [\vec{\lambda}_{\vec{Q}}]_{0,1} \leqslant 1$. If all these conditions are satisfied, the segment and the triangle intersect, see Figure 4.4. If the system (4.18)-(4.19) is singular, the intersection (if any) can be identified by analysing another face of the tetrahedron, keeping the same triangle from the interface grid. The complete algorithm is described in Algorithm 2.

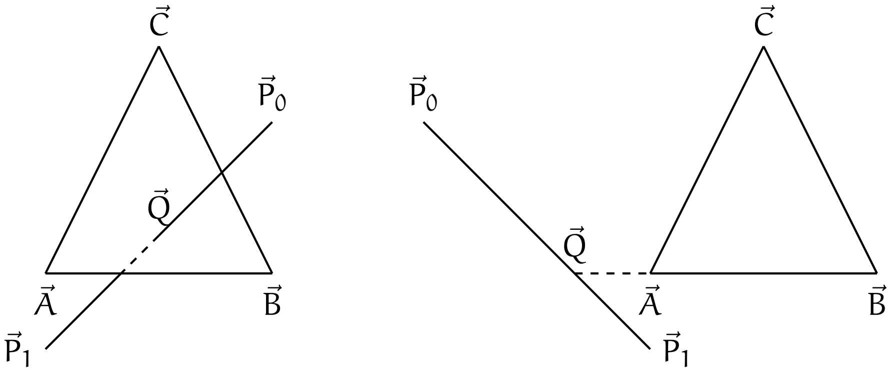

Figure 4.4: Possible cases of intersection between an edge of a tetrahedron and a triangle.

4.4.4 *Bulk mesh adaptation*

Our *unfitted* finite element approximation (4.2a)-(4.2e) is based on triangulations $\mathcal{T}^m$ of the whole domain Ω, which vary in time. Here the aim is to use an adaptive mesh for Ω, where we resolve the regions close to Γ^m much finer than far away from the interface. The exact procedure for the refinement of the bulk grid is detailed below, where we follow [104, § 4.1]. It is worth mentioning that the DUNE software stores any mesh as a *hierarchical* grid (see A.2 for details), which makes traversing through successive levels of refinement easy. This feature will be used in § 4.4.6, where we investigate the selection of the elements that belong to $\mathcal{T}^m_+$, recall (4.1).

Given a polyhedral approximation Γ^m, $m \geqslant 0$, of the interface, we employ the following mesh adaptation strategy for the bulk mesh triangulation $\mathcal{T}^m$. Note that the same strategy has been used in [14] for an *unfitted* finite element approximation of anisotropic solidification problems. It returns a

Algorithm 2: Detecting if a triangle $\vec{Q}_0\vec{Q}_1\vec{Q}_2$ intersects a tetrahedron $\vec{P}_0\vec{P}_1\vec{P}_2\vec{P}_3$.

```
Data: Bulk tetrahedron $o = \vec{P}_0\vec{P}_1\vec{P}_2\vec{P}_3$ and interface triangle
      $\sigma = \vec{Q}_0\vec{Q}_1\vec{Q}_2$
Result: Identification of a possible intersection between o and σ
begin
    for $i = 1, 2, 3$ do
        if $Q_i \in o$ then
            return TRUE;
        end
    end
    for $i = 1, 2, 3$ do
        $e_i \longleftarrow$ edge i of the triangle σ;
        for $j = 1, \ldots, 4$ do
            $f_j \longleftarrow$ face j of the tetrahedron o;
            if $e_i$ *intersects* $f_j$ then
                return TRUE;
            end
        end
    end
    for $i = 1, \ldots, 6$ do
        $e_i \longleftarrow$ edge i of the tetrahedron o;
        if $e_i$ *intersects* σ then
            return TRUE;
        end
    end
    return FALSE;
end
```

fine mesh around Γ^m and a coarse mesh further away from it. In particular, given two integer parameters $N_f > N_c$, we set:

$$h_f = \frac{2\,L_d}{N_f} \quad \text{and} \quad h_c = \frac{2\,L_d}{N_c}\,, \tag{4.20a}$$

$$\mathrm{vol}_f = \frac{h_f^d}{d!} \quad \text{and} \quad \mathrm{vol}_c = \frac{h_c^d}{d!}\,, \tag{4.20b}$$

that is, for $d = 3$, vol_f denotes the volume of a tetrahedron with three right-angled and isosceles faces with side length h_f, while for $d = 2$ it denotes the area of a right-angled and isosceles triangle with side length h_f, and similarly for vol_c.

Now starting with the triangulation $\mathcal{T}^{m-1}$ from the previous time step, our aim is to obtain $\mathcal{T}^m$. Here for convenience we define $\mathcal{T}^{-1}$ to be the initial, uniform partitioning of Ω into elements of mesh size h_c. In the following and throughout the thesis, $\mathcal{T}^{-1}$ will also be denoted as the initial *macro-triangulation*. We are now ready to present the iterative algorithm for obtaining $\mathcal{T}^m$. First any element $o^{m-1} \in \mathcal{T}^{m-1}$ satisfying $\mathcal{L}^d(o^{m-1}) \geqslant 2\,\mathrm{vol}_f$ and $o^{m-1} \cap \Gamma^m \neq \emptyset$ is marked for refinement. In addition, any element satisfying $\mathcal{L}^d(o^{m-1}) \geqslant 2\,\mathrm{vol}_f$, for which a direct neighbour intersects Γ^m, is also marked for refinement. Similarly, an element that is not marked for refinement is marked for coarsening if it satisfies $\mathcal{L}^d(o^{m-1}) \leqslant \frac{1}{2}\,\mathrm{vol}_c$ and $o^{m-1} \cap \Gamma^m = \emptyset$. Note that starting with an initial triangulation of uniform width h_c is crucial in order to have elements far from the interface with the desired mesh size. If those elements were too coarse in $\mathcal{T}^{m-1}$, they would not be marked for refinement. In fact, the strategy outlined above considers for refinement only an element cut by Γ^m or a neighbour of a cut element. Now all the elements marked for refinement are halved into two smaller elements with the help of a simple bisectioning procedure, see [114]. In order to avoid hanging nodes, this will in general lead to the refinement of elements that were not originally marked for refinement. Similarly, an element that is marked for coarsening is coarsened only if all of its neighbouring elements are marked for coarsening as well. For more details on the refining and coarsening itself, we refer to [114]. We designed the macro-triangulation $\mathcal{T}^{-1}$ in such a way that, after every refinement step performed by the ALBERTA grid manager, the bulk mesh will still be composed of right-angled, isosceles triangles or of tetrahedra with such triangular faces; see Figure 4.5 for $d = 2$.

This marking and refinement process is repeated until no more elements are required to be refined or coarsened. In practice, we observed that only at the first time step, $m = 0$, are more than one of the described refinement cycles needed.

4.4.5 *Interface mesh adaptation*

In theory it is possible to use local mesh refinement also for the discrete interface Γ^m, which is totally independent from $\mathcal{T}^m$. However, the good mesh properties shown in Theorem 4.2 and Remark 4.3 guarantee that in practice no refinement of Γ^m is necessary. In fact, the vertices of Γ^m are in general very well distributed, so that the user does not have to perform any mesh smoothing operation.

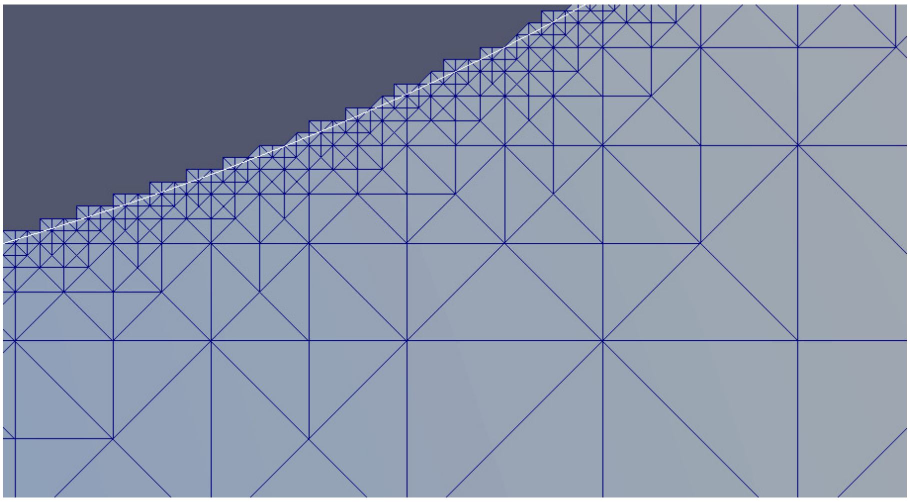

Figure 4.5: Portion of the computational domain showing the bulk mesh close to the discrete interface (from [104, Fig. 3]). The macro-triangulation $\mathcal{T}^{-1}$ is designed in such a way that, after every refinement step performed by the ALBERTA grid manager, the bulk mesh is still composed of right-angled, isosceles triangles.

4.4.6 *Definition of the bulk region*

A crucial aspect of the *unfitted* approach is the identification of a suitable region $\Omega_+^{m,h}$ where the approximations Φ^{m+1} and $\vec{U}^{m+1}$ to the electric potential and the elastic stress are to be computed. As described in § 4.2, our method is based on an adapted triangulation $\mathcal{T}^m$ of the whole domain Ω. In order to define the submesh $\mathcal{T}_+^m$, which defines the computational domain $\Omega_+^{m,h}$, we need to perform two steps. First we need to find all the elements that are cut by the discrete interface Γ^m, and then we can select all the elements of $\mathcal{T}_+^m$, recall (4.1). For the first step we need to be able to detect whether a $(d-1)$-dimensional simplex of the interface grid Γ^m and a d-dimensional simplex from the bulk triangulation $\mathcal{T}^m$ intersect or not. Such intersections can be detected with the help of Algorithm 1 for the case $d = 2$ and Algorithm 2 for the case $d = 3$.

The general procedure, valid for both $d = 2, 3$, for identifying all the bulk elements of $\mathcal{T}^m$ intersected by interface elements of Γ^m is described in Algorithm 3. It is worth noting that we make use of the hierarchical structure of the bulk grid, which allows an efficient traversal of the grid. We refer to A.2 for all the relevant definitions, including the notions of *grid level*, *leaf element*, and *father relation*, used in Algorithm 4. In practice, thanks to the features of a hierarchical grid, we do not need to analyse those bulk elements obtained from refinement of elements of the initial macro-triangulation $\mathcal{T}^{-1}$ which are not cut by any $\sigma^m \in \Gamma^m$.

Once all the cut elements have been correctly identified, we are able to select the elements of the collection $\mathcal{T}_+^m$, with the help of Algorithm 5. Here we are inspired by the strategy applied in [7], where an *unfitted* finite element approximation for dendritic crystal growth with thermal convection was studied.

Algorithm 5 produces a partition of the original bulk triangulation $\mathcal{T}^m$, where the *outside* elements are the elements in $\mathcal{T}_+^m$. A visual representation of the procedure is shown in Figure 4.6, where we see that the *currentFront* composed of *outside* elements moves from the outer boundary of the bulk towards the void, until all the elements in $\mathcal{T}_+^m$ are correctly labelled. We note that all the elements intersected by Γ^m are contained in $\mathcal{T}_+^m$. This is

Algorithm 3: Identifying all the intersections between bulk and interface elements.

```
Data: Bulk triangulation $\mathcal{T}^m$ and interface mesh $\Gamma^m$
Result: Identification of all the elements of $\mathcal{T}^m$ cut by the interface $\Gamma^m$
begin
    for $\sigma^m \in \Gamma^m$ do
        for $o_0 \in \mathcal{T}^{-1}$ do
            if $o_0 \cap \sigma^m \neq \emptyset$ then
                Algorithm 4 $(o_0, \sigma^m)$;
            end
        end
    end
end
```

Algorithm 4: Hierarchic search applied to the identification of intersections between bulk and interface elements.

```
Data: Bulk element $o$ and interface element $\sigma^m$
Result: Identification of all leaf elements in the hierarchical grid
        starting from $o$ which are cut by $\sigma^m$
begin
    if $o$ is a leaf element then
        Mark $o$ as cut;
        return;
    end
    for $c_i$ child of $o$ do
        if $c_i \cap \sigma^m \neq \emptyset$ then
            Algorithm 4 $(c_i, \sigma^m)$;
        end
    end
end
```

Algorithm 5: Finding the subset $\mathcal{T}^m_+$ of $\mathcal{T}^m$.

```
Data: Bulk triangulation $\mathcal{T}^m$ and interface mesh $\Gamma^m$. The latter does not
      intersect the boundary $\partial\Omega$.
Result: Selection of all the elements belonging to $\mathcal{T}^m_+$
begin
    Mark all the elements of $\mathcal{T}^m$ as clear;
    Mark all the elements of $\mathcal{T}^m$ cut by the interface as outside;
    Mark all the boundary elements of $\mathcal{T}^m$ as outside and put them in
    currentFront;
    while currentFront is not empty do
        for $o \in$ currentFront do
            for $o_n$ neighbour of $o$ do
                if $o_n$ is clear then
                    put $o_n$ in newFront;
                    mark $o_n$ as outside;
                end
            end
        end
        currentFront = newFront;
        Clear newFront;
    end
end
```

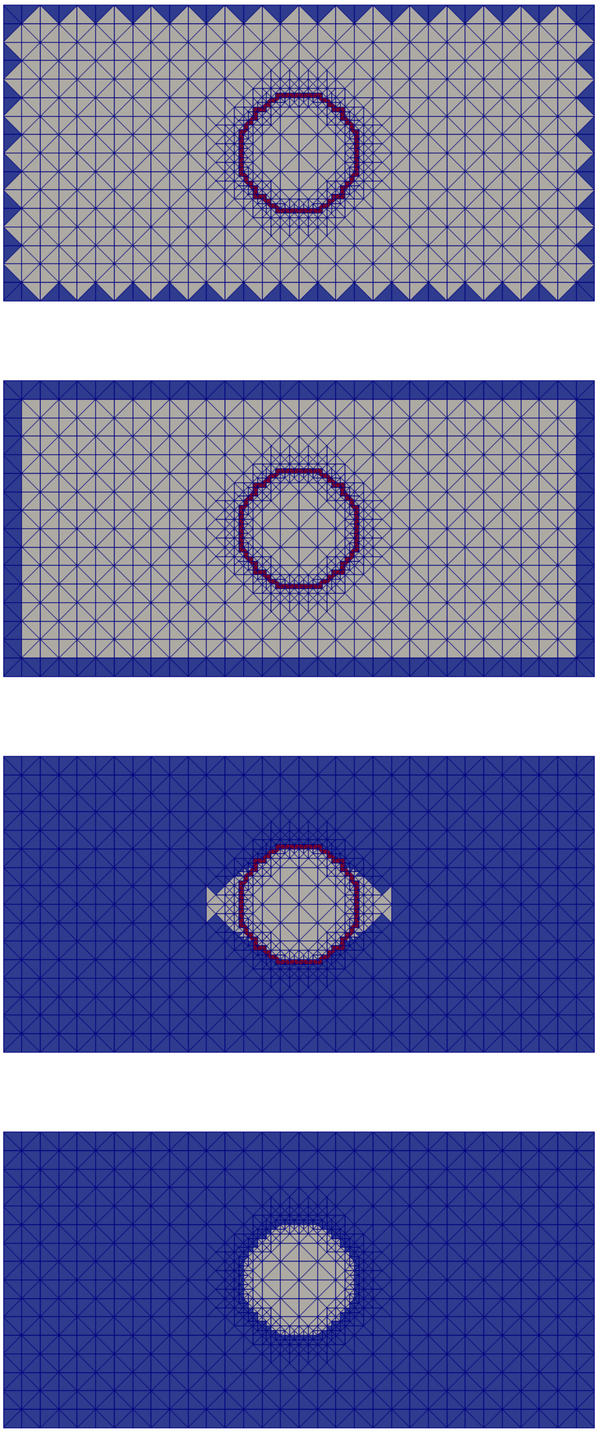

Figure 4.6: Visual representation of the labelling routine in Algorithm 5, for $d = 2$. *Outside* elements are coloured in blue, *clear* elements in grey, and *cut* elements in red. At the final step (bottom picture), all the elements in $\mathcal{T}^m_+$ are correctly labelled, including those intersected by Γ^m.

motivated by the fact that we need to evaluate $\nabla_s \Phi^{m+1}$ and $\nabla_s E^{m+1}$ on Γ^m in (4.2d), and this is only guaranteed to be well-defined if Φ^{m+1} and E^{m+1} are defined on all such intersected elements.

4.5 NUMERICAL SIMULATIONS

In this section we present several numerical simulations to test the accuracy of the *unfitted* method. Unless stated otherwise, we use uniform time steps $\tau_m = \tau$, $m = 0, \ldots, M-1$, for all the experiments in this section. As we will compare our numerical results to the phase field computations in [20, 3], we also fix $\alpha_1 = \frac{1}{16}\pi^2$ throughout this section. Moreover, when stress-migration is considered, we assume that (3.6) holds. We first present the simulations for $d = 2$ and then for $d = 3$.

Our first experiment with $d = 2$ refers to the test case (3.15), where the true solution of a circular void, moving at constant speed through an infinite conductor with $d = 2$, is known. It is worthwhile to note that in this particular case we prescribe Dirichlet boundary conditions on the entire outer boundary of Ω, and not only on $\partial_2\Omega$. The case has been presented in [104, Table 1], and we report it here with additional data for the benefit of the reader. We choose the following parameters: $L_1 = 1.5$, $L_2 = 0.5$, $\alpha_2 = 3\pi^2$, $\alpha_3 = 0$. The initial geometry is a circle with radius $R = 0.25$ and centre $z = (-0.5, 0)$, while $T = 2 \times 10^{-3}$. Following [12], we define the error

$$\mathcal{E}_\Gamma = \|\vec{X} - \vec{x}\|_{L^\infty} := \max_{m=1\to M} \|\vec{X}^m - \vec{x}\,(\cdot, t_m)\|_{L^\infty}\,, \tag{4.21}$$

where

$$\|\vec{X}^m - \vec{x}\,(\cdot, t_m)\|_{L^\infty} := \max_{k=1\to K_\Gamma^m} \{ \min_{\vec{y}\in\Gamma(t)} |\vec{X}^m(\vec{q}_k^m) - \vec{x}\,(\vec{y}, t_m))|\}, \tag{4.22}$$

between $\vec{X}$ and the true solution on the interval $[0, T]$. The L^∞ norm in (4.22) can be easily interpreted as follows. For every vertex $\vec{q}_k^m$ of the interface mesh, we compute the Euclidean norm of the difference between the position of the vertex and its projection onto the unit sphere; the L^∞ norm is then given by the maximum over all the vertices $\{\vec{q}_k^m\}_{k=1}^{K_\Gamma^m}$. In addition, we define the error

$$\mathcal{E}_{\text{bulk}} := \|\Phi - \phi\|_{L^\infty(0,T;H^1(\Omega_+^{m,h}))} := \max_{m=1\to M} \|\Phi^m - \phi(\cdot, t_m)\|_{H^1(\Omega_+^{m-1,h})}$$

between Φ and the exact electric potential ϕ on the interval $[0,T]$. Since the exact solution is known, it is worth calculating an experimental order of convergence. To this end, we employ (4.20a) with

$$N_f^{(i)} = 16\, N_c^{(i)} = 2^{7+i}\,, \quad i = 0 \to 3\,.$$

In addition, we set $J_\Gamma^{m(i)} = K_\Gamma^{m(i)} = N_f^{(i)}$. Therefore, the initial interface grid has a resolution proportional to h_f. The experimental orders of convergence are computed as $\frac{1}{\log(2)} \log\left(\frac{\mathcal{E}^{(i-1)}}{\mathcal{E}^{(i)}}\right)$, for both $\mathcal{E}_\Gamma$ and $\mathcal{E}_{\text{bulk}}$. The corresponding errors are listed in Table 4.1, where it appears that we observe a convergence of at least $\mathcal{O}(h_f)$ in the measured error for the interface, and at least $\mathcal{O}(h_f^{1/2})$ for the H^1-error for the approximation of the electric potential in the bulk. We also report on the CPU times and the average number of bulk degrees freedom. These quantities will aid a direct comparison with the four *fitted* approach runs presented in § 5.4, Table 5.1. Recall from [8] that an optimal $\mathcal{O}(h)$ convergence rate in the H^1-norm could be obtained for a more sophisticated *unfitted* finite element approximation for a related Neumann boundary value problem. However, we are satisfied that our more practical method performs well for the electro-stress migration problem of our interest.

i	$\tau \cdot 10^6$	$h_f \cdot 10^3$	$h_c \cdot 10^2$	$\mathcal{E}_{\text{bulk}} \cdot 10^2$	EOC_{bulk}	$\mathcal{E}_\Gamma \cdot 10^3$	EOC_Γ	Time [s]
0	8	7.81	12.5	9.037	–	16.459	–	18
1	2	3.91	6.25	5.919	0.61	7.556	1.12	174
2	0.5	1.95	3.13	3.650	0.69	3.474	1.12	2098
3	0.125	0.977	1.56	2.143	0.77	1.524	1.19	37861

Table 4.1: Results of the convergence test for the *unfitted* case, reproduced from [104, Table 1]. Note that the average number of bulk degrees of freedom for the four runs are 990, 2296, 5987 and 17677, respectively.

Our next experiment corresponds to [104, Figs. 4 and 5], see also [20, Fig. 2] and [27, Fig. 4]. Only electro-migration is involved. We choose the radius of the initially circular void to be relatively large compared to the width of the conductor, $2L_2$. We use the following parameters:

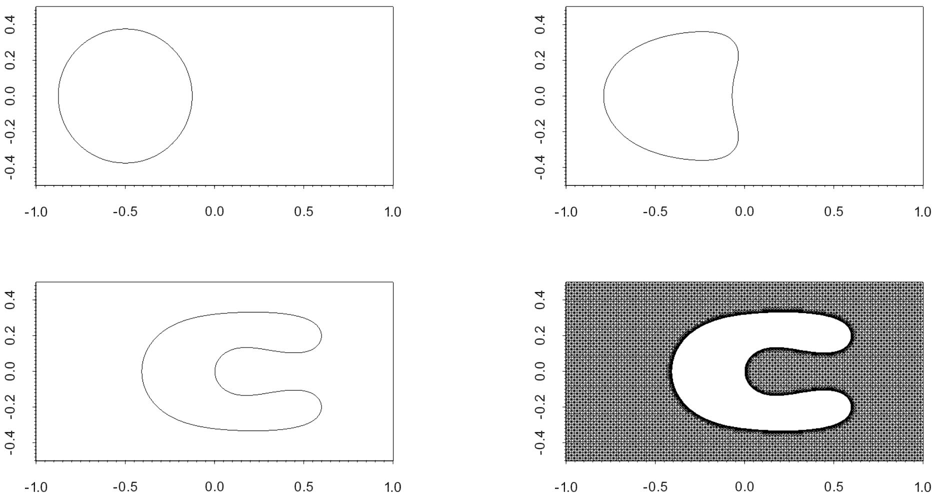

Figure 4.7: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$) Plots of the interface curve at times $t = 0, 8 \times 10^{-5}, T = 3.6 \times 10^{-4}$, and bulk mesh at time $t = T$.

$L_1 = 1, L_2 = 0.5$, $\alpha_2 = \frac{256}{9}\pi^2, \alpha_3 = 0$, $\tau = 4.5 \times 10^{-7}$, $T = 3.6 \times 10^{-4}$ and $J^m_\Gamma = K^m_\Gamma = 1024$. As initial data we choose a circle with radius 0.375 and centre $(-0.5, 0)$; the bulk refinement parameters are $N_f = 1024$ and $N_c = 64$, respectively. We note that the experiment corresponds to [104, Fig. 4], where Dirichlet boundary conditions were applied, as in (3.3b). In Figure 4.7 we plot the results of the simulation at times $t = 0, 8 \times 10^{-5}$, and $T = 3.6 \times 10^{-4}$. The total CPU time was 1557s, while the average number of bulk degrees of freedom was 17893. Recall that in (2.33) we defined $r_h := h_{\Gamma^m}/l_{\Gamma^m}$ where h_{Γ^m} and l_{Γ^m} represent the maximum and minimum segment of Γ^m, respectively. In Figure 4.8 we plot the temporal evolution of the interface mesh quality indicator r_h. We note that the vertices of the interface mesh remain well distributed.

As already noted in [104, § 5], Dirichlet boundary conditions make the drifting of the void slower, compared to the plots in [20, Fig. 2]. Therefore, in order to perform a direct comparison, we prescribe the new boundary conditions:

$$\frac{\partial \phi}{\partial \vec{\nu}_{\partial\Omega}} = 0 \text{ on } \partial_1\Omega\,, \quad 2\,\frac{\partial \phi}{\partial \vec{\nu}_{\partial\Omega}} + \phi = g_R := x_1 \pm 2 \quad \text{on } \partial_2^{\pm}\Omega\,, \qquad (4.23)$$

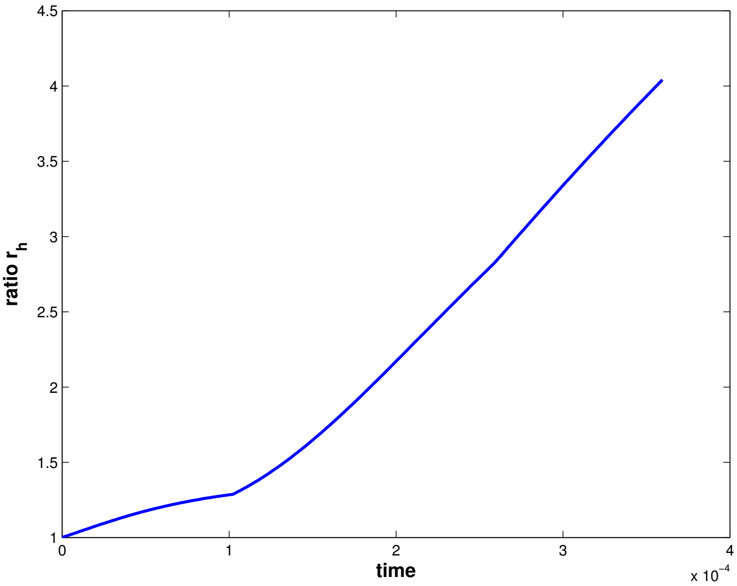

Figure 4.8: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$) Plot of the quality mesh indicator r_h for the interface mesh plotted in Figure 4.7.

which means that (4.2a) needs to be replaced by

$$\int_{\Omega_+^{m,h}} \nabla\Phi^{m+1} \cdot \nabla\psi \, d\mathcal{L}^d + \frac{1}{2}\int_{\partial_2\Omega} \Phi^{m+1}\,\psi\, d\mathcal{H}^{d-1} = \frac{1}{2}\int_{\partial_2\Omega} g_R\,\psi\, d\mathcal{H}^{d-1}$$

for all test functions $\psi \in S^{m,h}$. In Figure 4.9 we plot the results of the simulation at times $t = 0, 8 \times 10^{-5}$, and $T = 3.6 \times 10^{-4}$, where the new boundary conditions (4.23) have been applied. The total CPU time was 1927s, while the average number of bulk degrees of freedom was 19674. We now note the good agreement with the results in [20, Fig. 2] and [27, Fig. 4]. In Figure 4.10 we plot the temporal evolution of the interface mesh quality indicator r_h. We note that the vertices on the interface mesh remain well distributed. The presence of a stronger electric field, induced by Robin boundary conditions, contributes to increasing the value of r_h, compared to the case with Dirichlet boundary conditions in Figure 4.8.

The next experiments involve again electro-migration only, but in this case the interface undergoes topological changes, due to the magnitude of the electric field. We first prescribe Dirichlet boundary conditions (3.3b). We choose the following parameters: $L_1 = 1.5$, $L_2 = 0.5$, $\alpha_2 = 16\pi^2$, $\alpha_3 = 0$, $T = 10^{-3}$. The initial interface is composed of an ellipse and a

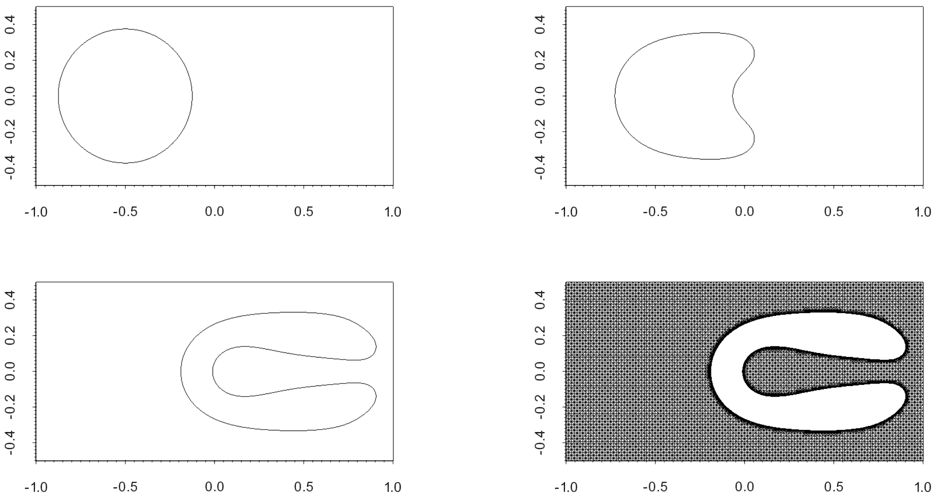

Figure 4.9: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface curve at times $t = 0$, 8×10^{-5}, $T = 3.6 \times 10^{-4}$, and bulk mesh at time $t = T$.

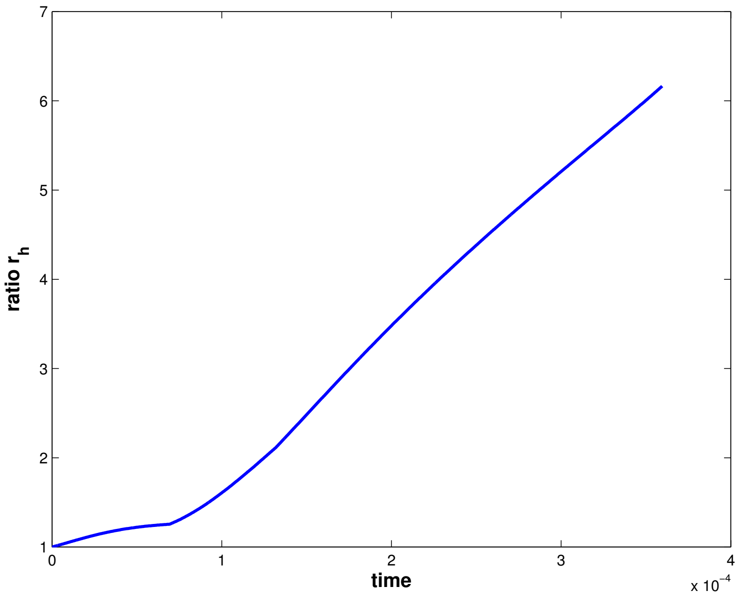

Figure 4.10: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$, with Robin boundary conditions (4.23)) Plot of the quality mesh indicator r_h for the interface mesh plotted in Figure 4.9.

circle, the former with horizontal semiaxis 0.2, vertical semiaxis 0.3 and centre $(-1.1, 0)$, the latter with radius 0.2 and centre $(-0.5, 0)$. The refinement parameters are $N_f = 512$ and $N_c = 32$. The experiment corresponds to [20, Fig. 8], where the authors considered a phase field approximation with Robin boundary conditions. The voids undergo a topological change, which we handle with the help of the library El-Topo, as explained in Section 2.3.1. The simulation is split into different parts with variable time steps, since the newly-created voids exhibit corners with a very high curvature immediately after the topological change has taken place. The first part of the simulation is run with time step $\tau = 10^{-7}$ from $t = 0$ to $T_1 = 6.238 \times 10^{-4}$, at which time the ellipse pinches off into two new voids, while the original circle has elongated and drifted in the conductor. In order to capture the evolution correctly and avoid area loss, we now reduce the time step to $\tau = 10^{-10}$ until $T_2 = 6.338 \times 10^{-4}$. The last part is from $t = T_2$ to $T = 10^{-3}$, again with $\tau = 10^{-7}$. At this time the simulation cannot continue any longer, because the original circular void has reached the right boundary of the conductor. The final interface configuration is composed of three voids. The complete evolution of the system can be seen in Figure 4.11, where we note that El-Topo preserves the symmetry of the geometry.

It is worth mentioning that also in this case the choice of Dirichlet boundary conditions makes the drifting of the voids slower, compared to the results shown in [20, Fig. 8]. In order to perform a direct comparison, we now prescribe Robin boundary conditions (4.23) and take $T = 7.91 \times 10^{-4}$. Similarly to the previous experiment, the simulation is split into different parts with variable time steps. The first part of the simulation is run with time step $\tau = 10^{-7}$ from $t = 0$ to $T_1 = 4.702 \times 10^{-4}$, at which time the ellipse pinches off into two new voids, while the original circle has elongated and drifted in the conductor. We now reduce the time step to $\tau = 10^{-10}$ until $T_2 = 4.802 \times 10^{-4}$. The third part is from $t = T_2$ to $T_3 = 6.457 \times 10^{-4}$, again with $\tau = 10^{-7}$. At this point, the two smaller voids originated from the initial ellipse pinch off again. We now use $\tau = 10^{-10}$ until $T_4 = 6.487 \times 10^{-4}$, and finally set $\tau = 10^{-7}$ until $T = 7.91 \times 10^{-4}$. The final interface configuration is therefore composed of five voids. The complete evolution of the system can be seen in Figure 4.12. We note that also in this case El-Topo

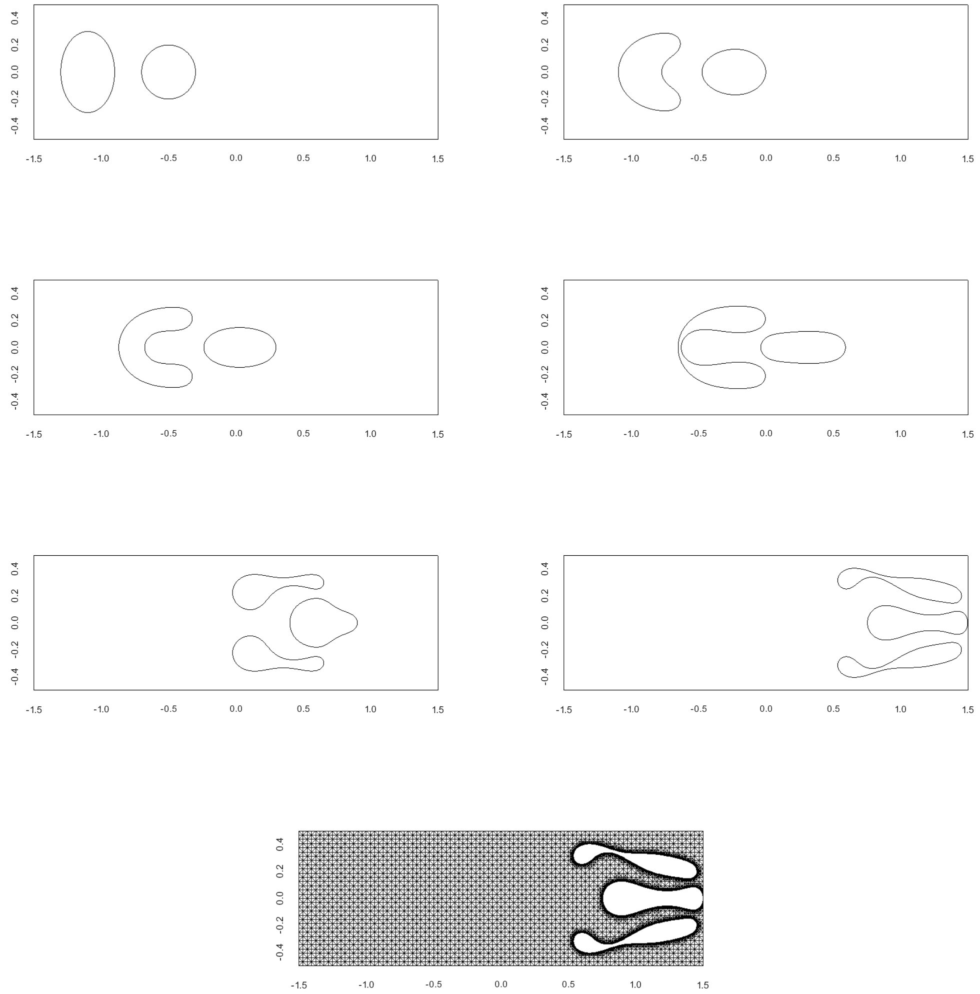

Figure 4.11: ($\alpha_2 = 16\pi^2, \alpha_3 = 0$) Plots of the interface curve at times $t = 0, 2 \times 10^{-4}, \ldots, T = 10^{-3}$ and bulk mesh at $t = T$. The interface at $t = T$ is composed of three voids.

preserves the symmetry and returns results which are in good agreement with the phase field computations presented in [20, Fig. 8].

Our next experiment corresponds to [105, Fig. 3] and involves both electro-migration and stress-migration. We choose the following parameters: $L_1 = 2.5, L_2 = 0.5, \alpha_2 = 10\,\pi^2$, $\alpha_3 = \frac{1}{8}\,\pi$, $\mu = \lambda = 1$, $\underline{\underline{S}} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 0 \end{smallmatrix}\right)$, $\tau = 5 \times 10^{-7}$, $J_\Gamma^m = K_\Gamma^m = N_f = 1024$, $N_c = 16$. The initial geometry is a circle with radius $R = 0.25$ and centre $z = (-1.5,\ 0)$, while $T = 3.75 \times 10^{-3}$. The total CPU time was 11141s, while the average number of bulk degrees of freedom (counting both Φ^{m+1} and $\vec{U}^{m+1}$) was 23325. In Figure 4.13 we plot the results of the simulation at times $t = 0$, 1.25×10^{-3}, 2.5×10^{-3} and $T = 3.75 \times 10^{-3}$. It can be seen that the void moves through the conductor due to the presence of the electric field, while the elastic stress contributes to flattening the vertical front of the void. The plots show a good agreement with the results in [105, Fig. 3], see also Figure 5.12 below.

In order to evaluate the contribution to the drifting of the void produced by the two external loadings, we now repeat the experiment by considering electro-migration and stress-migration separately. In Figure 4.14 we plot the results of the simulations with $\alpha_2 = 0$, $\alpha_3 = \frac{1}{8}\,\pi$. We note that the void does not drift, but slightly elongates in the vertical direction. In Figure 4.15, instead, we plot the results of the simulations with $\alpha_2 = 10\,\pi^2$, $\alpha_3 = 0$. We note that the void drifts along the conductor, keeping its original circular profile.

In the next experiment only stress-migration is considered. We choose the following parameters: $L_1 = L_2 = 0.5$, $\alpha_2 = 0$, $\alpha_3 = \frac{1}{8}\,\pi$, $\mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$, $\tau = 10^{-7}$, $J_\Gamma^m = K_\Gamma^m = 2048$, $N_f = 1024$, $N_c = 64$. The initial geometry is composed of two circles with radius $R = 0.18$ and centres $z_\pm = (\pm 0.25,\ 0)$, while $T = 5 \times 10^{-5}$. The total CPU time was 1254s, while the average number of bulk degrees of freedom was 23893. In Figure 4.16 we plot the results of the simulation at times $t = 0$ and $t = T$. For this experiment we observe that for larger times T a singularity develops. In particular, the two voids exhibit sharp corners and the elastic energy becomes unbounded. This can be seen in Figure 4.17. where the developing singularity leads to a breakdown in the numerical approximation. We now investigate whether this breakdown is related to numerical errors. To this end, we compute

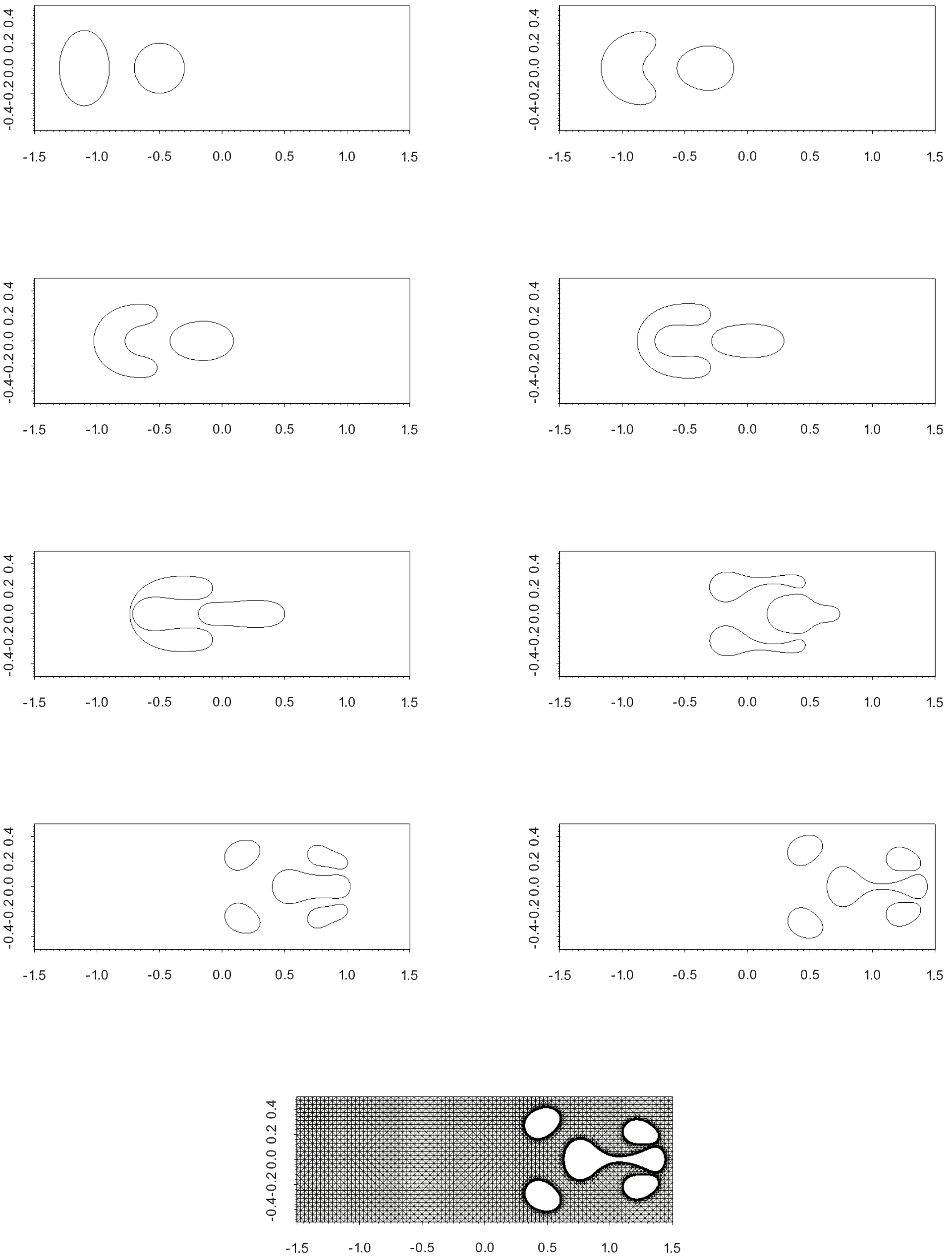

Figure 4.12: ($\alpha_2 = 16\pi^2, \alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface curve at times $t = 0, 1.13 \times 10^{-4}, \ldots, T = 7.91 \times 10^{-4}$ and bulk mesh at $t = T$. The interface at $t = T$ is composed of five voids.

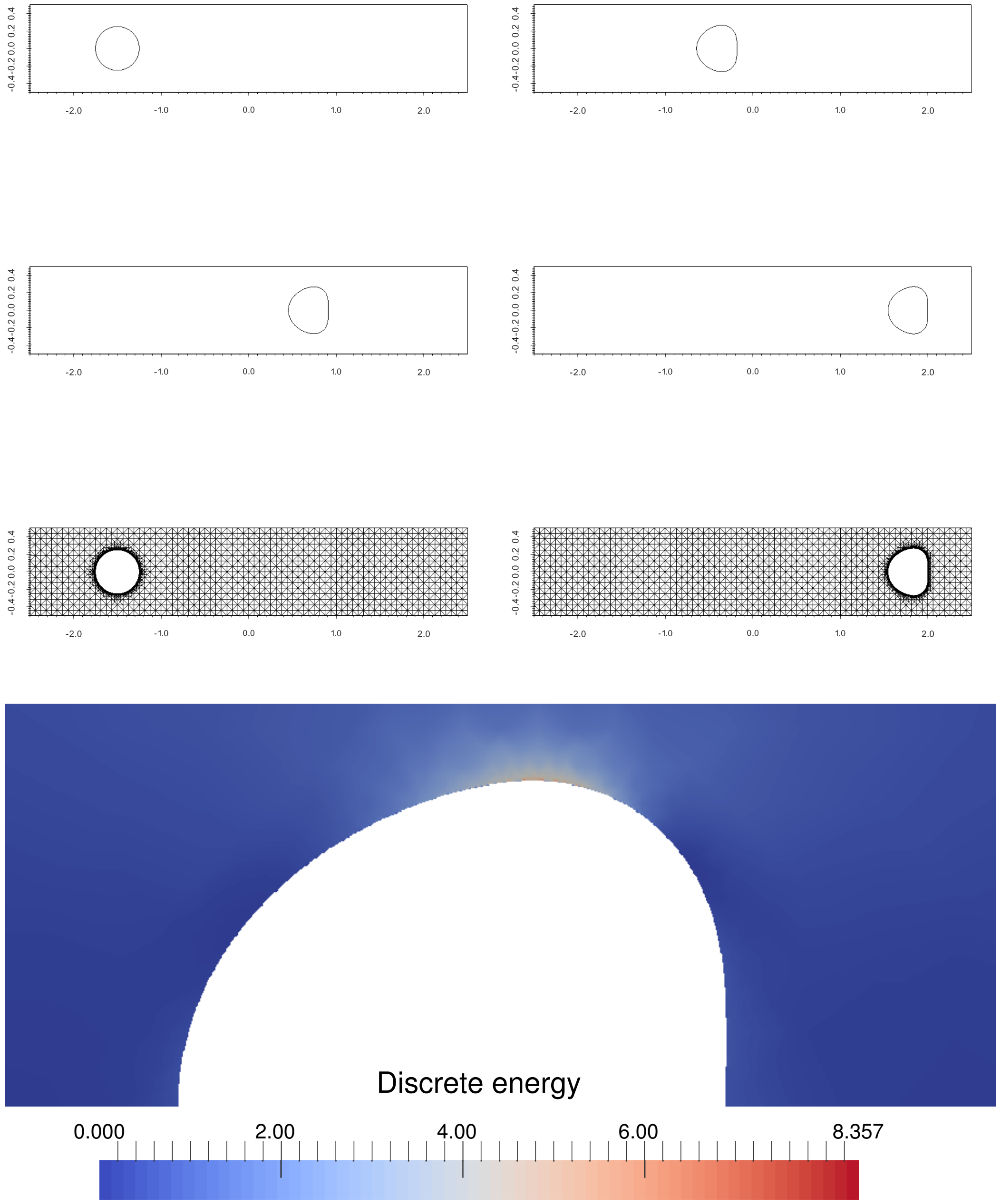

Figure 4.13: ($\alpha_2 = 10\pi^2$, $\mu = \lambda = 1$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$) Plots of the interface curve at times $t = 0, 1.25 \times 10^{-3}, 2.5 \times 10^{-3}$ and $T = 3.75 \times 10^{-3}$; bulk mesh at $t = 0, T$ and elastic energy density at $t = T$.

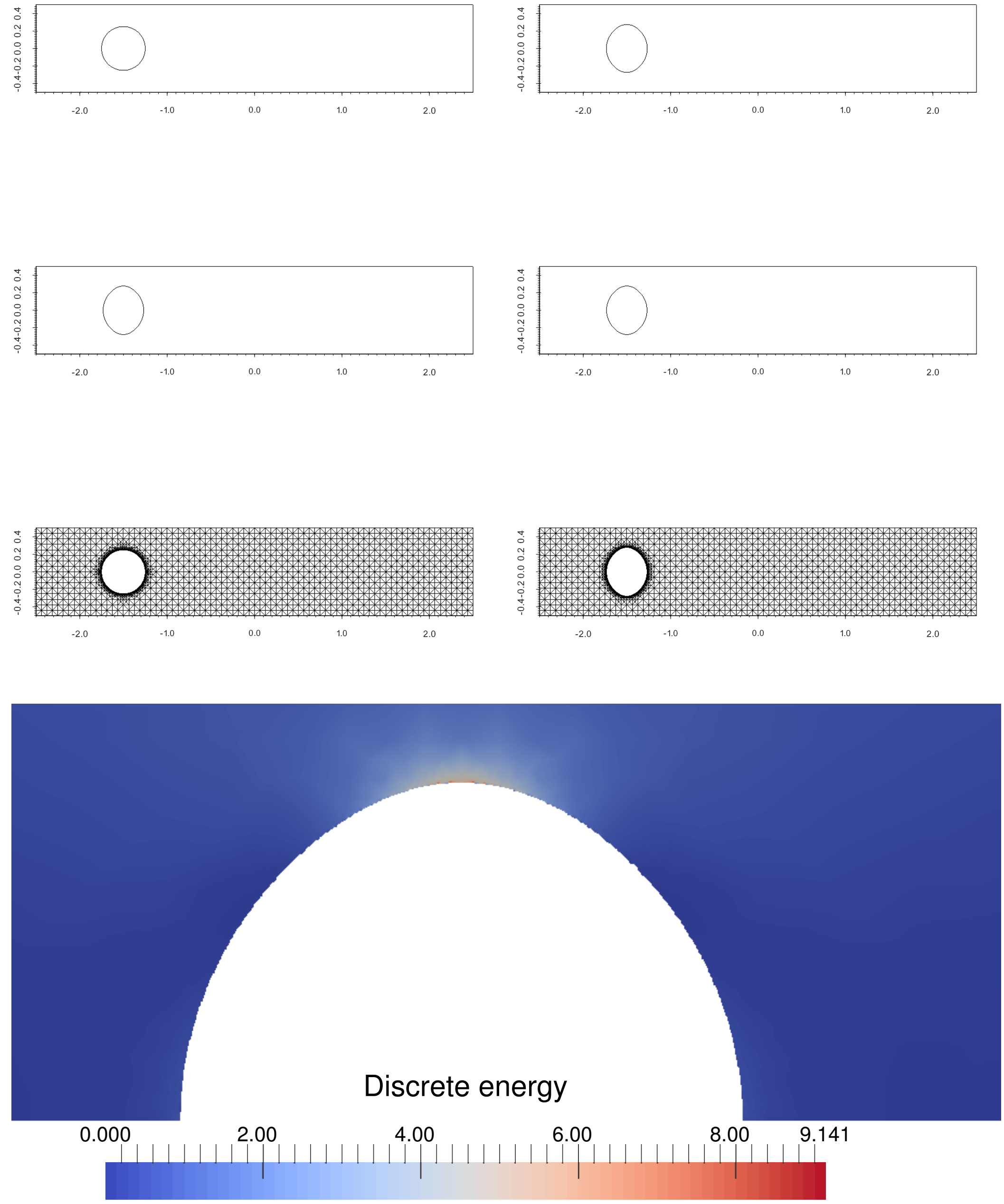

Figure 4.14: ($\alpha_2 = 0$, $\mu = \lambda = 1$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$) Plots of the interface curve at times $t = 0, 1.25 \times 10^{-3}$, 2.5×10^{-3} and $T = 3.75 \times 10^{-3}$; bulk mesh at $t = 0$, T and elastic energy density at $t = T$.

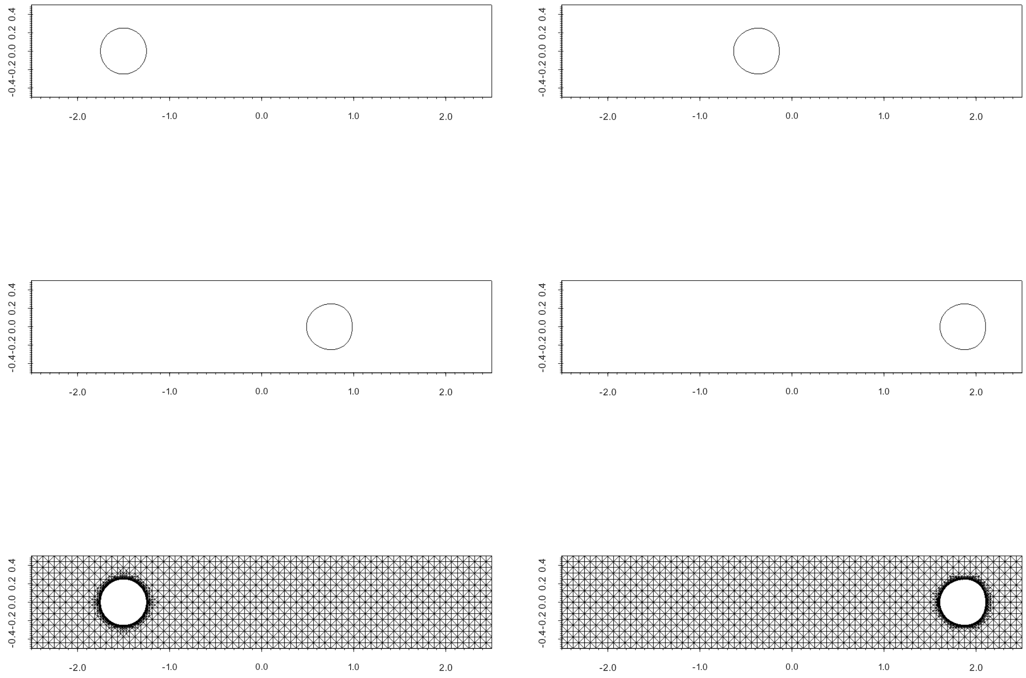

Figure 4.15: ($\alpha_2 = 10\,\pi^2$, $\alpha_3 = 0$) Plots of the interface curve at times $t = 0, 1.25 \times 10^{-3}$, 2.5×10^{-3} and $T = 3.75 \times 10^{-3}$, and bulk mesh at $t = 0$, T.

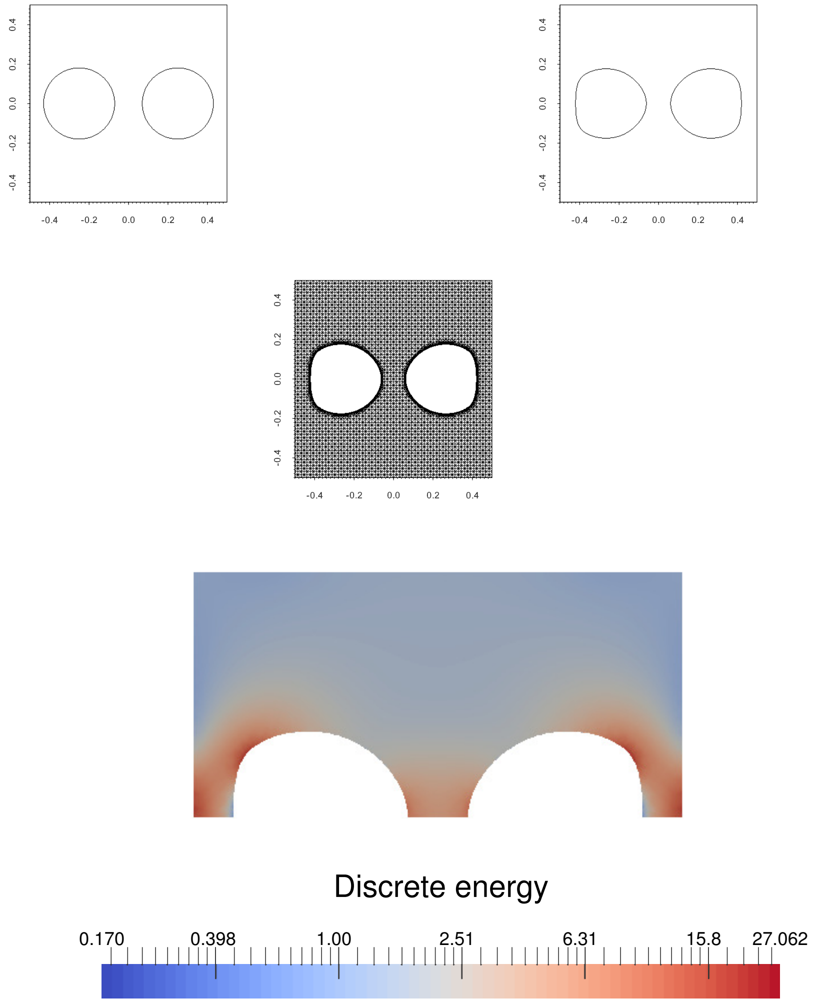

Figure 4.16: ($\alpha_2 = 0, \mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right)$) Plots of the interface curve at times $t = 0$ and $T = 5 \times 10^{-5}$; bulk mesh and elastic energy density at $t = T$. The energy density is colour coded with a logarithmic scale for the sake of visualisation.

the total discrete energy $\alpha_1\,\mathcal{H}^{d-1}(\Gamma^m) + \alpha_3\,(\int_{\Omega_+^{m,h}} E(\vec{U}^{m+1})\;d\mathcal{L}^d - \int_{\partial\Omega} \vec{g}\cdot\vec{U}^{m+1}\;d\mathcal{H}^{d-1})$ and plot it in Figure 4.17 (bottom). We note that the discrete energy is overall decreasing in time, even though not monotonically, recall (3.12). This is most likely due to the fact that $\Omega_+^{m,h}$ is given by a union of bulk mesh elements, and therefore its boundary $\partial\Omega_+^{m,h}$ is very rough, with right-angled corners. In addition, the evaluation of the elastic energy density in (4.2d) is vertex-based. However, the observed singularity and the behaviour of the elastic energy density appear to be consistent with the sharp interface model (3.1), (3.7a)-(3.7c) itself.

In the last experiment for $d = 2$ only stress-migration is considered. The test corresponds to [105, Fig. 5]. We choose the following parameters: $L_1 = L_2 = 0.5$, $\alpha_2 = 0$, $\alpha_3 = \frac{1}{8}\pi$, $\mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$, $\tau = 10^{-6}$, $J_\Gamma^m = K_\Gamma^m = 2048$, $N_f = 1024$, $N_c = 16$. The initial geometry is composed of two circles with radius $R = 0.15$ and centres $z_\pm = (\pm 0.22, 0)$, while $T = 1.5\times 10^{-3}$. The total CPU time was 2117s, while the average number of bulk degrees of freedom was 15909. In Figure 4.18 we plot the results of the simulation at times $t = 0$, 5×10^{-4}, 10^{-3} and $T = 1.5\times 10^{-3}$. Similarly to the previous experiment, we observe that also in this case a singularity develops for larger times T. In particular, the two voids exhibit sharp corners and the elastic energy becomes unbounded. This can be seen in Figure 4.19, where the developing singularity leads to a breakdown in the numerical approximation. We compute the total discrete energy, which appears to be overall decreasing in time, even though not monotonically. Hence also for this experiment the observed singularity is consistent with the sharp interface model (3.1), (3.7a)-(3.7c) itself. We note the good agreement with the results in [105, Figs. 5 and 6], see also Figures 5.17 and 5.18 below.

Our first simulation with $d = 3$ corresponds to [3, Fig. 7], and involves electro-migration only. The initial profile models a cylindrical void with radius $R = 0.375$, penetrating the conductor. Every x_3 cross section corresponds to the two-dimensional geometry presented in [20, Fig. 2]. We choose the following parameters: $\alpha_2 = \frac{57}{2}\pi^2$, $\alpha_3 = 0$, $L_1 = 1$, $L_2 = L_3 = 0.5$, $\tau = 10^{-7}$, $T = 3.6\times 10^{-4}$. The mesh refinement parameters are $N_f = 128$ and $N_c = 16$. We consider Robin boundary conditions (4.23) for a direct

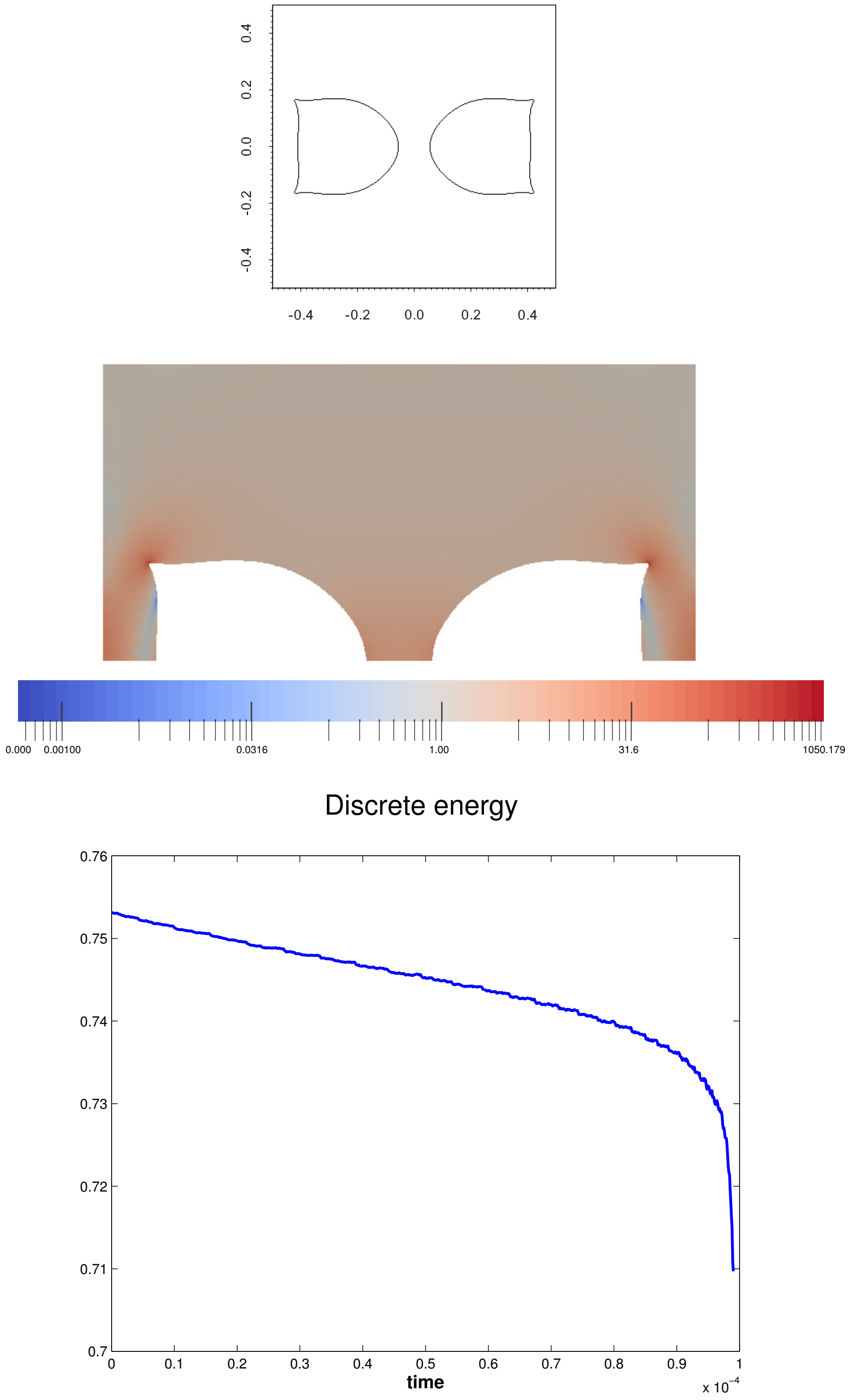

Figure 4.17: ($\alpha_2 = 0, \mu = 0.5, \lambda = 0, \underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right)$) Plots of the interface curve and the elastic energy density at $t = 9.9 \times 10^{-5}$. The energy density is colour coded with a logarithmic scale for the sake of visualisation. Below a plot of the total discrete energy $\alpha_1 \,\mathcal{H}^{d-1}(\Gamma^m) + \alpha_3 \,(\int_{\Omega_+^{m,h}} E(\vec{U}^{m+1}) \,d\mathcal{L}^d - \int_{\partial\Omega} \vec{g} \cdot \vec{U}^{m+1} \,d\mathcal{H}^{d-1})$ over time.

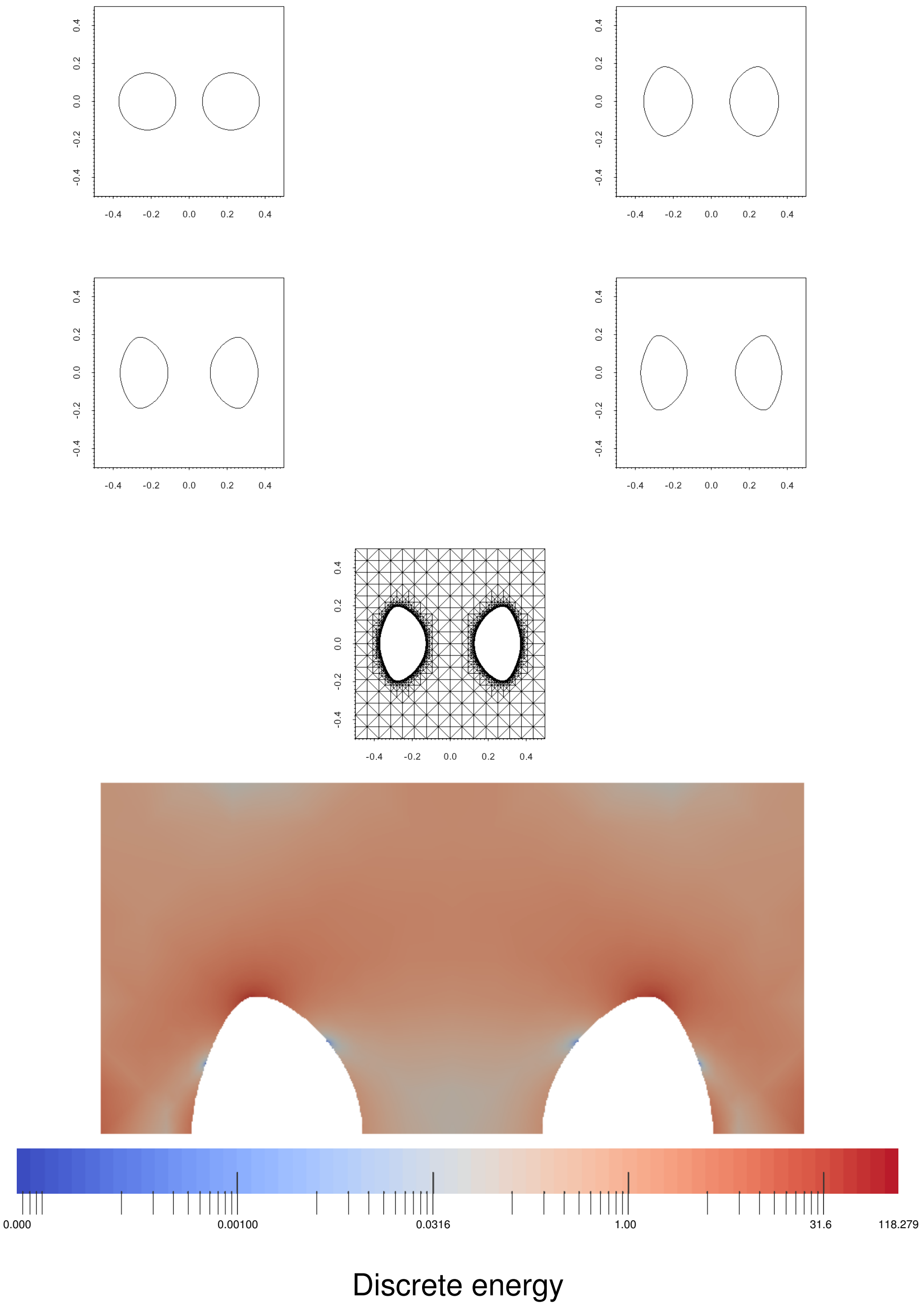

Figure 4.18: ($\alpha_2 = 0$, $\mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$) Plots of the interface curve at times $t = 0$, $5 \times 10^{-4}, 10^{-3}$, and $T = 1.5 \times 10^{-3}$, bulk mesh and the elastic energy density at time $t = T$. The energy density is colour coded with a logarithmic scale for the sake of visualisation.

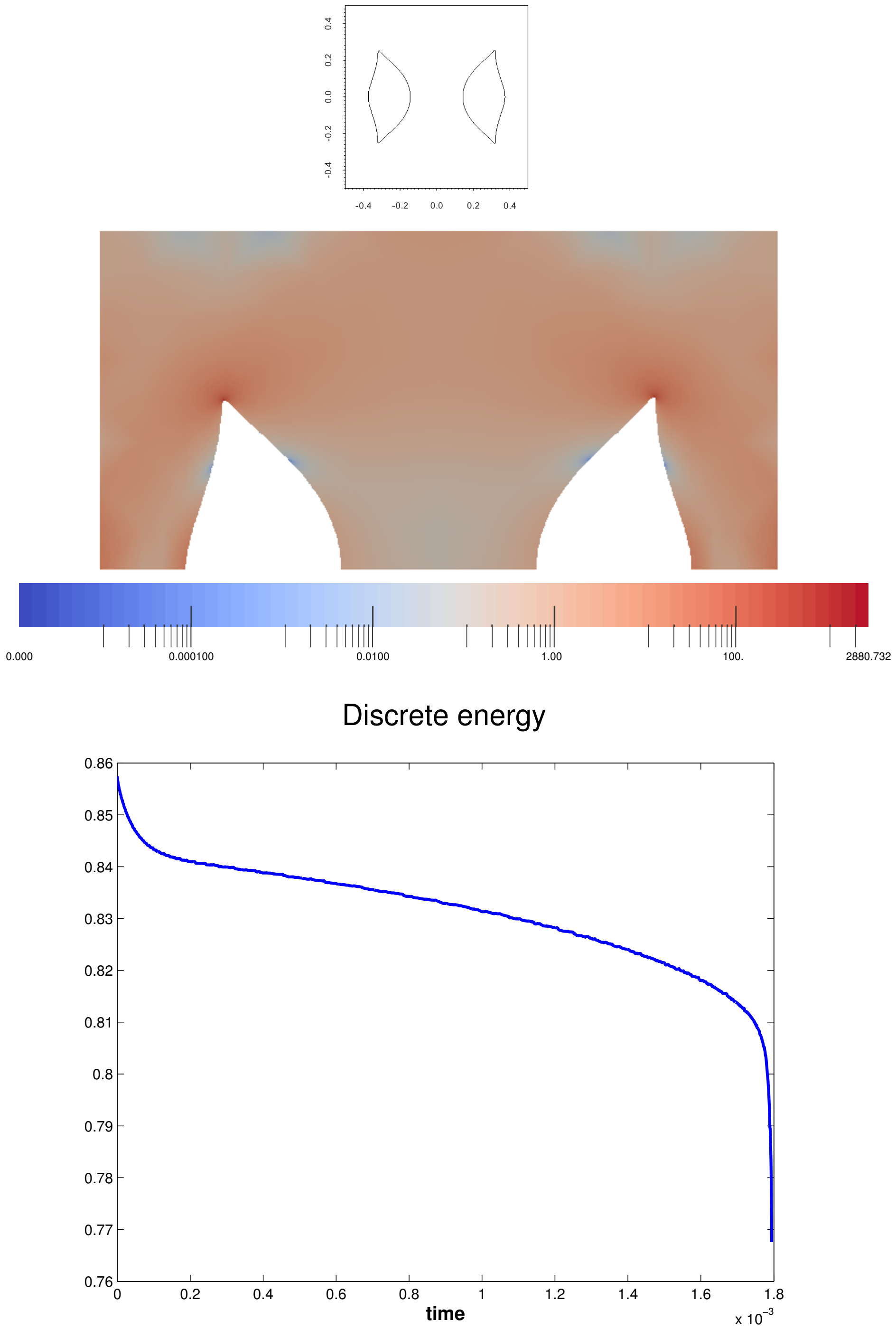

Figure 4.19: ($\alpha_2 = 0, \mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$) Plots of the interface curve and the elastic energy density at $t = 1.794 \times 10^{-3}$. The energy density is colour coded with a logarithmic scale for the sake of visualisation. Below a plot of the total discrete energy $\alpha_1 \mathcal{H}^{d-1}(\Gamma^m) + \alpha_3 \left(\int_{\Omega_+^{m,h}} E(\vec{U}^{m+1}) \, d\mathcal{L}^d - \int_{\partial\Omega} \vec{g} \cdot \vec{U}^{m+1} \, d\mathcal{H}^{d-1}\right)$ over time.

comparison. It is straightforward to notice that in this case the cylindrical interface has boundary intersections, where appropriate boundary conditions need to be prescribed. We naturally impose a *free-slip* condition for the displacement $\delta\vec{X}^{m+1}$, recall § 4.2.1. The matrix and the right-hand side of the system (4.13) need to be adjusted appropriately, recall Remark 4.6. Moreover, the set-up routines of Algorithm 5 are to be modified as well. Precisely, instead of considering *all* the boundary elements of $\mathcal{T}^m$ as *outside*, only the boundary elements belonging to $\partial_2\Omega$ have to be considered as *outside* and put into the *currentFront*. The total CPU time was 908266s, while the average number of bulk degrees of freedom was 109439. We note the good agreement between our results in Figure 4.20, the plots in [3, Fig. 7] and the corresponding two-dimensional results presented in [20, Fig. 2]. We now investigate the quality of the interface mesh. The temporal behaviour of the indicator r_a, recall (2.34), is plotted in Figure 4.21. We note that the interface mesh exhibits good properties, even in the presence of a strong electric field that induces a great change in the shape of the cylindrical void.

Our next simulation corresponds to [3, Fig. 8], and again only electromigration is considered. The experiment shows a fully three-dimensional situation, where the initial profile of the void is a sphere with radius $R = 0.375$. We keep all the simulation parameters as in the previous case. The total CPU time was 576840s, while the average number of bulk degrees of freedom was 99192. We note the good agreement between our results in Figure 4.22 and the plots in [3, Fig. 8]. We further investigate the drifting of the void, when a stronger electric loading is applied. To this end, we keep all the parameters unchanged bar α_2, which we now set to $75\,\pi^2$, and T, which we now set to 1.25×10^{-4}. The total CPU time was 197695s, while the average number of bulk degrees of freedom was 99435. The evolution of the void is plotted in Figure 4.23. Compared to [3, Fig. 9], the evolution shown in our plots seems slightly slower, since at time $t = T$ the interface is not yet undergoing the topological change observed in [3, Fig. 9].

Our last experiment is presented in Figure 4.24, and only stress-migration is involved. We choose the following parameters: $L_1 = L_2 = L_3 = 0.5$, $\alpha_2 = 0$, $\alpha_3 = \frac{1}{8}\pi$, $\mu = \lambda = \frac{4}{5\pi}$, $\underline{\underline{S}} = \left(\begin{smallmatrix} 0&0&0 \\ 0&1&0 \\ 0&0&0 \end{smallmatrix}\right)$, $\tau = 10^{-6}$, $N_f = 128, N_c = 16$.

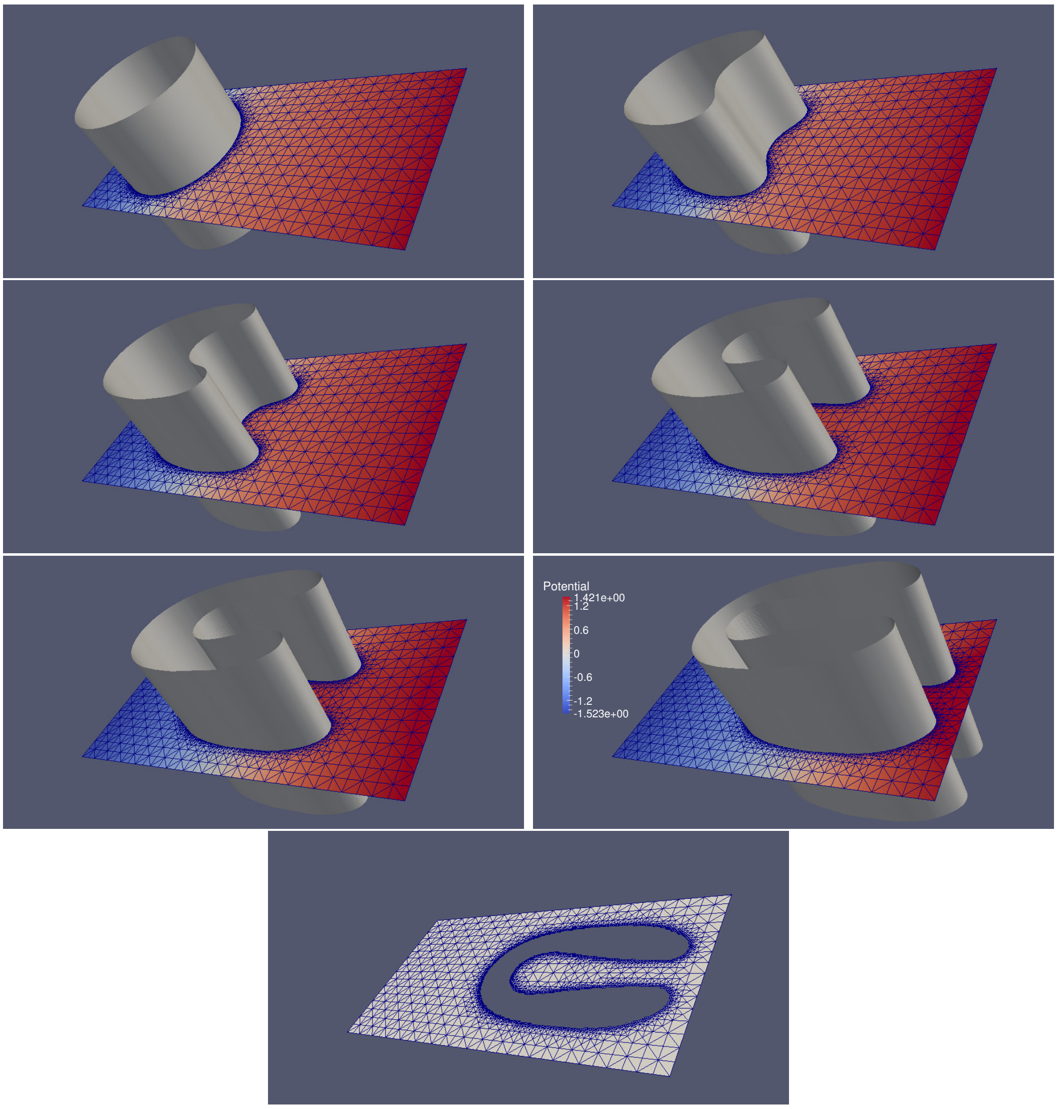

Figure 4.20: ($\alpha_2 = \frac{57}{2}\pi^2, \alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface mesh and cross section for $x_3 = 0$ of the bulk mesh at times $t = 0, 8 \times 10^{-5}$, 1.2×10^{-4}, 2×10^{-4}, 2.4×10^{-4} and $T = 3.6 \times 10^{-4}$. The electric potential is colour coded according to the legend shown at $T = 3.6 \times 10^{-4}$.

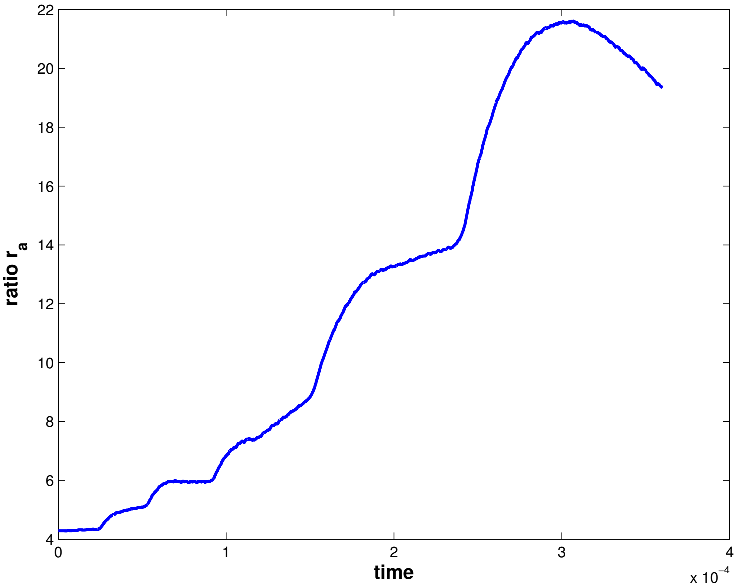

Figure 4.21: Plot of the ratio r_a (eq. (2.34)) for the interface mesh in Figure 4.20.

The initial geometry is a sphere with radius $R = 0.25$ and centre $z = (0, 0, 0)$, while $T = 0.02$. The average number of bulk degrees of freedom was 184586. The evolution of the system is plotted in Figure 4.24. The bulk mesh hardly changes after the 1000^{th} time step; it took 3.15×10^5s to reach the stationary state.

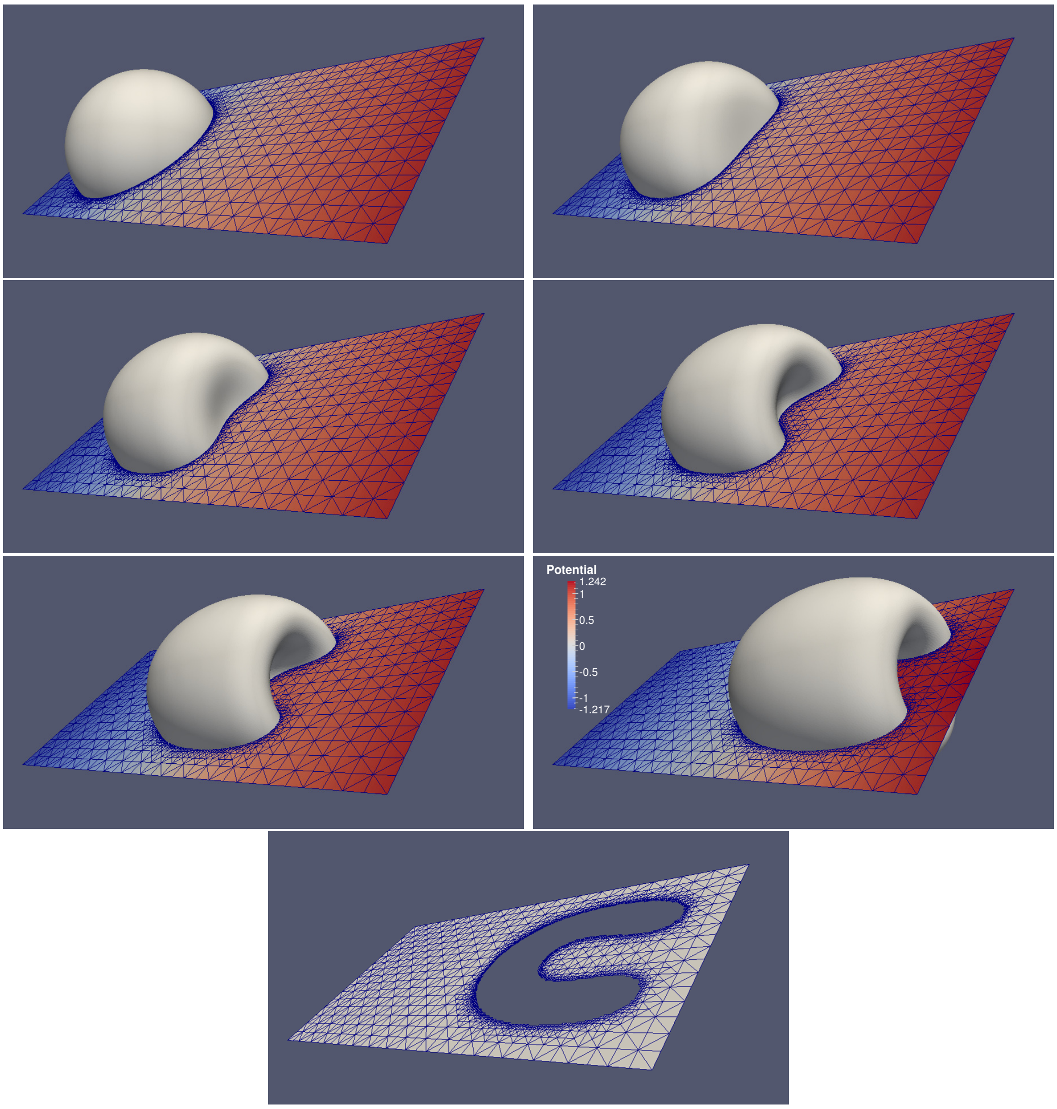

Figure 4.22: ($\alpha_2 = \frac{57}{2}\pi^2, \alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface mesh and cross section for $x_3 = 0$ of the bulk mesh at times $t = 0, 8 \times 10^{-5}$, 1.2×10^{-4}, 2×10^{-4}, 2.4×10^{-4} and $T = 3.6 \times 10^{-4}$. The electric potential is colour coded according to the legend shown at $T = 3.6 \times 10^{-4}$.

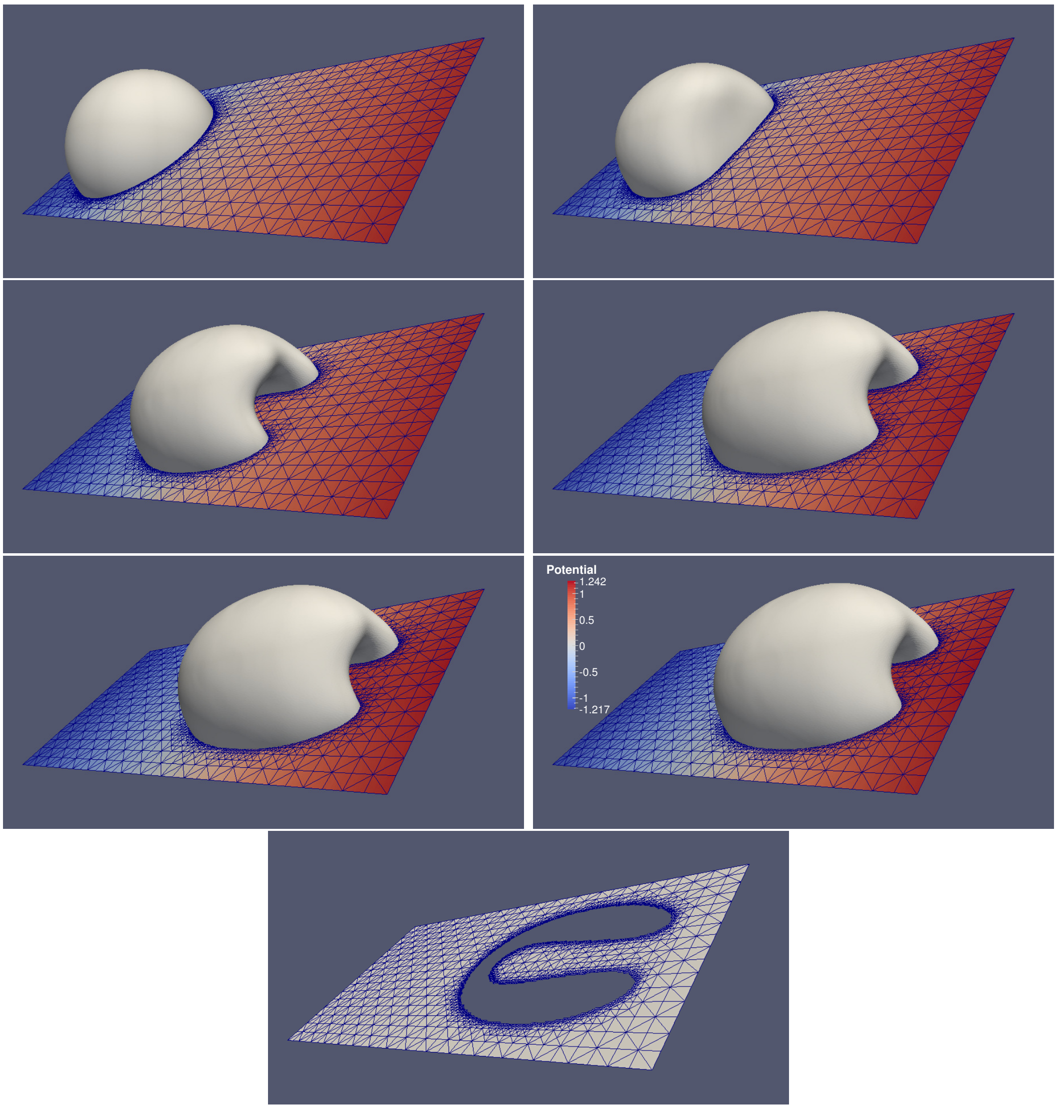

Figure 4.23: ($\alpha_2 = 75\,\pi^2, \alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface mesh and cross section for $x_3 = 0$ of the bulk mesh at times $t = 0, 2.5 \times 10^{-5}$, 7.5×10^{-5}, 1.15×10^{-4}, 1.2×10^{-4} and $T = 1.25 \times 10^{-4}$. The electric potential is colour coded according to the legend shown at $T = 1.25 \times 10^{-4}$.

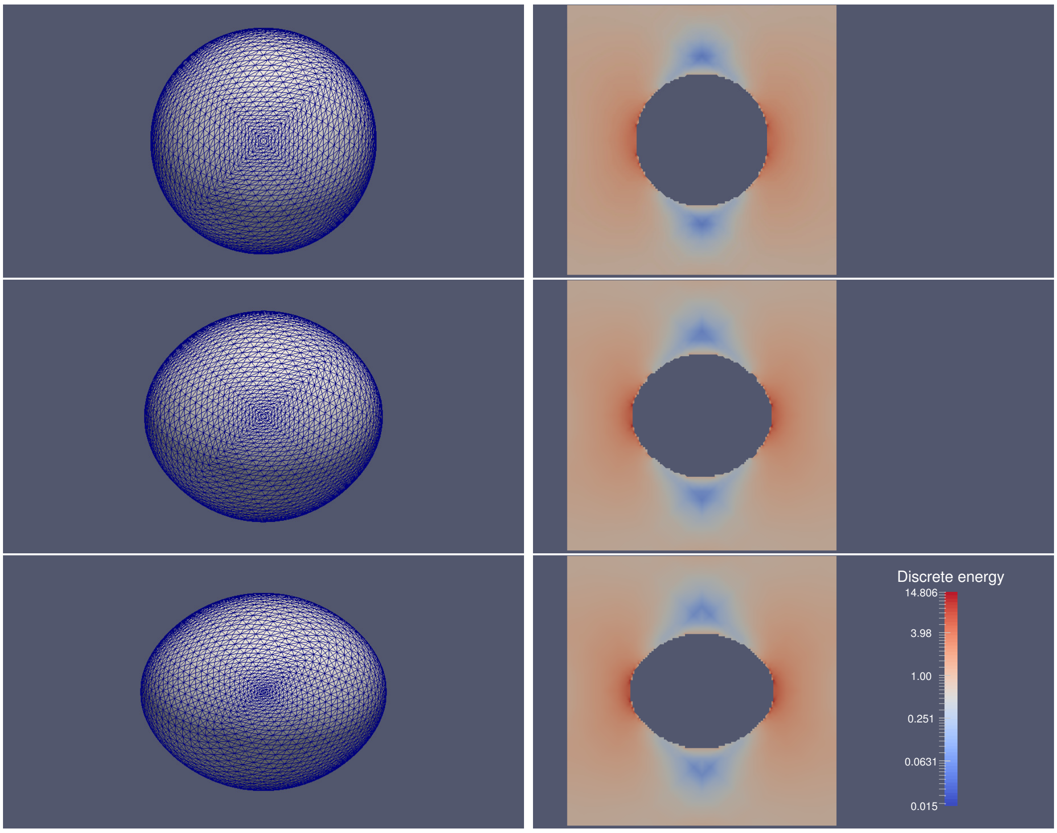

Figure 4.24: $\left(\alpha_2 = 0,\ \mu = \lambda = \frac{4}{5\pi},\ \underline{\underline{S}} = \begin{pmatrix} 0&0&0 \\ 0&1&0 \\ 0&0&0 \end{pmatrix}\right)$ Plots of the interface curve and the elastic energy density at times $t = 0, 2.5 \times 10^{-4}$ and $T = 0.02$. The energy density is colour coded with a logarithmic scale for the sake of visualisation.

5 THE PARAMETRIC APPROACH: THE *FITTED* METHOD

Let us recall what we described in Section 1.5. The numerical solution of partial differential equations for systems with moving boundaries can be approached in different ways. Three possible strategies for handling the interface have been introduced in the literature, namely *parametric approach* (§ 1.3.1), *level set approach* (§ 1.3.2) and *phase field approach* (§ 1.3.3).

In this thesis we consider the *parametric approach*. The user can employ two different techniques: an *unfitted approach* and a *fitted approach*, depending on whether the topological compatibility between bulk and interface meshes is preserved. In this chapter we present the *fitted approach* in detail. Our discussion expands on the exposition given in [105], where we analysed the *fitted* approach with $d = 2$ only.

The chapter is organised as follows: in Section 5.1 we give a definition of the *fitted approach*, highlighting its most important features. In Section 5.2 we introduce a *fitted* finite element approximation for the electro-stress migration problem described in Chapter 3. We investigate existence and uniqueness of our finite element approximation, and discuss some properties of its semidiscrete, continuous-in-time counterpart. In Section 5.3 we describe in detail all the mesh operations needed in our coupled bulk-interface framework. Special attention is paid to the routines for the *smoothing* of the bulk mesh, whose quality deteriorates over time due to the drifting of the voids present in the conductor. In particular, three different techniques, namely *Laplacian smoothing*, *harmonic smoothing* and *linear elastic smoothing*, are analysed and compared. Finally, in Section 5.4 we present several numerical simulations, for both $d = 2$ and $d = 3$, including a convergence experiment for a test case where the exact solution is known.

5.1 features of the *fitted* method

In Chapter 4 we introduced an *unfitted* finite element method for the electro-stress migration problem of our interest. In this chapter we consider a *fitted* approach, which means that the interface mesh is always part of the boundary of the bulk grid; see Figure 5.1 for an example with $d = 2$. Recall that in the *unfitted* approach the bulk mesh and the parametric interface mesh are totally independent. As a consequence, no smoothing of the bulk mesh needs to be performed. In addition, standard strategies for refinement and coarsening can be employed for the bulk mesh. However, the implementation of the *unfitted* approach requires a delicate communication between the two grids, and a simple approximation of (3.3a) and (3.7a) leads to additional approximation errors. Below we list the main features of the *fitted* approach considered in this chapter:

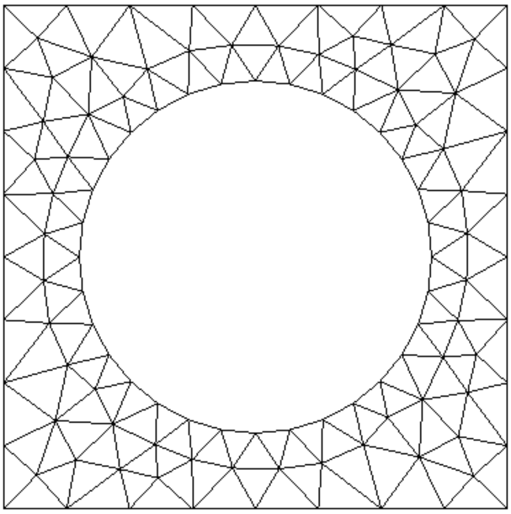

Figure 5.1: Example of a *fitted* interface mesh, with $d = 2$.

- only the exterior of the interface is triangulated, and so there is no necessity of filtering the bulk grid to identify a subset of elements on which the electric potential and the elastic displacement are to be computed;

- the vertices of the interface grid can be easily identified within the bulk mesh by a mapping, which needs to be computed only once at the beginning;
- the interpolation of bulk finite element functions over the the interface can be easily done with the previous mapping.

However, since the interface mesh drifts under the effects of surface tension, electric field and elastic energy, and the preservation of the consistency between the two grids implies that the topology of the bulk grid remains unmodified, at certain time steps a *mesh smoothing* may need to be applied, to avoid overlap between bulk elements. Once the quality of the mesh is compromised and cannot be smoothed any longer, a complete, costly re-meshing of the bulk region is required.

5.2 FINITE ELEMENT APPROXIMATION

We begin with the finite element approximation for quantities defined over the bulk mesh. Let $\mathcal{T}^m$ be a partitioning of Ω_+^m, a polyhedral approximation of $\Omega_+(t_m)$, into disjoint open d-dimensional simplices o with $h_o := \operatorname{diam}(o)$ and $h := \max_{o\in\mathcal{T}^m} h_o$ so that $\Omega_+^m = \bigcup_{o\in\mathcal{T}^m} \overline{o}$. Let Γ^m be the inner boundary of Ω_+^m, so that $\partial\Omega_+^m = \Gamma^m \cup \partial\Omega$. We can now define the standard finite element space of piecewise linear functions:

$$S^m := \{\chi \in C(\Omega_+^m) : \chi\,|_o \text{ is linear } \forall\, o \in \mathcal{T}^m\},$$

as well as

$$S_g^m := \{\chi \in S^m : \chi\,|_{\partial_2^\pm\Omega} = g^\pm\} \qquad \text{and} \qquad S_0^m := \{\chi \in S^m : \chi\,|_{\partial_2^\pm\Omega} = 0\}.$$

In addition, we can define the standard finite element space of vector-valued, piecewise linear functions:

$$\underline{Q}^m := [S^m]^d,$$

as well as

$$\underline{\widehat{Q}}^m := \{\vec{q} \in \underline{Q}^m : \int_{\Omega_+^m} \vec{q}\cdot\vec{v}\,dx = 0 \quad \forall\, \vec{v} \in \underline{\mathrm{RM}}\}.$$

We introduce now the finite element spaces needed for the approximation of quantities on the moving boundary $\Gamma(t)$. Recall that Γ^m is by definition the inner boundary of Ω^m_+: it is therefore a *polyhedral* surface, approximating the closed surface $\Gamma(t_m)$, $m = 0, \ldots, M$. Let us recall the notation already introduced in Section 2.2. Let Γ^m be a $(d-1)$-dimensional polyhedral surface, defined as $\Gamma^m = \bigcup_{j=1}^{J^m_\Gamma} \overline{\sigma^m_j}$, where $\{\sigma^m_j\}_{j=1}^{J^m_\Gamma}$ is a family of mutually disjoint open $(d-1)$-simplices with vertices $\{\vec{q}^m_k\}_{k=1}^{K^m_\Gamma}$. The parametric finite element spaces for the approximation of the position vector $\vec{x}$ and the curvature κ in (3.1)-(3.2) have been defined in (2.10). Moreover, recall (2.12) and (2.13) for the definition of the mass lumped inner product $\langle \cdot, \cdot \rangle^h_{\Gamma^m}$ and the unit normal $\vec{\nu}^m$ to Γ^m, respectively. We assume that the local ordering of the vertices used in (2.13) is such that $\vec{\nu}^m$ is the normal to Γ^m pointing into Ω^m_-.

We propose the following finite element approximation of (3.14a), (3.14b) and (3.14c)-(3.14d): Given Γ^0, a polyhedral approximation of Γ_0, for $m = 0, \ldots, M-1$ find functions $\{\Phi^{m+1}, \vec{U}^{m+1}, E^{m+1}, \vec{X}^{m+1}, \kappa^{m+1}\} \in S^m_g \times \underline{\widehat{Q}}^m \times S^m \times \underline{V}(\Gamma^m) \times W(\Gamma^m)$ such that for all $\psi \in S^m_0$, $\vec{\zeta} \in \underline{Q}^m$, $\chi \in W(\Gamma^m)$ and $\vec{\eta} \in \underline{V}(\Gamma^m)$

$$\int_{\Omega^m_+} \nabla \Phi^{m+1} \cdot \nabla \psi \, d\mathcal{L}^d = 0\,, \tag{5.1a}$$

$$\int_{\Omega^m_+} \mathcal{C}\,\underline{\underline{\mathcal{E}}}(\vec{U}^{m+1}) : \underline{\underline{\mathcal{E}}}(\vec{\zeta}) \, d\mathcal{L}^d = \int_{\partial\Omega} \vec{g} \cdot \vec{\zeta} \, d\mathcal{H}^{d-1}\,, \tag{5.1b}$$

$$\langle E^{m+1}, \chi \rangle_{\Gamma^m} = \langle E(\vec{U}^{m+1}), \chi \rangle_{\Gamma^m}\,, \tag{5.1c}$$

$$\begin{aligned} \left\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi\, \vec{\nu}^m \right\rangle^h_{\Gamma^m} - \alpha_1 \langle \nabla_s\, \kappa^{m+1}, \nabla_s\, \chi \rangle_{\Gamma^m} & \\ = - \langle \nabla_s\, (\alpha_2\, \Phi^{m+1} + \alpha_3\, E^{m+1}), \nabla_s\, \chi \rangle_{\Gamma^m}\,, & \end{aligned} \tag{5.1d}$$

$$\langle \kappa^{m+1}\, \vec{\nu}^m, \vec{\eta} \rangle^h_{\Gamma^m} + \langle \nabla_s\, \vec{X}^{m+1}, \nabla_s\, \vec{\eta} \rangle_{\Gamma^m} = 0\,. \tag{5.1e}$$

We note that E^{m+1} is the $L^2(\Gamma^m)$-projection of $E(\vec{U}^{m+1})$, which is a discontinuous function, onto the space of piecewise linear functions $W(\Gamma^m)$. Moreover, we note that the vector-valued function $\vec{g}$ in the right-hand side of (4.2b) denotes a traction boundary condition. For simplicity we consider $\vec{g} = \underline{\underline{\mathcal{S}}}\, \vec{\nu}_{\partial\Omega} = \mathcal{C}\, \underline{\underline{\mathcal{S}}}^*\, \vec{\nu}_{\partial\Omega}$, recall (3.8). Then set $\Gamma^{m+1} = \vec{X}^{m+1}(\Gamma^m)$ and find a suitable triangulation $\mathcal{T}^{m+1}$ of the domain Ω^{m+1}_+ with boundary

$\partial\Omega_+^{m+1} = \Gamma^{m+1} \cup \partial\Omega$. The latter aspect is described in more detail in § 5.3, below. In addition, it is worth noting that (5.1a) is a standard finite element approximation of (3.14a), (5.1b) is a standard finite element approximation of (3.14b), while (5.1d)-(5.1e) for the case $\alpha_2 = \alpha_3 = 0$ collapse to the scheme (2.17a)-(2.17b) for (3.1) with $\alpha_2 = \alpha_3 = 0$, i.e. for the geometric evolution law of *surface diffusion*.

Before we can proceed to prove existence and uniqueness of a solution to the system (5.1a)-(5.1e), let us recall the assumption $(\mathcal{A})$ from § 2.2. In particular, we note that Γ^m, which approximates the closed surface $\Gamma(t_m)$, $m = 0, \ldots, M$, easily verifies $(\mathcal{A})$, since it is a surface without self-intersections. We are now ready to prove existence and uniqueness of our discrete solution.

THEOREM 5.1. *Let the assumption* $(\mathcal{A})$ *from § 2.2 hold. Then there exists a unique solution* $\{\Phi^{m+1}, \vec{U}^{m+1}, E^{m+1}, \vec{X}^{m+1}, \kappa^{m+1}\} \in S_g^m \times \underline{\widehat{Q}}^m \times S^m \times \underline{V}(\Gamma^m) \times W(\Gamma^m)$ *to the system* (5.1a)-(5.1e).

Proof. We first notice that the equations for Φ^{m+1}, $\vec{U}^{m+1}$, E^{m+1} and $\{\vec{X}^{m+1}, \kappa^{m+1}\}$ decouple. Then we can straightforwardly adapt the same arguments used in the proof of Theorem 4.1, on noting that the only difference lies in solving the bulk equations over Ω_+^m and not over $\Omega_+^{m,h}$. ✓

REMARK 5.2. *As discussed in § 4.2.2, it is worthwhile to consider a continuous-in-time semidiscrete variant of the fully discrete scheme* (5.1a)-(5.1e). *Completely analogous to the unfitted approach, also in the fitted approach it is possible to show that the semidiscrete scheme satisfies the natural discrete analogue to the continuous volume preservation property* (3.11). *Moreover, it can be shown that in the case* $d = 2$ *the vertices of the discrete interface equidistribute, recall Theorem 2.8. While it does not seem possible to prove the equidistribution property for the fully discrete scheme* (5.1a)-(5.1e), *in practice we observe that the vertices on* Γ *asymptotically equidistribute. For the case* $d = 3$, *we can naturally extend to the fitted approach the considerations in Remark 4.3.*

REMARK 5.3. *Our analysis in § 4.3, concerning the solution methods for the bulk and interface linear systems within the unfitted approach, immediately carries over to the fitted approach with two minor modifications. First, the matrices arising from* *the bulk equations are assembled over Ω_+^m and not over $\Omega_+^{m,h}$. Second, the vertex-based interpolation of the bulk quantities in* (4.14) *is replaced by an edge-based interpolation (for the potential Φ^{m+1}) and L^2-projection (for the elastic energy density E^{m+1}), since the true exterior of the discrete interface Γ^m is immediately available.*

5.3 MESH OPERATIONS

Our *fitted* finite element approximation (5.1a)-(5.1e) is based on triangulations $\mathcal{T}^m$ of the discrete conducting regions Ω_+^m, which vary in time. In addition, recall that Γ^m is by definition the inner boundary of Ω_+^m. In our implementation this relationship between the interface mesh Γ^m and the bulk triangulation $\mathcal{T}^m$ is realised with the help of an index map that stores a list of bulk mesh vertices and their connectivities, that make up the interface Γ^m. This map needs to be computed only once, at time t_0.

We use an adaptive mesh for Ω_+^0, where we resolve the regions close to Γ^0 much finer than far away from the interface. In particular, we choose two integer parameters $N_f > N_c$ to be the number of subdivisions on $\partial^\pm\Omega$ and Γ^0, respectively. This choice will aid a direct comparison, in terms of degrees of freedom and CPU times, with our *unfitted* approach calculations, where we used adaptive meshes, as described in § 4.4.4. We then pass these two parameters to the mesh generator GMSH, which generates a Delaunay triangulation with the desired mesh width on $\partial^\pm\Omega$ and Γ^0; see Figure 5.2 for an example with $d = 2$, $N_f = 1024$ and $N_c = 4$.

Once the electro-stress migration problem 5.1 is solved, we can use the value of $\vec{X}^{m+1}$ at all the vertices of the interface grid in order to update the position of the corresponding vertices of the bulk grid. In theory, the remaining bulk vertices could be left as they are. However, in time the movement of the void will lead to bulk elements being deformed or even overlapping. Therefore, the application of some *smoothing* technique is required, in order to prevent the bulk grid from deteriorating too quickly.

This section is organised as follows. In 5.3.1 we describe what mesh *smoothing* is, recalling from the literature some mesh operations, aiming at reducing element distortion or early breakdowns in numerical simula-

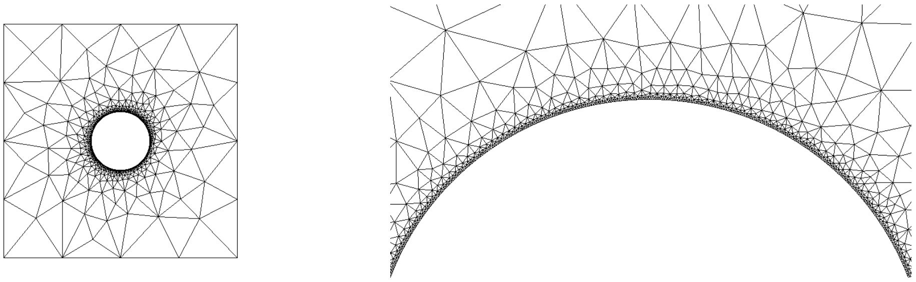

Figure 5.2: Example of a *fitted* interface mesh (left), with a zoomed portion of the domain (right), for the case $d = 2$, with $N_f = 1024$ and $N_c = 4$. We note that the package GMSH generates a Delaunay triangulation with the desired mesh width on $\partial^{\pm}\Omega$ and Γ^0.

tions. We then discuss three *smoothing* techniques, namely *Laplacian smoothing* (§ 5.3.2), *harmonic smoothing* (§ 5.3.3), and *linear elastic smoothing* (§ 5.3.4). § 5.3.5 compares the three aforementioned strategies for a test case with $d = 2$.

5.3.1 *Remarks on mesh smoothing*

We present here some considerations about *mesh quality* and *mesh smoothing*, mainly following the overview given in [90].

In industrial applications where finite element and finite volume discretisations are employed, numerical computations start with a geometric model, from which an unstructured simplicial mesh is usually generated. Before assembling all the relevant finite element quantities, the user needs to apply heuristic mesh improvement strategies (also known as *mesh clean-up*). The purpose of these initial controls is to reduce the likelihood of early breakdowns in numerical algorithms due to the poor quality of the underlying mesh. The overall quality of a given mesh is usually obtained via the calculation of a specific quality measure for all the elements of the mesh. Poor-quality meshes can have bad effects on interpolation error, discretisation error, and stiffness matrix conditioning.

Improvement strategies already presented in the literature can be classified into two main categories: *smoothing* and *topological transformations*. *Smoothing* consists of relocating one or more mesh vertices, in order to improve the quality of the elements adjoining them. *Smoothing*, however, does not change the connectivity of the mesh. *Topological transformations*, instead, are operations that modify the number of elements by removal or insertion of vertices, edges or faces. The topological structure of the mesh is therefore changed in the process. The two techniques can be performed in concert to obtain a better mesh.

In order to evaluate the quality of a given mesh, several *mesh quality metrics* have been proposed in the literature, and nearly all of them consist of scalar functions of the shape of mesh elements. The most popular metrics make use only of geometrical information about elements and vertices, without any dependence on the solution of the physical problem of interest. Moreover, as noted in [91], engineers tend to develop expertise as to what constitutes a good mesh for the problem at hand. This experimental knowledge is often a 'rule of thumb' that is translated into requirements on mesh quality. In addition, the convenience of such metrics lies in the fact that they can be easily evaluated by looping over the elements of the mesh, with no need to perform any visual inspection. Mesh quality metrics used for all the simulations presented in this thesis fall into this category. We refer to [110] for the case $d = 2$, where several triangle quality measures are investigated. The authors showed that some of these measures are equivalent, i.e. they display the same extremal and asymptotic behaviour. For the case $d = 3$, see [109, 99]. For a unified study of simplex shape measures and their generalisation to Riemannian spaces for anisotropic meshes, see [49].

We now discuss three common strategies for *mesh smoothing* and test them for a void electro-migration case with $d = 2$. More sophisticated *mesh smoothing* strategies, used in combination with optimisation techniques, can be found in [23, 117, 49, 36].

5.3.2 *Laplacian smoothing*

The most famous *smoothing* technique is the so-called *Laplacian smoothing*, see [81]. Every vertex of the grid is moved to the centroid of the vertices to which it is connected. Namely, for every vertex $\vec{x}$ we set

$$\vec{x}^* = \sum_{k=1}^{N(\vec{x})} \vec{x}_k \,, \tag{5.2}$$

where $N(\vec{x})$ represents the number of vertices to which $\vec{x}$ is connected and $\vec{x}^*$ represents the new location of $\vec{x}$. Special attention is to be paid to the vertices on the interface, which should not be relocated but remain in the position given by the vector $\vec{X}^{m+1}$, which we obtain solving the system of equations (5.1a)-(5.1e). Moreover, we need to impose a *free-slip* boundary condition for those vertices lying on the outer boundary, so that they do not move away from it.

The *smoothing* routine (5.2) is typically applied to each mesh vertex in sequence, and several iterations are performed, where each iteration moves every vertex only once. The stopping criterion is usually given in terms of relative variation between the new and the old location of each vertex. When the maximum variation is below a certain threshold, say $\varepsilon_{\text{smooth}}$, the iterative algorithm (5.2) stops and the new mesh, represented by the updated position of all the vertices, is returned.

Laplacian smoothing is a very simple technique, which requires neither the solution of a linear system nor the application of a sophisticated optimisation algorithm, hence its popularity. However, this naive method faces several limitations, see [79]. We recall here the main points for the benefit of the reader. First, it exhibits the drawback of shrinking the geometries to which it is applied. If the configuration to be smoothed is a mesh where no constraints are imposed on boundary points, the routine (5.2) reduces the initial mesh to a single point. Therefore, the choice of the stopping criterion and the boundary constraints are crucial issues. Second, *Laplacian smoothing* lacks motivation, because it is not directly connected to any specific mesh quality criterion. However, it can be reformulated as an optimisation-based problem, where the iteration matrix is given by a Markovian matrix with diagonal elements $m_{ii} = 0$ and, for the i-th vertex of the grid $\vec{x}_i$, off-diagonal elements $m_{ij} = \frac{1}{N(\vec{x}_i)}$ for pairs $\{i, j\}$ of neighbouring vertices, and $m_{ij} = 0$ otherwise. From well-established results on the eigenvalues of Markovian matrices, it follows that the convergence of this iterative algorithm is guaranteed. However, since the eigenvalues may be positive or negative, the convergence is not monotonic and oscillations could appear.

In order to overcome the drawbacks of the basic *Laplacian smoothing*, a number of variants have been proposed in the literature: see [120, 122, 79] and the references therein.

5.3.3 *Harmonic smoothing*

The second *smoothing* technique that we take into consideration is the so-called *harmonic smoothing*; see [70, § 5.2]. Precisely, for a given m, $0 \leqslant m \leqslant M-1$, we want to find the vector field $\vec{Z}^{m+1}$, which is the solution to the following problem:

$$-\triangle\vec{Z}^{m+1} = \vec{0} \qquad \text{in } \Omega^0_+\,, \tag{5.3a}$$

$$\vec{Z}^{m+1} = \vec{X}^{m+1} \qquad \text{on } \Gamma^0\,, \tag{5.3b}$$

$$\vec{Z}^{m+1} = \vec{id} \qquad \text{on } \partial\Omega\,, \tag{5.3c}$$

i.e. a Laplace problem for the vector field $\vec{Z}^{m+1}$, which represents the new location for every vertex of the grid. Similarly to the constraints imposed in the *Laplacian smoothing* case, the Dirichlet boundary condition (5.3b) prescribes a specific position for the vertices on the interface, which we naturally choose to be the position $\vec{X}^{m+1}$ obtained from (5.1a)-(5.1e). The boundary condition (5.3c) prevents vertices on the outer boundary from moving away from it. In order to introduce a finite element approximation of (5.3), we need to define the following:

$$\underline{S}^0_{\vec{X}^{m+1}} := \{\vec{\chi} \in \underline{Q}^0 : \vec{\chi}\,|_{\Gamma^0} = \vec{X}^{m+1}\} \qquad \text{and} \qquad \underline{S}^0_{\vec{0}} := \{\vec{\chi} \in \underline{Q}^0 : \vec{\chi}\,|_{\Gamma^0} = \vec{0}\}\,. \tag{5.4}$$

The finite element approximation of (5.3) is then the following: Find $\vec{Z}^{m+1} \in \underline{S}^0_{\vec{X}^{m+1}}$ such that

$$\int_{\Omega^0_+} \nabla\vec{Z}^{m+1} : \nabla\vec{\zeta}\, d\mathcal{L}^d = 0 \qquad \forall\, \vec{\zeta} \in \underline{S}^0_{\vec{0}}\,, \tag{5.5}$$

where, with a slight abuse of notation, we have used $\vec{Z}^{m+1}$ to denote the solutions to both the strong formulation (5.3) and its finite element approximation (5.5). We note that it is crucial to choose a nice initial bulk triangulation $\mathcal{T}^0$ in order that *harmonic smoothing* may recover nice meshes. In fact, on choosing $\mathcal{T}^0$ with ill-shaped elements and $\vec{X} = \vec{id}$ for the Dirichlet boundary conditions (5.3b), the only solution to (5.5) would clearly be $\vec{Z} = \vec{id}$, i.e. no smoothing would be performed.

Equation (5.5) gives rise to a symmetric linear system, which is solved over Ω_+^0 for all $m = 0, \dots, M-1$. This implies that the associated stiffness matrix needs to be assembled only once. The right-hand side of (5.5), instead, is to be modified for each m, in order to include the Dirichlet boundary conditions (5.3b). The system (5.5) is symmetric and positive definite, and we can solve it with the sparse factorisation package UMFPACK. Since the system is always solved over Ω_+^0, the associated stiffness matrix needs to be factorised only once. This features makes this *smoothing* technique cheap in practice. The value of $\vec{Z}^{m+1}$ at every vertex represents the new position of the vertex in the smoothed grid.

5.3.4 *Linear elastic smoothing*

The third smoothing technique that we take into consideration is the so-called *linear elastic smoothing*; see [103, 88, 87], where this technique was applied to aerodynamic design optimisation and viscous layer insertion for fluid dynamics problems, respectively. Specifically, *linear elastic smoothing* was successfully applied to aerodynamic profiles where large surface deformations were observed and unstructured meshes with highly distorted cells were employed. Linear elasticity relationships, borrowed from structural mechanics, provide the governing partial differential equations for smoothly relocating the position of the vertices of the existing mesh. Precisely, for a given m, $0 \leqslant m \leqslant M-1$, we want to find the displacement field $\vec{\Psi}^{m+1}$, which is the solution to the following problem:

$$\begin{aligned} \nabla \cdot (2\,\underline{\underline{\mathcal{E}}}(\vec{\Psi}^{m+1}) + \mathrm{Tr}(\underline{\underline{\mathcal{E}}}(\vec{\Psi}^{m+1}))) &= \vec{0} && \text{in } \Omega_+^m\,, && (5.6a)\\ \vec{\Psi}^{m+1} &= \delta\vec{X}^{m+1} && \text{on } \Gamma^m\,, && (5.6b)\\ \vec{\Psi}^{m+1} \cdot \vec{\nu}_{\partial\Omega} &= 0 && \text{on } \partial\Omega\,, && (5.6c) \end{aligned}$$

i.e. a linear elasticity problem with Lamé moduli $\mu = \lambda = 1$. The Dirichlet boundary condition (5.6b) prescribes a specific displacement on the interface, which we naturally choose to be the variation $\delta\vec{X}^{m+1} := \vec{X}^{m+1} - \vec{\mathrm{id}}^m$ from (2.57). Note that (5.6b) slightly differs from (5.3b), since here we are solving for a displacement, i.e. a variation in the position of the vertices, and not for the actual position itself. The *free-slip* boundary condition (5.6c) prevents vertices on the outer boundary from moving away from it. In order to introduce a finite element approximation of (5.6), we need to define the following:

$$\underline{S}^m_{\delta\vec{X}} := \{\vec{\chi} \in \underline{Q}^m : \vec{\chi}\,|_{\Gamma^m} = \delta\vec{X}^{m+1}\} \qquad \text{and} \qquad \underline{S}^m_{\vec{0}} := \{\vec{\chi} \in \underline{Q}^m : \vec{\chi}\,|_{\Gamma^m} = \vec{0}\}\,. \quad (5.7)$$

The finite element approximation of (5.6) is then the following: Find $\vec{\Psi}^{m+1} \in \underline{S}^m_{\delta\vec{X}}$ such that

$$\int_{\Omega_+^m} (2\,\underline{\underline{\mathcal{E}}}(\vec{\Psi}^{m+1}) : \underline{\underline{\mathcal{E}}}(\vec{\zeta}) + \nabla\cdot(\vec{\Psi}^{m+1})\,\nabla\cdot(\vec{\zeta}))\,\mathrm{d}\mathcal{L}^d = 0 \qquad \forall\,\vec{\zeta} \in \underline{S}^m_{\vec{0}}\,, \quad (5.8)$$

where, with a slight abuse of notation, we have used $\vec{\Psi}^{m+1}$ to denote the solutions to both the strong formulation (5.6) and its finite element approximation (5.8). Equation (5.8) gives rise to a symmetric linear system. Both matrix and right-hand side need to be adjusted appropriately in order to include both Dirichlet and *free-slip* boundary conditions. Such a system is symmetric and positive definite, and we can solve it with the sparse factorisation package UMFPACK. The value of $\vec{\Psi}^{m+1}$ at every vertex represents the variation that has to be applied to the old position of the vertex to obtain its new one in the smoothed grid.

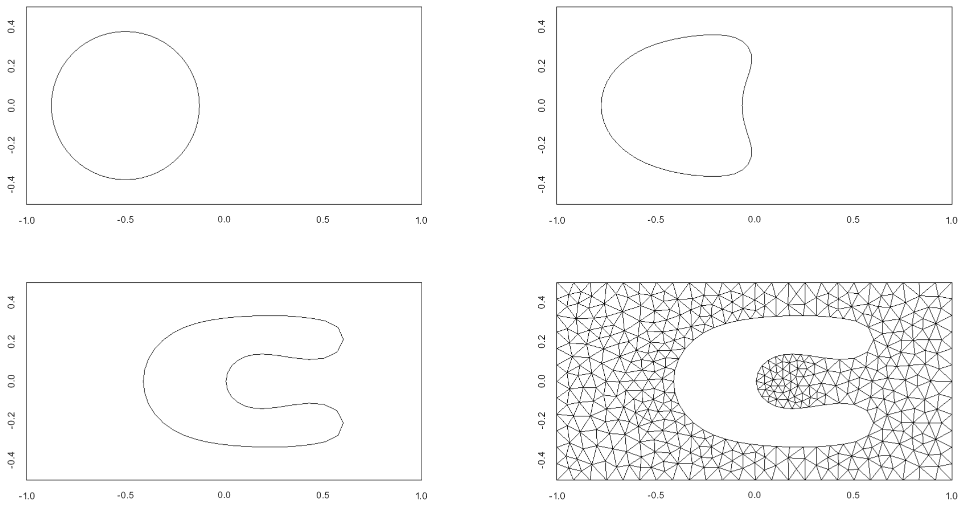

Figure 5.3: ($\alpha_1 = \frac{1}{16}\pi^2, \alpha_2 = \frac{256}{9}\pi^2, \alpha_3 = 0$) Plots of the interface curve at times $t = 0, 9 \times 10^{-5}, T = 3.6 \times 10^{-4}$, and adaptive bulk mesh at time $t = T$, with *linear elastic smoothing*.

5.3.5 *Comparison between different smoothing techniques*

We now compare the three *smoothing* techniques introduced above in a simple test case for $d = 2$, where only electro-migration is present. The geometry for this experiment corresponds to [104, Fig. 4]. We choose the radius of the initially circular void to be relatively large compared to the width of the conductor, $2L_2$. We use the following parameters: $L_1 = 1$, $L_2 = 0.5$, $\alpha_1 = \frac{1}{16}\pi^2$, $\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$, $\tau = 4.5 \times 10^{-7}$, $T = 3.6 \times 10^{-4}$. As initial data we choose a circle with radius 0.375 and centre $(-0.5, 0)$; the bulk refinement parameters are $N_f = 56$ and $N_c = 12$, respectively. We choose a coarse interface, given by $K_\Gamma^m = J_\Gamma^m = 56$. We note that the *smoothing* routine is applied at each time step. In Figure 5.3 we plot the results of the simulation at times $t = 0, 8 \times 10^{-5}$, and $t = T$, when *linear elastic smoothing* is applied. In addition, we investigate the quality of the interface mesh. Recall that in (2.33) we defined $r_h := h_{\Gamma^m}/l_{\Gamma^m}$ where h_{Γ^m} and l_{Γ^m} represent the maximum and minimum segment of Γ^m, respectively. The temporal evolution of the indicator r_h is shown in Figure 5.4, where we note that the vertices on the interface mesh remain well distributed.

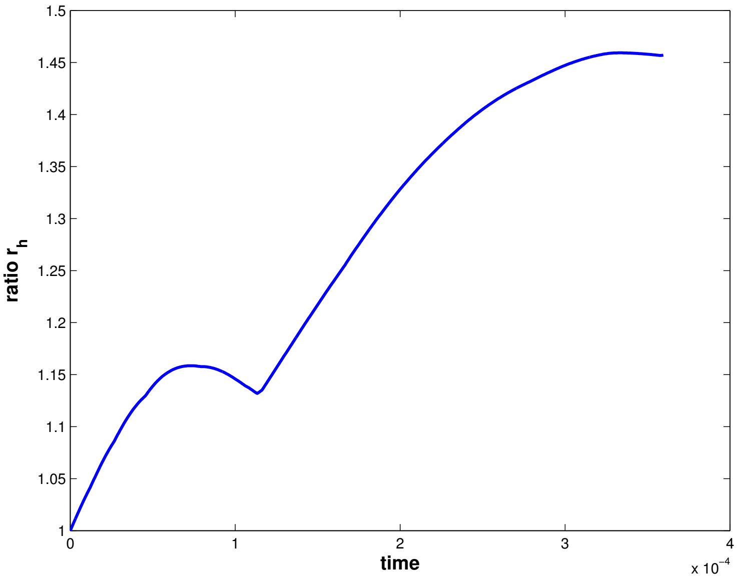

Figure 5.4: Plot of the ratio r_h (eq. (2.33)) for the interface mesh in Figure 5.3, with *linear elastic smoothing* applied to the bulk mesh. We note that the vertices on the interface mesh remain well distributed.

In order to compare the three different *smoothing* algorithms, in Figure 5.5 we plot the temporal evolution of the ratio

$$r_A := \max_{o \in \mathcal{T}^m} \mathcal{L}^d(o) \,/ \min_{o \in \mathcal{T}^m} \mathcal{L}^d(o) \tag{5.9}$$

for *Laplacian smoothing* (red), *harmonic smoothing* (green) and *linear elastic smoothing* (blue). For *Laplacian smoothing*, we take $\varepsilon_{\text{smooth}} = 10^{-6}$ as a stopping criterion for the iterative procedure (5.2).

Smoothing alone, however, does not prevent the bulk grid from deteriorating in the long run. This is due to the physical motion of the void through the conductor. In order to assess the bulk mesh quality, we make of use of the *minimum* and *maximum angle* as quality metric; namely, we check all the angles of all the triangles of the smoothed mesh, and if we find any angle $\theta \leqslant 20°$ or $\theta \geqslant 130°$, we then re-mesh the conducting region Ω_+^m altogether. The re-meshing is performed with the mesh generator GMSH, to which we pass a geometric model where the position of the vertices on both the inner and outer boundary of Ω_+^m is preserved.

Considering again Figure 5.5, we notice that *linear elastic smoothing* performs best, since it requires fewer complete re-meshings than the other two

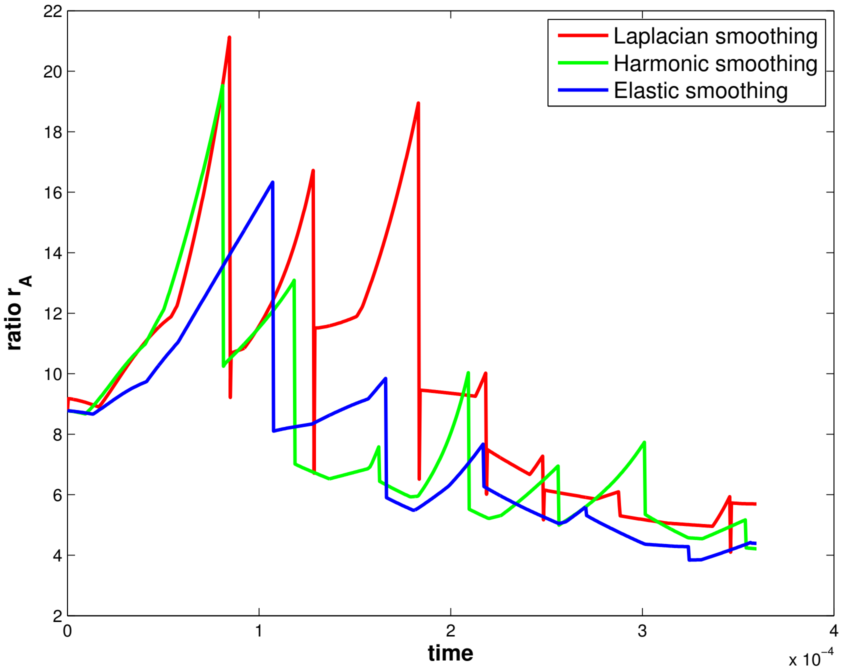

Figure 5.5: Plots of the ratio r_A (eq. (5.9)) for *Laplacian smoothing* (red), *harmonic smoothing* (green) and *elastic smoothing* (blue), for an initially circular void under electro-migration (coarser mesh). The discontinuities in the value of r_A correspond to the re-meshings of the bulk, performed with the help of the package GMSH. *Linear elastic smoothing* requires six re-meshings, *harmonic smoothing* and *Laplacian smoothing* seven re-meshings.

approaches. The discontinuities in the value of r_A correspond to the re-meshing of the bulk, performed with the help of the package GMSH. In particular, at time $t = 8.1 \times 10^{-5}$, *harmonic smoothing* cannot recover any longer a bulk mesh where the *minimum-maximum angle* check is not violated. Therefore, a complete re-meshing is necessary. For a visual comparison between the meshes obtained by the three different *smoothing* strategies at time $t = 8.1 \times 10^{-5}$, see Figure 5.6.

We repeat the experiment with a much finer grid. We now choose $N_f = K_\Gamma^m = J_\Gamma^m = 1024$ and $N_c = 16$, keeping all the other parameters as in the previous case. In Figure 5.7 we plot the ratio r_A for the three approaches, and again we see that *linear elastic smoothing* performs best. For this reason,

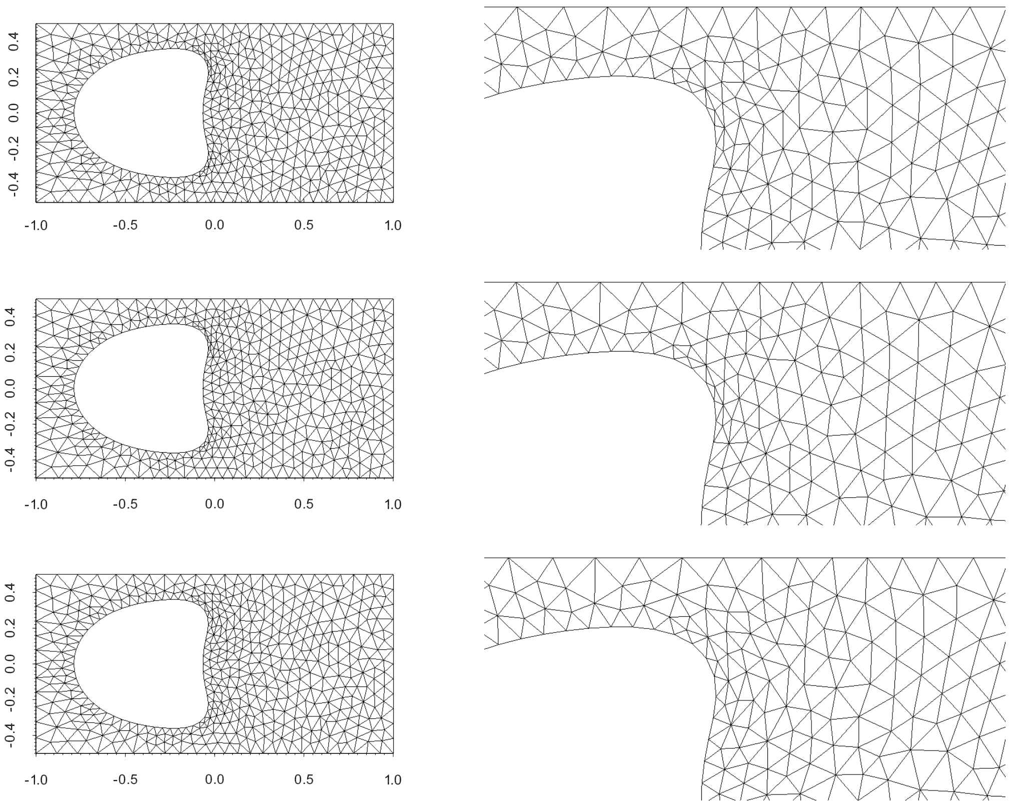

Figure 5.6: $(\alpha_1 = \frac{1}{16}\pi^2, \alpha_2 = \frac{256}{9}\pi^2, \alpha_3 = 0)$ Plots of the interface curve at time $t = 8.1 \times 10^{-5}$ for *harmonic smoothing* (top), *Laplacian smoothing* (centre) and *elastic smoothing* (bottom). At this time step, *harmonic smoothing* cannot smooth the bulk mesh any longer, so a complete re-meshing is necessary.

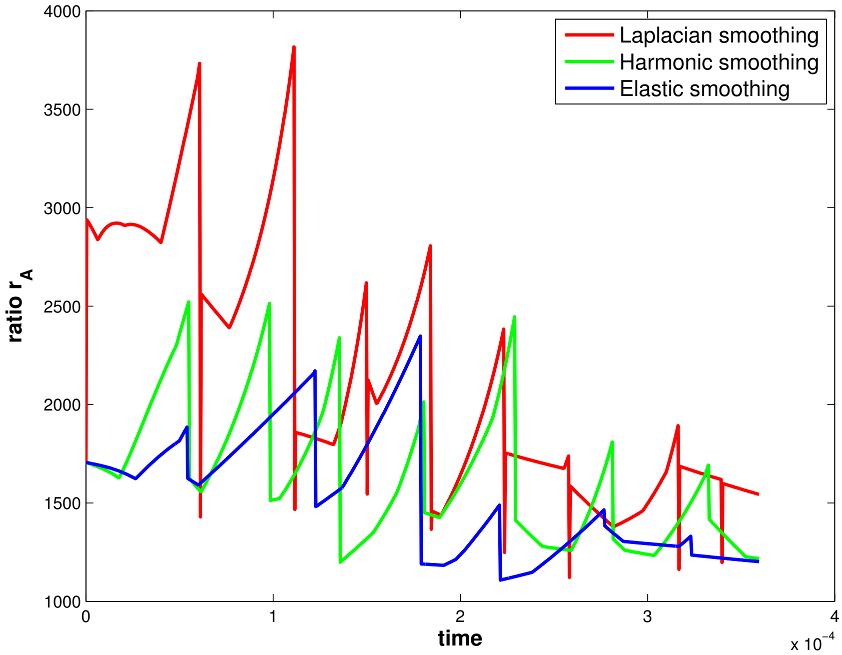

Figure 5.7: Plots of the ratio r_A (eq. (5.9)) for *Laplacian smoothing* (red), *harmonic smoothing* (green) and *elastic smoothing* (blue) for an initially circular void under electro-migration (finer mesh). The discontinuities in the value of r_A correspond to the re-meshings of the bulk, performed with the help of the package GMSH. *Linear elastic smoothing* requires six re-meshings, *harmonic smoothing* seven re-meshings, *Laplacian smoothing* eight re-meshings.

we employ *linear elastic smoothing* at each time step in all the numerical simulations presented from now on.

For the case $d = 3$, we apply *linear elastic smoothing* to the bulk, tetrahedral mesh as well. In order to assess whether the mesh quality is compromised, we make use of the quality control routines already available in the mesh generator GMSH. In particular, for every tetrahedron $o \in \mathcal{T}^m$, we calculate the dimensionless index q_{tetra} defined as

$$q_{\text{tetra}} = 6 \times \sqrt{6} \times \frac{\mathcal{L}^3(o)}{(\sum_{i=1}^{4} \mathcal{H}^2(f_i)) \cdot \max_{i=1}^{6} \mathcal{H}^1(e_i)}, \tag{5.10}$$

where f_i and e_i denote the i-th face and edge of tetrahedron, respectively. Note that multiplicative coefficient $6 \times \sqrt{6}$ simply scales q_{tetra} such that it attains the value 1 for a regular tetrahedron. For the sake of comparison, it is worth mentioning that the quality index q_{tetra} is constant for all the three-dimensional meshes used in Chapter 4. In fact, for all the tetrahedra in such *unfitted* bulk grids, three out of four triangular faces are right-angled, isosceles triangles. Specifically, the quality index q_{tetra} attains the value $\frac{2\times\sqrt{6}}{\sqrt{2}\times(3+\sqrt{3})} \approx 0.732$. If we find any tetrahedron with quality index $q_{\text{tetra}} < 10\%$, we then re-mesh the conducting region Ω_+^m altogether. Similarly to the case for $d = 2$, the re-meshing of the bulk is performed with the package GMSH. The user has to pass to GMSH the vertices located on both the inner and the outer boundary of Ω_+^m, which are used to generate a new bulk mesh.

5.4 NUMERICAL SIMULATIONS

In this section we present several numerical tests for the *fitted* method. Unless stated otherwise, we use uniform time steps $\tau_m = \tau$, $m = 0, \ldots, M-1$, for all the experiments in this section. Similarly to § 4.5, we set $\alpha_1 = \frac{1}{16}\pi^2$. We first present the simulations for $d = 2$ and then for $d = 3$.

We begin with a convergence experiment for the true solution (3.15) and compare the results with the corresponding experiments shown in Table 4.1. Recall that (3.15) describes a circular void that moves at constant speed through an infinite conductor. It is worthwhile to note that in this particular case we prescribe Dirichlet boundary conditions on the entire outer boundary of Ω, and not only on $\partial_2\Omega$. Here only electro-migration is considered, therefore we set $\alpha_3 = 0$. The definition of the error $\mathcal{E}_\Gamma$ is precisely the same introduced in (4.21), while the norm of the error on the bulk is now the following: $\mathcal{E}_{\text{bulk}} := \max_{m=1,\ldots,M} \|\Phi^m - \phi(\cdot, t_m)\|_{H^1(\Omega_+^{m-1})}$ between Φ and the exact electric potential ϕ on the interval $[0, T]$. All the parameters are the same used for the simulations in Table 4.1. Errors and experimental orders of convergence are listed in Table 5.1, where we reproduce [105, Table 1]. We observe a convergence of $\mathcal{O}(h_\Gamma^2)$ in the measured error for the interface, and $\mathcal{O}(h_\Gamma)$ for the H^1-error for the approximation of the electric potential in the bulk. Comparing Tables 4.1 and 5.1, it appears that the *fitted* method converges with a faster rate. In addition, we note that in each case the absolute errors for the *fitted* approach runs are smaller than the

i	$\tau \cdot 10^6$	$h_\Gamma \cdot 10^3$	$h_{bnd} \cdot 10^2$	$\mathcal{E}_{bulk} \cdot 10^2$	EOC_{bulk}	$\mathcal{E}_\Gamma \cdot 10^3$	EOC_Γ	CPU time	
								Total	Smoothing
0	8	7.81	12.5	4.369	–	2.378	–	15s	2s
1	2	3.91	6.25	2.201	0.99	0.625	1.93	201s	39s
2	0.5	1.95	3.13	1.123	0.97	0.155	2.01	3287s	764s
3	0.125	0.977	1.56	0.564	0.99	0.0399	1.96	72551s	22143s

Table 5.1: Results of the convergence test for the *fitted* case, reproduced from [105, Table 1], with h_Γ and h_{bnd} being the width of the mesh on the inner and the outer boundary of Ω, respectively. Note that the average number of bulk degrees of freedom for the four runs are 886, 3125, 12033 and 48977, respectively.

corresponding errors for the *unfitted* approach counterparts. However, this result is achieved at the cost of higher CPU times.

Our next experiment corresponds to the *unfitted* approach test analysed in Figures 4.7 and 4.9, see also [104, Fig. 5], [20, Fig. 2] and [27, Fig. 4]. Only electro-migration is involved. We keep the same simulation parameters used for the *unfitted* runs. The total CPU time was 2300s, with mesh smoothing and re-meshing accounting for 27% of that time. The re-meshing routine was applied six times. The number of bulk degrees of freedom for Φ^{m+1} was between 26715 (minimum value) and 35023 (maximum value). In Figure 5.9 we plot the temporal evolution of the interface mesh quality indicator r_h, recall (2.33). We note that the vertices of the interface mesh remain well distributed. Moreover, r_h attains values slightly lower than the ones shown in Figure 4.8 for the *unfitted* experiment. In order to make a direct comparison with Figure 4.9, we impose Robin boundary conditions (4.23), which means that (5.1a) needs to be replaced by

$$\int_{\Omega_+^m} \nabla\Phi^{m+1} \cdot \nabla\psi \, d\mathcal{L}^d + \frac{1}{2}\int_{\partial_2\Omega} \Phi^{m+1}\,\psi \, d\mathcal{H}^{d-1} = \frac{1}{2}\int_{\partial_2\Omega} g_R\,\psi \, d\mathcal{H}^{d-1}$$

for all test functions $\psi \in S^m$. The total CPU time was 3684s, with mesh smoothing and re-meshing accounting for 26% of that time. The re-meshing routine was applied ten times. The number of bulk degrees of freedom for Φ^{m+1} was between 26715 (minimum value) and 35551 (maximum value). In Figure 5.10 we plot the results of the simulation at times $t = 0, 8 \times 10^{-5}$,

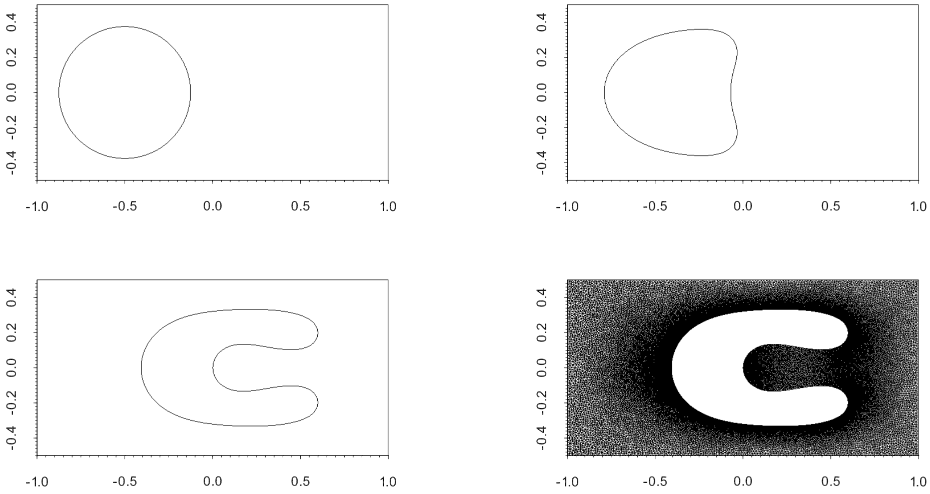

Figure 5.8: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$) Plots of the interface curve at times $t = 0$, 8×10^{-5}, $T = 3.6 \times 10^{-4}$, and bulk mesh at time $t = T$.

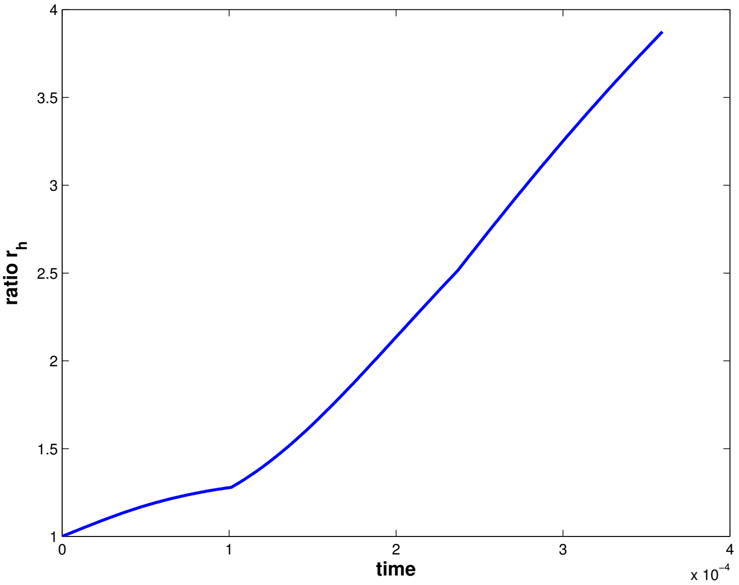

Figure 5.9: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$) Plots of the quality mesh indicator r_h for the interface mesh plotted in Figure 5.8.

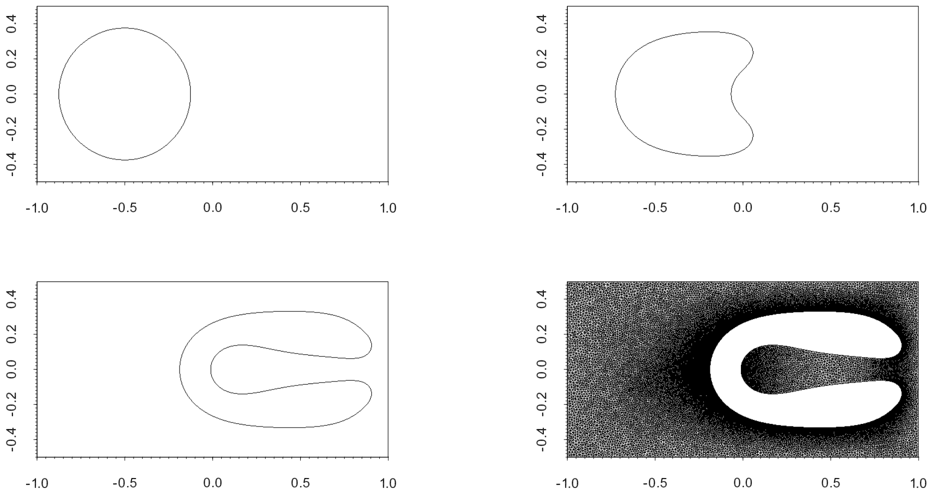

Figure 5.10: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface curve at times $t = 0, 8 \times 10^{-5}$, $T = 3.6 \times 10^{-4}$, and bulk mesh at time $t = T$.

and $T = 3.6 \times 10^{-4}$. The temporal evolution of the interface mesh quality indicator r_h is shown in Figure 5.11, where we see that the vertices of the interface mesh remain well distributed. Also in this case r_h attains values slightly lower than the ones shown in Figure 4.10 for the *unfitted* experiment. We note the good agreement between the plots with the *unfitted* approach in Figure 4.9, the plots with the *fitted* approach in Figure 5.10 and the results in [20, Fig. 2] and [27, Fig. 4].

Our next experiment has been proposed in [105, Fig. 3]. We report compare the results of the simulations to Figure 4.13 for the *unfitted* case. The test involves both electro-migration and stress-migration. The discretisation parameters are identical to the ones used in the *unfitted* experiment. The total CPU time was 14403s, with mesh smoothing and re-meshings accounting for 12% of that time. We applied the re-meshing routine nine times. The total number of bulk degrees of freedom (for both Φ^{m+1} and $\vec{U}^{m+1}$) was between 30378 (minimum value) and 33259 (maximum value). As discussed for the *unfitted* results in Figure 4.13, we note that the void moves through the conductor due to the presence of the electric field, with elastic stress contributing to flattening the front of the void; see Figure 5.12 for the com-

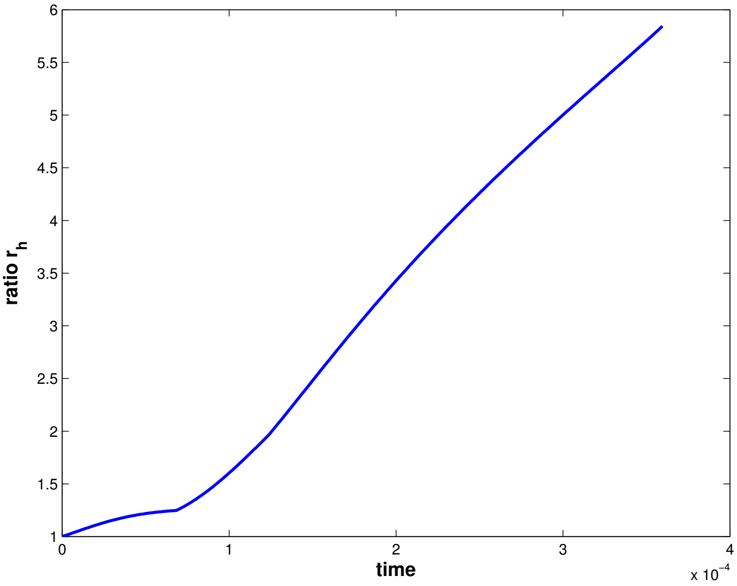

Figure 5.11: ($\alpha_2 = \frac{256}{9}\pi^2$, $\alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the quality mesh indicator r_h for the interface mesh plotted in Figure 5.10.

plete evolution of the system. In addition, from Figure 5.12 (bottom) we note that the evolution with the *fitted* method (black curve) is slightly faster than the one with the *unfitted* method (red curve). In order to evaluate the contribution to the drifting of the void produced by the two external loadings, we now repeat the experiment by considering electro-migration and stress-migration separately. In Figure 5.13 we plot the results of the simulations with $\alpha_2 = 0$, $\alpha_3 = \frac{1}{8}\pi$. We note that the void does not drift, but slightly elongates in the vertical direction. We note the good agreement with the plots in Figure 4.14. In Figure 5.14, instead, we plot the results of the simulations with $\alpha_2 = 10\pi^2$, $\alpha_3 = 0$. We note that the void drifts along the conductor, keeping its original circular profile. Also in this case we note the good agreement with the plots in Figure 4.15.

Our next test corresponds to the experiment already analysed with the *unfitted* approach in Figures 4.16 and 4.17. The discretisation parameters are identical to the ones used in the *unfitted* experiment. The total CPU time was 21587s, with mesh smoothing and re-meshings accounting for 16% of that time. Here the re-meshing routine was applied only once. The number

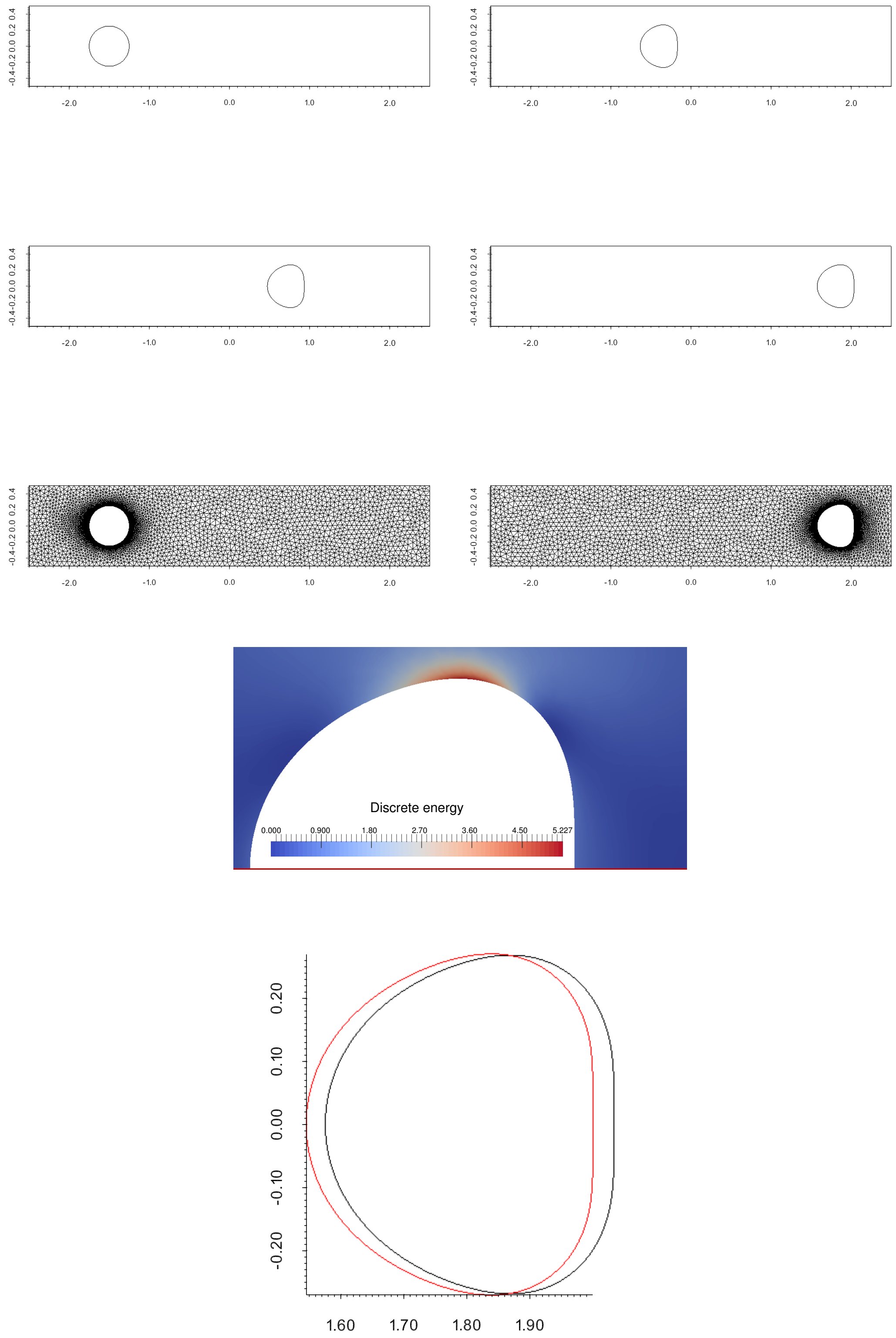

Figure 5.12: ($\alpha_2 = 10\,\pi^2$, $\mu = \lambda = 1$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$) Plots of the interface curve at times $t = 0$, 1.25×10^{-3}, 2.5×10^{-3} and $T = 3.75 \times 10^{-3}$; bulk mesh at time $t = 0$, T and the elastic energy density at time $t = T$; comparison between *fitted* (black) and *unfitted* (red) approach at time $t = T$.

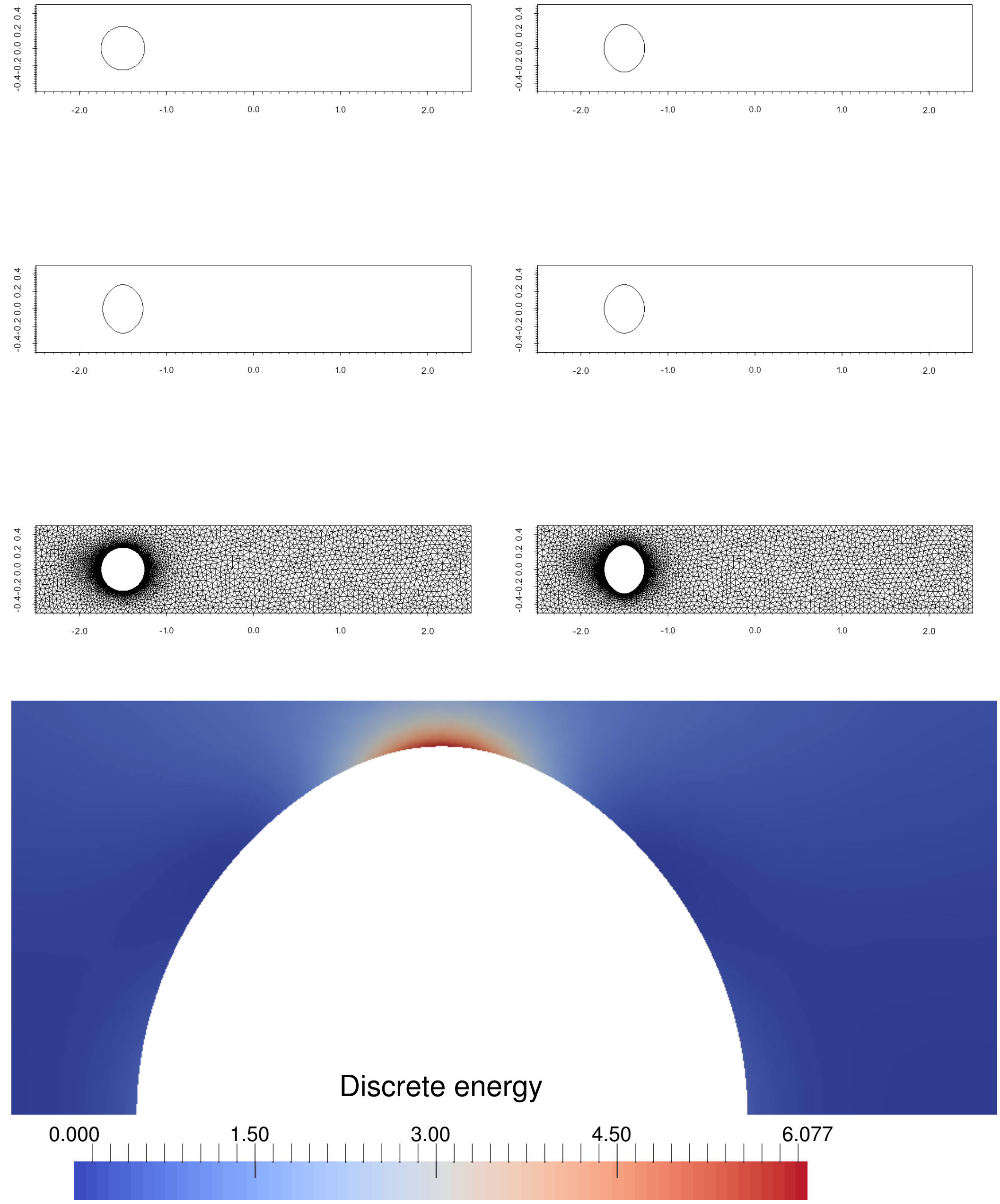

Figure 5.13: ($\alpha_2 = 0$, $\mu = \lambda = 1$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$) Plots of the interface curve at times $t = 0$, 1.25×10^{-3}, 2.5×10^{-3} and $T = 3.75 \times 10^{-3}$; bulk mesh at time $t = 0$, T and the elastic energy density at time $t = T$.

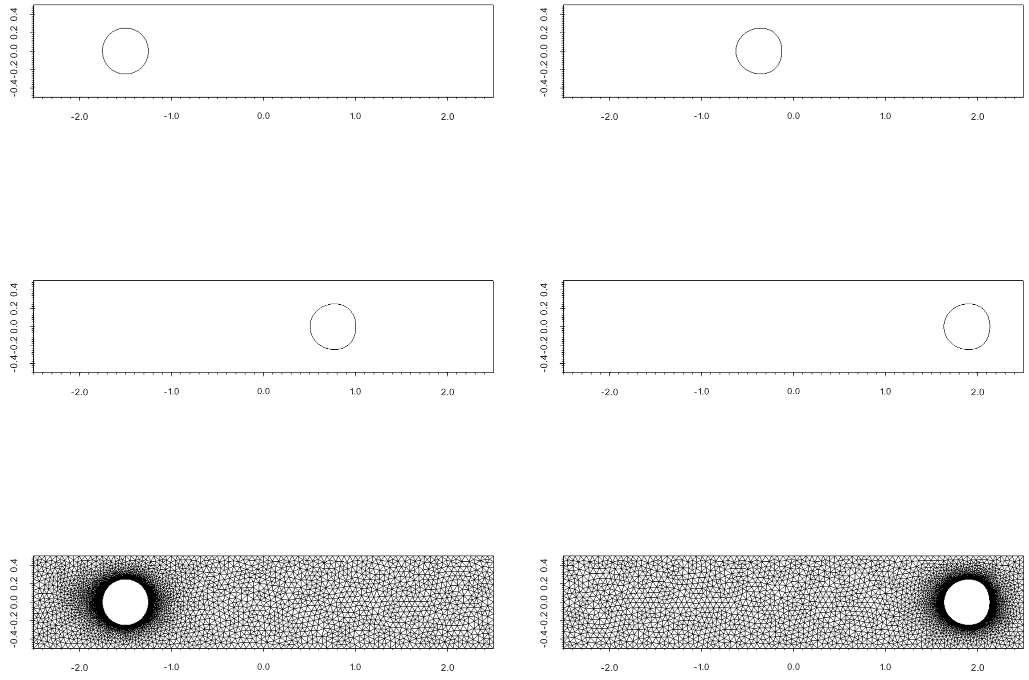

Figure 5.14: ($\alpha_2 = 10\,\pi^2$, $\alpha_3 = 0$) Plots of the interface curve at times $t = 0$, 1.25×10^{-3}, 2.5×10^{-3} and $T = 3.75 \times 10^{-3}$, and bulk mesh at time $t = 0$, T.

of degrees of freedom for $\vec{U}^{m+1}$ was between 143500 (minimum value) and 158876 (maximum value). In Figure 5.15 we plot the results of the simulation at times $t = 0$ and $t = T$. As in Figure 4.17, we observe that for larger times T a singularity develops. In particular, the two voids exhibit sharp corners and the elastic energy becomes unbounded. This can be seen in Figure 5.16, where the developing singularity leads to a breakdown in the numerical approximation. We now investigate whether this breakdown is related to numerical errors. To this end, we calculate the total discrete energy $\alpha_1 \mathcal{H}^{d-1}(\Gamma^m) + \alpha_3 \left(\int_{\Omega_+^m} E(\vec{U}^{m+1}) \, d\mathcal{L}^d - \int_{\partial\Omega} \vec{g} \cdot \vec{U}^{m+1} \, d\mathcal{H}^{d-1}\right)$ and plot it in Figure 5.16 (bottom). We note that the discrete energy is monotonically decreasing, recall (3.12). Hence the observed singularity appears to be consistent with the sharp interface model (3.1), (3.7a)-(3.7c) itself. Moreover, we note that the results agree with the temporal evolution in Figures 4.16 and 4.17, where we also observed a breakdown in the numerical simulations and a similar decay of the total discrete energy.

Our last experiment for $d = 2$ has been presented in [105, Figs. 5 and 6]. Only stress-migration is considered. The discretisation parameters are identical to the ones used for the *unfitted* experiment in Figures 4.18 and 4.19. The total CPU time was 18828s, with mesh smoothing and re-meshings accounting for 14% of that time. Here the re-meshing routine was applied only once. The number of degrees of freedom for $\vec{U}^{m+1}$ was between 62290 (minimum value) and 67361 (maximum value). In Figure 5.17 we plot the results of the simulation at times $t = 0$, 5×10^{-4}, 10^{-3} and $T = 1.5 \times 10^{-3}$. We observe that also for this experiment a singularity develops for larger times T. In particular, the interface shows corners and the elastic energy becomes unbounded. This can be seen in Figure 5.18. However, the presented plot of the total discrete energy indicates that the singularity is not due to numerical errors. In fact, the discrete energy is monotonically decreasing, recall also (3.12). Again, the observed singularity appears to be consistent with the sharp interface model (3.1), (3.7a)-(3.7c) itself and with the *unfitted* plots in Figures 4.18 and 4.19.

Our first experiment with $d = 3$ corresponds to [3, Fig. 8] and Figure 4.22. It involves electro-migration only, i.e. $\alpha_3 = 0$, and illustrates a fully three-dimensional situation. We keep the same parameters used for the

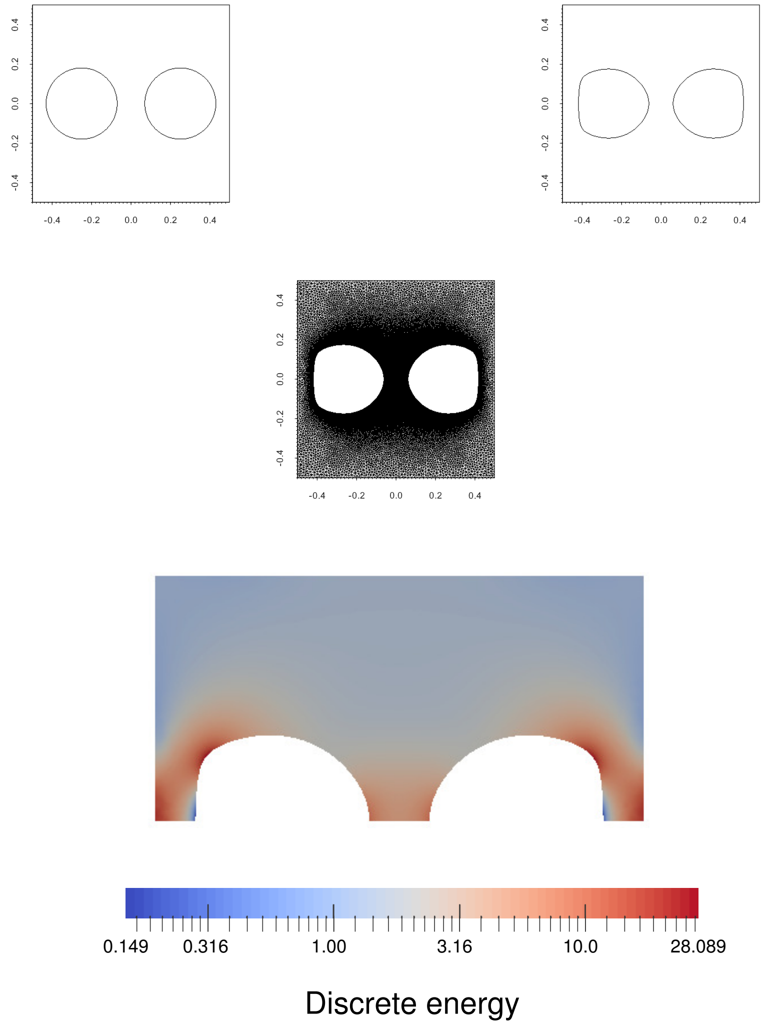

Figure 5.15: ($\alpha_2 = 0, \mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right)$) Plots of the interface curve at times $t = 0$ and $T = 5 \times 10^{-5}$; bulk mesh and elastic energy density at $t = T$. The energy density is colour coded with a logarithmic scale for the sake of visualisation.

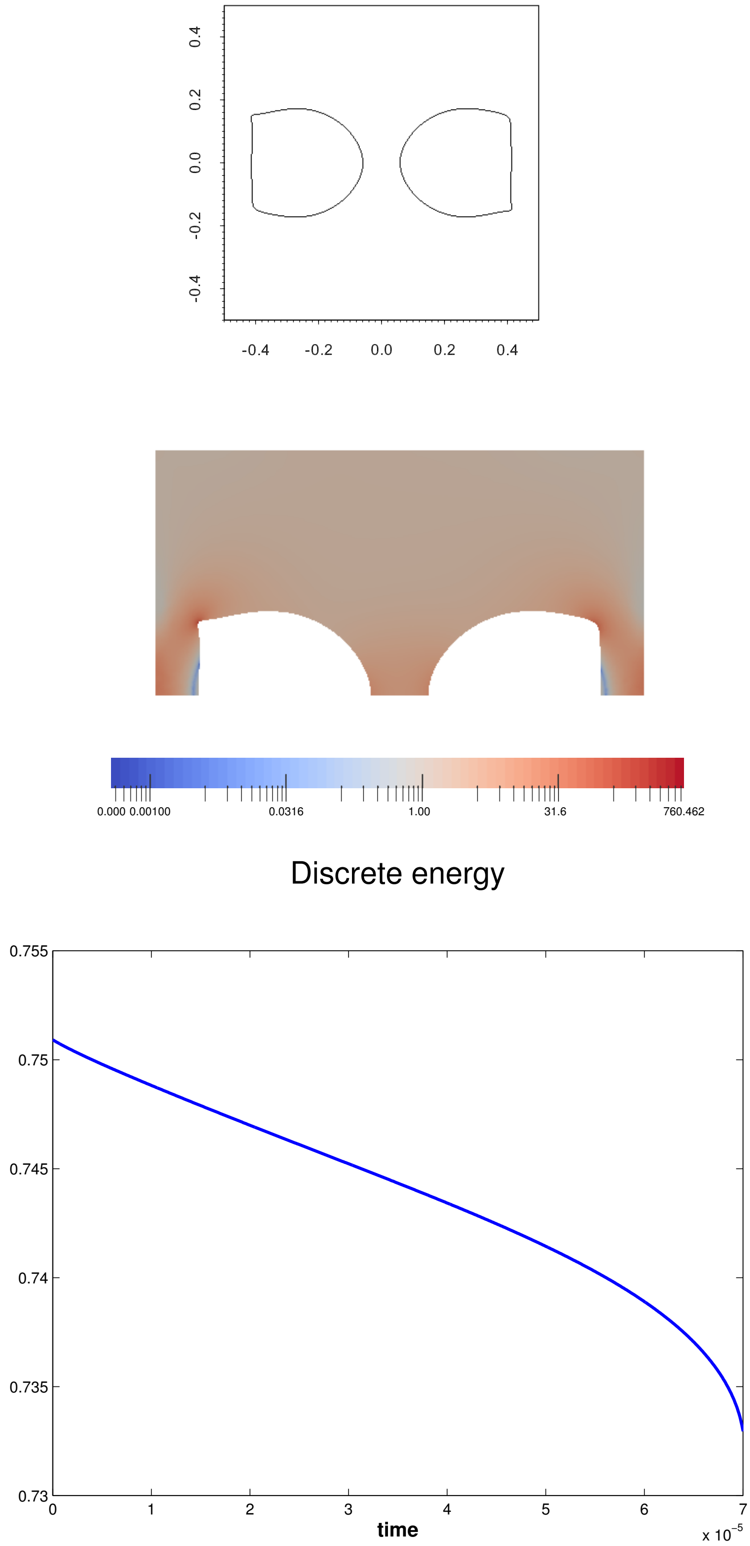

Figure 5.16: ($\alpha_2 = 0, \mu = 0.5, \lambda = 0, \underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right)$) Plots of the interface curve and the elastic energy density at $t = 7 \times 10^{-5}$. The energy density is colour coded with a logarithmic scale for the sake of visualisation. Below a plot of the total discrete energy $\alpha_1 \mathcal{H}^{d-1}(\Gamma^m) + \alpha_3 \left(\int_{\Omega_+^{m,h}} E(\vec{U}^{m+1})\, d\mathcal{L}^d - \int_{\partial\Omega} \vec{g} \cdot \vec{U}^{m+1}\, d\mathcal{H}^{d-1}\right)$ over time.

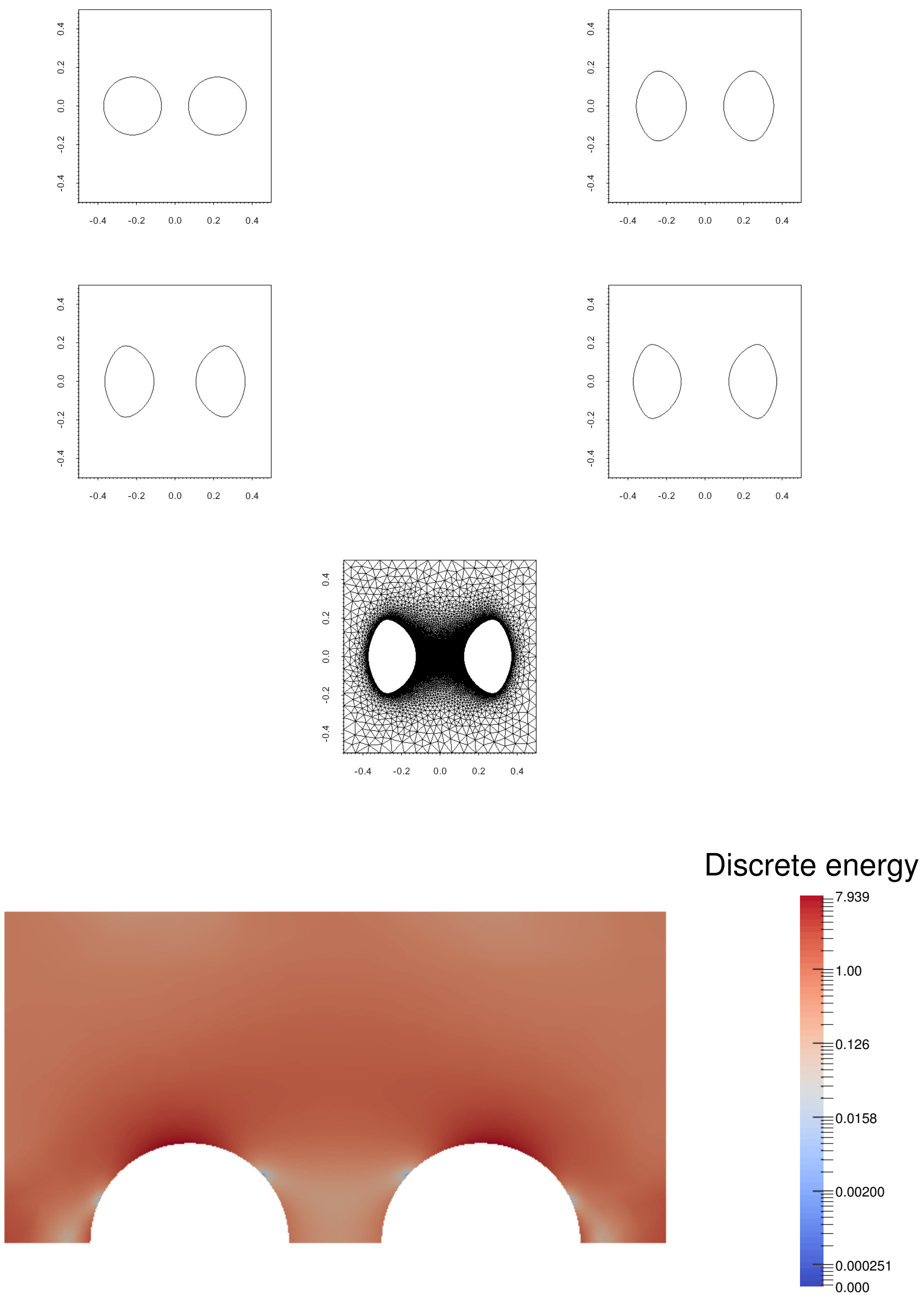

Figure 5.17: ($\mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$) Plots of the interface curve at times $t = 0$, $5 \times 10^{-4}, 10^{-3}$, and $T = 1.5 \times 10^{-3}$, bulk mesh and the elastic energy density at time $t = T$. The energy density is colour coded with a logarithmic scale for the sake of visualisation.

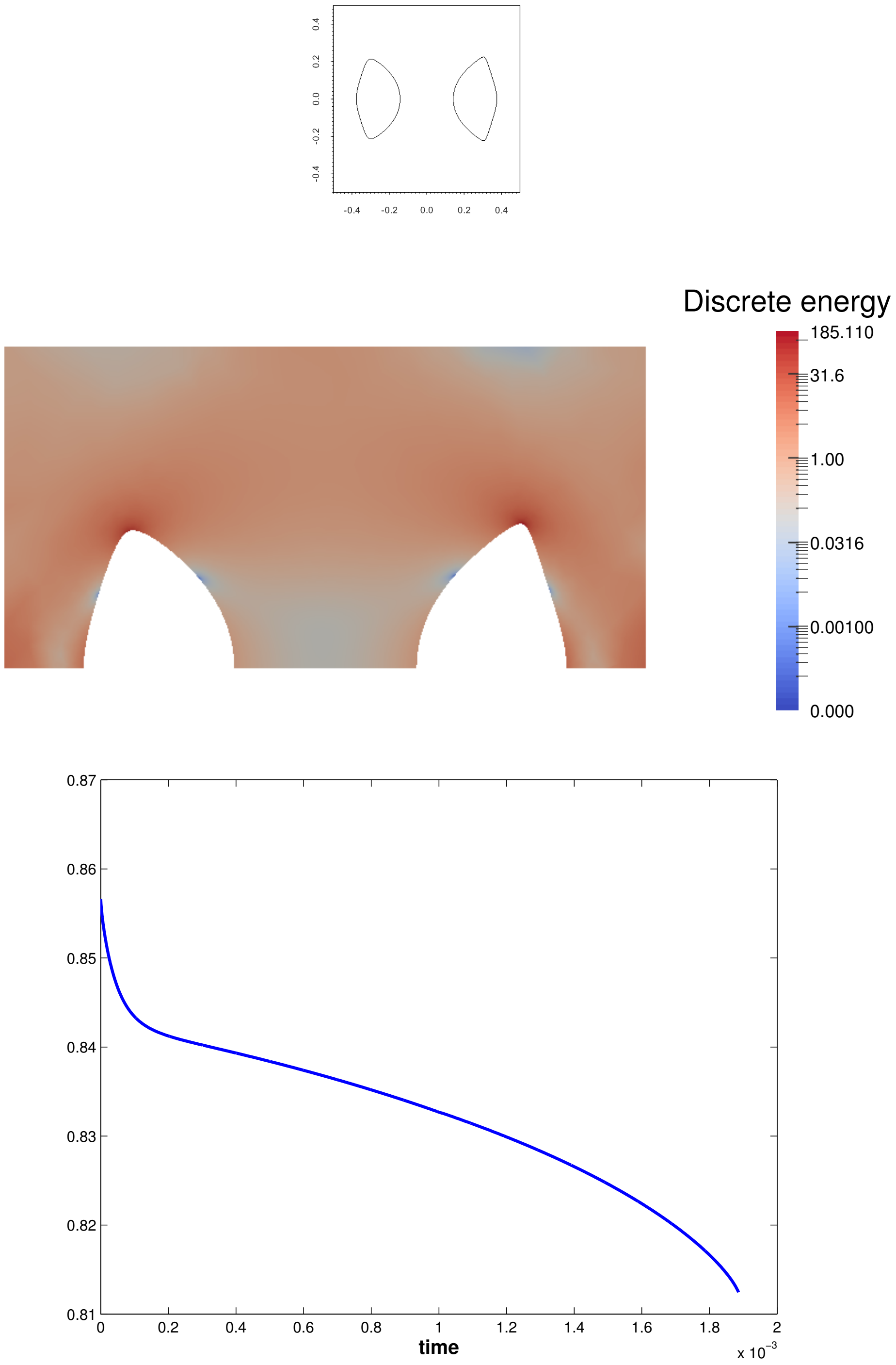

Figure 5.18: ($\mu = 0.5, \lambda = 0$, $\underline{\underline{S}} = \left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$) Plots of the interface curve and the elastic energy density at time $t = 1.886 \times 10^{-3}$. The energy density is colour coded with a logarithmic scale for the sake of visualisation. Below a plot of the total discrete energy $\alpha_1 \mathcal{H}^{d-1}(\Gamma^m) + \alpha_3 \left(\int_{\Omega_+^m} E(\vec{U}^{m+1}) \, d\mathcal{L}^d - \int_{\partial\Omega} \vec{g} \cdot \vec{U}^{m+1} \, d\mathcal{H}^{d-1}\right)$ over time.

unfitted simulations. The total CPU time was 812738s, with mesh smoothing and re-meshings accounting for 40% of that time. Here the re-meshing routine was applied ten times. The number of bulk degrees of freedom was between 221061 (minimum value) and 251737 (maximum value). Our results are plotted in Figure 5.19, and we notice a good agreement with both the *unfitted* approach results in Figure 4.22 and the plots in [3, Fig. 8]. We further investigate the drifting of the void, when a stronger electric loading is applied. To this end, we keep all the parameters unchanged bar α_2, which we now set to $75\,\pi^2$, and T, which we now set to 1.25×10^{-4}. The total CPU time was 293750s, with mesh smoothing and re-meshings accounting for 42% of that time. Here the re-meshing routine was applied twenty times. The number of bulk degrees of freedom was between 221061 (minimum value) and 250524 (maximum value). The evolution of the void is plotted in Figure 5.20. Compared to [3, Fig. 9], the evolution shown in our plots seems slightly slower, since at time $t = T$ the interface is not yet undergoing the topological change observed in [3, Fig. 9]. We note the good agreement with the *unfitted* approach plots shown in Figure 4.23.

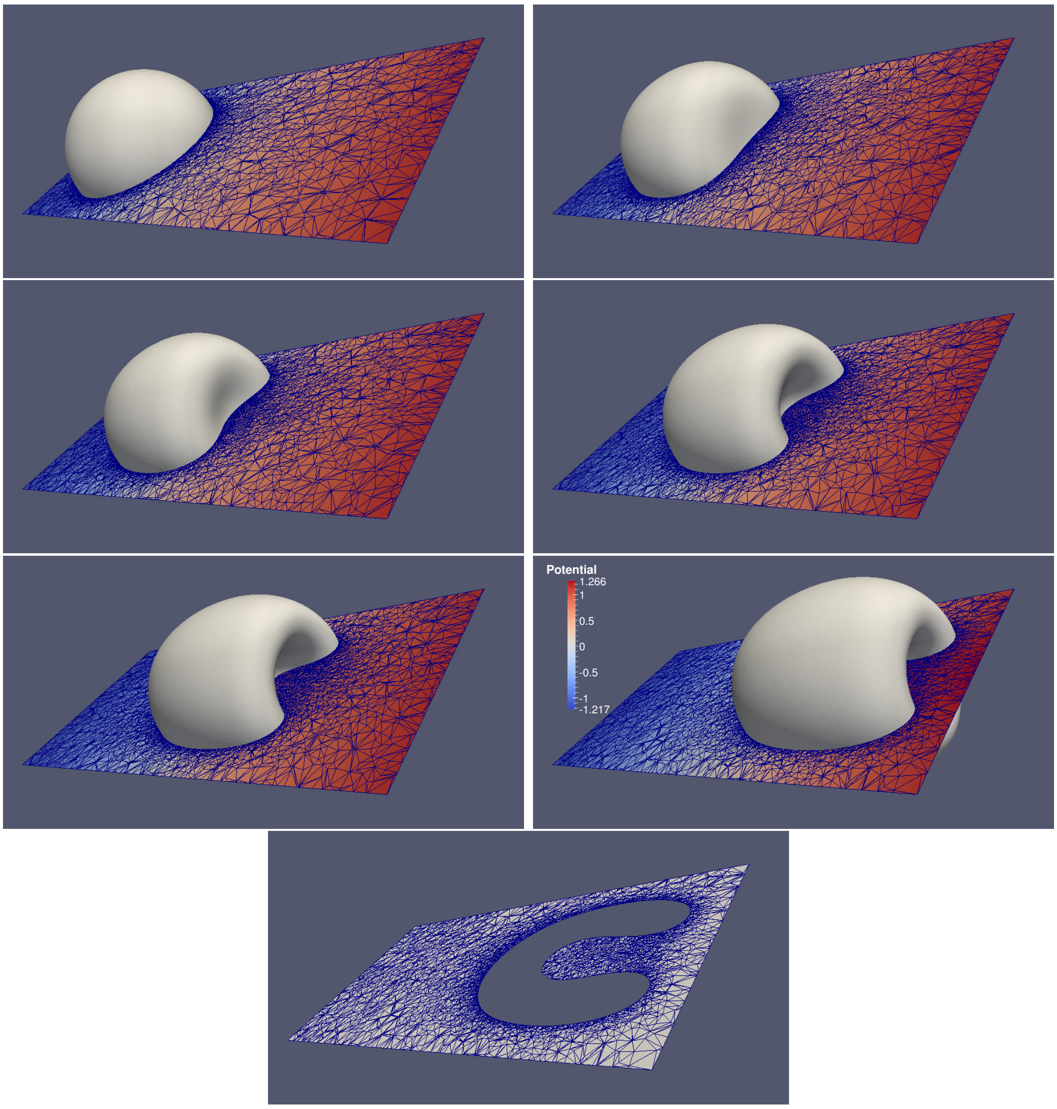

Figure 5.19: ($\alpha_2 = \frac{57}{2}\pi^2, \alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface mesh and cross section for $x_3 = 0$ of the bulk mesh at times $t = 0, 8 \times 10^{-5}$, 1.2×10^{-4}, 2×10^{-4}, 2.4×10^{-4} and $T = 3.6 \times 10^{-4}$. The electric potential is colour coded according to the legend shown at $T = 3.6 \times 10^{-4}$.

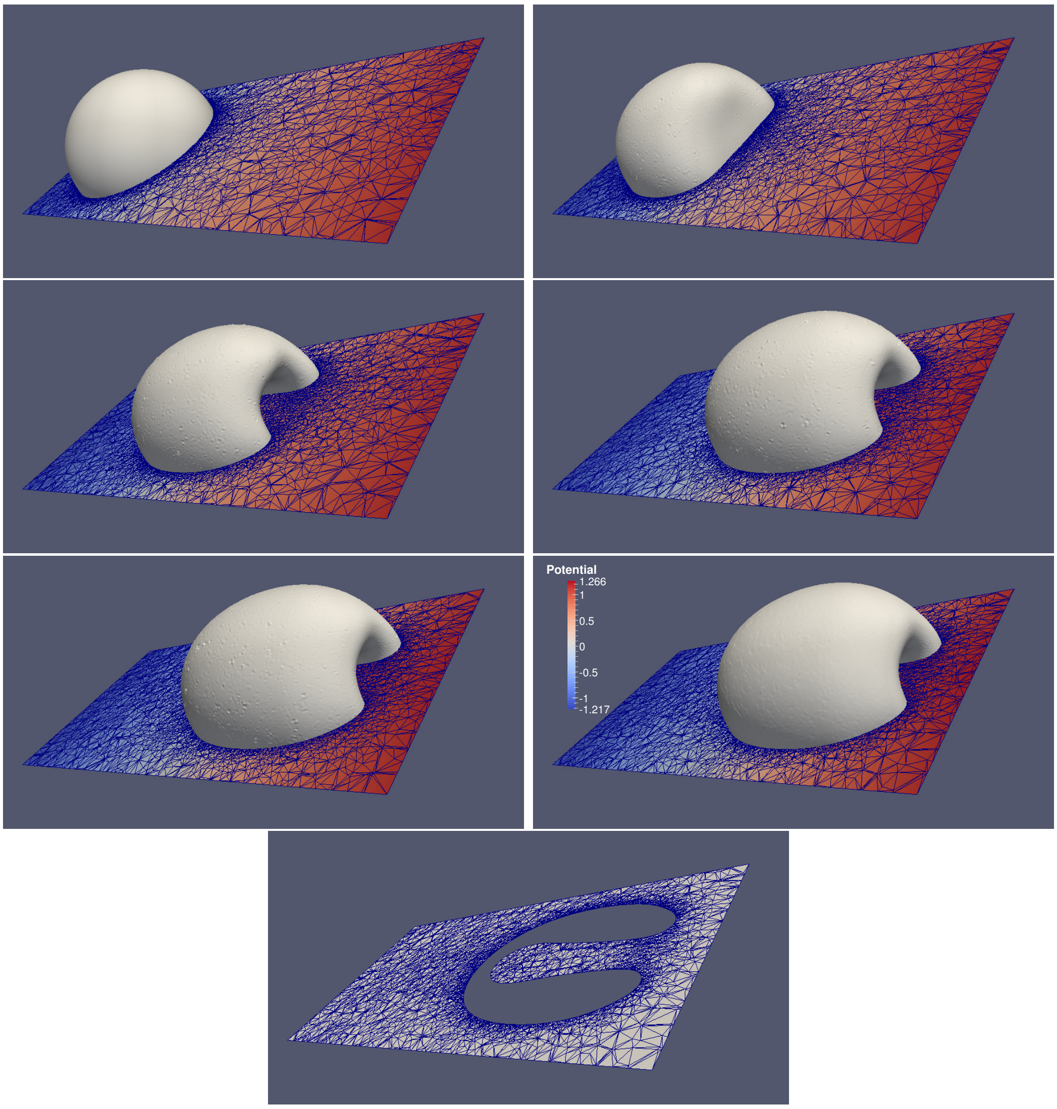

Figure 5.20: ($\alpha_2 = 75\,\pi^2, \alpha_3 = 0$, with Robin boundary conditions (4.23)) Plots of the interface mesh and cross section for $x_3 = 0$ of the bulk mesh at times $t = 0, 2.5 \times 10^{-5}$, 7.5×10^{-5}, 1.15×10^{-4}, 1.2×10^{-4} and $T = 1.25 \times 10^{-4}$. The electric potential is colour coded according to the legend shown at $T = 1.25 \times 10^{-4}$.

6

COMPARISON BETWEEN *UNFITTED* AND *FITTED* METHOD

In Chapters 4 and 5 we have presented and analysed two front-tracking, parametric finite element methods, namely the *unfitted* and *fitted* approach, for the numerical approximation of a void electro-stress migration problem. In this chapter we compare the two strategies, highlighting their common features, advantages and disadvantages.

The two approaches share some properties, which we briefly summarise as follows:

- the interface mesh exhibits good properties, and no redistribution of its vertices is necessary in practice. In particular, for the case $d = 2$, for a semidiscrete, continuous-in-time counterpart of the fully discrete finite element schemes, we are able to prove that the vertices of the discrete interface equidistribute and the volume enclosed by the surface is preserved exactly, recall Theorem 4.2 and Remark 5.2;

- although it does not appear possible to derive analogous results for the fully discrete *unfitted* and *fitted* schemes, in practice we observe that the vertices are always well distributed and the enclosed volume is preserved up to a very small tolerance;

- in the absence of external forces, the schemes are unconditionally stable. We note that in all our computations the methods are also stable in the presence of forces due to the applied electric field and to the elastic stresses.

The main advantage of the *unfitted* method is that bulk and interface meshes are totally independent, so there is no need to apply any routine in order to preserve a topological compatibility between the two grids. From the point of view of software implementation, this feature means that the two meshes can be stored and manipulated as instances of two independent classes. Moreover, standard refinement and coarsening strategies can be employed for the bulk mesh, and no mesh smoothings or re-meshings are necessary. The communication between the bulk mesh and the independent, lower-dimensional interface mesh needs to be implemented manually, since it is not part of standard finite element packages. In practice, as described in § 4.4.4, the user needs to perform cutting/labelling routines on the bulk elements and draw a mapping between cut bulk elements and interface vertices. These routines, which are rather laborious especially in the case $d = 3$, can be speeded up by using the hierarchical structure of the bulk grid, recall Algorithm 4 and Appendix A.2.

The main disadvantage of the *unfitted* method is that without significant effort in the implementation of an approximate exterior, the weak approximation of the natural boundary conditions in (3.3a) and (3.7b) introduces additional numerical errors. In particular, bulk quantities such as the electric potential and the elastic energy density are interpolated vertex-wise and not edge-wise. In our computational expertise, this turns out to be a less reliable strategy. In fact, we observe that the plots for the discrete energy are overall decreasing, but not monotonically. This is most likely due to the fact that the computational domain in the *unfitted* case, i.e. $\Omega_+^{m,h}$, is given by a union of bulk mesh elements, and therefore its boundary $\partial\Omega_+^{m,h}$ is very rough, with right-angled corners.

On the other hand, the main advantage of the *fitted* approach is precisely that the true exterior of the discrete interface is immediately available. Modern finite element packages provide the implementation of trace finite element spaces on lower-dimensional submeshes as standard, see e.g. [92]. In this thesis we have implemented the necessary trace finite element spaces ourselves, within the framework of the modular toolbox DUNE, see [22, 21] and the discretisation module `dune-fem`, see [48]. Having the exact exterior of the discrete interface and the exact traces of the bulk quantities on the discrete interface available means that the *fitted* method is more accurate in practice. In particular, we have shown that it exhibits an optimal experimental order of convergence for a test case with $d = 2$, where the exact solution is known, recall Table 5.1.

The main disadvantage of the *fitted* approach is that the movement of the interface implies that the bulk mesh cannot remain static. Rather, we need to apply *mesh smoothing* after every time step, in order to avoid the creation of very degenerate or even overlapping bulk elements. In our simulations, the mesh smoothing can account for up to 42% of the total CPU time in the case of electro-migration, while the higher computational demands for the stress-migration problem mean that the mesh smoothing accounts for only up to 16% of the total CPU times when both external forces are involved.

The aforementioned increased accuracy of the *fitted* approach is achieved at the cost of a higher CPU time. This is motivated by two reasons. Firstly, the mesh smoothing performed after each time step requires the resolution of the linear elasticity problem (5.6), which contributes to the total CPU time. Secondly, the Delaunay meshes generated with the help of the package GMSH and employed in Chapter 5 show a smoother transition in density of elements from regions with finer width to regions with coarser width, compared to the right-angled, isosceles-triangle meshes used in Chapter 4. As we can notice by corresponding experiments in § 4.5 and § 5.4, the usage of the package GMSH leads to a larger number of bulk degrees of freedom, which in turn increases the CPU time needed to solve the relevant systems of equations.

Regarding the generation and the manipulation of the bulk grid, we now add some remarks about both methods. In the *unfitted* case, we make use of the grid generator ALBERTA, which employs a very simple bisectioning routine for mesh refinement. In combination with an *ad hoc* choice of the initial macro-triangulation $\mathcal{T}^{-1}$, we are able to obtain, after every cycle of refinement, for both $d = 2$ and $d = 3$, regular meshes that are always composed of right-angled, isosceles triangles or of tetrahedra with such triangular faces. However, since ALBERTA can only halve a given element marked for refinement, in (4.20a) we need to choose the refinement parameter N_f to be a multiple of N_c by a power of 2, in order to obtain bulk elements close to the interface with the desired width h_f.

In the *fitted* case, instead, it is possible to calibrate the choice of the so-called *characteristic length* (see [73]), in order to mimic the adaptive strategy described in § 4.4.4. Further investigation would help understand how to choose those refinement parameters also for the *fitted* case. Moreover, we note that in the package GMSH more 3D mesh generation algorithms are available, in addition to the Delaunay strategy that we have always used in practice. A possible line of research would be to test these algorithms, namely *frontal*, *Delaunay-frontal* and *R-tree*, and compare them in terms of quality of the tetrahedral mesh and number of bulk degrees of freedom.

Finally, we note that we have used multi-component interfaces and implemented the necessary routines to link DUNE with the package El-Topo only for the *unfitted* approach. It is possible to generalise also the *fitted* approach to the case where the interface is composed of a family of polyhedral surfaces, and not just one surface. It is worth noting that in this framework, should the interface grid undergo a topological change, the bulk grid would immediately require a complete re-meshing, regardless of the actual quality of its elements, in order to preserve the topological compatibility with the newly-created interface grid.

A

APPENDIX 1

Partial differential equations (PDEs) are ubiquitous in science and engineering. In order to solve them, one can choose from a variety of numerical methods, including for instance finite element, finite volume, finite difference, or mesh-free methods. Given the size of modern technological problems, where millions of unknowns are to be computed, the choice of an efficient software implementation is crucial. Ideally the user would like to reuse legacy programmes or codes, without affecting performance and, if possible, being guaranteed generalisation to parallel computation. In this appendix we present an overview of the software DUNE (Distributed and Unified Numerics Environment), which has been used for all the numerical simulations presented in this thesis. For up-to-date information about new releases or modules, we refer to the official website `http://www.dune-project.org/`.

A.1 DESIGN PRINCIPLES OF DUNE

DUNE is a modular toolbox written in C++ for solving partial differential equations with grid-based strategies. Widely used methods like finite elements, finite volumes and finite differences can be easily implemented with the help of DUNE. We refer to [22] for a detailed presentation of the abstract framework of the software, while [21] describes the C++ design choices followed by its core developers. The main concept is given by the notion of "grid", which is always "hierarchical". More precise definitions and examples will be given in A.2. Let us recall from [22, § 2] the main design principles of DUNE:

- *Flexibility*: users should be able to write their own codes, which can be run on any grid satisfying the abstract DUNE grid interface.

- *Efficiency*: a clean and slim interface allows the user to optimise performance, with the possibility of parallelisation for both grid managers and linear solvers.
- *Legacy code* DUNE incorporates previously developed libraries, which can then be reused.

The main design idea for the DUNE grid interface is the *separation of data structures and algorithms by abstract interfaces*. This separation offers flexibility for codes, ensures maintainability and extendibility of the framework, and allows the reuse of existing finite element packages with a large body of functionality. Furthermore, generic programming techniques allow optimised implementations of methods or algorithms for a certain grid, while still offering a compatible interface. For example, one of the core modules of DUNE is `dune-istl`, dedicated to iterative algorithms for the solution of linear systems.

We are now ready to present the main features of the DUNE grid interface. In A.2 we first present the three key features of any DUNE grid, paying particular attention to the hierarchical structure and the refinement operations. These two features are crucial for the correct and efficient implementation of some of the algorithms discussed in this thesis. We refer to [22, § 2] for a more in-depth analysis of the usage of generic programming in DUNE. Moreover, we refer to [22, § 4] and to the aforementioned website for a non-exhaustive list of projects and applications developed with the help of DUNE.

A.2 GRID IMPLEMENTATION

Any DUNE grid is a "hierarchical" grid, and consists of three key concepts: an *entity complex*, a *geometric realisation*, and a *father relation*, see [22, Defs. 1-13] for a rather technical definition. We recall here the main ideas for the benefit of the reader.

An *entity complex* $\mathcal{E}_k$ is a collection of nested mesh objects with different dimensions. For instance, in the case $d = 3$ the entity complex will contain vertices (which are naturally objects of dimension 0), edges (dimension 1), faces (dimension 2) and elements (dimension 3). The definition of entity complex requires every element of codimension c ($0 < c \leqslant d$, where the codimension is the difference between d and the dimension of the object in question) to be contained in at least one object of codimension $c - 1$. It is straightforward to see that an entity complex contains all the necessary topological information of a grid on a specific refinement level. In fact, an entity complex expresses the connectivity of the mesh, without any information about how its elements were constructed, e.g. by refining a coarser grid or by coarsening a finer one.

A *geometric realisation* $\mathcal{M}$, instead, provides an entity complex with a specific geometric shape, according to a reference element. DUNE employs the well-known simplicial reference elements for $d = 1, 2, 3$.

Moreover, two entity complexes are in a *father relation* if one of them is obtained through refinement of the other. In other words, given any element e_r of the refined complex, it is always possible to identify one element e_c of the coarser complex which is called the *father* of e_r, because e_r is created from e_c through refinement.

A d-dimensional *hierarchical grid* is then a triple $(\mathcal{E}, \mathcal{M}, \mathcal{F})$, consisting of a finite set of d-dimensional complexes $\mathcal{E} = \{\mathcal{E}_0, \dots, \mathcal{E}_k\}$, a set of geometrical realisations $\mathcal{M} = \{m_0, \dots, m_k\}$, where each m_i is a geometric realisation of $\mathcal{E}_i$ into $\mathbb{R}^d$, and a set of father relations $\mathcal{F} = \{\mathcal{F}_0, \dots, \mathcal{F}_{k-1}\}$, such that $\mathcal{F}_i$ connects $\mathcal{E}_i$ with $\mathcal{E}_{i+1}$ for all $0 \leqslant i \leqslant k - 1$. The pair $(\mathcal{E}_i, m_i)$ is called a "level grid". It is possible to identify certain mesh objects, of different codimension and belonging to different grid levels, which have no children, since no additional refinement is performed on them. These objects are called "leaf" entities. With appropriate set operations on the *level entity complexes* and the *level geometric realisations*, see [22, Defs. 13-18] for details, it is possible to define a *leaf entity complex* and a *leaf geometric realisation*, which in combination form the so called "leaf grid". An example of hierarchical DUNE grid with $d = 2$ is given in [48, Fig. 1], where quadrangular elements are considered. We can observe that the complex $\mathcal{E}_0$ contains all the information about the initial macro-triangulation. In this particular example, the macro-element with label 1 is refined into four children, and then two of these new elements are refined into four smaller elements each. DUNE provides the user with special iterators that allow an efficient traversal of the grid. In fact, given any element at any grid level, it is easy to enumerate all its children obtained via successive cycles of refinement.

A.2.1 *Hierarchical structure and bulk/interface intersections*

In this section we analyse in more detail the connection between the hierarchical structure of the bulk grid and the implementation of certain coupling routines.

Recall from Section 4.4.6 that in the *unfitted* approach we need to identify all the bulk *leaf* elements that are cut by the discrete interface Γ^m. Algorithms 3 and 4 are coded making use of the hierarchical structure of the grid. More precisely, we first analyse the elements on level 0 of the grid. If a level-0 element is not cut, neither are all the *leaf* elements which are its children by successive cycles of refinement, performed according to the procedure described in § 4.4.4. The same argument can be applied recursively in order to speed up the coupling routines. In practice, we do not test for intersection with Γ^m any bulk element which has a father in the hierarchical tree that is not cut by any interface element. By so doing, we can reduce the overall computational cost of Algorithms 3 and 4 to be of order $\mathcal{O}(J_\Gamma^m \times \log|\mathcal{T}^m|)$.

A.2.2 *Mesh refinement*

Adaptive mesh refinement is a common operation performed within the framework of finite element methods. It aims at increasing the accuracy of discrete solutions and reducing the overall cost of numerical simulations. The DUNE grid interface provides methods for refining and coarsening the grid. Following [21, § 3.3], we briefly present how DUNE deals with mesh refinement. We also address the implementation details for the problem of our interest. Every grid element possesses a label which is to be set according to the desired operation. To this end, the user can employ a specific method provided by the grid abstract interface. The method `mark (ref, e)` is called to mark an entity `e` for refinement (`ref = 1`) or coarsening (`ref = -1`). Once all the entities of a grid have been marked (note that a label `0` means that no operation is to be performed on that entity), the adaptation is done in the following way:

1. Call the grid's method `preAdapt()`. This method sets up the grid for adaptation. It returns true if at least one entity was marked for coarsening, otherwise no adaptation can be performed.

2. If `preAdapt()` returned true, any data associated with entities that might be coarsened during the following adaptation cycle have to be projected onto the father entities. This represents a crucial feature especially for boundary elements, where the information on boundary conditions needs to be propagated appropriately.

3. Call `adapt()`. The grid is modified according to the refinement marks.

4. If `adapt()` returned true, new entities were created.

5. Call `postAdapt()` to clean up refinement labels.

Recall that in Algorithm 5 we described the procedure for filtering the grid $\mathcal{T}^m$ and identifying the collection $\mathcal{T}^m_+$, that constitutes the union of elements over which bulk quantities are to be computed. A sophisticated, C++-style coding of the aforementioned algorithm would require the manipulation of two `std::sets` containing the *currentFront* and the *newFront*. To this end, the user should define an appropriate order relation between grid elements, in order to save them in a `std::set`. However, this elegant implementation is not practical and causes a severe computational overhead. A faster, C-style implementation is possible, and makes use of the aforementioned labels attached to any grid element. The user just needs to define different dummy labels to distinguish between boundary, cut, *currentFront* and *newFront* elements. To filter the grid in practice, we make use of the class `Dune::Fem::FilteredGridPart`, which allows the selection of a subset of the *leaf* elements according to a specified boolean index, which we naturally choose to be the output returned by Algorithm 5 for every *leaf* element. Despite being less elegant, the latter implementation is far more practical and from our computational expertise proves to be, on average, 20 times faster than the C++-style implementation. This speed-up is particularly remarkable for three-dimensional simulations.

A.3 FINITE ELEMENT IMPLEMENTATION

We performed all the numerical simulations presented in this thesis with the help of the module `dune-fem`. We refer to the website `dune.mathematik.uni-freiburg.de` for up-to-date information about new releases or features. Moreover, we refer to [48] for all the software choices adopted in the implementation of all the quantities involved in finite element approximations. We report here some specific aspects for the benefit of the reader.

The `dune-fem` module is based on the `dune-grid` interface library, and aims at extending it by a number of discretisation algorithms for the resolution of linear and nonlinear systems of partial differential equations. The main notion is that of a "spatial discrete operator" L_h, which models a mapping between two finite element function spaces: $L_h : A_h \to B_h$. Examples of such spaces could be $\underline{V}(\Gamma^m)$ and $W(\Gamma^m)$, recall (2.10). In our case, we have employed this notion of operator to code the Laplace problem, the linear elasticity problem, and the interface problem. For the last one, according to which solution method is chosen (see Section 2.4), the set of equations for the interface can be coded as a block system (using the class `Dune::Fem::TupleDiscreteFunctionSpace` to define a combined space of curvature and position) or with four different operators, corresponding to the four submatrices in (2.58) and equivalent formulations. A discrete operator can be inverted using one of the several inverse operators available, which include, at present, preconditioned Krylov space methods (CG, BiCGSTAB, GMRes), Newton-type solvers, and sparse factorisation solvers. Discrete operators are constructed by choosing a continuous function space, a set of basis functions, and a view of the underlying grid, which determines that part of the grid on which the functions are defined. For instance, we employ the class `Dune::Fem::FilteredGridPart` for the resolution of the *unfitted* bulk equations. In addition, we make use of the implementations of both continuous and discontinuous Lagrange finite element spaces already available in `dune-fem`.

In collaboration with Marco Agnese, fellow PhD student in Applied Mathematics at Imperial College London, we have coded an inverse operator class for the numerical resolution of linear systems with sparse factorisation packages UMFPACK, LDL and SPQR. These solvers are available from the stable version of `dune-fem` 2.4.0, which has been released to be compatible with the stable version of DUNE 2.4, distributed in September 2015.

BIBLIOGRAPHY

[1] P. R. Amestoy, T. A. Davis, and I. S. Duff, *Algorithm 837: AMD, an approximate minimum degree ordering algorithm*, ACM Trans. Math. Softw., 30 (2004), pp. 381–388.

[2] A. Averbuch, M. Israeli, I. Ravve, and I. Yavneh, *Computation for electromigration in interconnects of microelectronic devices*, J. Comput. Phys., 167 (2001), pp. 316–371.

[3] L. Baňas and R. Nürnberg, *Phase field computations for surface diffusion and void electromigration in $\mathbb{R}^3$*, Comput. Visual. Sci., 12 (2009), pp. 319–327.

[4] E. Bänsch, F. Hausser, O. Lakkis, B. Li, and A. Voigt, *Finite element method for epitaxial growth with attachment–detachment kinetics*, J. Comput. Phys., 194 (2004), pp. 409–434.

[5] E. Bänsch, F. Hausser, and A. Voigt, *Finite element method for epitaxial growth with thermodynamic boundary conditions*, SIAM J. Sci. Comput., 26 (2005), pp. 2029–2046.

[6] E. Bänsch, P. Morin, and R. H. Nochetto, *A finite element method for surface diffusion: the parametric case*, J. Comput. Phys., 203 (2005), pp. 321–343.

[7] E. Bänsch and A. Schmidt, *Simulation of dendritic crystal growth with thermal convection*, Interfaces Free Bound., 2 (2000), pp. 95–115.

[8] J. W. Barrett and C. M. Elliott, *A finite-element method for solving elliptic equations with Neumann data on a curved boundary using unfitted meshes*, IMA J. Numer. Anal., 4 (1984), pp. 309–325.

[9] J. W. Barrett, H. Garcke, and R. Nürnberg, *Finite element approximation of a phase field model for surface diffusion of voids in a stressed solid*, Math. Comp., 75 (2006), pp. 7–41.

[10] ——, *A parametric finite element method for fourth order geometric evolution equations*, J. Comput. Phys., 222 (2007), pp. 441–467.

[11] ——, *A phase field model for the electromigration of intergranular voids*, Interfaces Free Bound., 9 (2007), pp. 171–210.

[12] ——, *On the parametric finite element approximation of evolving hypersurfaces in* $\mathbb{R}^3$, J. Comput. Phys., 227 (2008), pp. 4281–4307.

[13] ——, *Parametric approximation of Willmore flow and related geometric evolution equations*, SIAM J. Sci. Comput., 31 (2008), pp. 225–253.

[14] ——, *On stable parametric finite element methods for the Stefan problem and the Mullins–Sekerka problem with applications to dendritic growth*, J. Comput. Phys., 229 (2010), pp. 6270–6299.

[15] ——, *Parametric approximation of surface clusters driven by isotropic and anisotropic surface energies*, Interfaces Free Bound., 12 (2010), pp. 187–234.

[16] ——, *The approximation of planar curve evolutions by stable fully implicit finite element schemes that equidistribute*, Numer. Methods Partial Differential Equations, 27 (2011), pp. 1–30.

[17] ——, *Phase field models versus parametric front tracking methods: are they accurate and computationally efficient?*, Commun. Comput. Phys., 15 (2014), pp. 506–555.

[18] ——, *A stable parametric finite element discretization of two-phase Navier–Stokes flow*, J. Sci. Comp., (2014), pp. 1–40.

[19] ——, *Stable phase field approximations of anisotropic solidification*, IMA J. Numer. Anal., 34 (2014), pp. 1289–1327.

[20] J. W. Barrett, R. Nürnberg, and V. Styles, *Finite element approximation of a phase field model for void electromigration*, SIAM J. Numer. Anal., 42 (2004), pp. 738–772.

[21] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klöfkorn, R. Kornhuber, M. Ohlberger, and O. Sander, *A generic grid interface for parallel and adaptive scientific computing. Part II: Implementation and tests in DUNE*, Computing, 82 (2008), pp. 121–138.

[22] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klöfkorn, M. Ohlberger, and O. Sander, *A generic grid interface for parallel and adaptive scientific computing. Part I: Abstract framework*, Computing, 82 (2008), pp. 103–119.

[23] A. Belyaev and Y. Ohtake, *A comparison of mesh smoothing methods*, in Proceedings of the Israel-Korea Bi-National Conference on Geometric Modelling and Computer Graphics, 2003, pp. 83–87.

[24] H. Benninghoff and H. Garcke, *Efficient image segmentation and restoration using parametric curve evolution with junctions and topology changes*, SIAM J. Imaging Sci., 7 (2014), pp. 1451–1483.

[25] D. N. Bhate, A. Kumar, and A. F. Bower, *Diffuse interface model for electromigration and stress voiding*, J. Appl. Phys., 87 (2000), pp. 1712–1721.

[26] W. Boettinger, J. Warren, C. Beckermann, and A. Karma, *Phase-field simulation of solidification*, Annu. Rev. Mater. Res., 32 (2002), pp. 163–194.

[27] A. Bower and L. Freund, *Finite element analysis of electromigration and stress induced diffusion in deformable solids*, MRS Proceedings, 391 (1995), pp. 177–188.

[28] A. F. Bower and D. Craft, *Analysis of failure mechanisms in the interconnect lines of microelectronic circuits*, Fatigue Fract. Eng. M., 21 (1998), pp. 611–630.

[29] T. Brochu, C. Batty, and R. Bridson, *Matching fluid simulation elements to surface geometry and topology*, ACM Trans. Graph., 29 (2010), pp. 1–9.

[30] T. Brochu and R. Bridson, *Robust topological operations for dynamic explicit surfaces*, SIAM J. Sci. Comput., 31 (2009), pp. 2472–2493.

[31] T. Brochu, T. Keeler, and R. Bridson, *Linear-time smoke animation with vortex sheet meshes*, in Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA 2012, 2012, pp. 87–95.

[32] F. Cacho and X. Federspiel, *Modeling of electromigration phenomena*, in Electromigration in Thin Films and Electronic Devices: Materials and Reliability, Elsevier, 2011, pp. 3–44.

[33] J. W. Cahn, C. M. Elliott, and A. Novick-Cohen, *The Cahn-Hilliard equation with a concentration dependent mobility: motion by minus the Laplacian of the mean curvature*, European J. Appl. Math., 7 (1996), pp. 287–301.

[34] J. W. Cahn and J. E. Taylor, *Overview no. 113: surface motion by surface diffusion*, Acta Metall. Mater., 42 (1994), pp. 1045–1063.

[35] L. Chen, *Phase-field models for microstructure evolution*, Annu. Rev. Mater. Res., 32 (2002), pp. 113–140.

[36] L. Chen and M. Holst, *Efficient mesh optimization schemes based on optimal Delaunay triangulations*, Comput. Methods Appl. Mech. Engrg., 200 (2011), pp. 967–984.

[37] S. Chen, B. Merriman, M. Kang, R. Caflisch, C. Ratsch, L. Cheng, M. Gyure, R. Fedkiw, C. Anderson, and S. Osher, *A level set method for thin film epitaxial growth*, J. Comput. Phys., 167 (2001), pp. 475–500.

[38] X. Chen, K. C. Toh, and K. K. Phoon, *A modified sSOR preconditioner for sparse symmetric indefinite linear systems of equations*, Int. J. Numer. Meth. Eng., 65 (2006), pp. 785–807.

[39] F. Da, C. Batty, and E. Grinspun, *Multimaterial mesh-based surface tracking*, ACM Trans. Graph., 33 (2014), pp. 112:1–112:11.

[40] T. A. Davis, *Algorithm 832: UMFPACK v4.3 – an unsymmetric-pattern multifrontal method*, ACM Trans. Math. Softw., 30 (2004), pp. 196–199.

[41] ——, *Algorithm 849: A concise sparse Cholesky factorization package*, ACM Trans. Math. Softw., 31 (2005), pp. 587–591.

[42] ——, *Algorithm 915, SuiteSparseQR: Multifrontal multithreaded rank-revealing sparse QR factorization*, ACM Trans. Math. Softw., 38 (2011), p. 8.

[43] P. de Mottoni and M. Schatzman, *Geometrical evolution of developed interfaces*, Trans. Amer. Math. Soc., 347 (1995), pp. 1533–1589.

[44] K. Deckelnick and G. Dziuk, *Convergence of numerical schemes for the approximation of level set solutions to mean curvature flow*, in Numerical methods for viscosity solutions and applications (Heraklion, 1999), vol. 59 of Ser. Adv. Math. Appl. Sci., World Sci. Publ., River Edge, NJ, 2001, pp. 77–93.

[45] K. Deckelnick and G. Dziuk, *Numerical approximation of mean curvature flow of graphs and level sets*, in Mathematical aspects of evolving interfaces (Funchal, 2000), vol. 1812 of Lecture Notes in Math., Springer, Berlin, 2003, pp. 53–87.

[46] K. Deckelnick, G. Dziuk, and C. M. Elliott, *Computation of geometric partial differential equations and mean curvature flow*, Acta Numer., 14 (2005), pp. 139–232.

[47] K. Deckelnick, C. M. Elliott, and V. Styles, *Numerical diffusion-induced grain boundary motion*, Interfaces Free Bound., 3 (2001), pp. 393–414.

[48] A. Dedner, R. Klöfkorn, M. Nolte, and M. Ohlberger, *A generic interface for parallel and adaptive discretization schemes: abstraction principles and the Dune-Fem module*, Computing, 90 (2010), pp. 165–196.

[49] J. Dompierre, M. G. Vallet, P. Labbé, and F. Guibault, *An analysis of simplex shape measures for anisotropic meshes*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 4895–4914.

[50] G. Dziuk, *An algorithm for evolutionary surfaces*, Numer. Math., 58 (1991), pp. 603–611.

[51] ——, *Convergence of a semi-discrete scheme for the curve shortening flow*, Math. Mod. Meth. Appl. S., 4 (1994), pp. 589–606.

[52] ——, *Numerical schemes for the mean curvature flow of graphs*, in IUTAM Symposium on Variations of Domain and Free-Boundary Problems in Solid Mechanics, Springer, 1999, pp. 63–70.

[53] K. Ecker, *Regularity theory for mean curvature flow*, vol. 57, Springer, 2004.

[54] C. M. Elliott and H. Garcke, *Existence results for diffusive surface motion laws*, Adv. Math. Sci. Appl., 7 (1997), pp. 467–490.

[55] C. M. Elliott and S. Maier-Paape, *Losing a graph with surface diffusion*, Hokkaido Math. J., 30 (2001), pp. 297–305.

[56] C. M. Elliott and V. Styles, *Computations of bidirectional grain boundary dynamics in thin metallic films*, J. Comput. Phys., 187 (2003), pp. 524–543.

[57] ——, *An ALE ESFEM for solving PDEs on evolving surfaces*, Milan J. Math., 80 (2012), pp. 469–501.

[58] J. Escher, U. F. Mayer, and G. Simonett, *The surface diffusion flow for immersed hypersurfaces*, SIAM J. Math. Anal., 29 (1998), pp. 1419–1433.

[59] A. Eslami, F. Esfandiarpour, A. Shakourirad, and F. Farahmand, *A multiscale phase field method for joint segmentation-rigid registration — application to motion estimation of human knee joint*, Biomed. Eng.-App. Bas. C., 23 (2011), pp. 445–456.

[60] L. C. Evans, H. M. Soner, and P. E. Souganidis, *Phase transitions and generalized motion by mean curvature*, Comm. Pure Appl. Math., 45 (1992), pp. 1097–1123.

[61] L. C. Evans and J. Spruck, *Motion of level sets by mean curvature. I*, J. Differ. Geom., 33 (1991), pp. 635–681.

[62] ——, *Motion of level sets by mean curvature. II*, Trans. Amer. Math. Soc., 330 (1992), pp. 321–332.

[63] ——, *Motion of level sets by mean curvature. III*, J. Geom. Anal., 2 (1992), pp. 121–150.

[64] ——, *Motion of level sets by mean curvature. IV*, J. Geom. Anal., 5 (1995), pp. 77–114.

[65] R. Eymard, A. Handlovičová, and K. Mikula, *Study of a finite volume scheme for the regularized mean curvature flow level set equation*, IMA J. Numer. Anal., 31 (2011), pp. 813–846.

[66] X. Feng and A. Prohl, *Numerical analysis of the Allen-Cahn equation and approximation for mean curvature flows*, Numer. Math., 94 (2003), pp. 33–65.

[67] M. Fried, *A level set based finite element algorithm for the simulation of dendritic growth*, Comput. Vis. Sci., 7 (2004), pp. 97–110.

[68] M. Gage and R. S. Hamilton, *The heat equation shrinking convex plane curves*, J. Differ. Geom., 23 (1986), pp. 69–96.

[69] J. Gallier, *Schur complements and applications*, in Geometric Methods and Applications, vol. 38 of Texts in Applied Mathematics, Springer New York, 2011, pp. 431–437.

[70] S. Ganesan, *Finite element methods on moving meshes for free surface and interface flows*, PhD thesis, Otto-von-Guericke-Universität Magdeburg, 2006.

[71] H. Garcke and V. Styles, *Bi-directional diffusion induced grain boundary motion with triple junctions*, Interfaces Free Bound., 6 (2004), pp. 271–294.

[72] B. Geden, *Understand and avoid electromigration (EM) & IR-drop in custom IP blocks*, Synopsys White Paper, (2011), pp. 1–6.

[73] C. Geuzaine and J.-F. Remacle, *GMSH: A 3d finite element mesh generator with built-in pre-and post-processing facilities*, Int. J. Numer. Meth. Eng., 79 (2009), pp. 1309–1331.

[74] Y. Giga, *Surface evolution equations - A level set method*, Technical Report Series of Department of Mathematics, Hokkaido University, (2002).

[75] Y. Giga and K. Ito, *On pinching of curves moved by surface diffusion*, Commun. Appl. Anal., (1998), pp. 393–405.

[76] D. Gilbarg and N. S. Trudinger, *Elliptic partial differential equations of second order*, vol. 224, Springer, 2001.

[77] M. A. Grayson, *The heat equation shrinks embedded plane curves to round points*, J. Differ. Geom., 26 (1987), pp. 285–314.

[78] ——, *A short note on the evolution of a surface by its mean curvature*, Duke Math. J., 58 (1989), pp. 555–558.

[79] Y. Hamam and M. Couprie, *An optimisation-based approach to mesh smoothing: reformulation and extensions*, in Graph-Based Representations in Pattern Recognition, vol. 5534 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 31–41.

[80] C. S. Hau-Riege, *An introduction to Cu electromigration*, Microelectron. Reliab., 44 (2004), pp. 195–205.

[81] L. R. Herrmann, *Laplacian-isoparametric grid generation scheme*, J. Eng. Mech. Div.-ASCE, 102 (1976), pp. 749–907.

[82] P. S. Ho, *Motion of inclusion induced by a direct current and a temperature gradient*, J. Appl. Phys., 41 (1970), pp. 64–68.

[83] G. Huisken, *Flow by mean curvature of convex surfaces into spheres*, J. Differ. Geom., 20 (1984), pp. 237–266.

[84] D. Jacqmin, *Calculation of two-phase Navier-Stokes flows using phase-field modeling*, J. Comput. Phys., 155 (1999), pp. 96–127.

[85] D. Juric and G. Tryggvason, *A front-tracking method for dendritic solidification*, J. Comput. Phys., 123 (1996), pp. 127–148.

[86] A. Karma and M. Plapp, *Spiral surface growth without desorption*, Phys. Rev. Lett., 81 (1998), pp. 4444–4447.

[87] S. L. Karman, *Unstructured viscous layer insertion using linear-elastic smoothing*, AIAA J., 45 (2007), pp. 168–180.

[88] S. L. Karman, W. K. Anderson, and M. Sahasrabudhe, *Mesh generation using unstructured computational meshes and elliptic partial differential equation smoothing*, AIAA J., 44 (2006), pp. 1277–1286.

[89] T. Keeler and R. Bridson, *Ocean waves animation using boundary integral equations and explicit mesh tracking*, in ACM SIGGRAPH 2014 posters, 2014.

[90] B. M. Klingner and J. R. Shewchuk, *Aggressive tetrahedral mesh improvement*, in Proceedings of the 16th International Meshing Roundtable, Springer Berlin Heidelberg, 2008, pp. 3–23.

[91] P. Knupp, *Remarks on mesh quality*, tech. rep., Sandia National Laboratories, Albuquerque, NM, United States, 2007.

[92] D. Köster, O. Kriessl, and K. G. Siebert, *Design of finite element tools for coupled surface and volume meshes*, Numer. Math. Theory Methods Appl., 1 (2008), pp. 245–274.

[93] R. Kozubski, G. Murch, and P. Zieba, *A phase field model for grain growth and thermal grooving in thin films with orientation dependent surface energy*, Solid State Phenomena, 129 (2007), pp. 89–94.

[94] O. Kraft and E. Arzt, *Electromigration mechanisms in conductor lines: Void shape changes and slit-like failure*, Acta Mater., 45 (1997), pp. 1599–1611.

[95] Z. Li, H. Zhao, and H. Gao, *A numerical study of electro-migration voiding by evolving level set functions on a fixed cartesian grid*, J. Comput. Phys., 152 (1999), pp. 281–304.

[96] J. Lie, M. Lysaker, and X. Tai, *A binary level set model and some applications to mumford-shah image segmentation*, Image Processing, IEEE Transactions on, 15 (2006), pp. 1171–1181.

[97] J. Lienig and G. Jerke, *Electromigration-aware physical design of integrated circuits*, in Proceedings of the 18th International Conference on VLSI Design, IEEE Computer Society, 2005, pp. 77–82.

[98] C. Liu and J. Shen, *A phase field model for the mixture of two incompressible fluids and its approximation by a fourier-spectral method*, Phys. D, 179 (2003), pp. 211–228.

[99] S. H. Lo, *Optimization of tetrahedral meshes based on element shape measures*, Comput. Struct., 63 (1997), pp. 951–961.

[100] M. Mahadevan and R. M. Bradley, *Phase field model of surface electromigration in single crystal metal thin films*, Phys. D, 126 (1999), pp. 201–213.

[101] ——, *Simulations and theory of electromigration-induced slit formation in unpassivated single-crystal metal lines*, Phys. Rev. B, 59 (1999), pp. 11037–11046.

[102] J. Nečas and I. Hlaváček, *Mathematical theory of elastic and elasto-plastic bodies: an introduction*, Elsevier, Amsterdam, 1981.

[103] E. J. Nielsen and W. K. Anderson, *Recent improvements in aerodynamic design optimization on unstructured meshes*, AIAA J., 40 (2002), pp. 1155–1163.

[104] R. Nürnberg and A. Sacconi, *An unfitted finite element method for the numerical approximation of void electromigration*, J. Comput. Appl. Math., 270 (2014), pp. 531–544.

[105] ——, *A fitted finite element method for the numerical approximation of void electro-migration*, Appl. Numer. Math., (submitted).

[106] E. Olsson and G. Kreiss, *A conservative level set method for two phase flow*, J. Comput. Phys., 210 (2005), pp. 225–246.

[107] E. Olsson, G. Kreiss, and S. Zahedi, *A conservative level set method for two phase flow {II}*, J. Comput. Phys., 225 (2007), pp. 785–807.

[108] S. Osher and R. Fedkiw, *Level set methods and dynamic implicit surfaces*, vol. 153 of Applied Mathematical Sciences, Springer-Verlag, New York, 2003.

[109] V. N. Parthasarathy, C. M. Graichen, and A. F. Hathaway, *A comparison of tetrahedron quality measures*, Finite Elem. Anal. Des., 15 (1994), pp. 255–261.

[110] P. P. Pébay and T. J. Baker, *Analysis of triangle quality measures*, Math. Comput., 72 (2003), pp. 1817–1839.

[111] T. Preusser, M. Droske, C. Garbe, A. Telea, and M. Rumpf, *A phase field method for joint denoising, edge detection, and motion estimation in image sequence processing*, SIAM J Appl. Math., 68 (2008), pp. 599–618.

[112] A. Rätz and A. Voigt, *Various phase-field approximations for epitaxial growth*, J. Cryst. Growth, 266 (2004), pp. 278–282.

[113] J. Rubinstein, P. Sternberg, and J. B. Keller, *Fast reaction, slow diffusion, and curve shortening*, SIAM J. Appl. Math., 49 (1989), pp. 116–133.

[114] A. Schmidt and K. G. Siebert, *Design of Adaptive Finite Element software: The Finite Element Toolbox ALBERTA*, vol. 42 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin, 2005.

[115] J. A. Sethian, *Level set methods and fast marching methods*, vol. 3 of Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, second ed., 1999.

[116] D. Silvester and A. Wathen, *Fast iterative solution of stabilised stokes systems part ii: using general block preconditioners*, SIAM J. Numer. Anal., 31 (1994), pp. 1352–1367.

[117] S. L. Sun and J. F. Liu, *An efficient optimization procedure for tetrahedral meshes by chaos search algorithm*, J. Comput. Sci. Tech., 18 (2003), pp. 796–803.

[118] X. C. Tai, O. Christiansen, P. Lin, and I. Skjælaaen, *Image segmentation using some piecewise constant level set methods with MBO type of projection*, Int. J. Comput. Vision, 73 (2007), pp. 61–76.

[119] C. M. Tan and F. He, *Electromigration modeling at circuit layout level*, SpringerBriefs in Reliability, Springer, 2013.

[120] G. Taubin, *A signal processing approach to fair surface design*, in Proceedings of the 22nd annual conference on Computer Graphics and Interactive Techniques, ACM, 1995, pp. 351–358.

[121] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y. Jan, *A front-tracking method for the computations of multiphase flow*, J. Comput. Phys., 169 (2001), pp. 708–759.

[122] J. Vollmer, R. Mencl, and H. Müller, *Improved Laplacian smoothing of noisy surface meshes*, Comput. Graph. Forum, 18 (1999), pp. 131–138.

[123] L. Xia, A. Bower, Z. Suo, and C. Shih, *A finite element analysis of the motion and evolution of voids due to strain and electromigration induced surface diffusion*, J. Mech. Phys. Solids, 45 (1997), pp. 1473–1493.

[124] X. Zhang, J. Chen, and S. Osher, *A multiple level set method for modeling grain boundary evolution of polycrystalline materials*, Inter. Mult. Mech., 1 (2008), pp. 191–209.