DISSERTATION
submitted
to the
Combined Faculties for the Natural Sciences and for Mathematics
of the
Ruperto-Carola University of Heidelberg, Germany
for the degree of
Doctor of Natural Sciences

Put forward by
Diplom-Physiker Felix Heimann
Born in Mannheim
Oral examination: 15.07.2013

An Unfitted Higher-Order Discontinuous Galerkin Method for Incompressible Two-Phase Flow with Moving Contact Lines

Advisor: Prof. Dr. Peter Bastian

Acknowledgments

I would like to gratefully acknowledge the financial support which enabled the research presented in this thesis. Funding was received from the project *Zweiphasenströmung in komplex berandeten Gebieten* (BA 1498/8-1) of the *Deutsche Forschungsgemeinschaft* (DFG) as well as from the interdisciplinary DFG research unit *Dynamic Capillary Fringes* (DyCap).
The efforts of Andreas Nüßing and Christian Engwer, who implemented a highly efficient realization of a Marching Cubes 33 algorithm with volume partitions, also constitute an important contribution to the research presented in this thesis. Generally, I extend my thanks to all my co-developers and anyone who committed ideas and code to any of the modules of the *Distributed and Unified Numerics Environment* DUNE.

I thank my colleagues, from the present and the past, for contributing, all in their way, to a unique and kaleidoscopic life experience in academia.

Furthermore, I would like to thank my main supervisor Peter Bastian and my mentor Olaf Ippisch for a thorough education in numerical mathematics and for the free-spirited atmosphere in which I was allowed to conduct my research.

Most of all, I would like to express my gratitude to Valeria, for her support and for being the most worthwhile distraction.

Abstract

The author presents an unfitted discontinuous Galerkin method for incompressible two-phase flow applicable to dynamic regimes with significant surface tension. The method is suitable for simulations in complex geometries and a recursive algorithm is proposed, which allows the generation of piecewise linear sub-triangulations resolving both the domain boundaries and the interface between the two immiscible phases. Hence, discontinuous finite element spaces can be employed to capture the irregularities in the solution along the interface, i.e. the jump in the pressure field and in the velocity derivatives.

While the sub-triangulation is based on a linear Cartesian cut-cell approach, its resolution is decoupled from the resolution of the finite element mesh thus enabling the application of higher-order finite element spaces. The time development of the two subdomains is realized by level set methods and an unfitted discretization for the solution of the corresponding equations is described. Multiple approaches for the numerical treatment of surface tension in the context of unfitted discretizations are discussed and compared. Furthermore, these methods are extended to allow simulations with contact lines taking into account the occurrence of microscopic deformations of the contact angle. All proposed methods are verified by numerical test simulations in two and three dimensions.

Zusammenfassung

Diese Arbeit präsentiert eine neue Methode zur Berechnung von inkompressibler Zweiphasenströmung in komplex berandeten Gebieten unter Einfluss hoher Oberflächenspannung. Das Verfahren basiert auf einem rekursiven Algorithmus zur Konstruktion einer Sub-Triangulierung, welche sowohl die Gebietsgrenze als auch die Phasengrenze zwischen den nicht mischbaren Fluiden stückweise linear approximiert. Ausgehend von dieser Approximation, können Finite-Element-Räume definiert werden, in welchen sich die physikalischen Randbedingungen an der Phasengrenze unstetig darstellen lassen. Letztere erlauben einen Sprung sowohl im Druck als auch in den Ableitungen des Geschwindigkeitsfeldes.
Die verwendeten Finite-Element-Räume werden auf einem Gitter dargestellt, dessen Auflösung weitgehend unabhängig von der Auflösung der Sub-Triangulierung gewählt werden kann. Dieser Ansatz erlaubt es den geometrischen Fehler in der Darstellung des Gebietsrandes von der Anzahl der Freiheitsgrade des resultierenden algebraischen Problems zu entkoppeln und ermöglicht dadurch den nutzbringenden Einsatz von Ansatzräumen hoher polynomialer Ordnung.
Die zeitliche Entwicklung der beiden durch die Phasengrenze definierten Teilgebiete wird durch die Level-Set-Gleichung beschrieben. Da die Standardverfahren zur Lösung dieser Gleichung mit dem Ansatz dieser Arbeit nicht direkt kombinierbar sind, werden verschiedene Erweiterungen diskutiert und in numerischen Experimenten verglichen.
Ebenso werden Methoden zur Diskretisierung der Oberflächenspannung vorgestellt, welche insbesondere mit den algorithmischen Anforderungen einer effizienten Implementierung der Sub-Triangulierung kompatibel sind. Zusätzlich werden Erweiterungen diskutiert, die es erlauben bewegliche Kontaktlinien der Phasengrenze mit dem Gebietsrand in diese Methoden zu integrieren. Im Vordergrund steht hierbei die numerische Behandlung des mikroskopischen Kontaktwinkels, welcher in realen physikalischen Systemen von großer Bedeutung ist.
Die Leistungsfähigkeit des vorgestellten Verfahrens wird in zwei- und dreidimensionalen numerischen Experimenten demonstriert.

Contents

Symbols and Acronyms

Symbols

Symbol	Description	Page
d	The dimension of the computational domain Ω.	31
ε_p	A *regularizing* partition operator is required to construct cell-parts with a minimum edge length not below ε_p.	50
$\mathcal{E}_h$	In two dimensions (three dimensions) it denotes the set of all edges (faces) of the fundamental mesh $\mathcal{T}_h$. With the superscript I, $\mathcal{E}_h^I$ denotes only the interior, i.e. non-boundary edges (faces). Similarly, the superscript B restricts the set to the boundary edges.	31
$\mathcal{E}_\varkappa$	In two dimensions (three dimensions) it denotes the set of all edges (faces) of the geometry mesh $\mathcal{T}_\varkappa$. With the superscript I, $\mathcal{E}_\varkappa^I$ denotes only the interior, i.e. non-boundary edges (faces). Similarly, the superscript B restricts the set to the boundary edges.	31
$\mathfrak{e}, \mathfrak{f}$	Small variables in fractur script indicate cell-parts	32
h	Maximum edge length of a cell in the fundamental mesh $\mathcal{T}_h$.	32
h_e	A geometric scaling factor depending on the volumes of the adjacent cells of a face e.	64
$\varkappa$	Maximum edge length of a cell in the geometry mesh $\mathcal{T}_\varkappa$.	32
$\mathbf{H}_+^{m,k}$	Broken Sobolev space of order m which allows for discontinuities across the discrete two-phase interface.	59
$\mathcal{M}^+, \mathcal{M}^-$	Partition operators which provide a sub-triangulation of a single cell-part relative to the values of a level-set function in its corner nodes.	34
$\mathcal{M}_{16}^+, \mathcal{M}_{16}^-$	Partition operators corresponding to the extended Marching Cubes 16 algorithm.	44
$\mathcal{M}_{33}^+, \mathcal{M}_{33}^-$	Partition operators corresponding to the extended Marching Cubes 33 algorithm.	45
Ω_r	The rectangular domain in which all sub-domains Ω_i must be embedded and which is spanned by the fundamental mesh $\mathcal{T}_h$.	53
Ω	The open Lipschitz-domain in d-dimensional space. Within this thesis, it always indicates the classical *computational* domain for which a PDE, i.e. flow problems shall be solved. Together with a subscript it indicates a sub domain of the global domain Ω.	31
$\hat{\mathbf{\Omega}}$	An operator which maps a mesh to the open domain in $\mathbb{R}^d$ which is spanned by the given mesh.	31
$\mathbb{P}^k$	Polynomial space of order k.	59
$\mathbf{P}^k$	Cell-wise polynomial space of order k which allows for jumps across the cell faces.	59
$\mathbb{Q}^k$	Polynomial space of maximum order k.	59
$\mathbf{Q}^k$	Cell-wise polynomial space of maximum order k which allows for jumps across the cell faces.	59
$\mathcal{S}$	The sub-triangulation is a cut-cell mesh which consists only of cells with simple geometries including simplices, cubes and triangular prisms. The restriction of the fundamental mesh $\mathcal{T}_h$ with regard to $\mathcal{S}$ defines the cut-cell mesh $\mathcal{T}$. The sub-triangulation $\mathcal{S}$ is only employed for the numerical integration of the multi-linear forms resulting from the DG formulation. It does not constitute a classical refinement of $\mathcal{T}$ and the finite elements are never defined relative to the cell-parts in $\mathcal{S}$, i.e. the number of cell-parts in $\mathcal{S}$ is in general not correlated to the number of degrees of freedom.	35
$\mathcal{T}_h$	The fundamental mesh, a Cartesian mesh which must overlap the computational domain. The piecewise linear approximation of the (sub-) domain boundaries is eventually employed to restrict (cut) this mesh to the cut-cell mesh $\mathcal{T}$. It is formally described as a set of rectangular open neighborhoods with maximum edge length H.	31
$\mathcal{T}_\varkappa$	The geometry mesh, a Cartesian mesh resulting from a uniform refinement of the fundamental mesh. It is employed to construct the sub-triangulation $\mathcal{S}$ of the domain Ω. It is formally described as a set of rectangular open neighborhoods with maximum edge length h.	32

Acronyms

Acronym	Expansion
DG	Discontinuous Galerkin
QGM	Quasi geometric method (section 4.3 on page 92)
MC16	Standard marching cubes algorithm with 16 topological cases
MC33	Extended marching cubes algorithm with 33 topological cases
MDMP	Multi-domain multi-physics
NBP	Narrow band projection method (section 4.2.2 on page 89)
PDE	Partial differential equation(s)
SLS	Standard level set method
CLS	Conservative level set method

Introduction

The simulation of multi-phase flow is applied in various fields of science and engineering. Based on the encountered fluid mechanical regimes, very different numerical methods and model equations (e.g. turbulence models) may be required to obtain robust results with high predictive capability.

However, even when focusing on two-phase flow in laminar and low turbulence regimes, there remains a diversity of practical applications ranging from microfluidic system technologies to the analysis of flow properties in both natural and industrial porous media. Research in these fields is naturally focused on flow in complex domains with extents much larger then the fluid mechanical scales. The domain itself is thereby the object of interest itself.

Sophisticated finite difference discretizations have long since been able to produce both theoretically and experimentally verified results and handle arbitrary topological changes (e.g. coalescence of bubbles) with local mass conservation [1, 2, 3]. Although promising approaches to applications in complex domains were presented in [4, 5], these employed first order approximations at the boundary nodes while not supporting any adaptive mesh refinement.

Lattice Boltzmann methods are regularly shown to have a robust behavior for almost arbitrarily complicated domain geometries and scale very well in parallel application. Their ability to handle density jumps corresponding to realistic fluid-gas setups has been improved significantly over the recent years [6, 7, 8] though such jumps still entail problems with regard to mass conservation. In principle these methods lack an analogue of the *hp*-refinement technique for finite elements.

Due to their ability to adjust the degrees of freedom optimally to the local features of the solution via *hp*-refinement, the finite element methods have traditionally been the first choice for the solution of problems with highly localized force terms and complicated boundaries. However, the coupling of the interface movement with the Navier-Stokes equations is not straightforward. The various attempts to solve this issue may be roughly sorted into interface tracking and interface capturing methods.

Interface tracking methods which transport the mesh in each time step according to the flow velocity field are usually based on the arbitrary Lagrangian-Eulerian (ALE) technique and may be coupled with a Riemann solver at the interface [9, 10, 11, 12, 13, 14]. They align the mesh cells to the interface and thus allow its treatment as an internal boundary.

Interface capturing methods implicitly describe the moving interface by a scalar function. In this context, the level set method proposed in [15] has become extremely popular. When combined with a continuum surface force model, smearing out the sharp interface over a finite width, it may be efficiently applied using standard finite element methods for the solution of the instationary Navier-Stokes equations. Other authors proposed and analyzed techniques which explicitly reconstruct some approximation of the interface from a level set function without aligning the mesh cells themselves [16, 17]. However, they still enforced a smooth transition of the fluids' viscosity and density across the interface.

For a sharp interface, the surface tension and viscous forces will in general induce discontinuities of the velocity derivatives and the pressure. In order to represent such discontinuities, most discretization methods require a mesh with edges aligned to the sharp interface. Even if such a mesh can be constructed for the initial setup of a multi-phase flow simulation, deformation and topological changes of the fluid phases will eventually

necessitate a remeshing of the domains. For sufficiently difficult domains, this is a very challenging problem as common finite element discretizations entail multiple requirements on the mesh cells such as convexity and angle conditions.

The authors of [18, 19, 20] presented an unfitted DG (UDG) method for elliptic, parabolic and stationary Stokes problems in complex domains. There, the interior penalty discontinuous Galerkin method [21, 22, 23] was applied on a Cartesian cut-cell mesh, which results from cutting a complex domain out of a Cartesian structured mesh based on a level set function. As the resolution of the resulting discrete domain approximation was decoupled from the resolution of the finite element mesh, the accuracy of the former could be adapted to the polynomial order of the finite elements employed for a specific application. More detailed elaborations on the involved principles are given in chapter 1.

This thesis describes an extension of the UDG approach to incompressible two-phase flow with surface tension. While the original method allowed only a single domain boundary, the simulation of two-phase flow benefits from meshes which actually resolve the two-phase interface. In this case, discontinuous finite element spaces may be employed to capture the interface conditions including the jump of the pressure and of the velocity derivatives across the interface. However, the efficient construction of a consistent triangulation for such multi-domain setups entails various difficulties. Chapter 2 describes how the Marching Cubes Algorithm, which was employed in the original approach, can be utilized as part of a recursive algorithm allowing the construction of sub-triangulations for an, in principle, arbitrary number of level set functions and subdomains. This algorithm was implemented according to the generic unfitted software interface presented in [24].

The system equations as well as the corresponding time and space discretizations are presented in chapter 3. The time-dependency of the subdomains is resolved by an operator splitting which effectively decouples the Navier-Stokes equations from the level set equation. It will be shown how the high algorithmic flexibility in the construction of the sub-triangulations can be exploited to employ standard methods (i.e. method of lines) for both equations in spite of the underlying time dependent finite element spaces. In [25], a similar approach to discretization of two-phase model problems on time dependent meshes was analyzed for the one-dimensional case in combination with extended finite elements and compared quite favorably relative to space-time finite element methods.

Other finite element approaches which achieve an accurate representation of the discontinuous interface conditions include the extended finite element method [26, 27] and the discontinuous Galerkin method. The latter was applied in combination with a local remeshing in [12] and for an immersed boundary cut-cell method in [28]. However, both methods rely on more complicated schemes for the time discretization.

The highly local nature in which the cut-cell mesh is constructed entails difficulties for the discretization of surface tension forces. For a discrete representation of the interface that is only cell-wise smooth, curvature is not anymore a cell-local property as kinks (or even jumps) in the interface across the cell faces have to be accounted for. Essentially geometric methods for the discretization of surface tension, introduced in [9], are generally suited to the sharp interface reconstructions as implicitly provided by the sub-triangulation. While it was shown in [29] that such schemes can be adapted to the needs of the UDG approach, the required modifications entail additional demands on

the construction algorithm for the sub-triangulation. More suitable approaches for the discretization of surface tension are presented in chapter 4.

The ability of these discretization methods to handle contact lines, i.e. points where the interface meets the domain boundary, are discussed and appropriate extensions are proposed. Alas, the very principals of the microscopic contact line dynamics and the verification of possible macroscopic models with numerical simulations are still a subject of current research [30, 31, 32, 33]. Hence, the conceptual verification of the proposed discretizations was based on the assumption of a constant microscopic contact angle equal to zero. It is noteworthy that this actually comprises the practically relevant case of a perfectly wetting domain boundary.

All proposed models are verified by numerical simulations in two and three dimensions. The results are presented in chapter 5. Whenever feasible, the author tried to establish meaningful comparisons between concurring methods and estimate the sensitivity of the results to the non-physical discretization parameters. However, the reader should be aware that time and especially run-time set severe limits to such efforts.

Among the computed test setups, the two-dimensional benchmark presented in [34] deserves special attention due to its quantitative nature and the numerous participating research groups, e.g. [10, 16, 17].

This work is based on the discretization presented in [29] which constitutes the first application of the UDG method to two-phase flow. Furthermore, part of the research which was conducted for this thesis has already been published in [35]. The reader should be aware that parts of that publication are reproduced and cited within this thesis and explicit references are only given to point the reader to additional materials or alternative descriptions.

1 The Concept of Unfitted DG for Multi-Domain Problems

This chapter is intended to introduce the principle idea behind the original *unfitted discontinuous Galerkin method* (UDG) as well as its extension to multi-domain multi-physics problems. The terms and concepts introduced in this chapter will be used throughout the thesis.

1.1 It Does Not Fit

What is an *unfitted* method? In its most general interpretation, any numerical method in which the discrete mesh overlaps but does not fit the boundary of the computational domain, may be considered an unfitted method. While this definition encompasses a wide variety of numerical methods, some concepts are shared by all of them (even if the corresponding terminology does not, see section 1.2 for details):

The *fundamental mesh* $\mathcal{T}_h$ which overlaps the *computational domain* Ω is typically a structured mesh or a related structure with similar inherent regularity. In order to account for the discrepancy between the mesh and the domain, unfitted methods typically need to modify the local numerical approximation operator at (or even near) those cells which are intersected by the domain boundary. Independent of whether these modifications for a specific method are of algebraic or geometric nature, they always require a cell-wise approximation of at least some local geometric properties of the domain boundary. The requirements range from a few properties (like normal vectors and intersection points with the mesh edges) to a piecewise reconstruction of the domain boundary using low order approximations of the boundary, i.e. linear edges (planar faces in 3D) or splines. For the latter, it is useful to define the *cut-cell mesh* $\mathcal{T}$ which incorporates the geometric approximations of the domain boundary, although there are methods for which this mesh is never explicitly constructed and algebraic constraints are employed to enforce appropriate approximation at the boundary. Figure 1.1 shows the simplest approach, where the domain boundary is approximated by connecting the intersection points with linear edges.

It is obvious, that depending on the utilized modification at the domain boundary, unfitted methods may have quite different requirements with regard to how the domain boundary may be described. While some may benefit from smooth geometric CAD representations of the domain boundary, cell-wise linear approximations of the boundary are very common.

1.2 One, Two, Many Unfitted Methods

In the past, numerical methods incorporating Cartesian meshes (or simple simplex meshes) overlapping but not fitting the boundaries of the computational domain have been used in various ways to solve a multitude of different flow problems. The first publications in this direction include [36, 37, 38, 39] and were mainly concerned with potential and inviscid flows. Due to the generality of the underlying idea, the further development of these methods proceeded with little coherence with regard to methodology of even

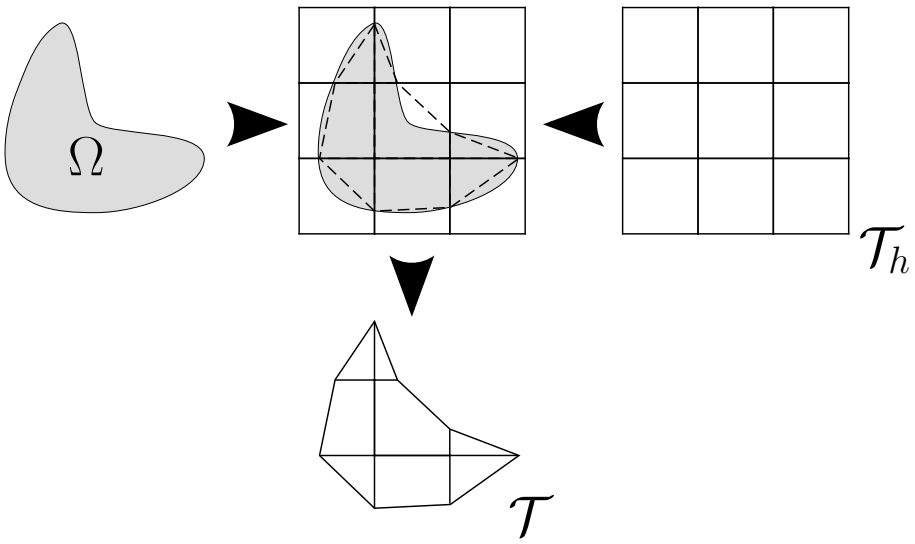


Figure 1.1: A shared concept of all *unfitted* methods: The computational domain Ω is overlapped by a simple mesh (the *fundamental mesh* $\mathcal{T}_h$. The edges of the latter are not in any way fitted to the domain boundary. An *unfitted* method may utilize a cell-wise geometric approximation of the domain boundary to modify its local algebraic approximation operators. In the simplest approach which is shown here - a typical cut cell method - the domain boundary is approximated with cell-wise linear edges implicitly defining the *cut-cell mesh* $\mathcal{T}$.

nomenclature. Labels like *immersed boundary method*, *fictitious domain method*, *adaptive Cartesian mesh method*, *unfitted Galerkin method* or *extended finite element method* correspond to intersecting sets of methods containing many hybrid specimen which allow only a vague categorization. The authors of [40, 41] give an overview of the different approaches without attempting to setup any real taxonomy.

In the following, we will exclusively use the term *unfitted method* for any numerical approach in which the discrete mesh merely overlaps the computational domain. If the discrepancy between the mesh and the domain is resolved by simply cutting away part of the mesh and thereby introducing (at least conceptually) a sub-triangulation of the discrete mesh, we will sometimes refer to it as a *cut-cell method*.

With few exceptions, unfitted methods are based on low order geometric approximations near the boundary. This is at least partly motivated by the demand for methods which can reliably handle geometries determined by a measurement (e.g. any kind of tomography) that provides a binary in/out field on a Cartesian mesh. Especially when complex topological structures are involved, meshing algorithms may fail or generate triangulations that require manual post-processing.

Furthermore, dealing with the low order approximations still holds challenges. Cut-cell meshes resulting from the naive geometric approach illustrated in figure 1.1 are strikingly unsuited for many of the most popular mesh-based numerical methods. Intersected cells

may become arbitrarily small compared to their neighbors and their inner angles may come arbitrarily close to zero. These properties may escalate the time step condition in explicit time stepping schemes or corrupt the convergence properties of Galerkin methods.

Unfitted methods which enforce boundary conditions strongly encounter additional problems known as *boundary locking*[42]. This term is used when the degrees of freedom of the numerical approximation are insufficient or chosen inappropriately to match the discrete boundary conditions. While various methods which solve this problem have been proposed, they usually rely on the specifics of the considered numerical scheme and appear to defy any attempts to their generalization as would be required in the context of multi-domain and multi-physics problems.

The standard finite element method as presented in countless textbooks comes with the capital ability to employ piecewise high-order polynomial function spaces such that the approximate solution achieves a correspondingly high-order of convergence with regard to the mesh refinement. However, it is a well known fact that poor (i.e. piecewise linear) approximations of the domain boundary may corrupt the order of convergence. This may be especially severe in flow problems with high Reynolds number where the resolution of the boundary can be crucial. Hence, it is not surprising that there are (to our knowledge) no publications on unfitted methods from the Galerkin family combining linear approximations of the domain boundary with high-order shape functions.

1.3 Unfitted Galerkin and the Idea of Weakly Enforced Boundary Conditions

Focusing on Galerkin methods, the generation of a discrete mesh fitting the computational domain may be seen as a preparation step for the representation of a basis of a discrete function space which may conveniently be restricted to the essential boundary conditions of a given problem. For a finite elements basis with its degrees of freedom given by values in some Lagrange interpolation points, this may be achieved by simply restricting the function space to those elements which have appropriate values in interpolation points on the domain boundary.

Hence, an *unfitted* Galerkin method has to solve the problem of either restricting the function space in some other way or finding the correct solution in the unrestricted space. The first ideas in this direction were formulated by Nitsche in [43]. In this approach, the space restrictions at the boundary were completely dropped, the trial space including functions with arbitrary values at the domain boundary. Instead, the boundary conditions were enforced via integral penalty terms on the domain boundary. The resulting discrete solutions meet the boundary conditions only in an integral sense, i.e. they are *weakly enforced* by the discretization. Some essential extensions of this idea to elliptic problems with interior interfaces or discontinuous coefficients may be found in [44, 45].

The domain decomposition methods which use essential boundary conditions to ensure continuity on the interior subdomain boundaries may be combined with discretizations that weakly enforce these conditions to give rise to the class of *interior penalty* discontinuous Galerkin (IPDG) methods initially developed for elliptic problems [46, 47, 48]. These methods employ penalty terms not only at the domain boundary but at all interior

faces (or edges in 2D) of the mesh, dropping all inter-element continuity restrictions to the trial and test spaces. This method was extended to various other applications including incompressible flow problems [22, 23, 49]. It was shown in [50] that at least for elliptic problems, the interior penalty method is just a variant of a much wider range of discontinuous Galerkin methods which differ only in the choice of some numerical fluxes.

The application of the IPDG method in the context of *unfitted* methods is quite straight-forward. As the penalty terms at the domain boundary are the same as the penalty terms at the interior edges (or faces in 3D) of the mesh, there is no need for any special treatment for essential boundary conditions and the resulting numerical method is equivalent to the application of the IPDG method to a mesh where those parts of each cell that overlap the domain are simply *cut away*.

However, due to the unfortunate properties of the mesh cells in the final cut-cell mesh, it is not a priori clear whether the established theoretical boundaries for the penalty parameter [51, 52] have any relevance in this case. This topic will be discussed in this thesis in the context of the given numerical examples.

All numerical experiments as presented in this work were done using the IPDG method. However, there seems to be no distinct reason to assume, that any of the methods (i.e. the diffusive numerical fluxes) as categorized in [50] should be distinguished with regard to their performance on cut-cell meshes. Whether there is a way to successfully apply the recently proposed hybridization methods [53, 54] for discontinuous Galerkin discretizations to hierarchical cut-cell meshes is a quite different question and can not be answered without further research.

1.4 Unfitted Galerkin and Adaptive Integration - Opening the Door to High-Order Approximations

The application of high-order discontinuous Galerkin methods to flow problems in domains with curved boundaries can be a futile waste of resources if these boundaries are approximated with coarse piecewise linear geometries. The authors of [56] gave a memorable example for the tremendous importance of the accurate approximation of curved boundaries for high-order methods in the case of a purely convective flow problem. While this may be considered a worst case scenario, it has been well established, that for many problems the boundary approximation can become a severe bottleneck in the application of high-order finite elements. This problem drives the development of meshes which can represent curved boundaries [57, 58, 59].

The construction of such meshes is difficult even if a smooth representation of the domain geometry is provided. The UDG method avoids this difficulty by decoupling the resolution of the domain geometry from the resolution of the finite element mesh. Therefore, a uniform refinement of the fundamental mesh, the *geometry mesh* $\mathcal{T}_\varkappa$, is utilized to construct a piecewise linear approximation of the smooth domain. The preliminary cut-cell mesh $\mathcal{T}^*$ is then defined as the restriction of the fundamental mesh with regard to this approximation of the domain. This concept is illustrated in figure 1.2. Obviously, the cells resulting from this restriction may have a very complicated form. In general, the remnant of a given cell of the fundamental mesh will be neither convex nor connected

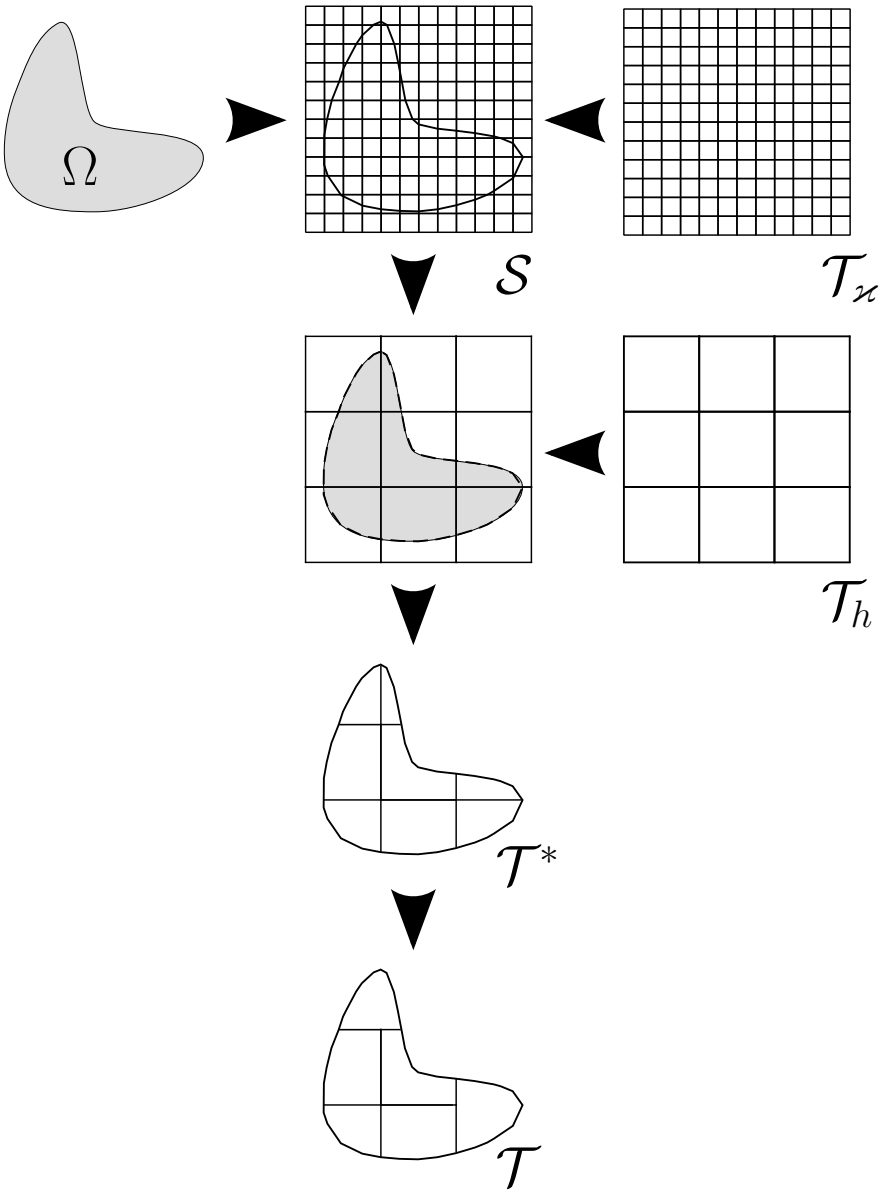


Figure 1.2: The UDG method, as introduced by Christian Engwer and Peter Bastian [55], employs an additional *geometry mesh* $\mathcal{T}_\varkappa$ which is constructed as an uniform refinement of the *fundamental mesh* $\mathcal{T}_h$. The geometry mesh is used to construct a piecewise linear approximation of the domain boundary and a corresponding cut-cell triangulation $\mathcal{S}$ which is only used for the numerical integration. The fundamental mesh is then restricted to the interior of this boundary approximation to define the *cut-cell mesh* $\mathcal{T}^*$. Small cells are then merged to their neighbors to define the final cut-cell mesh $\mathcal{T}^*$. Notice that the effective mesh resolution and therefore the dimension of the finite element basis is still determined by the resolution of the fundamental mesh.

with arbitrarily small volume and extent. Especially the latter property results in a number of issues when such a mesh is employed for numerical simulations. Most prominent among them are the stability issues arising in explicit time stepping schemes and bad convergence properties of iterative linear solvers applied to the linear systems resulting from implicit time stepping schemes and stationary problems.

To avoid these problems, small cells may be merged to their neighbors, thereby constructing the more benign cut-cell mesh $\mathcal{T}$. This merging procedure is also illustrated in figure 1.2 for a very simple domain setup. In general, it might be necessary to merge much more than two cut-cells in $\mathcal{T}^*$ into a single cell in $\mathcal{T}$ in order to construct a sufficiently good mesh for the numerical simulation. In extreme cases of narrow channels, such cells may span vast parts of the domain.

While it may be applied to solve the aforementioned issues, the merging procedure severely increases the algorithmic complexity of any implementation as it effectively disables many advantages of structured Cartesian meshes: After merging, cells in the cut-cell mesh $\mathcal{T}$ may have an arbitrary number of neighbors, the exact number depending on the domain boundary and the merging algorithm. Furthermore, the index of a cell can not be computed from its coordinates any more. Therefore, additional data-structures are required to associate mesh cells with their neighbor cells and with the degrees of freedom of their respective finite element basis.

Naturally, distinguishing a geometry mesh from the fundamental mesh is reasonable only if either an exact representation of the domain or a representation with a higher resolution than the fundamental mesh is available. With regard to the nature of this representation, the UDG approach is rather flexible as its construction of the piecewise linear approximation is based on a distance function. The value of this function at a given point is defined by the signed distance of this point to the domain boundary (*signed* indicates that the value would be positive inside of the domain and negative otherwise). As the zero level set of this function describes the domain boundary, it is also referred to as a *level set function*.

The provision of such a distance function may appear to be a demanding requirement as in practice domains are typically described either by an explicit boundary representation obtained from some CAD file or as a phase field resulting from, e.g. a tomographic measurement. However, based on an overlapping structured mesh, continuous approximations of distance functions for both cases can be computed efficiently by the *fast marching method* [60], a sophisticate algorithm for the solution of the Eikonal equation with a $\mathcal{O}(n \log n)$ complexity in the number of nodes in the geometry mesh.

The construction and management of the domain approximation on the geometry mesh $\mathcal{T}_\varkappa$ is of course the core engine of the UDG method. Although different ways to construct this approximation are conceivable, in this thesis we consider only piecewise linear approximations of the domain boundary. Even this simple approach, especially in three dimensions, involves many peculiarities as described in chapter 2. To avoid any confusion, it should be emphasized that the sub-triangulation $\mathcal{S}$ of the geometry grid $\mathcal{T}_\varkappa$, which consists of simple geometries (simplices, cubes, etc.) is used only for the numerical integration of the multi-linear forms resulting from the DG discretizations. Therefore, the

UDG scheme does in no way resemble an adaptive h-refinement scheme as the resolution of the final cut-cell mesh $\mathcal{T}$ is essentially given by the resolution of the fundamental mesh $\mathcal{T}_h$.

It is important to point out, that the relative resolution $\varkappa$ of the geometry mesh $\mathcal{T}_\varkappa$ can - in principle - be chosen independent of the resolution h of the fundamental mesh $\mathcal{T}_h$. As the numerical integration of the multi-linear forms describing the PDE problem has to be performed on the cells and cell remnants of the geometry mesh $\mathcal{T}_\varkappa$, it is obvious that the cost for the numerical quadrature in d dimensions will grow with $O(\varkappa^{-d})$ even if h is kept constant.

Therefore, in practice, the ratio $\frac{\varkappa}{h}$ will have a lower bound and no numerical examples with $\frac{\varkappa}{h} < \frac{1}{8}$ are presented in this thesis. Nevertheless, in three dimensions, such a value already implies that on a single given cell, the boundary approximation is constructed on a sub-grid with 512 cells (64 cells in two dimensions).

Especially for some hyperbolic problems (e.g. in [56]), where the reflection of waves on the boundary becomes the crucial bottleneck for the solution's accuracy, one might suspect that such a value for $\frac{\varkappa}{h}$ might still be too high to allow a beneficial application of a higher-order finite element basis.

On the other hand, for problems of a more diffusive nature, the situation might be very different. These were the setups for which the UDG method was originally developed and they were analyzed in [19, 20, 55]. Especially in laminar flow simulations with no-flow boundary conditions, decoupling of the boundary approximation from the finite element basis seems to be a very promising approach. Notice that even if, for a given problem, the UDG scheme applied in combination with a higher-order finite element basis does not, for $\varkappa \to 0$ result in a higher-order convergence rates, on coarse grids the approximation may yet be more accurate and/or more robust. This is a very favorable property with regard to numerical upscaling and the authors of [55] gave a memorable illustration of the approximation properties of the UDG methods on coarse grids.

1.5 Unfitted Galerkin in Multi-Domain Problems

A huge number of systems analyzed in industrial engineering are classified as multi-domain multi-physics (MDMP) problems. Generally speaking, this includes all problems, for which the computational domain consists of a number of adjacent or even overlapping subdomains, for each of which the system dynamics is described by individual sets of equations which are coupled at the boundaries or on some overlap of the respective domains.

The extension of unfitted methods to such MDMP problems is in general not trivial and yet there are many applications which may benefit from the flexibility of unfitted methods with regard to moving domains and topological changes. The principal difficulty arises from the fact, that algorithms for the construction of the cut-cell grid have to account for the way a (sub-)domain boundary intersects with a given cell. This defines a number of *topological cases* for a given cell type, each determined by the two disjoint sets of vertices of the cell which are *inside* and *outside* of the subdomain boundary. For a three-dimensional cube, the 2^8 topological cases can be reduced via symmetry considerations to merely

16 residual cases. However, if an unfitted method's cut-cell algorithm has to provide specialized behavior in each of these cases, e.g. to avoid boundary locking problems, then even such a number may pose a severe effort with regard to the implementation of the complete scheme.

Sadly, this dilemma is multiplied by the fact that the number of topological cases of a given cell grows exponentially with the number of interfaces intersecting that cell. Therefore, it is obvious that a robust unfitted method for MDMP problems can not involve any implementation details specific to the topological cases defined by the intersection of more than one interface.

The two big challenges of the extension of UDG to MDMP problems are the construction of the multi-domain sub-triangulation and the subsequent merging of the cut-cells. The details of the responsible algorithm and a discussion of all its intricacies will be given in chapter 2. At this point only the core ideas are presented:

The construction of the discrete domain approximation which is computed on the geometry mesh $\mathcal{T}_\varkappa$ is a completely local operation which, for a given cell in $\mathcal{T}_\varkappa$, depends only on the values of the level set function in the cell's vertices. Its result is a partition of that cell consisting of elementary geometries like simplicies, cubes or prisms. Hence, if the algorithm can be extended to work not only for cubes but for all of these elementary geometries, it may be applied recursively for an, in principle, arbitrary number of different level set functions.

For the definition of the multi-domain setup in the context of UDG methods, it is assumed that the subdomains can be described by a number of I level set functions, each describing an interface. One may consider the space spanned by the fundamental mesh $\mathcal{T}_h$ as a kind of *maximum domain*. Since every level set function partitions this maximum domain into an *inside* and *outside* region, I level set functions define at most 2^I subdomains.

The principle idea of the recursive application of the cut-cell algorithm for the construction of the multi-domain mesh is illustrated in figure 1.3. Its indiscriminate application results again in the intermediate cut cell grid $\mathcal{T}^*$ which may contain extremely small and anisotropic cells and requires some post-processing, i.e. merging of cells.

The merging itself is not as straightforward as in the case with only a single domain: The result must depend on the actual set of subdomains which constitute the support of the employed finite element spaces. This is illustrated for the case of two interfaces in figure 1.4. These examples show that optimally merged cut-cell grids for finite-element spaces on overlapping subdomains will in general be incompatible with each other, meaning that each of them may contain edges (or faces in three dimensions) not contained in the other. However, for two *overlapping* finite element spaces X and Y a typical MPMD problem will most likely require the computation and integration of a function product $f \cdot g$ with $f \in X$ and $g \in Y$. Hence, for applications that require such overlapping spaces, multiple cut-cell grids have to be constructed (i.e. managed) such that integrals of this kind can be computed efficiently. A detailed description of how these issues can be naturally resolved in the context of the UDG method is given in chapter 2.

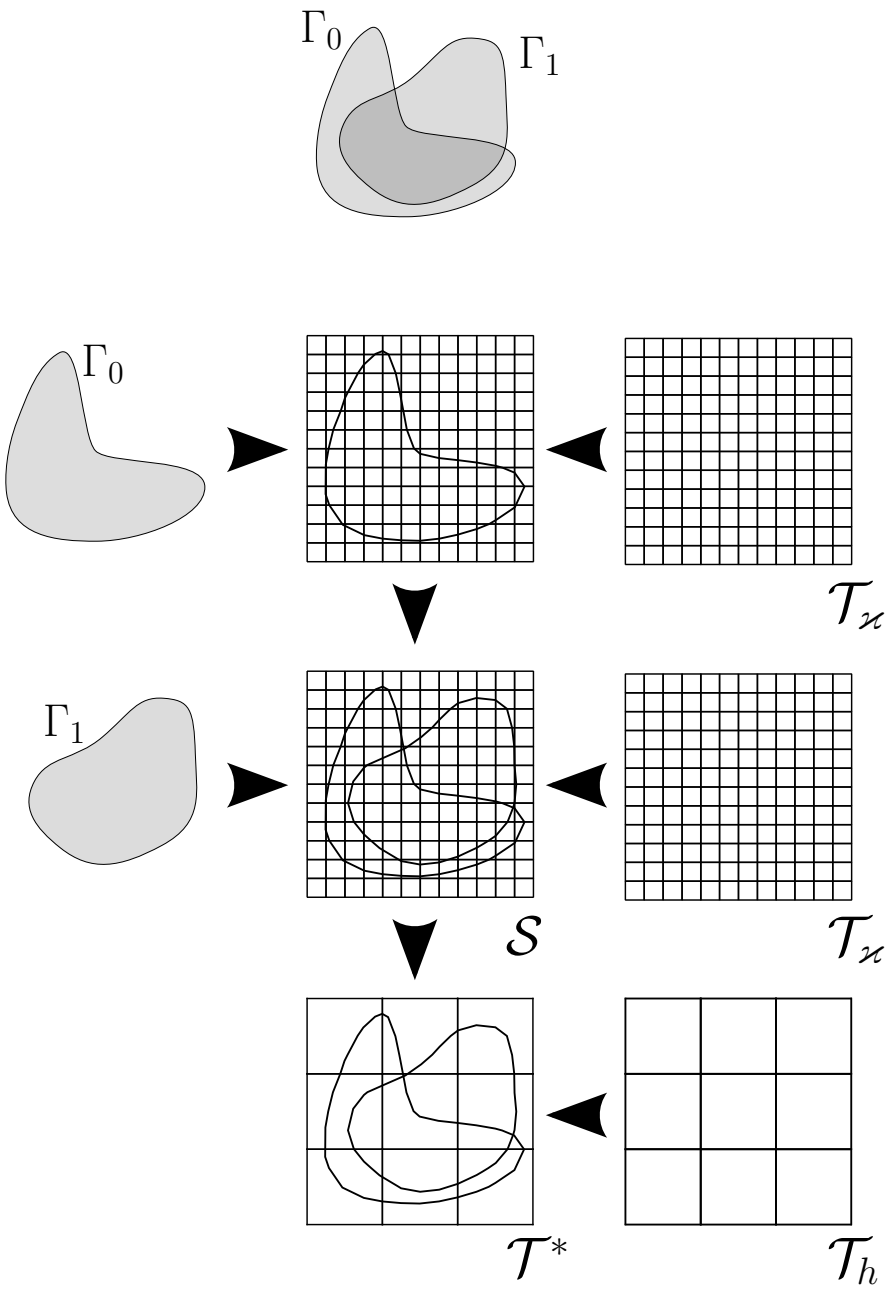


Figure 1.3: The UDG method can be applied to multi-domain problems. For two interfaces Γ_0 and Γ_1, the sub-triangulation procedure can be applied recursively to the geometry mesh $\mathcal{T}_\varkappa$, first for Γ_0 and then for Γ_1 to produce the intermediate cut cell mesh $\mathcal{T}^*$. If the boundary of the underlying fundamental mesh $\mathcal{T}_h$ is considered as an additional ultimate domain boundary, the two interfaces Γ_0 and Γ_1 do in principle allow to distinguish four domains (inside/outside Γ_0 and inside/outside Γ_1 and inside $\partial\mathcal{T}_h$). The final step of merging the cut-cells has to be done with regard to the sub-set of domains which support the chosen finite-element basis. The sub-triangulation $\mathcal{S}$ is used only for the purpose of numerical integration.

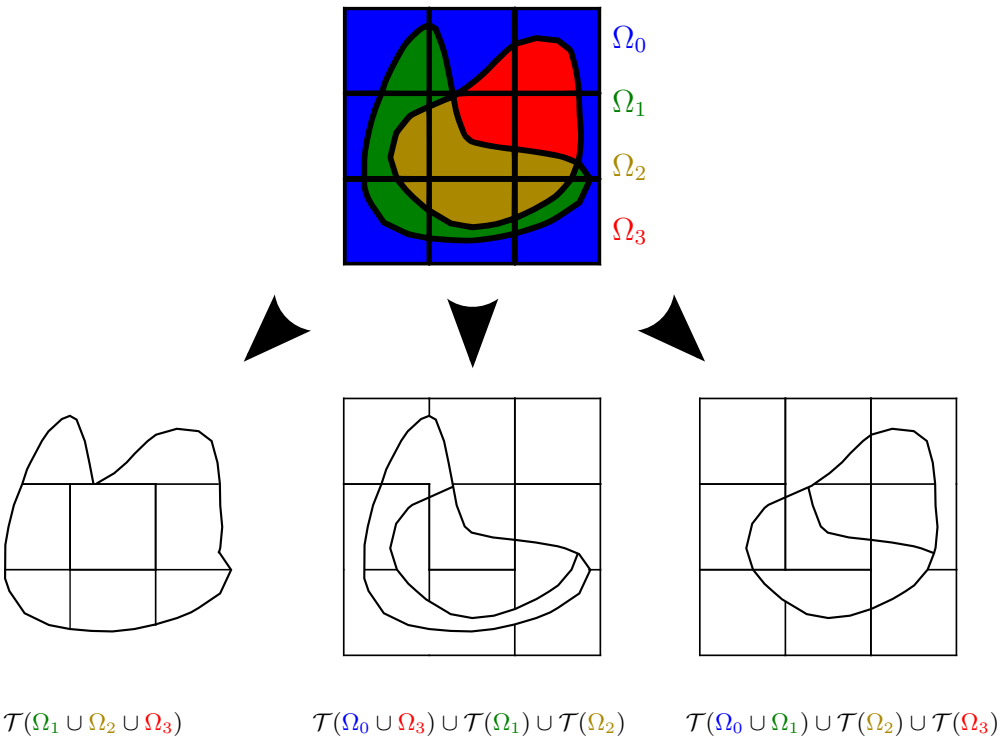


Figure 1.4: The illustration above depicts the four subdomains implicitly defined by two interfaces corresponding to two level set functions. The actual subdomains which constitute the support of a desired finite element space may however be a union of any or all of them. The merging procedure must account for this fact. The final cut-cell meshes as described above show realizations for different finite element spaces. In the first example (left hand side), we consider a - quasi single domain - problem involving a single finite element space defined on the cut-cell mesh $\mathcal{T}(\Omega_1 \cup \Omega_2 \cup \Omega_3)$ which approximates $\Omega_1 \cup \Omega_2 \cup \Omega_3$ and thus resolves only the subdomain boundary of Ω_0. In this case, all other subdomain boundaries were consumed during the merging process. Comparing this mesh to the second and third example reveals the variety of meshes that may result from the same two level set functions.

1.6 Limits of Unfitted Methods

Naturally, unfitted methods do not require a grid that is tailored to the boundaries of the computational domain. In many typical engineering problems where the domain is defined via a CAD geometry containing a vast number of details with specific properties, this advantage may be nullified if the representation of these details becomes too demanding with regard to the resolution of the fundamental grid. This is unfortunate as the construction of a suitable finite element mesh is especially complicated for those setups as well. However, this dilemma was recently diminished by the vast advances of the isogeometric analysis [61]. While these methods are in principle designed for surface problems, coupled approaches of isogeometric analysis with unfitted finite element

methods have been successfully attempted [62].

Furthermore, many solutions for CFD applications in the range of high Reynolds numbers are characterized by so-called boundary layers. These are thin layers along those parts of the boundary which are subjected to high viscous stress. The spatial grid resolution in these regions strongly influences the accuracy (and possibly the stability) of the simulation. Therefore, grid cells in the boundary layer are usually aligned to the boundary in thin sheets with high aspect ratios and their extents differ by orders of magnitude from the cells in the bulk area of the domain. As typical unfitted methods can hardly accommodate such requirements, they are rarely applied in such setups. However, in a recent publication [63], a specially adapted sub-grid suitable for the representation of boundary layers was incorporated into an unfitted Cartesian grid via extended finite elements. Such approaches that are based on a more modularized representation of the domain boundaries promise to extent the applicability of unfitted methods into new and unexpected fields.

1.7 Further Potential of Unfitted Methods

The problems involved in the construction of finite element meshes become most prominent when the process of mesh construction has to be automatized. This can become necessary if the mesh is moving or if the simulation is part of an automated process which involves the actual determination of the domain geometry as one of its sub-steps, e.g. in the evaluation of data obtained by a tomography scan of a large number of objects. In such cases, human post-processing of the finite element grid is not possible and heuristics are required in order to create robust simulation environments.

Furthermore, the class of applications which requires a transient representation of the domain geometry also includes inverse problems in the area of shape and topology optimization. Many shape optimization problems which occur in industrial engineering are focused on the determination of small grid deformations which can be captured by either arbitrary Lagrangian-Eulerian mappings of the domain or even direct manipulation of the grid vertices. However, topology optimization problems are completely incompatible with the conforming finite element paradigm of grids fitted to the computational domain [64]. The potential of higher-order unfitted methods in this area has been examined only recently [65].

2 On Hierarchic Cartesian Cut-Cell Triangulations for Multi-Domain Problems

The actual construction of a sub-triangulation for both the cells and the faces in the cut-cell grid is somewhat tailored to the needs of discontinuous Galerkin methods. For low order Cartesian cut-cell methods, knowledge of the geometries of all the cut-cell faces (including corner coordinates and normal vector) is usually sufficient for their definition and evaluation. However, in the context of high-order discontinuous Galerkin methods, the piecewise polynomial representation of the trial and test function spaces inevitably introduces non-trivial volume integrals into the weak problem formulation. The latter have to be approximated with quadrature rules that are sufficiently accurate to not compromise the overall approximation error of the numerical method.

In the approach presented in this thesis, this quadrature is realized by creating a partition consisting of simple geometric objects (cubes, triangular prisms, and simplices) of each grid cell of the geometry grid which is intersected by an interface. As Gaussian quadrature rules of arbitrary order are known for each of these geometric objects, the overall integration can be constructed by simply summing up the contributions of all quadrature points for all elements in the partition.

Hence, the main difficulty consists in constructing a partition for each cell in the geometry mesh $\mathcal{T}_\varkappa$ such that it allows the definition of the multi-domain cut-cell meshes according to the principle ideas presented in the previous section.

2.1 Principal Definitions

2.1.1 Domains

Throughout this thesis, it is assumed that the problems are formulated on a d-dimensional Lipschitz domain $\Omega \subset \mathbb{R}^d$ consisting of N_Ω subdomains Ω_i with $i \in \{1, \ldots, N_\Omega\}$ such that

$$\overline{\Omega} = \bigcup_{i \in \{1, \ldots, N_\Omega\}} \overline{\Omega}_i. \tag{2.1}$$

All of these domains are contained within a finite rectangular domain which is spanned by a Cartesian mesh, the fundamental mesh $\mathcal{T}_h$, with cells of maximal edge length h.

Wherever this mesh will be referred to in a formal context, it is assumed to be a set of open neighborhoods in $\mathbb{R}^d$ - the mesh cells. The set of all cell faces contained in the mesh $\mathcal{T}_h$ are denoted by $\mathcal{E}_h$ and are also understood as open neighborhoods. The sets $\mathcal{E}_h^I$ and $\mathcal{E}_h^B$ denote the interior faces and the boundary faces respectively. To allow a flexible notation, the operators $\hat{\boldsymbol{\Omega}}$ and $\overline{\boldsymbol{\Omega}}$ are introduced to describe a mapping of a mesh to the (open) domain spanned by its cells according to:

$$\overline{\boldsymbol{\Omega}}(\mathcal{T}_h) := \bigcup_{E \in \mathcal{T}_h} \overline{E} \quad \text{and} \quad \hat{\boldsymbol{\Omega}}(\mathcal{T}_h) := \overline{\boldsymbol{\Omega}}(\mathcal{T}_h) \setminus \partial\overline{\boldsymbol{\Omega}}(\mathcal{T}_h). \tag{2.2}$$

All definitions in this paragraph are correspondingly extended to all meshes and sub-triangulations described in this thesis.

Notice that in the following, the term *partition* will appear frequently and for an open domain $U \subset \mathbb{R}^d$ the set of open disjoint subdomains $\{u_1, \ldots, u_n\}$ with $u_i \subset \mathbb{R}^d$ and

$$\overline{U} = \bigcup_{1 \leq i \leq n} \overline{u_i} \tag{2.3}$$

will be called a partition of U (without qualifier).

2.1.2 Level Set Functions

The boundaries of all subdomains are described by the zero level sets of a number of N_ϕ level set functions ϕ_j with $j \in \{1, \ldots, N_\phi\}$. Each of the level set functions ϕ_j defines a partition $\{\Omega^+(\phi_j), \Omega^-(\phi_j)\}$ of the rectangular domain $\hat{\mathbf{\Omega}}(\mathcal{T}_h)$ according to

$$\Omega^+(\phi_j) = \{\mathbf{x} \in \hat{\mathbf{\Omega}}(\mathcal{T}_h) \,|\, \phi_j(\mathbf{x}) > 0\}, \tag{2.4}$$

$$\Omega^-(\phi_j) = \{\mathbf{x} \in \hat{\mathbf{\Omega}}(\mathcal{T}_h) \,|\, \phi_j(\mathbf{x}) < 0\}, \tag{2.5}$$

$$\overline{\mathbf{\Omega}}(\mathcal{T}_h) = \overline{\Omega}^+(\phi_j) \cup \overline{\Omega}^-(\phi_j). \tag{2.6}$$

For each subdomain Ω_i there must exist sub-sets P_i, $M_i \subset \{1, \ldots, N_\phi\}$ such that

$$\overline{\Omega}_i = \left(\bigcap_{j \in P_i} \overline{\Omega}^+(\phi_j) \right) \bigcap \left(\bigcap_{j \in M_i} \overline{\Omega}^-(\phi_j) \right). \tag{2.7}$$

This restriction implies that the maximum number of non-overlapping subdomains is 2^{N_ϕ}. Furthermore, for every point $\mathbf{x}$ located on the common boundary of two adjacent subdomains there must be a level set function ϕ_i for which $\phi_i(\mathbf{x}) = 0$. Obviously, for the definitions in this section to be meaningful, the level set function ϕ_i must be at least continuous on $\hat{\mathbf{\Omega}}(\mathcal{T}_h)$.

2.2 Sub-Triangulations

The main idea, as described in chapter 1 and illustrated in figure 1.3 on page 27, is to construct fine sub-triangulations $\mathcal{S}^i$ which approximate the subdomains Ω_i nicely but utilize a higher spatial resolution than the fundamental mesh. The resolution of the latter is chosen to allow an optimal application of the higher order DG basis. Then the cells of the fundamental mesh $\mathcal{T}_h$ are simply restricted with regard to the sub-triangulations to define a final cut-cell mesh $\mathcal{T}$. The sub-triangulation itself will thereafter be employed for purposes of numerical integration only.

The following discussion is restricted to the three-dimensional case. References to the two-dimensional case are made in some isolated contexts where these are instructive or where the generalization to the two-dimensional case might be ambiguous.

2.2.1 Properties and Requirements

These sub-triangulations $\mathcal{S}^i$ of the subdomains Ω_i are constructed based on the geometry mesh $\mathcal{T}_\varkappa$, a Cartesian mesh corresponding to a uniform refinement of the fundamental

mesh $\mathcal{T}_h$ with the correspondingly smaller resolution $\varkappa$. This implies that each vertex in $\mathcal{T}_h$ is also contained in $\mathcal{T}_\varkappa$ and that each cell in $\mathcal{T}_\varkappa$ is included in one and only one cell of $\mathcal{T}_h$.

Each such sub-triangulation is essentially a mesh of its own. For the implementation used in this thesis, this mesh, in the two-dimensional case, consists of triangles and quadrilaterals and, in the three-dimensional case, consists of simplices, hexahedra and triangular prisms. In the following, these objects will be referred to as *cell-parts* (always with hyphen) to not confuse them with the actual mesh cells of the cut-cell mesh $\mathcal{T}$. Accordingly, the sub-triangulations themselves will never be referred to as meshes.

At this point, some emphasis should be put on the fact that we require cell-parts to have straight edges. This property will simplify the construction of consistent sub-triangulation without gaps or overlapping cells. It is also the reason for some of the more peculiar definitions and terms introduced in this chapter. As will be revealed to the reader, the recursive approach for the generation of the multi-domain sub-triangulation entails enough complexity even if this geometric constraint is enforced. Notice that the faces of the cell-parts, however, may be curved as both hexahedra and triangular prisms contain quadrilateral faces for which the mapping of lowest polynomial order which maintains straight edges is yet bi-linear.

All cell-parts of the same geometric type, i.e. all simplices, all cubes, or all triangular prisms, can be associated with a reference geometry of the same type, which is assumed to have planar faces and a diameter of approximately 1, e.g. the unit cube, the unit simplex, and an appropriate cut-off of the unit cube along an arbitrary Cartesian axes. While the exact choice of the reference geometries is not important, in the following it is assumed that all cell-parts of the same geometry type are associated with the same reference geometry and for a given cell-part $\mathfrak{e}$ (cell-parts will always be denoted by letters in fraktur script) there is a bijection $\Phi_\mathfrak{e}$ mapping it to its respective reference element $\hat{\mathfrak{e}}$ such that

$$\Phi_\mathfrak{e}(\mathfrak{e}) = \hat{\mathfrak{e}}. \tag{2.8}$$

In the following, it is assumed that $\Phi_\mathfrak{e}$ is the mapping with the lowest possible polynomial order which maintains straight edges and maps the corners of $\mathfrak{e}$ to the corners of $\hat{\mathfrak{e}}$. This implies that for a given $\mathfrak{e}$, $\Phi_\mathfrak{e}$ is linear for simplices, and at most tri-linear for cubes and triangular prisms.

Notice that the mapping $\Phi_\mathfrak{e}$ maintains only the straightness of the edges of $\mathfrak{e}$ itself. Other straight lines, arbitrarily positioned in $\mathfrak{e}$, may be mapped into curves. For the definition of the recursive algorithm, which will be employed for the construction of the multi-domain sub-triangulation, it will be useful to consider the additional operator $\mathbf{\Phi}_\mathfrak{e}$ which maps a given cell-part $\tilde{\mathfrak{f}}$ with the corner nodes $\mathbf{x}_1, \ldots, \mathbf{x}_{N_\mathfrak{f}}$, all contained within $\hat{\mathfrak{e}}$ to the cell-part $\mathfrak{f}$ which corresponds to the corner nodes $\Phi_\mathfrak{e}(\mathbf{x}_1), \ldots, \Phi_\mathfrak{e}(\mathbf{x}_{N_\mathfrak{f}})$ (again using the polynomial mapping of lowest order which maintains straight edges). This implies that

$$\mathbf{\Phi}_\mathfrak{e}(\mathfrak{e}) = \Phi_\mathfrak{e}(\mathfrak{e}) = \hat{\mathfrak{e}} \qquad \text{but in general} \qquad \mathbf{\Phi}_\mathfrak{e}(\mathfrak{f}) \neq \Phi_\mathfrak{e}(\mathfrak{f}) \tag{2.9}$$

unless $\mathfrak{e}$ is a simplex or $\Phi_\mathfrak{e}$ is linear.

Before discussing the actual construction of the sub-triangulations $\mathcal{S}^i$, it is insightful

to consider the following list of *desirable* properties which these sub-triangulations should fulfill:

1. Naturally, it should be a good geometric approximation of Ω_i, hence the approximation error
$$\int_{\hat{\mathbf{\Omega}}(\mathcal{T}_h)} \|\chi_{\Omega_i} - \chi_{\hat{\mathbf{\Omega}}(\mathcal{S}^i)}\| \tag{2.10}$$
should fall quickly as $h \to 0$.

2. As a DG scheme involves integrals over cell faces, the sub-triangulation should be conforming wherever the faces of the cell-parts coincide with faces of the final cut-cell grid. Otherwise, the actual intersections of the faces of adjacent cell-parts would have to be determined and constructed in order to allow a consistent numerical integration. While this is feasible, it adds another layer of geometric operations.

3. The sub-triangulation of a given cell in the geometry mesh $\mathcal{T}_\varkappa$ should depend only on the restriction of the level set function to the closure of this cell. This property allows for a completely parallel computation of the sub-triangulation in the context of domain-decomposition methods. This provides the individual nodes with the footing required to efficiently communicate the non-local aspects of the cut-cell mesh construction, i.e. the merging of small cells in the intermediary cut-cell mesh.

2.2.2 Local Partition Operator

As soon as a construction algorithm fulfills the last of the aforementioned properties, it may be uniquely defined by its application to a single cell in the geometry mesh. To describe such an algorithm more generally, it is useful to define the *local partition operators* $\mathcal{M}^+$ and $\mathcal{M}^-$. For a given cell-part $\mathfrak{e}$ (i.e. a cube, simplex or triangular prism) and a level set function ϕ, the application of each operator is assumed to provide a finite approximation of the volume fraction of the cell-part $\mathfrak{e}$ on the positive ($\mathcal{M}^+$) or the negative ($\mathcal{M}^-$) side of the zero level set of the level set function. This approximations should themselves consist of cell-parts and be such that their union is a partition of the original cell-part $\mathfrak{e}$. The number of cell-parts created by these operators should be bounded by a finite positive number $N_\mathcal{M}$ independent of the given cell-part or the level set function values. This corresponds to
$$\mathcal{M}^{+/-}(\mathfrak{e}, \phi|_{\bar{\mathfrak{e}}}) = \{\mathfrak{f}_1, \ldots, \mathfrak{f}_N\} \tag{2.11}$$
for a set of cell-parts $\mathfrak{f}_i$ with $i = 1, \ldots, N$ and $N \leq N_\mathcal{M}$ with the additional property
$$\bigcup \left\{ \overline{\mathbf{\Phi}_\mathfrak{e}(\mathfrak{f})} \;\middle|\; \mathfrak{f} \in \mathcal{M}^+(\mathfrak{e}, \cdot) \cup \mathcal{M}^-(\mathfrak{e}, \cdot) \right\} = \overline{\Phi_\mathfrak{e}(\mathfrak{e})} = \bar{\hat{\mathfrak{e}}} \tag{2.12}$$
that holds independent of the level set function values. Furthermore, the cell-parts $\mathfrak{f}_i$ ($i = 1, \ldots, N$) are required to be *conforming* in the traditional sense, Hence, two such cell-parts will intersect either in a (full) common face, a (full) common edge, a vertex or not at all.

Notice that the definition so far does not imply that $\mathcal{M}^{+/-}$ provides a real partition of $\mathfrak{e}$, since

$$\overline{\mathcal{M}^{+}(\mathfrak{e},\cdot)\bigcup\mathcal{M}^{-}(\mathfrak{e},\cdot)} = \bar{\mathfrak{e}} \tag{2.13}$$

is true only if $\mathfrak{e}$ is a simplex or $\Phi_{\mathfrak{e}}$ is linear. However, in general (2.13) is **not fulfilled** as (2.12) only guarantees that the created cell-parts can be mapped via $\mathbf{\Phi}_{\mathfrak{e}}$ to a partition, i.e. a local sub-triangulation, of the reference element $\hat{\mathfrak{e}}$. As this local sub-triangulation is not required to be conforming, its image via $\mathbf{\Phi}_{\mathfrak{e}}^{-1}$ may contain gaps and overlaps of cell-parts. Full conformity, however, constitutes a severe restriction with regard to the actual implementation of the partition operators. More subtle additional requirements that allow a convenient implementation and yet result in consistent sub-triangulations will be discussed in section 2.3.

2.2.3 Recursive Construction of Local Sub-Triangulations

Notice that the nature of this local partition operator is inherently suited for its recursive application. Given a cell-part $\mathfrak{e}$ and its local restriction of the level set function $\phi|_{\bar{\mathfrak{e}}}$, it provides a new list of cell-parts for each of which the corners are located within the original cell-part $\mathfrak{e}$. Hence, given a cell E of the geometry mesh $\mathcal{T}_{\varkappa}$ and the cell-local restrictions of the level set functions $\phi_j|_E\,(j=1,\ldots,N_\phi)$, the partition operators can be applied recursively to this cell, i.e. to the cell-parts resulting from each application. Notice that in the first step of this application we implicitly interpret the cell E as a cell-part.

To formalize this approach it is necessary to introduce the special partition operator $\mathcal{M}_i^j$ which equals $\mathcal{M}^+$ if the subdomain Ω_i is located on the positive side of the zero level set of ϕ_j, i.e. $j\in P_i$ and it equals $\mathcal{M}^-$ if $j\in M_i$. Depending on the actual choice of the partition operators $\mathcal{M}^{+/-}$, the additional operators $\mathcal{I}^j$ might be required to provide some pre-processing of the local level set function, i.e. some local interpolation or projection on $\mathfrak{e}$. In the simplest case, it is a pure restriction: $\mathcal{I}^j(\mathfrak{e},\phi_j)=\phi_j|_{\bar{\mathfrak{e}}}$.

The final sub-triangulation of a given cell E of the geometry mesh $\mathcal{T}_{\varkappa}$ may then be recursively defined as

$$S_0^i(E) = \{E\} \tag{2.14}$$

$$S_j^i(E) = \bigcup_{\mathfrak{e}\in S_{j-1}} \mathcal{M}_i^j(\mathfrak{e},\phi_j^{\mathfrak{e}}) \tag{2.15}$$

$$\phi_j^{\mathfrak{e}} = \mathcal{I}^j\left(\mathfrak{e},\phi_j|_E\right),\quad \phi_j^{\mathfrak{e}}\in C^0(\bar{\mathfrak{e}}). \tag{2.16}$$

The sub-triangulation $\mathcal{S}^i$ corresponding to the subdomain Ω_i is given by

$$\mathcal{S}^i = \bigcup\left\{S_{N_\phi}^i(E)\,|\,E\in\mathcal{T}_{\varkappa}\right\} \tag{2.17}$$

and the complete sub-triangulation corresponding to the global domain Ω results from

$$\mathcal{S} = \bigcup\left\{\mathcal{S}^i\,|\,i\in\{1,\ldots,N_\Omega\}\right\} \tag{2.18}$$

This definition assumes that $\mathcal{M}_i^j$ is well defined for any $j\in\{1,\ldots,N_\phi\}$ and therefore it must hold that $M_i\cup P_i=\{1,\ldots,N_\phi\}$ and $M_i\cap P_i=\emptyset$. Notice that this is more restrictive

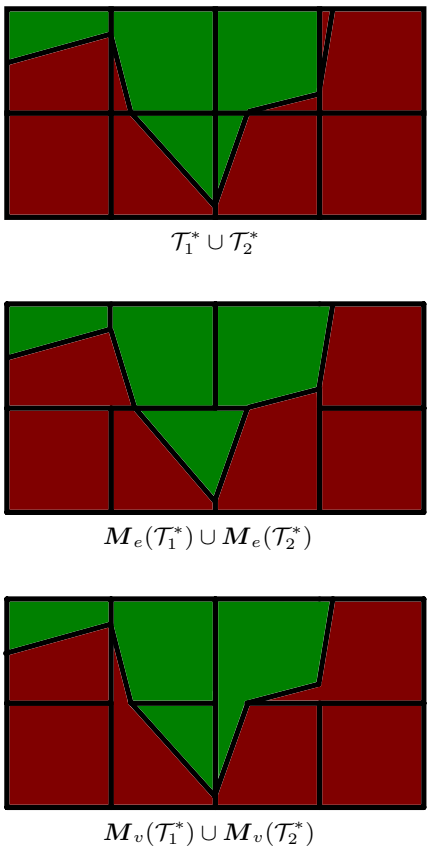


Figure 2.1: The pictures show exemplary partitions of a fundamental mesh $\mathcal{T}_h$ with 4×2 cells. The mesh is divided into subdomains Ω_1 (red) and Ω_2 (green). The first picture (top) shows the union of the intermediate meshes $\mathcal{T}_1^*$, $\mathcal{T}_2^*$. The final cut-cell meshes are defined by $\mathcal{T}_i = \boldsymbol{M}_e(\mathcal{T}_i^*)$ for $i \in \{1, 2\}$. The merging operator $\boldsymbol{M}_e$ will merge a cell that does not fulfill the necessary size/extent criterion, by merging it to the neighbor cell with which it shares the largest common face area. On the contrary, $\boldsymbol{M}_v$ will merge such a cell to the neighbor cell of biggest volume. The results are shown in the second picture (middle) for $\boldsymbol{M}_e$ and in the third picture (bottom) for $\boldsymbol{M}_v$. This example illustrates the tendency of $\boldsymbol{M}_v$ to create cells with cusps and spikes along the common boundary of both subdomains.

than the original definition of the subdomains Ω_i in section 2.1.2. However, the 2^{N_ϕ} subdomains which conform to this assumption form a partition of Ω. Furthermore, any subdomain conforming to the original definition can be constructed as a union of a sub set of this partition. An algorithm realizing a sub-triangulation according to the definition (2.14) is therefore suitable to realize a sub-triangulation for any of the subdomains which conform to the original definition in section 2.1.2.

Notice that, without any further and non-trivial requirement on the partition operator $\mathcal{M}^{+/-}$ in (2.11), the exact sub-triangulation defined via the recurrence in (2.14) will depend on the enumeration order of the level set functions ϕ_j with $(j = 1, \ldots, N_\phi)$.

2.2.4 Construction of the Cut-Cell Mesh

Intermediate cut-cell meshes for each of the subdomains Ω_i with $i \in \{1, \ldots, N_\Omega\}$ may be defined as

$$\mathcal{T}_i^* := \left\{ E \cap \hat{\boldsymbol{\Omega}}(\mathcal{S}^i) \mid E \in \mathcal{T}_h \right\} \tag{2.19}$$

and their union is denoted by

$$\mathcal{T}^* := \bigcup \{ \mathcal{T}_i^* \mid i \in \{1, \ldots, N_\Omega\} \} \tag{2.20}$$

Hence, $\mathcal{T}^*$ results directly from the restriction of the fundamental mesh $\mathcal{T}_h$ with regard to the sub-triangulation $\mathcal{S}$.

The intermediate meshes $\mathcal{T}_i^*$ and $\mathcal{T}^*$ may contain arbitrarily small cells which may appear in various problems as discussed in section 1.4 on page 22. These problems can be resolved by merging such cells to their neighbors until a specific size criterion is fulfilled. Efficient choices for this criterion may involve a minimum value for the cell volume or a minimum extent of its convex hull in the directions of the Cartesian coordinates.

There is no obvious optimal choice for such a criterion. To avoid a deteriorated convergence behavior of iterative linear solvers, it may be sufficient to merge only cells whose size is many orders of magnitude below the size of its neighbors. When explicit time-stepping schemes are employed it might be beneficial to merge any cut-cell independent of its size.

As soon as there is more than one subdomain, i.e. $N_\Omega > 1$, the construction of the final cut-cell mesh $\mathcal{T}$ has to be done with regard to the needs of the specific MDMP problem to be simulated. For a given subset $P_\mathcal{T} \subset \{1, \ldots, N_\Omega\}$ a cut-cell mesh is given by

$$\mathcal{T} := \boldsymbol{M}\left(\bigcup \{\mathcal{T}_i^* \,|\, i \in P_\mathcal{T}\}\right), \tag{2.21}$$

where $\boldsymbol{M}$ defines an operator which performs the merging procedure. While the details of this operator may be specific to the simulation problem, for two general meshes $\mathcal{A}$ and $\mathcal{B}$ with edges $\mathcal{E}_\mathcal{A}$ and $\mathcal{E}_\mathcal{B}$ such that $\mathcal{B} = \boldsymbol{M}(\mathcal{A})$ it must hold that

$$\mathcal{E}_\mathcal{B} \subset \mathcal{E}_\mathcal{A} \qquad \text{and thus} \qquad \forall E \in \mathcal{A} : \exists\, F \in \mathcal{B} : E \subset F. \tag{2.22}$$

Notice that, as illustrated in figure 1.4 on page 28, not only the extent but also the internal structure of the cut-cell mesh depends on the choice of $P_\mathcal{T}$.

Primitive but effective implementations of the merging operator $\boldsymbol{M}$ can be realized by simple procedures which operate individually on each cell of the given mesh. For such a cell, it reduces to the question whether this cell fails to meet the minimal size criterion and if so, then to which of its neighbor cell should it be merged. This process might be repeated if the merged cell does still not satisfy the size criterion.

The selection of the partner cell for merging is guided by the principal that the resulting merged cell should be neither too anisotropic nor contain any sharp spikes or cusps. An obvious choice would be to take the neighbor with the biggest cell volume. Alternatively one might decide to choose the neighbor with which the maximal face area is shared. The performance of both approaches for a very simple example geometry is illustrated in figure 2.1 on the facing page.

For all examples computations as given in this thesis, the merging was performed in a cell-wise approach until each cell in the final cut-cell mesh fulfilled a minimum volume criterion defined relative to the cell volumes in the fundamental mesh $\mathcal{T}_h$. Cells which failed this criterion were always merged with the neighbor with whom they shared the largest face area.

Although this simple method worked well for all the computations performed for this thesis, especially the three-dimensional examples were restricted to rather benign geometries. Therefore, the importance of a further development of appropriate merging procedures which are suited to the geometry and to the needs of the numerical method should not be underestimated.

2.3 Conformity Conditions for Partition Operators

The rather abstract definition of the multi-domain sub-triangulation as given in section 2.2 is not yet complete since it relies on the local partition operators $\mathcal{M}^{+/-}$. Its principal properties, given in section 2.2.2, can not guarantee the suitability of a sub-triangulation constructed according to the recursive scheme in (2.14), mostly due to the violation of (2.13). This chapter presents a classification of partition operators according to geometric conformity conditions and their influence on the suitability of the final sub-triangulation.

2.3.1 Consistency and Adjacency of Sub-Triangulations

In addition to the mere computation of the geometries of the sub-triangulation, an implementation must also provide an interface suitable for the assembling process involved in setting up the linear (sub-)problems resulting from DG discretizations of a general PDE.

This implies, that each element, i.e. cell-part, of the sub-triangulation must be identifiable. Hence, for each cell-part there must be a corresponding key which identifies it uniquely and can be used to associate it with a cell in the final cut-cell mesh $\mathcal{T}$. Furthermore, it must be possible to associate with each cell-part a list of those cell-parts with which it (at least partially) shares a common face.

While, of the two requirements, the former is usually easy to realize as long as the partition operators are deterministic, the *adjacency* information of the cell-parts is more difficult to obtain. The partition operators could, in principle, provide an adjacency list for the local partition of a given cell-part but their inherently local character (i.e. them depending only on the level set function values in the corner nodes of the considered cell-part) entails the need for an external mechanism establishing adjacency information.

Naturally, adjacency is only well defined as long as the sub-triangulation is *consistent*, i.e. as long as it does not contain any gaps or overlaps of the constituent cell-parts. The condition (2.13), while sufficient, is too restrictive with regard to an actual implementation as it inherently requires the handling of curved edges and faces.

Hence, more suitable and instructive properties of the partition operators which allow a concise classification of the consistency and adjacency of the resulting sub-triangulation are mandatory.

2.3.2 k-Conformity

The partition provided by the operators $\mathcal{M}^{+/-}$ for a given three-dimensional cell-part, does, in addition, effectively declare a partition for both its faces (co-dimension 1 sub-entities) and its edges (co-dimension 2 sub-entities). These partitions are canonically determined by the intersections of the elements (i.e. the cell-parts) in the partition with each of these sub-entities. The same holds correspondingly for two-dimensional cell-parts. For a more formal approach, consider a co-dimension k sub-entity $\mathfrak{f}$ of a cell-part $\mathfrak{e}$ and the level set function ϕ. Then there exists a unique set $F^+ := \{f_1^+, \ldots, f_{N_\mathfrak{f}^+}^+\}$ with $N_\mathfrak{f}^+$ elements and a unique set $F^- := \{f_1^-, \ldots, f_{N_\mathfrak{f}^-}^-\}$ with $N_\mathfrak{f}^-$ elements of open environments

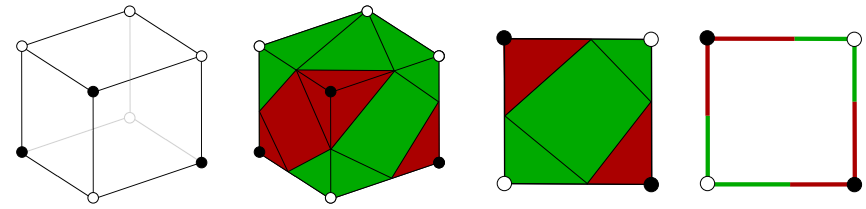


Figure 2.2: The pictures above illustrate the functionality of the partition operators $\mathcal{M}^{+/-}$ with regard to a cell-part and its sub-entities. The first picture from the left shows a cubical cell-part. The level set values in its corners are indicated by either black dots for a positive value of white dots for a negative value. The actual partition will of course depend not only on the sign but the actual magnitude of these values. The second picture shows a possible partition of the cube (which, of course, is its own co-dim 0 sub-entity), where the red cell-parts would be provided by $\mathcal{M}^{+}$ and the green cell-parts would be provided by $\mathcal{M}^{-}$. All cell parts in this partition are either cubes, simplices, or prisms and together fill the whole cube. The third picture shows the partition of the front face, i.e. a co-dim 1 sub-entity, which is implicitly determined by the volume partition from the second picture. Similarly, the fourth picture shows the partitions of the edges (co-dim 2 sub-entities) contained in the front face as implicitly determined by the volume partition in the second picture.

in $\mathfrak{f}$, such that

$$\left(\overline{\bigcup F^{+}}\right) \bigcup \left(\overline{\bigcup F^{-}}\right) = \bar{\mathfrak{f}} \tag{2.23}$$

and for every $f_i^{+/-}$ there is exactly one cell-part $\mathfrak{f}$ in $\mathcal{M}^{+/-}(\mathfrak{e}, \phi|_{\bar{\mathfrak{e}}})$ such that

$$\overline{\mathbf{\Phi}_{\mathfrak{e}}(\mathfrak{f})} \cap \overline{\Phi_{\mathfrak{e}}(\mathfrak{f})} = \overline{\Phi_{\mathfrak{e}}(f_i^{+/-})}. \tag{2.24}$$

As the sets $F^{+/-}$ describe a kind of restriction of $\mathcal{M}^{+/-}$ to the sub-entity $\mathfrak{f}$, the following notation is employed:

$$\mathcal{M}^{+/-}(\mathfrak{e}, \phi|_{\bar{\mathfrak{e}}})\Big|_{\mathfrak{f}} := F^{+/-}. \tag{2.25}$$

This is illustrated by an example in figure 2.2.

If a partition operator $\mathcal{M}^{+}$ or $\mathcal{M}^{-}$ guarantees that the partition of this sub-entity $\mathfrak{f}$, as determined by the total partition of the whole cell-part, does only depend on the restriction $\phi|_{\mathfrak{f}}$ of the level set function to $\mathfrak{f}$, then it is considered to be *k-sub-conforming* in that sub-entity. This implies that, independent of the level set function $\phi(\cdot)$, there exists an operator $\mathcal{M}_{\mathfrak{f}}^{+/-}$ such that

$$\mathcal{M}_{\mathfrak{f}}^{+/-}(\mathfrak{e}, \phi|_{\mathfrak{f}}) = \mathcal{M}^{+/-}(\mathfrak{e}, \phi|_{\mathfrak{e}})\Big|_{\mathfrak{f}}. \tag{2.26}$$

If this is not the case but a $\mathcal{M}_{\mathfrak{f}}^{+/-}$ which fulfills at least

$$\overline{\bigcup \mathcal{M}_{\mathfrak{f}}^{+/-}(\mathfrak{e}, \phi|_{\mathfrak{f}})} = \overline{\bigcup \mathcal{M}^{+/-}(\mathfrak{e}, \phi|_{\mathfrak{e}})\Big|_{\mathfrak{f}}}, \tag{2.27}$$

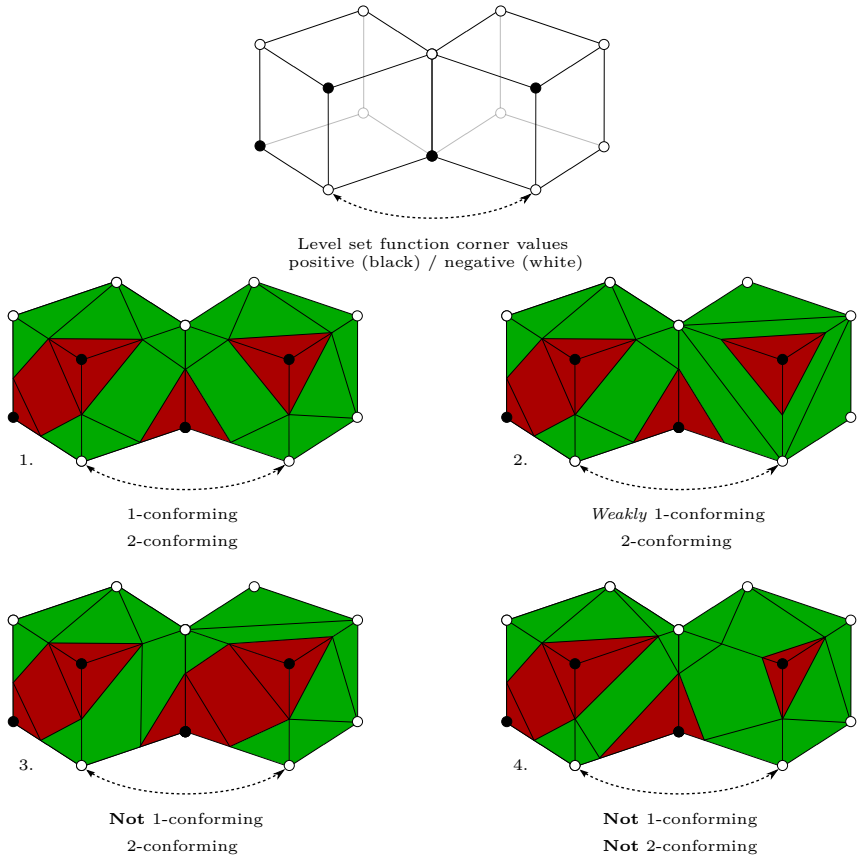


Figure 2.3: The pictures above illustrate the k-conformity property of the partition operators $\mathcal{M}^{+/-}$. The first picture on the top describes the input data for the operators consisting of two adjacent cubes and their level set function corner values. Only the sign of the values is given and indicated by either a black (positive value) or a white (negative value) circle. Notice that already in the second case, which shows the application of partition operators that is only weakly 1-conforming, the resulting sub-triangulation may already contain cells with non-matching faces. None of the presented examples indicates a violation of 3-conformity.

does exist, then the partition operator is considered to be *weakly k-sub-conforming* in the sub-entity f.

If a partition operator is *(weakly) k-sub-conforming* in every sub-entity, then it is considered *(weakly) k-conforming*, see figure 2.3.

According to the dependencies defined in (2.11), any *local* partition operator $\mathcal{M}^{+/-}$ is 0-conforming, meaning that the partition of a cell part does not depend on anything but the restriction $\phi|_{\mathfrak{e}}$ of the level set function to the operand cell-part $\mathfrak{e}$.

Notice that, although the partition of a node essentially reduces to a binary property, d-conformity is non-trivial as even a non-pathological partition operator might very well

decide to place a node in a subdomain that does not correspond to the level set values in this node. This appears especially reasonable if the magnitude of such a value is relatively small and all the other nodes in the cell-part correspond to level set values of opposite sign. Such an approach would effectively prevent the construction of minuscule cell-parts.

2.3.3 Fully k-Conforming Partition Operators

The properties of the final sub-triangulation as defined by (2.14) in section 2.2.3 depend heavily on the local partition operators $\mathcal{M}^{+/-}$. If those are assumed to be k-conforming in all co-dimensions, then the recursion in (2.14) leads to a *conforming* sub-triangulation (here, *conforming* is meant in the traditional sense, i.e. perfectly matching faces and edges).

To see this, consider the first step of (2.14). By definition, the cell-parts resulting from a single application of $\mathcal{M}^{+/-}$ to the geometry grid cell E are conforming. Furthermore, two cell-parts $\mathfrak{e}$ and $\mathfrak{f}$ that share a full sub-entity $\mathfrak{f}$ of co-dimension k also share the level set function values in $\mathfrak{f}$ (as we assumed ϕ to be continuous) and the application of the partition operators to $\mathfrak{e}$ and $\mathfrak{f}$ will, due to the definition of k-conformity, again result in only perfectly matching, i.e. conforming, cell-parts.

Full k-conformity of the partition operators does also simplify the computation of the adjacency information. In this case, in each step of the recurrence relationship as given in (2.14), the partition operators provide perfectly matching cell parts. Furthermore, the face-partitions of the co-dimension 1 sub-entities of each cell-part only depend on the level set function values in that face (1-conformity). Hence, the full adjacency information can be deduced - in each step of the recursive construction - from an ordering of the elements of this face-partition as long as it depends only on these level set function values itself. As the construction of such an ordering is a trivial task compared to the actual implementation of a full k-conforming operator, it is valid to say that such operators do inherently provide adjacency information.

2.3.4 Weakly 1-Conforming Linear Regularizing Partition Operators

Relaxing only the requirement of 1-conformity, changes the situation already tremendously. Even for a weakly 1-conforming set of partition operators $\mathcal{M}^{+/-}$ that is (strongly) k-conforming in all other co-dimensions, the recursion (2.14) may result in a sub-triangulation with rather undesirable properties.

Even if only one level set function ϕ_j is used, i.e. $N_\phi = 1$, then the resulting sub-triangulation $\mathcal{S}$ will not be conforming as can easily be seen in the example case 2 in figure 2.3. However, as the geometry mesh is a structured Cartesian mesh, the cell E has a linear reference mapping Φ_E and thus the partition property (2.13) is actually fulfilled, which implies that the resulting sub-triangulation will not have any gaps or overlaps of cell-parts. Furthermore, while the local sub-triangulations of two adjacent cells E and F in the geometry mesh $\mathcal{T}_\varkappa$ need not be conforming with each other, the restriction of their local domain approximations to their common face $\mathfrak{f}$ will (by definition of weak

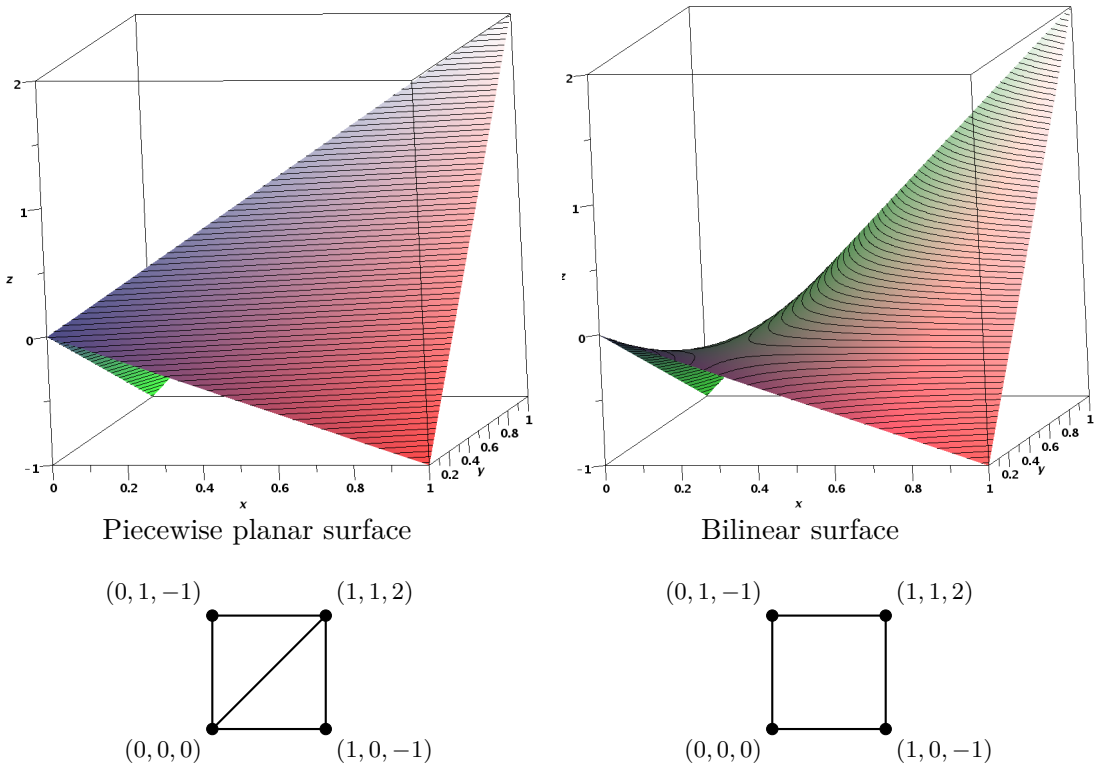


Figure 2.4: The pictures above show two 2D manifolds in 3D space, both with the same corner nodes. In one case (left-hand side), the manifold is constructed from two planar triangles, each connecting three of the nodes. In the other case (right-hand side), the manifold is given by a smooth surface defined by the bi-linear mapping, that is uniquely determined by the coordinates of the four corner nodes. Notice that while the border edges of both surfaces are identical, the discrepancy of the surfaces themselves is significant. If those surfaces would correspond to faces of adjacent cells, the resulting mesh would include a gap and could not be suitable for DG discretizations.

1-conformity) be equal, i.e.

$$\overline{\bigcup \mathcal{M}^{+/-}(E,\cdot)\Big|_{\mathfrak{f}}} = \overline{\bigcup \mathcal{M}^{+/-}(F,\cdot)\Big|_{\mathfrak{f}}}. \tag{2.28}$$

Hence, as long as all cell-parts within a given geometry mesh cell $E \in \mathcal{T}_\varkappa$ that belong to the same subdomain are located in the same mesh cell of the final cut-cell mesh $\mathcal{T}$, this cut-cell mesh will be conforming.

If the subdomain Ω_i is defined with regard to more than one level set function ϕ_j, i.e. a true MDMP setup, then the situation deteriorates significantly. As soon as $S_1 = \mathcal{M}_i^0(E,\cdot)$ in equation (2.14) contains two cell-parts $\mathfrak{e}$, $\mathfrak{f}$ which have a common quadrilateral face $\mathfrak{f}$ that is not planar (i.e. bi-linear), the lack of 1-conformity implies the possibility that its partitions as determined by $\mathcal{M}_i^1(\mathfrak{e},\cdot)\Big|_{\mathfrak{f}}$ and $\mathcal{M}_i^1(\mathfrak{f},\cdot)\Big|_{\mathfrak{f}}$ may not match and consist of an arbitrary mixture of linear and bi-linear sub-faces. Hence, for N_ϕ level set functions, the

resulting sub-triangulation S_{N_ϕ} of E may contain overlapping cell-parts as well as gaps in between them. As illustrated in figure 2.6, the approximation of a bi-linear surface with linear surfaces may be quite poor and therefore the integrity of the sub-triangulation can be severely compromised.

As such sub-triangulations are not desirable, it is obvious that in absence of 1-conformity additional requirements on the partition operators have to be made in order to ensure a consistent sub-triangulation. While there may be many ways to achieve this, it follows from the description above, that a partition operator which does not produce curved interior faces would be sufficient to prevent the appearance of internal gaps or overlaps. To this end, the pre-processing operators $\mathcal{I}^j$ can be employed ensuring a local linearization of the level set function ϕ_j. A partition operator which guarantees that for a linear level set function the resulting partition contains only cell-parts with linear faces is, in the following, called a *linear* partition operator. Details on the implementation of such operators will follow in section 2.4.

Assuming that the weakly 1-conform partition operators $\mathcal{M}^{+/-}$ is at least linear and the pre-processing operators $\mathcal{I}^j$ provide a suitable linearization of ϕ_j, then, although the resulting sub-triangulation will be benign, any algorithm destined to establish adjacency information of the cell-parts produced in the recursive approach of (2.14) still has to handle the problem of associating non-matching faces. Therefore, the aforementioned method based on a consistent ordering of each cell-partition is no more applicable and a geometric mapping approach based on the numerical coordinates of the cell-part faces can not be avoided.

While such an approach is far more awkward to implement correctly, the impact on computational efficiency is tolerable. For typical MDMP applications, only a few cells in the geometry mesh $\mathcal{T}_\varkappa$ will be actually intersected by two or more subdomain interfaces. Hence, the average number of cell-parts created per geometry grid cell is small. Brute-force attempts to match the cell-part faces are thus eligible and may be further improved by choosing a cheap intermediary test (e.g. comparing the normal vectors of two faces) in order to sort out most non-matching pairs quickly.

However, as any such matching algorithm has to make decisions based on coordinate values that have been compromised by errors from numerical floating point computations, it must rely on a minimum face size or edge length. Otherwise, the faces of very small adjacent cells might not be matched correctly and thus become subject to system dependent and possible non-deterministic numerical round-off errors. This translates into another requirement on the partition operators. In the following, any partition operator which guarantees, that all constructed cell-parts have an edge length greater than a given minimum value will be called *regularizing* operators.

As long as the operator is regularizing, this geometric approach can be implemented such that it provides reliable matching information for rather general sub-triangulations including non-matching faces. Therefore, it is more general than the method presented for the fully k-conforming operators. The example computations presented in this thesis utilize a geometric matching approach to set up the sub-triangulation $\mathcal{S}$.

2.4 Marching Cubes Partition Operators

The multi-domain sub-triangulation defined in section 2.2 depended strongly on the local partition operators $\mathcal{M}^{+/-}$. Their principal properties, as given in section 2.2.2, leave many freedoms with regard to the actual definition of such operators. The corresponding choices will not only determine the final sub-triangulation but also influence the complexity of its actual implementation. In section 2.3, very specific requirements on the partition operators were formulated ensuring the integrity of the resulting sub-triangulation as well as the reliability of the corresponding implementation for arbitrary input data.

The discussion in section 2.3 revealed, that any eligible partition operator should be *regularizing*. Furthermore it should be either *fully k-conforming* or at least *k-conforming* in all co-dimensions greater than one, *weakly* 1-*conforming* and *linear*.

In this section, the actual implementation of local partition operators is discussed with regard to the aforementioned requirements and other desirable properties, e.g. efficiency, topological accuracy and metric accuracy.

2.4.1 The Standard Marching Cubes Algorithm (MC16)

The problem of defining and implementing a local partition operator according to the requirements in section 2.2.2 is similar to the problem solved by the standard *marching cubes method* (MC16) [66]. This algorithm, which was developed for computer visualization, creates a piecewise linear approximation of the surface defined by the zero level set of a level set function. The main idea is to simply determine the zero level set's intersection point on each edge of a cell via linear interpolation of the two level set function values in the two nodes adjacent to this edge. Then these intersection points are connected according to a predefined pattern which assures that the result is a number of consistent polygons giving a reasonable approximation of the isosurface. This can be achieved by utilizing a predefined look-up map for each of the *topological cases* of the considered cell. As the MC16 partition of a cell depends only on the level set function value in its corners, such a topological case is determined by the sign of the level set function's corner values. Hence, for a three-dimensional cube, the number of topological cases is equal to 2^8. However, taking into account all relevant symmetries, there remain only 16 topological cases for which the connection pattern has to be actually implemented.

Although originally designed for the reconstruction of the iso-surface only, the MC16 can naturally be extended to fulfill the requirements of the UDG method by associating the 16 topological cases not only to a connection pattern of the intersection points, but to a full set of 16 *reference partitions* of the reference geometry of the considered cell into cell-parts. This principal approach has already been utilized and discussed in earlier publications on the UDG method [20, 55, 67]. In the following, all references to the MC16 method refer to this extended approach.

According to this extension, the MC16 may be interpreted as a partition operator. For a cell-part $\mathfrak{e}$ with $N_\mathfrak{e}$ corner nodes $\mathbf{x}_1, \ldots, \mathbf{x}_{N_\mathfrak{e}}$ it provides a mapping to $N < N_{\mathcal{M}_{16}}$ cell-parts according to

$$\mathcal{M}_{16}^{+/-}(\mathfrak{e}, \phi_j^\mathfrak{e}(\mathbf{x}_1), \ldots, \phi_j^\mathfrak{e}(\mathbf{x}_{N_\mathfrak{e}})) = \{\mathfrak{f}_1, \ldots, \mathfrak{f}_N\} \tag{2.29}$$

It is noteworthy that the MC16 method when utilized for the definition of a partition operator automatically entails 3-conformity and 2-conformity. Furthermore, it is easily possible to choose the 16 reference partitions such that the resulting operator is linear according to the definition in section 2.3.4. Sadly, for three-dimensional cubes (or any other geometry containing a face with more than three corners), even weak 1-conformity is not given, see case 3 in figure 2.3 on page 40. The problem results from the fact that a face with more than three nodes may contain more than two intersected edges. Hence, a decision on how to connect the respective intersection points within the face can not be uniquely derived from the binary information in each of the face's corners. For the MC16, this decision is implicitly derived from the choice of one of the 16 reference partitions and the location of the considered face within the reference geometry. Therefore, the result depends on the enumeration order of the corners of the face within the respective cube and is therefore effectively random. Two adjacent cubes are thus likely to produce non-matching partitions for their common face, thereby violating the requirement of even weak 1-conformity.

The concept of the MC16 is easily extended to other geometric objects. Especially for a simplex, the look-up table of a Marching Simplex algorithm becomes trivial and the lack of faces with more than three corners allows a simple implementation of a corresponding operator $\mathcal{M}_S^{+/-}$ with weak 1-conformity. Utilizing triangular prisms, fully k-conforming partition operators for simplices can be easily defined. In the context of the recursive construction method given in (2.14), this entails the need to also define a partition operator for such triangular prisms. While this is possible, the contained quadrilateral faces introduce the same problems with regard to 1-conformity as described for the MC16.

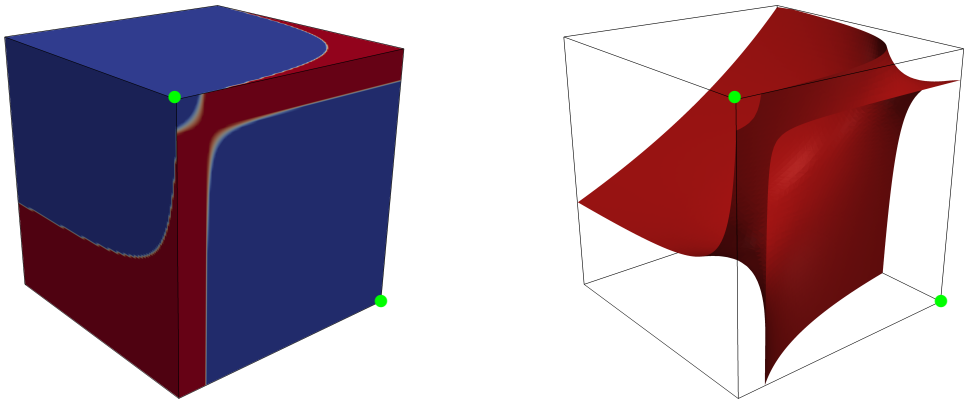
2.4.2 Marching Cubes Algorithms with Topological Guarantees

To obtain weak 1-conformity, the standard marching cubes method has to be modified such that the constructed iso-surfaces are necessarily continuous across the common face of two adjacent cubes. This corresponds to a decision about which corners of the face are connected within this face. As a Marching Cubes based partition operator has no information about the level set function within a face apart from the corner values, the optimal result would correspond to the connectivity defined by the corresponding interpolation polynomial, e.g. a bi-linear polynomial for a quadrilateral face. In the context of such iso-surface reconstruction methods, this property is referred to as a *topological guarantee* or *topological correctness*.

An extended version of the marching cubes method which utilizes 33 instead of only 16 cases and provides some degree of topological correctness was presented in [68] and is in the following called MC33. It utilizes the fact that the connectivity of the face's corners with regard to the interpolation functions can be expressed as a (very cheap arithmetic) test, the results of which can be used to simply extend the key for the look-up map.

As those tests utilize only the level set function values in the corners of the cell-parts, a set of partition operators $\mathcal{M}_{33}^{+/-}$ can be defined with the same dependencies as the MC16 operator 2.29.

However, as shown in figure 2.6, due to the finite number of intersection points, there are cases for which the connectivity of the corners within the volume of the cube and



Level set function values

Zero level set

Figure 2.5: The pictures above show the subdomains (red-blue) as well as the interface between them corresponding to a tri-linear level set function with the corner values [0.05, -0.4, 0.5, 0.9, -0.005, 0.05, -0.9, -0.005]. The two corners indicated by the green dots lie within the same subdomain and a path connecting them can easily be found as long as it may penetrate the cube volume. However, when considering only the cube's surface, those corners are not connected. Therefore, a partition operator like the $\mathcal{M}_{33}$, which utilizes only the vertices that coincide with either the cube corners or the intersection points of the cube's edges and the zero level set, can not connect the two green corners if the topological cases of all six faces are to be respected.

their connectivity within all faces can not be both fulfilled by any set of piecewise linear polygons connecting those intersection points. The table presented in [68] gives preference to the volume connectivity (e.g. case 12.1.2, table 1) and would thus result in a local partition operator that violates weak 1-conformity.

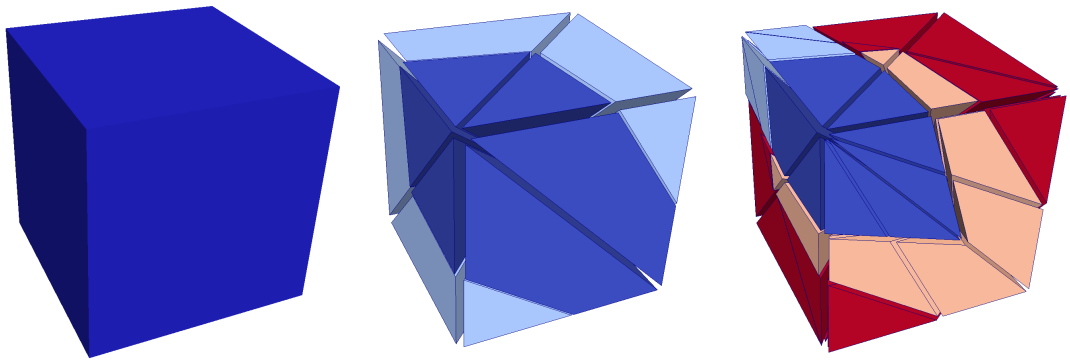
There are multiple ways to fix this problem: The simplest approach would be to change the look-up map such that the implicit priority of volume connectivity is discarded. If this neglect of topological guarantees is not acceptable, additional corner nodes for the constructed iso-surface (and thus also the partition into cell-parts) must be introduced and positioned appropriately within the considered cell-part.

Using the latter modification, it is possible to define the reference partitions such that the resulting operator is fully k-conforming.

Look-up tables and partitions for a fully k-conforming partition operator $\mathcal{M}_P^{+/-}$ operating on triangular prisms can be defined in the same way.

2.4.3 Linearization and the Marching Cubes

The implementation of a MC33 partition operator including the actual volume partition which is crucial for any UDG application is an extremely time consuming effort even without considering the additional requirements resulting from full k-conformity or curved iso-surfaces as outlined in section 2.4.2. A swift glance at the popular libraries utilized by the wider community of scientific computing reveals that the simplicity of implementation



Initial cell | Partitioning, first step | Partitioning, second step

Figure 2.6: To ensure a sound sub-triangulation suitable for DG discretizations, the initial cell may be pre-partitioned into simplices. On each simplex, the values of the level set functions in its corners may be utilized to define planar interfaces. This pre-partitioning avoids the difficulties resulting from non-matching triangulations of adjacent entity parts as well as awkward topological cases (see figure 2.6 and figure 2.3 respectively). The pictures above show exemplary how such a sub-triangulation is created in two steps by recursively applying the partitioning algorithm to an initially cubic cell of the geometry mesh $\mathcal{T}_\varkappa$.

may very well influence the popularity of an algorithm just as strongly as its performance.

Therefore it is worth mentioning that any implementation of a partition operator based on the standard MC33 look-up table may be utilized to define a partition operator suitable to the applied in the recursive scheme (2.14) to define a MDMP setup. This is achieved by local linearization of the level set function. As the problematic cases of the MC33, which result in a conflict of the volume and the face corner connectivity, only occur for curved isosurfaces (see figure 2.6), it is straight-forward to modify the level set function such that the tri-linear function corresponding to the eight values in the cube's corners do actually correspond to a linear function. A simple modification of this sort, which guarantees that the overall isosurface is still continuous consists of a single pre-processing step which pre-partitions each cubical cell E of the geometry mesh $\mathcal{T}_\varkappa$ into 6 simplices. The restriction of the level set functions $\phi_j\big|_E$ may then be replaced by the continuous piecewise linear interpolation functions $\phi_j^{P1(E)}\big|_E$ implicitly defined by the corner values of each simplex. This mapping can be incorporated into the recursive scheme (2.14) by setting the pre-processing operator to $\mathcal{I}^j(\mathfrak{e}, \phi_j) := \phi_j^{P1(E)}\big|_{\mathfrak{e}}$.

The first step S_1 of the recursive sub-triangulation in (2.14) then corresponds to this pre-partition of the cubical cell, followed by an application of the Marching Simplex algorithm to each of the constructed simplices. If in each of the following steps S_i with $i > 1$ the corner values of the considered cell-parts are evaluated for the function $\phi_j^{P1(E)}\big|_E$, the corresponding isosurface is guaranteed to be linear within that part. Therefore, the topo-

logical guarantees provided by the linear MC33 operator will suffice to ensure an at least weakly 1-conforming partition operator in spite of the issues described in section 2.4.2. A more formal description of this scheme is given in section 2.4.7.

2.4.4 Implementation

The numerical examples presented in this thesis rely on an implementation based on the look-up table in [68]. It resulted from a yet unpublished effort of *Andreas Nüßing and Christian Engwer* from the *Westfälische Wilhelms-Universität Münster*. In addition to the look-up table as given in [68], they implemented reference partitions for each of the topological cases. While it is in principle possible to realize the partition in a way that ensures strong 1-conformity of the corresponding partition operator, the requirements for an implementation that is *only* weakly 1-conforming were already so demanding, that the additional effort required for full k-conformity could not be completed in time to be considered in this thesis.

In its outlines, the implementation is similar to the one employed for the previous publications on the UDG method [20, 55]. In addition to the MC33, *Nüßing and Engwer* devised and implemented look-up tables with topological guarantees similar to the MC33 and the corresponding reference partitions (also without enforcing 1-conformity) for triangular prisms and pyramids (although this thesis relies on a version which does not utilize pyramids).

2.4.5 Regularization

The implementation of *Nüßing and Engwer* was extended to ensure a *regularizing* partition operator as defined in section 2.3.4. In each application of the partition operator, the constructed cell-parts are searched for edges that do not satisfy the requirement of a minimum length, in the following denoted by ε_p. If such an edge is found, the corresponding intersection point is relocated, i.e. moved along its edge until the requirement is satisfied. To ensure that this is possible and the final triangulation has sufficiently large edges even in a worst case scenario, the parameter for the minimum length has to be chosen individually for each step S_j in (2.14) as $2^{N_\phi - j}\varepsilon_p$.

Sadly, this approach in principle corrupts the weak 1-conformity of the resulting partition operator. This problem is illustrated in figure 2.7 for a vastly exaggerated minimum length parameter. It is obvious that the geometric extent of the infringement is of the order of magnitude of the parameter ε_p itself. Hence, the error introduced in the numerical integration of the face is of the order of magnitude of the error introduced by the relocation of the intersection point. Therefore, in disregard of this issue, throughout the following chapters the utilized partition operator is referred to as weakly 1-conforming.

2.4.6 Unique Evaluation of the Level Set Functions

To this point, no consideration was given to the fact that numerical errors in the evaluation of the level set function might complicate the unique association of a cell-part corner node $\mathbf{x}$ to the corresponding level set function value in that node. This association was implicitly assumed in the definition of the local partition operators $\mathcal{M}^{+/-}$

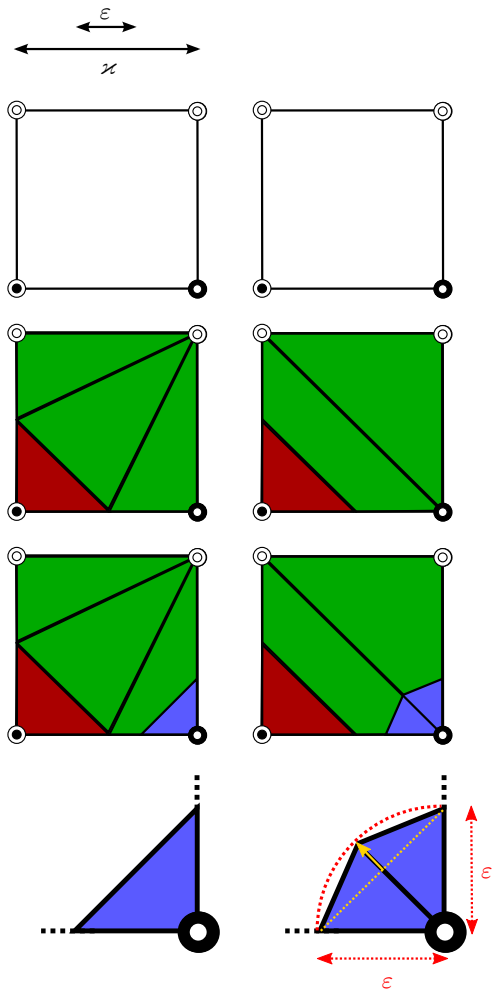


Figure 2.7: The pictures to the left illustrate two possible partitions of a common face of two adjacent cubes, assuming a linear weakly 1-conforming partition operator which was implemented based on the MC33 look-up table. The first three pictures depict the three stages S_0, S_1, and S_2 of the recursive sub-triangulation process defined in (2.14) assuming two level set functions the corner values of which are depicted by black and white dots for positive and negative values (interior dot corresponds to ϕ_1, exterior dot corresponds to ϕ_2). For purposes of illustration, the minimum edge length parameter is exaggerated to amount to about $\frac{1}{4}h$. Although the linear operator ensures straight isosurfaces in principle, the resulting cell-parts in S_2 would violate the minimum length requirement of the edge separating the two blue cell-parts in the third picture to the right. Relocating the intersection point according to yellow arrow in the zoom in the last picture to the right, introduces a discontinuity of the iso-surface and a violation of weak 1-conformity

in section 2.2.2 and in the subsequent analysis of the sub-triangulations constructed via recursive application of these operators. Evidently, the influence of such errors will me most prominent in corners in which a level set function evaluates to a value so close to zero that the numerical error may flip its sign. Especially in cases were both corners of an edge correspond to such small values, numerical errors have the potential to severely corrupt any kind of k-conformity nominally fulfilled by the employed partition operators, see figure 2.8.

For many practical applications, this issue may be negligible as the probability of such incidents may be quite low. However, benchmark problems and example setups chosen for teaching purposes are often based on simple geometries with nicely aligned borders or moving shock waves. Implementations based on ambiguous evaluations of the level set functions may thus thrive in the complex and yet fail in the simple.

A robust implementation of the sub-triangulation must therefore guarantee a unique evaluation of the level set functions at the corners of the cell-parts. For such corners that coincide with corners of the Cartesian geometry mesh $\mathcal{T}_\varkappa$, there may be many different

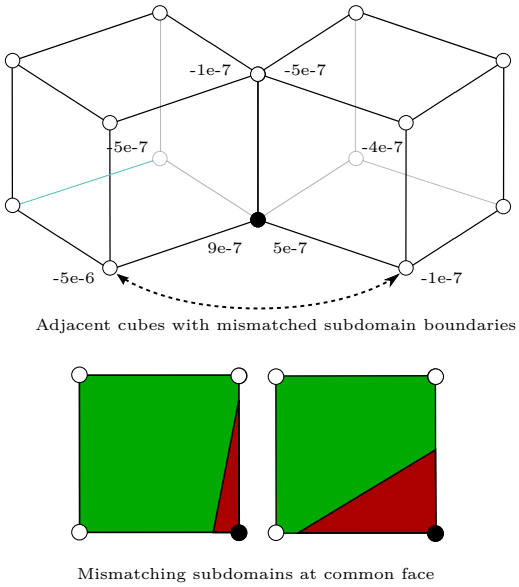


Figure 2.8: The evaluation of the level set function values is subject to numerical errors. The upper picture indicates the determined subdomain boundaries for the given level set function values, assuming that within each cube the numerical error modifies the original values within the range of 10^{-6}. Here it is assumed that the partition operator simply connects the zero level set position on each edge computed by linear interpolation to determine the subdomain boundary. Since in this example the values themselves are in the range of numerical error, the resulting partition of the common face, computed for each cube respectively, does not match at all (lower picture). Although this is an extreme example, it illustrates that small numerical errors may result in large geometric deformations.

evaluation schemes that guarantee a deterministic program path. However, as soon as there is more than one level set function and thus (2.14) involves more than one level of recursion, the corner coordinates of the processed cell-parts will eventually entail numerical errors that will inevitably break such a deterministic evaluation.

Given a set of *regularizing* partition operators $\mathcal{M}^{+/-}$, the minimum distance between two given corners can be utilized as a key ingredient for an algorithm that is able to recognize identical coordinates in spite of their numerical errors. Hence, this property is obviously desirable, even for fully k-conforming partition operators. For the numerical examples presented in this thesis, this was realized by a hash map for the level set function values. The keys for this map were constructed from the one-dimensional projection of the coordinate values onto a Cartesian sub-grid with a grid length equal to the regularization parameter ε_p.

2.4.7 Definition of Partition Operators

Summing up the results presented in this section, suitable definitions of partition operators may be presented:

The following two definitions are based on the partition operators $\mathcal{M}_{33}^{+/-}$, $\mathcal{M}_{S}^{+/-}$, and $\mathcal{M}_{P}^{+/-}$ for cubes, simplices and triangular prisms as introduces in section 2.4.1 and section 2.4.2. However, extended implementations which guarantees full k-conformity are denoted by $\mathcal{M}_{33k}^{+/-}$, $\mathcal{M}_{Sk}^{+/-}$, and $\mathcal{M}_{Pk}^{+/-}$ while $\mathcal{M}_{33}^{+/-}$, $\mathcal{M}_{S}^{+/-}$, and $\mathcal{M}_{P}^{+/-}$ are assumed to refer to implementations which simply realize some kind of volume partition corresponding to the topological guarantees as given in [68]. All operators are assumed to be *linear* and *regularizing* as defined in section 2.4.3 and section 2.4.7. Furthermore, it is assumed that the problem of unique level set function values, described in section 2.4.6, is somehow resolved.

When utilizing the fully k-conforming operators, no modifications are necessary and setting

$$\mathcal{M}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}) := \begin{cases} \mathcal{M}_{33k}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}(\mathbf{x}_1), \dots, \phi_j^{\mathfrak{e}}(\mathbf{x}_8)) & \text{if } \mathfrak{e} \text{ is a cube,} \\ \mathcal{M}_{Sk}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}(\mathbf{x}_1), \dots, \phi_j^{\mathfrak{e}}(\mathbf{x}_4)) & \text{if } \mathfrak{e} \text{ is a simplex,} \\ \mathcal{M}_{Pk}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}(\mathbf{x}_1), \dots, \phi_j^{\mathfrak{e}}(\mathbf{x}_6)) & \text{if } \mathfrak{e} \text{ is a triangular prism} \end{cases} \tag{2.30}$$

and

$$\mathcal{I}^j(\mathfrak{e}, \phi_E) := \phi\big|_{\mathfrak{e}} \tag{2.31}$$

ensures a consistent sub-triangulation for any number of level set functions.

If fully k-conforming implementations are not available, a consistent sub-triangulation can be achieved by setting

$$\mathcal{M}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}) := \begin{cases} \mathcal{M}_{33}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}(\mathbf{x}_1), \dots, \phi_j^{\mathfrak{e}}(\mathbf{x}_8)) & \text{if } \mathfrak{e} \text{ cubic and } \phi_j^{\mathfrak{e}} \text{ is linear} \\ \mathcal{M}_{P}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}(\mathbf{x}_1), \dots, \phi_j^{\mathfrak{e}}(\mathbf{x}_6)) & \text{if } \mathfrak{e} \text{ is a triangular prism and } \phi_j^{\mathfrak{e}} \text{ is linear} \\ \mathcal{M}_{S}^{+/-}(\mathfrak{e}, \phi_j^{\mathfrak{e}}(\mathbf{x}_1), \dots, \phi_j^{\mathfrak{e}}(\mathbf{x}_4)) & \text{if } \mathfrak{e} \text{ is a simplex,} \\ \bigcup \left\{ \mathcal{M}_{S}^{+/-}(\mathfrak{f}) \mid \mathfrak{f} \in \mathcal{L}(\mathfrak{e}) \right\} & \text{otherwise.} \end{cases} \tag{2.32}$$

Here $\mathcal{L}$ describes the linearization operator described in section 2.4.3, which provides a uniform and regular partition of a given cube into simplices. In order to enforce cell-parts with only planar faces, the pre-processing operator must be set to

$$\mathcal{I}^j(\mathfrak{e}, \phi_E) := \phi^{P1(E)}\big|_{\mathfrak{e}} \tag{2.33}$$

as defined in section 2.4.3. Naturally, the uniform partition of a cube into simplices as implied by both the definition of $\phi^{P1(E)}$ and $\mathcal{L}$ must match. Furthermore, it must be such that the full linearization of all cells in the fundamental mesh, i.e. the simplex mesh resulting from the application of $\mathcal{L}$ to all cells in $\mathcal{T}_h$, is conforming.

Notice that in (2.32), the linearization will only be performed in the initial step and thus $\mathcal{L}$ needs to be defined for cubes only. However, $\mathcal{L}$ can be naturally extended to triangular prisms, since such geometries can of course be partitioned into simplices. This procedure will be relevant for the modification of (2.32) as proposed in section 3.7.

2.5 A Note on Memory Consumption

All considerations so far were restricted to *purely local* partition operators $\mathcal{M}^{+/-}$, which suggests the potential to reduce the memory consumption required by the sub-triangulation to $\mathcal{O}(1)$ (assuming that the memory required for the representation of the level set function is also $\mathcal{O}(1)$). However, while this would be feasible for a data structure that provides nothing but an interface to the cell-parts in the sub-triangulation $\mathcal{S}$, maintaining this restriction on the memory consumption for a data structure which associates this sub-triangulation with the final cut-cell mesh $\mathcal{T}$ will result in some impractical limitations.

This is a direct consequence of the non-locality of the cut-cell mesh $\mathcal{T}$ which contains *merged* cells overlapping multiple cells in the fundamental mesh $\mathcal{T}_h$. A full *on-the-fly* operation to determine the association of a cell-part to a cell in $\mathcal{T}$ appears undesirable as it would necessarily imply an *on-the-fly* determination of the merging pattern. Not only does the non-locality of the merging operation indicate a disappointing performance of such an approach, furthermore, the algorithm determining the merging pattern would have to provide consistent results independent of the cell it is applied to and the ordering of this cell's neighbors - a tricky problem in itself.

Therefore, the pre-computation of a map associating the cell-parts with the cells in the cut-cell mesh $\mathcal{T}$ is unavoidable. As the number of cell-parts which can be constructed for a given cell in the geometry grid $\mathcal{T}_\varkappa$ is bounded by a constant (depending only on the number of level set function N_ϕ), the memory requirement scales with the number of cells in the geometry mesh, i.e. $\mathcal{O}(\varkappa^{-d})$.

However, in many practical applications involving connected subdomains with smooth boundaries, the number of cells in the fundamental grid that are actually intersected by the boundary of a subdomain will be small. For a cell E_H of the fundamental mesh $\mathcal{T}_h$ not intersected by any subdomain boundary, the storage of the relation of its interior cell-parts to their respective cut-cell grid cell E can be neglected as necessarily $E_H \subset E$.

Furthermore, the faces of cell-parts which coincide with (or are located on) faces in the cut-cell mesh $\mathcal{T}$ are distinguished in the sense that typical DG discretizations require the computation of surface integrals on such a face. Whether it is beneficial to store this adjacency information, which has to be pre-computed for the merging of the cut-cell mesh anyway, or whether it should be recomputed *on-the-fly* during assembling is not obvious and might depend on the chosen order of the finite element basis. While the latter approach is doubtless feasible, for all numerical experiments as presented in this thesis the adjacency information was pre-computed and stored in memory.

With regard to the initial assumption, that the memory consumption of the data-structures representing the level set functions would correspond to $\mathcal{O}(1)$, it is important to point out, that this can only be true for functions given by an analytical expression and thus not very likely to be encountered in practical applications. With regard to the memory consumption, the UDG scheme develops its full potential in applications that do inherently require the representation of a level set or phase field function. Alternative approaches with unstructured grids will in such situations inevitably create a redundant representation of the geometry in memory.

3 Two-Phase Flow

The principal laws governing the dynamics of two-phase flow are presented in this chapter and some variants of suitable UDG discretizations are proposed. These methods constitute a generalization of the approach in [35] which was restricted to the *fitted* case of rectangular domain, did not allow for any contact line with microscopic contact angle, and was exclusively based on the standard level set method in combination with a finite-difference discretization of the surface tension. Details on the principals and suitable discretizations of surface tension are deferred to chapter 4.

3.1 Geometric Setup

The continuous formulation of two-phase flow owes a significant part of its complexity to the time evolution of the problem domains. Generally, the two immiscible fluids are assumed to be contained within the subdomains $\Omega_1(t)$ and $\Omega_2(t)$ of a global domain Ω and separated by the common interface Γ such that

$$\Omega_1(t) \cup \Omega_2(t) \cup \Gamma(t) = \Omega = \text{const.} \tag{3.1}$$

The evolution of the interface is determined by the velocity field as given by the system equations (presented in section 3.2) describing essentially the conservation of mass and momentum. Due to the particularities of the UDG method, the two domains are assumed to be embedded in a rectangular domain Ω_r and the boundaries Γ_O which coincide with the boundaries of Ω_r are distinguished from the interior boundaries Γ_I in order to prescribe appropriate boundary conditions, see figure 3.1.

Each of the disconnected parts of the boundaries Γ_I is assumed to be smooth (in order to simplify the upcoming considerations of the microscopic contact angle). The *contact lines* γ are defined as the intersection of the two fluids' interface Γ with the interior boundaries Γ_I according to

$$\gamma := \overline{\Gamma} \cap \Gamma_I. \tag{3.2}$$

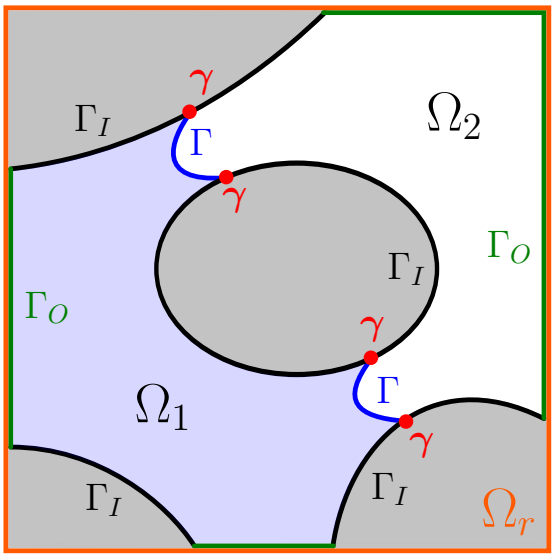


Figure 3.1: The two-phase flow problem consists of two time-dependent domains Ω_1 and Ω_2 separated by a smooth interface Γ. Due to the requirements of the UDG method, the problem setups are assumed to be embedded in a rectangular domain Ω_r and distinguish domain boundaries Γ_I located within Ω_r from boundaries Γ_O which coincide with its own boundaries.
The interface Γ may intersect the interior boundary Γ_O in what is called the *contact line* (in two dimensions the contact line reduces to a point).

For two-dimensional setups, the contact lines reduce to a number of separate points.

Assuming that Γ is smooth, the subdomains $\Omega_1(t)$ and $\Omega_2(t)$ may be associated with level set functions $\phi_B \in C^2(\Omega_r)$ and $\phi \in C^0([t_0, T]; C^2(\Omega_r))$ according to

$$\begin{aligned} \Gamma_I &:= \{\mathbf{x} \in \Omega_r \mid \phi_B(\mathbf{x}) = 0\}, & \Omega &:= \{\mathbf{x} \in \Omega_r \mid \phi_B(\mathbf{x}) > 0\}, \\ \Gamma(t) &:= \{\mathbf{x} \in \Omega \mid \phi(t, \mathbf{x}) = 0\}, & \Omega_1(t) &:= \{\mathbf{x} \in \Omega \mid \phi(t, \mathbf{x}) > 0\}, \\ & & \Omega_2(t) &:= \{\mathbf{x} \in \Omega \mid \phi(t, \mathbf{x}) < 0\}. \end{aligned} \tag{3.3}$$

Notice that these definition of the subdomains are consistent with those given in section 2.1.1 and section 2.1.2 and thus suitable for the construction of a sub-triangulation that can be employed for a UDG method.

3.2 Governing Equations

The time development of

$$\begin{aligned} &\text{the velocity fields } \mathbf{u}_i \in C^0\left([t_0, T]; C^2(\Omega_i(t))\right)^d \quad \text{and} \\ &\text{the pressure fields } p_i \in C^0\left([t_0, T]; C^1(\Omega_i(t))\right) \end{aligned} \tag{3.4}$$

within the domains Ω_i for $i \in \{1, 2\}$ filled by the two immiscible fluids is given by the incompressible Navier-Stokes equations together with the transport equation for the level set function:

$$\rho_i \partial_t \mathbf{u}_i + \rho_i \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p_i + \nabla \cdot \mathcal{D}_i(\mathbf{u}) \qquad \forall \mathbf{x} \in \Omega_i(t), \tag{3.5}$$

$$\begin{aligned} \mathcal{D}_i(\mathbf{u}) &= \mu_i \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T\right) && \forall \mathbf{x} \in \Omega_i(t), \\ \nabla \cdot \mathbf{u} &= 0 && \forall \mathbf{x} \in \Omega, \\ \partial_t \phi + \mathbf{u} \cdot \nabla \phi &= 0 && \forall \mathbf{x} \in \Omega. \end{aligned} \tag{3.6}$$

The parameters $\rho_i \in \mathbb{R}^+$ and $\mu_i \in \mathbb{R}^+$ describe the density and the dynamic viscosity of the two fluids which are assumed to be constant both in time and within each of the domains $\Omega_i(t)$. These equations describe the conservative transport of momentum within (and by) immiscible and incompressible Newtonian fluids accounting for the inertia of the fluids, the conservation of angular momentum and the viscous stress represented by the stress tensors $\mathcal{D}_i(\mathbf{u})$. No thermodynamical effects are considered.

In order to form a well-posed problem, boundary conditions which prescribe either a velocity field

$$\mathbf{u}_i^D \in C^0\left([t_0, T]; C^2(\Gamma_i^D(t))\right)^d \text{ on the Dirichlet boundaries } \Gamma_i^D \tag{3.7}$$

or a momentum flux

$$\mathbf{f}_i^N \in C^0\left([t_0, T]; C^1(\Gamma_i^D(t))\right)^d \text{ on the Neumann boundaries } \Gamma_i^N \tag{3.8}$$

have to be imposed on the system equations according to

$$\mathbf{u}_i(t,\mathbf{x}) = \mathbf{u}_i^D(t,\mathbf{x}) \qquad \forall \mathbf{x} \in \Gamma_i^D(t), \tag{3.9}$$

$$(-p_i\mathbf{I} + \mathcal{D}_i(\mathbf{u})) \cdot \mathbf{n} = \mathbf{f}_i^N(t,\mathbf{x}) \qquad \forall \mathbf{x} \in \Gamma_i^N(t), \tag{3.10}$$

and

$$\bigcup_{i\in\{0,1\}} \Gamma_i^D \cup \Gamma_i^N = \Gamma_O \cup \Gamma_I. \tag{3.11}$$

The interface conditions imposed on Γ ensure the continuity of the velocity field and the normal momentum across the interface. Denoting the jump of a quantity across the interface $\Gamma(t)$ by $[\![\cdot]\!]_\Gamma$, it thus holds that

$$[\![\mathbf{u}]\!]_\Gamma = 0 \qquad \text{and} \qquad [\![-p\mathbf{I} + \mathcal{D}(\mathbf{u})]\!]_\Gamma \cdot \mathbf{n}_\Gamma = \mathbf{f}_\Gamma, \qquad \forall \mathbf{x} \in \Gamma(t). \tag{3.12}$$

These conditions can be motivated from fluid mechanical principles [69] but essentially describe the natural coupling of the two subdomain problems defined in (3.5) using the boundary conditions in (3.9).

The temporal boundary conditions reduce to the prescription of initial velocity fields $\mathbf{u}_i^0 \in C^2(\Omega_i(t_0))$ with $\mathbf{u}_i^0(\mathbf{x}) = \mathbf{u}_i(t_0,\mathbf{x})$.

When conforming to some (severe) restrictions on the driving forces, the existence of solutions to the continuous Navier-Stokes equations for single phase flow is well established [70]. However, to the present day, non-trivial solutions to the two-phase flow problem as described by (3.5) are not known and no definite results on the solvability for general input data are available. Solution spaces for the primary unknowns were chosen in analogy with the classical analysis for the single phase flow problem and have no further relevance.

The two-phase flow model presented in this chapter is a widely used standard approach and well documented in textbooks, e.g. [69].

3.3 Contact Line Dynamics

At the contact lines γ, the fluids are subjected to molecular forces which may modify the contact angle of the fluid interface Γ with the domain boundary $\partial\Omega$. As illustrated in figure 3.2 on the following page, the angle α_M at which the surface apparently intersects with the domain boundary may thereby be modified on a scale which is typically orders of magnitudes below the characteristic scale of the underlying two-phase flow problem. The resulting *contact angle* α_c may deviate significantly from the angle α_M observed on the macro-scale. However, while the geometric error involved may be negligible, the resulting dynamics caused by the surface tension can very well dominate the global system dynamics.

Incorporating the contact line dynamics into the two-phase flow model as defined by (3.5) and (3.6), requires the introduction of additional sources of momentum defined on the contact line γ only. Different possibilities of how to define such terms with regard to the proposed discretizations of the two-phase flow problem will be given in chapter 4 on page 85.

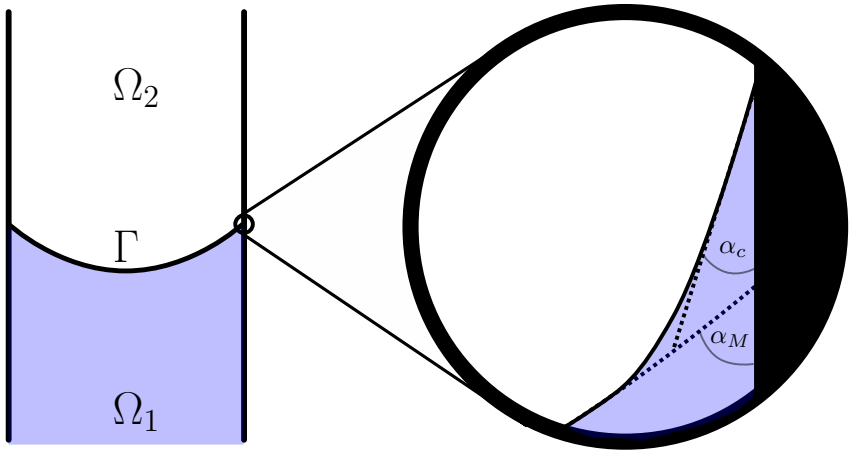


Figure 3.2: Due to molecular dynamics, the interface is distorted in the vicinity of the three-phase contact line modifying the macroscopic contact angle α_M. The resulting contact angle α_c is not uniquely determined by the two-phase flow equations (3.5), (3.6) and requires additional modeling. Experimental observations suggest, that it may in general depend on additional material parameters as well as the dynamics of all involved fluids and might show phenomena of hysteresis.

To the present day, there is ongoing research not only on accurate and efficient methods for the numerical treatment of contact line dynamics [31, 71, 72] but also the physical model on both the macro and the micro scale is not yet developed to a satisfying degree. This is due to the fact that the formation of the contact angle is subject to molecular dynamics. Finding a physical model which effectively describes the resulting dynamics on the characteristic scale of the flow problem constitutes a classical upscaling problem. While stationary equilibrium conditions can easily be derived from the relative surface tensions of the three materials defining the contact line γ, the mathematical modeling of the contact angle under dynamic conditions is still subject of current research. Models which describe the dynamic contact angle based on the fluid's properties, their relative surface tensions, and the velocity of the contact line have been proposed in [33, 69] and compared to both real and numerical experiments [31, 72].

Even if the concept of a dynamic contact angle is neglected, a moving contact line does invariably introduce a modeling problem for the boundary condition of the Navier-Stokes equations (3.5) in the vicinity of the contact line. The *no-slip* ($\mathbf{u} = 0$) condition which is traditionally imposed at all stationary rigid domain boundaries does in general not allow any movement of the contact line γ. On the other hand, there is comprehensive experimental evidence that the no-slip condition is the best modeling choice for many relevant flow systems. An extensive discussion of this *kinematic paradox* can be found in [69]. Slip conditions which resolve this inconsistency were already proposed and theoretically analyzed in [73, 74]. The simplest approach which has been widely adapted is the *Navier-slip* condition which scales the momentum transfer across the domain boundary

proportionally to the fluids' tangential velocity, according to

$$\forall \mathbf{x} \in \partial\Omega : \qquad \mathbf{u} \cdot \mathbf{n} = 0 \quad \text{and} \quad (\mathbf{I} - \mathbf{nn}) \left(\beta_N \mathbf{u} - (-p\,\mathbf{I} + \mathcal{D}) \cdot \mathbf{n}\right) = 0. \tag{3.13}$$

The parameter β_N scales the momentum transfer across the boundary. The choice $\beta_N = 0$ results in pure slip boundary condition while the classical no-slip boundary condition can be realized only in the limit $\beta_N \to \infty$. By varying β_N in the vicinity of the contact line γ, the Navier-slip condition may also be employed to realize local slip conditions which allow tangential velocities only in the vicinity of the contact line. However, effectively β_N is also a model parameter incorporating molecular dynamics and experimental predictions vary in orders of magnitude depending on the actual setup. A comprehensive overview of the existing models including their numerical comparison is given in [10].

In this thesis, discretization methods based on the UDG approach which allow a stable and efficient simulation of dynamic contact lines are presented. However, the scope is limited to the numerical difficulties and thus the utilized models were restricted to a constant contact angle $\alpha_c = 0$ and momentum transfer parameters β_N.

3.4 Time Discretization

The complexity of any time discretization for the two-phase flow problem originates from the instationary domains and the strong coupling between the domain geometry and the governing equations due to the surface tension. In this thesis, an operator splitting scheme is employed to decouple the equations (3.5) and (3.6). In each time step, this corresponds to the solution of (at least) two separate sub-problems on a constant mesh followed by the generation of a new cut-cell mesh for the next time step.

While some of the involved discrete function spaces are defined on constant meshes (e.g. for the level set function), other field variables (like the pressure which jumps across the interface) are represented on a sequence of meshes which track the moving interface between both fluids. This following section is focused on a formal definition of the meshes and function spaces involved in the solution of the Navier-Stokes and level set sub-problems.

3.4.1 Domains and Function Spaces

Assuming a constant domain geometry Ω given by the level set function $\tilde{\phi}_B \in C^0(\Omega_r)$, a discrete approximation of the domains at a given time t_k requires another function $\tilde{\phi}_h^k \in C^0(\Omega_r)$ which corresponds to a discrete approximation of the level set function $\phi(t_k)$ in (3.6), describing the position of the two-phase interface within Ω.

Let $\mathcal{S}^k$, $\mathcal{S}_1^k$, and $\mathcal{S}_2^k$ denote the sub-triangulations for Ω, $\Omega_1(t_k)$, and $\Omega_2(t_k)$ constructed according to the algorithm described in chapter 2 based on a fundamental mesh $\mathcal{T}_h$ and a geometry mesh $\mathcal{T}_\varkappa$ with

$$\hat{\mathbf{\Omega}}(\mathcal{T}_h) = \hat{\mathbf{\Omega}}(\mathcal{T}_\varkappa) = \overline{\Omega}_r$$

and level set functions for the subdomains Ω_1 and Ω_2

$$\phi_1 := \tilde{\phi}_B \qquad \text{and} \qquad \phi_2 := \tilde{\phi}_h^k.$$

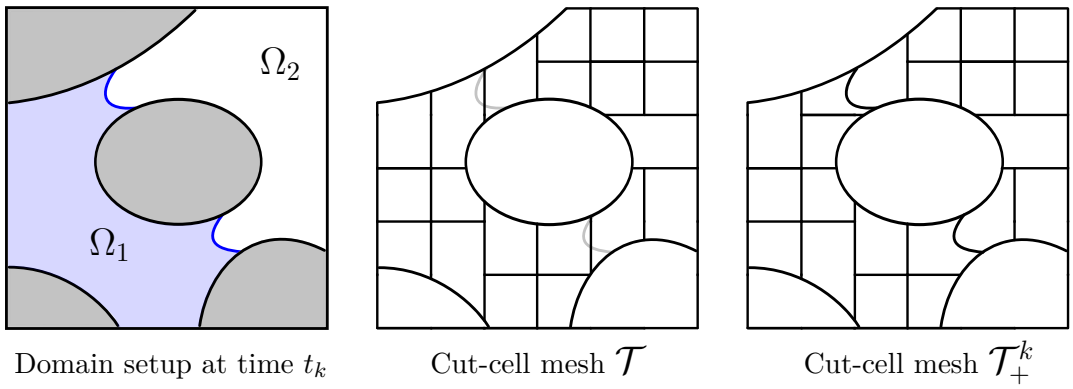


Figure 3.3: The pictures above illustrate a set of possible cut-cell meshes corresponding to the definition in 3.14 on the facing page. In general, $\mathcal{T}_+^k$ necessarily resolves the discrete interface $\Gamma^{h,k}$ at time t_k which is in general not true for the constant mesh $\mathcal{T}$. As the pressure field may involve jumps across the interface, it is beneficial to choose a finite element space with regard to $\mathcal{T}_+$. However, the level set function is smooth across the interface and as $\Gamma^{h,k}$ changes in every time step its finite element basis should be chosen with regard to $\mathcal{T}$. For the velocity field, both choices have advantages as discussed in section 3.5.3 on page 66.

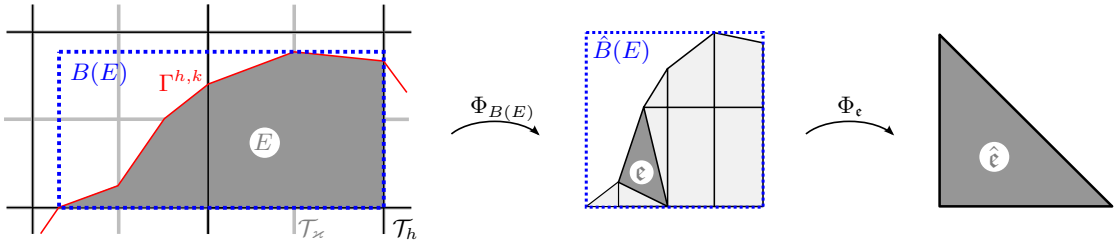


Figure 3.4: The pictures above illustrate the relations between the cut-cells, the bounding box and the sub-triangulation. At discrete time $t = t_k$, a cell $E \in \mathcal{T}_+^k$ may span multiple cells in the geometry mesh $\mathcal{T}_\varkappa$ and the fundamental mesh $\mathcal{T}_h$ but it may not overlap the discrete interface $\Gamma^{h,k}$ (though cells in $\mathcal{T}$ might). The shape functions of the finite element basis (e.g. monomials 1, x, y, z, x^2 ...) are defined on the reference element $\hat{B}(E)$ of the bounding box $B(E)$ of cell E. This is always a unit square (or unit cube if $d = 3$) and the mapping $\Phi_{B(E)}$ is always linear. Integrals of these shape functions (or products of them) are always computed on the cell-parts of the sub-triangulation. As a given cell-part $\mathfrak{e}$ is always a simple geometric object, i.e. a simplex, a cube or a triangular prism, Gaussian quadrature rules for exact integration of polynomials may be applied and the shape functions can be evaluated after mapping the quadrature points via $\Phi_{\mathfrak{e}}^{-1}$ on $\hat{B}(E)$.

To ensure a more suggestive notation, the discrete domains Ω^h, $\Omega_1^{h,k}$, $\Omega_2^{h,k}$ are introduced according to

$$\Omega^h = \hat{\mathbf{\Omega}}(\mathcal{S}^k), \qquad \Omega_1^{h,k} = \hat{\mathbf{\Omega}}(\mathcal{S}_1^k), \qquad \Omega_2^{h,k} = \hat{\mathbf{\Omega}}(\mathcal{S}_2^k).$$

The discretization is based on the cut-cell meshes $\mathcal{T}_+^k$, $\mathcal{T}_1^k$, $\mathcal{T}_2^k$ and $\mathcal{T}$:

$$\begin{aligned} \mathcal{T} &= \boldsymbol{M}\left(\mathcal{T}^*\right), & \mathcal{T}^* &= \left\{\Omega^h \cap E \,|\, E \in \mathcal{T}_h\right\}, \\ \mathcal{T}_i^k &= \boldsymbol{M}\left(\mathcal{T}_i^{*,k}\right), & \mathcal{T}_i^{*,k} &= \left\{\Omega_i^{h,k} \cap E \,|\, E \in \mathcal{T}_h\right\}, \qquad i \in \{1,2\}, \\ \text{and} \quad \mathcal{T}_+^k &= \mathcal{T}_1^k \cup \mathcal{T}_2^k, \end{aligned} \tag{3.14}$$

with the corresponding edges $\mathcal{E}_1^k, \mathcal{E}_2^k, \mathcal{E}_+^k, \mathcal{E}$ and $\mathcal{E}_\Gamma^k = \mathcal{E}_1^k \cap \mathcal{E}_2^k$.

Although the exact partition defined by the meshes $\mathcal{T}_+^k$ and $\mathcal{T}$ will be significantly influenced by the choice of the merging algorithm, these meshes differ fundamentally in that $\mathcal{T}$ is constant in time while the edges of $\mathcal{T}_+^k$ always resolve the interface, i.e. $\mathcal{E}_\Gamma^k \subset \mathcal{E}_+^k$ in every time step, see figure 3.3 on the facing page.

These cut-cell meshes are employed for the definition of the broken Sobolev spaces

$$\mathbf{H}_+^{m,k} := \left\{ v \in L_2(\Omega^h) \quad | \quad \forall i \in \{1,2\} : \; v|_{\Omega_i^{h,k}} \in H^m(\Omega_i^{h,k}) \right\}, \tag{3.15}$$

Notice that H^m denotes the standard Sobolev spaces of order m and that $\mathbf{H}_+^{m,k}$ is only *broken* along the discrete interface Γ^h. In contrast, the definition of the broken polynomial spaces

$$\begin{aligned} \mathbf{P}_+^{m,k} &:= \left\{ v \in L_2(\Omega_h) \quad | \quad \forall E \in \mathcal{T}_+^k : \; v|_E \in \mathbb{P}^m(E) \right\}, \\ \mathbf{P}^{m} &:= \left\{ v \in L_2(\Omega_h) \quad | \quad \forall E \in \mathcal{T} : \; v|_E \in \mathbb{P}^m(E) \right\}, \end{aligned} \tag{3.16}$$

and

$$\begin{aligned} \mathbf{Q}_+^{m,k} &:= \left\{ v \in L_2(\Omega_h) \quad | \quad \forall E \in \mathcal{T}_+^k : \; v|_E \in \mathbb{Q}^m(E) \right\}, \\ \mathbf{Q}^{m} &:= \left\{ v \in L_2(\Omega_h) \quad | \quad \forall E \in \mathcal{T} : \; v|_E \in \mathbb{Q}^m(E) \right\}. \end{aligned} \tag{3.17}$$

which will be utilized for the final discretization allows for discontinuities across every edge of the respective mesh. Using the multi-index notation $\alpha = (i_1, \ldots, i_d)$ one may give compact definitions of the polynomial space of order m

$$\mathbb{P}^m := \left\{ \sum_\alpha C_\alpha \, x_1^{i_1} \ldots x_d^{i_d} \;\middle|\; C_\alpha \in \mathbb{R}, \; |\alpha|_1 \leq m \right\} \tag{3.18}$$

and the polynomial space of maximum component order m

$$\mathbb{Q}^m := \left\{ \sum_\alpha C_\alpha \, x_1^{i_1} \ldots x_d^{i_d} \;\middle|\; C_\alpha \in \mathbb{R}, \; |\alpha|_\infty \leq m \right\}. \tag{3.19}$$

The principal idea of the UDG method as described in chapter 1 requires the definition of finite element function spaces on meshes containing non-convex cells of almost arbitrary

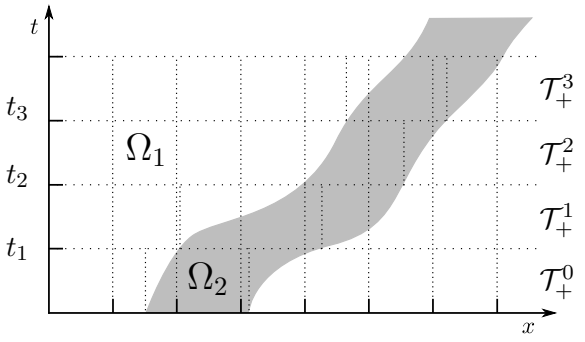


Figure 3.5: The picture shows an exemplary cross-section through the space-time mesh implicitly defined by the sequence of cut-cell meshes $\mathcal{T}_+^k$. On each interval $[t_k, t_{k+1}]$, the mesh $\mathcal{T}_+^k$ is employed for the solution of the Navier-Stokes equations. The local time-integration is performed via method of lines.

shape. While there are many ways to define a DG basis on such cells, here the basis is formulated according to the ideas in [55] and the basis functions on a given cell $E \in \mathcal{T}$ (or $E \in \mathcal{T}_+^k$) are *scaled* with the bounding box

$$B(E) := \left\{ \mathbf{x} \in \mathbb{R}^d \;\middle|\; \forall\, i \in \{1, \dots, d\} : \exists\, \hat{\mathbf{x}} \in E : \hat{x}_i = x_i \right\} \tag{3.20}$$

such that the actual finite element basis of $\mathbf{P}_+^{m,k}$ is given by

$$\Psi(\mathbf{P}_+^{m,k}) := \bigcup_{E \in \mathcal{T}_+^k} \left\{ \phi \in L_2(\Omega^h) \;\middle|\; \phi|_{\Omega^h \setminus E} = 0,\ \phi|_E = m \circ \Phi_{B(E)}^{-1},\ m \in \Psi(\mathbb{P}^m) \right\}, \tag{3.21}$$

where $\Psi(\mathbb{P}^m)$ defines the monomial basis of $\mathbb{P}^m$, see figure 3.4 on page 58. Analogous definitions shall be assumed for $\Psi(\mathbf{P}^m), \Psi(\mathbf{Q}_+^{m,k}), \Psi(\mathbf{Q}^m)$ (compare with notes on the implementation in section 5.1.4 on page 100).

3.4.2 Space-Time Propagation

This section is intended to give an overview of the time propagation scheme *before* the details of the discretizations employed for the individual equations of the two-phase flow system are discussed. The occurring variables $\mathbf{u}_h^k$, p_h^k and ϕ_h^k refer to discrete solutions computed with discontinuous Galerkin methods. It holds that at all times t^k either

$$\mathbf{u}_h^k \in (\mathbf{P}_+^{M_v,k})^d \text{ or } \mathbf{u}_h^k \in (\mathbf{P}^{M_v})^d, \qquad p_h^k \in \mathbf{P}_+^{M_p,k}, \qquad \phi_h^k \in \mathbf{P}^{M_\phi}. \tag{3.22}$$

or

$$\mathbf{u}_h^k \in (\mathbf{Q}_+^{M_v,k})^d \text{ or } \mathbf{u}_h^k \in (\mathbf{Q}^{M_v})^d, \qquad p_h^k \in \mathbf{Q}_+^{M_p,k}, \qquad \phi_h^k \in \mathbf{Q}^{M_\phi}. \tag{3.23}$$

The impact of the respective choices will be discussed in the course of this chapter.

Given a fundamental mesh $\mathcal{T}_h$, a geometry mesh $\mathcal{T}_\varkappa$ and the level set function $\tilde{\phi}_B \in C^0(\Omega_r)$ as well as the corresponding cut-cell mesh $\mathcal{T}$, all constant in time, then based on the definitions in section 3.4.1 on page 57, the principal propagation scheme to solve the system equations (3.5) and (3.6) may be given as follows:

1. At time t_k, based on given approximations ϕ_h^k and $\mathbf{u}_h^k$ of the level set function and velocity field (and the mesh $\mathcal{T}_+^k$ if $\mathbf{u}_h^k \in (\mathbf{P}_+^{M_v,k})^d$ or $\mathbf{u}_h^k \in (\mathbf{Q}_+^{M_v,k})^d$), an approximation ϕ_h^{k+1} of the level set function at time t_{k+1} is computed by solving the level set equation (3.6) on the time interval $[t_k, t_{k+1}]$. This approximation is computed with the spatial DG discretization described in section 3.6 which is based on the stationary mesh $\mathcal{T}$.

2. Compute a continuous approximation (and extension if $\Omega^h \neq \Omega_r$) $\tilde{\phi}_h^{k+1} \in C^0(\Omega_r)$ of the discrete discontinuous level set function ϕ_h^{k+1} as the latter is not suited for the generation of the new meshes (compare section 2.1.2 on page 32 and section 2.4.6 on page 48).

3. For the continuous approximation of the level set function $\tilde{\phi}_h^{k+1} \in C^0(\Omega_r)$, generate the cut-cell mesh $\mathcal{T}_+^{k+1}$ according to the general procedure described in chapter 2.

4. For the given approximation of the velocity field $\mathbf{u}_h^k$ at time t_k, integrate the Navier-Stokes equations (3.5) on the time interval $[t_k, t_{k+1}]$ based on the spatial DG discretization described in section 3.5. The latter is based on the meshes $\mathcal{T}$, $\mathcal{T}_+^{k+1}$ (and $\mathcal{T}_+^k$ if $\mathbf{u}_h^k \in (\mathbf{P}_+^{M_v,k})^d$ or $\mathbf{u}_h^k \in (\mathbf{Q}_+^{M_v,k})^d$). The result is $\mathbf{u}_h^{k+1}$ and p_h^{k+1}, approximations of the velocity and pressure fields at time t_{k+1}. Increment k and go to step 1.

Notice that the computation of the continuous extension $\tilde{\phi}_h^{k+1}$ in step 2 is problematic if $\Omega^h \neq \Omega_r$ and may be avoided, see section 3.7 on page 82. However, in the simplest case it may be realized by a standard Q^1 finite element basis on the geometry mesh $\mathcal{T}_\varkappa$. The discontinuous function ϕ_h^{k+1} represented in a DG basis is then interpolated (or extrapolated) at the corner points of $\mathcal{T}_\varkappa$ and arithmetic averages are computed at such nodes which coincide with corners, edges of faces in the cut-cell mesh $\mathcal{T}$.

Notice that for some of the numerical examples presented in chapter 5, the consecutive solving of the Navier-Stokes Equations and the Level Set Equation was substituted by a more costly Strang-Operator-Splitting [75]. However, due to the generally small time step sizes required by the explicit evaluation of the surface-tension in all examples, this difference is assumed to be negligible and is not discussed any further.

The sequences of meshes $\mathcal{T}_+^k$ and $\mathcal{T}$ effectively describe a space-time grid, an exemplary cross-section of which is shown in figure 3.5 on the preceding page. The illustration reveals the simplicity of the utilized time-propagation scheme. More sophisticated methods based on *arbitrary Lagrangian-Eulerian* mappings (e.g. [12]) should be applicable in combination with the UDG method but have not been analyzed within the scope of this thesis. However, it should be mentioned that unless they are employed to realize a time-stepping scheme which allows for an implicit evaluation of the surface tension, thereby alleviating the time-step size restriction $\Delta t \propto h^{3/2}$, the additional degrees of freedom for

the mapping function (a vector field) may very well deteriorate the ratio of run-time to accuracy.

The numerical methods to solve the sub-problems formulated in steps 4 and 1 are described in section 3.5 and section 3.6 on page 70 respectively. As these sub-problems are defined on constant meshes, standard discontinuous Galerkin methods can be applied but must be adapted to the needs of the UDG cut-cell meshes.

3.5 Spatial Discretization of Flow Problem

The discretization of the two-phase flow problems as presented in this thesis is based on an operator-splitting approach described in section 3.4. It results in Navier-Stokes sub-problems on stationary meshes the solution of which is discussed in this section. The presented discussion is focused on issues of the discretization's consistency and no formal proof of its stability or convergence is provided.

3.5.1 Weak Formulation of Flow Problem

The operator-splitting presented in section 3.4.2 on page 60 requires the solution of a Navier-Stokes sub-problem. According to step 4 in the listing on page 61, this requires the discrete solution equation (3.5) on the constant domains $\Omega_1^{k,h}$ and $\Omega_2^{k,h}$ for the time interval $[t_k, t_{k+1}]$. A suitable weak formulation of this sub-problem can be derived as:

$$
\begin{aligned}
&\forall\, \mathbf{v} \in (\mathbf{H}_+^{1,k})^d : \qquad (3.24)\\
&\Big(\sum_{i\in\{1,2\}} \rho_i\, \partial_t \left(\mathbf{u}, \mathbf{v}\right)_{\Omega_i^{h,k}} + \rho_i \left(\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{v}\right)_{\Omega_i^{h,k}} - \left(\mathcal{D}_i(\mathbf{u}) - \mathbf{I}\,p, \nabla\mathbf{v}\right)_{\Omega_i^{h,k}} \\
&\quad + \left\langle \mathcal{D}_i(\mathbf{u}) - \mathbf{I}\,p, \mathbf{n}\mathbf{v}\right\rangle_{\partial\Omega_i^{h,k}\setminus\Gamma^{h,k}} \Big) \\
&+ \left\langle [\![\mathcal{D}(\mathbf{u})]\!], \mathbf{n}\{\!\{\mathbf{v}\}\!\}\right\rangle_{\Gamma^{h,k}} + \left\langle \{\!\{\mathcal{D}(\mathbf{u})\}\!\}, \mathbf{n}[\![\mathbf{v}]\!]\right\rangle_{\Gamma^{h,k}} - \left\langle [\![p]\!], \mathbf{n}\cdot\{\!\{\mathbf{v}\}\!\}\right\rangle_{\Gamma^{h,k}} - \left\langle \{\!\{p\}\!\}, \mathbf{n}\cdot[\![\mathbf{v}]\!]\right\rangle_{\Gamma^{h,k}} = 0,
\end{aligned}
$$

$$
\begin{aligned}
&\forall q \in L_2^0(\Omega_h) : \\
&\quad \sum_{i\in\{1,2\}} \left(q, \nabla\cdot\mathbf{u}\right)_{\Omega_i^{h,k}} = 0.
\end{aligned}
$$

In this context, the double braces $\{\!\{\cdot\}\!\}$ and $[\![\cdot]\!]$ denote the average and the jump of the traces of a given $\mathbf{H}_+^m$ function across the respective face. To ensure a well-posed problem, the function space for the pressure field enforces the usual normalization property:

$$
L_2^0(\Omega^h) := \left\{ \int_{\Omega^h} q = 0 \;\middle|\; q \in L_2(\Omega^h) \right\}. \qquad (3.25)
$$

To allow a convenient notation, the function

$$
\mathcal{D}(\mathbf{u}) := \begin{cases} \mathcal{D}(\mathbf{u})_1 & \mathbf{x} \in \Omega_1^{h,k} \\ \mathcal{D}(\mathbf{u})_2 & \mathbf{x} \in \Omega_2^{h,k} \end{cases} \qquad (3.26)
$$

was introduced. The forms $(\cdot,\cdot)_E$ and $\langle\cdot,\cdot\rangle_e$ denote the standard scalar products in L_2 on a cell E and a face e in the respective mesh (different braces were used to help the reader distinguishing volume and face integrals). The weak formulation (3.24) can be derived using integration by parts and the algebraic identity

$$[\![a \cdot b]\!] = [\![a]\!]\{\!\{b\}\!\} + [\![b]\!]\{\!\{a\}\!\}. \tag{3.27}$$

Notice that (3.24) essentially describes two separate Navier-Stokes problems on the domains $\Omega_1^{h,k}$ and $\Omega_2^{h,k}$ coupled via the interface terms on $\Gamma^{h,k}$. Comparing these interface terms to (3.12) reveals

$$\langle [\![\mathcal{D}(\mathbf{u})]\!], \mathbf{n}\{\!\{\mathbf{v}\}\!\}\rangle_{\Gamma^{h,k}} - \langle [\![p]\!], \mathbf{n}\cdot\{\!\{\mathbf{v}\}\!\}\rangle_{\Gamma^{h,k}} = \langle \mathbf{f}_\Gamma, \mathbf{n}\{\!\{\mathbf{v}\}\!\}\rangle_{\Gamma^{h,k}} \tag{3.28}$$

and

$$\langle \{\!\{q\}\!\}, \mathbf{n}\cdot[\![\mathbf{u}]\!]\rangle_{\Gamma^{h,k}} = 0. \tag{3.29}$$

Therefore, the weak form (3.24) is equivalent to

$$\forall \mathbf{v} \in (\mathbf{H}_+^{1,k})^d : \tag{3.30}$$
$$\Big(\sum_{i\in\{1,2\}} \rho_i\, \partial_t\, (\mathbf{u},\mathbf{v})_{\Omega_i^{h,k}} + \rho_i\, (\mathbf{u}\cdot\nabla\mathbf{u},\mathbf{v})_{\Omega_i^{h,k}} - (\mathcal{D}_i(\mathbf{u}) - \mathbf{I}p, \nabla\mathbf{v})_{\Omega_i^{h,k}} + \langle \mathcal{D}_i(\mathbf{u}) - \mathbf{I}p, \mathbf{n}\mathbf{v}\rangle_{\partial\Omega_i^{h,k}\setminus\Gamma^{h,k}} \Big)$$
$$+ \langle \mathbf{f}_\Gamma, \mathbf{n}\{\!\{\mathbf{v}\}\!\}\rangle_{\Gamma^{h,k}} + \langle \{\!\{\mathcal{D}(\mathbf{u})\}\!\}, \mathbf{n}[\![\mathbf{v}]\!]\rangle_{\Gamma^{h,k}} - \langle \{\!\{p\}\!\}, \mathbf{n}\cdot[\![\mathbf{v}]\!]\rangle_{\Gamma^{h,k}} = 0,$$

$$\forall q \in L_2^0(\Omega_h) :$$
$$\sum_{i\in\{1,2\}} (q, \nabla\cdot\mathbf{u})_{\Omega_i^{h,k}} - \langle \{\!\{q\}\!\}, \mathbf{n}\cdot[\![\mathbf{u}]\!]\rangle_{\Gamma^{h,k}} = 0.$$

3.5.2 Interior Penalty Discretization of Flow Problem

When ignoring all the coupling terms on $\Gamma^{h,k}$, (3.30) describes two uncoupled incompressible Navier-Stokes problems with homogeneous Neumann boundaries on $\Gamma^{h,k}$. Each of these problems may thus individually be discretized by a standard discontinuous Galerkin method for incompressible flow on $\Omega_1^{h,k}$ and $\Omega_2^{h,k}$ respectively. In this thesis, the *interior penalty discontinuous Galerkin* (IPDG) method for incompressible flow [23, 46, 49] is utilized as a representative of robust discontinuous Galerkin methods with purely local couplings which is suitable for non-mixed formulations. Notice however that both the UDG method and its application to two-phase flow is not restricted to this particular choice among the various available schemes. Different discontinuous Galerkin approaches with the capability of handling a sharp interface at the two-phase boundary have been published by other authors, see e.g. [12] for an ALE approach and [28] for a space-time DG approach.

The term $\langle \{\!\{\mathcal{D}(\mathbf{u})\}\!\}, \mathbf{n}[\![\mathbf{v}]\!]\rangle_{\Gamma^{h,k}}$ in (3.30) describes the viscous flux across the two-phase interface $\Gamma^{h,k}$. In the standard terminology of discontinuous Galerkin methods for diffusion problems, this corresponds to an unstable numerical flux function with *central fluxes*,

see [50, 76] [1]. In order to stabilize the flux of momentum due to viscous stress across the interface, this term is replaced by the standard numerical flux of the IPDG method

$$\langle \{\!\{\mathcal{D}(\mathbf{u})\}\!\}, \mathbf{n}[\![\mathbf{v}]\!] \rangle_{\Gamma^{h,k}} - \epsilon \langle \{\!\{\mathcal{D}(\mathbf{v})\}\!\}, \mathbf{n}[\![\mathbf{u}]\!] \rangle_{\Gamma^{h,k}} + \frac{\sigma_{IP}}{h_e} \langle [\![\mathbf{u}]\!], [\![\mathbf{v}]\!] \rangle_{\Gamma^{h,k}} \tag{3.31}$$

as applied at the interior faces of $\mathcal{T}_1^k$ and $\mathcal{T}_2^k$. All examples presented in this thesis generally employ the non-symmetric scheme of the IPDG method with $\epsilon = 1$ in (3.36). The penalty term was always scaled with a geometric factor based on the volumes of the adjacent cells E and F according to

$$h_e := \left(\frac{1}{|E|} + \frac{1}{|F|} \right)^{(1-d)/d} \tag{3.32}$$

and with the maximum viscosity

$$\sigma_{IP} = \alpha \max(\mu_1, \mu_2) \tag{3.33}$$

where α was varied in the range $1 - 100$ depending on the given problem setup.

According to (3.12) it holds that the velocity is continous across the interface $\Gamma^{h,k}$ and thus the two additional terms in (3.31) do not compromise the consistency of the resulting discretization as both are proportional to the jump $[\![\mathbf{u}]\!]$ of the velocity across the interface. Hence, utilizing the multi-linear forms

$$m(\mathbf{u}, \mathbf{v}) = (\rho\, \mathbf{u}, \mathbf{v})_{\Omega^h}\,, \tag{3.34}$$

$$c^k(\mathbf{u}, \mathbf{v}, \mathbf{w}) = \sum_{E \in \mathcal{T}_+^k} (\rho\, \mathbf{w} \cdot \nabla \mathbf{u}, \mathbf{v})_E\,, \tag{3.35}$$

$$\begin{aligned} a^k(\mathbf{u}, \mathbf{v}) = & - \sum_{E \in \mathcal{T}_+^k} (\mathcal{D}(\mathbf{u}), \nabla \mathbf{v})_E\,, \\ & + \sum_{e \in \mathcal{E}_+^k} \langle \{\!\{\mathcal{D}(\mathbf{u})\}\!\}, \mathbf{n}[\![\mathbf{v}]\!] \rangle_e - \epsilon \langle \{\!\{\mathcal{D}(\mathbf{v})\}\!\}, \mathbf{n}[\![\mathbf{u}]\!] \rangle_e + \frac{\sigma_{IP}}{h_e} \langle [\![\mathbf{u}]\!], [\![\mathbf{v}]\!] \rangle_e\,, \end{aligned} \tag{3.36}$$

$$b^k(\mathbf{v}, p) = \sum_{E \in \mathcal{T}_+^k} (p, \nabla \cdot \mathbf{v})_E - \sum_{e \in \mathcal{E}_+^k} \langle \{\!\{p\}\!\}, \mathbf{n} \cdot [\![\mathbf{v}]\!] \rangle_e\,, \tag{3.37}$$

$$f^k(\mathbf{v}) = \sum_{e \in \mathcal{E}_\Gamma^{h,k}} \langle \mathbf{f}_\Gamma, \mathbf{n}\{\!\{\mathbf{v}\}\!\} \rangle_e \tag{3.38}$$

and choosing appropriate inf-sup stable finite element spaces [23, 49]

$$\mathbf{X}^k := \left(\mathbf{P}_+^{M_v,k} \right)^d \qquad \text{and} \qquad Y^k := \mathbf{P}_+^{M_p,k}, \tag{3.39}$$

with $M_p = M_v - 1$, a consistent interior penalty DG discretization of the two-phase flow problem as given in (3.30) may be written as:

1 Notice that this is not equivalent to the numerical flux function in the method of Bassi and Rebay [56, 77] which is often referred to as *the central flux* and describes a stable flux, though entailing a sub-optimal convergence order

Find velocity field $\boldsymbol{u}_h$ and pressure field p_h such that

$$\begin{aligned} \forall \mathbf{v}_h \in \mathbf{X}^k: \quad & \partial_t m(\mathbf{u}_h, \mathbf{v}_h) + c^k(\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h) \\ & + a^k(\mathbf{u}_h, \mathbf{v}_h) + b^k(\mathbf{v}_h, p_h) + f^k(\mathbf{v}_h) && = 0, \\ \forall q_h \in Y^k: \quad & b^k(\mathbf{u}_h, q_h) && = 0. \end{aligned} \tag{3.40}$$

To allow for a compact notation, in the definitions (3.34) the function ρ was introduced in analogy to (3.26). In this notation, the problem (3.40) reduces to a *quasi-inhomogeneous* Navier-Stokes problem with a standard interior penalty DG discretitzation and a source term on the interface $\Gamma^{h,k}$.

For each face $e \in \mathcal{E}_+^k$ which is situated on the domain boundary $\partial\Omega^h$, the jumps $[\![\cdot]\!]$ and averages $\{\!\{\cdot\}\!\}$ occurring in (3.40) are not well defined and must be explicitly chosen to realize appropriate boundary conditions. Generally, for a cell $E \in \mathcal{T}_+^k$ adjacent to the boundary face e, the jumps and averages of the test functions are defined equal to the traces of these functions on E, i.e.

$$[\![\mathbf{v}_h]\!] := \{\!\{\mathbf{v}_h\}\!\} := \mathbf{v}_h\big|_E \quad \text{and} \quad [\![q_h]\!] := \{\!\{q_h\}\!\} := q_h\big|_E. \tag{3.41}$$

For Dirichlet boundaries with boundary function $\mathbf{u}^D \in (C^0(\partial\Omega^h))^d$, the natural choice for the remaining terms is given by

$$[\![\mathbf{u}_h]\!] = \mathbf{u}_h\big|_E - \mathbf{u}^D \quad \text{and} \quad \{\!\{\mathcal{D}(\mathbf{u}_h) - \mathbf{I}\,p\}\!\} \cdot \mathbf{n} = (\mathcal{D}(\mathbf{u}_h) - \mathbf{I}\,p)\big|_E \cdot \mathbf{n}. \tag{3.42}$$

In case of a Neumann boundary with a flux function $\mathbf{f}^N \in (C^0(\partial\Omega^h))^d$, the terms were chosen according to

$$[\![\mathbf{u}_h]\!] = 0 \quad \text{and} \quad \{\!\{\mathcal{D}(\mathbf{u}_h) - \mathbf{I}\,p\}\!\} \cdot \mathbf{n} = \mathbf{f}^N \tag{3.43}$$

For the Navier-slip boundary condition (3.13), only the velocity component normal to the boundary is known and the interior penalty term

$$\frac{\sigma_{IP}}{h_e} \langle [\![\mathbf{u}_h]\!], [\![\mathbf{v}_h]\!] \rangle_e \quad \text{was replaced by} \quad \frac{\sigma_{IP}}{h_e} \langle \mathbf{n} \cdot [\![\mathbf{u}_h]\!], \mathbf{n} \cdot [\![\mathbf{v}_h]\!] \rangle_e . \tag{3.44}$$

The remaining terms were chosen as

$$[\![\mathbf{u}_h]\!] = \mathbf{u}_h\big|_e \quad \text{and} \quad \{\!\{\mathcal{D}(\mathbf{u}_h) - \mathbf{I}\,p\}\!\} \cdot \mathbf{n} = (\mathbf{I} - \mathbf{n}\mathbf{n})\,(\mathcal{D}(\mathbf{u}_h) - \mathbf{I}\,p + \beta_N \mathbf{u}_h \mathbf{n})\big|_E \cdot \mathbf{n}. \tag{3.45}$$

Notice that full-slip boundary conditions can be achieved by setting $\beta_N = 0$. For the computational examples presented in chapter 5, these choices provided stable results.

The time integration required for a full discretization of (3.40) can be realized with standard methods, e.g. the method of lines. The one-step schemes utilized for this thesis include the Implicit Euler and the Crank-Nicholson method. After time integration, a choice of $(\mathbf{u}_h, p_h)\big|_{t=t_k} \in (\mathbf{X}^k, Y^k)$ defines a Galerkin-type sub-problem for each stage of the one-step methods

However, the actual choice of $(\mathbf{X}^k, Y^k)$ may introduce further complications with regard to the stability and the implementation of the method. In the following sections 3.5.3 and 3.5.4, the implications resulting from the respective choices of the finite element spaces $(\mathbf{X}^k, Y^k)$ as $((\mathbf{P}_+^{M_v,k})^d, \mathbf{P}_+^{M_p,k})$ or $((\mathbf{P}^{M_v})^d, \mathbf{P}_+^{M_p,k})$ will be discussed.

3.5.3 Discretization of Flow Problem with Hybrid Meshes

The choice of the discrete finite element space $\mathbf{X}^k := (\mathbf{P}_+^{M_v,k})^d$ for the velocity which generally allows the representation of jumps across the interface $\Gamma^{h,k}$ (unlike $\mathbf{X}^k := (\mathbf{P}^{M_v})^d$) may seem rather restrictive considering that the velocity is required by the interface conditions (3.12) to be continuous across the interface. However, while the velocity field itself is globally continuous, its derivatives will in general involve jumps across $\Gamma(t)$.

Nevertheless, the majority of publications on two-phase flow are based on discretizations which employ either a fully continuous finite element basis [10, 16, 34, 78] or utilize a finite-differences approach assuming additional global regularity [1, 2, 79, 80]. The rather recent attempts to apply extended finite elements and special enrichment-spaces were also restricted to the representation of the pressure jump across the interface [26, 27, 81]. The presented results suggest that these approaches, which implicitly or explicitly regularize the viscosity of the two immiscible fluids across the interface, can be applied to a wide range of two-phase flow regimes.

The popularity of such schemes which impose additional regularity on the velocity field is likely a result of the simplicity of their derivation and implementation. The discretization of the time-derivative of the velocity field is severely more complex if its finite element basis is time-dependent. However, this is mandatory if the discrete velocity solution is supposed to capture the interface conditions accurately. Arbitrary-Lagrangian-Eulerian methods have been successfully applied in flow simulations with moving meshes tracking the interface [12] and space-time discontinuous Galerkin methods were shown to work in geometries defined by time-dependent cut-cell meshes [28]. The latter publication is especially interesting due to its methodological proximity to the UDG approach. The complexity of the time discretization of both methods is severe and shared by other numerical schemes that depend on accurate tracking of interfaces which entail local irregularities of the solution variable, among them the whole range of methods for fluid-structure interaction problems [82].

The multi-domain UDG approach entails an unusually high flexibility with regard to numerical integration of complex domains enabling an unusually direct and simplistic approach for the time discretization, as was described in section 3.4 on page 57. However, even this approach involves additional complexity that may not be justified for a given simulation setup. Hence, the choice $(\mathbf{X}^k, Y^k) := ((\mathbf{P}^M)^d, \mathbf{P}_+^{M-1,k})$ is interesting as it allows a representation of the jump in the discrete pressure field but still maintains a time-independent finite element basis for the discrete velocity field. Hence, standard time-stepping schemes for the time discretization of equation (3.40) can be directly applied - in contrast to the more complicated scenario presented in section 3.5.4 on the facing page.

The implementation of this choice, however, is complicated by circumstances of initially mundane appearance: The merging process as described in section 2.2.4 on page 36 will in general not produce matching cut-cell meshes $\mathcal{T}_+^k$ and $\mathcal{T}$, i.e.

$$\mathcal{E} \not\subset \mathcal{E}_+^k, \quad \mathcal{E}_+^k \not\subset \mathcal{E} \tag{3.46}$$

as can be readily seen in figure 3.3 on page 58. Nonetheless, since the merging process is fundamental for stability and/or efficiency of the numerical simulation, so must this

problem be considered fundamental in its nature. The key ingredient to the solution of this problem resides in the common sub-triangulation $\mathcal{S}$ utilized for the construction of both $\mathcal{T}_+^k$ and $\mathcal{T}$. Hence, for every cell E in either $\mathcal{T}_+^k$ or $\mathcal{T}$, there is a set of cell-parts $\{\mathfrak{e}_1, \ldots, \mathfrak{e}_N\}$ in $\mathcal{S}$ such that

$$\overline{E} = \overline{\bigcup} \{\mathfrak{e}_1, \ldots, \mathfrak{e}_N\}. \tag{3.47}$$

Therefore, a *hybrid* mesh $\mathcal{T}_H^k$ may be defined according to

$$\mathcal{T}_H^k := \left\{ E \cap F \mid E \in \mathcal{T}_+^k,\, F \in \mathcal{T} \right\} \tag{3.48}$$

and utilized for the definition of the multi-linear forms $m_H(\cdot,\cdot), c_H(\cdot,\cdot,\cdot), a_H(\cdot,\cdot), b_H(\cdot,\cdot)$ equivalent to (3.34) but with sums over $\mathcal{T}_H^k$ instead of $\mathcal{T}_+^k$. As the numerical integration is anyway performed on the sub-triangulation $\mathcal{S}$, the additional computational cost of an integration on such a hybrid grid is limited to the additional computations resulting from the possibly different shape functions utilized by the test and the trial space basis on a given cell in $\mathcal{T}_H^k$ and is thus restricted to the computation of $b_H(\mathbf{v}, p)$.

A more subtle shortcoming of the hybrid grid approach is entailed in the unproven inf-sub stability of the function space pair $(\mathbf{X}^k, Y^k) := ((\mathbf{P}^M)^d, \mathbf{P}_+^{M-1,k})$. In the context of this thesis, no attempt is made to prove this condition or to establish an alternative stability estimate. However, it should be pointed out that the typical methods utilized in the proofs of inf-sub stability [83] do not appear promising with regard to the almost arbitrary cell geometries as may occur in the merged UDG cut-cell meshes. Nevertheless, the numerical experiments as described in chapter 5 on page 99 show that the hybrid grid method may be applied to a vast range of applications for which it provides stable and accurate results.

As the finite element space $(\mathbf{P}^M)^d$ for the velocity field is unable to resolve the jump in the velocity derivatives, there is no obvious benefit in maintaining the piecewise constant approximation of the viscosity. Therefore, for this hybrid mesh approach, the viscosity field was chosen as

$$\mu(\mathbf{x}) := \mu_1\, \Theta^{\epsilon_\mu}(\phi(\mathbf{x})) + \mu_2\, [1 - \Theta^{\epsilon_\mu}(\phi(\mathbf{x}))] \tag{3.49}$$

with the regularized Heaviside function

$$\Theta^{\epsilon}(x) := \begin{cases} 0, & x < -\epsilon, \\ \frac{1}{2} + \frac{x}{2\epsilon} + \frac{1}{2\pi} \sin(\frac{\pi x}{\epsilon}), & -\epsilon \le x < \epsilon, \\ 1, & \epsilon \le x. \end{cases} \tag{3.50}$$

While no systematic tests were performed, the general sensitivity of the flow simulation on the parameter ϵ_μ was low as long as it was chosen significantly smaller than the mesh resolution h. For all simulations presented in this thesis, the value $\epsilon_\mu = 0.05\, h$ was employed.

3.5.4 Discretization of Flow Problem with Aligned Meshes

Considering both the practical difficulties and the theoretical uncertainties of the choice $(\mathbf{X}^k, Y^k) := ((\mathbf{P}^M)^d, \mathbf{P}_+^{M-1,k})$, it might appear both simpler and safer to base a dis-

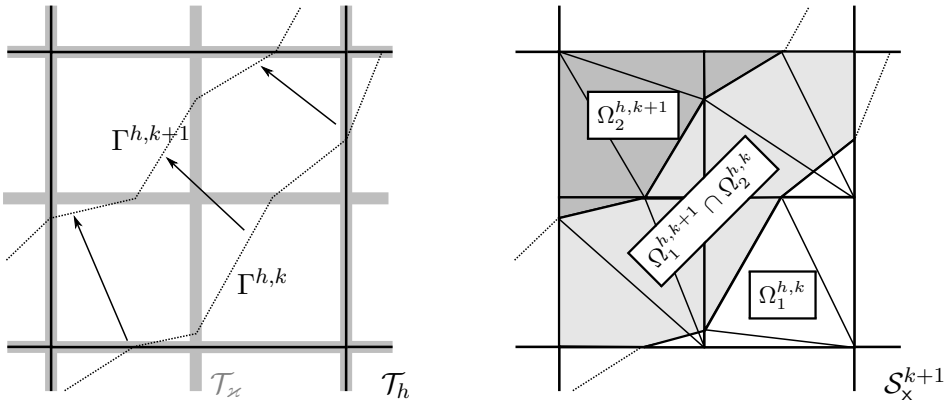


Figure 3.6: This figure illustrates the *extended* sub-triangulation $\mathcal{S}_\varkappa^{k+1}$ which needs to be constructed to allow a proper numerical integration of the *mixed* terms in 3.51, i.e. scalar products of functions in $\mathbf{P}^{M,k+1}$ with functions in $\mathbf{P}^{M,k}$. An exact integration of these terms is only possible if the extended sub-triangulation $\mathcal{S}_\varkappa^{k+1}$ resolves the interface of the previous time step $\Gamma^{h,k}$.

cretization on the finite element spaces $((\mathbf{P}_+^{M,k})^d, \mathbf{P}_+^{M-1,k})$. Sadly, this approach entails difficulties with regard to the time integration.

The problem can be understood by taking a look at the full discretization resulting from the application of a simple One-Step-θ scheme to (3.40):

Find velocity field $\boldsymbol{u}_h^{k+1} \in \boldsymbol{X}^{k+1}$ *and pressure field* $p_h^{k+1} \in Y^{k+1}$ *at time* t_{k+1} *such that for a given initial velocity field* $\boldsymbol{u}_h^k$ *at time* t_k *with* $\Delta t_k = t_{k+1} - t_k$ *it holds that*

$$\begin{aligned}
\forall \mathbf{v}_h \in \mathbf{X}^{k+1}: \quad & \frac{1}{\Delta t_k}\Big(m(\mathbf{u}_h^{k+1}, \mathbf{v}_h) - m(\mathbf{u}_h^k, \mathbf{v}_h)\Big) + b^{k+1}(\mathbf{v}_h, p_h^{k+1}) + f^{k+1}(\mathbf{v}_h) \\
& + \theta \quad \Big(c^{k+1}(\mathbf{u}_h^{k+1}, \mathbf{u}_h^{k+1}, \mathbf{v}_h) + a^{k+1}(\mathbf{u}_h^{k+1}, \mathbf{v}_h)\Big) \\
& + (1-\theta) \quad \Big(c^{k+1}(\mathbf{u}_h^{k}, \mathbf{u}_h^{k}, \mathbf{v}_h) + a^{k+1}(\mathbf{u}_h^{k}, \mathbf{v}_h)\Big) \qquad = 0, \\
\forall q_h \in Y^{k+1}: \quad & b^{k+1}(\mathbf{u}_h^{k+1}, q_h) \qquad = 0.
\end{aligned} \tag{3.51}$$

Notice that even in the case $\theta = 1$, which corresponds to an implicit Euler method, the computation of the term $m(\mathbf{u}_h^k, \mathbf{v}_h)$ is non-trivial as it involves the *mixed* scalar product of a function in $\mathbf{P}_+^{M,k+1}$ with a function in $\mathbf{P}_+^{M,k}$ over Ω^h. This would require a set of cell-parts $\{\mathfrak{e}_1, \ldots, \mathfrak{e}_N\}$ such that for any given cells $E \in \mathcal{T}^{k+1}$ and $F \in \mathcal{T}^k$ it holds that

$$\overline{E} \cap \overline{F} = \bigcup \{\overline{\mathfrak{e}}_1, \ldots, \overline{\mathfrak{e}}_N\}. \tag{3.52}$$

In general however, such a set is not available, neither in the sub-triangulation $\mathcal{S}^{k+1}$ nor in $\mathcal{S}^k$ (not even in $\mathcal{S}^{k+1} \cup \mathcal{S}^k$) due to the fact that the sub-triangulations at different time-steps are constructed completely independent of each other.

It is seductive to argue that this problem may just be ignored as long as the time step size is small enough, i.e. the interface $\Gamma^{h,k+1}$ is always *close enough* to $\Gamma^{h,k}$ such that the integral over the small band in between is negligible. Revisiting equation (3.51) and considering the additional *mixed* scalar products (the coefficients of $(1-\theta)$) which would

be relevant in case of $\theta < 1$, it becomes apparent that this would introduce an error into the momentum balance close to the most important source of the system's dynamics. It is also not obvious how such an approach could be implemented in a robust way, as there is no clear association of cells in $\mathcal{T}_+^k$ to those in $\mathcal{T}_+^{k+1}$, i.e. cells may be created or disappear and there is no efficient way to determine for a point $\mathbf{x}$ in a cell E in $\mathcal{T}_+^{k+1}$ the cell F in $\mathcal{T}_+^k$ which contains it.

With regard to simplicity of implementation, the next best approach would be to project the solution $\mathbf{u}_h^k \in (\mathbf{P}_+^{M,k})^d$ onto a temporary solution $\hat{\mathbf{u}}_h^k \in (\mathbf{P}_+^{M,k+1})^d$ and then utilize $\hat{\mathbf{u}}_h^k$ for the evaluations of the solution at $t = t_{k+1}$ in (3.51). However, it is not clear how to realize such a projection in a way that preserves the scheme's advantage of accurately representing the jump in the gradients of the velocity field.

Another very intuitive, if somewhat *brute-force*, way to compute the *mixed* scalar products accurately was implemented for the numerical computations presented in this thesis and published in [35]. It is realized by the introduction of an additional level set function representing the *old* interface $\Gamma^{h,k}$. While $\mathcal{S}^{k+1}$ was constructed with regard to the level set functions

$$\phi_1 := \tilde{\phi}_B \qquad \text{and} \qquad \phi_2 := \tilde{\phi}_h^{k+1},$$

the *extended* sub-triangulation $\mathcal{S}_\times^{k+1}$ is constructed with regard to the three level set functions

$$\phi_1 := \tilde{\phi}_B \qquad \text{and} \qquad \phi_2 := \tilde{\phi}_h^{k+1}, \qquad \text{and} \qquad \phi_3 := \tilde{\phi}_h^{k}. \tag{3.53}$$

When redefining $\mathcal{T}_+^{k+1}$ and $\mathcal{T}_+^k$ with regard to $\mathcal{S}_\times^{k+1}$, a set of cell-parts resolving any intersections of the cells in both meshes according to (3.52) can always be found thus enabling a convenient numerical integration of all terms in (3.51), see figure 3.6 on the facing page.

This, however, is preconditioned on the ability of $\mathcal{S}_\times^{k+1}$ to actually resolve the interface $\Gamma^{h,k}$ of the last time step, i.e. it is not overlapped by some cell-part in $\mathcal{S}_\times^{k+1}$ such that

$$\forall \mathfrak{e} \in \mathcal{S}_\times^{k+1} : \qquad \mathfrak{e} \cap \Omega_1^{h,k} = \emptyset \quad \text{or} \quad \mathfrak{e} \cap \Omega_2^{h,k} = \emptyset. \tag{3.54}$$

This is a non-trivial requirement, as the recursive construction of the sub-triangulations according to (2.14) does in general depend on their order. Therefore, a sub-triangulation $\hat{\mathcal{S}}_\times^{k+1}$ created for

$$\phi_1 := \tilde{\phi}_B \qquad \text{and} \qquad \phi_2 := \tilde{\phi}_h^{k}. \qquad \text{and} \qquad \phi_3 := \tilde{\phi}_h^{k+1}, \tag{3.55}$$

will in general not be identical to $\mathcal{S}_\times^{k+1}$, as was already mentioned in section 2.2.3 on page 35. This will introduce an integration error with a magnitude in the order of the total domain approximation error caused by the piecewise linear approximation of the domain boundary. While this may seem benign, it would entail the problem of efficiently finding a cell in $\mathcal{T}^k$ that almost matches a cell in $\mathcal{T}^{k+1}$.

Notice that the recursive sub-triangulation construction scheme (2.32) presented in section 2.4.7 on page 50 would due to the utilized linearization guarantee (3.54) as long as no regularization (according to section 2.4.7) is required. However, the numerical examples based on the space-pair $((\mathbf{P}_+^{M,k})^d, \mathbf{P}_+^{M-1,k})$ which are presented in this thesis

are restricted to the two-dimensional case where such geometric guarantees can be realized much more easily.

This approach increases the computational effort for the numerical integration close to the interface where more cell-parts have to be created. However, many of the additional parts are bound to be small as part of them resolves the domain in between $\Gamma^{h,k}$ and $\Gamma^{h,k+1}$ and a local reduction of the quadrature rules might be a viable way to improve efficiency. However, the benefit of such optimizations was not analyzed systematically and the examples presented in this thesis do not utilizes such optimizations.

3.6 Discretization of the Level Set Problem

The classical level set equation

$$\partial_t \phi + \mathbf{u} \cdot \nabla \phi = 0, \quad \forall \mathbf{x} \in \Omega \tag{3.56}$$

describes a hyperbolic problem which is linear as long as $\mathbf{u}$ does not depend on ϕ. However, the governing equations (3.5) and (3.6) of two-phase flow imply a coupling between the velocity field and the level set function mediated by the domain geometries of Ω_1 and Ω_2. In all setups for which the surface tension or any transmission effects due to the jumps in the density and viscosity fields dominate the system dynamics, this coupling will be of a very strong nature which has to be accounted for by the numerical method. However, the scope of this thesis is restricted to such systems for which the coupling is moderate enough to allow an operator splitting of the Navier-Stokes equations (3.5) and the level set equation (3.6). In this case, for the purposes of integrating (3.56) over $[t_k, t_{k+1}]$, i.e. solving sub-problem in step 1 of the operator-splitting presented in section 3.4.2 on page 60, the velocity field $\mathbf{u}$ may be considered independent of ϕ and the equation itself may thus be treated as a linear hyperbolic equation.

3.6.1 Standard Level Set Method (SLS)

As the subdomain geometries are determined by the zero level set of the level set function only, its non-zero initial values at time $t = t_0$ may be chosen to suit the needs of the numerical scheme. The classic approach as presented in [60] is based on ϕ being initialized as the distance function of the initial interface $\Gamma(t = t_0)$. As both high and low gradients in the level set function may result in severe numerical instabilities, a distance function is quite desirable as it guarantees that $|\nabla \phi| = 1$.

Sadly, the propagation of the level set function as described by (3.56) does in general not conserve the distance function property. However, this property can be enforced by reconstructing or modifying the level set function in the course of its propagation such that the distance function property is regained without modifying the zero level set. For a given interface $\Gamma(t)$, this *reinitialization* can be achieved by solving the *Eikonal Equation*:

$$|\nabla \phi| = 1, \qquad \forall \mathbf{x} \in \Gamma(t) : \phi(\mathbf{x}) = 0. \tag{3.57}$$

Efficient numerical schemes for this equation have been the subject of intensive research. Although (3.57) can be expressed as an equivalent pseudo-time-dependent hyperbolic

problem

$$\partial_\tau \phi = \text{sign}(\phi)\,(|\nabla\phi| - 1) \tag{3.58}$$

for which stable numerical schemes have be devised, comprehensive comparative studies as given in [84, 85] seem to favor the so-called direct approaches, e.g. the Fast Marching Algorithm, which utilize the particular properties of the Eikonal equation to construct its solution in an incremental fashion, following the characteristics originating from the interface. Alas, all these methods entail erroneous shifts of the zero level set resulting in an unintended displacement of the interface, thus introducing errors in the fluid's volume balance. However, due to the continuous interest in this subject, more complex schemes have been developed to reduce these undesirable if unavoidable erroneous shifts or at least enforce local volume conservation, see e.g [1, 86, 87, 88] and the papers cited there.

Even if an optimal numerical scheme for the reinitialization of the level set function is employed, a discretization for (3.56) which realizes true volume conservation is not easily devised. Although, for a solenoidal velocity field $\mathbf{u}$, this equation is equivalent to the conservative transport equation

$$\partial_t \phi + \nabla\,(\mathbf{u} \cdot \phi) = 0, \quad \forall \mathbf{x} \in \Omega, \tag{3.59}$$

the classical conservative discretization schemes resulting from finite-volume or higher-order discontinuous Galerkin approaches will only conserve the integral of the transported field variable ϕ such that

$$\int_\Omega \phi(\mathbf{x})\,dx \quad \text{is conserved, while the phase volume} \quad \int_\Omega \Theta(\phi(\mathbf{x}))\,dx \tag{3.60}$$

will in general be subject to numerical errors (here Θ denotes the Heaviside function). Relevant ideas on how to best solve such non-conservative linear hyperbolic equations like (3.56) were recently given in [89, 90]. However, to the author's knowledge, a method which obtains *true* volume-conservation without applying any kind of post-processing to the solution field (typically introduced during the aforementioned reinitialization step) is yet to be discovered.

The *volume-of-fluid* method (VOF), a low-order scheme realizing a volume conserving front tracking method has already been proposed in [91]. It is based on a *phase-field*, a scalar indicator function with a range of $[0, 1]$. For the VOF method, this value is chosen piecewise constant within each cell and describes the ratio of the two fluid's volume fractions which fill the cell's interior. For this low-order approximation, it is possible to devise numerical schemes which enforce volume conservation even for non-conservative fluxes utilizing local redistribution algorithms. In [1], such a VOF method was coupled to a finite-difference level set approach thereby improving volume conservation by many orders of magnitude. This improvement, on the other hand, requires strong modifications performed by the post-processing steps which may compromise the order of convergence and, due to the nature of the redistribution schemes, typically impede a comprehensive numerical analysis of the full scheme.

3.6.2 Conservative Level Set Method (CLS)

A generalization of this approach was proposed in [92, 93]. Instead of coupling the level set function to a piecewise constant phase-field φ, the latter is directly represented by a higher-order finite element basis and the level set function is simply set to $\phi := \varphi - 0.5$, i.e. the $\varphi = 0.5$ level set of the phase-field function is interpreted as the two fluid's interface. Therefore, at time t_0, the phase-field must be initialized appropriately with a regularized Heaviside function $\varphi(t = t_0) := \varphi_0$ which represents the phase distribution and includes a smooth transition band of width Δ_φ chosen as a multiple of the mesh size h.

When considering the conservative form of the level set equation (3.59) formulated for the hybrid phase-field / level set function φ as

$$\partial_t \varphi + \nabla \left(\mathbf{u} \cdot \varphi \right) = 0, \quad \forall \mathbf{x} \in \Omega, \tag{3.61}$$

it is evident that this approach corresponds simply to the standard level set method with a rather inappropriate initial solution whose gradients are zero everywhere but in the direct vicinity of the interface. In order to ensure volume conservation, the introduction of phase-field values outside of the range $[0, 1]$ must be prohibited and the transition area Δ_φ must be kept constant. The authors of [92, 93] proposed the *conservative level set method* (CLS) which meets these requirements by introducing recompression terms into the equation, effectively modifying (3.61) such that

$$\partial_t \varphi + \nabla \left(\mathbf{u} \cdot \varphi \right) + \alpha \left(\nabla \cdot (\varphi(1 - \varphi)\mathbf{n}_\varphi) - \varepsilon_\varphi \nabla \cdot \left(\nabla \varphi \cdot \mathbf{n}_\varphi \right) \mathbf{n}_\varphi \right) = 0, \quad \forall \mathbf{x} \in \Omega, \tag{3.62}$$

with

$$\mathbf{n}_\varphi := \frac{\nabla \varphi}{|\nabla \varphi|}. \tag{3.63}$$

The generic coefficient α has to be chosen problem-dependent and must ensure, that the recompression is sufficient to maintain an approximately constant transition width Δ_φ and suppress any spurious values $\varphi > 1$ and $\varphi < 0$. In the absence of robust heuristics for the choice of α, the equation (3.62) may be solved in an operator splitting approach by separately solving (3.61) and the pure recompression equation

$$\partial_t \varphi + \nabla \cdot (\varphi(1 - \varphi)\mathbf{n}_\varphi) - \varepsilon_\varphi \nabla \cdot \left(\nabla \varphi \cdot \mathbf{n}_\varphi \right) \mathbf{n}_\varphi = 0, \quad \forall \mathbf{x} \in \Omega. \tag{3.64}$$

As both the convective and diffusive fluxes are directed along the streamlines of the vector field $\mathbf{n}_\varphi$, it is noteworthy that a stationary solution for the one-dimensional case is given by

$$\varphi_c(\mathbf{x}) := \frac{e^{x/\varepsilon_\varphi}}{1 + e^{x/\varepsilon_\varphi}} \tag{3.65}$$

which adheres to

$$\lim_{x \to -\infty} \varphi_c(x) = 0 \quad \text{and} \quad \lim_{x \to \infty} \varphi_c(x) = 1. \tag{3.66}$$

This provides the useful estimate

$$\Delta_\varphi \approx 6 \cdot \varepsilon_\varphi, \tag{3.67}$$

which relates the artificial diffusion parameter ε_φ to the transition width Δ_φ. Theoretically, this approximation is only valid if the streamlines in the vicinity of the interface are not too distorted.

3.6.3 Discontinuous Galerkin for the Standard Level Set Method

Most of the numerical examples presented in this thesis utilize the CLS method given in section 3.6.2 on the facing page. Some few examples which were computed with a legacy code utilize the standard level set approach based on a repeatedly reinitialized distance function. In both cases, the simulations were based on the standard DG discretizations for hyperbolic problems described in [94]. Therefore the discretizations are just stated shortly and the relevant practical issues concerning their extensions to the UDG approach are discussed.

Notice that the analytical results on stability and error estimates for these methods, as presented in the original works [94, 95] and their textbook re-digestions [76], can not be trivially extended to the UDG approach. The relevant theorems involve local cell-wise estimates which are available for simplices and hexahedra but may not be easily extended to the almost arbitrary geometric shapes of the merged cut-cell meshes underlying the UDG approach. These shapes may defy even efforts like [96], where stability and error estimates for *star-shaped* cell geometries are provided. The scope of this thesis does not include any theoretical attempts to close this gap and the stability and accuracy of the presented approach is motivated by the numerical results in chapter 5 on page 99 only.

In order to apply a standard DG discretization to equation (3.56), one may follow the ideas in [89] and reformulate the level set equation as

$$\partial_t \phi + \nabla \cdot (\mathbf{u}\phi) = \nabla \cdot \mathbf{u}\, \phi, \quad \forall \mathbf{x} \in \Omega. \tag{3.68}$$

If the term on the right-hand side is treated as a source term, then, using the multi linear forms

$$\begin{aligned} m_\phi(\phi, v) = & \sum_{E\in\mathcal{T}} (\phi, v)_E \,, \\ c_\phi(\phi, v) = & \sum_{E\in\mathcal{T}} \left(\phi\, \mathbf{u}_h^{k+1}, \nabla v\right)_E, + \sum_{e\in\mathcal{E}} \left\langle \{\!\{\mathbf{u}_h^{k+1}\}\!\}\, \phi^+, \mathbf{n}[\![v]\!] \right\rangle_e \,, \\ f_\phi(\phi, v) = & \sum_{E\in\mathcal{T}} (\nabla \cdot \mathbf{u}_h\, \phi, v)_E \,, \end{aligned} \tag{3.69}$$

a corresponding DG discretization with a standard upwind flux approximation for the sub-problem in step 1 of the operator splitting is given by:

Find a level set function ϕ_h such that for $t \in [t_k, t_{k+1}]$ and $\boldsymbol{u}_h^{k+1} \in (\mathbf{P}_+^{M_v,k+1})^d$:

$$\forall v_h \in Z: \qquad \partial_t m_\phi(\phi_h, v_h) + c_\phi(\phi_h, v_h) = f_\phi(\phi_h, v_h). \tag{3.70}$$

In the definition of $c_\phi(\phi, v)$, the quantity ϕ^+ denotes the trace of ϕ on the upwind cell with regard to $\{\!\{\mathbf{u}_h^{k+1}\}\!\}$. This discretization is stable if the time integrator is chosen appropriately. The well-known *Heun-method*, a second order explicit Runge-Kutta type integrator was employed for all numerical examples presented in this thesis.

However, explicit time integrators always entail a time step restrictions typically associated with an upper boundary for the *Courant-Number*

$$C = \frac{u\Delta t}{\Delta x}. \tag{3.71}$$

Here, the variables u, Δt and Δx provide some measure for the maximum transport velocity, the time step size and the spatial resolution. Useful values for Δx are difficult to estimate for the UDG approach. Cautious choices like the minimal edge length of the sub-triangulation are bound to result in unreasonably small time steps as in this case Δx might not exceed the magnitude of the regularization parameter ε_p (see section 2.4.7). For the computations presented in this thesis, a Courant-Number was estimated by the resolution of the fundamental mesh $\Delta x := h$ and $u := \|\mathbf{u}_h^k\|_\infty$ and in each step Δt was chosen such that a value of $C \leq 0.1$ was ensured. While this choice allowed a stable transport for all simulations presented in this thesis, it should not be mistaken for a general statement on the stability of this approach for arbitrary cut-cell meshes.

The coupling of the phase-field transport (3.61) with the recompression equation (3.64) ensures that the phase-field is smooth at all times and thus the projection i.e. limiting methods [76, 94, 95] typically employed for the stabilization (or reduction of total variation) of this DG scheme are not required.

3.6.4 A Discrete Reinitialization Method for UDG

The authors of [85] compared the most common direct methods for the reinitialization of the level set function, i.e. for solving the Eikonal equation. They observed that the standard fast marching method proposed in [60] provided the best ratio of accuracy to computation time. The method itself is based on node values as provided by finite-difference schemes and for n nodes its run-time complexity is equal to $\mathcal{O}(n \log n)$.

As an alternative to direct methods, the hyperbolic problem (3.58) may be solved to determine a solution to the Eikonal equation. The extra effort involved in solving a full time-dependent PDE is only justified if it provides an otherwise unreachable measure of precision, i.e. smaller erroneous displacements of the zero level set. It was shown in [86] that this can be achieved by coupling the solution of the PDE with a constrained optimization procedure.

The implementation of such sophisticated schemes, however, was beyond the scope of this thesis. The fast marching method, on the other hand, is especially suited to the Cartesian structured grids underlying the sub-triangulations of the UDG method. A reinitialization of a given discrete DG level set function $\phi_h^k \in \mathbf{P}^{M_\phi}$ can be realized as follows:

1. To obtain node values of a given discrete DG level set function ϕ_h^k, it may be interpolated in the vertices of the geometry mesh $\mathcal{T}_\varkappa$ (and averaged if ϕ_h^k is discontinuous at the node coordinate).

2. The fast marching method can then be applied to these node values. This results in new node values which may be interpreted as a standard $\mathcal{Q}^1$ finite element function $\phi_{\mathcal{Q}^1}^k$ on $\mathcal{T}_\varkappa$.

3. A reinitialized discrete DG level set function $\hat{\phi}_h^k$ can then be obtained via L_2-projection of $\phi_{\mathcal{Q}^1}^k$ into $\mathbf{P}^{M_\phi}$.

Notice that step 1 implicitly assumes that ϕ_h^k is defined in each node of the geometry mesh $\mathcal{T}_\varkappa$ which is obviously not the case as Ω^h may in general be a real sub-set of Ω_r.

While the Fast Marching algorithm itself works well on a sub-set of the nodes (connected by edges) in $\mathcal{T}_\varkappa$, the L_2 projection in step 3 requires $\phi^k_{\mathcal{Q}^1}$ to be well defined on all cells which are overlapped, if only partially, by Ω^h.

Hence, on all cells in $\mathcal{T}_\varkappa$ where at least one node is inside of Ω^h and may thus be associated with a well defined level set function value, the values in all other nodes which are outside of Ω^h would have to be *constructed* by an appropriate extension method. For a given cell $E \in \mathcal{T}_\varkappa$ which contains part of the interface, this construction should result in a local $\mathcal{Q}^1$ function on E with a zero level set which is as close as possible to the zero level set defined by ϕ^k_h on $\Omega^h \cap E$.

This problem is equivalent to the one described in section 3.7 on page 82 where the intricacies of such extension methods are discussed in more detail. However, within the scope of this thesis, no solution to this problem is proposed and the presented examples which depend on a reinitialization of the level set function are restricted to the *fitted* case $\Omega^h = \Omega = \Omega_r$ and the conservative level set method is used in all other cases.

3.6.5 Discontinuous Galerkin for the Conservative Level Set Method

A complete realization of the conservative level set method as described in section 3.6.2 on page 72 requires the separate solution of the phase-field transport equation (3.61) and of the recompression equation (3.64) in an operator-splitting approach. Additionally, a stopping criterion for the recompression must be defined which robustly determines whether the phase-field is already *recompressed enough*, i.e. whether the stationary equilibrium solution of (3.64) has been reached up to a desired accuracy.

It is obvious that the solution of the phase-field transport equation can be achieved analogously to the case of the standard level set method by solving (3.70) as described in section 3.6.3, except that the source term $f(\cdot,\cdot)$ must be set to zero.

The recompression equation (3.64) may be approached in a similar manner. The discretization employed for this thesis follows the ideas in [97] for the discretization of the diffusive terms with the interior penalty method. This results in a discretization based on the multi-linear forms

$$
\begin{aligned}
m_\varphi(\varphi, v) = &\quad \sum_{E\in\mathcal{T}} (\varphi, v)_E \,, \qquad (3.72)\\
b_\varphi(\varphi, v, \mathbf{n}_\varphi) = &\quad \sum_{E\in\mathcal{T}} (\varphi\,(1-\varphi)\mathbf{n}_\varphi, \nabla v)_E \,, + \sum_{e\in\mathcal{E}} \langle \mathsf{F}, \mathbf{n}[\![v]\!]\rangle_e \,,\\
a^{\mathbf{n}}_\varphi(\varphi, v, \mathbf{n}_\varphi) = &- \sum_{E\in\mathcal{T}} \varepsilon_\varphi \,(\mathbf{n}_\varphi\cdot\nabla\varphi, \mathbf{n}_\varphi\cdot\nabla v)_E\\
&+ \sum_{e\in\mathcal{E}} \varepsilon_\varphi \,\langle \{\!\{\mathbf{n}_\varphi\mathbf{n}_\varphi\cdot\nabla\varphi\}\!\}, \mathbf{n}[\![v]\!]\rangle_e - \epsilon\,\varepsilon_\varphi \,\langle \mathbf{n}_\varphi\mathbf{n}_\varphi\cdot\nabla v\}\!\}, \mathbf{n}[\![\varphi]\!]\rangle_e\\
&\qquad + \frac{\sigma^\varphi_{IP}}{h_e}\,\langle [\![\varphi]\!], [\![v]\!]\rangle_e
\end{aligned}
$$

given by

$$\forall v_h \in Z: \qquad \partial_\tau m_\varphi(\varphi_h, v_h) + b_\varphi(\varphi_h, v_h, \mathbf{n}_\varphi) + a_\varphi^{\mathbf{n}}(\varphi, v, \mathbf{n}_\varphi) = 0, \tag{3.73}$$

$$\text{with } \mathbf{n}_\varphi := \frac{\nabla \varphi_h}{|\nabla \varphi_h|}, \tag{3.74}$$

where τ indicates the virtual (non-physical) time of the recompression process in contrast to the physical time t. Here, F must be chosen as an appropriate numerical flux function for the convective fluxes and the parameters ϵ, h_e and σ_{IP}^{φ} have the same meaning as in the IPDG flow discretization in (3.34). Whenever a face in $\mathcal{E}$ coincides with the domain boundary $\partial\Omega^h$, the jumps and averages in (3.72) are chosen to enforce homogeneous Neumann boundary conditions in order to preserve the global conservation property of the scheme (3.73).

Alas, the equation (3.73) is of rather malign character as it couples the phase-field φ with its *normalized* gradient field. Hence, time integration of (3.73) is difficult for both explicit and implicit methods as the former require minuscule step sizes and the latter lead to extremely ill-conditioned algebraic systems. To avoid these issues, the authors of [93] propose to keep $\mathbf{n}_\varphi$ constant within each time step, i.e. decouple it from φ and thus simplify both the hyperbolic term and the diffusion term.

Notice that the bandwidth controlling parameter ε_φ can not be chosen arbitrarily small as the bandwidth must at all times remain big enough to define a sane and accurate normal field $\mathbf{n}_\varphi$ which can be employed for both the recompression and the computation of the interface's curvature required for the surface tension, see chapter 4 on page 85. Too large values however, will amplify the erroneous diffusive fluxes of the recompression and thus reduce accuracy. Furthermore, the larger the bandwidth the stronger will be its distortion entailed by the propagation with the velocity field $\mathbf{u}$ resulting in an increased virtual time interval that has to be simulated in order to achieve a satisfying recompression of the phase-field. The actual value for ε_φ should thus be as small as possible without corrupting the stability of the scheme. Whether this is the case depends also on the polynomial approximation order of the finite element space used for the representation of φ. For $\varphi_h \in \mathbf{Q}^3$ a choice of $\varepsilon_\varphi = 0.3\,h$ was found to be stable under all encountered circumstances while the same simulations would become unstable for $\varphi_h \in \mathbf{Q}^2$ unless the higher value of $\varepsilon_\varphi = 0.5\,h$ was employed.

For all examples presented in this thesis, the penalty parameter σ_{IP}^{φ} on a face e in $\mathcal{E}$ with neighboring cells E and F was chosen as $\sigma_{IP}^{\varphi} = 1$ Utilizing the one-step-θ method (which will be used exclusively for all computational examples with the Crank-Nicholson choice $\theta = 0.5$) one may thus formulate the complete discretization as:

For a given $\varphi_h^{\kappa-1} \in \mathbf{P}^{M_\phi}$ and $\boldsymbol{n}_\varphi^{\kappa-1,*} \in (\mathbf{P}^{M_\phi})^d$ at time $\tau = \tau_{\kappa-1}$, find a phase-field $\varphi_h^{\kappa} \in \mathbf{P}^{M_\phi}$ such that

$$\forall v_h \in Z: \tag{3.75}$$

$$\begin{aligned} \frac{m_\varphi(\varphi_h^{\kappa}, v_h) - m_\varphi(\varphi_h^{\kappa-1}, v_h)}{\tau_\kappa - \tau_{\kappa-1}} \quad &+ \theta \quad \Big(b_\varphi(\varphi_h^{\kappa}, v_h, \mathbf{n}_\varphi^{\kappa-1,*}) + a_\varphi^{\mathbf{n}}(\varphi_h^{\kappa}, v_h, \mathbf{n}_\varphi^{\kappa-1,*})\Big) \\ &+ (1-\theta) \quad \Big(b_\varphi(\varphi_h^{\kappa-1}, v_h, \mathbf{n}_\varphi^{\kappa-1,*}) + a_\varphi^{\mathbf{n}}(\varphi_h^{\kappa-1}, v_h, \mathbf{n}_\varphi^{\kappa-1,*})\Big) \\ &= 0. \end{aligned}$$

Again, the variable τ is used to distinguish the *virtual* time of the recompression from the *physical* time t of the two-phase flow process. Also, the superscript κ is used to indicate the discrete solutions of φ at time τ_κ[2].

In the original publication of the CLS method [93], the authors represent the phase-field by a standard continuous piecewise linear basis and propose to compute its normal field via a L_2-projection of $\frac{\nabla\varphi}{|\nabla\varphi|}$ into the same (vector) space of piecewise linear polynomials. While for continuous piecewise polynomial spaces the L_2-projection is a global operation, for the DG spaces $\mathbf{P}^{M_\phi}$ this is a cell-wise local operation and as $\mathbf{P}^{M_\phi-1} \subset \mathbf{P}^{M_\phi}$ this local L_2-projection would be identical to

$$\mathbf{n}_\varphi^{\kappa,*} := \frac{\nabla\varphi_h^\kappa}{|\nabla\varphi_h^\kappa|}. \tag{3.76}$$

However, this choice results in a quite undesirable behavior of the recompression process as it introduces discontinuities and wiggles in the interface even at rather high mesh resolutions, see section 3.6.6 on the next page. This can be improved by utilizing a *diffusive* L_2-projection which couples neighboring cells with the diffusive fluxes of the IPDG method:

$$\begin{aligned} &\forall \mathbf{v}_h \in (Z)^d : \\ &\sum_{E\in\mathcal{T}} (\mathbf{n}_\varphi^{\kappa,*} - \frac{\nabla\varphi_h^\kappa}{|\nabla\varphi_h^\kappa|}, \mathbf{v}_h)_E - \varepsilon_{\mathbf{n}} \left(\nabla\mathbf{n}_\varphi^{\kappa,*}, \nabla\mathbf{v}_h\right)_E \\ &+ \sum_{e\in\mathcal{E}} \varepsilon_{\mathbf{n}} \left\langle \{\!\{\nabla\mathbf{n}_\varphi^{\kappa,*}\}\!\}, \mathbf{n}[\![\mathbf{v}_h]\!] \right\rangle_e - \epsilon\,\varepsilon_{\mathbf{n}} \left\langle \{\!\{\nabla\mathbf{v}_h\}\!\}, \mathbf{n}[\![\mathbf{n}_\varphi^{\kappa,*}]\!] \right\rangle_e + \frac{\sigma_{IP}^{\mathbf{n}}}{h_e} \left\langle [\![\mathbf{n}_\varphi^{\kappa,*}]\!], [\![\mathbf{v}_h]\!] \right\rangle_e = 0. \end{aligned} \tag{3.77}$$

Notice that, according to the aforementioned argument, this choice is equivalent to (3.76) if $\varepsilon_{\mathbf{n}} = 0$. Whenever the faces in $\mathcal{E}$ coincide with the domain boundary of Ω^h, the jump terms and averages in the formulation above must be adapted to weakly enforce suitable boundary conditions for this elliptic problem according to the theory of the IPDG method. Whenever such a face is not in the vicinity of the discrete interface, homogeneous Neumann boundary conditions could be considered an appropriate choice. However, if the face is near to the interface or actually touches the contact line, the recompression process may influence the contact angle of the interface with the boundary and the boundary conditions of the normal field projection must thus be chosen carefully. As this matter is deeply entwined with the discretization of the surface tension, its discussion is deferred to section 4.4 on page 94.

The CLS method as formulated in (3.62) is an advancement on the original publication [92] which proposed a recompression scheme in which the diffusive fluxes were not directed along the streamlines of the normal vector field:

$$\partial_\tau\varphi + \nabla\cdot(\mathbf{u}\cdot\varphi) + \alpha\left(\nabla\cdot(\varphi(1-\varphi)\mathbf{n}_\varphi) - \varepsilon_\varphi\,\Delta\varphi\right) = 0, \quad \forall \mathbf{x}\in\Omega. \tag{3.78}$$

2 This notation introduces an ambiguity in the superscript of φ_h^i which is suffered to allow a convenient notation. However, to keep confusion to a minimum, for actual number values of κ, e.g. $\kappa = 3$ the corresponding phase-field will always be denoted as $\varphi_h^{\kappa=3}$ and never as φ_h^3. The latter notation uniquely indicates the phase-field solution at time $t = t_3$.

Mesh Level	$\varepsilon_{\mathbf{n}} = 0$, a_φ	$\varepsilon_{\mathbf{n}} = 0$, $a_\varphi^{\mathbf{n}}$	$\varepsilon_{\mathbf{n}} = 0.1\,h^2$, a_φ	$\varepsilon_{\mathbf{n}} = 0.1\,h^2$, $a_\varphi^{\mathbf{n}}$
0	4.036E-2 (13)	3.856E-2 (12)	1.314E-3 (14)	1.182E-2 (11)
1	6.987E-3 (19)	1.048E-2 (200)	6.005E-3 (20)	5.841E-3 (20)
2	7.006E-3 (131)	1.070E-2 (200)	1.776E-3 (46)	1.210E-3 (70)
3	1.561E-2 (51)	6.777E-3 (200)	6.145E-4 (81)	1.231E-3 (65)

Table 3.1: The table lists the interface length errors err_Γ as given by (3.85) for the test case in section 3.6.6. In paranthesis behind each error value follows the number of virtual time steps needed to reach the stopping criterion (3.80). Each column corresponds to a simulation run on all four grid levels and $\varepsilon_{\mathbf{n}}$ indicates the utilized diffusivity for the L_2 projection of the normal field $\mathbf{n}_\varphi^{\kappa,*}$ of φ_h^κ while a_φ or $a_\varphi^{\mathbf{n}}$ indicate whether the simulations were run using the *isotropic* multi-linear form in (3.79) or the *anisotropic* one in (3.72). When the stopping criterion was not reached after 200 steps, then the propagation was aborted and the err_Γ value simply corresponds to $\varphi^{\kappa=200}$.

For the DG discretization, this corresponds to a modification of the multi-linear form $a_\varphi^{\mathbf{n}}(\varphi, v, \mathbf{n}_\varphi)$ according to

$$\begin{aligned} a_\varphi(\varphi, v) = \quad & - \quad \sum_{E\in\mathcal{T}} \varepsilon_\varphi \left(\nabla\varphi, \nabla v\right)_E \\ & + \quad \sum_{e\in\mathcal{E}} \varepsilon_\varphi \left\langle \{\!\{\nabla\varphi\}\!\}, \mathbf{n}[\![v]\!]\right\rangle_e - \epsilon\,\varepsilon_\varphi \left\langle \{\!\{\nabla v\}\!\}, \mathbf{n}[\![\varphi]\!]\right\rangle_e + \frac{\sigma_{IP}^\varphi}{h_e} \left\langle [\![\varphi]\!], [\![v]\!]\right\rangle_e . \end{aligned} \tag{3.79}$$

A comparison of the relative performance of both approaches for a benign test case is given in section 3.6.6.

The stopping criterion of the propagation scheme (3.75) was chosen as

$$\frac{\zeta(\varphi_h^\kappa - \varphi_h^{\kappa-1})}{\zeta(\varphi_h^{\kappa-1})} < \delta_\varphi \tag{3.80}$$

with

$$\zeta(\varphi_h) := \|0.25 - (\varphi_h - 0.5)^2\|_{\Omega^h}. \tag{3.81}$$

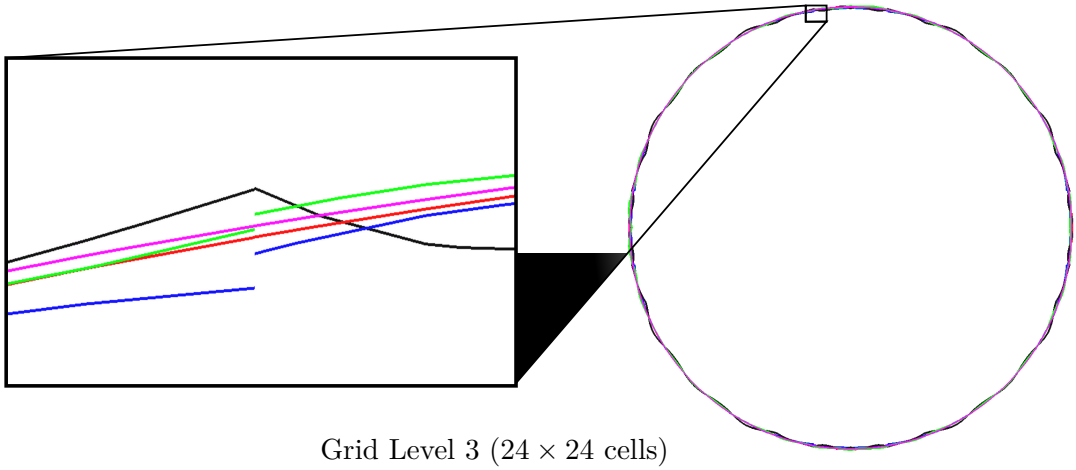
In comparison with the criterion

$$\frac{\|\varphi_h^\kappa - \varphi_h^{\kappa-1}\|_{\Omega^h}}{\tau_\kappa - \tau_{\kappa-1}} < \delta_\varphi \tag{3.82}$$

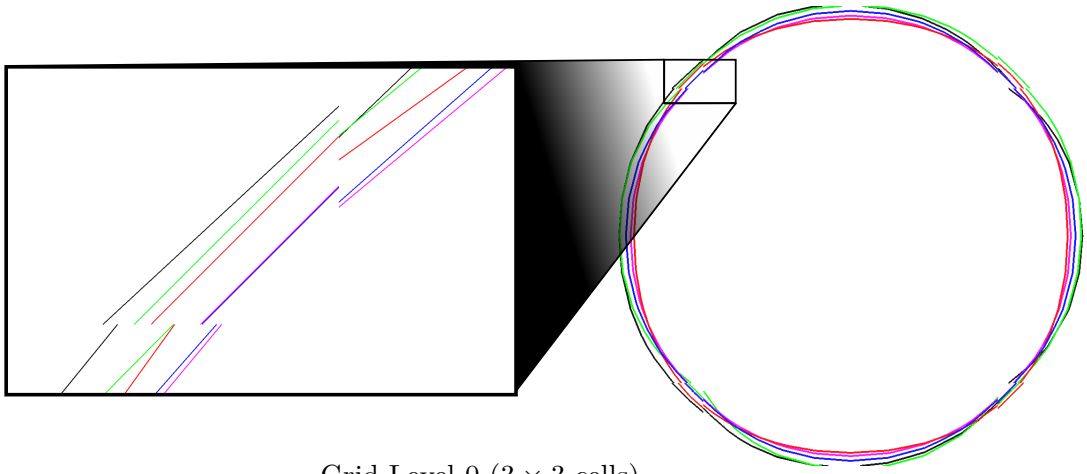
as proposed in [93], (3.80) is less dependent on the ratio of the interface length to the domain size but essentially equivalent as long as the time step of the propagation is scaled with the mesh resolution h.

3.6.6 Discontinuous Galerkin Recompression of Circular Interface

To obtain a quantitative estimate of its accuracy, the DG discretization of the CLS method as described above was applied to a benign test problem simulating the recompression of



Grid Level 3 (24×24 cells)



Grid Level 0 (3×3 cells)

Figure 3.7: The two pictures give an impression of the distortions introduced into the interface approximation due to the recompression of the phase-field φ for the test case described in section 3.6.6. The pictures show the 0.5-level sets of the discrete phase-fields after recompression as well as the reference level set for $\varphi_h^{\kappa=0} = 0.5$ *(pink)* on the finest and the coarsest mesh level.
For simulations based on the discretization with the anisotropic diffusion term $a_\varphi^{\mathbf{n}}$ (3.79), the resulting interface approximation is shown in *blue* for $\varepsilon_{\mathbf{n}} := 0.0$ and in *green* for $\varepsilon_{\mathbf{n}} := 0.1\,h^2$. Analogously, for simulations with the isotropic diffusion term a_φ defined in (3.72), the *black* interface corresponds to $\varepsilon_{\mathbf{n}} := 0.0$ and the *red* interface to $\varepsilon_{\mathbf{n}} := 0.1\,h^2$.
The magnifications of the solution with high resolution reveal that the interface computed with the isotropic term a_φ and the diffusive L_2-projection is smoother than the interfaces computed in the other simulations.

a circular interface. Foregoing the peculiarities of unfitted domains, this example utilizes a square domain of size (1×1) and is simulated for the Cartesian fundamental meshes $\mathcal{T}_h^0, \ldots, \mathcal{T}_h^3$, starting with a mesh $\mathcal{T}_h^0$ with 3×3 cells and ending with $\mathcal{T}_h^3$ with 24×24 cells. The circular interface is centered in the domain and has a radius of 0.25. A corresponding initial choice for $\varphi_h^{\kappa=0}$ is given by

$$\varphi^0(\mathbf{x}) := \Theta^{\epsilon_\varphi}\left(|\mathbf{x} - (0.5, 0.5)^T|\right) \tag{3.83}$$

with the regularized Heaviside function

$$\Theta^{\epsilon_\varphi}(x) := \begin{cases} 0, & x < -\epsilon_\varphi \\ \frac{1}{2} + \frac{x}{2\epsilon_\varphi} + \frac{1}{2\pi}\sin(\frac{\pi x}{\epsilon_\varphi}), & -\epsilon_\varphi \leq x < \epsilon_\varphi \\ 1, & \epsilon_\varphi \leq x \end{cases} \tag{3.84}$$

and the starting solution $\varphi_h^{\kappa=0}$ of the Crank-Nicholson scheme in (3.75) is obtained via a standard (non-diffusive) L_2-projection into the finite element space $\mathbf{Q}^3$ (which is here respectively defined with regard to the cells in $\mathcal{T}_h^0, \ldots, \mathcal{T}_h^3$). All simulations were run for $\varepsilon_\varphi = 0.3\,h$ corresponding to a bandwidth of approximately $\Delta_\varphi = 1.8\,h$ and the parameter ϵ_φ in the initial solution (3.83) was set to $\epsilon_\varphi := \Delta_\varphi$, i.e. the initial solution was already setup with the correct bandwidth such that all resulting changes in the course of the propagation must be due to numerical errors and the slight difference between the two Heaviside functions (3.84) and (3.65).

For each of the mesh levels, four simulations were performed comparing the choices for the diffusivity of the normal field projection $\varepsilon_{\mathbf{n}} := 0.0$ and $\varepsilon_{\mathbf{n}} := 0.1\,h^2$ as well as the influence of the suppression of diffusive fluxes tangential to the normal field in $a_\varphi^{\mathbf{n}}(\cdot,\cdot,\cdot)$ relative to $a_\varphi(\cdot,\cdot)$. In all cases, the time step size $\tau_\kappa - \tau_{\kappa-1}$ was chosen as $0.1\,h$ and the stopping criterion was given by (3.80) with $\delta_\varphi = 0.001$.

The results of the simulation are shown in figure 3.7 on the preceding page. Due to the high polynomial order of the employed finite element space $\mathbf{Q}^3$, the initial interface approximation realized by φ_h^0 even on mesh level 0 is already quite good considering that the radius of the circle is $0.75\,h$ in this case. On the highest mesh level, the simulation run utilizing the isotropic diffusivity term a_φ and the diffusive L_2-projection is noticeably smoother

A more quantitative measure of the relative performance is given in table 3.1 on page 78 which lists the number of steps K_φ needed to reach the stopping criterion and the error of the final interface length

$$\mathsf{err}_\Gamma := \Big| |\Gamma^{h,\kappa=K_\varphi}| - |\Gamma^{h,\kappa=0}| \Big|, \tag{3.85}$$

with

$$\Gamma^{h,\kappa} := \{\mathbf{x} \in \Omega_r \mid \varphi_h^\kappa(\mathbf{x}) = 0.5\}. \tag{3.86}$$

It is noticeable that in the simulations with $\varepsilon_{\mathbf{n}} = 0.0$ and the anisotropic diffusion term the stopping criterion could not be reached on almost all grid levels. The isotropic case is more robust but the numbers of required step sizes appear erratic considering the

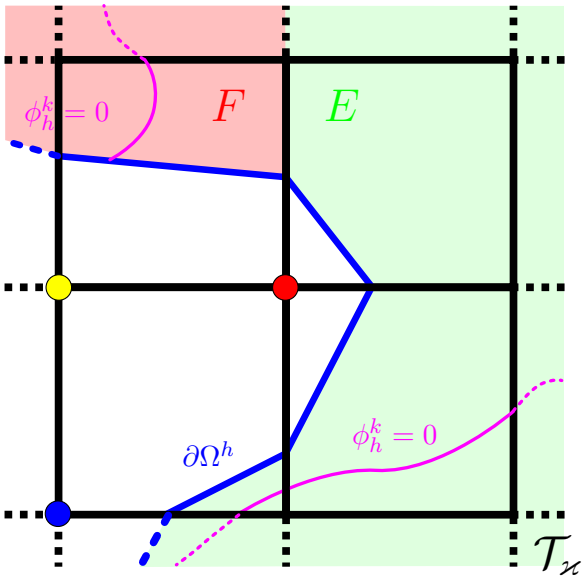


Figure 3.8: The picture illustrates the problem of extending ϕ_h^k on Ω_r by defining a continuous Q^1 solution. For the given example showing the intersection area of two cut-cell mesh cells $E, F \in \mathcal{T}$, such an extension would require the determination of suitable function values in the red, blue, and yellow nodes. As they are outside of Ω^h, function values of ϕ_h^k can not be directly obtained by interpolation (and possibly averaging). In the blue node, simply using the canonical extension of the polynomial defining ϕ_h^k on E is surely suitable. Assuming that the jump of ϕ_h^k across the edge between E and F is small, it is also valid to do the same for the red node with regard to the polynomials in both E and F and obtain the final unique value by averaging. The same does not hold for the yellow node as the extension from E may produce a vastly different value than the extension from F, hence averaging would potentially corrupt the interface in both cells.

high value for mesh level 2. The interface errors for the choice $\varepsilon_{\mathbf{n}} = 0.1\,h^2$ are in general smaller but only the isotropic variant shows robust convergence over all mesh levels.

Notice that the errors err_Γ are always computed with regard to the interface length corresponding to $\varphi_h^{\kappa=0}$ on the same mesh level.

The test case presented in this section is by its nature extremely benign and was chosen merely to illustrate that the ideas on the plain L_2-projection of the normal field $\mathbf{n}_\varphi^{\kappa,*}$ as proposed for the standard finite element implementation with first order linear basis functions presented in [92] and [93], can not be applied in combination with a DG discretization without further modifications. Furthermore, while the improved anisotropic diffusion term $a_\varphi^{\mathbf{n}}$ was shown in [93] to improve convergence relative to the choice of the isotropic term a_φ, the results obtained for the test case in this section show that it results in a less smooth interface approximation. This suggests that the improved convergence results, at least for the proposed discontinuous Galerkin discretization, may reduce the stability of this method in the context of two-phase flow systems with high surface tension.

A more intelligent and possibly adaptive choice of the diffusion parameter $\varepsilon_{\mathbf{n}}$ for the projection of the normal field might help to alleviate this problem but a systematic analysis of this option is beyond the scope of this thesis. For all further numerical examples presented in this thesis, the isotropic diffusion term a_φ was employed to eliminate a possible source of stability issues.

3.7 Continuous Extensions of the Level Set Function

This section is intended to discuss the step 2 of the operator splitting described in section 3.4.2 on page 60 consisting of the computation of a continuous extension $\tilde{\phi}_h^k \in C^0(\Omega_r)$ of a level set function $\phi_h^k \in \mathbf{P}^{M_\phi}$.

In case $\Omega^h = \Omega = \Omega_r$, i.e. the case of a *fitting* Cartesian mesh, this corresponds to a projection, or more generally, a mapping of ϕ_h^k into a continuous space where its image $\tilde{\phi}_h^k \in C^0(\Omega_r)$ should provide a good approximation of its zero level set. A canonical approach would be the definition of a standard Q^1-finite element basis by a simple interpolation of ϕ_h^k in all nodes of the geometry mesh $\mathcal{T}_\varkappa$. For such nodes that coincide with nodes, edges or faces in $\mathcal{T}$, this interpolation is not unique as ϕ_h^k will in general entail jumps across the faces in $\mathcal{T}$. However, in order to allow a stable propagation of the advancing interface, especially in the case of dominant surface tension, these jumps are required to be very small. This is typically achieved as a byproduct of the repeated reinitialization or recompression of the level set or phase-field function. Hence, the canonical approach of combining the trace values of ϕ_h^k in such points via an arithmetic average is justified and often effectively equivalent to arbitrarily choosing one of the trace values.

However, for a node $\mathbf{x}_c$ in the geometry mesh $\mathcal{T}_\varkappa$ which is positioned outside of Ω^h a *good* value is not easily found. Assume $\mathbf{x}_c$ is a node in the cell $E \in \mathcal{T}_\varkappa$ in the geometry mesh which is intersected by the level set function $\phi_B \in C^0(\Omega_r)$ (otherwise the exact value in the node is of no interest). Furthermore, E is intersected by the zero level set of the level set function ϕ_h^k. Although ϕ_h^k is only locally defined on $E \cap \Omega^h$, its polynomial representation allows a natural smooth extension of ϕ_h^k to the whole cell E. However, as illustrated in figure 3.8 on the previous page, the definition of a unique value in $\mathbf{x}_c$ may be problematic if significant distortions of the interface in its vicinity are to be avoided. Especially in the case of a contact line with a microscopic contact-angle, the influence of such distortions on the system dynamics may be especially severe.

The problem of a global continuous extension of ϕ_h^k can actually be avoided. The assumption that all level set functions are elements of $C^0(\Omega_r)$ was convenient for the definition of the subdomains and the sub-triangulation but is not truly necessary. Due to the nature of the recursive construction of the sub-triangulation in (2.14), with regard to the geometries involved in the two-phase flow problem defined in section 3.1 on page 53, it is actually sufficient for $\phi_2(t_k) = \phi_h^k$ to be an element of $C^0(\Omega^h)$ as long as the operator $\mathcal{I}^2\left(\mathfrak{e}, \phi_2\big|_E\right)$ for a given cell E and a cell-part $\mathfrak{e}$ does only depend on the restriction $\phi_2\big|_{\mathfrak{e}}$ instead of $\phi_2\big|_E$. This is obviously true for the fully k-conforming partition operator proposed in (2.30) in section 2.4.7 on page 50 as according to (2.31):

$$\mathcal{I}^2(\mathfrak{e}, \phi_E) := \phi\big|_{\mathfrak{e}}. \tag{3.87}$$

However, for the other, only weakly 1-conforming linear partition operator given in (2.32), this is not the case as the employed pre-processing operator was chosen as

$$\mathcal{I}^2(\mathfrak{e}, \phi_E) := \phi^{P1(E)}\big|_{\mathfrak{e}}. \tag{3.88}$$

To the author's knowledge, there is no clean way to solve this problem. However, since an implementation of a fully k-conforming partition operator was not available, a *working solution* was obtained as follows:

Instead of using (3.88), the pre-processing operator $\mathcal{I}^2$ was modified according to

$$\mathcal{I}^2(\mathfrak{e}, \phi_E) := \phi^{P1(\mathfrak{e})}\big|_{\mathfrak{e}}. \tag{3.89}$$

As the result of this operation is in general not a linear function on $\mathfrak{e}$ (but a piece-wise linear one), the scheme in (2.32) defaults to the fourth option in both recursion steps. This effectively means, that both the initial cell of the geometry mesh and all cell-parts of the first recursion step will be linearized, i.e. pre-partitioned into simplices before the actual sub-triangulation takes place (which will thus employ the Marching-Simplex algorithm only).

Alas, here finally even weak 1-conformity is lost. Consider two adjacent cells $E, F \in \mathcal{T}_\varkappa$ intersected by the zero level sets of both level set functions. After the first recursion step in (2.14), the resulting cell-parts $S_1^i(E)$ and $S_1^i(F)$ may contain adjacent cell-parts $\mathfrak{e} \in S_1^i(E)$ and $\mathfrak{f} \in S_1^i(F)$ (though not necessarily in a full face). The modification (3.89) will result in non-linear level set functions on both cell-parts and even the surfaces defined by their exact zero level sets on each cell-part need not be continuously connected.

For the two-phase flow problem, the breakdown of weak 1-conformity occurs in the last step of the two-step recursive construction scheme. Therefore, considering the remarks in section 2.3.4 on page 41, it becomes clear that the impact of this short-coming is rather benign. The resulting sub-triangulation will under no circumstances contain gaps or overlaps. Nevertheless, the reconstructed interface will in general not be continuous, a fact which must be considered especially for the discretization of the surface tension. Also, unless the exact intersection areas of two adjacent cell-parts with non-matching faces are reconstructed for the numerical integration (which our implementation does not realize), the numerical fluxes will involve errors wherever the weak 1-conformity was actually violated. Though as long as the fluxes from Ω_1^h to Ω_2^h are always exclusively computed for the cell-part faces in $\mathcal{S}_1$ (or exclusively for the cell-parts faces in $\mathcal{S}_2$) mass conservation properties of the corresponding DG scheme will be maintained.

4 Surface Tension

In spite of the many references in the previous chapters, all details of the discretization of surface tension were deferred to this chapter and hidden inside the interface momentum source term $\mathbf{f}_\Gamma$ defined on the discrete interface $\Gamma^{h,k}$.

For discontinuous Galerkin methods and especially for the UDG approach, discretization of the surface tension holds many challenges which are not encountered in conforming finite element discretizations on either matching or non-matching grids. Many standard approaches implicitly rely on the natural volume coupling realized by the overlapping supports of the finite element basis functions. Independent on whether those methods rely on the geometric discretization realized by the integration by parts of the Laplace-Beltrami Operator (as proposed by [9]) or on a direct computation of the curvature. In both cases, this natural coupling is used to evade the problem resulting from the curvature being actually not a cell-local property. Within the context of DG methods, this *natural* coupling, realized by the continuity property of standard finite element spaces, is not available and a surrogate coupling must be introduced.

Four different approaches to the discretization of the surface tension are presented within this chapter. One of them is restricted to the *fitted* case $\Omega^h = \Omega_r$ and was already described in [29]. However, as some of the simulations given in chapter 5 are based on this method it is shortly recapitulated. The other three methods are suitable for applications in the full unfitted case. A comparison of their performance is realized for a simple test case in 5.3 on page 104.

4.1 On Terminology

When devising a numerical scheme for the simulation of moving interfaces with surface tension, a decision on two decisive aspects of the numerical model has to be made:

I. How is the interface represented by a finite number of degrees of freedom and according to which algorithm are they to be modified such that the movement of the interface along the streamlines of a given velocity field is approximated accurately?

II. How can the surface tension be discretized in a way that is both suitable for the chosen representation of the interface and the chosen physical flow model as well as its discretization.

The questions I. and II. effectively define two dimensions along which a given numerical scheme may be categorized and a new numerical scheme may be positioned.

The first dimension, i.e. question I., spans a terminological range from *surface tracking* (or often *front tracking*) to *surface capturing* methods. The former represent the position of a simulated interface via some kind of geometric markers which may in the most canonical way be realized by the vertices of a surface mesh that may or may not be embedded in the volume mesh utilized for the flow simulation. Such schemes, which effectively move the whole mesh along, are easy to realize even with standard finite element software but are typically restricted to very small deformations of the interface unless elaborate remeshing algorithms are applied. However, more sophisticated schemes with marker points or the control points of splines have been proposed and successfully realized [9, 12, 98, 99].

As an alternative to the direct representation and manipulation of geometric entities, the *surface capturing* or *implicit interface* methods define the current position of the interface implicitly by referring to scalar valued functions whose time propagation is supposed to capture the geometric evolution of the simulated interface. The most popular realizations of such scalar functions include the level set function and phase-field function methods as introduced in section 3.6 on page 70.

The *surface capturing* methods generally entail high computational effort typically involving the solution of (at least) one partial differential equation which has to be solved (at least) in the vicinity of the interface. The requirements of reinitializing (section 3.6.1) or recompressing (section 3.6.5) the involved scalar functions add to this natural overhead of such implicit interface methods. However, the direct manipulation of complex geometric entities may, especially in three-dimensions, lead surprisingly fast to methods of tremendous algorithmic complexity. The popularity of *surface capturing* methods may thus, at least partly, be explained by the tremendous effort involved in creating, maintaining, and extending the software-frameworks required for stable and accurate simulations based on elaborate *surface tracking* methods.

The second dimension, i.e. question II., distinguishes the possible discretization schemes for the actual surface tension force emitted by $\mathbf{f}_\Gamma$ along the explicit or implicit geometric representation of the discrete interface. In the classical *continuum surface force* (CSF) model [100], the contribution of $\mathbf{f}_\Gamma$ is *smeared* along a discrete interval across the interface. As this interval must be resolved by the mesh resolution although its width effectively defines a lower bound of geometric accuracy, such methods typically rely on local adaptive refinement of the computational mesh and low order approximations at least in the vicinity of the interface. On the other hand, they have been shown to be robust and entail good convergence properties for many two-phase flow setups [3, 16, 100, 101, 102, 103]. It was long argued that this approach would be inherently unsuited for interfaces with high surface energy, as it involves steep gradients of the pressure field across the interface which amplify any approximation error of the discrete solution. However, it was shown in [104] that the resulting spurious velocities can be controlled if the approximation of the (directed) curvature and the pressure fields are *balanced*, i.e. they are represented in the same finite element basis.

Whenever a discrete representation of the interface geometry is available, a *sharp interface* method can be applied as an alternative to the continuum surface force model. This approach, in principle, eliminates the numerical error resulting from the *smearing* of the interface in the CSF method. Furthermore, while the CSF method relies on the explicit computation of the interface curvature based on higher derivatives of the level set or phase-field function, a geometric representation of the interface allows to assess the surface tension in a geometric approach which can be formulated with minimal regularity requirements on the level set or phase-field function. Realizations of such essentially geometric discretization methods for surface tension were already applied for two-phase flow problems in [9] and have been adapted in [10, 26, 27, 98].

However, sharp interface methods are flexible and can also be combined with methods that directly compute the interface curvature from the scalar function implicitly repre-

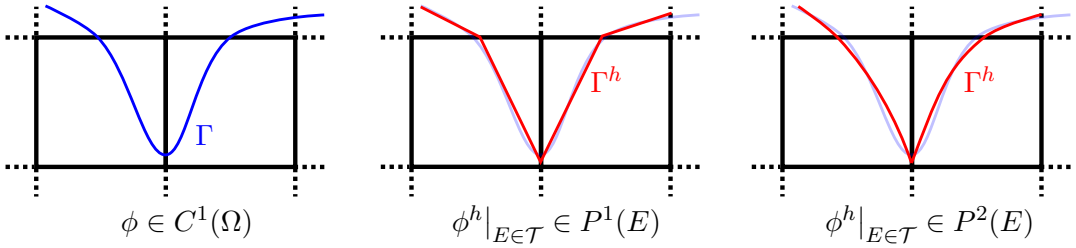


Figure 4.1: The pictures above illustrate some possible results which may be obtained by approximating the level set function which represents the interface Γ with cell-wise polynomials. Using first order approximations results in a piecewise linear zero level set (picture in the middle). Obviously, when restricted to a single cell, the curvature of the local interface approximation is zero. The picture to the right illustrates how higher-order approximations need not necessarily improve the approximation of the local curvature as long as the discrete approximation is only locally smooth.

senting the interface [17]. In principle, sharp interface methods can be applied for both surface tracking and surface capturing methods although the latter require an additional reconstruction of the interface geometry. Such reconstructions have been successfully applied to improve existing schemes even in the context of finite difference methods [79].

An exclusive advantage of sharp interface methods is the ability to combine the surface tension discretization with a discretization of the velocity and pressure fields which allow for an exact representation of the interface conditions (3.12) including the jump in the pressure field and the velocity derivatives using discontinuous Galerkin or extended finite element methods [12, 27, 35]. Only such methods which eliminate all geometric regularizations commonly applied to the viscosity field, the pressure field and the interface thickness may exploit the full potential of higher-order finite element spaces.

Locating the methods which are suitable for the UDG approach within the presented terminology is rather straight-forward. As a discrete geometric representation of the interface $\Gamma^{h,k}$ is available and fitted to the cells of the final cut-cell mesh, a sharp interface method is suitable and allows the discontinuous representation of the full interface conditions (3.12) (for the aligned mesh approach, section 3.5.4) or at least the pressure jump (for the hybrid mesh approach, section 3.5.3). The surface capturing methods utilized for the propagation and reconstruction of the discrete interface were already described in 3.6 on page 70.

4.2 Direct Methods

Direct methods for the discretization of the surface tension rely on an explicit computation of the curvature based on the derivatives of the level set function according to

$$\kappa = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}. \tag{4.1}$$

Notice that this formulation implies the existence of second derivatives of the level set function. It is thus not surprising that such direct methods were first employed in finite difference schemes [60, 105].

Utilizing the relation (4.1) directly in a finite element method is problematic. When the level set function is represented in a piecewise linear finite element basis - which is sufficient for solving the level set equation - the cell-local curvature approximation reduces everywhere to $\kappa = 0$. But even when higher-order basis functions are employed, the direct evaluation of expression (4.1) may not result in good approximations of the local interface curvature. This is a direct consequence of the fact that curvature of a non-smooth approximation of the interface Γ is simply not a cell-local property as kinks coinciding with cell edges or faces are not accounted for, see figure 4.1 on the preceding page.

4.2.1 Finite Difference Method

In [29], a direct method for the approximation of the curvature and thus the surface tension was presented which was found suitable for the special requirements of UDG two-phase flow discretizations. Here, at a given time t_k, the continuous discrete approximation of the level set function $\tilde{\phi}_h^k \in C^0(\Omega_r)$ (the same which is utilized in step 2 of the operator splitting in section 3.4.2 on page 60 to construct the sub-triangulation $\mathcal{S}^k$) is interpolated in all nodes of the geometry mesh $\mathcal{T}_\varkappa$. Approximations of the local curvature are then computed in each node $\mathbf{x}_n$ according to standard methods [60] by:

$$\kappa_h^k(\mathbf{x}_n) = \frac{\left\{ \begin{array}{c} (\partial_{yy}^h\tilde{\phi} + \partial_{zz}^h\tilde{\phi})\,(\partial_x^h\tilde{\phi})^2 + (\partial_{xx}^h\tilde{\phi} + \partial_{zz}^h\tilde{\phi})\,(\partial_y^h\tilde{\phi})^2 + (\partial_{xx}^h\tilde{\phi} + \partial_{yy}^h\tilde{\phi})\,(\partial_z^h\tilde{\phi})^2 \\ -2\,\partial_x^h\tilde{\phi}\,\partial_y^h\tilde{\phi}\,\partial_{xy}^h\tilde{\phi} - 2\,\partial_x^h\tilde{\phi}\,\partial_z^h\tilde{\phi}\,\partial_{xz}^h\tilde{\phi} - 2\,\partial_y^h\tilde{\phi}\,\partial_z^h\tilde{\phi}\,\partial_{yz}^h\tilde{\phi} \end{array} \right\}}{((\partial_x^h\tilde{\phi})^2 + (\partial_y^h\tilde{\phi})^2 + (\partial_z^h\tilde{\phi})^2)^{\frac{3}{2}}}. \tag{4.2}$$

Here, $\tilde{\phi}_h^k$ was abbreviated by $\tilde{\phi}$ and $\partial_\cdot^h$ denotes second order central finite-difference approximations of the partial derivatives based on the values of $\tilde{\phi}_h^k$ in the nodes of $\mathcal{T}_\varkappa$.

Interpreting the node values computed by (4.2) as a continuous cell-wise Q^1 function (according to the standard Lagrangian Q^1 basis employed for conforming finite elements) $\tilde{\kappa}_h^k \in C^0(\hat{\mathbf{\Omega}}(\mathcal{T}_\varkappa))$ on the geometry mesh $\mathcal{T}_\varkappa$ and setting

$$\mathbf{f}_\Gamma := \sigma_s\,\tilde{\kappa}_h^k\,\mathbf{I} \tag{4.3}$$

allows a direct integration of the term

$$f^k(\mathbf{v}) = \sum_{e \in \mathcal{E}_\Gamma^{h,k}} \langle \mathbf{f}_\Gamma, \mathbf{n}\{\{\mathbf{v}\}\}\rangle_e \tag{4.4}$$

as defined in (3.34). It was shown in [29] for a simple test case that this approach provides a better approximation of the local curvature than the application of the geometric method via integration by parts of the Laplace-Beltrami operator on Γ_h^k.

Notice that the assumed existence of a global continuous level set function $\tilde{\phi}_h^k$ is problematic as ϕ_h^k is in general only defined on Ω^h and a continuous extension on Ω_r may entail

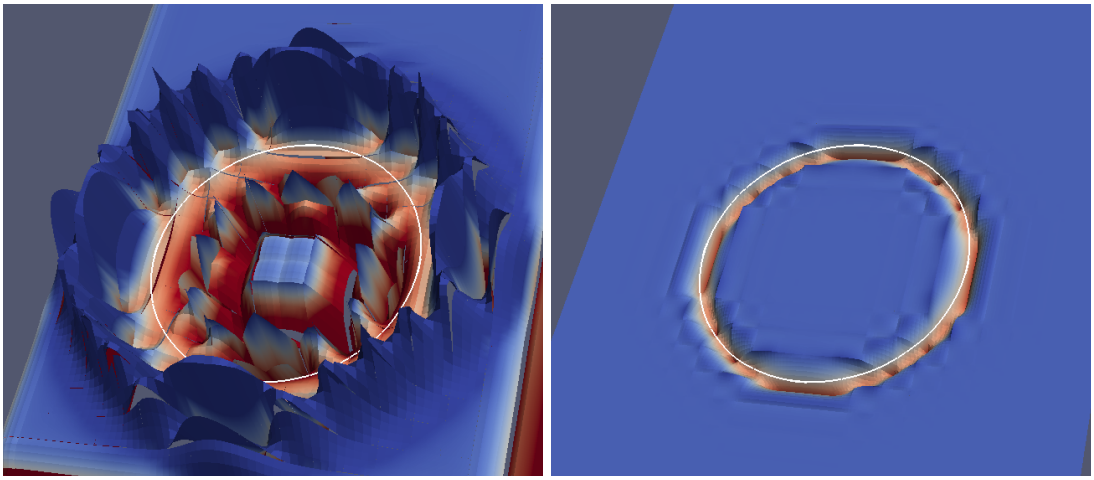


Figure 4.2: The pictures above give an idea of the curvature field resulting from the projection methods (4.8) (including the regularization in (4.9)) *(left)* and (4.11) *(right)* when applied to a phase-field representing a circular interface *(white line)*. Both pictures were computed for $\varepsilon_\kappa = (0.1\,h)^2$ and the DG space $\mathbf{Q}^3$ was employed for the appoximation of the phase-field, the normal field projection and the curvature projection. Notice that the narrow band utilized for the projection in the right picture hardly spans three grid cells which is possible due to the high polynomial order of the basis functions.

unexpected complications, as described in section 3.7 on page 82. Correspondingly, the examples presented in [29, 35] and chapter 5 were restricted to the *fitted* case $\Omega_r = \Omega^h$.

4.2.2 Projection Methods

The issues of the finite difference approach with regard to unfitted domains suggest that a coherent discretization in the context of the UDG method should be formulated by exclusive reference to the *natural* UDG spaces $\mathbf{P}^m$, $\mathbf{P}_+^{m,k}$ and $\mathbf{Q}^m$, $\mathbf{Q}_+^{m,k}$. Consider the standard L_2-projection of the normal field divergence

$$\nabla \cdot \mathbf{n}_h^k \qquad \text{with} \qquad \mathbf{n}_h^k := \frac{\nabla \phi_h^k}{|\nabla \phi_h^k|} \tag{4.5}$$

to obtain the discrete curvature $\kappa_h^k \in \mathbf{P}^{M_\phi}$. After integration by parts and using the identity (3.27) such a projection may be written as:

$$\forall v_h \in Z : \tag{4.6}$$
$$\left(\kappa_h^k, v_h\right)_{\Omega^h} - \left(\mathbf{n}_h^k, \nabla v_h\right)_{\Omega^h} + \sum_{e \in \mathcal{E}} \left(\left[\!\left[\mathbf{n}_h^k \right]\!\right], \mathbf{n} \{\!\{ v_h \}\!\} \right)_e + \left(\{\!\{ \mathbf{n}_h^k \}\!\}, \mathbf{n} \left[\!\left[v_h \right]\!\right] \right)_e = 0.$$

Obviously, the term $\left[\!\left[\mathbf{n}_h^k \right]\!\right]$ will in general not be zero but revisiting figure 4.1 on page 87 reveals that this jump of the discrete normal field mediates exactly the geometric coupling required to capture kinks which coincide with cell faces. Based on this geometric argument

a *non-local* projection of the curvature field can be formulated as:

$$\forall v_h \in Z : \qquad (4.7)$$
$$\left(\kappa_h^k, v_h\right)_{\Omega^h} - \left(\mathbf{n}_h^k, \nabla v_h\right)_{\Omega^h} + \sum_{e\in\mathcal{E}} \left(\{\!\{\mathbf{n}_h^k\}\!\}, \mathbf{n}\,[\![v_h]\!]\right)_e = 0.$$

However, very high curvature values have the potential of destabilizing the interface propagation and may naturally appear in two-phase flow setups, e.g. when separated phase regions touch and coalesce. But also numerical approximation errors may introduce additional spurious surface tension which may deteriorate the time step size limit required for a stable simulation. These issues may be alleviated by adding artificial diffusion to the projection (4.7) thus reducing the impact of erroneous peak values:

$$\forall v_h \in Z : \qquad (4.8)$$
$$\left(\kappa_h^k, v_h\right)_{\Omega^h} - \left(\mathbf{n}_h^k, \nabla v_h\right)_{\Omega^h} + \sum_{e\in\mathcal{E}} \left(\{\!\{\mathbf{n}_h^k\}\!\}, \mathbf{n}\,[\![v_h]\!]\right)_e - \varepsilon_\kappa \left(\nabla\kappa_h^k, \nabla v_h\right)_{\Omega_h}$$
$$+ \sum_{e\in\mathcal{E}} \varepsilon_\kappa \left\langle \{\!\{\nabla\kappa_h^k\}\!\}, \mathbf{n}[\![v_h]\!]\right\rangle_e - \epsilon\,\varepsilon_\kappa \left\langle \{\!\{\nabla v_h\}\!\}, \mathbf{n}[\![\kappa_h^k]\!]\right\rangle_e + \frac{\sigma_{IP}^\kappa}{h_e} \left\langle [\![\kappa_h^k]\!], [\![v_h]\!]\right\rangle_e = 0.$$

The additional terms in the second row of (4.8) correspond to diffusive fluxes based on the interior-penalty method, similar to the L_2-projection in (3.77). For a robust convergence behavior, ε_κ should be scaled $\propto h^{-2}$. Notice that these diffusive fluxes serve only to smooth the curvature field but do not enforce a coupling of the cell-local curvatures, thus adding the same diffusive fluxes to (4.6) would still result in $\kappa_h^k = 0$ as long as the level set function is linear, i.e. $\phi_h^k \in \mathbf{P}^{1,k}$.

The choice of $\mathbf{n}_h^k$ given in (4.5) is not suited to be employed in the projection scheme (4.8). If the conservative level set method is employed, $|\nabla\phi_h^k|$ must be expected to be arbitrarily small in a vast part of the domain and thus (4.5) can not provide meaningful values for the normal field. Also, for the classical level set method, sharp transitions may occur in the gradient field $\nabla\phi_h^k$ thereby introducing spurious vortices into the normal field which will inevitably introduce large values for the local curvature κ_h^k. As soon as a non-zero diffusion coefficient ε_κ is employed in (4.8), such local numerical singularities, even if located in supposedly sufficient distance to the interface may - depending on their magnitude - result in a global deterioration of the curvature approximation due to the in principal non-local character of the diffusion process. Hence, within the implementation realized for this thesis, (4.5) was replaced by

$$\mathbf{n}_h^{k,\varepsilon} := \frac{\nabla\phi_h^k}{|\nabla\phi_h^k|}\,\Theta^\varepsilon\left(|\nabla\phi_h^k| - \varepsilon\right). \qquad (4.9)$$

Here Θ^ε denotes the regularized Heaviside function as defined in (3.84). For the standard level set method where the expected gradient magnitude is $|\nabla\phi_h^k| \approx 1$, the generic regularization parameter ε should be chosen as a constant sufficiently smaller than 1. However, when applying the conservative level set method, ε should be scaled with $\frac{1}{\Delta_\varphi}$ which describes the average magnitude of the gradient field in this case. All computational examples in this thesis which utilize this projection approach employ the conservative

level set method and ε was chosen as $\frac{10^{-2}}{\Delta_\varphi}$.

The curvature field κ_h^k projected according to (4.8) using the regularized normal field (4.9) may yet be subjected to significant erroneous artifacts. These may be a result of approximation errors of the level set function or may be due to *modeling* errors, i.e. the interpretation of $\nabla\phi_h^k$ as a normal field at points where this interpretation is not justified and not yet suppressed by the regularization enforced in (4.9). Even the regularization itself introduces erroneous curvature values as the employed Heaviside function reduces the absolute value of the normal field from 1 to 0 over a distance ε and thereby produces an *apparent divergence*. While the latter problem can be thoroughly controlled by the regularization parameter, figure 4.2 on page 89 shows that within the projected curvature field the spurious distortions hardly allow even its qualitative visual assessment. To account for this insufficiency, another projection method is presented:

As illustrated in figure 4.1 on page 87 the cell-local curvature can not be directly employed but must be coupled to account for possible kinks in the discrete interface approximation, i.e. the jumps in the discrete approximation of the normal field of the level set function. While the projection in (4.8) utilizes exactly this jump to enforce the required coupling, the obvious alternative is to employ a projection of the normal field which does not contain any jumps, either by projecting $\mathbf{n}_h^k$ into a continuous finite element space *or* by utilizing the diffusive DG L_2-projection

$$\begin{aligned}
&\forall \mathbf{v}_h \in (Z)^d : \qquad (4.10)\\
&\sum_{E\in\mathcal{T}} (\mathbf{n}_h^{k,*} - \mathbf{n}_h^k, \mathbf{v}_h)_E - \varepsilon_{\mathbf{n}} \left(\nabla \mathbf{n}_h^{k,*}, \nabla \mathbf{v}_h\right)_E \\
+ &\sum_{e\in\mathcal{E}} \varepsilon_{\mathbf{n}} \left\langle \{\{\nabla \mathbf{n}_h^{k,*}\}\}, \mathbf{n}[\![\mathbf{v}_h]\!]\right\rangle_e - \epsilon\, \varepsilon_{\mathbf{n}} \left\langle \{\{\nabla \mathbf{v}_h\}\}, \mathbf{n}[\![\mathbf{n}_h^{k,*}]\!]\right\rangle_e + \frac{\sigma_{IP}^{\mathbf{n}}}{h_e} \left\langle [\![\mathbf{n}_h^{k,*}]\!], [\![\mathbf{v}_h]\!]\right\rangle_e = 0.
\end{aligned}$$

This scheme allows to control the jumps of the projection $\mathbf{n}_h^{k,*}$ with the penalty parameter $\sigma_{IP}^{\mathbf{n}}$ and the artificial diffusivity $\varepsilon_{\mathbf{n}}$. This scheme is equivalent to (4.10) presented in section 3.6.5 on page 75 for the computation of a normal field $\mathbf{n}^{\kappa,*}$ suitable for the recompression of the phase-field employed in the conservative level set method. Hence, in this case (4.10) does not have to be computed at all, but $\mathbf{n}^{\kappa,*}$ of the last recompression step may be utilized instead.

As already mentioned in the context of (3.77), the artificial diffusive fluxes in (4.10) were discretized with the interior penalty method and appropriate boundary conditions for this essentially elliptic problem have to be set, i.e. incorporated into the jump and average terms on domain boundary faces. If such a face is sufficiently far away from the interface, simple homogeneous Neumann boundary conditions may be appropriate but close to the interface the boundary conditions have to account for the influence of the contact angle. The detailed discussion is thus deferred to section 4.4.

As discontinuities in $\mathbf{n}_h^{k,*}$ are suppressed by the penalty terms, $\nabla\cdot\mathbf{n}_h^{k,*}$ may be directly evaluated to obtain a local curvature approximation. If stability requires additional

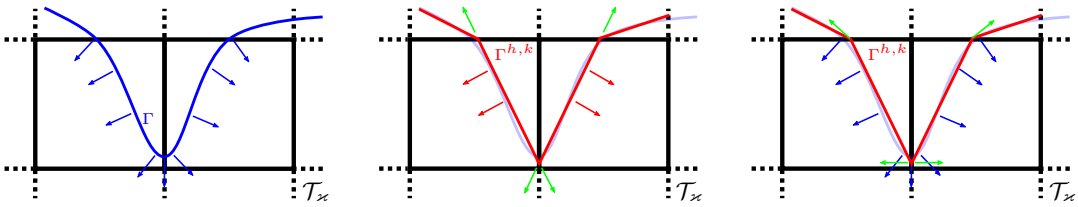


Figure 4.3: This figure qualitatively illustrates the quasi-geometric method as proposed in section 4.3. The picture to the left shows the interface $\Gamma(t)$ at time $t = t_k$ and depicts exemplarily the projection $\mathbf{n}_h^{k,*}$ which approximates its normal vector field. The picture in the middle shows a possible discrete approximation $\Gamma^{h,k}$ (red lines) of the interface together with its normal vector field (red arrows) which is cell-wise constant within each cell in the geometry mesh $\mathcal{T}_\varkappa$. The outer normal vectors ($\boldsymbol{\nu}$ in (4.15)) of each of the edge boundaries (which are points in two dimensions) are indicated by green arrows. The picture to the right illustrates the effect of the quasi-geometric method according to (4.17). The projected normal field $\mathbf{n}_h^{k,*}$ (blue arrows) is integrated along the discrete interface $\Gamma^{h,k}$ (red lines) and the edge boundary terms are evaluated with regard to $\boldsymbol{\nu}_h^{k,*}$ (green arrows).

regularization of the curvature, another projection may be applied to obtain a smooth localized curvature field:

$$\begin{aligned}
&\forall v_h \in Z : \qquad (4.11)\\
&\left(\kappa_h^k, v_h\right)_{\Omega^h} - \left((\nabla \cdot \mathbf{n}_h^k)\ \overline{\delta}_b(\phi_h^k), v_h\right)_{\Omega^h} - \varepsilon_\kappa \left(\nabla \kappa_h^k, \nabla v_h\right)_{\Omega_h}\\
&+ \sum_{e \in \mathcal{E}} \varepsilon_\kappa \left\langle \{\!\{\nabla \kappa_h^k\}\!\}, \mathbf{n}[\![v_h]\!] \right\rangle_e - \epsilon\, \varepsilon_\kappa \left\langle \{\!\{\nabla v_h\}\!\}, \mathbf{n}[\![\kappa_h^k]\!] \right\rangle_e + \frac{\sigma_{IP}^\kappa}{h_e} \left\langle [\![\kappa_h^k]\!], [\![v_h]\!] \right\rangle_e = 0.
\end{aligned}$$

Here $\delta_b(\phi_h^k)$ denotes some narrow band function. For the computations with the conservative level set method (which was exclusively used for this curvature approximation), the narrow band was defined by

$$\delta_b(x) = \Theta^{\beta/2}(x - \beta) \cdot \Theta^{\beta/2}(1 - \beta - x). \qquad (4.12)$$

The values for the parameter β employed for the computations employed in chapter 5 varied in the range of $\beta = 0.08 - 0.25$. For boundary faces, the jumps and averages in (4.11) may be adapted to enforce homogeneous Neumann boundary conditions.

Notice that in the course of this section, the DG space Z which is used for the spatial discretization of ϕ_h^k was also chosen as the range of all the proposed projection schemes. While this choice may be debatable, the relative performance of alternative configurations is beyond the scope of this thesis and was not analyzed systematically.

4.3 Quasi-Geometric Method

A class of geometric methods for the approximation of surface tension in two-phase flow problems is based on the surface-divergence theorem applied to the surface integral over

$\mathbf{f}_\Gamma = \sigma_s\,\kappa$. For a vector valued function $\mathbf{v} \in \left(H^1(\Omega)\right)^d$ and a constant surface energy σ_s, it holds that

$$\int\limits_{\Gamma(t)} \sigma_s\,\kappa\,\mathbf{n}\,\mathbf{v}\,ds = -\int\limits_{\Gamma(t)} \sigma_s\,\underline{\nabla}\cdot\mathbf{v}\,ds + \int\limits_{\partial\Gamma(t)} \sigma_s\,\mathbf{v}\cdot\boldsymbol{\nu}\,ds. \tag{4.13}$$

Here, $\underline{\nabla}$ denotes the surface-divergence and $\boldsymbol{\nu}$ indicates the outer normal of the contact line $\gamma(t) := \partial\Gamma(t)$. The latter is a geometry of co-dimension 2 and $\boldsymbol{\nu}$ is defined as the unit vector which is normal to both $\Gamma(t)$ and $\gamma(t)$. When interpreting $\mathbf{v}$ as a *conforming* test function (i.e. $\mathbf{v} \in \left(H^1(\Omega)\right)^d$), it becomes clear that (4.13) can be directly employed for a discretization based only on a (necessarily continuous) approximation of $\Gamma(t)$ but not explicitly dependent on the level set function. The method is - in this sense - purely geometric and thus generally suited for surface tracking methods. Naturally, for any method based on the surface-capturing paradigm, the discrete interface approximation will directly depend on the level set function and the distinction between geometric and direct methods is then of a rather taxonomical nature.

However, for the *non-conforming* DG discretization proposed in section 3.5 on page 62, the surface-divergence theorem has to be applied locally on the intersection of the discrete interface $\Gamma^{h,k}$ with the support of each DG shape function, i.e. on each intersected cell in $\mathcal{T}_+^k$ (for the aligned mesh discretization in section 3.5.4) or $\mathcal{T}$ (for the hybrid mesh discretization in section 3.5.3). Hence, choosing

$$\mathbf{f}_\Gamma\big|_e := \sigma_s\,\kappa_e \tag{4.14}$$

for any given face $e \in \mathcal{E}_\Gamma^k$ with κ_e denoting its local curvature, the linear-form $f^k(\mathbf{v})$ in (3.38) at time t_k is equivalent to

$$f^k(\mathbf{v}) := \sum_{e\in\mathcal{E}_\Gamma^{h,k}} \langle \mathbf{f}_\Gamma, \mathbf{n}\{\!\{\mathbf{v}\}\!\}\rangle_e = \sum_{e\in\mathcal{E}_\Gamma^{h,k}} -\int\limits_e \sigma_s\,\nabla\cdot\mathsf{P}(\mathbf{n})\cdot\{\!\{\mathbf{v}\}\!\} + \int\limits_{\partial e} \sigma_s\,\boldsymbol{\nu}\,\{\!\{\mathbf{v}\}\!\}, \tag{4.15}$$

with

$$\mathsf{P}(\mathbf{n}) := \mathbf{I} - \mathbf{n}\mathbf{n}. \tag{4.16}$$

However, as the faces $\mathcal{E}_\Gamma^{h,k}$ in $\Gamma^{h,k}$ are planar[1] it holds that $\kappa_e = 0$ for all faces $e \in \mathcal{E}_\Gamma^{h,k}$ and thus for the choice (4.14) the form $f^k(\mathbf{v})$ would evaluate to zero independent of the actual interface $\Gamma^{h,k}$. This is again a consequence of the simple fact that the restriction of the discrete level set function ϕ_h^k (or the resulting discrete interface) to a single cell in the geometry mesh $\mathcal{T}_\varkappa$ does not contain enough information for a sufficiently accurate approximation of the interface curvature, as was illustrated in figure 4.1 on page 87. Hence, in analogy to the direct methods, an additional coupling is required to account for kinks in the discrete interface which coincide with cell-faces.

A coupling, suitable for the UDG method, was described by the author of this thesis in [29]. However, it heavily relied on averaged normal vectors $\boldsymbol{\nu}$ of the faces in $\mathcal{E}_\Gamma^{h,k}$,

1 This is true only if the sub-triangulation is based on the partition operator and pre-processing in (2.32). The variant described in (2.30) may in general also entail bi-linear faces with a local curvature greater than zero. However, this does not influence the principal argument as such a local curvature would still be based on a purely local evaluation of the level set function and would not account for kinks in the discrete interface.

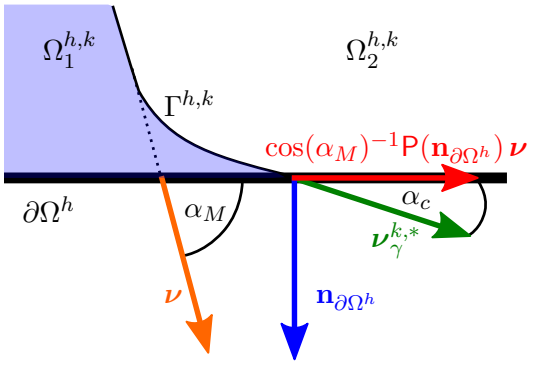


Figure 4.4: The figure illustrates the modification of the interface normal vector $\boldsymbol{\nu}$ at the contact line according to the geometric identity in (4.21). The modified interface normal $\boldsymbol{\nu}_\gamma^{k,*}$ captures the microscopic modification of the interface contact angle. Notice that $\cos\alpha_M = |\mathsf{P}(\mathbf{n}_{\partial\Omega^h})\boldsymbol{\nu}|$.

i.e. for a face $e \in \mathcal{E}_\Gamma^{h,k}$ all its neighbor faces in $\mathcal{E}_\Gamma^{h,k}$ must be determined. Due to the local nature of the sub-triangulation construction of the UDG approach, this information can not be accessed efficiently. Furthermore, for such partition operators which produce sub-triangulations that (may) involve (small) jumps in the discrete interface (see section 3.7 on page 82), the robustness of this approach is dubious.

Alternatively, one may follow the principal idea of the direct projection method (4.11) in section 4.2.2 on page 89 and utilize a projection of the normal field to account for the non-local properties of the discrete interface. Therefore, at time $t = t_k$, a normal field $\mathbf{n}_h^{k,*} \in (Z)^d$ is obtained via the diffusive L_2-projection defined in (4.10) and the linear form in (4.15) is reformulated with regard to this normal field as

$$f^k(\mathbf{v}) = \sum_{e\in\mathcal{E}_\Gamma^{h,k}} -\int_e \sigma_s \nabla\cdot\mathsf{P}(\mathbf{n}^{k,*})\cdot\{\{\mathbf{v}\}\} + \int_{\partial e}\sigma_s\,\boldsymbol{\nu}^{k,*}\,\{\{\mathbf{v}\}\} \tag{4.17}$$

with

$$\boldsymbol{\nu}^{k,*} := \frac{\mathsf{P}(\mathbf{n}^{k,*})\boldsymbol{\nu}}{|\mathsf{P}(\mathbf{n}^{k,*})\boldsymbol{\nu}|}. \tag{4.18}$$

A simple illustration describing the geometric effect of this modification is shown in figure 4.3 on page 92.

As $\mathsf{P}(\mathbf{n}^{k,*})$ depends directly on the discrete level set function, the proposed method is referred to as a *quasi-geometric* method. Notice that it does not involve any projections except for $\mathbf{n}^{k,*}$ which is identical to the normal field required for the recompression in the conservative level set method.

4.4 Surface Tension at the Contact Line

An optimal discretization of the surface tension at the contact line has to account for the peculiarities of its microscopic nature as delineated in section 3.3 on page 55. First among these is the local change of curvature induced by the microscopic bending of the interface enforced by the microscopic contact angle α_c[2]. Revisiting the quasi-geometric

2 The reader is reminded of the fact that for this physical model the interface forces are always supposed to be in equilibrium, i.e. the microscopic phase-distribution in the vicinity of the contact line adjusts

realization of the surface tension in (4.17), it is evident that this microscopic distortion of the interface, which on a macro-scale constitutes an effective discontinuity, may be incorporated into the co-dimension 2 integral by an additional modification of the interface normal $\boldsymbol{\nu}^{k,*}$ in (4.18). Assuming that part of the boundary ∂e of a face $e \in \mathcal{E}$ coincides with the discrete contact line

$$\gamma^{h,k} := \partial\Gamma^{h,k} \tag{4.19}$$

and using

$$|\mathsf{P}(\mathbf{n}_{\partial\Omega^h})\,\boldsymbol{\nu}| = \cos\alpha_M, \tag{4.20}$$

the normal vector on this part of the face boundary may be redefined as

$$\boldsymbol{\nu}^{k,*}\Big|_{\partial e\cap\gamma^{h,k}} := \boldsymbol{\nu}^{k,*}_{\gamma} := \mathsf{P}(\mathbf{n}_{\partial\Omega^h})\,\boldsymbol{\nu}\,\frac{\cos\alpha_c}{\cos\alpha_M} + \mathbf{n}_{\partial\Omega^h}\,\sin\alpha_c. \tag{4.21}$$

Here, $\mathbf{n}_{\partial\Omega^h}$ denotes the normal vector on the touching face of the discrete domain boundary $\partial\Omega^h$ pointing outward of Ω^h. This is nothing but a geometric statement defining $\boldsymbol{\nu}^{k,*}$ as the vector which is normal to the contact line $\partial e\cap\gamma^{h,k}$ and subtends the angle α_c to the domain boundary $\partial\Omega^h$. Therefore, a seamless extension of (4.21) to accommodate more complicated models for *dynamic contact angles*, e.g. [31] should be possible but exceeds the scope of this thesis.

Although this extension of the quasi-geometric discretization of surface tension appears suitable, it was neither analyzed nor implemented for this thesis. The reason for this omission is twofold: First, it is important to realize that in the sub-triangulation algorithm described in chapter 2 does not inherently provide neighborhood information for the faces of the resulting cut-cell meshes, e.g. for a face $e \in \mathcal{E}^k_\Gamma$ it provides neither its neighbors within $\mathcal{E}^k_\Gamma$ nor a possibly touching face in $\partial\Omega^h$ or its normal vector. Naturally, such topological information could be determined in additional geometric computations but the implementation of such extensions to the UDG sub-triangulation have not yet been attempted.

Furthermore, while similar numerical schemes based on such essentially *discontinuous* curvature approximations at the contact line [31, 71] have been shown to provide stable results, for the proof of concept attempted in this thesis, this additional possible source of numerical instability was gladly avoided.

The following alternative numerical approach is based on the modification of the normal field $\mathbf{n}^{k,*}$ which has been employed for the computation of the curvature in both the quasi-geometric method in section 4.3 and the narrow band projection in (4.11). It was mentioned that the projection of $\mathbf{n}^{k,*}$ in (4.11) is actually an elliptic problem and requires suitable boundary conditions. The principle idea is to use Dirichlet boundary conditions in order to enforce the contact angle α_c at the contact line. For the exclusively considered case of *wet* boundaries, i.e. $\alpha_c = 0$, the required boundary condition reduces to

$$\forall\mathbf{x} \in \partial\Omega^h : \mathbf{n}^{k,*} = -\mathbf{n}_{\partial\Omega^h}. \tag{4.22}$$

on a time scale significantly below the characteristic time scale of the macroscopic system.

This implies that the modification (4.21) of the interface normal vector $\boldsymbol{\nu}$ at the contact line is thus equivalent to

$$\boldsymbol{\nu}^{k,*}\Big|_{\partial e \cap \gamma^{h,k}} := \mathsf{P}(\mathbf{n}^{k,*})\,\boldsymbol{\nu}\,\frac{\cos\alpha_c}{\cos\alpha_M} + \mathbf{n}^{k,*}\,\sin\alpha_c. \tag{4.23}$$

which, in the case of $\alpha_c = 0$, reduces to (4.18). Hence, in this case, all special treatment of the contact angle is incorporated by the boundary conditions (4.22) and no further modification is required, neither for the quasi-geometric method nor for the narrow band projection approach.

In spite of its elegant implementation, the aforementioned method is essentially a regularization approach. As $\mathbf{n}^{k,*}$ varies (almost) continuously within Ω^h, the microscopic contact angle α_c is thus not prescribed only at the contact line but varies continuously within a boundary layer the width of which is determined by the artificial diffusivity $\varepsilon_{\mathbf{n}}$ of the normal field projection.

4.5 Interface Recompression at the Contact Line

As mentioned in section 3.6.5 on page 75, the projection of the normal vector field $\mathbf{n}^{k,*}$, which is the key component of the proposed handling of surface tension at the contact line, is constructed by the same projection as the normal vector field $\mathbf{n}_\varphi^{\kappa,*}$ in (3.77) which was used for the recompression (3.75) of the phase-field required for the conservative level set method. However, it is not obvious how to choose the boundary conditions for the diffusive projection of $\mathbf{n}_\varphi^{\kappa,*}$.

One aim of the recompression (3.75) of the phase-field is to keep the interface - and thus its (macroscopic) contact angle - unchanged. The most natural boundary condition for the projection of $\mathbf{n}_\varphi^{\kappa,*}$ is thus

$$\forall \mathbf{x} \in \partial\Omega^h : \mathbf{n}^{k,*} = \nabla\varphi_h^\kappa \tag{4.24}$$

which corresponds to the trace values of the source function of the projection. However, as shown in [106], the convective and diffusive fluxes in the vicinity of the domain boundary may corrupt the original contact angle in the course of the recompression even if appropriate boundary conditions have been enforced. Their analysis and proposed solution indicates that the deformation of the (macroscopic) contact angle is due to spurious fluxes tangential to the interface (which can possibly be alleviated, though not avoided, when $a_\varphi^{\mathbf{n}}$ (3.75) is used in the recompression instead of a_φ (3.79)). However, their derived numerical scheme is restricted to Cartesian meshes and not easily generalized to unfitted domain setups.

At this point, no general solution for the UDG approach in unfitted geometries is proposed. However, for the special case of $\alpha_c = 0$, an interesting result can be achieved by utilizing the boundary condition (4.22) instead of (4.24) for the projection of $\mathbf{n}_\varphi^{\kappa,*}$. In this case, all the recompression fluxes in the vicinity of the domain boundary $\partial\Omega^h$ are

distorted in normal direction to the boundary. Hence, all distortions of the contact angle are such that they will bring the macroscopic contact angle of the discrete interface closer to the microscopic contact angle $\alpha_c = 0$.

While this in itself is rather a relocation than a reduction of the numerical error, for setups with high surface tension and generally slow laminar flows in which the macroscopic contact angle will hardly deviate from the microscopic contact angle, this choice may result in a significant reduction of the global error introduced in the course of a long simulation.

As the chosen boundary conditions for the projection of $\mathbf{n}_{\varphi}^{\kappa,*}$ and $\mathbf{n}^{k,*}$ are in this case identical, the (last) normal vector field utilized for the recompression of the phase-field may thus be recycled for the computation of the surface tension, effectively reducing the numerical effort involved in the solution of the elliptic problem constituted by the diffusive projection of the normal vector field.

5 Numerical Experiments

5.1 On the Implementation

5.1.1 Contributors and Contributions

The software which was employed for the computations presented in this chapter was devised and implemented by the author in the course of his employment at *Heidelberg University* in the course of the project *Zweiphasenströmung in komplex berandeten Gebieten* funded by the *Deutsche Forschungsgemeinschaft* (BA 1498/8-1). However, Galerkin type methods do in general entail a high degree of algorithmic complexity and the UDG approach with its excessive references to multiple hierarchies of geometric objects is unusually challenging. Simulation software which is both robust and flexible enough to fulfill the requirements of academic research can not be written by individual scientists but must be the result of a collective effort. Hence, the simulation software created by the author was based on the following external software libraries and contributions by co-developers:

- The implementation is based on the C++ libraries of the *Distributed and Unified Numerics Environment* (DUNE) which provides diverse elementary algorithms and data structures taylored to the needs of solving partial differential equations. Apart from the DUNE *core-modules* which provide low-level functionality, the author co-developed the high-level discretization module `dune-pdelab` (stable 1.0) [107] and realized an interface which allowed him to utilize its high-level functionality in the context of the UDG approach.

- Together with *Christian Engwer*, the author developed the software module `dune-udg` which realizes the construction of the UDG sub-triangulation and its association with the final cut-cell meshes. The module enforces a highly flexible and consistent interface based on a given implementation of the partition operator [24].

- The implementation of the partition operator was based on the software module `dune-mc` of a volumetric marching-cubes method written by *Andreas Nüßing* and *Christian Engwer*.

5.1.2 Disconnected Cells

The simulations based on the hybrid mesh approach (section 3.5.3 on page 66, e.g. $(\mathbf{X}^k, Y^k) := (\mathbf{P}^M, \mathbf{P}_+^{M-1,k})$) utilized an implementation based on the `dune-pdelab` discretization module while the simulations with the aligned mesh method (section 3.5.4 on page 67, e.g. $(\mathbf{X}^k, Y^k) := (\mathbf{P}_+^{M,k}, \mathbf{P}_+^{M-1,k})$) utilized a different (legacy) module for the discretization, i.e. the assembling of the algebraic sub-problems. While both implementations are essentially equivalent, they differ with regard to the employed cell-merging algorithm.

Although both adhere to the description of the merging process as given in section 2.2.4 on page 36, the legacy variant is restricted in the sense that cell-parts which are situated within the same cell of the fundamental mesh $\mathcal{T}_h$ must also belong to the same cell in the final cut-cell mesh. This implies that for a cell $E \in \mathcal{T}_h$ in the fundamental mesh which is intersected by the interface $\Gamma^{h,k}$ (or the domain boundary $\partial\Omega^h$) such that the

restriction of the interface (or boundary) to the cell does not form a connected manifold the resulting cut-cell mesh $\mathcal{T}_+^k$ (or $\mathcal{T}$) will contain a disconnected cell. Here, the term *disconnected cell* indicates that *one* bounding box $B(E)$ with *one* corresponding set of finite element basis functions (i.e. degrees of freedom) will be defined with regard to a number of disconnected cell snippets simply because these were cut out of the same cell E in the fundamental mesh.

The `dune-pdelab` based variant is more flexible but entails a restriction which requires the number of cells in the final cut-cell mesh $\mathcal{T}_+^k$ (or $\mathcal{T}$) to be equal to or smaller than the number of cells in the fundamental mesh $\mathcal{T}_h$ times the number of subdomains N_Ω. If this is not the case, the algorithm tries to merge more cells in a greedy approach which may also produce the aforementioned *disconnected cells*.

The construction of *disconnected cells* was tracked in log files and for the legacy variant the occurrence was restricted to a number of time steps in the test case 2 of the 2D benchmark problem in section 5.4 where it did not result in any visible or measurable instabilities. For the `dune-pdelab` variant the occurrences were restricted to the random sub-triangulation tests which were exclusively simulated with this variant. The possible influence of these *disconnected cells* is discussed in the corresponding section 5.2.

5.1.3 Solving the Algebraic Problems

The non-linear algebraic sub-problems resulting from the discretizations as described in the previous chapters were solved with Newton's method including a line search utilizing the implementation in the `dune-pdelab` library. All linear algebraic problems were solved with the stabilized bi-conjugate gradient method with an inexact LU preconditioner independent of whether they resulted directly from the discretization or from the steps of Newton's method. For the simulations based on the aligned mesh discretization (section 3.5.4 on page 67), the special block structure (cell-wise blocking of velocity and pressure degrees of freedom) of the matrix was used to improve the computational performance of the linear solver by direct inversion of these block matrices. The same was not realized for the hybrid mesh approach (section 3.5.3 on page 66) where the degrees of freedom of the velocity and pressure field can not be blocked together as they are associated with different meshes.

5.1.4 Finite Element Spaces

Although the definition in (3.21) in section 3.4.1 on page 57 indicates that the local finite element basis was realized by a monomial basis on the reference cube (square) of the bounding boxes, the evaluation of the associated shape functions may be heavily corrupted by numerical errors for polynomial orders beyond $m \leq 3$. Therefore, the local monomial shape functions were employed only for polynomial orders of $m \leq 2$ (and only for the spaces $\Psi(\mathbf{P}_+^{m,k})$ and $\Psi(\mathbf{P}^m)$ which were never used for higher polynomial orders). For the spaces $\Psi(\mathbf{Q}_+^{m,k})$ and $\Psi(\mathbf{Q}^m)$, the local shape functions required for $\Psi(\mathbb{P}^m)$ in (3.21) were instead defined by the Q^m standard finite element basis for hexahedral (quadrilateral) meshes (e.g. [83]) on the geometry of the reference cube (square) of the bounding boxes.

Two Subdomain Setup with Random Level Set Function ϕ_1^R

	$m=2,\ L=5,\ \sigma_{IP}=1$				$m=2,\ L=6,\ \sigma_{IP}=1$				$m=3,\ L=6,\ \sigma_{IP}=1$			
	Err_2	rate	DOF	DC/C	Err_2	rate	DOF	DC/C	Err_2	rate	DOF	DC/C
Level 1	2.53e-02	0.0	90	0.50	2.44e-02	0.0	130	0.56	9.06e-03	0.0	260	0.56
Level 2	1.68e-02	0.7	560	0.18	9.73e-03	1.5	780	0.48	1.72e-03	2.8	1560	0.48
Level 3	6.17e-03	2.1	2400	0.00	7.69e-03	0.4	3960	0.14	1.41e-03	0.4	7920	0.14
Level 4	1.45e-04	5.8	16550	0.00	8.20e-04	4.6	16940	0.00	8.98e-05	5.7	33880	0.00
Level 5	1.32e-05	3.5	129920	0.00	2.41e-05	5.2	128290	0.00	1.18e-06	6.4	256580	0.00
Level 6					2.38e-06	3.3	1033510	0.00	1.31e-06	-0.2	2067020	0.00

Four Subdomain Setup with Random Level Set Functions ϕ_1^R, ϕ_2^R

	$m=2,\ L=5,\ \sigma_{IP}=1$				$m=2,\ L=6,\ \sigma_{IP}=1$			
	Err_2	rate	DOF	DC/C	Err_2	rate	DOF	DC/C
Level 1	2.61e-02	0.0	320	1.00	2.54e-02	0.0	320	1.00
Level 2	2.13e-02	0.3	2560	0.97	2.45e-02	0.1	2560	1.00
Level 3	6.58e-03	2.1	13720	0.21	2.10e-02	0.2	20480	0.98
Level 4	2.93e-04	11.0	32000	0.01	4.57e-03	2.9	99530	0.16
Level 5	2.56e-05	5.1	135270	0.00	4.74e-05	16.4	229360	0.01
Level 6					1.45e-05	2.3	1051990	0.00

Two Subdomain Setup with Periodic Level Set Function ϕ_1^P

$D=2, Sin$	$m=2,\ L=5,\ \sigma_{IP}=1$				$m=2,\ L=6,\ \sigma_{IP}=1$				$m=3,\ L=6,\ \sigma_{IP}=1$			
	Err_2	rate	DOF	DC/C	Err_2	rate	DOF	DC/C	Err_2	rate	DOF	DC/C
Level 1	2.63e-02	0.0	120	0.62	2.39e-02	0.0	160	0.94	1.00e-02	0.0	320	0.94
Level 2	8.34e-03	2.4	500	0.10	1.57e-02	0.9	680	0.27	4.69e-03	1.6	1360	0.27
Level 3	6.54e-03	0.6	1700	0.00	9.67e-03	1.8	1560	0.04	2.16e-03	2.8	3120	0.04
Level 4	3.28e-04	4.2	14050	0.00	3.15e-03	2.1	8050	0.01	5.07e-04	2.6	16100	0.01
Level 5	1.94e-05	3.9	122640	0.00	5.11e-04	2.4	80260	0.00	4.33e-05	3.2	160520	0.00
Level 6					4.70e-05	2.7	1089370	0.00	4.04e-06	2.7	2178740	0.00

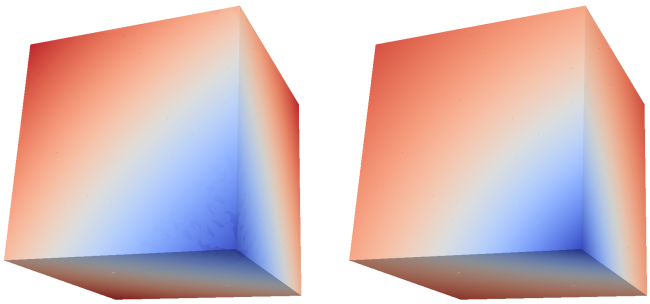
Four Subdomain Setup with Periodic Level Set Function ϕ_1^P, ϕ_2^P

$D=4, Sin$	$m=2,\ L=5,\ \sigma_{IP}=1$				$m=2,\ L=6,\ \sigma_{IP}=1$			
	Err_2	rate	DOF	DC/C	Err_2	rate	DOF	DC/C
Level 1	2.63e-02	0.0	320	1.00	2.27e-02	0.0	320	1.00
Level 2	1.97e-02	0.4	2560	0.96	2.11e-02	0.1	2560	1.00
Level 3	3.88e-03	2.7	15710	0.48	2.04e-02	0.0	20430	0.99
Level 4	1.29e-03	2.9	49780	0.05	1.02e-02	1.1	127710	0.59
Level 5	3.77e-04	3.4	146170	0.00	6.80e-03	1.0	437050	0.10
Level 6					4.58e-04	8.1	1184510	0.00

Table 5.1: The tables above show the convergence behavior of the discrete solutions $u \in \mathbf{P}_l^m$ of the Laplace problem computed by the interior penalty method (5.3) on the sequence of cut-cell meshes $\mathcal{T}^l$. The cut-cell mesh on level l is constructed for the fundamental meshes $\mathcal{T}_h^l$ with $h = 2^{-l}$ for $l \in \{1, \dots, L\}$ and the subdomain geometries described in section 5.2 and illustrated in figure 5.2. The tables list the L_2-error Err_2 of the solution, the number of degrees of freedom (DOF) and the ratio of disconnected cells to the total number of cells (DC/C) for different combinations of the polynomial order m and the resolution $\varkappa = 2^{-L}$ of the geometry mesh $\mathcal{T}_\varkappa$. In general, convergence can only be achieved on the last few mesh levels and is correlated to the number of disconnected cells. Notice that the solutions on the coarsest mesh do already constitute *good* approximations, compare figure 5.1.

5.2 Random-Domain Sub-Triangulation Tests

In the early development stage of the multi-domain sub-triangulations, test simulations were performed in order to validate the robustness of the underlying algorithms and their implementation. Therefore, some *worst-case* domains were realized based on level set functions returning (pseudo-)random values. In addition to the validation routine which checked various consistency conditions for the cell-parts and face-parts of the sub-triangulation, the resulting cut-cell meshes were employed for solving the Laplace equation with Dirichlet boundaries corresponding to some well known non-polynomial analytic solution. This section documents the errors of the obtained approximations and



Mesh Level 1 Mesh Level 5

Figure 5.1: The pictures show the discrete solution to the test problem (5.3) computed on the coarsest and the finest mesh level for the periodic domain with $m = 2$, $L = 5$ and $N_\Omega = 4$. Although the difference is visible, the coarse solution provides a good approximation considering that it is based on a cut-cell mesh with five cells, each consisting of thousands of disconnected patches.

thus delineates both the possibilities and the limitations of numerical upscaling in the limit of simplistic model equations with extremely complex domain geometries.

Two different choices for the level set functions are considered. In both cases, the functions are represented by a standard Q^1 conforming finite element basis defined by the scalar values in the nodes of the geometry mesh $\mathcal{T}_\varkappa$ such that for a given node $\mathbf{x}_n$:

$$\phi_i(\mathbf{x}_n) := \begin{cases} 0 & \text{if } |f_i(\mathbf{x}_n)| < \varepsilon_R, \\ f_i(\mathbf{x}_n) & \text{otherwise.} \end{cases} \tag{5.1}$$

For the first candidate ϕ_i^R (*random domain*), the scalar functions f_i are given by a random number generator which provides unique random numbers in the range $[-1, +1]$ for each node in $\mathcal{T}_\varkappa$. For the second candidate (*periodic domain*) ϕ_i^P, a periodic function

$$f_i(\mathbf{x}_n) = \sin\left(\frac{|\mathbf{x}_n|}{10^-3} + i\right) \tag{5.2}$$

is used to provide an alternative domain setup. In both cases, large values of the threshold parameter $\varepsilon_R \in [0, 1]$ increase the likelihood for zero values and thus provoke the construction of very thin anisotropic cells. The computational domain Ω^h is chosen as the three-dimensional unit cube and cut-cell meshes. The candidate level set functions ϕ_i^R and ϕ_i^P were applied both in a two-domain setup ($i \in \{1\}$) and a four-domain setup ($i \in \{1, 2\}$). For the two-domain setup, the threshold parameter was chosen as $\varepsilon_R = 0.3$ while for the four-domain setups it was reduced to $\varepsilon_R = 0.01$ to account for the higher probability of the occurrence of disconnected cells in the four domain setup. The resulting domains are shown in figure 5.2 on the facing page.

The merging procedure (see section 2.2.4 on page 36) was applied to each cell E in the preliminary cut-cell mesh $\mathcal{T}^*$ until a minimum size of h^d is achieved which corresponds to the cell size in the fundamental mesh. This rather high value is intended to reduce the number of disconnected cells (section 5.1.2) which are bound to appear in such pathological geometries.

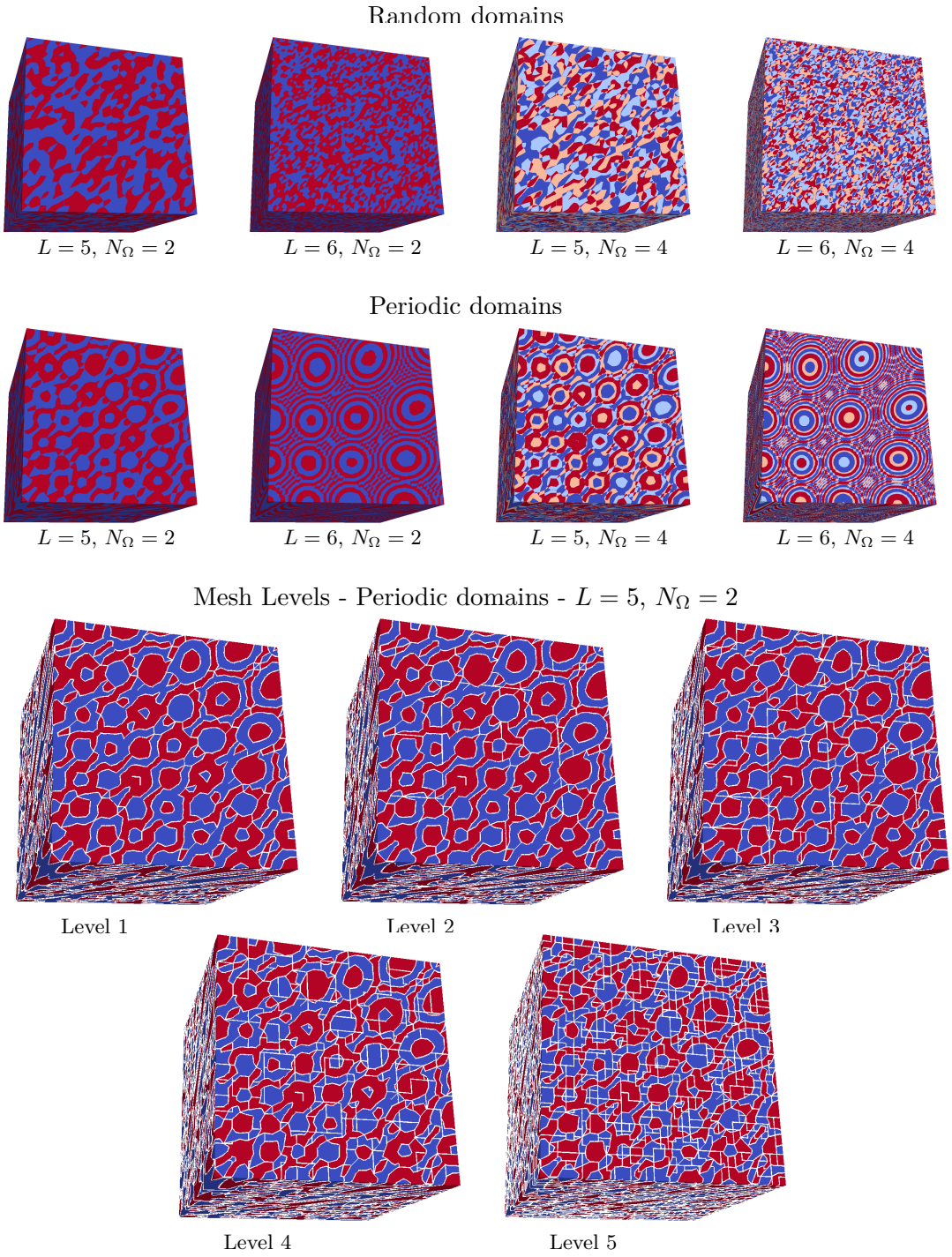


Figure 5.2: The upper pictures show the subdomains on the surface of the domain for which the convergence of the Laplace problem according to section 5.2 is analyzed. Notice that the subdomains are employed only to provide a sufficiently ugly mesh while the Laplace problem itself is homogeneous. The geometries are defined on the geometry mesh $\mathcal{T}_\varkappa$ with $\varkappa = 2^{-L}$.
In the lower pictures, the white lines indicate the edges of the cut-cell meshes $\mathcal{T}^l$ on mesh level l constructed for the fundamental meshes $\mathcal{T}_h^l$ with $h = 2^{-l}$ for $l \in \{1, \dots, L\}$.

Computations were performed for geometry meshes with $\varkappa = 2^{-L}$ for $L = 5$ and $L = 6$. In each case, approximate solutions were computed for cut-cell meshes $\mathcal{T}^l$ constructed for fundamental meshes $\mathcal{T}_h^l$ with $h = 2^{-l}$ for $l \in \{1, \dots, L\}$.

With regard to the resulting cut-cell mesh $\mathcal{T}^l$, the broken polynomial space $\mathbf{P}_l^m$ are defined in analogy to the definition (3.17) for the two-phase flow problem. The Laplace model problem is then discretized by the interior penalty method according to:

$$\begin{aligned} &\forall v_h \in \mathbf{P}_l^m : \\ &\quad \sum_{E \in \mathcal{T}} -(\nabla u, \nabla v_h)_E + \\ &\quad \sum_{e \in \mathcal{E}} \langle \{\{\nabla u\}\}, \mathbf{n}[\![v_h]\!] \rangle_e - \epsilon \langle \{\{\nabla v_h\}\}, \mathbf{n}[\![u]\!] \rangle_e + \sigma_{IP}^u \langle [\![u]\!], [\![v_h]\!] \rangle_e = 0. \end{aligned} \tag{5.3}$$

The jump terms are supposed to incorporate the Dirichlet boundary conditions of the analytical solution

$$u_D(\mathbf{x}) := (|\mathbf{x}|)^{-\frac{1}{2}} . \tag{5.4}$$

Notice that this corresponds to a homogeneous diffusion problem on the unit cube with an extremely ugly mesh as defined by the level set function(s). The convergence behavior of the L_2-error

$$Err_2 := \|u - u_D\|_2 \tag{5.5}$$

is shown in table 5.1 on page 101. It is evident, that convergence is generally observed only if $\varkappa/h \leq 2^3$ which gives an idea of the limits of numerical upscaling on such pathological domains. It is noteworthy that there is an apparent correlation between the number of disconnected cells (see section 5.1.2) and the convergence behavior, i.e. convergence can be observed not before the ratio of disconnected cells to total cells has fallen sufficiently. However, as the number of disconnected cells does itself depend on the ratio $\varkappa/h$, there is no way to determine whether this is not just a mere correlation. In any case, both the high rates of convergence always obtained on the last few mesh levels and the total L_2-errors indicate the consistency of the sub-triangulation.

5.3 Surface Tension

Among the discretization methods for surface tension as described in chapter 4, only the quasi-geometric method described in section 4.3 on page 92 and the projection methods in section 4.2.2 on page 89 are directly applicable to the UDG two-phase flow discretization in unfitted domains. For these methods, a numerical comparison was performed and the results are presented in the current section.

The definition of a suitable benchmark problem for the comparison of the relative performance of different methods for the discretization of surface tension is not straightforward. A good choice would optimally allow to separate the errors in the approximation of the surface tension force from other spurious influences of the two-phase flow discretization. However, while the projection methods provide curvature fields which can be evaluated at any point of the discrete interface and thus be compared to each other, the quasi-geometric method is embedded in the discretization of the Navier-Stokes equations.

Quasi-Geometric Method ($\varepsilon_\kappa = 0.01\,h^2$, $\varepsilon_{\mathbf{n}} = 0.1\,h^2$, $\varepsilon_\varphi = 0.3\,h$):

ΔP^{err}_{avg}	λ=3		λ=2		λ=1	
Level 2	2.172e+00		2.179e+00		1.803e+00	
Level 3	2.982e-01	2.9	5.061e-01	2.1	1.254e+00	0.5
Level 4	1.111e-01	1.4	1.754e-01	1.5	4.391e-01	1.5
Level 5	2.912e-02	1.9	4.996e-03	5.1	1.765e-01	1.3

P^{err}	λ=3		λ=2		λ=1	
Level 2	8.885e+00		5.892e+00		1.752e+01	
Level 3	1.102e+01		9.467e+00		1.403e+01	0.3
Level 4	3.026e+01		2.325e+01		2.176e+01	
Level 5	3.347e+01		3.143e+01		4.543e+01	

$\|\mathbf{v}\|_2$	λ=3		λ=2		λ=1	
Level 2	1.784e-02		1.758e-02		2.719e-02	
Level 3	5.261e-03	1.8	4.665e-03	1.9	7.124e-03	1.9
Level 4	5.699e-03		5.155e-03		5.541e-03	0.4
Level 5	5.520e-03		4.896e-03	0.1	5.586e-03	

Narrow Band Projection Method ($\varepsilon_\kappa = 0.04\,h^2$, $\varepsilon_{\mathbf{n}} = 0.4\,h^2$, $\varepsilon_\varphi = 0.6\,h$, $\beta = 0.08$):

ΔP^{err}_{avg}	λ=3		λ=2		λ=1	
Level 2	3.94e+01		4.10e+01		4.67e+01	
Level 3	1.95e+01	1.0	1.97e+01	1.1	2.09e+01	1.2
Level 4	4.56e+00	2.1	4.60e+00	2.1	4.89e+00	2.1
Level 5	3.06e-01	3.9	3.27e-01	3.8	2.49e-01	4.3

P^{err}	λ=3		λ=2		λ=1	
Level 2	9.85e+00		9.33e+00		8.14e+00	
Level 3	3.83e-01	4.7	9.43e-01	3.3	1.24e+00	2.7
Level 4	2.91e+00		3.00e+00		6.80e+00	
Level 5	2.62e+00	0.2	2.79e+00	0.1	8.56e-01	3.0

$\|\mathbf{v}\|_2$	λ=3		λ=2		λ=1	
Level 2	2.75e-03		2.73e-03		3.60e-03	
Level 3	7.75e-04	1.8	8.19e-04	1.7	1.16e-03	1.6
Level 4	3.33e-04	1.2	3.18e-04	1.4	8.12e-04	0.5
Level 5	2.09e-04	0.7	1.06e-04	1.6	9.10e-05	3.2

Global Projection Method ($\varepsilon_\kappa = 0.04\,h^2$, $\varepsilon_{\mathbf{n}} = 0.4\,h^2$, $\varepsilon_\varphi = 0.6\,h$):

$\Delta P^{err}_{avg}(t=T)$	λ=3		λ=2		λ=1	
Level 2	2.53e+01		2.46e+01		2.15e+01	
Level 3	4.46e-01	5.8	7.03e-01	5.1	1.18e+00	4.2
Level 4	1.98e+00		2.02e+00		2.23e+00	
Level 5	6.98e-01	1.5	6.92e-01	1.5	6.17e-01	1.9

P^{err}	λ=3		λ=2		λ=1	
Level 2	4.18e-01		5.97e-01		2.85e+00	
Level 3	1.05e+00		1.39e+00		6.60e+00	
Level 4	8.25e-01	0.4	9.44e-01	0.6	9.17e-01	2.8
Level 5	8.54e-01		1.05e+00		1.38e+00	

$\|\mathbf{v}\|_2$	λ=3		λ=2		λ=1	
Level 2	3.83e-04		4.86e-04		2.61e-03	
Level 3	1.15e-03		1.17e-03		1.33e-03	1.0
Level 4	1.68e-04	2.8	2.11e-04	2.5	2.30e-04	2.5
Level 5	1.23e-04	0.4	1.24e-04	0.8	8.98e-05	1.4

Table 5.2: Benchmark quantities for the two-dimensional surface tension test case: Errors of the jump in the average pressure ΔP^{err}_{avg} (5.7), maximum deviation of the average pressure P^{err} (5.8) and L_2-norm of the velocity field $\|\mathbf{v}\|_2$. Each value is followed by the rate of convergence in h relative to the previous mesh level (omitted where it would be negative). For each method, the regularization parameters where chosen to optimize the approximation of the average pressure jump. Notice that some methods are very sensitive with regard to the parameter choices, compare tables A.1, A.2, and A.3 in the appendix.

Quasi-Geometric Method ($\varepsilon_\kappa = 0.01\,h^2$, $\varepsilon_{\mathbf{n}} = 0.1\,h^2$, $\varepsilon_\varphi = 0.5\,h$):

$\Delta P^{err}_{avg}(t=T)$	λ=2		λ=1	
Level 2	3.93e+00		2.85e+00	
Level 3	1.65e+00	1.3	3.82e-01	2.9
Level 4	6.28e-01	1.4	5.24e-02	2.9

$P^{err}(t=T)$	λ=2		λ=1	
Level 2	8.80e+00		1.40e+01	
Level 3	3.90e+00	1.2	6.70e+00	1.1
Level 4	5.05e+00		3.88e+00	0.8

$\|\mathbf{v}\|_2(t=T)$	λ=2		λ=1	
Level 2	7.33e-04		1.60e-03	
Level 3	3.56e-04	1.0	6.37e-04	1.3
Level 4	5.23e-04		5.85e-04	0.1

Narrow Band Projection Method ($\varepsilon_\kappa = 0.04\,h^2$, $\varepsilon_{\mathbf{n}} = 0.4\,h^2$, $\varepsilon_\varphi = 0.6\,h$, $\beta = 0.08$):

$\Delta P^{err}_{avg}(t=T)$	λ=2		λ=1	
Level 2	6.45e+00		2.50e+00	
Level 3	2.27e+00	1.5	4.67e+00	
Level 4	5.28e-01	2.1	1.10e+00	2.1

$P^{err}(t=T)$	λ=2		λ=1	
Level 2	1.65e+01		1.73e+01	
Level 3	1.37e+00	3.6	1.58e+00	3.4
Level 4	1.40e+00		1.88e+00	

$\|\mathbf{v}\|_2(t=T)$	λ=2		λ=1	
Level 2	2.89e-03		2.90e-03	
Level 3	1.66e-04	4.1	2.13e-04	3.8
Level 4	6.13e-05	1.4	7.51e-05	1.5

Global Projection Method ($\varepsilon_\kappa = 0.04\,h^2$, $\varepsilon_{\mathbf{n}} = 0.4\,h^2$, $\varepsilon_\varphi = 0.6\,h$):

$\Delta P^{err}_{avg}(t=T)$	λ=2		λ=1	
Level 2	4.13e+01		6.25e+01	
Level 3	8.47e+00	2.3	6.43e+00	3.3
Level 4	2.37e+00	1.8	1.78e+00	1.9

$P^{err}(t=T)$	λ=2		λ=1	
Level 2	1.88e+00		1.96e+00	
Level 3	9.14e-01	1.0	1.15e+00	0.8
Level 4	1.35e+00		2.00e+00	

$\|\mathbf{v}\|_2(t=T)$	λ=2		λ=1	
Level 2	2.15e-04		1.52e-04	
Level 3	9.75e-05	1.1	1.67e-04	
Level 4	6.03e-05	0.7	7.42e-05	1.2

Table 5.3: Benchmark quantities for the three-dimensional surface tension test case: Errors of the jump in the average pressure ΔP^{err}_{avg} (5.7), maximum deviation of the average pressure P^{err} (5.8) and L_2-norm of the velocity field $\|\mathbf{v}\|_2$. Each value is followed by the rate of convergence in h relative to the previous mesh level (omitted where it would be negative). For each method, the regularization parameters where chosen to optimize the approximation of the average pressure jump.

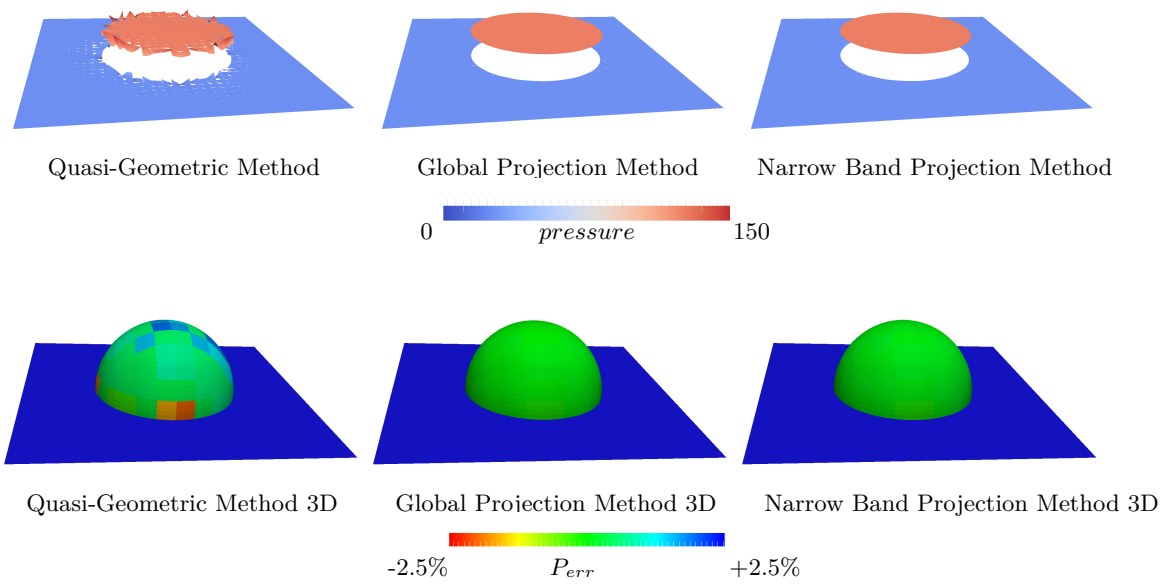


Figure 5.3: The pictures depict the pressure field for each of the three different surface tension discretization methods on a fundamental mesh with (32×32) cells in the two-dimensional case and $(16 \times 16 \times 16)$ in the three-dimensional case with the respective finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$ in 2D and $(Z, \mathbf{X}, Y) := (\mathbf{Q}^2, (\mathbf{Q}^1)^3, \mathbf{Q}_+^{0,k})$ in 3D. In both cases the pressure fluctuations for the quasi-geometric method are significantly higher than for the projection methods. In three dimensions, the fluctuations are smaller and the color range delineates a 2.5% fluctuation in the pressure. On the other hand, the color range used for the 2D case spans the whole range of pressure values.

On the other hand, as was shown in [104], the actual quality of the surface force approximation and thus the magnitude of the resulting *spurious velocities* (fluxes resulting from local errors in the surface force) may depend more on the balanced approximation quality of e.g. the pressure field relative to the curvature field than on the approximation quality of the curvatures itself.

Therefore, the comparison presented in this section is essentially a full two-phase flow simulation of a quasi-stationary setup given by an initially spherical (circular) distribution of phase 1 with radius $R = 0.25$ (corresponding to subdomain Ω_1) included in phase 2 (corresponding to Ω_2), the latter filling the rest of the rectangular domain Ω_r given by the unit square (unit cube). The physical parameters of the two fluids were chosen as

	Ω_1	Ω_2
ρ	100	1000
μ	1	10
σ_s	24.5	

(5.6)

which corresponds to the physical parameters of test case 1 of the 2D benchmark setup

in section 5.4.2.

In order to assess the quality of the discretization of surface tension, the complete two-phase flow problem is solved for the time interval $[0, T]$. For the last five time steps of each propagation, average values of the benchmark quantities are computed. The latter include the error in the jump of the average pressures in each phase, i.e.

$$\Delta P_{avg}^{err} := \left| \frac{(d-1)\,\sigma_s}{R} - \left| \overline{P}_1 - \overline{P}_2 \right| \right| \text{ with } \overline{P}_i := \frac{1}{|\Omega_i^h|} \int_{\Omega_i^h} p \, dx, \tag{5.7}$$

the maximum deviation from the pressure average value

$$P^{err} := \max_{i \in \{1,2\},\, \mathbf{x} \in \Omega_i^h} \left| \overline{P}_i - p \right| \tag{5.8}$$

and the standard L_2-norm of the global velocity field. As the setup essentially describes a stationary equilibrium of the two-phase flow problem, all velocities are considered erroneous. Notice that although the deformation of the initial interface in the course of the propagation is not explicitly tracked, it is expected that such a deformation should inevitably deteriorate the accuracy of the jump in the average errors.

The test cases are restricted to the conservative level set method. Even so, the three discretizations depend on a number of regularization parameters the optimal choice of which is not obvious and difficult to derive from theoretical considerations. In all cases, these parameters include the diffusivity $\varepsilon_{\mathbf{n}}$ (3.77) for the projection of the normal field and the bandwidth parameter ε_φ for the recompression of the phase-field (3.72). For the projection method there is an additional parameter ε_κ determining the diffusivity of the curvature projection in (4.11). For each of the discretization methods, values for each of these parameters were hand-picked in order to provide nearly optimal conditions. Furthermore, the sensitivity in all of these parameters was estimated in additional runs.

For each of the parameter configurations, simulations were performed on a number of consecutive mesh levels corresponding to the cut-cell meshes $\mathcal{T}^l$ determined by the fundamental mesh $\mathcal{T}_h^l$ with $h = 2^{-l}$ and the geometry mesh $\mathcal{T}_\varkappa$ with $\varkappa = 2^{-l-\lambda}$. The parameters of the time discretization were chosen to allow a stable simulation with reasonable runtimes (maximum of a few days). The time step size was chosen as $\Delta t = 1.25 \cdot 10^{-2} \cdot h$ with a total time of $T = 0.375$ for the two-dimensional case and $\Delta t = 4 \cdot 10^{-3} \cdot h$ with a total time $T = 0.01$ for the three-dimensional case.

The virtual time step size for the recompression was chosen as $\Delta\tau = 0.1\,h$ and the recompression limit in (3.80) was chosen as $\delta_\varphi = 0.05\,h$. The sensitivity of the results on these two parameters was not analyzed systematically as sample computations indicated it to be generally insignificant as long as δ_φ is reasonably small.

The results for the two-dimensional case are presented in table 5.2 on page 105. In each case, the default parameters were chosen as to optimally reduce the error of the average pressure jump ΔP_{avg}^{err} even if higher values for a respective diffusivity parameter would have resulted in reduced spurious velocities. Notice that especially the projection methods are highly sensitive to the choice of the regularization parameters. The sensitivity of each of the methods was analyzed in additional simulations, see tables A.1, A.2, and A.3 on page 151 in the appendix.

The data reveals that the approximation of the average pressure jumps is best for the quasi-geometric method but seems to deteriorate for $\lambda < 2$.

Both the quasi-geometric and the global projection method do not show any convergence with regard to the maximum pressure deviation and the spurious velocity norms. This is not completely unexpected considering that the geometric time step restriction based on the magnitude of capillary pressure waves [100] requires $\Delta t \propto h^{3/2}$ while in this test case the time step was chosen proportional to h. The more remarkable is the fact that the projection methods appears to result in spurious velocity fields which are generally small and appear to converge with $h \to 0$. Also the maximum pressure deviations are significantly smaller for the narrow band projection method than for any of the other methods. This behavior is shown in figure 5.3.

5.4 Rising Bubble Simulations

A comprehensive quantitative benchmark problem for two-dimensional incompressible two-phase flow was published in [34] including the computational results of three different numerical discretizations implemented by three independent groups of scientists. The simulated flow setup was given by the rise of an initially circular bubble due to buoyancy in two distinct parameter configurations corresponding to a high and a low capillarity setup. Other researchers have followed the example and published (at least partially) the corresponding benchmark results computed with their numerical schemes, e.g. [108, 109, 110, 111]. In the course of this thesis, the aligned mesh discretization (section 3.5.3) method has been employed to compute the benchmark problem and the corresponding benchmark results were published in [35]. In sections 5.4.2 and 5.4.3, these computations are analyzed and compared to computations based on the hybrid mesh discretization (section 3.5.4).

In section 5.4.4, results for three-dimensional simulations of rising bubbles are presented. While comprehensive quantitative 3D benchmark problems similar to [34] are not available for the three-dimensional case, some exemplary computations have been performed in [102, 112] for setups defined by the dynamic regime according to [113], i.e. the Morton and Eötvös numbers, and some agreement was found, at least for the general bubble shapes and the resulting Reynolds numbers.

Unless explicitly defined, the non-physical discretization parameters for all of the following examples were chosen as:

$$\Delta\tau = 0.1\,h,\ \delta_\varphi \leq 0.05,\ \varepsilon_{\mathbf{n}} = 0.1h^2,\ \varepsilon_\kappa = 0.01h^2,\ \varepsilon_\varphi = 0.5\,h,\ \beta = 0.25. \tag{5.9}$$

Also, as soon as the finite element space was chosen with a polynomial order equal or higher than in $\mathbf{Q}^3$, the diffusivity parameter for the phase field recompression was decreased to $\varepsilon_\varphi = 0.3\,h$ which reduces the number of required recompression steps. Notice that this choice turned out to be unstable for finite element spaces of lower polynomial order. As the simulation in this section were performed *before* the simulations in the previous section which were used to manually tune the regularization parameters, the

Test case	ρ_1	ρ_2	μ_1	μ_2	g	σ_s	Re	E_o	ρ_1/ρ_2	μ_1/μ_2
1	1000	100	10	1	0.98	24.5	35	10	10	10
2	1000	1	10	0.1	0.98	1.96	35	125	1000	100

Table 5.4: Physical parameters and dimensionless numbers defining the two test cases. The Eötvös numbers of both cases differ by an order of magnitude indicating that, in the first test case, the surface tension is more significant compared to the buoyancy forces.

	$1/h$	$N_{\mathcal{T}_h}$	N_k	$DOF((\mathbf{P}^2)^2, \mathbf{P}^1)$	$DOF((\mathbf{P}^2)^2, \mathbf{P}_+^{1,k})$	$DOF((\mathbf{P}^1)^2, \mathbf{P}^0)$	$DOF((\mathbf{P}^1)^2, \mathbf{P}_+^{0,k})$
Level 1	10	200	480	3000	2973	1400	1391
Level 2	20	800	960	12000	11940	5600	5580
Level 3	40	3200	1920	48000	47868	22400	22356
Level 4	80	12800	3840	192000	191760	89600	89520
Level 5	160	51200	7680	768000	-	358400	-

Table 5.5: The table above defines the mesh levels, i.e. the sequence of fundamental meshes, defined by the mesh resolution h and the number of cells $N_{\mathcal{T}_h}$ with corresponding time steps size $\Delta t = T/N_k$ (with $T = 3$) for which the 2D rising bubble benchmark problem was solved. Furthermore, the number of degrees of freedom (DOF) for some finite element function spaces $(\mathbf{X}, Y)$ for the velocity and pressure fields are given on all levels. For the time dependent function spaces $((\mathbf{P}^M)^2, \mathbf{P}_+^{M-1,k})$, these values have been taken from the simulation of the test case 1 and averaged over $[0, T]$. Notice that the spaces $((\mathbf{P}^M)^2, \mathbf{P}^{M-1})$ are actually never used for the computations presented in this thesis but just give an idea of how many DOF would be required by a *standard* discontinuous Galerkin method on the same fundamental mesh $\mathcal{T}_h$. It is evident that a UDG method that employs a reasonable merging procedure will in average require less DOF and is thus conceptually different and should not be mistaken for any kind of adaptive refinement method.

choices in (5.9) are close but not equal to the *optimal* settings presented in section 5.3 and may rather favor the quasi-geometric discretization method for the surface tension.

5.4.1 2D Benchmark

The two test cases considered in the two-dimensional benchmark [34] differ only in the physical parameters given in table 5.4. The general setup in both cases is given by a rectangular domain $\Omega = (0, 1) \times (0, 2)$ filled by two fluids within the respective subdomains $\Omega_1(t)$ and $\Omega_2(t)$. At time $t = 0$, the latter is given by a circle with radius $R = 0.25$ and barycenter at $(0.5, 0.5)$. In both cases, Ω_2 is filled with the lighter fluid and this two-dimensional *bubble* is thus rising from its initial position. No-slip boundary conditions are applied at the upper and lower boundary while full-slip conditions are applied at the sides. The simulated time interval $t \in [0, T]$ with $T = 3$ of the benchmark examples is chosen such that the bubble does not reach the domain boundary and thus the numerical implications of three-phase contact lines need not be considered.

In its original publication, the time development of three benchmark quantities are

evaluated. These comprise the bubble's center of mass, its rise velocity

$$u^Y(t) := \frac{1}{|\Omega_2|} \int_{\Omega_2} \mathbf{u}(t, \mathbf{x}) \cdot \mathbf{e}_y \, dx \tag{5.10}$$

as well as the bubble's circularity c which describes the inverse ratio of the bubble's surface area (i.e. surface length) to the surface area of a spherical (circular) bubble of the same volume:

$$c := \frac{2\pi R}{|\partial\Omega_2(t)|}. \tag{5.11}$$

In order to estimate the convergence properties of the respective discretization, the benchmark problems were solved on a series of fundamental meshes $\mathcal{T}_h^l$ with $(l = 1, \ldots, 4)$ and simulations were performed with a respectively refined time step size Δt. In the following, a simulation based on such a fundamental mesh $\mathcal{T}_h^l$ with $5 \cdot 2^l \times 10 \cdot 2^l$ cells and corresponding time step size will be referred to as a simulation on (mesh) level l. The relative refinement of the geometry mesh $\mathcal{T}_\varkappa$ with $5 \cdot 2^{l+\lambda} \times 10 \cdot 2^{l+\lambda}$ cells is determined by the parameter λ. The temporal and spatial resolutions on each level are given in table 5.5 on the preceding page together with some reference values for the corresponding degrees of freedom.

The quantitative nature of the benchmark allows for a meaningful comparison of different numerical methods. A comparison between the standard level set method (SLS) and the conservative level set method (CLS) is presented in the following section for both test cases. For the latter, additional simulations were performed in order to estimate the potential of p-refinement and the impact of different methods for the discretization of the surface tension.

Naturally, a comparison of the aligned mesh discretization method (section 3.5.4) and the hybrid mesh discretization method (section 3.5.3) for the Navier-Stokes equations independent of the discretization methods for the level set equation would have been highly desirable. Sadly, this was not possible[1]. Instead, for the presented simulations, the SLS method was always combined with the aligned mesh approach while the CLS method was always combined with the hybrid mesh approach.

Although this does not allow a clean evaluation of the respective performance of each of these discretizations individually, there are factors which indicate that the ability of the aligned mesh method to represent jumps in the velocity derivatives does not provide a significant advantage - at least with regard to the chosen benchmark quantities: The groups which provided the results for the original publication [34] employed, without exception, methods which regularized both the kink in the velocity field and the jump of the pressure field at the interface but yet achieved higher-order convergence in the benchmark quantities.

As the amount of data presented in this section is extensive by any standards, the least sensitive of the three benchmark quantities, the center of mass, has been dropped and is

1 The implementation of the aligned mesh method was created to realize but a proof of concept for the *fitted* case ($\Omega^h = \Omega = \Omega_r$) with a structured mesh in which the standard level set method can be directly applied. For the final implementation which was tailored for the application on unfitted domains, only the Conservative Level Set Method and the hybrid mesh approach were implemented.

not considered in the further analysis in order to simplify the extraction of the meaningful results. However, for some of the simulations presented in the following sections, these values may be found in [35].

5.4.2 2D Benchmark - High Surface Tension - Case 1

The movement of the rising bubble in test case 1 is illustrated in figure 5.4. Evidently, the deformations are rather benign and after the initial formation of a dimple on its down-wind side the bubble eventually takes on a convex shape. It is noteworthy that these pictures would look almost identical for all mesh levels > 1. This gives already a crude idea of the principal performance of the presented methods.

The graphs of the benchmark quantities in figure 5.5 on the next page and their corresponding error norms (over $[0, T]$) in table 5.7 on page 118 allow a more quantitative assessment of the relative performance of the standard level set method (SLS, with aligned meshes) and the conservative level set method (CLS, with hybrid meshes) for two sets of finite element spaces, i.e. $(\mathbf{P}^M, (\mathbf{P}_+^{M,k})^2, \mathbf{P}_+^{M-1,k})$ for aligned meshes and

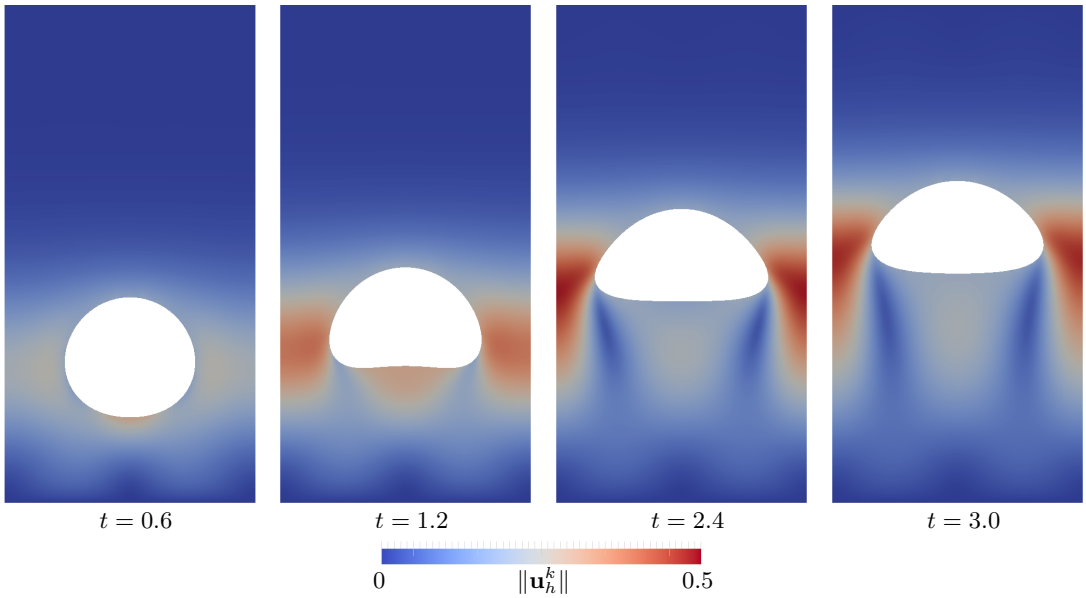


Figure 5.4: The graphs in this figure correspond to case 1 of the 2D benchmark [34]. It shows snapshots of the velocity field and the bubble shape at some selected times.
The shown solution was computed based on a fundamental mesh $\mathcal{T}_h$ with 80×160 mesh cells and the hybrid mesh discretization method based on the function spaces $(\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$. The relative refinement of the geometry mesh $\mathcal{T}_\varkappa$ was chosen as $\lambda = 3$.
It is noteworthy that this case is rather benign and even for the solution on the 20×40 fundamental mesh these pictures would appear almost identical. Furthermore, the aligned mesh approach and the hybrid mesh approach produce optically identical results, unlike in the low capillarity case presented in section 5.4.3.

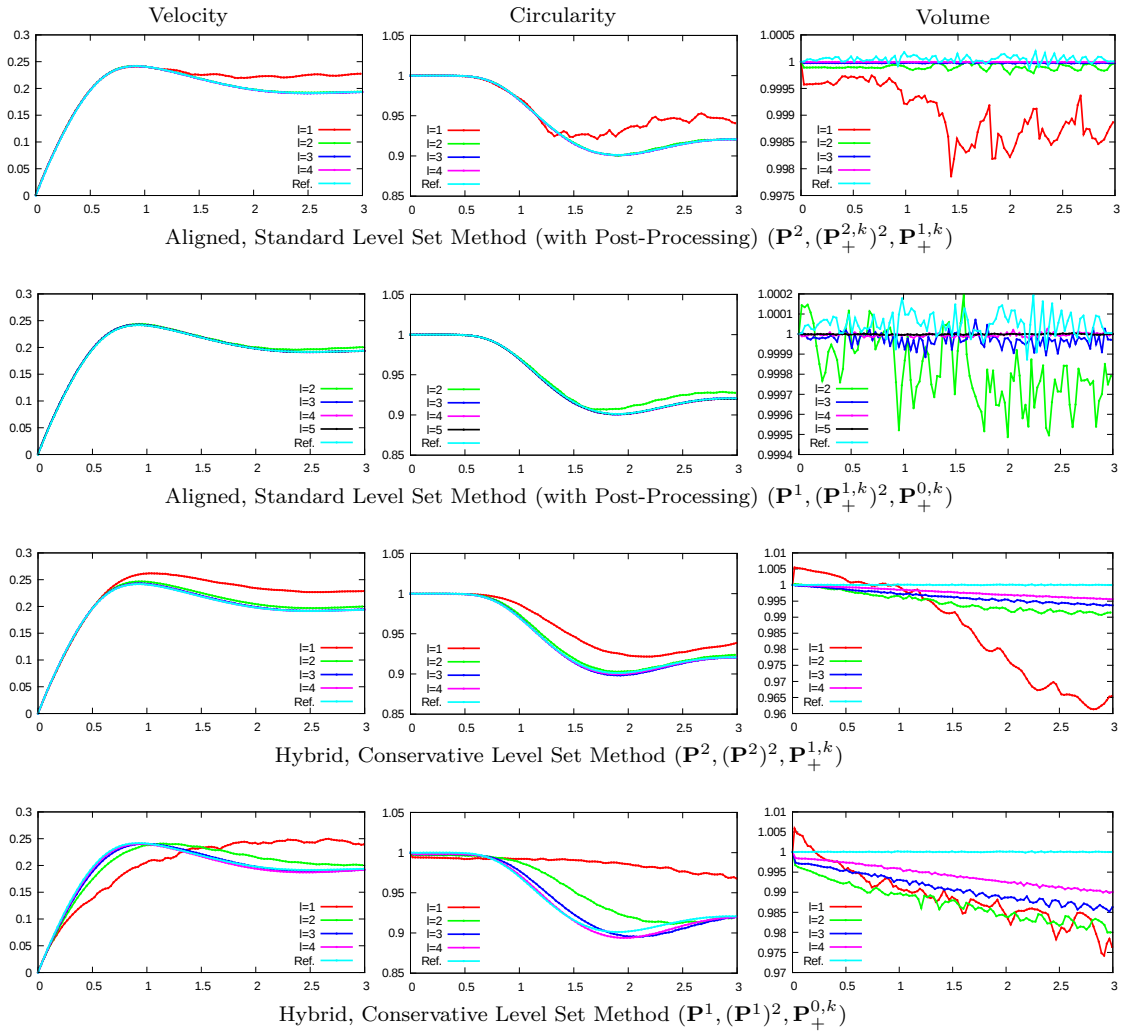


Figure 5.5: The graphs in this figure correspond to the data documented in table 5.7 on page 118 computed for the 2D benchmark [34]. It shows the average rising velocity of the bubble and its circularity as functions of the simulation time for both the aligned mesh discretization and the hybrid mesh discretization on mesh levels $l = \{1, \dots, 4\}$ - in each case for two sets of finite element spaces $(Z, \mathbf{X}, Y)$ (except for the aligned method of first order for which computations were performed on mesh levels $l = \{2, \dots, 5\}$). The graphs to the right show the development of the bubble's normalized volume (notice the varying scale of the y-axes).
The aligned method was used in combination with the standard level set method (including a post-processing for global volume conservation) while the hybrid method was combined with the conservative level set method.
The solutions were computed on a sequence of meshes based on a fundamental mesh $\mathcal{T}_h$ with $(5{\cdot}2^l){\times}(10{\times}2^l)$ mesh cells. In all cases, the relative refinement of the geometry mesh $\mathcal{T}_\varkappa$ was chosen as $\lambda = 3$ and the time-step size was chosen as $\Delta t = 2^{-l} \cdot 1.25 \cdot 10^{-2}$.

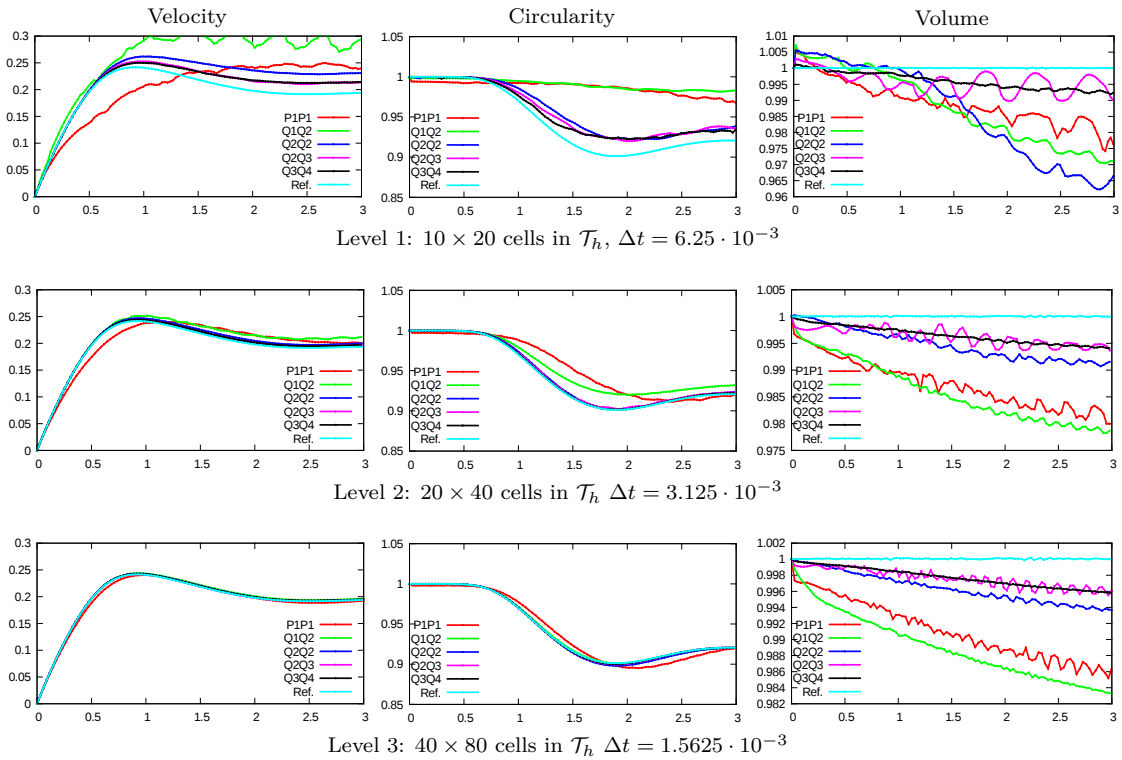


Figure 5.6: The graphs show the time development of the benchmark quantities, i.e. the bubble's rise velocity u_l^y and its circularity c_l together with the normalized bubble volume for different levels of p-refinement. This data corresponds to the numeric values in table 5.9 on page 120.
All values were computed with the hybrid mesh discretization combined with the conservative level set method on mesh levels $l = \{1, \dots, 3\}$ based on the fundamental meshes with $(2^l \cdot 5) \times (2^l \cdot 10)$ cells. The surface tension was discretized with the quasi-geometric method based on a geometry mesh with $(2^{l+\lambda} \cdot 5) \times (2^{l+\lambda} \cdot 10)$ and $\lambda = 3$.
Each graphs shows results for different finite element spaces. The following table shows the function spaces $(Z, \mathbf{X}, Y)$ corresponding to each label:

P1P1	$(\mathbf{P}^1, (\mathbf{P}^1)^2, \mathbf{P}_+^{0,k})$
Q1Q2	$(\mathbf{Q}^2, (\mathbf{Q}^1)^2, \mathbf{P}_+^{0,k})$
Q2Q2	$(\mathbf{Q}^2, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$
Q2Q3	$(\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$
Q3Q4	$(\mathbf{Q}^4, (\mathbf{Q}^3)^2, \mathbf{Q}_+^{2,k})$

Especially the results on mesh level 1 give the impression that the step from Q2Q3 to Q3Q4 does not provide any additional accuracy. This is consistent with the data in table 5.9.

	u^Y_{max}		$t^{\mathbf{v}}_{max}$		c_{min}		t^c_{min}	
	CLS	SLS	CLS	SLS	CLS	SLS	CLS	SLS
Level 1	0.2619	0.2411	1.0781	0.9406	0.9217	0.9211	2.141	1.709
Level 2	0.2467	0.2421	0.9422	0.9094	0.9029	0.9004	1.892	1.903
Level 3	0.2431	0.2415	0.9227	0.9219	0.8982	0.9009	1.924	1.900
Level 4	0.2421	0.2415	0.9168	0.9211	0.8997	0.9010	1.894	1.900
Others	0.2417-0.2421		0.9213-0.9313		0.9011-0.9013		1.875-1.904	

Table 5.6: This table documents the obtained results for some reference values as required by the 2D benchmark in [34]. These include the maximum rise velocity $\mathbf{v}_{max}$ of the bubble with corresponding time $t^{\mathbf{v}}_{max}$ of occurrence and the minimum circularity c_{min} with corresponding time t^c_{min}. The columns with title SLS contain values computed with the standard level set method and the aligned mesh discretization with finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{P}^2, (\mathbf{P}^{2,k}_+)^2, \mathbf{P}^{1,k}_+)$. The columns with title CLS contain values computed with the conservative level set method and the hybrid mesh discretization based on finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{P}^2, (\mathbf{P}^2)^2, \mathbf{P}^{1,k}_+)$.
For each of the reference values, the last rows give the range of values obtained by the participating groups in [34]. None of the values obtained on the highest level deviates from this range by more than 0.5%.

$(\mathbf{P}^M, (\mathbf{P}^M)^2, \mathbf{P}^{M-1,k}_+)$ for hybrid meshes with $M = 2$ and $M = 1$.

The SLS method is generally not volume conservative. As errors in the volume balance may quickly develop into a bottle-neck for the total accuracy of the computed benchmark quantities, especially for systems driven by buoyancy, some of the solutions presented in the original benchmark publication utilized a global post-processing which shifts in each time step the whole level set function by a constant factor in order to avoid global mass loss (e.g. [114]). To allow for a fair comparison, a similar procedure was implemented for the computations with SLS method presented in this thesis. Here, the shift was simply computed as the ratio of the current volume error to the current surface area of the bubble which is only an approximation of the correct factor unless the current bubble is a sphere (i.e. circle). As this process is not iterated, it does not provide exact volume conservation but works sufficiently well for the benchmark examples.

Here, it must be pointed out that this global post-processing may be quite problematic if the source of the volume error is strongly localized. The authors of [72] show its tendency to *rescale* the solution without improving it.

Notice that the error norms are generally computed with regard to the finest solution (i.e. the solution computed on the finest mesh level) that was computed with the same method (with the same finite element basis) as the discrete solution itself (except for table 5.9 analyzing p-refinement, for which all errors are computed relative to the method with the highest polynomial order on the finest mesh). This is necessary as the low order components of the error can not be identified reliably due to the lack of an analytical solution.

The operator splitting employed to decouple the solution of the Navier-Stokes Equations from the Level Set Equation limits the convergence of the time discretization to be of first order (the Strang Operator-Splitting employed for the aligned mesh discretization could improve this behavior but this has not been verified). However, the results presented in table 5.10 on page 121 show that the time step sizes were chosen sufficiently

small to disregard its influence on the accuracy of the computed benchmark quantities.

While this at least simplifies the interpretation of the results, the availability of four mesh levels allows the estimation of only two numbers for the asymptotic order of convergence in each of the benchmark quantities. Furthermore, as may be recognized by the time development of the benchmark quantities, the computations on the lowest level are hardly stable and their estimated convergence from level 1 to level 2 is thus rather dubious.

Keeping in mind the aforementioned restrictions, some interesting observations can be made with regard to the data in figure 5.5 on page 113 and table 5.7 on page 118:
First, for mesh levels $l > 1$ the solution graphs of the benchmark quantities are generally indistinguishable from the reference solution for all methods with polynomial order $M = 2$ (the reference solution corresponds to the finest solution computed with the $TP2D$ code in [34]). This is not true for the solutions computed with $M = 1$. In this case, the CLS method shows a significant deviation from the reference solution even on the finest mesh level while converging robustly in the same quantities.

The first order ($M = 1$) SLS simulation became unstable for mesh level 1 and thus no results have been plotted. However, in this case a simulation on the higher mesh level 5 was still feasible. This was not the case for the CLS method. The recompression of the phase-field required by the CLS approach takes approximately the same run-time as the solution of the Navier-Stokes Equations. In contrast, the run-time of the reinitialization of the distance functions via the Fast-Marching Method as required by the SLS approach is almost negligible compared to the total run time of typical simulations even when the projection and interpolation of the UDG basis to and from the geometry mesh are taken into account.

In [34], the authors tried to establish some reference benchmark values including the maximum rise velocity u^Y_{max} and the minimum circularity c_{min} as well as the time of their occurrence. Table 5.6 on the preceding page shows the corresponding values for both the SLS and the CLS method on all mesh levels and compares them with the values published in the original publication of the benchmark problem. On the finest mesh level the deviation of the computed benchmark values from the range of values obtained in [34] is at most 0.5%.

Of the two benchmark cases, this case is more strongly dominated by the surface tension relative to the driving force of buoyancy as is recognizable by the lower Eötvös number

$$E_o := \frac{\Delta\rho g L^2}{\sigma_s}. \tag{5.12}$$

Hence, the results of this case should be more sensitive to the choice of the surface tension discretization then those of case 2. To estimate the relative performance of the quasi-geometric method (QGM) (section 4.3 on page 92) and the narrow band projection method (NBP) ((4.11) on page 92), both methods were applied in combination with the CLS method and the hybrid mesh discretization based on the finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}^{1,k}_+)$. Furthermore, the impact of the uniform refinement λ of the geometry mesh $\mathcal{T}_\varkappa$ relative to the fundamental mesh $\mathcal{T}_h$ was evaluated. The results are presented in table 5.8 on page 119.

In spite of the low Eötvös number, the sensitivity of the benchmark values to the discretization method for the surface tension is rather low. On the lower levels it is actually negligible. While on the finest level that was employed for the comparison (level 3) the QGM method is generally superior, the difference between the benchmark values does not exceed a factor of 3 for any setup. Therefore, considering the generally high sensitivity of the surface tension discretizations, it is quite possible that a tuning of the parameters might bring a turn in favor of the projection method. When taking into account the results of the surface tension comparison in section 5.3, it is less surprising but still noteworthy that the sensitivity of the benchmark quantities on λ is rather low especially when considering the step from $\lambda = 3$ to $\lambda = 2$.

The author is not aware of any publication in which p-refinement was applied to and analyzed for incompressible two-phase flow with surface tension. There are good reasons to support a decision for low order finite elements for these problems as neither the velocity field nor the pressure field have high regularity at the interface and some schemes for the recompression / reinitialization might introduce low order errors into the solution of the Level Set Equation independent of the employed finite elements. Also, as soon as a phase-field is involved, the transition bandwidth Δ_φ in which the function fades between 0 and 1 introduces a natural bottle-neck for p-refinement beyond a certain order.

However, as long as the velocity field and the pressure are smooth enough within the two phases individually, a scheme like the aligned mesh discretization which allows to accurately represent the corresponding interface jumps may yet benefit from the usage of higher-order finite elements. The same is in principal true for the hybrid mesh approach although the kink in the velocity field will eventually become a limiting factor. For the benchmark case 1, these conditions are rather well met and an application of p-refinement is thus more promising than in case 2 (see figure 5.8) where the bubble is subjected to more severe deformations. Hence, for mesh levels $l \in \{1, 2, 3\}$, the simulation was performed for a number of different finite element spaces ranging up to fourth order approximations for the phase-field function. All simulations were based on the CLS method with the hybrid mesh discretization.

The corresponding results for the benchmark quantities are presented in figure 5.6 on page 114 and table 5.9 on page 120. With regard to the latter, it is noteworthy that the transition from the space triplet $(\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$ to $(\mathbf{Q}^4, (\mathbf{Q}^3)^2, \mathbf{Q}_+^{2,k})$ does not provide any additional accuracy with regard to the rise velocity values while the same is not true for the circularity. Furthermore, the graphs in figure 5.6 show that the same transition does not improve the volume error any further. As it involves the increase of the approximation order for all three field variables, it marks the apparent limit for p-refinement - at least for the given problem setup.

$\|u_l^y - u_{max}^y\|_\infty$	$(\mathbf{P}^2, (\mathbf{P}^2)^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}^1)^2, \mathbf{P}_+^{0,k})$		$(\mathbf{P}^2, (\mathbf{P}_+^{2,k})^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}_+^{1,k})^2, \mathbf{P}_+^{0,k})$	
Level 1	3.75e-02		6.40e-02		3.53e-02			
Level 2	6.66e-03	2.5	2.27e-02	1.5	1.26e-03	4.8	7.10e-03	
Level 3	1.30e-03	2.4	4.91e-03	2.2	4.18e-04	1.6	1.25e-03	2.5
Level 4							4.18e-04	1.6

$\|u_l^y - u_{max}^y\|_2$	$(\mathbf{P}^2, (\mathbf{P}^2)^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}^1)^2, \mathbf{P}_+^{0,k})$		$(\mathbf{P}^2, (\mathbf{P}_+^{2,k})^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}_+^{1,k})^2, \mathbf{P}_+^{0,k})$	
Level 1	1.61e-02		2.51e-02		1.95e-02			
Level 2	2.92e-03	2.5	8.82e-03	1.5	7.53e-04	4.7	3.10e-03	
Level 3	4.79e-04	2.6	1.76e-03	2.3	1.49e-04	2.3	6.20e-04	2.3
Level 4							2.17e-04	1.5

$\|c_l - c_{max}\|_\infty$	$(\mathbf{P}^2, (\mathbf{P}^2)^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}^1)^2, \mathbf{P}_+^{0,k})$		$(\mathbf{P}^2, (\mathbf{P}_+^{2,k})^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}_+^{1,k})^2, \mathbf{P}_+^{0,k})$	
Level 1	3.20e-02		9.34e-02		4.14e-02			
Level 2	4.52e-03	2.8	3.77e-02	1.3	1.92e-03	4.4	1.06e-02	
Level 3	1.53e-03	1.6	8.74e-03	2.1	3.02e-04	2.7	8.91e-04	3.6
Level 4							5.54e-04	0.7

$\|c_l - c_{max}\|_2$	$(\mathbf{P}^2, (\mathbf{P}^2)^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}^1)^2, \mathbf{P}_+^{0,k})$		$(\mathbf{P}^2, (\mathbf{P}_+^{2,k})^2, \mathbf{P}_+^{1,k})$		$(\mathbf{P}^1, (\mathbf{P}_+^{1,k})^2, \mathbf{P}_+^{0,k})$	
Level 1	1.07e-02		3.37e-02		2.07e-02			
Level 2	1.62e-03	2.7	1.14e-02	1.6	7.34e-04	4.8	5.71e-03	
Level 3	4.18e-04	2.0	2.45e-03	2.2	8.72e-05	3.1	3.61e-04	4.0
Level 4							2.45e-04	0.6

Table 5.7: This table documents the asymptotic convergence of the benchmark quantities u_l^y and c_l in [34] computed for mesh levels $l = \{1, \dots, 4\}$ for both the aligned discretization methods with finite element spaces $(\mathbf{P}^M, (\mathbf{P}_+^{M,k})^2, \mathbf{P}_+^{M-1,k})$ and the hybrid discretization method with finite element spaces $(\mathbf{P}^M, (\mathbf{P}^M)^M, \mathbf{P}_+^{M-1,k})$ (except for the aligned method of first order for which computations were performed on mesh levels $l = \{2, \dots, 5\}$). Notice that the aligned discretization of the Navier-Stokes equations was combined with the standard level set method including a post-processing which enforced (almost) perfect volume conservation while the hybrid mesh method was combined with the conservative level set method. The values in these tables corresponds to the graphs in figure 5.5.
For each combination of method and function space, all errors were computed with regard to the finest solution obtained by the same method and function space. The column titles indicate the choice for finite element spaces $(Z, \mathbf{X}, Y)$. The solutions were computed on a sequence of meshes based on a fundamental mesh $\mathcal{T}_h$ with $(5{\cdot}2^l){\times}(10{\times}2^l)$ mesh cells. In all cases, the relative refinement of the geometry mesh $\mathcal{T}_\varkappa$ was chosen as $\lambda = 3$ and the time-step size was chosen as $\Delta t = 2^{-l} \cdot 1.25 \cdot 10^{-2}$.

$\|u_l^y - u_4^y\|_\infty$	λ=3				λ=2				λ=1			
	QGM		NBP		QGM		NBP		QGM		NBP	
Level 1	2.09e-2		2.05e-2		2.16e-2		2.08e-2		2.47e-2		2.37e-2	
Level 2	3.46e-3	2.6	3.96e-3	2.4	3.68e-3	2.6	4.01e-3	2.4	4.28e-3	2.5	4.12e-3	2.5
Level 3	5.14e-4	2.8	9.99e-4	2.0	5.25e-4	2.8	9.86e-4	2.0	4.88e-4	3.1	1.09e-3	1.9

$\|c_l - c_4\|_\infty$	λ=3				λ=2				λ=1			
	QGM		NBP		QGM		NBP		QGM		NBP	
Level 1	2.39e-2		2.72e-2		2.45e-2		2.66e-2		3.04e-2		3.10e-2	
Level 2	3.90e-3	2.6	3.92e-3	2.8	3.79e-3	2.7	3.86e-3	2.8	4.83e-3	2.7	3.93e-3	3.0
Level 3	6.39e-4	2.6	1.51e-3	1.4	7.16e-4	2.4	1.54e-3	1.3	1.66e-3	1.5	2.29e-3	0.8

$\|u_l^y - u_4^y\|_2$	λ=3				λ=2				λ=1			
	QGM		NBP		QGM		NBP		QGM		NBP	
Level 1	8.66e-3		8.54e-3		8.92e-3		8.66e-3		1.01e-2		9.82e-3	
Level 2	1.37e-3	2.7	1.66e-3	2.4	1.45e-3	2.6	1.70e-3	2.3	1.55e-3	2.7	1.82e-3	2.4
Level 3	1.18e-4	3.5	2.83e-4	2.5	1.23e-4	3.6	2.81e-4	2.6	1.08e-4	3.8	3.12e-4	2.5

$\|c_l - c_4\|_2$	λ=3				λ=2				λ=1			
	QGM		NBP		QGM		NBP		QGM		NBP	
Level 1	8.84e-3		8.91e-3		8.85e-3		8.81e-3		9.71e-3		9.30e-3	
Level 2	1.18e-3	2.9	1.22e-3	2.9	1.14e-3	3.0	1.22e-3	2.9	1.38e-3	2.8	1.36e-3	2.8
Level 3	1.50e-4	3.0	2.81e-4	2.1	1.48e-4	2.9	2.79e-4	2.1	6.10e-4	1.2	6.82e-4	1.0

Table 5.8: The tables above document the sensitivity of the benchmark quantities in [34] (i.e. the rise velocity u_l^y of the bubble and its circularity c_l) on the relative geometry resolution λ for both the quasi-geometric method (QGM) and the narrow band projection method (NBP) for the discretization of surface tension. All values were computed with the hybrid mesh discretization combined with the conservative level set method on mesh levels $l = \{1, \dots, 4\}$ based on the fundamental meshes with $(2^l \cdot 5) \times (2^l \cdot 10)$ cells and finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$. The sub-triangulation which determines the accuracy of the discrete interface $\Gamma^{h,k}$ was constructed based on the geometry mesh with $(2^{l+\lambda} \cdot 5) \times (2^{l+\lambda} \cdot 10)$ cells.
All errors were computed with regard to the solution on Level 4 with highest geometry resolution $\lambda = 3$. It is noteworthy that the sensitivity of the circularity values is rather higher than for the velocities. Reducing the resolution from $\lambda = 3$ to $\lambda = 2$ is quite safe for both discretization methods.

$\|\mathbf{v}-\mathbf{v}_{ref}\|_\infty$	$(\mathbf{P}^1,(\mathbf{P}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^3,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^4,(\mathbf{Q}^3)^2)$	
Level 1	6.75e-02		1.11e-01		3.77e-02		2.00e-02		2.03e-02	
Level 2	2.28e-02	1.6	1.84e-02	2.6	6.40e-03	2.6	2.71e-03	2.9	2.86e-03	2.8

$\|c-c_{ref}\|_\infty$	$(\mathbf{P}^1,(\mathbf{P}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^3,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^4,(\mathbf{Q}^3)^2)$	
Level 1	8.74e-02		8.65e-02		2.84e-02		2.37e-02		2.41e-02	
Level 2	3.75e-02	1.2	2.20e-02	2.0	3.91e-03	2.9	3.67e-03	2.7	1.67e-03	3.9

$\|\mathbf{v}-\mathbf{v}_{ref}\|_2$	$(\mathbf{P}^1,(\mathbf{P}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^3,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^4,(\mathbf{Q}^3)^2)$	
Level 1	2.42e-02		4.21e-02	-35.3	1.62e-02		8.14e-03		8.16e-03	
Level 2	7.78e-03	1.6	6.98e-03	2.6	2.61e-03	2.6	8.60e-04	3.2	1.18e-03	2.8

$\|c-c_{ref}\|_2$	$(\mathbf{P}^1,(\mathbf{P}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^1)^2)$		$(\mathbf{Q}^2,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^3,(\mathbf{Q}^2)^2)$		$(\mathbf{Q}^4,(\mathbf{Q}^3)^2)$	
Level 1	3.20e-02		3.32e-02		9.84e-03		8.79e-03		7.89e-03	
Level 2	1.08e-02	1.6	7.63e-03	2.1	1.22e-03	3.0	1.12e-03	3.0	5.31e-04	3.9

Table 5.9: The tables above list the errors in the benchmark quantities, i.e. the bubble's rise velocity u_l^y and its circularity c_l together with the normalized bubble volume for different levels of p-refinement for the benchmark problem in [34]. This data corresponds to the graphs in figure 5.6.
The column titles indicate the employed finite element spaces $(Z, \mathbf{X})$ for the phase-field and the velocity field. The pressure space Y is always chosen identical to $\mathbf{X}$ but with its polynomial order reduced by one.
All values were computed with the hybrid mesh discretization combined with the conservative level set method on mesh levels $l = \{1, \dots, 3\}$ based on the fundamental meshes with $(2^l \cdot 5) \times (2^l \cdot 10)$ cells. The surface tension was discretized with the quasi-geometric method based on a geometry mesh with $(2^{l+\lambda} \cdot 5) \times (2^{l+\lambda} \cdot 10)$ and $\lambda = 3$.
All errors were computed with regard to the solution computed with the highest polynomial order, i.e. $(Z, \mathbf{X}, Y) = (\mathbf{Q}^4, (\mathbf{Q}^3)^2, \mathbf{Q}_+^{2,k})$ on mesh level $l = 3$.
Notice that the errors in the circularity improve monotonously with the polynomial order while this is not true for the velocities. In the last step of p-refinement, the velocity error actually increases.

$\|\mathbf{v} - \mathbf{v}_{ref}\|_\infty$	l=2	l=3
Δt	2.34e-02	4.11e-03
$\Delta t/2$	2.80e-02	4.52e-03
$\Delta t/4$	2.84e-02	4.25e-03
$\Delta t/8$	2.84e-02	4.70e-03

$\|c - c_{ref}\|_\infty$	l=2	l=3
Δt	2.98e-02	4.15e-03
$\Delta t/2$	3.73e-02	5.31e-03
$\Delta t/4$	3.76e-02	6.09e-03
$\Delta t/8$	3.66e-02	6.27e-03

Table 5.10: All the simulations of the benchmark problem [34] presented in this section have been simulated for time step sizes small enough to keep the numerical error introduced by the time discretization negligible. For the simulations based on the fundamental mesh of level l with $(2^l \cdot 5) \times (2^l \cdot 10)$ cells, the time step size was chosen as $\Delta t = 2^{-l} \cdot 1.25 \cdot 10^{-2}$.
The table above shows the errors in some benchmark quantities for constant mesh levels $l = 2$ and $l = 3$ and decreasing time step sizes.
All values were computed with the hybrid mesh discretization combined with the conservative level set method and finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$. The sub-triangulation which determines the accuracy of the discrete interface $\Gamma^{h,k}$ was constructed based on the geometry mesh with $(2^{l+\lambda} \cdot 5) \times (2^{l+\lambda} \cdot 10)$ cells and $\lambda = 2$.
Obviously, reducing the time step size does not improve the errors. The fluctuations are likely caused by the varying number of virtual time steps of the phase-fields recompression.

5.4.3 2D Benchmark - Low Surface Tension - Case 2

The benchmark test case 2 is remarkable for the fact that among the software frameworks which were employed for its solution in [34] no clear agreement about the final shape of the bubble could be established. Generally, in this case the bubble is subjected to stronger deformations than in test case 1 and the viscous drag at its sides eventually causes the development of bulges which - depending on the employed solver - either break off or remain connected and form skirt like extensions.

As may be seen in figure 5.8 on page 125, the application of the SLS and CLS method respectively does provide the two opposite extremes. While the computations with the SLS method result in a connected bubble with skirts the length of which exceeds the height of the bubble's main body, the CLS method does not develop similar structures. The pictures show results for two different sets of finite element spaces used in combination with the CLS method, i.e. $(\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$ and $(\mathbf{Q}^2, (\mathbf{Q}^1)^2, \mathbf{Q}_+^{0,k})$. While for the lower order approximation, two small satellites break off the bubble's main body and remain in its wake for the remaining time of the simulation, the higher-order approximation results in a permanently connected bubble - though without significant skirt extensions.

Although the large variety of solutions presented in [34] does not allow the determination of a *correct* solution, they do agree on the fact that the bubble's sides do either develop into skirts or at least keep a pointy profile at the attachment points of the skirt extensions to the main body. The latter option is also supported by fluid mechanical considerations, see [113]. The fact that the (potential) break-off points of the bubbles for the CLS method remain in a rather smoothly bulged shape does at least suggest that the diffusion introduced by the recompression of the phase-field severely reduces the effective spatial resolution.

The graphs in figure 5.7 on page 124 show the time development of the benchmark quantities for both methods on all mesh levels and additionally provide a comparison of

	$\mathbf{v}_{max}$		$t^{\mathbf{v}}_{max}$		c_{min}		t^{c}_{min}	
	CLS	SLS	CLS	SLS	CLS	SLS	CLS	SLS
Level 1	0.2622	0.3895	0.7563	0.2997	0.8119	0.4552	2.0188	2.9969
Level 2	0.2535	0.2503	0.7594	0.7281	0.7514	0.4401	1.8813	2.9906
Level 3	0.2520	0.2499	0.7391	0.7266	0.7234	0.5125	1.9031	2.7422
Level 4	0.2507	0.2500	0.7391	0.7281	0.7041	0.4903	2.0195	2.9992
Others	0.2502-0.2524		0.7281-0.7332		0.4647-0.5869		2.4004-3.0000	

Table 5.11: This table documents the obtained results for some reference values as required by the low capillarity 2D benchmark in [34]. These include the maximum rise velocity $\mathbf{v}_{max}$ of the bubble with corresponding time $t^{\mathbf{v}}_{max}$ and the minimum circularity c_{min} with corresponding time t^{c}_{min}. The columns with title SLS contain values computed with the standard level set method combined with the aligned mesh discretization for finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{P}^2, (\mathbf{P}_+^{2,k})^2, \mathbf{P}_+^{1,k})$. The columns with title CLS contain values computed with the conservative level set method combined with the hybrid mesh discretization for finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$. While the agreement for the rise velocity values is quite good for both methods, the CLS method does not reach similarly low circularity values as the other groups including the TP2D variant for which a satellite break-off occurs.

both methods on the highest mesh level with the results obtained by the different groups in the original benchmark publication [34]. The bubbles computed with the SLS method develop ever increasing values of their circularity on all mesh levels which corresponds to the development of skirt like extension. This is in stark contrast to the results for the CLS method which entails a clear minimum of circularity around $t = 2$.

The comparison with the results of the other groups reveals a surprisingly good agreement on the rise velocity which shows how much more sensitive the circularity reacts to changes of the bubble's shape. These visual conclusions are backed by the quantitative comparison of the benchmark reference values in table 5.11 on the preceding page.

Less striking but yet noteworthy is the fact that the SLS method develops instabilities on the lowest mesh level resulting in oscillations of the velocity field. The latter is not true for the CLS approach which actually provides a surprisingly good approximation of the bubble's final rise velocity.

For this benchmark case, the global volume post-processing scheme of the SLS method is not working well on the finest mesh level and results in an eventual volume error of 2.5%. This may possibly be the result of a rather unfortunate positioning of the bubble's thin skirt relative to the edges of the geometry mesh though this is only a guess and has not been analyzed systematically.

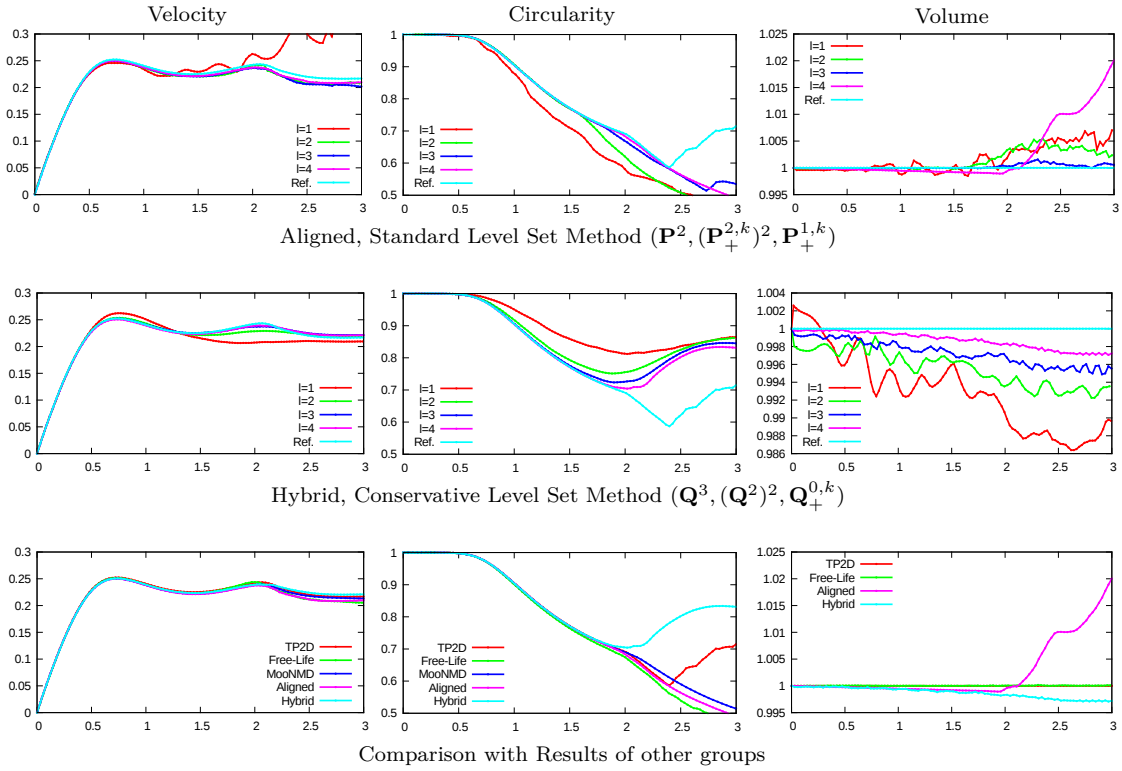
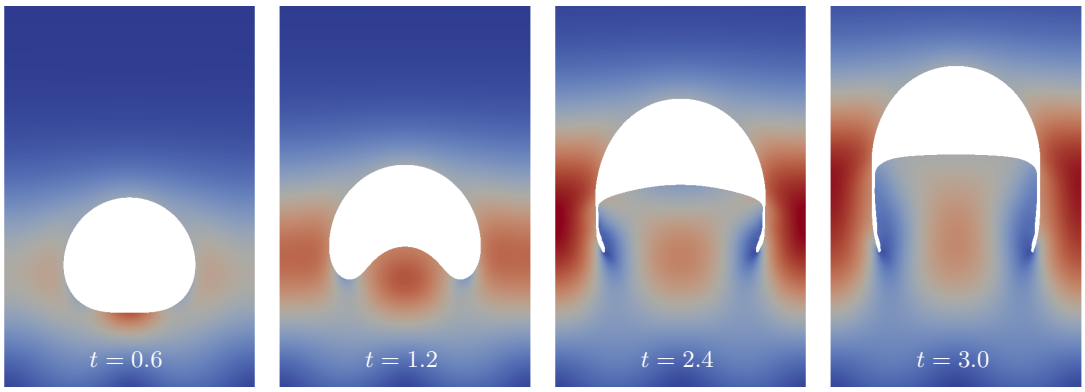
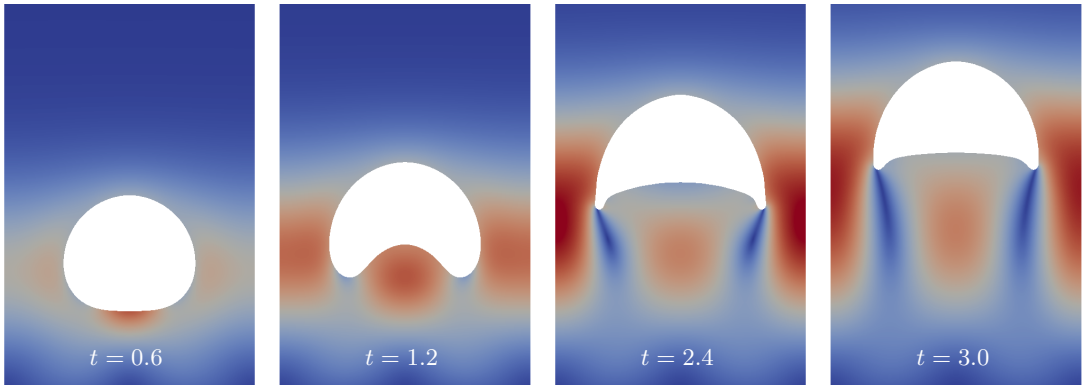


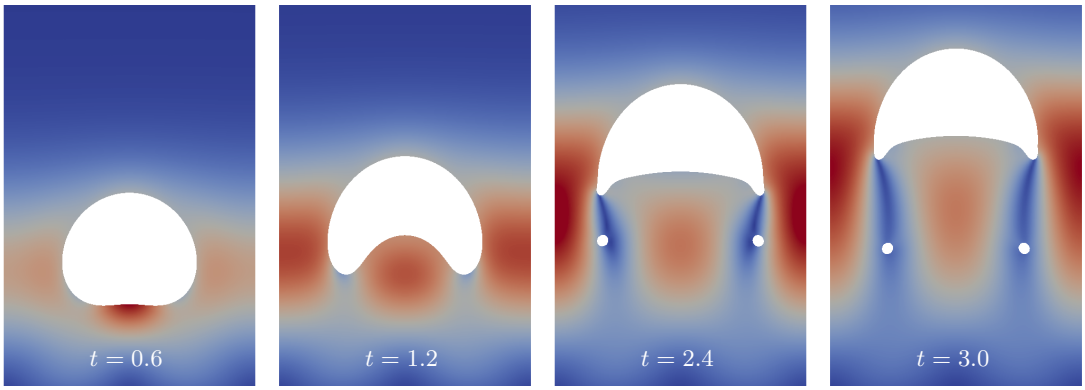
Figure 5.7: The graphs in this figure correspond to the test case 2 of the 2D benchmark [34]. They shows the average rising velocity of the bubble and its circularity as functions of the simulation time for both the SLS and the CLS method on mesh levels $l = \{1, \ldots, 4\}$. The employed finite element spaces $(Z, \mathbf{X}, Y)$ are given below the respective figures. The graphs to the right show the development of the bubble's normalized volume (notice the varying scale of the y-axes). The reference solution was chosen as the TP2D solution on the finest mesh level as given in [34]. In this solution, the bubble experiences a topological change and emits two small satellites. This behavior is recognizable in the kink in the development of the circularity value.
The graphs in the third row give a comparison of the results for both the SLS and CLS method with the results of different groups presented in the original benchmark publication, each on the finest available mesh level. From the circularity values it is quite obvious that all methods except the CLS and TP2D method result in bubble shapes with skirt like extensions.



Standard Level Set Method, $(\mathbf{P}^2, (\mathbf{P}_+^{2,k})^2, \mathbf{P}_+^{1,k})$



Conservative Level Set Method, $(\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$



Conservative Level Set Method, $(\mathbf{Q}^2, (\mathbf{Q}^1)^2, \mathbf{Q}_+^{0,k})$

Figure 5.8: The pictures in this figure correspond to the case 2 of the 2D benchmark [34]. They show snapshots of the velocity field and the bubble shape at some selected times for both the SLS and the CLS method. The employed function spaces $(Z, \mathbf{X}, Y)$ are given in the sub-title of each sequence. The color indicates the velocity magnitude within the outer fluid and its range goes from $|\mathbf{u}_h^k| = 0.0$ (blue) to $|\mathbf{u}_h^k| = 0.5$ (red). The shown solutions were computed based on a fundamental mesh $\mathcal{T}_h$ with 80×160 mesh cells. In all cases, the relative refinement of the geometry mesh $\mathcal{T}_\varkappa$ was chosen as $\lambda = 3$. Notice that these pictures were clipped and do not show the whole domain.

5.4.4 3D Rising Bubble

While there is no similarly comprehensive two-phase flow benchmark as [34] for the three-dimensional case, rising bubble simulations in three-dimensional domains were presented in [102, 112] for comparable fluid-dynamical regimes defined by the Eötvös number

$$E_o = \frac{\Delta\rho\, g\, 4\, r_B^2}{\sigma_s} \tag{5.13}$$

and the Morton number

$$M_o = \frac{g\mu_l^4 \Delta\rho}{\rho_l^2 \sigma_s^3}. \tag{5.14}$$

However, neither publication reveals the exact choice of all physical parameters required by the two-phase model such that any agreement is preconditioned on the validity of the fluid mechanical model in [113] describing a relationship between the Morton, the Eötvös and the Reynolds number

$$Re = \frac{\rho_l\, d_B u^\infty}{\mu_l}. \tag{5.15}$$

as obtained for the bubble's final rise velocity u^∞ and diameter d_B.

In both publications, the domain diameter was chosen as large as possible to avoid any influence of the domain boundaries on the simulation results. The results in [102] suggest that, for the dynamic regimes considered here, the distance between the bubble's center and the domain boundary should not be smaller than approximately five times the bubble radius in order to estimate the final rise velocity with an accuracy of a few percent relative to the velocity obtained for an unconstrained domain.

As the UDG method was originally developed for simulations in complex geometries, e.g. porous media, the current implementation is restricted to truly structured meshes

	D-Q2	D-Q3	E-Q2	E-Q3	S-Q2	S-Q3
Domain Ω^h	$0.02 \times 0.04 \times 0.02$					
Bubble Radius r_B	0.005					
Fund. Mesh $\mathcal{T}_h$	$16 \times 32 \times 16$					
Geom. Mesh $\mathcal{T}_\varkappa$	$2^{4+\lambda} \times 2^{5+\lambda} \times 2^{4+\lambda}$					
Geom. Mesh Res. λ	1	2	1	2	1	2
Phase Field Space Z	$\mathbf{Q}^2$	$\mathbf{Q}^3$	$\mathbf{Q}^2$	$\mathbf{Q}^3$	$\mathbf{Q}^2$	$\mathbf{Q}^3$
N.S.-Space $(\mathbf{X}, Y)$	$(\mathbf{Q}^1, \mathbf{Q}_+^{0,k})$	$(\mathbf{Q}^2, \mathbf{Q}_+^{1,k})$	$(\mathbf{Q}^1, \mathbf{Q}_+^{0,k})$	$(\mathbf{Q}^2, \mathbf{Q}_+^{1,k})$	$(\mathbf{Q}^1, \mathbf{Q}_+^{0,k})$	$(\mathbf{Q}^2, \mathbf{Q}_+^{1,k})$
Liquid Density ρ_l	1000					
Gas Densityρ_g	10					
Liquid Viscosity μ_l	0.566466		0.318547		0.1	
Gas Viscosity μ_g	0.001					
Surface Energy σ_s	0.01		0.1		0.01	
Morton Number Mo	10^3		0.1		0.97119	
Eötvös Number Eo	97.119		9.7119		97.119	
Dynamic Regime	Dimpled Bubble		Ellipsoid Bubble		Skirted Bubble	

Table 5.12: The table above describes the setups for the six 3D simulations presented in section 5.4.4. All are based on the same fundamental mesh although the relative refinements of the geometry mesh are adapted to the employed finite element spaces. The setups describe three different dynamic regimes according to [113], see figure 5.9.

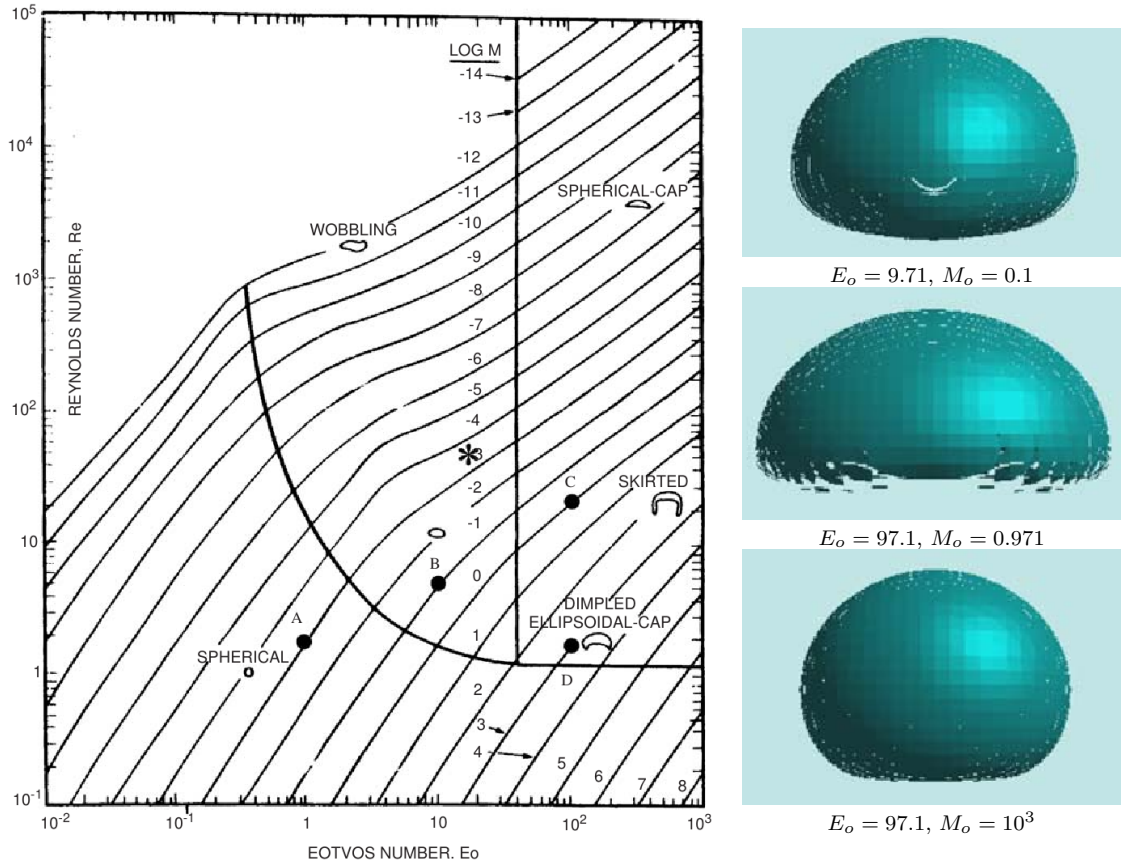


Figure 5.9: In [113], bubble shapes are associated to dynamic regimes given by the Reynolds number Re, the Morton number M_o and the Eötvös number E_o as indicated by the left diagram (copied from [115]). The pictures to the right show the results of simulations performed in [102] and correspond well to the results presented in figure 5.11.

without hanging nodes and does therefore not support any kind of h-refinement. Sadly, the run-times encountered for the structured meshes entailed a severe limit for the domain size relative to the bubble radius and for the simulations presented in this section the width of the rectangular domain could not be chosen to be more than twice the size of the initial bubble diameter.

In order to verify the applicability of the presented UDG scheme to three-dimensional problems, six simulations were performed, two for each of the three dynamical regimes in [112]. The latter are defined via the Morton, and Eötvös numbers according to [113], see figure 5.9. The three setups simulated and analyzed in [112] describe a sub-set of the simulations performed in [102] but at least allow a comparison of our results to two independently obtained reference values. For each dynamic regime two simulations were performed on the same fundamental mesh realizing one level of p-refinement. In all cases, the conservative level set method was used in combination with the hybrid mesh discretization and the surface tension was discretized with the narrow band projection

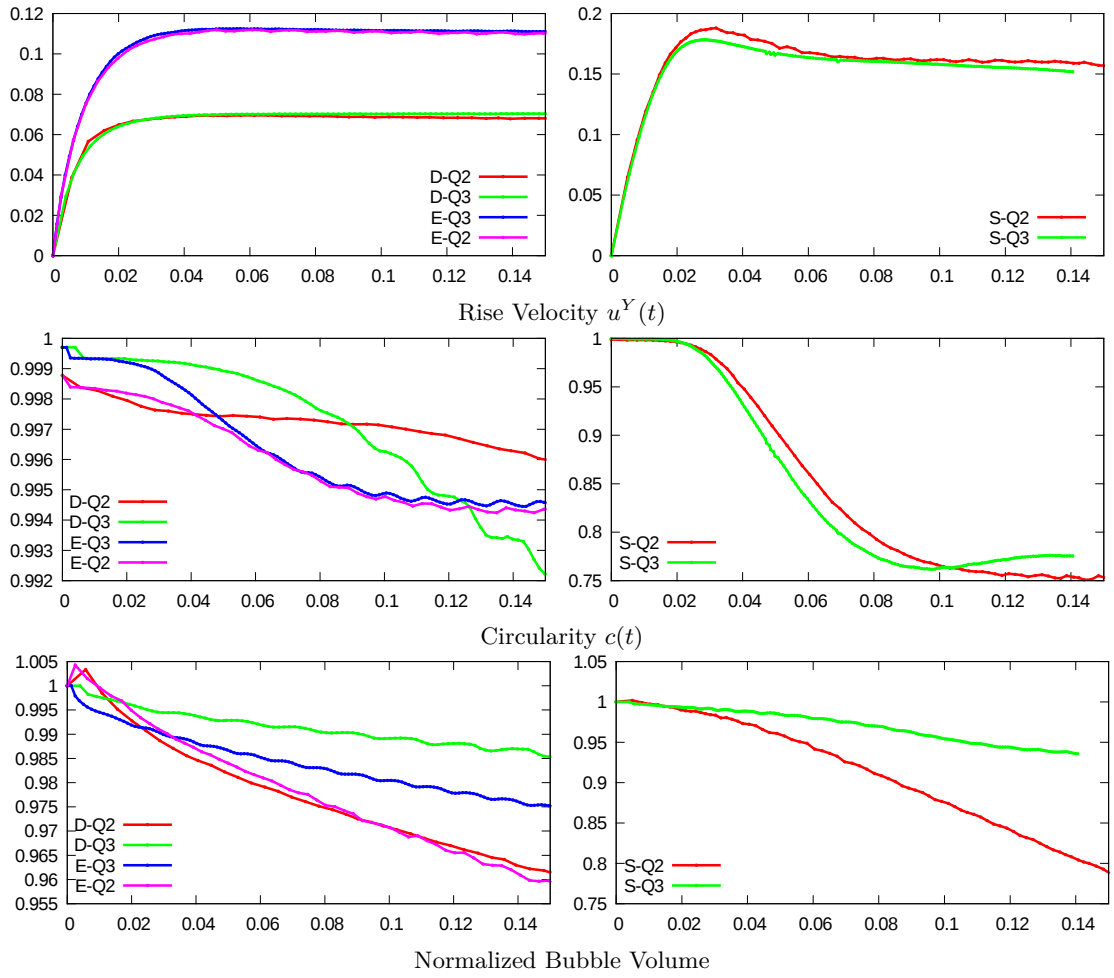


Figure 5.10: The graphs show the time development of the rise velocity of the bubble's center of mass as well as its circularity and its normalized volume for all simulation setups given in table 5.12 on page 126.

scheme. Comprehensive information about the domain, the finite element spaces and the physical parameters for all setups is given in table 5.12 on page 126.

Notice that the time step sizes for all simulations were chosen based on a heuristic criterion which adapts the time step size with regard to a *CFL-like* condition on the Navier-Stokes velocity field and the temporal resolution of capillary pressure waves according to

$$\Delta t \leq \sqrt{\frac{(\rho_0 + \rho_1)h^3}{4\pi\sigma_s}}. \tag{5.16}$$

as presented in [100]. The term *CFL-like* condition was chosen to indicate that the employed Navier-Stokes discretization does not require a CFL condition due to its implicit time stepping method. This condition is only required for the explicit solution of the level set / phase-field transport equation. However, the time step size of the latter was generally adapted whenever the time step size of the Navier-Stokes equation exceeded

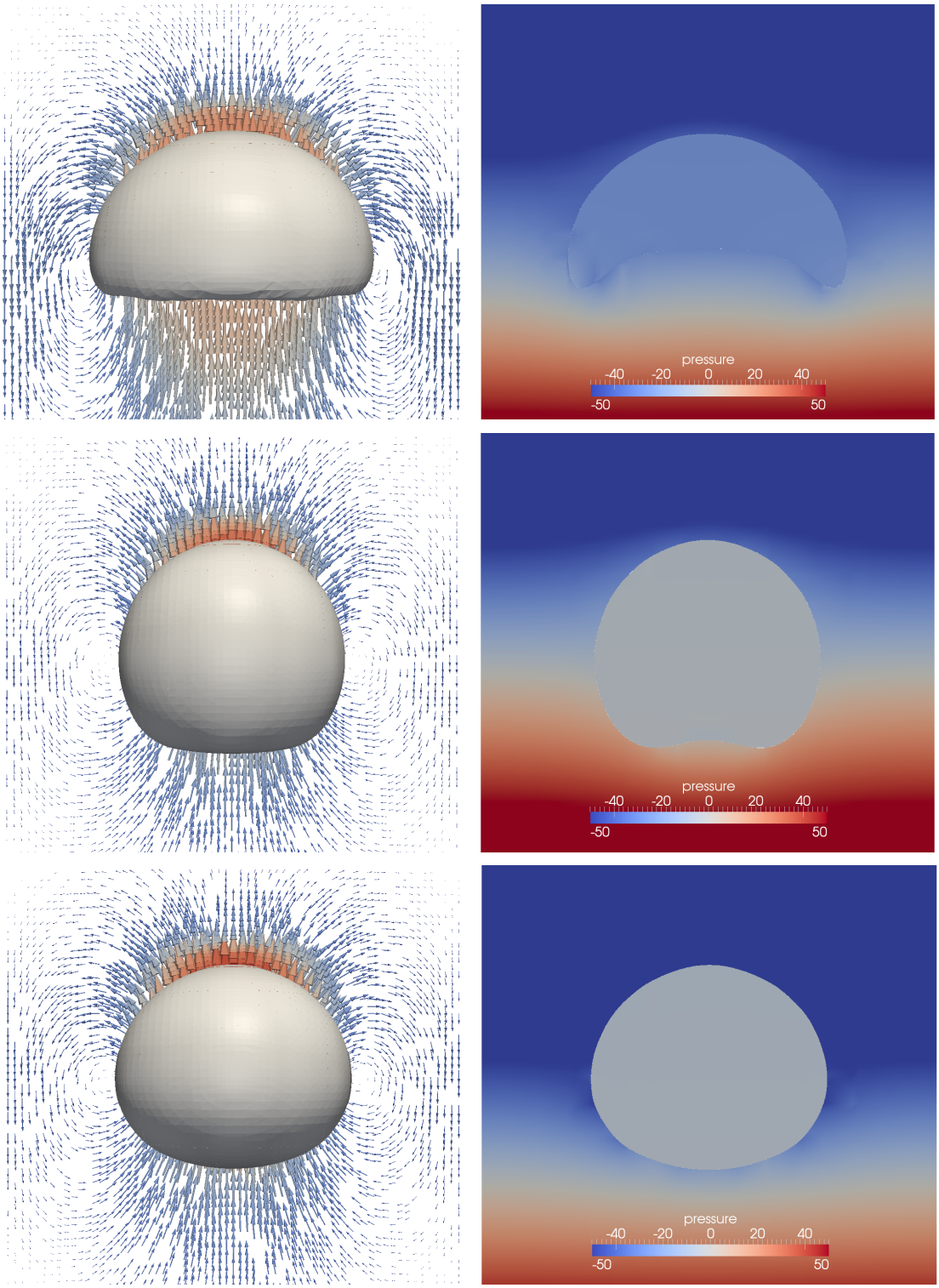


Figure 5.11: The pictures to the left show the final bubble shapes for the simulations S-Q3, D-Q3 and E-Q3 corresponding to the highest approximation order for each of the three dynamic regimes. The pictures to the right show the profile of the bubbles including the pressure field. They illustrate how accurately the UDG approach captures the sharp jump in the pressure field across the interface.

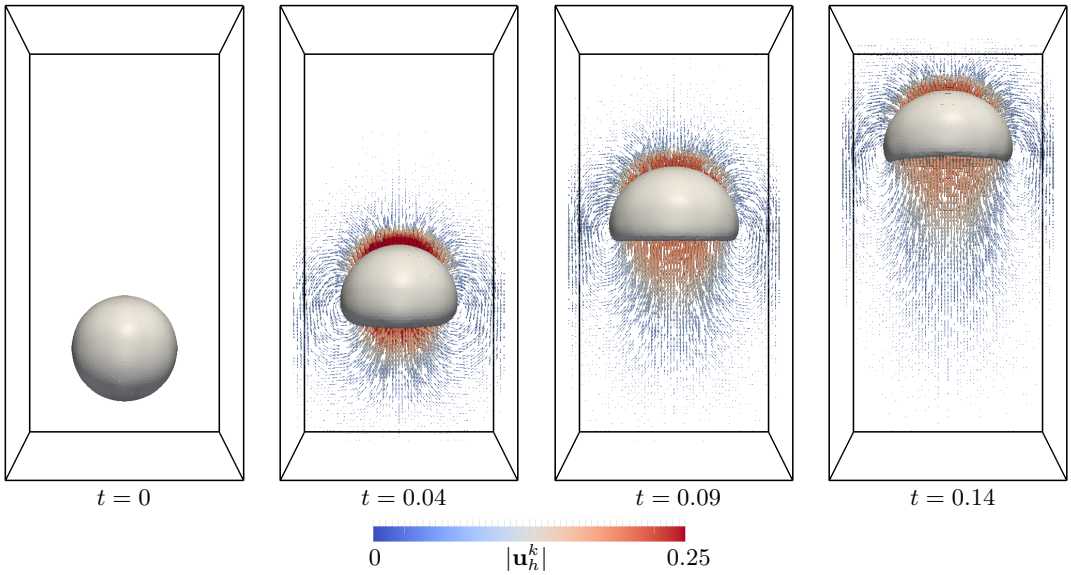


Figure 5.12: The pictures above show snapshots of the velocity field and the bubble surface for the simulation S-Q3. It is quite obvious that the interaction of the velocity field with the domain boundary is significant.

	D-Q2	D-Q3	E-Q2	E-Q3	S-Q2	S-Q3
$\langle u^Y \rangle \cdot 10^{-2}$	6.87 ± 0.05	7.02 ± 0.03	11.09 ± 0.08	11.16 ± 0.04	16.14 ± 0.18	15.79 ± 0.25
$\langle \Delta t \rangle$	1.4895e-03	1.7532e-03	9.7085e-04	1.1090e-03	7.4297e-04	8.8260e-04
$\max \Delta t$	2.8682e-03	3.2497e-03	1.2530e-03	1.2530e-03	2.4576e-03	2.7300e-03
N_k	89	70	125	109	175	147
N_r/N_k	3.21	4.46	3.32	5.22	3.77	4.89
T_R	8.7 days	17.1 days	12.6 days	31.8 days	18.4 days	49.6 days
d_B	0.0099	0.0100	0.0102	0.0104	0.0115	0.0123
$Re(\langle u^Y \rangle)$	1.20 ± 0.01	1.24 ± 0.01	3.55 ± 0.02	3.64 ± 0.01	18.56 ± 0.20	19.42 ± 0.30
Re_S [102]	1.7 ± 0.1		4.3 ± 0.1		18.0 ± 0.1	
Re_T [112]	2.03 ± 0.01		5.60 ± 0.01		18.0 ± 0.1	

Table 5.13: To allow a quantitative comparison the rise velocity u^Y of the bubble's center of mass was averaged over the interval $[0.8, 1.2]$ to compute $\langle u^Y \rangle$. Reynolds numbers $Re(\langle u^Y \rangle)$ were computed based on this averaged velocity. For comparison, the corresponding values presented in [102, 112] are given for each dynamic regime. However, notice that these numbers were obtained for different domains.
In addition to the Reynold numbers, the table above shows the average time step size $\langle \Delta t \rangle$, the maximum time step size $\max \Delta t$, the number of time steps N_k and the average number of recompression steps per time step N_r/N_k.
The given run-times T_R indicate the computation wall-time required for the respective simulation for the time interval $[0, 0.14]$ based on an *Intel(R) Xeon(R) CPU E7- 4870 @ 2.40GHz* processor. Notice that for the X-Q3 setups eight threads and for the X-Q2 setups four threads were employed for the assembling of the linear systems. Otherwise, the computations were performed sequentially. As the workload of the compute servers was not monitored and not all parts of the code were optimized, these numbers can only give a coarse idea of the principal performance.

the CFL limit of the transport. In such a case, multiple time steps were performed such that their sum is equal to the step size of the Navier-Stokes time step while individually conforming to the CFL condition (with some safety factor).

Although there is no data for comparison, the benchmark quantities which proved useful for the 2D setups in sections 5.4.2 and 5.4.3, i.e. the bubble's rise velocity u^Y, its circularity c and the volume error, were computed also for the three-dimensional simulations and are presented in figure 5.10 on page 128.

While the graphs hold no principal surprised, they reveal that the rise velocity is rather robust with regard to volume errors. This observation is most striking in the S-Q2 simulation in which the bubble, at the end of its rise, has endured a 20% mass loss while its rise velocity is still the same (even a bit higher) as in the S-Q3 simulation with a mass loss of approximately 5%. The relatively higher volume errors of the S-QX simulations are supposedly caused by the close proximity of the interface to the domain boundary. This configuration causes a distortion of the normal field whose diffusive projection is subject to boundary conditions which in this case were chosen such that the (high density) liquid phase was the wetting fluid. Notice that this *volume loss* suffered by the CLS method is substantially different from the volume loss typically encountered in level set methods. The phase-field itself being transported in a perfectly conservative way (up to floating point precision) implies that the lost phase volume may reappear as soon as the circumstances allow it, e.g. the bubble distances itself from the domain boundary again. Obviously, this principle advantage is not quite so helpful under the given circumstances.

The final bubble shapes for each dynamic regime are given in figure 5.11 on page 129 and correspond well to the principal forms predicted in [113]. For comparison, the shapes obtained by the simulations in [102] are given in figure 5.9 on page 127. While the visual agreement is not bad, the pictures do not allow and quantitative statement.

In order to obtain some quantitative measure of agreement, the bubble's rise velocity u^Y was averaged over the interval $[0.08, 0.12]$ to obtain $\left\langle u^Y \right\rangle$. The corresponding Reynolds numbers

$$Re = \frac{\rho_l \, d_B \left\langle u^Y \right\rangle}{\mu_l} \tag{5.17}$$

are given in table 5.13 on the facing page together with some additional statistics and compared to the results obtained in [102, 112]. Generally, these numbers do not agree too well. Especially in the *dimpled* regime, the Reynold numbers obtained by the D-QX simulations deviate about 50% from the corresponding numbers of the other groups, though this observation is somewhat alleviated by the fact that those values deviate approximately 20% from each other. On the other hand, for the *ellipsoid* bubbles, the values computed in the E-QX simulations and in [102] are rather similar compared to the one obtained in [112]. For the *skirted* bubble, the results of the other groups agree within their accuracy of 0.5% while values corresponding to the simulations S-QX deviate up to 7%. Notice that in this last case the bubble diameters d_B reach more than half the column width of the domain Ω used in S-QX. In these circumstance, on may pose the question of how meaningful the definition (5.17) can still be in this situation.

When considering the pictures in figures 5.11 on page 129 and 5.12 on the facing page

it becomes obvious that the velocity field in all cases is still significantly influenced by the domain boundary for which full slip boundary conditions were applied. These boundary interactions may very well account for a big part of the observed deviations in the Reynolds numbers. However, considering that the simulations in [102, 112] were performed on different domains with (possibly) different physical parameters, the obtained agreement is rather satisfying.

5.5 Capillary Tube

The examples given in the following two sections are intended to illustrate the ability of the UDG approach to handle a moving contact line and enforce a microscopic contact angle. As discussed in section 3.3, the simulation of multi-phase contact lines is yet impeded by multiple modeling issues. Numerical simulations appear to be the key technology for the development of new modeling approaches and the calibration of any entailed parameter values. However, due to the generally high numerical effort involved in any two-phase flow problem, these simulations must be performed by robust and optimized implementations.

The following examples are intended to give a proof of concept which show the principal applicability of the UDG method for the simulation of contact line dynamics. Therefore, each of the modeling issues described in section 3.3 is handled in the simplest way possible:

The modeling of dynamic contact angles, as realized in e.g. [31, 71, 72], is avoided and instead a constant microscopic contact angle of $\alpha_c = 0$ is enforced. This choice corresponds to the case of an initially wet domain boundary that is covered by a microscopic layer of the wetting fluid. This case is not only of tremendous practical relevance but generally poses a challenge for any particle based discretization method.

The kinematic paradox (the local suspension of the no-slip Navier-Stokes boundary condition) is here resolved via global Navier-Slip conditions, i.e. a tangential drag proportional to the tangential velocity. The corresponding Navier-Drag coefficient is chosen as $\beta_N = 1000$ which exceeds the viscosity values by at least two orders of magnitude. The physical parameters and the regularization parameters employed for the following examples are set equal to those in case 1 of the 2D benchmark, i.e. $\rho_1 = 1000$, $\rho_2 = 100$, $\mu_1 = 10$, $\mu_2 = 1$, and $\sigma_s = 24.5$ (see 5.9 for regularization parameters). Here, the denser fluid is chosen as the wetting fluid and no gravity force is applied in any of the following tube examples.

The *tube* is supposed to be closed and impenetrable at all boundaries which is realized via global Navier-Slip conditions. As no gravity force is applied, the surface tension forces are the only driving forces and the interface is expected to be rapidly deformed into a semispherical (semicircular) shape which touches the domain boundary tangentially and thus realizes the microscopic contact angle on the macro-scale. In this final stationary configuration (after some relaxation time), all remaining velocities must be considered as spurious velocities resulting from errors in the discretization of the surface tension or the force caused by the microscopic contact angle.

In both the two-dimensional and the three-dimensional example, the interface will be initially plane and enclose an angle of 90° with the domain boundary on the macro-scale.

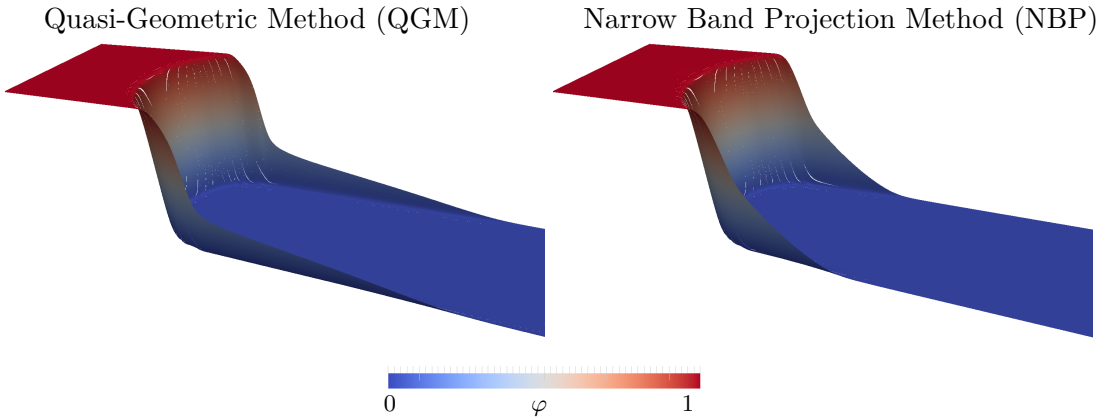


Figure 5.13: The pictures above show the final phase-field for the two-dimensional capillary tube example as obtained by both methods at time $t = 10$ on the highest mesh level 4. They illustrate how the phase-field *reaches* into the non-wetting phase in order to realize a macroscopic contact angle close to the microscopic contact angle $\alpha_c = 0$. This is a possible source of errors in the volume balance especially if amplified by spurious velocities along the contact line as it happened for the QGM method.

		QGM			NBP		
	$\mathcal{T}_h$	N_k	$\langle \Delta t \rangle$	$\max \Delta t$	N_k	$\langle \Delta t \rangle$	$\max \Delta t$
Level 1	4×16	1400	7.143e-3	7.472e-3	1400	7.143e-3	7.472e-3
Level 2	8×32	3800	2.632e-3	2.642e-3	3800	2.632e-3	2.642e-3
Level 3	16×64	10800	9.259e-4	9.340e-4	10800	9.359e-4	9.340e-4
Level 4	32×128	30333	3.298e-4	3.302e-4	30318	3.229e-4	3.302e-4

Table 5.14: The table above provides definitions of all four mesh levels employed for the simulation of the two-dimensional capillary tube example. For each simulation performed with the quasi-geometric method (QGM) and the narrow band projection method (NBP) the average time step size and the maximum time step size are given respectively. In all cases the geometry mesh $\mathcal{T}_\varkappa$ was chosen according to $2^\lambda \varkappa = h$ with $\lambda = 3$.

While a phase-field may easily represent this state, the final configuration in which the interface intersects (almost touches) the domain boundary at a very small angle (which due to the piecewise-linear nature of the edges in the sub-triangulation is always bigger than zero) requires the phase-field to extent a bit into the non-wetting phase in order to realize a possibly small contact angle. This effect may result in volume errors, see figure 5.13.

These decisions result in a physical model which will be employed to compare different methods for the discretization of surface tension, namely, the quasi-geometric method (QGM) from section 4.3 and the narrow band projection method (NBP) from section 4.2.2. All simulations presented in this section are based on the conservative level set method used in combination with the Hybrid Mesh approach.

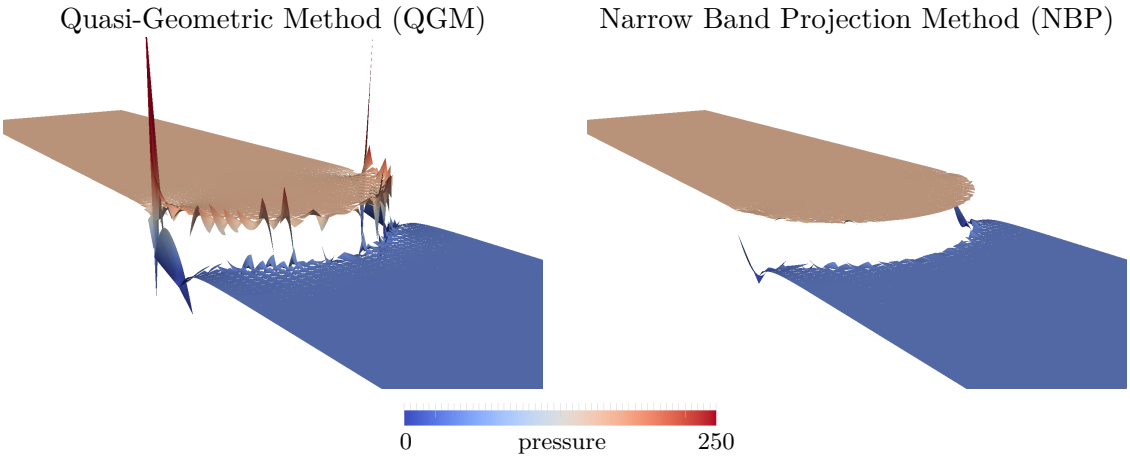


Figure 5.14: The pictures above show the warped pressure field obtained for the two-dimensional capillary tube example for both methods at $t = 0.5$. The oscillations are more pronounced for the QGM approach and largest at the contact points of all three phases (notice that the warp exceeds the picture height).

	$Err(\Delta \langle P \rangle)$		$Err(A)$		$\|\mathbf{u}_h\|_\infty$	
	QGM	NBP	QGM	NBP	QGM	NBP
Level 1	29.86 (24.38%)	1.63 (1.33%)	0.140 (22.26%)	0.109 (17.37%)	0.603	0.058
Level 2	19.08 (15.57%)	2.24 (1.83%)	0.109 (17.35%)	0.123 (19.49%)	1.004	0.034
Level 3	5.65 (4.62%)	1.61 (1.32%)	0.046 (7.29%)	0.033 (5.29%)	0.826	0.021
Level 4	0.48 (0.39%)	5.00 (4.08%)	0.017 (2.70%)	0.007 (1.15%)	0.714	0.012

Table 5.15: To allow a quantitative assessment of the two-dimensional capillary tube example, the values for the jump of the averaged pressures in each phase, the interface area (length), and the infinity norms of the velocity field as shown in figure 5.16 were averaged over the time interval $t \in [4, 10]$ and their deviation from the reference values is listed in the table above for all mesh levels (the relative deviation is always appended in parenthesis). The simulations were performed for both the quasi-geometric method (QGM) and the narrow band projection method (NBP) for the surface tension discretization. Reference values were computed according to the expected final state of the interface given by a semicircular shape with a pressure jump according to the Young-Laplace law. For the given setup, this corresponds to an interface area (length) of $A = 0.2\,\pi \approx 0.628$ and a pressure jump of $\Delta P = \sigma_s / 0.2 = 122.5$.

5.5.1 Capillary Tube 2D

The simulations presented in this section are based on a two-dimensional computational domain 0.4×1.6. Initially at time $t = 0$, the interface is horizontal and positioned such that it intersects the domain boundaries at $(0, 0.4)$ and $(0.4, 0.4)$.

In order to observe a convergence behavior, the simulations were performed for a sequence of fundamental meshes corresponding to mesh levels defined in table 5.14 on the previous page which also gives a listing of the employed time step sizes. The latter were chosen according to the heuristic described in section 5.4.4. For this two-dimensional case, the finite element spaces $(Z, \mathbf{X}, Y) := (\mathbf{Q}^3, (\mathbf{Q}^2)^2, \mathbf{Q}_+^{1,k})$ were used exclusively in all

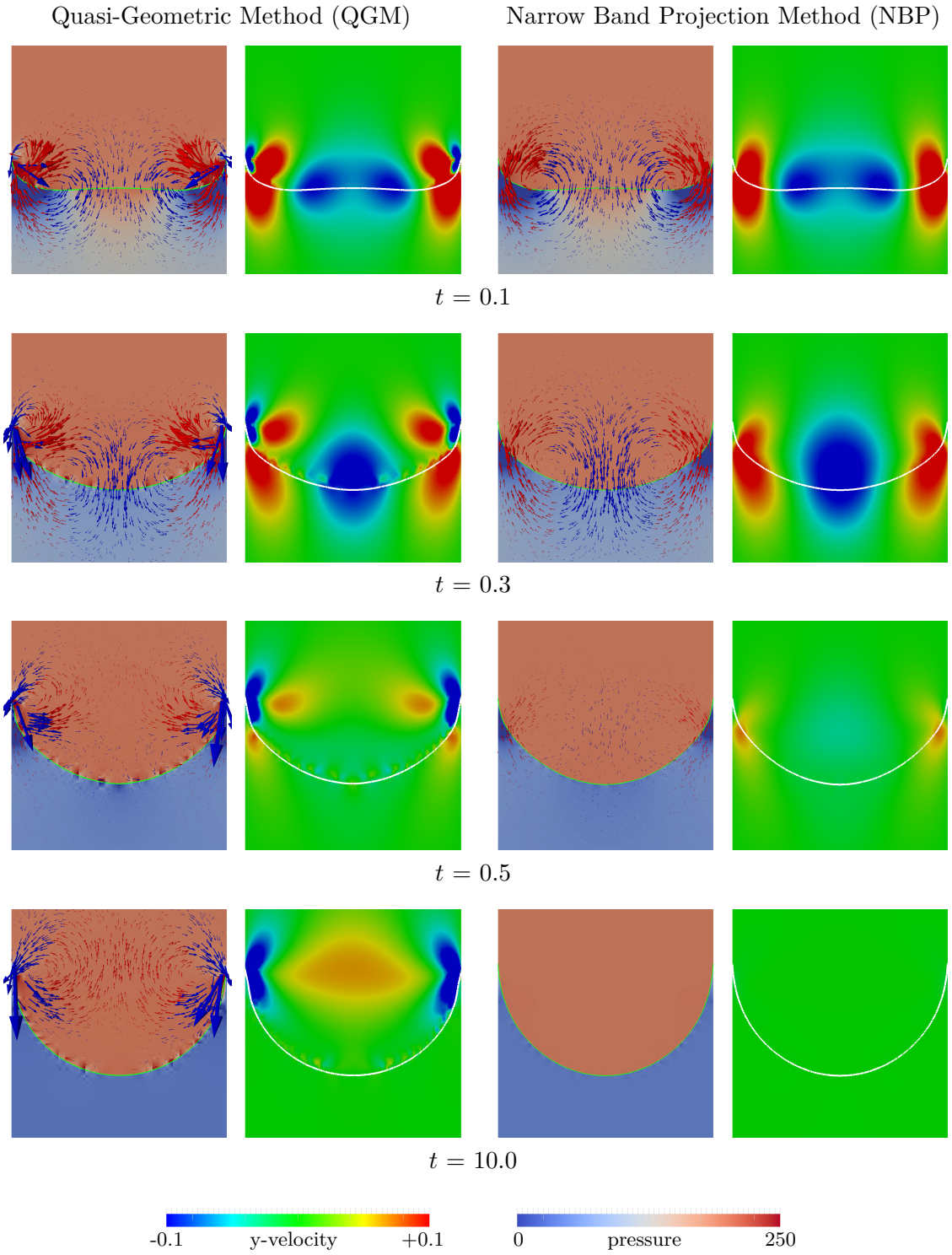


Figure 5.15: The pictures above show snapshots of the pressure and the y-velocity field for the two-dimensional capillary tube example. This corresponds to the simulation performed on mesh level 4 according to table 5.14. The velocity arrows were scaled with a factor of 0.2. The pictures are clipped from above and below and show only a part of the total computational domain. Notice that both the pressure and the y-velocity field can exceed the color ranges. See figure 5.16 and table 5.15 for a more quantitative assessment.

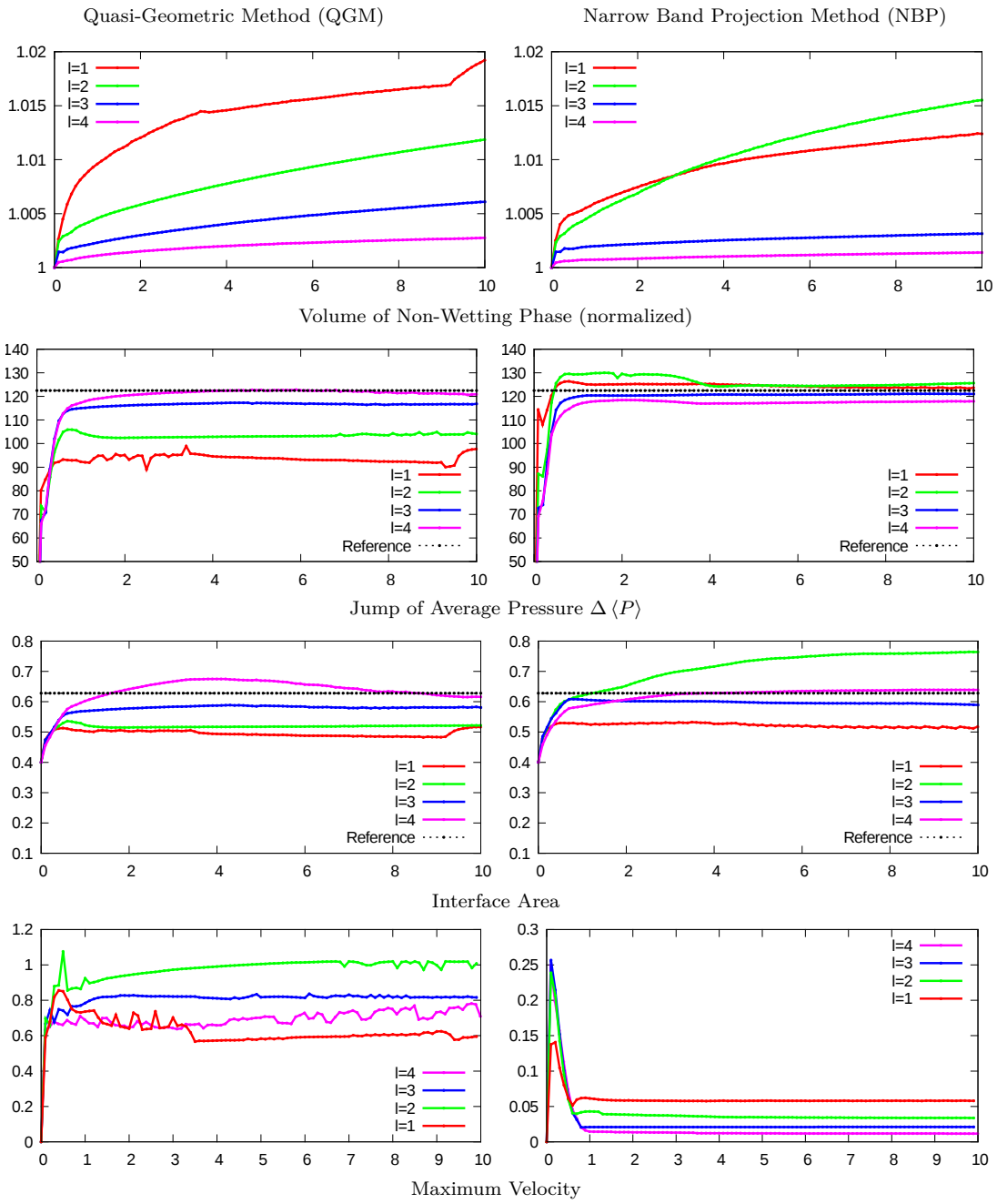


Figure 5.16: The graphs show the time development of some reference quantities including the normalized volume of the non-wetting phase Ω_2^h, the interface area (length) of the discrete interface $\Gamma^{h,k}$, the jump of the average pressure in each phase and the infinity norm of the velocity field. The *reference* value is assumed to be given by the expected final state of the interface, i.e. a semicircular interface with a pressure jump as determined by the Young-Laplace Law. Compare also with the results in table 5.15.

simulations.

The low positioning of the interface relative to the domain height serves to not artificially reduce the possible impact of the aforementioned source of volume error at the contact line, see figure 5.13.

The principal time evolution of the fluid phases may be observed in figure 5.15 which shows snapshots of the interface position including the corresponding velocity and pressure fields. Most prominent are the significantly higher spurious velocities observed for the quasi-geometric surface tension discretization which are by an order of magnitude higher than for the simulations utilizing the narrow band projection method. While the tests in section 5.3 revealed that the quasi-geometric discretization approach does in general produce more spurious velocities, one should point out that the piecewise-linear discrete interface $\Gamma^{h,k}$ can in principle not realize the microscopic contact angle $\alpha_c = 0$ on the macro-scale. This deficit could be more problematic for the quasi-geometric approach than for the projection method. However, this can not be said with certainty without considering simulations with finite contact angles α_c although the localization of the velocities and the oscillations in the pressure field as given in figure 5.14 are quite suggestive.

The graphs in figure 5.16 on the facing page show the time development of the volume (of the non-wetting phase), the jump of the average pressures, the surface area (length) of the discrete interface and the infinity norm of the velocity field for all four mesh levels.

They reveal that, in spite of the visual impression given by figure 5.13, the error in the volume balance is just about a factor of two and its time development does not indicate any dramatic changes.

While the jumps of the average pressures shows a rather robust convergence behavior for the QGM method, this is not the case for the NBP method, though it provides better approximations on the lower mesh levels (notice that this effect may yet be the result of poorly tuned regularization parameters).

These visual impressions are supported by the data in table 5.15 on page 134 which lists the errors in the pressure jump and the interface area relative to the analytically expected values (Young-Laplace law) for all four mesh levels.

		QGM			NBP		
	$\mathcal{T}_h$	N_k	$\overline{\Delta t}$	$\max \Delta t$	N_k	$\overline{\Delta t}$	$\max \Delta t$
Level 1	$8 \times 4 \times 4$	81	1.235e-2	1.494e-2	81	1.235e-2	1.494e-2
Level 2	$16 \times 8 \times 8$	208	4.808e-3	5.284e-3	200	5.000e-3	5.284e-3
Level 3	$32 \times 16 \times 16$	518	1.695e-3	1.868e-3	540	1.852e-3	1.868e-3

Table 5.16: The table above provides definitions for all three mesh levels employed for the simulation of the three-dimensional capillary tube example. For each simulation performed with the quasi-geometric method (QGM) and the narrow band projection method (NBP) for the discretization of surface tension, the average time step size and the maximum time step size are given respectively. In all cases the geometry mesh $\mathcal{T}_\varkappa$ was chosen according to $2\,\varkappa = h$.

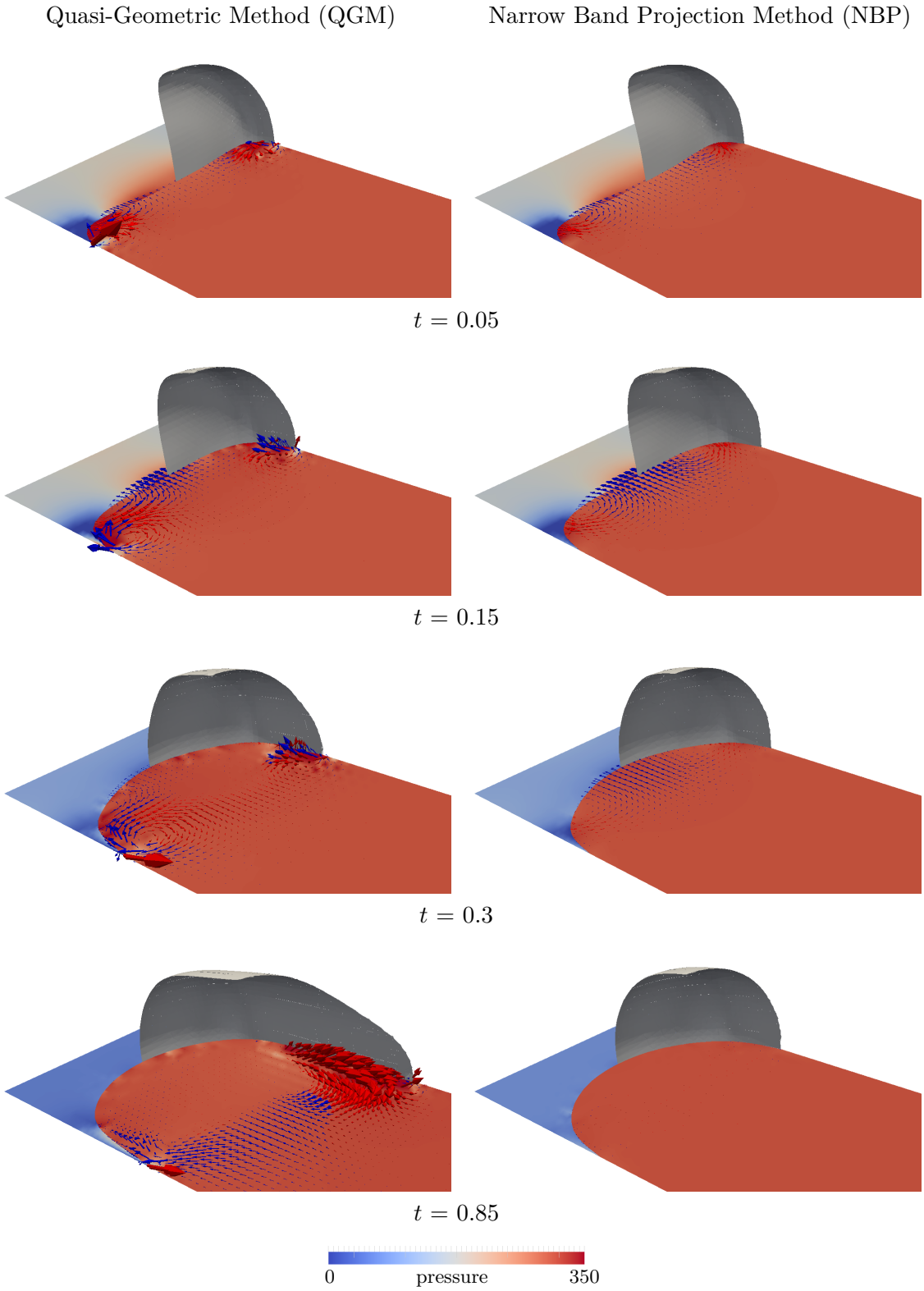


Figure 5.17: The snapshots above show the pressure field on a cross-section through the cylindrical tube together with the discrete interface $\Gamma^{h,k}$ (gray) and the arrows for the velocity field. The latter are scaled by a factor of 0.1 and colored red if the z-component is positive and blue otherwise.

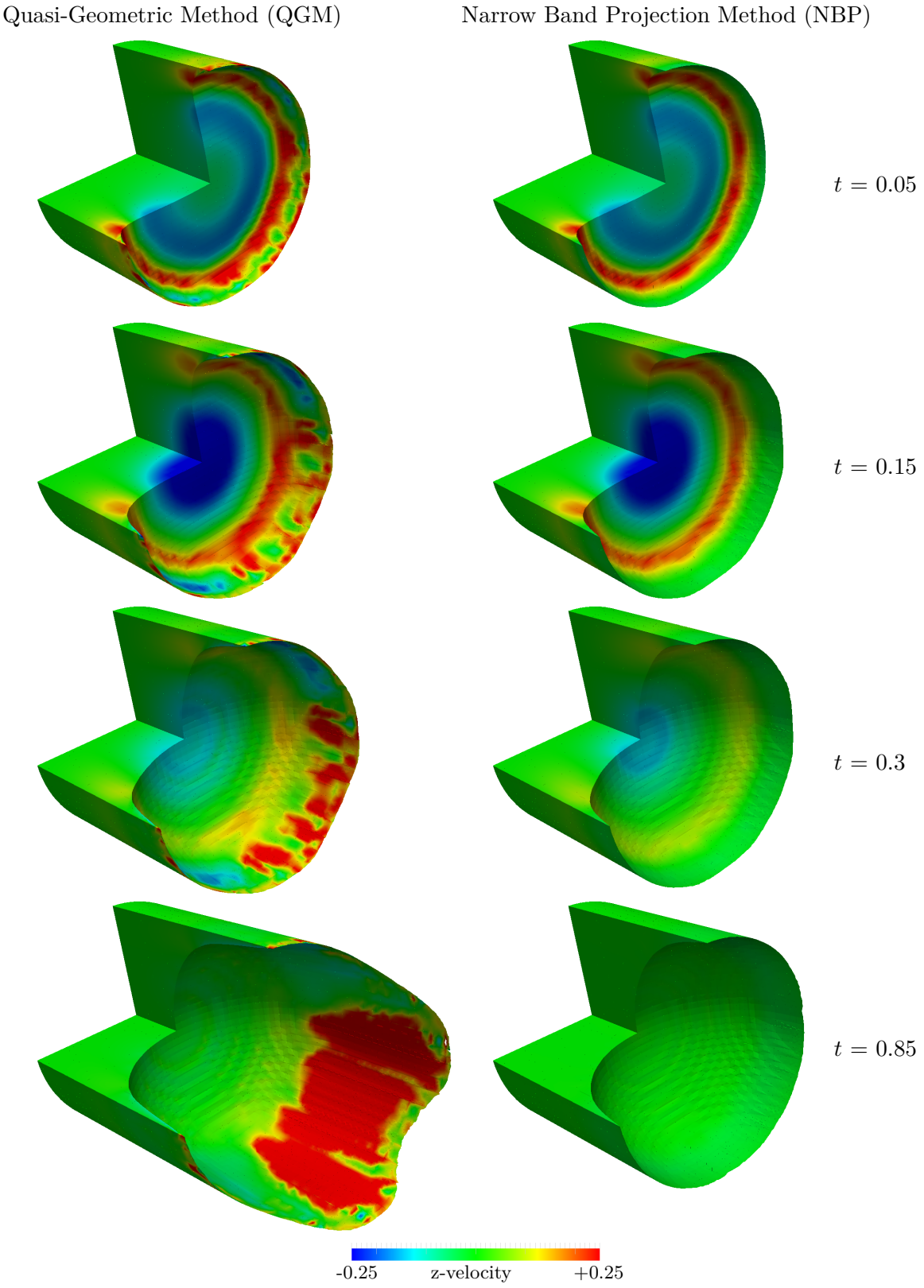


Figure 5.18: The pictures above show three of the four cylinder quadrants filled by the wetting fluid. The colors correspond to the z-velocity component (notice that the actual values may exceed the color range). Notice that, although for the QGM method the wetting fluid reaches further into the domain of the non-wetting fluid, this extension is actually a very thin film hardly contributing to the volume balance.

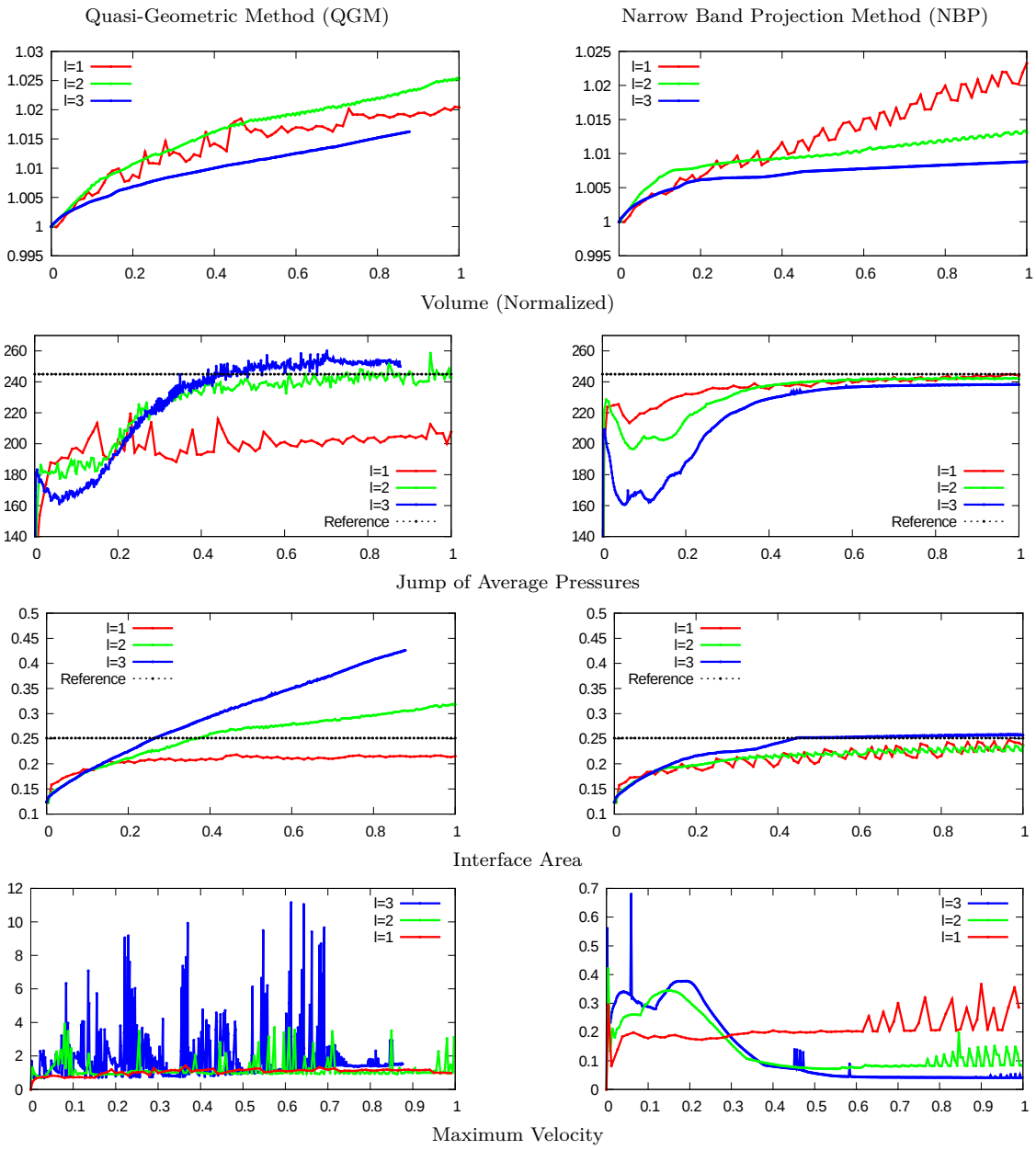


Figure 5.19: The graphs show the time development of some reference quantities including the volume of the non-wetting phase Ω_2^h, the interface area (length) of the discrete interface $\Gamma^{h,k}$, the jump of the average pressure in each phase and the infinity norm of the velocity field. The *reference* value is assumed to be given by the analytical values as would result from the expected final state of the interface, i.e. a semicircular interface with a pressure jump given by the Young-Laplace law. Compare also with the results in table 5.17.

5.5.2 Capillary Tube 3D

The two-dimensional capillary tube example in the previous section was applied on a *fitted* rectangular domain, i.e. $\Omega = \Omega^h$, spanned by the fundamental mesh. However, for the three-dimensional case, a cylindrical tube Ω of radius 0.2 was cut out of a rectangular domain with extent $\Omega_r = (0, 0.4) \times (0, 0.4) \times (0, 0.8)$. This ensures that the sub-triangulation at the contact line is actually constructed with the recursive sub-triangulation scheme described in chapter 2. Unlike the two-dimensional case, for which this construction holds little challenges, the three-dimensional reconstruction may even entail small discontinuities in the interface as described in section 3.7 on page 82. The discretization of surface tension should react rather sensitively to such disturbances and thus the following examples also serve as a validation of the quality of the sub-triangulation.

The initial interface $\Gamma^{h,0}$ at $t = 0$ is the planar set of all points $(x, y, 0.2)$ located within the approximation Ω^h of the cylindrical domain.

The definition of the mesh levels for the underlying fundamental meshes and the listing of the average and maximum time step sizes are given in table 5.16 on page 137 analogous to the two-dimensional case. In all cases, the finite element basis $(Z, \mathbf{X}, Y) := (\mathbf{Q}^2, (\mathbf{Q}^1)^2, \mathbf{Q}_+^{0,k})$ was used exclusively. The geometry mesh $\mathcal{T}_\varkappa$ was generally chosen according to $2\,\varkappa = h$.

The system dynamics is visualized by the snapshots of the pressure field in figure 5.17 on page 138 and the z-velocity field in figure 5.18 on page 139. In analogy to the two-dimensional case, the spurious velocities occurring for the quasi-geometric method discretization (QGM) of surface tension are up to two orders of magnitude higher than for the narrow band projection method (NBP). However, while in the two-dimensional case these disturbances did not appear to have any significant influence on the position and form of the discrete interface, here they apparently introduce a slip of the interface into the non-wetting fluid.

However, these deformations of the interface result only in a thin layer situated slightly

	$Err(\Delta\overline{P})$		$Err(A)$		$\|\mathbf{u}_h\|_\infty$	
	QGM	NBP	QGM	NBP	QGM	NBP
Level 1	41.94 (17.12%)	2.68 (1.09%)	3.75e-2 (14.93%)	2.25e-2 (8.94%)	1.137	0.245
Level 2	2.30 (0.94%)	2.97 (1.21%)	4.62e-2 (18.41%)	2.40e-2 (9.55%)	1.218	0.095
Level 3	7.41 (3.02%)	7.20 (2.94%)	13.76e-2 (54.75%)	0.47e-2 (1.86%)	1.945	0.042

Table 5.17: To allow a quantitative assessment of the three-dimensional capillary tube example, the values for the jump of the averaged pressures in each phase, the interface area (length), and the infinity norms of the velocity field as shown in figure 5.16 were averaged over the time interval $t \in [0.6, 1.0]$ and their deviation from the reference value is listed in the table above for all mesh levels (the relative deviation is always appended in parenthesis). All simulations were performed for both the quasi-geometric (QGM) and the narrow band projection method (NBP) for the discretization of surface tension. Reference values were computed according to the expected final state of the interface given by a semicircular shape with a pressure jump according to the Young-Laplace law. For the given setup, this corresponds to an interface area (length) of $A = 2\,\pi 0.2^2 \approx 0.251$ and a pressure jump of $\Delta P = 2\,\sigma_s/0.2 = 245$.

above the domain boundary. As the microscopic contact angle of $\alpha_c = 0$ corresponds to a perfectly *wetting* boundary, this rather awkward and unexpected shape of the interface does not truly correspond to an error in the approximation of the underlying model: If the boundary is assumed to be covered by a microscopic layer of the wetting phase, it does not matter whether the discrete interface of the computed approximation is positioned slightly above the domain boundary (i.e. inside of the domain) or slightly below it (i.e. outside of the domain). However, the numerical implications of these *sticky* interfaces for the volume balance and the resulting spurious velocities appear to be quite problematic.

The graphs in figure 5.19 reveal that the vast error in the surface area of the discrete interface (eventually over 50%) does not correspond to similarly high error in the approximation of the jump of average pressures which is similarly good for both methods (at least on the finest two mesh levels). It should be pointed out that according to the aforementioned argument, the actual computation of the error in the surface area is not very robust (i.e. fair to the QGM method) as it does not distinguish those *sticky* parts of the interface which are effectively positioned on the domain boundary.

The assessed errors of the time averaged interface area and pressure jumps are given in table 5.17 on the preceding page. Leaving aside the errors in the interface area, the data is different from the two-dimensional case in that the QGM method does not show a similarly robust convergence behavior in the jump of the average pressures. Here, the NBP method again provides good ($< 3\%$) approximations on all mesh levels but no observable convergence. Whether the apparent offset is due to sub-optimal regularization parameters or an unavoidable result of the regularization (possibly in the projection of the normal vector field $\mathbf{n}_\varphi^{k,*}$) can not be determined based on the available data.

5.6 2D Experiments on Pore-Geometries

The UDG method, as originally presented in [18, 67], was designed for the application in complex geometries encountered in both natural and industrial porous media. In [20], an application of laminar and stationary Stokes flow to a pore-geometry obtained via

Description	UDG Sim.	Sim. [116]	Exp. [116]
Model length	400	400	30 mm
Model width	400	400	30 mm
Obstacle diam.	27	27	2 mm
Contact angle α_c	$0°$	$33°$	$< 70°$
Wetting fluid viscosity μ_1	30	0.3	$3.6 \cdot 10^{-3}$ Pa · s
Non-wetting fluid viscosity μ_2	30	0.3	$1.9 \cdot 10^{-5}$ Pa · s
Wetting fluid density ρ_1	3.0	3.0	$1.10 \cdot 10^{3}$ kg · m^{-3}
Non-wetting fluid density ρ_2	3.0	3.0	1.2 kg · m^{-3}
Surface energy σ_s	0.7	0.7	$6.2 \cdot 10^{-2}$ N · m^{-1}

Table 5.18: The parameters listed above describe the hydraulic models employed for the simulation on a 2D pore-geometry in section 5.6. For the implemented UDG approach, the fluid viscosity values had to be further increased in order to allow a stable time stepping. The employed implementation was restricted to a microscopic contact angle $\alpha_c = 0$.

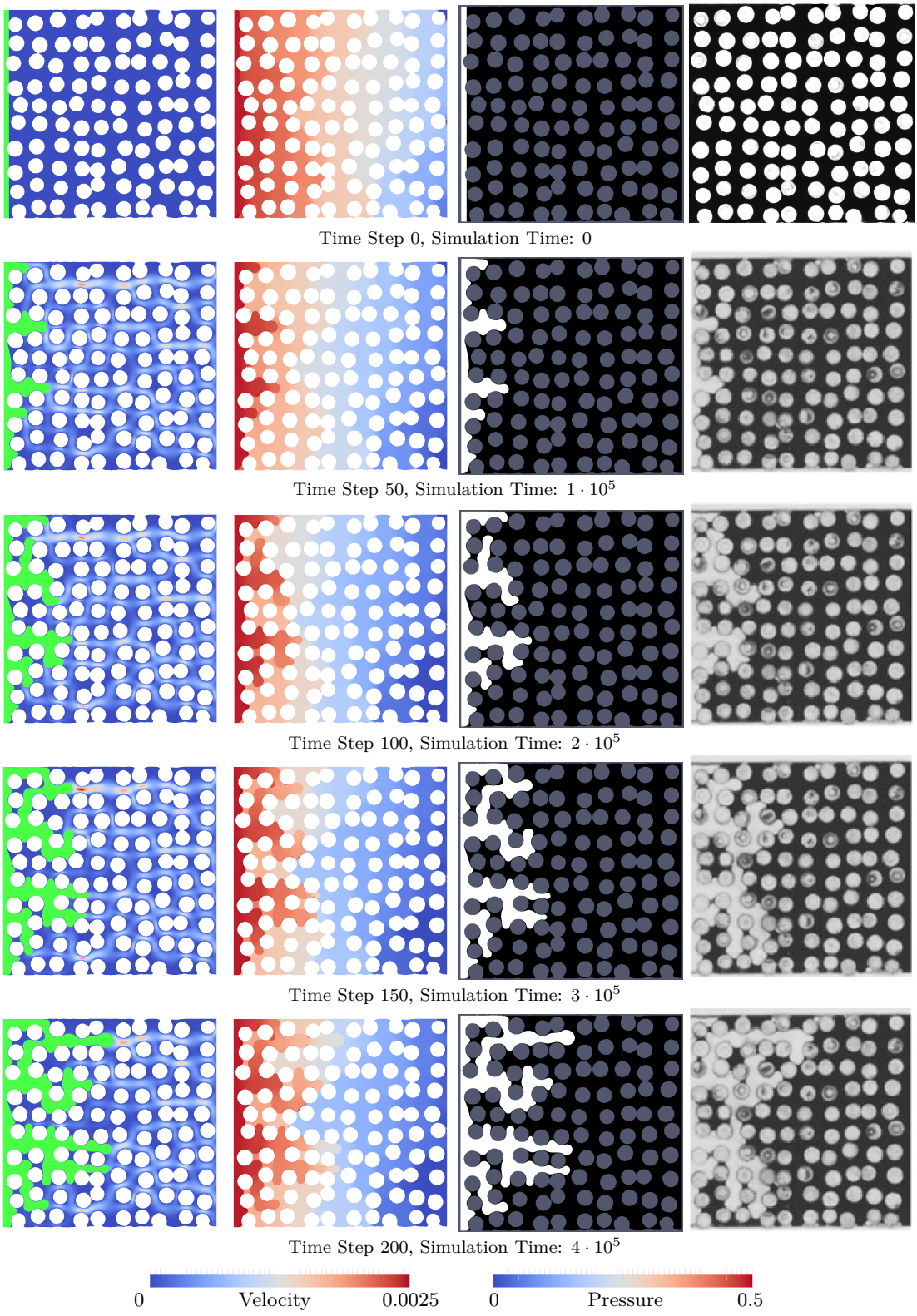


Figure 5.20: The snapshots above show the velocity field (only in wetting phase), the pressure field and the phase-field distribution of the pore-geometry simulation described in section 5.6. The pictures to the very right show the phase distribution as observed during the experiment conducted in [116] which was performed in combination with a lattice Boltzmann simulation (the phase distribution obtained by their simulation is optically identical for the presented time interval). Notice that the UDG simulation was based on a different hydraulic model, given in table 5.18.

X-ray tomography was presented. The results indicated that the considered geometry, despite its size being at the limit set by the hardware resources, was yet not large enough to comprise a *representative elementary volume* (REV) for the sampled porous media. Furthermore, the linear Krylov-space solver employed for these examples appeared sensitive to the domain geometry and did not show a robust convergence behavior. These factors taken together reveal that the UDG approach still requires further development before it may be applied in productive simulations in the context of numerical upscaling of pore-geometries - even for linear and well defined models.

The simulation of two-phase flow problems is further complicated by the heterogeneity of the viscosity and density fields and entails the additional effort of propagating the level set or phase-field function. In case of the presented UDG approach, additional costs are also produced by the construction of new sub-triangulations in every time step. While the run-time required for the construction of the actual geometries is negligible for typical simulation setups, it severely reduces the performance of classical caching strategies for the assembling of the linear systems and also increases the complexity of their implementation. All these problems are amplified by the increased number of time steps required for two-phase flow simulations. Due to the typically applied operator splitting of the flow and transport problems which is also utilized in this thesis, the time step sizes do not only have to be much smaller than in the case of single-phase flow, but are subjected to a stability condition which scales $\propto \Delta t \leq h^{3/2}$.

While none of the problems above is so fundamental that a resourceful mind may not conceive a path towards a possible solution, it is undebatable that current publications on two-phase flow simulations in pore-geometries which refer to any kind of relevant experimental setup are exclusively based on lattice Boltzmann methods or low order finite volume schemes based on volume-of-fluid approximations, e.g. [117, 118, 119]. While fair and comprehensive comparisons of Lattice-Boltzmann methods with Galerkin approaches as given in [120] are sparse, their continuous application in industry suggests a predictive capability at least for some fluid mechanical regimes [121]. Furthermore, it is quite surprising that the actual accuracy required for the approximation of the hydraulic dynamics is often very low as long as one is satisfied with only determining principal flow paths or final phase distributions (this is also the likely cause for the unbroken popularity of *quasi-stationary* methods [122]).

The authors of [116] fabricated an effectively two-dimensional porous medium from plexi-glass with dimensions of $30\,\text{mm} \times 30\,\text{mm}$ and approximately 10×10 randomly distributed pores. For this Hele-Shaw cell with a channel height of $1.75\,\text{mm}$ they performed drainage experiments. The medium was initially saturated with a glycerol-water solution which was subsequently invaded by air due to a constant drainage flux applied to one side of the cell.

Utilizing a Lattice-Boltzmann code, they were able to accurately reproduce the time development of the phase-distribution up to visual accuracy with small localized exceptions. Surprisingly, this simulation was based on a severely modified hydraulic model with fluid parameters changed by orders of magnitude and a resulting Capillary number of $Ca = 7.6 \cdot 10^{-3}$ which differed significantly from the value for the experimental setup $Ca \approx 2.6 \cdot 10^{-5}$.

Setting aside the insights which this experiment reveals with regard to the peculiar properties of flow in porous media, it is also interesting for its rather well defined two-dimensional geometry which is small enough to allow the application of the UDG two-phase flow implementation in its current state. Sadly, the employed viscosity values relative to the surface tension forces were yet too small to allow a stable simulation at reasonable time step sizes. Hence, the physical parameters were further adapted by increasing the viscosity as described in table 5.18 on page 142.

For the UDG simulation, the geometry was realized based on a gray-scale photograph with 845×847 pixels which was generously made available to us by Olav Aursjo. After a simple post-processing of this picture including a smoothing downscale to 513×513 pixels and a contrast manipulation, both performed with standard image manipulation software, the picture could be directly employed in our software as a description of the computational domain. This illustrates a key performance factor of the UDG approach considering that the application of a standard finite element method would have at least required a meshing and - keeping in mind the gray-scale reference picture - possibly an additional hand operated manipulation of the resulting mesh geometry.

While the lattice Boltzmann simulation required a rather complicated mechanism in order to enforce a constant flux rate on the outflow boundary, within the Galerkin framework the outflow condition was simply realized by a constant velocity Dirichlet condition on the right side of the domain with $u_x = 1.3379 \cdot 10^{-4}$.

The UDG simulation were performed based on the conservative level set method and the hybrid mesh discretization with $(Z, \mathbf{X}, Y) := (\mathbf{Q}^2, (\mathbf{Q}^1)^2, \mathbf{Q}_+^{0,k})$ finite element spaces and a fundamental mesh $\mathcal{T}_h$ with 256×256 cells. The geometry mesh $\mathcal{T}_\varkappa$ was adapted to the reference picture and thus spanned 512×512 cells. Regularization parameters were chosen according to (5.9).

The authors of [116] do not document according to which procedure or criteria the physical parameters employed for their Lattice-Boltzmann simulation were chosen. While it is futile to speculate on the influence of the individual parameters based on the single presented simulation, their choice for the microscopic contact angle of $\alpha_c = 33°$ is bound to be of significant influence. As the implementation of the UDG method allowed only for a contact angle of $\alpha_c = 0°$, a real comparison of the two methods was not possible.

The results of the simulations were visualized in figure 5.20 on page 143 together with photographs of the phase distribution as observed during the experiment. The presented results encompass 200 time steps with an average time step size $\langle \Delta t \rangle = 1.4676e{+}02$ and a maximum time step size of $\max \Delta t = 1.9453e + 02$ (time step sizes were chosen according to the heuristic described in section 5.4.4). The wall-time required for the simulation to this point was 33 days (sequential run on an *Intel(R) Xeon(R) CPU E7- 4870 @ 2.40GHz* processor).

In this context it should be mentioned that the Krylov space methods applied for the solution of the linear systems were not always convergent for this simulation. However, due to the small domain size, the application of the direct linear solver *SuperLU* was possible and exclusively used for the presented simulation results.

The simulation in [116] was able to predict the time development of the phase distribution quite well (although on an arbitrary time scale which had to be matched to the

experimental observations) and identified the eventually invaded pores almost without error. As the UDG simulation employed different parameter choices for both the contact angle and the viscosity, the two simulations of this high capillarity setup can not be expected to provide matching predictions of the interface evolution. However, the pictures in figure 5.20 indicate some agreement with the experimental results during its early phase (on an arbitrary time-scale).

While the presented results of the UDG simulation may not be verified based on the data presented in [116], it illustrates that the proposed UDG method for two-phase flow with contact lines may be applied to geometries obtained from experimental data without the need for direct meshing or any additional *hand-operated* manipulations of the cut-cell mesh.

6 Conclusions & Outlook

The method for two-phase flow simulations as presented in this thesis combines a sophisticated algorithm for the spatial discretization of the domain geometry with standard discontinuous Galerkin methods for the discretization of the system equations. The latter were adapted only where they conflicted with the peculiar needs and restrictions of the multi-domain UDG approach. It was shown that the recursive construction of multi-domain geometries based on an arbitrary number of level set functions is no algorithmic dead end. While the geometric complexity of this procedure entails many pitfalls, the formalization provided in this thesis delineates multiple paths to its implementation.

Many publications in this field propose a single method and show its application to a handpicked set of test cases. However, within the framework of the UDG approach, multiple different discretization methods for the Navier-Stokes equations, the level set transport and the surface tension were proposed and compared, not only to each other but also to reference values from benchmarks and data obtained by independent research groups.

The robustness and accuracy of the proposed schemes was verified by numerous numerical examples realizing the two-phase flow simulations in two and three dimensions. The presented examples include setups equivalent or similar to those presented in recent publications of other scientists which utilized state-of-the-art Galerkin discretizations based on software frameworks that resulted from decades of continuous development.

It is noteworthy that the overall complexity of such setups which are accessible to Galerkin methods has not increased significantly within the last ten years. To the present day, there exist not a single implementation of a higher-order method for multi-phase flow simulations in non-trivial pore geometries. Due to the lack of a scalable parallel implementation, the results presented in this thesis, can not ultimately answer the question whether higher-order unfitted methods, particularly the UDG method, constitute a viable path to extent their applicability to more complex domains and more challenging parameter regimes. For the UDG method to develop its full potential, it must necessarily be extended to include parallel domain decomposition methods. Any interesting two-phase flow setup with a complex three-dimensional domain geometry is bound to exceed the computational resources provided by a single computing core of any architecture.

Furthermore, when higher-order methods are applied to systems with localized dynamics (e.g. around an interface), h-adaptivity may become mandatory to avoid efficiency bottle-necks. Alas, the implementation of such features is quite time consuming, even for standard Galerkin methods. The nested mesh geometries and recursive cell-local construction mechanisms which constitute the heart of the multi-domain UDG approach provide a significant flexibility which entails a correspondingly complex code base. The latter is a product of serious software-engineering which has reached a scale at which its advance requires laborious efforts and scales poorly with the invested human resources.

However, considering the flexibility and robustness of the presented approach, its further development is rather promising. Among the high priority topics of such efforts must be the derivation of scalable parallel solvers for the resulting linear systems. There is also a noticeable imbalance between the sophisticated spatial discretization and the rather simplistic method of lines approach employed for the discretization of time. While this would possibly turn into an advantage for some model problems, in the case of two-phase flow, the involved operator splitting results in a burdening restriction on the time

step size. From the authors perspective, a fully coupled arbitrary Lagrangian-Eulerian approach appears to be the only promising path to resolve this issue but implies a significantly increased size of the algebraic systems and is thus preconditioned on the existence of an efficient and scalable linear solver.

Nevertheless, the overall maturity of the numerical scheme as presented in this thesis is already quite evolved. The proposed discontinuous Galerkin variant of the conservative level set method could be shown to provide control over the phase volume errors and to harmonize with the discretization schemes realizing the surface tension force. Extensions of the latter that allowed the introduction of a microscopic contact angle were incorporated and verified in numerical test cases. The amount of available software frameworks which provide a comparable flexibility are sparse. The author is not aware of any other implementation of a sharp interface method with contact line which can be applied for non-trivial three-dimensional domains and realizes accurate interface conditions as well as a robust handle on volume conservation.

A Appendix

This appendix provides some additional results computed for the two-dimensional surface tension test setup presented in section 5.3. Tables A.1, A.2, and A.3 provide some additional information on the sensitivity of the regularization parameters which were employed for the discretization of surface tension. All computations were performed for the same setups and with the same finite element spaces as described in section 5.3.

These results are given as an indication for researchers developing similar methods and no specific interpretation is provided. As a general observation it should be pointed out that the response of the error quantities on changes of a given parameter ranges from small fluctuations too erratic jumps by multiple orders of magnitude.

Optimal Parameters - Varying Interface Resolution:

ΔP_{avg}^{err}	λ=3		λ=2		λ=1	
Level 2	2.17e+00		2.17e+00		1.80e+00	
Level 3	2.98e-01	2.9	5.06e-01	2.1	1.25e+00	0.5
Level 4	1.11e-01	1.4	1.75e-01	1.5	4.39e-01	1.5
Level 5	2.91e-02	1.9	4.99e-03	5.1	1.76e-01	1.3

P^{err}	λ=3		λ=2		λ=1	
Level 2	8.88e+00		5.89e+00		1.75e+01	
Level 3	1.10e+01		9.46e+00		1.40e+01	0.3
Level 4	3.02e+01		2.32e+01		2.17e+01	
Level 5	3.34e+01		3.14e+01		4.54e+01	

$\|\mathbf{v}\|_2$	λ=3		λ=2		λ=1	
Level 2	1.78e-02		1.75e-02		2.71e-02	
Level 3	5.26e-03	1.8	4.66e-03	1.9	7.12e-03	1.9
Level 4	5.69e-03		5.15e-03		5.54e-03	0.4
Level 5	5.52e-03		4.89e-03	0.1	5.58e-03	

Parameter Sensitivity Runs:

ΔP_{avg}^{err}	$\varepsilon_{\mathbf{n}}$=0.4 h^2		$\varepsilon_{\mathbf{n}}$=1.6 h^2		ε_{φ}=0.6 h		ε_{φ}=1.2 h	
Level 2	2.52e+00		1.76e+00		1.00e+01		6.54e+01	
Level 3	5.62e-01	2.2	6.71e-01	1.4	4.15e-01	4.6	5.22e+01	0.3
Level 4	2.93e-01	0.9	4.66e-01	0.5	8.32e-02	2.3	1.73e+00	4.9
Level 5	2.50e-02	3.6	1.32e-01	1.8	1.29e-01		2.43e+00	

P^{err}	$\varepsilon_{\mathbf{n}}$=0.4 h^2		$\varepsilon_{\mathbf{n}}$=1.6 h^2		ε_{φ}=0.6 h		ε_{φ}=1.2 h	
Level 2	1.96e+01		2.52e+01		9.04e+01		5.03e+03	
Level 3	9.27e+00	1.1	3.95e+01		6.67e+01	0.4	4.89e+04	
Level 4	1.67e+01		2.08e+01	0.9	5.15e+01	0.4	1.11e+03	5.5
Level 5	3.55e+01		2.94e+01		4.65e+01	0.1	7.44e+02	0.6

$\|\mathbf{v}\|_2$	$\varepsilon_{\mathbf{n}}$=0.4 h^2		$\varepsilon_{\mathbf{n}}$=1.6 h^2		ε_{φ}=0.6 h		ε_{φ}=1.2 h	
Level 2	2.57e-02		1.84e-02		2.26e-02		8.18e-02	
Level 3	7.68e-03	1.7	2.53e-02		7.71e-03	1.6	1.07e-01	
Level 4	4.67e-03	0.7	9.06e-03	1.5	2.34e-03	1.7	4.87e-02	1.1
Level 5	5.21e-03		3.90e-03	1.2	1.27e-03	0.9	1.58e-02	1.6

Table A.1: Quasi-Geometric Method: Errors of the jump in the average pressure ΔP_{avg}^{err} (5.7), maximum deviation of average pressure P^{err} (5.8) and L_2-norm of velocity field $\|\mathbf{v}\|_2$. Each value is followed by the rate of convergence in h relative to the previous mesh level (omitted where it would be negative). Unless explicitly defined in the column header, parameters were set to $\varepsilon_\kappa = 0.01\,h^2$, $\varepsilon_{\mathbf{n}} = 0.1\,h^2$, $\varepsilon_\varphi = 0.3\,h$, $\beta = 0.25$, $\lambda = 2$.

Optimal Parameters - Varying Interface Resolution:

$\Delta P^{err}_{avg}(t=T)$	λ=3		λ=2		λ=1	
Level 2	3.94e+01		4.10e+01		4.67e+01	
Level 3	1.95e+01	1.0	1.97e+01	1.1	2.09e+01	1.2
Level 4	4.56e+00	2.1	4.60e+00	2.1	4.89e+00	2.1
Level 5	3.06e-01	3.9	3.27e-01	3.8	2.49e-01	4.3

$P^{err}(t=T)$	λ=3		λ=2		λ=1	
Level 2	9.85e+00		9.33e+00		8.14e+00	
Level 3	3.83e-01	4.7	9.43e-01	3.3	1.24e+00	2.7
Level 4	2.91e+00		3.00e+00		6.80e+00	
Level 5	2.62e+00	0.2	2.79e+00	0.1	8.56e-01	3.0

$\|\mathbf{v}\|_2(t=T)$	λ=3		λ=2		λ=1	
Level 2	2.75e-03		2.73e-03		3.60e-03	
Level 3	7.75e-04	1.8	8.19e-04	1.7	1.16e-03	1.6
Level 4	3.33e-04	1.2	3.18e-04	1.4	8.12e-04	0.5
Level 5	2.09e-04	0.7	1.06e-04	1.6	9.10e-05	3.2

Parameter Sensitivity Runs:

$\Delta P^{err}_{avg}(t=T)$	ε_φ=1.2 h		ε_φ=2.4 h		ε_κ=0.16 h^2		ε_κ=0.64 h^2	
Level 2	6.66e+01		1.30e+02		2.62e+01		8.51e+00	
Level 3	3.69e+01	0.9	1.23e+02	0.1	2.06e+01	0.3	1.33e+01	
Level 4	1.43e+01	1.4	4.35e+01	1.5	2.57e+00	3.0	9.69e+00	0.5
Level 5	3.73e+00	1.9	4.49e+01		2.86e+00		1.60e+01	

$P^{err}(t=T)$	ε_φ=1.2 h		ε_φ=2.4 h		ε_κ=0.16 h^2		ε_κ=0.64 h^2	
Level 2	2.59e+01		2.22e+02		1.11e+01		8.17e+00	
Level 3	1.45e+00	4.2	1.39e+02	0.7	4.01e-01	4.8	1.19e+00	2.8
Level 4	3.17e+01		7.62e+01	0.9	2.72e+00		6.35e-01	0.9
Level 5	1.23e+01	1.4	1.04e+02		1.93e+00	0.5	9.52e-01	

$\|\mathbf{v}\|_2(t=T)$	ε_φ=1.2 h		ε_φ=2.4 h		ε_κ=0.16 h^2		ε_κ=0.64 h^2	
Level 2	4.91e-03		7.82e-03		3.05e-03		3.30e-03	
Level 3	4.07e-04	3.6	1.67e-02		7.51e-04	2.0	3.60e-04	3.2
Level 4	1.96e-03		5.09e-03	1.7	1.56e-04	2.3	1.99e-04	0.9
Level 5	5.55e-04	1.8	1.55e-02		8.71e-05	0.8	6.55e-05	1.6

Table A.2: Narrow Band Projection Method: Errors of the jump in the average pressure ΔP^{err}_{avg} (5.7), maximum deviation of average pressure P^{err} (5.8) and L_2-norm of velocity field $\|\mathbf{v}\|_2$. Each value is followed by the rate of convergence in h relative to the previous mesh level (omitted where it would be negative). Unless explicitly defined in the column header, parameters were set to, $\varepsilon_\kappa = 0.04\,h^2$, $\varepsilon_\mathbf{n} = 0.4\,h^2$, $\varepsilon_\varphi = 0.6\,h$, $\beta = 0.08$, $\lambda = 2$

Optimal Parameters - Varying Interface Resolution:

$\Delta P^{err}_{avg}(t=T)$	λ=3		λ=2		λ=1	
Level 2	2.53e+01		2.46e+01		2.15e+01	
Level 3	4.46e-01	5.8	7.03e-01	5.1	1.18e+00	4.2
Level 4	1.98e+00		2.02e+00		2.23e+00	
Level 5	6.98e-01	1.5	6.92e-01	1.5	6.17e-01	1.9

P^{err}	λ=3		λ=2		λ=1	
Level 2	4.18e-01		5.97e-01		2.85e+00	
Level 3	1.05e+00		1.39e+00		6.60e+00	
Level 4	8.25e-01	0.4	9.44e-01	0.6	9.17e-01	2.8
Level 5	8.54e-01		1.05e+00		1.38e+00	

$\|\mathbf{v}\|_2(t=T)$	λ=3		λ=2		λ=1	
Level 2	3.83e-04		4.86e-04		2.61e-03	
Level 3	1.15e-03		1.17e-03		1.33e-03	1.0
Level 4	1.68e-04	2.8	2.11e-04	2.5	2.30e-04	2.5
Level 5	1.23e-04	0.4	1.24e-04	0.8	8.98e-05	1.4

Parameter Sensitivity Runs:

$\Delta P^{err}_{avg}(t=T)$	$\varepsilon_{\mathbf{n}} = 1.6h^2$		$\varepsilon_{\mathbf{n}} = 6.4h^2$		$\varepsilon_{\varphi} = 1.2h$	
Level 2	5.99e+01		2.46e+01		3.65e+01	
Level 3	2.24e+01	1.4	7.03e-01	5.1	5.30e+01	
Level 4	5.21e+00	2.1	2.02e+00		1.55e+01	1.8
Level 5	9.21e+00		6.92e-01	1.5	4.30e+00	1.9

$\Delta P^{err}_{avg}(t=T)$	ε_{κ}=0.16 h^2		ε_{κ}=0.64 h^2	
Level 2	4.11e+01		6.91e+01	
Level 3	2.74e+00	3.9	2.12e+01	1.7
Level 4	1.92e+00	0.5	1.82e+00	3.5
Level 5	1.28e+00	0.6	6.06e+00	

P^{err}	$\varepsilon_{\mathbf{n}} = 1.6\,h^2$		$\varepsilon_{\mathbf{n}} = 6.4\,h^2$		$\varepsilon_{\varphi} = 1.2\,h$	
Level 2	3.94e-01		5.97e-01		3.22e+00	
Level 3	9.80e-01		1.39e+00		1.02e+01	
Level 4	9.66e-01		9.44e-01	0.6	2.30e+01	
Level 5	6.54e-01	0.6	1.05e+00		2.78e+01	

P^{err}	$\varepsilon_{\kappa} = 0.16h^2$		$\varepsilon_{\kappa} = 0.64h^2$	
Level 2	3.89e-01		6.22e-02	
Level 3	1.30e+00		1.32e+00	
Level 4	1.02e+00	0.4	1.33e+00	
Level 5	9.10e-01	0.2	1.07e+00	0.3

$\|\mathbf{v}\|_2(t=T)$	$\varepsilon_{\mathbf{n}} = 1.6\,h^2$		$\varepsilon_{\mathbf{n}}$= $6.4\,h^2$		$\varepsilon_{\varphi} = 1.2\,h$	
Level 2	3.58e-04		4.86e-04		1.16e-03	
Level 3	8.83e-04		1.17e-03		3.16e-03	
Level 4	6.05e-04	0.5	2.11e-04	2.5	1.84e-03	0.8
Level 5	5.57e-05	3.4	1.24e-04	0.8	8.84e-04	1.1

$\|\mathbf{v}\|_2(t=T)$	ε_{κ}=0.16 h^2		ε_{κ}=0.64 h^2	
Level 2	3.59e-04		8.67e-05	
Level 3	1.45e-03		1.25e-03	
Level 4	1.83e-04	3.0	3.98e-04	1.7
Level 5	8.07e-05	1.2	7.79e-05	2.4

Table A.3: Global Projection Method: Errors of the jump in the average pressure ΔP^{err}_{avg} (5.7), maximum deviation of average pressure P^{err} (5.8) and L_2-norm of velocity field $\|\mathbf{v}\|_2$. Each value is followed by the rate of convergence in h relative to the previous mesh level (omitted where it would be negative). Unless explicitly defined in the column header, parameters were set to $\varepsilon_{\kappa} = 0.04\,h^2$, $\varepsilon_{\mathbf{n}} = 0.4\,h^2$, $\varepsilon_{\varphi} = 0.6\,h$, $\lambda = 2$.

Bibliography

[1] Sussman M, Puckett EG. A coupled level set and volume-of-fluid Method for computing 3D and axisymmetric incompressible two-phase flows. *J. Comput. Phys.* Aug 2000; **162**(2):301–337.

[2] Sussman M. A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *J. Comput. Phys.* May 2003; **187**(1):110–136.

[3] Croce R, Griebel M, Schweitzer MA. Numerical simulation of bubble and droplet deformation by a level set approach with surface tension in three dimensions. *Int. J. Numer. Meth. Fluids* 2010; **62**:963–993.

[4] Verleye B, Klitz M, Croce R, Roose D, Lomov SV, Verpoest I. Computation of the permeability of textiles with experimental validation for monofilament and non crimp fabrics. *Computational Textile*, vol. 55. Springer Berlin / Heidelberg, 2007; 93–109.

[5] Verleye B, Croce R, Griebel M, Klitz M, Lomov SV, Morren G, Sol H, Verpoest I, Roose D. Permeability of textile reinforcements: simulation, influence of shear, validation. *Composites science and technology*, Elsevier Applied Science Publishers, 2008.

[6] Inamuro T, Ogata T, Tajima S, Konishi N. A lattice Boltzmann method for incompressible two-phase flows with large density differences. *J. Comput. Phys.* 2004; **198**(2):628–644.

[7] Lee T, Lin CL. A stable discretization of the lattice Boltzmann equation for simulation of incompressible two-phase flows at high density ratio. *J. Comput. Phys.* 2005; **206**(1):16–47.

[8] Huang H, Zheng H, Lu Xy, Shu C. An evaluation of a 3D free-energy-based lattice Boltzmann model for multiphase flows with large density ratio. *Int. J. Numer. Meth. Fluids* 2010; **63**(10):1193–1207.

[9] Bänsch E. Finite element discretization of the {Navier-Stokes} equations with a free capillary surface. *Numer. Math.* 2001; **88**:203–235.

[10] Ganesan S. Finite element methods on moving meshes for free surface and interface flows. PhD Thesis, Otto-von-Guericke Universität Magdeburg 2006.

[11] Matthies G. Finite element methods for free boundary value problems with capillary surfaces. PhD Thesis, Otto-von-Guericke-Universität, Fakultät für Mathematik, Magdeburg 2002.

[12] Nguyen V, Peraire J, Khoo BC, Persson P. A discontinuous Galerkin front tracking method for two-phase flows with surface tension. *Computers and Fluids* 2010; **39**(1):1–14.

[13] Sollie WEH, Bokhove O, van der Vegt JJW. Two fluid space-time discontinuous Galerkin finite element method. Part I: Numerical algorithm. *Memorandum (University of Twente)* 2009; **1909**.

[14] Sollie WEH, van der Vegt JJW. Two fluid space-time discontinuous Galerkin finite element method. Part II: Applications. *Memorandum (University of Twente)* Nov 2009; **1910**.

[15] Osher SJ, Sethian JA. Fronts propagating with curvature dependent speed: algorithms based on Hamilton–Jacobi formulations. *J. Comput. Phys.* 1988; **79**:12–49.

[16] Hysing SR. Numerical simulation of immiscible fluids with FEM level set techniques. PhD Thesis, Universität Dortmund 2007.

[17] Parolini N. Computational fluid dynamics for naval engineering problems. PhD Thesis, École Polytechnique Fédérale de Lausanne 2004.

[18] Engwer C, Bastian P. An unfitted finite element method using discontinuous Galerkin. *Int. J. Numer. Meth. Eng.* 2009; **79**:1557–1576.

[19] Bastian P, Engwer C, Fahlke J, Ippisch O. An unfitted discontinuous Galerkin method for pore-scale simulations of solute transport. *Mathematics and Computers in Simulation* Jun 2011; **81**(10):2051–2061.

[20] Engwer C. An unfitted discontinuous Galerkin scheme for micro-scale simulations and numerical upscaling. PhD Thesis, Ruprecht-Karls Universität Heidelberg 2009.

[21] Wheeler MF. An elliptic collocation-finite element method with interior penalties. *SIAM J. Numer. Anal.* 1978; **15**:152–161.

[22] Rivière B, Girault V. Discontinuous Finite Element Methods for Incompressible Flows on subdomains with non-matching interfaces. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(25-28):3274–3292.

[23] Girault V, Rivière B, Wheeler MF. A splitting method using discontinuous Galerkin for the transient incompressible Navier-Stokes equations. *ESAIM: Mathematical Modelling and Numerical Analysis* 2004; **39**(6):1115–1147.

[24] Engwer C, Heimann F. DUNE-UDG: A cut-cell framework for unfitted discontious Galerkin methods. *Advances in DUNE - Proceedings of the DUNE User Meeting*, Springer Verlag, 2011.

[25] Fries TP, Zilian A. On time integration in the XFEM. *Int. J. Numer. Meth. Fluids* 2009; **79**(1):69–93.

[26] Groß, Reusken A. An extended pressure finite element space for two-phase incompressible flows with surface tension. *J. Comput. Phys.* May 2007; **224**(1):40–58.

[27] Groß. Numerical methods for three-dimensional incompressible two-phase flow problems. PhD Thesis, RWTH Aachen 2008.

[28] Sollie WEH, Bokhove O, van der Vegt JJW. Space-time discontinuous Galerkin finite element method for two-fluid flows. *J. Comput. Phys.* 2011; **230**(3):789–817.

[29] Heimann F. An unfitted discontinuous Galerkin method for two-phase flow. Master's Thesis, University of Heidelberg 2009.

[30] Shikhmurzaev YD, Sprittles JE. Wetting front dynamics in an isotropic porous medium. *Journal of Fluid Mechanics* 2012; **694**:399–407.

[31] Sprittles JE, Shikhmurzaev YD. Finite element framework for describing dynamic wetting phenomena. *Int. J. Numer. Meth. Fluids* 2012; **68**(2011):1257–1298.

[32] Fried E, Jabbour M. Dynamical equations for the contact line of an evaporating or condensing sessile drop. *Journal of Fluid Mechanics* 2012; **703**:1–34.

[33] Yokoi K, Vadillo D, Hinch J, Hutchings I. Numerical studies of the influence of the dynamic contact angle on a droplet impacting on a dry surface. *Physics of Fluids* 2009; **21**(7):072 102.

[34] Hysing S, Turek S, Kuzmin D, Parolini N, Burman E, Ganesan S, Tobiska L. Quantitative benchmark computations of two-dimensional bubble dynamics. *Int. J. Numer. Meth. Fluids* 2009; **60**(11):1259–1288.

[35] Heimann F, Engwer C, Ippisch O, Bastian P. An unfitted interior penalty discontinuous Galerkin method for incompressible Navier-Stokes two-phase flow. *Int. J. Numer. Meth. Fluids* 2012; **71**(3):269–293.

[36] Peskin CS. Flow patterns around heart valves : A digital computer method for solving the equations of motion. PhD Thesis, Albert Einstein College of Medicine, Yeshiva University 1972.

[37] Young D, Melvin R, Bieterman M, Johnson F, Saman S, Bussoletti J. A locally refined rectangular grid Finite Element method: Application to computational fluid dynamics and computational physics. *J. Comput. Phys.* 1991; **92**(1):1–66.

[38] Chiang YL, van Leer B, Powell KG. Simulation of unsteady inviscid flow on an adaptively refined cartesian grid. *30th Aerospace Sciences Meeting & Exhibit*, 1992.

[39] Zeeuw DD, Powell KG. An adaptively refined Cartesian mesh solver for the Euler equations. *J. Comput. Phys.* 1993; **104**:56–58.

[40] Ingram D, Causon D, Mingham C. Developments in Cartesian cut cell methods. *Mathematics and Computers in Simulation* Jan 2003; **61**(3-6):561–572.

[41] Mittal R, Iaccarino G. Immersed Boundary Methods. *Annual Review of Fluid Mechanics* Jan 2005; **37**(1):239–261.

[42] Lew A, Buscaglia G. A discontinuous-Galerkin-based immersed boundary method. *Int. J. Numer. Meth. Eng.* 2008; **76**(4):427–454.

[43] Nitsche J. Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg* 1971; **36**(1):9–15.

[44] Babuska I. The Finite Element Method for Elliptic Equations with Discontinuous Coefficients. *Computing* 1970; **5**:207–213.

[45] Hansbo A, Hansbo P. An unfitted finite elementmethod for elliptic interface problems. *Comput. Methods Appl. Mech. Engrg.* 2002; **191**:5537 – 5552.

[46] Arnold DN. An interior penalty finite element method with discontinuous elements. *SIAM J. Numer. Anal* 1982; **19**(4):742–760.

[47] Oden T, Babuska I, Baumann CE. A Discontinuous hp Finite Element Method for Diffusion Problems. *J. Comput. Phys.* 1998; **519**:491–519.

[48] Rivière B, Wheeler MF. A Discontinuous Galerkin method applied to nonlinear parabolic equations. *Theory, Computation and Applications* 1999; **11**:231–244.

[49] Girault V, Rivière B, Wheeler MF. A discontinuous Galerkin method with nonoverlapping domain decomposition for the Stokes and Navier-Stokes problems. *Mathematics of Computation* 2004; **74**(249):53–84.

[50] Arnold DN, Brezzi F, Cockburn B, Marini D. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis* 2002; **39**(5):1749–1779.

[51] Epshteyn Y, Rivière B. Estimation of penalty parameters for symmetric interior penalty Galerkin methods. *Journal of Computational and Applied Mathematics* Sep 2007; **206**(2):843–872.

[52] Ern A, Stephansen AF, Zunino P. A discontinuous Galerkin method with weighted averages for advection-diffusion equations with locally small and anisotropic diffusivity. *IMA Journal of Numerical Analysis* Apr 2008; **29**(2):235–256.

[53] Cockburn B, Gopalakrishnan J, Lazarov R. Unified hybridization of discontinuous Galerkin, mixed and continuous Galerkin methods for second order elliptic problems. *SIAM J. Numer. Anal* 2009; **47**(2):1319–1365.

[54] Huynh L, Nguyen N, Peraire J. A high-order hybridizable discontinuous Galerkin method for elliptic interface problems. *Int. J. Numer. Meth. Eng.* 2013; **93**(2):183–200.

[55] Engwer C, Bastian P. An unfitted finite element method using discontinuous Galerkin. *Int. J. Numer. Meth. Eng.* 2009; **79**:1557–1576.

[56] Bassi F, Rebay S. High-Order Accurate Discontinuous Finite Element Solution of the 2D Euler Equations. *J. Comput. Phys.* Dec 1997; **138**(2):251–285.

[57] Sherwin SJ, Peiro J. Mesh generation in curvilinear domains using high-order elements. *Int. J. Numer. Meth. Eng.* Jan 2002; **53**(1):207–223.

[58] Shephard MS, Flaherty JE, Jansen KE, Li X, Luo X, Chevaugeon N, Remacle JF, Beall MW, O'Bara RM. Adaptive mesh generation for curved domains. *Applied Numerical Mathematics* Feb 2005; **52**(2-3):251–271.

[59] Dey S, Shephard MS, Flaherty JE. Geometry representation issues associated with p-version finite element computations. *Computer Methods in Applied Mechanics and Engineering* Dec 1997; **150**(1–4):39–55.

[60] Sethian JA. Level Set Methods and Fast Marching Methods. Cambridge University Press, 1999.

[61] Hughes TJR, Cottrell JA, Bazilevs Y. Isogeometric Analysis: Toward Integration of CAD and FEA. John Wiley & Sons, 2009.

[62] Benson DJ, Bazilevs Y, De Luycker E, Hsu MC, Scott M, Hughes TJR, Belytschko T. A generalized finite element formulation for arbitrary basis functions: From isogeometric analysis to XFEM. *Int. J. Numer. Meth. Eng.* 2010; **83**(6):765–785.

[63] Shahmiri S, Gerstenberger A, Wall W. An XFEM-based embedding mesh technique for incompressible viscous flows. *Int. J. Numer. Meth. Fluids* 2011; **65**(2010):166–190.

[64] Eschenauer Ha, Olhoff N. Topology optimization of continuum structures: A review. *Applied Mechanics Reviews* 2001; **54**(4):331.

[65] Parvizian J, Düster A, Rank E. Topology optimization using the finite cell method. *Optimization and Engineering* 2012; **13**(1):57–78.

[66] Cline HE, Lorensen WE. Marching Cubes: A high resolution 3D surface construction algorithm. *Siggraph* 1987; **21**:163–169.

[67] Bastian P, Engwer C. Solving partial differential equations in complicated domains. *Oberwolfach Reports* 2005; **4**.

[68] Lewiner T, Lopes H, Vieira AW, Tavares G. Efficient implementation of Marching Cubes' cases with topological guarantees. *Journal of Graphics Tools* 2003; **8**(2):1–15.

[69] Shikhmurzaev YD. Capillary flows with forming interfaces. Chapman & Hall / CRC, 2008.

[70] Ladyzhenskaya O. The Mathematical Theory of Viscous Incompressible Flows. 2nd ed. edn., Gordon and Breach, 1969.

[71] Bao K, Shi Y, Sun S, Wang XP. A Finite Element Method for the Numerical Solution of the Coupled Cahn-Hilliard and Navier-Stokes System for Moving Contact Line Problems. *J. Comput. Phys.* Aug 2012; **231**(24):8083–8099.

[72] Griebel M, Klitz M. Simulation of droplet impact with dynamic contact angle boundary conditions. *Technical Report 1302*, Rheinische Friedrich-Wilhelms-Universität Bonn, INS Preprint No. 1302 2013.

[73] Huh C, Scriven LE. Hydrodynamic model of steady movement of a solid/liquid/fluid contact line. *Journal of Colloid and Interface Science* 1971; **35**(1):85–101.

[74] Hocking LM. A moving fluid interface. Part 2. The removal of the force singularity by a slip flow. *Journal of Fluid Mechanics* 1977; **79**(02):209–229.

[75] Strang G. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis* 1968; **5**(3):506–517.

[76] Li BQ. Discontinuous finite elements in fluid dynamics and heat transfer. Springer-Verlag London, 2006.

[77] Bassi F, Rebay S. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *J. Comput. Phys.* 1997; **131**:267–279.

[78] Hysing S. Mixed element FEM level set method for numerical simulation of immiscible fluids. *J. Comput. Phys.* Mar 2012; **231**(6):2449–2465.

[79] Sussman M, Smith K, Hussaini M, Ohta M, Zhi-Wei R. A sharp interface method for incompressible two-phase flows. *J. Comput. Phys.* Feb 2007; **221**(2):469–505.

[80] Croce R, Griebel M, Schweitzer MA. A parallel level-set approach for two-phase flow problems with surface tension in three space dimensions. *Technical Report*, Sonderforschungsbereich 611, Universität Bonn 2004.

[81] Ausas R, Buscaglia G, Idelsohn S. A new enrichment space for the treatment of discontinuous pressures in multi–fluid flows. *Int. J. Numer. Meth. Fluids* 2011; **70**(7):829–850.

[82] Bungartz HJ, Mehl M, Schäfer M. Fluid Structure Interaction II: Modelling, simulation, optimization (Lecture notes in computational science and engineering). Springer, 2010.

[83] Ern A, Guermond JL. Theory and practice of finite elements. Springer Verlag, 2004.

[84] Gremaud P, Kuster C, Li Z. A study of numerical methods for the level set approach. *Applied Numerical Mathematics* May 2007; **57**(5-7):837–846.

[85] Hysing SR, Turek S. The Eikonal equation: Numerical efficiency vs. algorithmic complexity on quadrilateral grids. *Proceedings of ALGORITMY*, Ls Iii, 2005; 22–31.

[86] Hartmann D, Meinke M, Schröder W. The constrained reinitialization equation for level set methods. *J. Comput. Phys.* 2010; **229**(5):1514–1535.

[87] Ausas R, Dari E, Buscaglia G. A geometric mass-preserving redistancing scheme for the level set function. *Int. J. Numer. Meth. Fluids* 2011; **65**(8):989–1010.

[88] der Pijl SP, Segal A, Vuik C, Wesseling P. Computing three-dimensional two-phase flows with a mass-conserving level set method. *Comput. Vis. Sci.* 2008; **11**(4-6):221–235.

[89] Pietro DAD, Forte SL, Parolini N. Mass preserving finite element implementations of the level set method. *Applied Numerical Mathematics* 2006; **56**:1159–1179.

[90] Rhebergen S, Bokhove O, van der Vegt JJW. Discontinuous Galerkin finite element methods for hyperbolic nonconservative partial differential equations. *J. Comput. Phys.* Jan 2008; **227**(3):1887–1922.

[91] Hirt CW, Nichols BD. Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.* 1981; **39**(1):201–225.

[92] Olsson E, Kreiss G. A conservative level set method for two phase flow. *J. Comput. Phys.* Nov 2005; **210**(1):225–246.

[93] Olsson E, Kreiss G, Zahedi S. A conservative level set method for two phase flow II. *J. Comput. Phys.* Jul 2007; **225**(1):785–807.

[94] Cockburn B, Hou S, Shu CW. The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV. The multidimensional case. *Math. Comp.* 1990; **54**(190):545–581.

[95] Cockburn B, Shu CW. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework. *Math. Comp.* 1989; **52**:411–435.

[96] Dolejsi V, Feistauer M, Sobotikova V. Analysis of the discontinuous Galerkin method for nonlinear convection–diffusion problems. *Comput. Methods Appl. Mech. Engrg.* 2005; **194**:2709–2733.

[97] Houston P, Schwab C, Süli E. Discontinuous hp-finite element methods for advection-diffusion-reaction problems. *SIAM J. Numer. Anal* 2013; **39**(6):2133–2163.

[98] Ganesan S, Tobiska L. An accurate finite element scheme with moving meshes for computing 3D-axisymmetric interface flows. *Int. J. Numer. Meth. Fluids* 2008; **57**(2):119–138.

[99] LeVeque RJ, Li Z. Immersed interface methods for Stokes flow with elastic boundaries or surface tension. *SIAM Journal on Scientific Computing* May 1997; **18**(3):709–735.

[100] Brackbill JU, Kothe DB, Zemach C. A continuum method for modeling surface tension. *J. Comput. Phys.* 1991; **100**:335–254.

[101] Croce R. Ein paralleler, dreidimensionaler Navier-Stokes-Löser für inkompressible Zweiphasenströmungen mit Oberflächenspannung, Hindernissen und dynamischen Kontaktflächen. Diplomarbeit, Institut für Angewandte Mathematik, Universität Bonn, Bonn, Germany 2002.

[102] van Sint Annaland M, Deen N, Kuipers J. Numerical simulation of gas bubbles behaviour using a three-dimensional volume of fluid method. *Chemical Engineering Science* Jun 2005; **60**(11):2999–3011.

[103] Sussman M, Almgren AS, Bell JB, Colella P, Howell LH, Welcome ML. An adaptive level set approach for incompressible two-phase flows. *J. Comput. Phys.* 1999; **148**(1):81–124.

[104] Zahedi S, Kronbichler M, Kreiss G. Spurious currents in finite element based level set methods for two-phase flow. *Int. J. Numer. Meth. Fluids* 2012; **69**:1433–1456.

[105] Sussman M, Smereka P, Osher S. A level set approach for computing solutions to incompressible two-phase flow. *J. Comput. Phys.* 1994; **114**(1):146–159.

[106] Sato Y, Ničeno B. A new contact line treatment for a conservative level set method. *J. Comput. Phys.* May 2012; **231**(10):3887–3895.

[107] Bastian P, Heimann F, Marnach S. Generic implementation of finite element methods in the distributed and unified numerics environment (dune). *Kybernetika* 2010; **46**(2):294–315.

[108] Aland S, Voigt A. Benchmark computations of diffuse interface models for two-dimensional bubble dynamics. *Int. J. Numer. Meth. Fluids* 2012; **69**(2011):747–761.

[109] Sauerland H, Fries TP. The extended finite element method for two-phase and free-surface flows: A systematic study. *J. Comput. Phys.* May 2011; **230**(9):3369–3390.

[110] Cho MH, Choi HG, Choi SH, Yoo JY. A Q2Q1 finite element/level-set method for simulating two-phase flows with surface tension. *Int. J. Numer. Meth. Fluids* 2011; **70**(4):468–492.

[111] Klostermann J, Schaake K, Schwarze R. Numerical simulation of a single rising bubble by VOF with surface compression. *Int. J. Numer. Meth. Fluids* 2013; **71**(8):960–982.

[112] Turek S, Mierka O, Hysing S, Kuzmin D. Numerical study of a high order 3D FEM-Level Set approach for immiscible flow simulation. *Technical Report*, Institut für Angewandte Mathematik, TU Dortmund 2012.

[113] Grace JR. Shapes and velocities of bubbles rising in infinite liquids. *Chemical Engineering Research and Design* 1973; **51**(a):116 – 120.

[114] Smolianski A. Numerical Modeling of Two-Fluid Interfacial Flows. PhD Thesis, University of Jyvaskyla 2001.

[115] Clift R, Grace JR, Weber M. Bubbles, Drops and Particles. Academic Press, New York, 1978.

[116] Aursjo O, Lovoll G, Knudsen HA, Flekkoy EG, Maloy KJ. A direct comparison between a slow pore scale drainage experiment and a 2D lattice Boltzmann simulation. *Transport in Porous Media* Jul 2010; **86**(1):125–134.

[117] Papafotiou A, Helmig R, Schaap J, Lehmann P, Kaestner A, Flühler H, Neuweiler I, Hassanein R, Ahrenholz B, Tölke J, *et al.*. From the pore scale to the lab scale: 3-D lab experiment and numerical simulation of drainage in heterogeneous porous media. *Advances in Water Resources* 2008; **31**(9):1253–1268.

[118] Pan C, Hilpert M, Miller CT. Lattice-Boltzmann simulation of two-phase flow in porous media. *Water Resources Research* 2004; **40**.

[119] Raeini AQ, Blunt MJ, Bijeljic B. Modelling two-phase flow in porous media at the pore scale using the volume-of-fluid method. *J. Comput. Phys.* 2012; **231**(17):5653–5668.

[120] Geller S, Krafczyk M, Turek S, Hron J. Benchmark computations based on lattice-Boltzmann, finite element and finite volume methods for laminar flows. *Computers & Fluids* 2006; **35**(2006):888–897.

[121] Grader A, Mu Y, Toelke J, Baldwin C, Fang Q, Carpio G, Stenger B, Dayyani TA, Kalam Z. Estimation of relative permeability using the lattice Boltzmann method for fluid flows in a cretaceous formation. *International Petroleum Exhibition & Conference*, Md, Abu Dhabi, 2010; 1–9.

[122] Shabro V, Arns CH, Bryant SL, Torres-verdin C, Knackstedt MA. Pore scale modeling of two-phase flow. *XVIII International Conference on Water Resources*, Barcelona, 2010; 1–8.

List of Figures

List of Tables