A FINITE ELEMENT BASED LEVEL SET METHOD FOR TWO-PHASE INCOMPRESSIBLE FLOWS

SVEN GROß, VOLKER REICHELT, ARNOLD REUSKEN

Abstract. We present a method that has been developed for the efficient numerical simulation of two-phase incompressible flows. For capturing the interface between the flows the level set technique is applied. The continuous model consists of the incompressible Navier-Stokes equations coupled with an advection equation for the level set function. The effect of surface tension is modeled by a localized force term at the interface (so-called continuum surface force approach). For spatial discretization of velocity, pressure and the level set function conforming finite elements on a hierarchy of nested tetrahedral grids are used. In the finite element setting we can apply a special technique to the localized force term, which is based on a partial integration rule for the Laplace-Beltrami operator. Due to this approach the second order derivatives coming from the curvature can be eliminated. For the time discretization we apply a variant of the fractional step θ-scheme. The discrete saddle point problems that occur in each time step are solved using an inexact Uzawa method combined with multigrid techniques. For reparametrization of the level set function a new variant of the Fast Marching method is introduced. A special feature of the solver is that it combines the level set method with finite element discretization, Laplace-Beltrami partial integration, multilevel local refinement and multigrid solution techniques. All these components of the solver are described. Results of numerical experiments are presented.

AMS subject classifications. 65M60, 65T10, 76D05, 76D45, 65N22

1. Introduction. In this paper we present an overview of a method that has been developed for the efficient simulation of two-phase incompressible flows. The main characteristics of the method are the following:

- For capturing the interface between the two phases the level set method is applied [16, 28, 27].
- The spatial discretization is based on a hierarchy of tetrahedral grids. These grids are constructed in such a way that they are consistent (no hanging nodes) and that the hierarchy of triangulations is stable. The main ideas are taken from [11, 12, 6, 7, 23]. An important property is that local refinement and coarsening are easy to realize.
- For discretization of velocity, pressure and the level set function we use conforming finite elements. An example (used in the numerical experiments) is the Hood-Taylor $P_2 - P_1$ finite element pair for velocity and pressure and piecewise quadratic P_2 finite elements for the level set function.
- For the time discretization we apply a variant of the fractional step θ-scheme, due to [15].
- In each time step discrete Stokes problems and a discrete nonlinear elliptic system for the velocity unknowns must be solved. For the former we use an inexact Uzawa method with a suitable multigrid preconditioner. The latter problems are solved by a Picard iteration combined with a Krylov subspace method.
- For numerical and algorithmic purposes it is advantageous to keep the level set function close to a signed distance function during the time evolution. To realize this a reparametrization technique is needed. We apply a variant of

*Institut für Geometrie und Praktische Mathematik, RWTH Aachen, D-52056 Aachen, Germany; email: reusken@igpm.rwth-aachen.de. This work was supported by the German Research Foundation through SFB 540

the Fast Marching method [21, 34].

The above list can be extended by two model-specific points:

- The effect of surface tension is modeled by using a special localized force term at the interface. This is known as the continuum surface force (CSF) technique [13, 16].
- For the treatment of the localized force term we apply a technique based on a partial integration rule for the Laplace-Beltrami operator, cf. [2, 3, 18]. Due to this approach the second order derivatives coming from the curvature can be eliminated.

The level set technique has been successfully used in many two-phase incompressible flow simulations. By far most of these simulations use finite difference or finite volume discretization methods (cf. [28, 36] and the references therein). There are only few papers in which the level set method is combined with finite element discretization techniques. Such a combination for a 2D simulation is presented in [41, 42]. Another reference is [30]. In [2, 3] the Laplace-Beltrami partial integration technique in combination with finite elements for the treatment of curvature terms is applied to a one-phase flow problem with a free capillary surface.

We do not know of any paper in which the level set method, finite element discretization, Laplace-Beltrami partial integration, multilevel local refinement and multigrid solution techniques are all combined.

A solver based on such a combination is implemented in the DROPS package, which is under development in an interdisciplinary project (SFB 540 "Model-based Experimental Analysis of Kinetic Phenomena in Fluid Multi-phase Reactive Systems", cf. [37]) where the modeling of certain flow phenomena (e. g., mass transfer between liquid drops and a surrounding fluid or the fluid dynamics and heat transport in a laminar falling film) is a central issue. The numerical simulation of such models requires a flexible efficient and robust CFD package. Our aim is to provide such a tool.

Before going into detail we discuss some important properties of the components listed above.

Important reasons why we use the level set technique are the following. Firstly, the coupling of the Navier-Stokes equations (for the flow variables) with a scalar advection equation for the level set function allows a modular structure in which discretization and solution routines that are implemented for a one-phase flow problem can be reused both for the Navier-Stokes part and for the advection equation. Secondly, no reconstruction of the interface as with a VOF method or explicit representation of the whole interface as with front-tracking methods is needed.

Due to the nested multilevel hierarchy of tetrahedral meshes which allows simple refinement and coarsening routines we can realize *high resolution close to the interface*. For the local refinement in that area we need a marking strategy. For this the level set function is very well suited, because it yields a good approximation of the distance to the interface.

The finite element method is a flexible discretization method, which can deal with complex geometries. Due to the use of finite element discretizations on a nested multilevel grid hierarchy it is relatively easy to implement multigrid solution techniques. Hence, we use multigrid preconditioners in the inexact Uzawa method, which results in efficient solvers for discretized Stokes equations.

A further nice property of the finite element approach is that we can apply partial integration to the Laplace-Beltrami operator and thus eliminate the second order

derivatives that occur in the curvature. A disadvantage of this approach is that we have to compute the (implicitly given) interface because the partial integration has to be done on the interface. The computations, however, can be done element wise and differ from the reconstructions needed in a VOF setting.

The main reason for the choice of the time integration method for the Navier-Stokes equations is a pragmatic one. The variant of the fractional step θ-scheme that we use for the time integration is based on an operator splitting technique that decouples the incompressibility and nonlinearity in the Navier-Stokes equations and yields two Stokes type of problems and a nonlinear elliptic system for the velocity unknowns. For the Stokes subproblems we have efficient solvers available, cf. [29]. The problem for the velocity unknowns can be solved using a standard Krylov subspace method. The fractional step θ-scheme makes it possible to reuse this already existing software. In the near future the Stokes solver will be replaced by a Navier-Stokes solver. Then other variants of the fractional step θ-scheme can be used in which the decoupling of incompressibility and nonlinearity is avoided. This modification, which does not change the structure of the outer solver, will probably improve efficiency.

Related to the reparametrization method we note that we investigated techniques that are based on an evolution equation, whose limit solution fulfills the Eikonal equation $\|\nabla\psi\| = 1$. However, the Fast Marching approach that aims at solving the Eikonal equation directly turned out to better conserve the location of the interface.

The paper is organized as follows. In Section 2 we formulate the continuous two-phase problem. The CSF technique and the coupling with the level set function are described. In Section 3 we discuss the discretization methods that are used. A brief description of the tetrahedral refinement method is given. The finite element discretization of the Navier-Stokes equation is presented. The treatment of the curvature localized force term, which is based on partial integration of the Laplace-Beltrami operator, is explained. We also discuss how we deal with the discontinuities in the viscosity and density coefficients. In Section 4 we describe the time integration method. In Section 5 we explain the solvers that are used for the systems that occur in the time integration method. In Section 6 the reparametrization method for the level set function and other algorithms to preserve the quality of the level set function are discussed. Finally, in Section 7 we present results of numerical experiments.

2. A continuous model for two-phase flows. We consider a domain $\Omega \subset \mathbb{R}^3$ which contains two different immiscible incompressible newtonian phases (fluid-fluid or fluid-gas). The model problem is a liquid drop contained in a surrounding fluid. The time-dependent domains which contain the phases are denoted by $\Omega_1 = \Omega_1(t)$ and $\Omega_2 = \Omega_2(t)$ with $\overline{\Omega}_1 \cup \overline{\Omega}_2 = \overline{\Omega}$. We assume $\partial\Omega_1 \cap \partial\Omega = \emptyset$. The interface between the two phases ($\partial\Omega_1 \cap \partial\Omega_2$) is denoted by $\Gamma = \Gamma(t)$. To model the forces at the interface we make the standard assumption that the surface tension balances the jump of the normal stress on the interface, i. e., we have a free boundary condition

$$[\boldsymbol{\sigma}\mathbf{n}]_\Gamma = \tau\kappa\mathbf{n} \ ,$$

with $\mathbf{n} = \mathbf{n}_\Gamma$ the unit normal at the interface (pointing from Ω_1 in Ω_2), τ the surface tension coefficient (material parameter), κ the curvature of Γ and $\boldsymbol{\sigma}$ the stress tensor, i. e.,

$$\boldsymbol{\sigma} = -p\mathbf{I} + \mu\mathbf{D}(\mathbf{u}), \quad \mathbf{D}(\mathbf{u}) = \nabla\mathbf{u} + (\nabla\mathbf{u})^T \ ,$$

with $p = p(x,t)$ the pressure, $\mathbf{u} = \mathbf{u}(x,t)$ the velocity vector and μ the viscosity. We assume continuity of the velocity across the interface. In combination with the conservation laws of mass and momentum this yields the following standard model:

$$\begin{cases} \rho_i\big(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\big) = -\nabla p + \rho_i \mathbf{g} + \operatorname{div}(\mu_i \mathbf{D}(\mathbf{u})) & \text{in } \Omega_i \\ \operatorname{div}\mathbf{u} = 0 & \text{in } \Omega_i \end{cases} \quad \text{for } i = 1,2 \tag{2.1}$$

$$[\boldsymbol{\sigma}\mathbf{n}]_\Gamma = \tau\kappa\mathbf{n}, \quad [\mathbf{u}\cdot\mathbf{n}]_\Gamma = 0\ . \tag{2.2}$$

The vector $\mathbf{g}$ is a known external force (gravity). In addition we need initial conditions for $\mathbf{u}(x,0)$ and boundary conditions at $\partial\Omega$. For simplicity we assume homogeneous Dirichlet boundary conditions. In our experiments we considered other (more realistic) boundary conditions.

This model for a two-phase incompressible flow problem is often used in the literature. The effect of the surface tension can be expressed in terms of a localized force at the interface, cf. the so-called continuum surface force (CSF) model [13, 16]. This localized force is given by

$$f_\Gamma = \tau\kappa\delta_\Gamma \mathbf{n}_\Gamma\ .$$

Here δ_Γ is a Dirac δ-function with support on Γ. Its action on a smooth test function ψ is given by

$$\int_\Omega \delta_\Gamma(x)\psi(x)\,dx = \int_\Gamma \psi(s)\,ds\ .$$

This localization approach can be combined with the level set method for capturing the unknown interface. We outline the main idea, for a detailed treatment we refer to the literature [16]. The level set function, denoted by $\phi = \phi(x,t)$ is a scalar function. At the initial time $t = 0$ we assume a function $\phi_0(x)$ such that $\phi_0(x) < 0$ for $x \in \Omega_1(0)$, $\phi_0(x) > 0$ for $x \in \Omega_2(0)$, $\phi_0(x) = 0$ for $x \in \Gamma(0)$. It is desirable to have the level set function as an approximate signed distance function.

For the evolution of the interface we consider the trace $x(t)$ of a single particle $x(0) \in \Omega$ over time. A particle on the interface remains on the interface for all time, i. e., for all $x(0) \in \Gamma(0)$ and all $t \geq 0$ we have $x(t) \in \Gamma(t)$. This is equivalent to the condition $\phi(x(t),t) = 0$ $(t \geq 0)$ which we extend to the whole domain as $\phi(x(t),t) = \phi(x(0),0)$ for all $x(0) \in \Omega$ and all $t \geq 0$. By differentiating this condition with respect to t we obtain $\phi_t + \nabla\phi(x,t)\cdot x_t = 0$. The displacement of a particle coincides with the velocity field, i. e., $x_t = \mathbf{u}$ holds. Hence one obtains the first order differential equation $\phi_t + \mathbf{u}\cdot\nabla\phi = 0$ for $t \geq 0$ and $x \in \Omega$.

The jumps in the coefficients ρ and μ can be described using the level set function (which has its zero level set precisely at the interface Γ) in combination with the Heaviside function $H : \mathbb{R} \to \mathbb{R}$:

$$H(\zeta) = 0 \quad \text{for } \zeta < 0\ , \qquad H(\zeta) = 1 \quad \text{for } \zeta > 0\ .$$

For ease one can take $H(0) = \frac{1}{2}$. We define

$$\begin{aligned} \rho(\phi) &:= \rho_1 + (\rho_2 - \rho_1)H(\phi) \\ \mu(\phi) &:= \mu_1 + (\mu_2 - \mu_1)H(\phi) \end{aligned} \tag{2.3}$$

Combination of the CSF approach with the level set method leads to the following model for the two-phase problem in $\Omega \times [0,T]$

$$\rho(\phi)\Big(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\Big) = -\nabla p + \rho(\phi)\mathbf{g} + \operatorname{div}(\mu(\phi)\mathbf{D}(\mathbf{u})) + \tau\kappa\delta_\Gamma \mathbf{n}_\Gamma \tag{2.4}$$

$$\operatorname{div}\mathbf{u} = 0 \tag{2.5}$$

$$\phi_t + \mathbf{u}\cdot\nabla\phi = 0 \tag{2.6}$$

together with suitable initial and boundary conditions for $\mathbf{u}$ and ϕ. This is the continuous problem that we use to model our two-phase problem. It is also used in, for example, [16, 28, 30, 39, 40, 41, 42].

3. Discretization methods. In this section we discuss techniques that are used for the discretization of the continuous model (2.4)-(2.6).

3.1. Multilevel tetrahedral grid hierarchy. We outline the basic ideas of the multilevel grid hierarchy on which our finite element discretization method is based. We only consider multilevel *tetrahedral* meshes based on *red/green refinement strategies* (cf. [4, 6, 10]). The idea of a *multilevel* refinement (and coarsening) strategy was introduced in [6] and further developed in [8, 10, 12, 23, 24, 45]. This grid refinement technique is used in UG [44]; for an overview we refer to [7, 9]. Similar techniques are used in several other packages, for example, in KASKADE [20], PML/MG [38].

We first introduce a few basic notions. We assume that Ω is a polyhedral domain.

DEFINITION 1 (Triangulation). A finite collection $\mathcal{T}$ of tetrahedra $T \subset \overline{\Omega}$ is called a *triangulation* of Ω (or $\overline{\Omega}$) if the following holds:

1. $\operatorname{vol}(T) > 0$ for all $T \in \mathcal{T}$,
2. $\bigcup_{T\in\mathcal{T}} T = \overline{\Omega}$,
3. $\operatorname{int}(S) \cap \operatorname{int}(T) = \emptyset$ for all $S, T \in \mathcal{T}$ with $S \neq T$.

DEFINITION 2 (Consistency). A triangulation $\mathcal{T}$ is called *consistent* if the intersection of any two tetrahedra in $\mathcal{T}$ is either empty, a common face, a common edge or a common vertex.

DEFINITION 3 (Stability). A sequence of triangulations $\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \ldots$ is called *stable* if all angles of all tetrahedra in this sequence are uniformly bounded away from zero.

DEFINITION 4 (Refinement). For a given tetrahedron T a triangulation $\mathcal{K}(T)$ of T is called a *refinement* of T if $|\mathcal{K}(T)| \geq 2$ and any vertex of any tetrahedron $T' \in \mathcal{K}(T)$ is either a vertex or an edge midpoint of T. In this case T' is called a child of T and T is called the parent of T'. A triangulation $\mathcal{T}_{k+1}$ is called *refinement* of a triangulation $\mathcal{T}_k \neq \mathcal{T}_{k+1}$ if for every $T \in \mathcal{T}_k$ either $T \in \mathcal{T}_{k+1}$ or $\mathcal{K}(T) \subset \mathcal{T}_{k+1}$ for some refinement $\mathcal{K}(T)$ of T.

DEFINITION 5 (Multilevel triangulation). A sequence of consistent triangulations $\mathcal{M} = (\mathcal{T}_0, \ldots, \mathcal{T}_J)$ is called a *multilevel triangulation* of Ω if the following holds:

1. For $0 \leq k < J$: $\mathcal{T}_{k+1}$ is a refinement of $\mathcal{T}_k$.
2. For $0 \leq k < J$: $T \in \mathcal{T}_k \cap \mathcal{T}_{k+1} \Rightarrow T \in \mathcal{T}_J$.

DEFINITION 6 (Hierarchical decomposition). Let $\mathcal{M} = (\mathcal{T}_0, \ldots, \mathcal{T}_J)$ be a multilevel triangulation of Ω. For every tetrahedron $T \in \mathcal{M}$ a unique level number $\ell(T)$ is defined by $\ell(T) := \min\{\, k \mid T \in \mathcal{T}_k \,\}$. The set $\mathcal{G}_k \subset \mathcal{T}_k$

$$\mathcal{G}_k := \{\, T \in \mathcal{T}_k \mid \ell(T) = k \,\}$$

is called the *hierarchical surplus* on level k ($k = 0, \ldots, J$). Note that $\mathcal{G}_0 = \mathcal{T}_0$, $\mathcal{G}_k = \mathcal{T}_k \setminus \mathcal{T}_{k-1}$ for $k = 1, \ldots, J$. The sequence $\mathcal{H} = (\mathcal{G}_0, \ldots, \mathcal{G}_J)$ is called the *hierarchical*

decomposition of $\mathcal{M}$. Note that the multilevel triangulation $\mathcal{M}$ can be reconstructed from its hierarchical decomposition.

REMARK 1. Let $\mathcal{M}$ be a multilevel triangulation and V_k $(0 \leq k \leq J)$ be the corresponding finite element spaces of continuous functions $p \in C(\overline{\Omega})$ such that $p_{|T} \in P_q$ for all $T \in \mathcal{T}_k$ $(q \geq 1)$. The refinement property 1 in definition 5 implies nestedness of these finite element spaces: $V_k \subset V_{k+1}$.

Now assume that based on some error indicator certain tetrahedra in the finest triangulation $\mathcal{T}_J$ are marked for refinement. In many refinement algorithms one then modifies the finest triangulation $\mathcal{T}_J$ resulting in a new one, $\mathcal{T}_{J+1}$. Using such a strategy (which we call a *one*-level method) the new sequence $(\mathcal{T}_0, \ldots, \mathcal{T}_{J+1})$ is in general not a multilevel triangulation because the nestedness property 1 in definition 5 does not hold. We also note that when using such a method it is difficult to implement a reasonable coarsening strategy. In *multilevel* refinement algorithms the whole sequence $\mathcal{M}$ is used and as output one obtains a sequence $\mathcal{M}' = (\mathcal{T}'_0, \ldots, \mathcal{T}'_{J'})$, with $\mathcal{T}'_0 = \mathcal{T}_0$ and $J' \in \{J-1, J, J+1\}$. In general one has $\mathcal{T}'_k \neq \mathcal{T}_k$ for $k > 0$. We list a few important properties of this method:

- Both the input and output are *multilevel triangulations*.
- The method yields *stable* and *consistent* triangulations.
- Local *refinement* and *coarsening* are treated in a similar way.
- The implementation uses only the hierarchical decomposition of $\mathcal{M}$. This allows *relatively simple data structures* without storage overhead.
- The costs are proportional to the number of tetrahedra in $\mathcal{T}_J$.

For a detailed discussion of these and other properties we refer to the literature ([6, 10, 23, 19]). In our implementation we use the multilevel refinement algorithm described in [19].

3.2. Finite element discretization. In this section we briefly discuss the discretization of the equations (2.4)-(2.6) using standard (LBB stable) finite element spaces. The discretization of the nonstandard localized force term that occurs in (2.4) will be explained in Section 3.3.

In the standard weak formulation of the Navier-Stokes equations (with homogeneous Dirichlet boundary conditions for the velocity) one uses the spaces

$$\mathbf{V} := H_0^1(\Omega)^3, \quad Q := L_0^2(\Omega) = \{\, q \in L^2(\Omega) \mid \int_\Omega q = 0 \,\}$$

We will use the notation $V := H^1(\Omega)$ and introduce the bilinear forms

$$\begin{aligned} m : L^2(\Omega)^3 \times L^2(\Omega)^3 \to \mathbb{R} \ : \quad & m(\mathbf{u}, \mathbf{v}) = \int_\Omega \rho(\phi)\, \mathbf{u} \cdot \mathbf{v}\, dx \\ a : \mathbf{V} \times \mathbf{V} \to \mathbb{R} \ : \quad & a(\mathbf{u}, \mathbf{v}) = \frac{1}{2} \int_\Omega \mu(\phi)\ \mathrm{tr}\left(\mathbf{D}(\mathbf{u})\mathbf{D}(\mathbf{v})\right)\, dx \\ b : \mathbf{V} \times Q \to \mathbb{R} \ : \quad & b(\mathbf{u}, q) = \int_\Omega q\, \mathrm{div}\, \mathbf{u}\, dx \end{aligned}$$

and the trilinear form

$$c : \mathbf{V} \times \mathbf{V} \times \mathbf{V} \to \mathbb{R} \ : \quad c(\mathbf{u}; \mathbf{v}, \mathbf{w}) = \int_\Omega \rho(\phi)\, (\mathbf{u} \cdot \nabla \mathbf{v}) \cdot \mathbf{w}\, dx$$

Note, that the bilinear forms a and m as well as the trilinear form c depend on ϕ. For

the L^2 scalar product we use the notation

$$(f,g)_0 := \int_\Omega fg\,dx \qquad \text{for } \; f,g \in L^2(\Omega)$$

Now testing the equations (2.4)-(2.6) with $\mathbf{v} \in \mathbf{V}$, $q \in Q$, and $v \in V$ we get

$$\begin{aligned} m(\mathbf{u}_t(t),\mathbf{v}) + a(\mathbf{u}(t),\mathbf{v}) + c(\mathbf{u}(t);\mathbf{u}(t),\mathbf{v}) - b(\mathbf{v},p(t)) &= m(\mathbf{g},\mathbf{v}) + f_\Gamma(\mathbf{v}) \\ b(\mathbf{u}(t),q) &= 0 \\ (\phi_t(t),v)_0 + (\mathbf{u}(t)\cdot\nabla\phi(t),v)_0 &= 0 \end{aligned}$$

with

$$f_\Gamma(\mathbf{v}) = \tau \int_\Omega \kappa\delta_\Gamma \mathbf{n}_\Gamma \cdot \mathbf{v}\,dx = \tau \int_\Gamma \kappa \mathbf{n}_\Gamma \cdot \mathbf{v}\,ds$$

which is Γ-dependent and therefore also ϕ-dependent.

Let $\mathcal{M} = (\mathcal{T}_0,\ldots,\mathcal{T}_J)$ be a multilevel triangulation of Ω. With each triangulation $\mathcal{T}_k$ $(0 \le k \le J)$ we associate a mesh size parameter $h = h_k$. Let $\mathbf{V}_h \subset \mathbf{V}$, $Q_h \subset Q$ and $V_h \subset V$ be standard polynomial finite element spaces corresponding to the triangulation $\mathcal{T}_k$. We assume the pair $(\mathbf{V}_h, Q_h)$ to be LBB stable. In our numerical experiments we use the Hood-Taylor $P_2 - P_1$ pair.

The finite element discretization leads to the following variational problem: Find $\mathbf{u}_h(t) \in \mathbf{V}_h$, $p_h(t) \in Q_h$ and $\phi_h(t) \in V_h$ such that for $t \in [0,T]$:

$$\begin{aligned} m((\mathbf{u}_h)_t(t),\mathbf{v}_h) + a(\mathbf{u}_h(t),\mathbf{v}_h) + c(\mathbf{u}_h(t);\mathbf{u}_h(t),\mathbf{v}_h) - b(\mathbf{v}_h,p_h(t)) & \\ = m(\mathbf{g},\mathbf{v}_h) + f_{\Gamma_h}(\mathbf{v}_h) \qquad & \forall \mathbf{v}_h \in \mathbf{V}_h \quad (3.1) \\ b(\mathbf{u}_h(t),q_h) = 0 \qquad & \forall q_h \in Q_h \quad (3.2) \\ ((\phi_h)_t(t),v_h)_0 + (\mathbf{u}_h(t)\cdot\nabla\phi_h(t),v_h)_0 = 0 \qquad & \forall v_h \in V_h \quad (3.3) \end{aligned}$$

with a, m, and c now depending on $\phi_h(t)$. The term $f_{\Gamma_h}(\mathbf{v}_h)$ is an approximation of $f_\Gamma(\mathbf{v}_h)$ (which is discussed in detail in Section 3.3) with Γ_h being a polyhedral approximation of Γ. In addition we have initial conditions $\mathbf{u}_h(0)$ and $\phi_h(0)$.

Since a, m, c, and f_{Γ_h} depend on the discrete level set function $\phi_h(t)$ the system (3.1)-(3.3) forms a strongly coupled system of ordinary differential equations for the unknowns $\mathbf{u}_h(t)$, $p_h(t)$ and $\phi_h(t)$.

The discretization (3.3) of the hyperbolic level set equation (2.6) is not stable. It can be stabilized using a streamline diffusion technique. This streamline diffusion stabilization applied to the level set equation can be seen as a Petrov-Galerkin method, with trial space V_h and test functions $\hat{v}_h$. For each tetrahedron $T \in \mathcal{T}_k$ a stabilization parameter δ_T is chosen. The test functions are then defined as

$$\hat{v}_h(x) = v_h(x) + \delta_T \mathbf{u}_h(x,t)\cdot\nabla v_h(x)$$

for $x \in T$. For an analysis of the streamline diffusion method and reasonable choices for the stabilization parameter δ_T we refer to [33]. This leads to the following stabilized variant of (3.3):

$$\sum_{T\in\mathcal{T}_k} ((\phi_h)_t(t) + \mathbf{u}_h(t)\cdot\nabla\phi_h(t), v_h + \delta_T \mathbf{u}_h(t)\cdot\nabla v_h)_{0,T} = 0 \qquad \forall v_h \in V_h \qquad (3.4)$$

We now derive a representation of (3.1), (3.2), (3.4), using bases of the finite element spaces $\mathbf{V}_h$, Q_h and V_h. Let $\{\boldsymbol{\xi}_i\}_{1\le i\le N}$, $\{\psi_i\}_{1\le i\le K}$ and $\{\chi_i\}_{1\le i\le L}$ be the standard

nodal bases of the finite element spaces $\mathbf{V}_h$, Q_h and V_h, respectively. For $\phi_h \in V_h$ and $\mathbf{u}_h \in \mathbf{V}_h$ we introduce the following (mass and stiffness) matrices:

$$\begin{aligned}
\mathbf{M}(\phi_h) \in \mathbb{R}^{N\times N}, &\quad \mathbf{M}(\phi_h)_{ij} = \int_\Omega \rho(\phi_h)\boldsymbol{\xi}_i \cdot \boldsymbol{\xi}_j \; dx \\
\mathbf{A}(\phi_h) \in \mathbb{R}^{N\times N}, &\quad \mathbf{A}(\phi_h)_{ij} = \frac{1}{2}\int_\Omega \mu(\phi_h)\,\mathrm{tr}\left(\mathbf{D}(\boldsymbol{\xi}_i)\mathbf{D}(\boldsymbol{\xi}_j)\right) \; dx \\
\mathbf{B} \in \mathbb{R}^{K\times N}, &\quad \mathbf{B}_{ij} = -\int_\Omega \psi_i \,\mathrm{div}\,\boldsymbol{\xi}_j \; dx \\
\mathbf{C}(\phi_h,\mathbf{u}_h) \in \mathbb{R}^{N\times N}, &\quad \mathbf{C}(\phi_h,\mathbf{u}_h)_{ij} = \int_\Omega \rho(\phi_h)(\mathbf{u}_h\cdot\nabla\boldsymbol{\xi}_j)\cdot\boldsymbol{\xi}_i \; dx \\
\mathbf{E}(\mathbf{u}_h) \in \mathbb{R}^{L\times L}, &\quad \mathbf{E}(\mathbf{u}_h)_{ij} = \sum_{T\in\mathcal{T}_k}\int_T \chi_j(\chi_i + \delta_T\mathbf{u}_h\cdot\nabla\chi_i) \; dx \\
\mathbf{H}(\mathbf{u}_h) \in \mathbb{R}^{L\times L}, &\quad \mathbf{H}(\mathbf{u}_h)_{ij} = \sum_{T\in\mathcal{T}_k}\int_T (\mathbf{u}_h\cdot\nabla\chi_j)(\chi_i + \delta_T\mathbf{u}_h\cdot\nabla\chi_i) \; dx
\end{aligned}$$

We also need the following vectors:

$$\begin{aligned}
\vec{\mathbf{g}}(\phi_h) \in \mathbb{R}^N, &\quad \vec{\mathbf{g}}(\phi_h)_i = \int_\Omega \rho(\phi_h)\mathbf{g}\cdot\boldsymbol{\xi}_i \; dx \\
\vec{\mathbf{f}}_{\Gamma_h}(\phi_h) \in \mathbb{R}^N, &\quad \vec{\mathbf{f}}_{\Gamma_h}(\phi_h)_i = f_{\Gamma_h}(\boldsymbol{\xi}_i)
\end{aligned}$$

We now represent $\mathbf{u}_h(t) \in \mathbf{V}_h$, $p_h(t) \in Q_h$ and $\phi_h(t) \in V_h$ as follows:

$$\begin{aligned}
\mathbf{u}_h(t) &= \sum_{j=1}^N u_j(t)\boldsymbol{\xi}_j, \quad \vec{\mathbf{u}}(t) := (u_1(t),\ldots,u_N(t)) \\
p_h(t) &= \sum_{j=1}^K p_j(t)\psi_j, \quad \vec{\mathbf{p}}(t) := (p_1(t),\ldots,p_K(t)) \\
\phi_h(t) &= \sum_{j=1}^L \phi_j(t)\chi_j, \quad \vec{\boldsymbol{\phi}}(t) := (\phi_1(t),\ldots,\phi_L(t))
\end{aligned}$$

Using this notation we obtain the following equivalent formulation of the coupled system of ordinary differential equations (3.1), (3.2), (3.4): Find $\vec{\mathbf{u}}(t) \in \mathbb{R}^N$, $\vec{\mathbf{p}}(t) \in \mathbb{R}^K$ and $\vec{\boldsymbol{\phi}}(t) \in \mathbb{R}^L$ such that for all $t \in [0,T]$

$$\begin{aligned}
\mathbf{M}(\vec{\boldsymbol{\phi}}(t))\frac{d\vec{\mathbf{u}}}{dt}(t) + \mathbf{A}(\vec{\boldsymbol{\phi}}(t))\vec{\mathbf{u}}(t) + \mathbf{C}(\vec{\boldsymbol{\phi}}(t),\vec{\mathbf{u}}(t))\vec{\mathbf{u}}(t) + \mathbf{B}^T\vec{\mathbf{p}}(t) & \\
= \vec{\mathbf{g}}(\vec{\boldsymbol{\phi}}(t)) + \vec{\mathbf{f}}_{\Gamma_h}(\vec{\boldsymbol{\phi}}(t)) & \quad (3.5) \\
\mathbf{B}\vec{\mathbf{u}}(t) = 0 & \quad (3.6) \\
\mathbf{E}(\vec{\mathbf{u}}(t))\frac{d\vec{\boldsymbol{\phi}}}{dt}(t) + \mathbf{H}(\vec{\mathbf{u}}(t))\vec{\boldsymbol{\phi}}(t) = 0 & \quad (3.7)
\end{aligned}$$

3.3. Discretization of the curvature localized force term. In this section we explain how the approximation of the localized curvature force term, $f_{\Gamma_h}(\mathbf{v}_h)$ in (3.1), is constructed. We use the technique presented in [3, 18]. For this we first need

some notions from differential geometry. We assume that Γ is a closed smooth (C^2) surface in $\mathbb{R}^3$.

Let $\omega : D \to \mathbb{R}^3$, with $D \subset \mathbb{R}^2$, be a local parametrization of Γ. We use the notation: $z = (z_1, z_2) \in D$, $x = (x_1, x_2, x_3) \in \mathbb{R}^3$. The Jacobi-matrix of ω and the metric tensor at $z \in D$ are denoted by

$$J_\omega(z) = \left(\frac{\partial \omega_i(z)}{\partial z_j} \right) \in \mathbb{R}^{3\times 2}, \quad G_\omega(z) := J_\omega(z)^T J_\omega(z) \in \mathbb{R}^{2\times 2}$$

Furthermore, we define

$$P_\omega(z) := J_\omega(z) G_\omega(z)^{-1} J_\omega(z)^T \in \mathbb{R}^{3\times 3}$$

Note that $P_\omega^2 = P_\omega = P_\omega^T$ and thus $P_\omega(z)$ is an orthogonal projection on the span of the columns of $J_\omega(z)$ which is the tangent space of Γ at $x = \omega(z)$.

Suppose $f : \Gamma \to \mathbb{R}$ is sufficiently smooth. The tangential derivative of f is defined by projecting the derivate to the tangent space of Γ, i. e.

$$\underline{\nabla} f(x) := P_\omega(z) \nabla f(x) \tag{3.8}$$

An alternative form is

$$\underline{\nabla} f = \nabla f - \nabla f \cdot \mathbf{n}_\Gamma \, \mathbf{n}_\Gamma$$

where $\mathbf{n}_\Gamma$ denotes the unit normal of Γ.

If f has second derivatives we define the *Laplace-Beltrami operator* of f on Γ by

$$\underline{\Delta} f := \underline{\nabla} \cdot \underline{\nabla} f$$

For vector valued functions f, $g : \Gamma \to \mathbb{R}^3$ we define

$$\underline{\Delta} f := (\underline{\Delta} f_1, \underline{\Delta} f_2, \underline{\Delta} f_3)^T, \qquad \underline{\nabla} f \cdot \underline{\nabla} g := \sum_{i=1}^{3} \underline{\nabla} f_i \cdot \underline{\nabla} g_i$$

The following basic result from differential geometry (cf., for example, Lemma 2.1 in [17]) shows how curvature can be related to the Laplace-Beltrami operator and how partial integration can be applied.

THEOREM 3.1. *Let* $\mathrm{id}_\Gamma : \Gamma \to \mathbb{R}^3$ *be the identity on* Γ, $\kappa = \kappa_1 + \kappa_2$ *the sum of the principal curvatures, and* $\mathbf{n}_\Gamma$ *the outward unit normal on* Γ. *Then for all sufficiently smooth vector functions* $\mathbf{v}$ *on* Γ *the following holds:*

$$\int_\Gamma \kappa \, \mathbf{n}_\Gamma \cdot \mathbf{v} \, ds = -\int_\Gamma (\underline{\Delta} \, \mathrm{id}_\Gamma) \cdot \mathbf{v} \, ds = \int_\Gamma \underline{\nabla} \, \mathrm{id}_\Gamma \cdot \underline{\nabla} \, \mathbf{v} \, ds$$

We show how this result is used to construct an approximation $f_{\Gamma_h}(\mathbf{v}_h)$ of the curvature localized force term in the important case (which we also use in our experiments) that *the finite element space* V_h *for the level set function consists of piecewise quadratics.*

For ease of notation, we choose a fixed t and suppress the time-dependence throughout the rest of this section. Let $\phi_h \in V_h$ be a given approximation of the level set function ϕ. Recall that $\Gamma = \{ x \in \Omega \mid \phi(x) = 0 \}$. First we explain how an approximation Γ_h of Γ is constructed. Let $\mathcal{K}(T)$ be the set of 8 children of T

that results from a regular refinement of a tetrahedron $T \in \mathcal{T}_k$. Let $I(\phi_h)$ be the continuous *piecewise linear* function which interpolates ϕ_h at all four vertices ν of T', for all $T' \in \mathcal{K}(T)$, i. e.:

$$\forall\, T \in \mathcal{T}_k,\ \ \forall\, T' \in \mathcal{K}(T),\ \ \forall \text{ vertices } \nu \text{ of } T' : \quad I(\phi_h)(\nu) = \phi_h(\nu)$$

The approximation of the interface is defined by

$$\Gamma_h := \{\, x \in \Omega \mid I(\phi_h)(x) = 0 \,\} \tag{3.9}$$

Note that Γ_h depends on the given approximation $\phi_h \in V_h$ and that Γ_h consists of planar segments. Hence, Γ_h is a $C^{0,1}$ embedding of a manifold in $\mathbb{R}^3$. We now describe how Γ_h is used to compute an approximation $f_{\Gamma_h}(\mathbf{v}_h)$ of the curvature localized force term. There is a collection of child tetrahedra $\mathcal{I} \subset \{\, T' \in \mathcal{K}(T) \mid T \in \mathcal{T}_k \,\}$ such that

$$\begin{cases} \Gamma_{T'} := \Gamma_h \cap T' \quad \text{is a plane segment, for all } T' \in \mathcal{I}, \\ \text{and} \ \ \Gamma_h = \cup_{T' \in \mathcal{I}} \Gamma_{T'} \end{cases} \tag{3.10}$$

This collection of children $\mathcal{I}$ can easily be determined by looking at the signs of $\phi_h(\nu)$ for all vertices ν of each T'. We apply the partial integration described in theorem 3.1 and use Γ_h as an approximation for Γ. Thus for a function $\mathbf{v} \in \mathbf{V}_h$ we get

$$\begin{aligned} f_\Gamma(\mathbf{v}) &= \tau \int_\Gamma \kappa \mathbf{n}_\Gamma \cdot \mathbf{v}\, ds = \tau \int_\Gamma \underline{\nabla}\, \mathrm{id}_\Gamma \cdot \underline{\nabla}\, \mathbf{v}\, ds \\ &\approx \tau \int_{\Gamma_h} \underline{\nabla}\, (\mathrm{id}_{\Gamma_h}) \cdot \underline{\nabla}\, \mathbf{v}\, ds = \tau \sum_{T' \in \mathcal{I}} \int_{\Gamma_{T'}} \underline{\nabla}\, (\mathrm{id}_{\Gamma_{T'}}) \cdot \underline{\nabla}\, \mathbf{v}\, ds \\ &= \tau \sum_{T' \in \mathcal{I}} \sum_{i=1}^{3} \int_{\Gamma_{T'}} \underline{\nabla}\, (\mathrm{id}_{\Gamma_{T'}})_i \cdot \underline{\nabla}\, \mathbf{v}_i\, ds =: f_{\Gamma_h}(\mathbf{v}) \end{aligned} \tag{3.11}$$

Because $\mathbf{v}_i$ lies in V_h we have to evaluate integrals of the form

$$\int_{\Gamma_{T'}} \underline{\nabla}\, (\mathrm{id}_{\Gamma_{T'}})_i \cdot \underline{\nabla}\, v\, ds \tag{3.12}$$

with $\Gamma_{T'} = \Gamma_h \cap T'$ a plane segment and $v \in V_h$ a piecewise quadratic function. We derive simple explicit formulas for these integrals. One easily verifies that $\Gamma_{T'}$ is either a triangle (=: case 1) or a quadrilateral (=: case 2). Let $\mathcal{Q}$ be the (possibly degenerated) reference quadrilateral with vertices $(0,0)$, $(1,0)$, $(0,1)$, (a,b), $a > 0, b > 0$, and let $P \in \mathbb{R}^3$, $A \in \mathbb{R}^{3\times 2}$ be such that the affine mapping $\omega(z) = Az + P$ is a parametrization of $\Gamma_{T'}$, i. e., $\omega(\mathcal{Q}) = \Gamma_{T'}$. In case 1 we take $(a,b) = (\frac{1}{2}, \frac{1}{2})$ and in case 2 we have $a + b \neq 1$. The vertices of $\Gamma_{T'}$ are denoted by

$$P = \omega \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ Q := \omega \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ R := \omega \begin{pmatrix} 0 \\ 1 \end{pmatrix},\ S := \omega \begin{pmatrix} a \\ b \end{pmatrix}$$

The following elementary result holds.

LEMMA 3.2. *For $f \in P_1$ we have*

$$\int_{\Gamma_{T'}} f(s)\, ds = \frac{1}{6} \sqrt{\det A^T A} \Big[f(P) + f(Q) + f(R) + (a+b-1)\big(f(S) + f(Q) + f(R)\big) \Big]$$

Proof. Apply the integral transformation formula

$$\int_{\Gamma_{T'}} f(s)\,ds = \sqrt{\det A^T A}\int_{\mathcal{Q}} f(\omega(z))\,dz$$

and use the following exact quadrature rule for a linear function $\ell : \mathcal{Q} \to \mathbb{R}$

$$\int_{\mathcal{Q}} \ell(z)\,dz = \frac{1}{6}\big(\ell(0,0)+\ell(1,0)+\ell(0,1)\big) + \frac{1}{6}(a+b-1)\big(\ell(a,b)+\ell(1,0)+\ell(0,1)\big)$$

□

We then obtain simple explicit formulas for computing the integrals in (3.12):

THEOREM 3.3. *Define $B := A(A^TA)^{-1}A^T$ and let e_i be the i-th basis vector in $\mathbb{R}^3$. For $v \in P_2$ the following identity holds:*

$$\int_{\Gamma_{T'}} \underline{\nabla}\,(\mathrm{id}_{\Gamma_{T'}})_i \cdot \underline{\nabla}\,v\,ds = \frac{1}{6}\sqrt{\det A^TA}\,(Be_i)^T\Big[\nabla v(P)+\nabla v(Q)+\nabla v(R) + (a+b-1)\big(\nabla v(S)+\nabla v(Q)+\nabla v(R)\big)\Big]$$

Proof. From (3.8) we get

$$\underline{\nabla}\,(\mathrm{id}_{\Gamma_{T'}})_i(x) = \underline{\nabla}\,x_i = A(A^TA)^{-1}A^T\nabla x_i = Be_i$$
$$\underline{\nabla}\,v(x) = A(A^TA)^{-1}A^T\nabla v(x) = B\nabla v(x)$$

Since $B = B^T$ and $BB = B$ this yields

$$\underline{\nabla}\,(\mathrm{id}_{\Gamma_{T'}})_i(x) \cdot \underline{\nabla}\,v(x) = (Be_i)\cdot(B\nabla v(x)) = (Be_i)^T\nabla v(x)$$

Finally note that $x \to (Be_i)^T\nabla v(x)$ is linear and apply lemma 3.2. □

Summarizing, given a piecewise quadratic approximation ϕ_h of the level set function ϕ and a piecewise quadratic nodal basis function $\boldsymbol{\xi} = (\xi_1,\xi_2,\xi_3) \in \mathbf{V}_h$ we have the following procedure for computing $f_{\Gamma_h}(\boldsymbol{\xi}) \approx f_\Gamma(\boldsymbol{\xi})$:

1. Determine the piecewise planar interface Γ_h — which is the 0-level set of the linear interpolation $I(\phi_h)$ — and the associated collection of child tetrahedra $\mathcal{I}$ (as in (3.9), (3.10)).
2. For $i = 1,2,3$ and $T' \in \mathcal{I}$ with $T' \in \mathcal{K}(T)$, $T \subset \mathrm{supp}(\xi_i)$ determine

$$\int_{\Gamma_{T'}} \underline{\nabla}\,(\mathrm{id}_{\Gamma_{T'}})_i \cdot \underline{\nabla}\,\xi_i\,ds$$

   using the exact quadrature rule from theorem 3.3 and add these contributions as described in (3.11).

Note that P, Q, R, S lie on edges of T' and thus the values of the *linear* function $\nabla\xi_i$ at these points can easily be obtained from the values of $\nabla\xi_i$ at the vertices of T.

3.4. Treatment of discontinuities in density and viscosity. In the discrete variational formulation integrals of the form

$$I_h := \int_\Omega \sigma(\phi_h(x))G(x)\,dx$$

with a discontinuous piecewise constant function $\sigma(\phi_h)$ as in (2.3) and a continuous (piecewise) smooth function G have to be approximated. Note that the continuous

smooth level set Γ (0-level of ϕ) has been approximated by a piecewise planar approximation Γ_h (0-level of ϕ_h). For $G(x) = 1$ this yields an error

$$\left| \int_\Omega \sigma(\phi_h(x))\,dx - \int_\Omega \sigma(\phi(x))\,dx \right| = \mathcal{O}(h^2)$$

Hence, in general it does not make sense to compute I_h with very high accuracy. For the approximation of I_h there are two obvious approaches. Firstly, one can determine the discrete interface Γ_h (as explained in the previous section) and split the integration in an integration over two subdomains where $\sigma(\phi_h)$ is constant:

$$I_h = \int_\Omega \sigma(\phi_h(x))G(x)\,dx = \sigma_1 \int_{\Omega_{h,1}} G(x)\,dx + \sigma_2 \int_{\Omega_{h,2}} G(x)\,dx$$

A favorable property of this approach is that the integration now has to be performed only for the (smooth) function G. A less nice property is that technical difficulties arise due to the fact that in general certain tetrahedra are intersected by Γ_h and thus one has to integrate over *parts* of these tetrahedra.

In the second approach the Heaviside function is replaced by a regularized one, for example:

$$H_\varepsilon(x) := \begin{cases} 0 & \text{if } x \le -\varepsilon \\ \nu(\frac{x}{\varepsilon}) & \text{if } |x| < \varepsilon \\ 1 & \text{if } x \ge \varepsilon \end{cases} \tag{3.13}$$

with (cf. [41])

$$\nu(\zeta) = \frac{1}{2} + \frac{1}{32}(45\zeta - 50\zeta^3 + 21\zeta^5) \tag{3.14}$$

The function $\sigma_\varepsilon(\phi)$ is as defined in (2.3) but with H replaced by H_ε. We then apply quadrature to the integral

$$I_\varepsilon := \int_\Omega \sigma_\varepsilon(\phi_h(x))G(x)\,dx$$

Clearly, this method is much easier to implement than the first one. A disadvantage of the second approach is that one has to chose an appropriate value for the regularization parameter ε. An extensive analysis and comparison of these two approaches for the 2D case is given in [41]. Based on these investigations the second approach is used in [42].

In our implementation we also use the approach based on the regularized Heaviside function with a smoothing function ν as in (3.14) and with $\varepsilon \approx h$ in (3.13). In the examples in Section 7 the jumps in the coefficients are rather small and there is only a very weak dependence of the results on the value of ε (as long as $\varepsilon \approx h$). For the quadrature we use rules that are exact for polynomials of degree 2.

4. Time discretization. In this section we discuss the fractional step θ-scheme which we use for time discretization. This technique is introduced in [15] (cf. also [2, 32, 43]). The main motivation for using this particular method is that it allows the reuse of fast iterative solvers for discretized (instationary) Stokes equations that have already been implemented.

We briefly recall the underlying main ideas from [15]. Let there be given a (Hilbert) space H and $F : H \to H$, $f : [0,T] \to H$, $u_0 \in H$. Consider an initial value problem of the form

$$\frac{du}{dt} + F(u) = f(t), \qquad u(0) = u_0$$

For a given decomposition $F = F_1 + F_2$ and a given parameter $\theta \in (0, \frac{1}{2})$, the fractional step θ-scheme for time discretization is based on a subdivision of each time interval $[n\Delta t, (n+1)\Delta t]$ in three subintervals with endpoints $(n+\theta)\Delta t$, $(n+1-\theta)\Delta t$, $(n+1)\Delta t$. For given u^n the approximations $u^{n+\theta}$, $u^{n+1-\theta}$, u^{n+1} at these endpoints are defined by

$$\frac{u^{n+\theta} - u^n}{\theta \Delta t} + F_1(u^{n+\theta}) + F_2(u^n) = f^{n+\theta} \tag{4.1}$$

$$\frac{u^{n+1-\theta} - u^{n+\theta}}{(1-2\theta)\Delta t} + F_1(u^{n+\theta}) + F_2(u^{n+1-\theta}) = f^{n+1-\theta} \tag{4.2}$$

$$\frac{u^{n+1} - u^{n+1-\theta}}{\theta \Delta t} + F_1(u^{n+1}) + F_2(u^{n+1-\theta}) = f^{n+1} \tag{4.3}$$

There are several ways to apply this approach to the instationary Navier-Stokes equations

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} - \Delta \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p &= \mathbf{f} \\ \operatorname{div} \mathbf{u} &= 0 \end{aligned} \tag{4.4}$$

A popular variant (cf. [31, 43]) is based on a splitting of the Navier-Stokes operator $F = \alpha F + (1-\alpha)F$. This then leads to a scheme as in (4.1)-(4.3) in which one has to solve Navier-Stokes equations in each of the three substeps. We refer to [43] for a detailed explanation of this variant.

We follow another approach, introduced in [15] and further analyzed in [22]. This method is based on a splitting in the subspace of divergence free functions of the operator $F(\mathbf{u}) = -\Delta \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u}$ into $F_1(\mathbf{u}) = -\alpha \Delta \mathbf{u}$ and $F_2(\mathbf{u}) = -(1-\alpha)\Delta \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u}$, which leads to the following method for the problem (4.4), presented in [15]:

$$\begin{cases} \frac{\mathbf{u}^{n+\theta} - \mathbf{u}^n}{\theta \Delta t} - \alpha \Delta \mathbf{u}^{n+\theta} + \nabla p^{n+\theta} \\ \qquad = \mathbf{f}^{n+\theta} + (1-\alpha)\Delta \mathbf{u}^n - (\mathbf{u}^n \cdot \nabla)\mathbf{u}^n \\ \operatorname{div} \mathbf{u}^{n+\theta} = 0 \end{cases} \tag{4.5}$$

$$\begin{cases} \frac{\mathbf{u}^{n+1-\theta} - \mathbf{u}^{n+\theta}}{(1-2\theta)\Delta t} - (1-\alpha)\Delta \mathbf{u}^{n+1-\theta} + (\mathbf{u}^{n+1-\theta} \cdot \nabla)\mathbf{u}^{n+1-\theta} \\ \qquad = \mathbf{f}^{n+1-\theta} + \alpha \Delta \mathbf{u}^{n+\theta} - \nabla p^{n+\theta} \end{cases} \tag{4.6}$$

$$\begin{cases} \frac{\mathbf{u}^{n+1} - \mathbf{u}^{n+1-\theta}}{\theta \Delta t} - \alpha \Delta \mathbf{u}^{n+1} + \nabla p^{n+1} \\ \qquad = \mathbf{f}^{n+1} + (1-\alpha)\Delta \mathbf{u}^{n+1-\theta} - (\mathbf{u}^{n+1-\theta} \cdot \nabla)\mathbf{u}^{n+1-\theta} \\ \operatorname{div} \mathbf{u}^{n+1} = 0 \end{cases} \tag{4.7}$$

An important property of this method is that the nonlinearity and incompressibility condition in the Navier-Stokes equations are decoupled. The problems in (4.5), (4.7)

are linear Stokes type of equations and the problem in (4.6) consists of a nonlinear elliptic system for the velocity unknowns.

Clearly, this approach can also be applied after discretization of the space variables. We use this technique for the coupled system in (3.5)-(3.7), where for clarity we drop the arrow notation in $\vec{\mathbf{u}}$ and in the other quantities. This results in the following variant of the fractional step θ-scheme, with $\beta_1 := \frac{1}{\theta\Delta t}$, $\beta_2 := \frac{1}{(1-2\theta)\Delta t}$, $\theta' := 1-\theta$, $\alpha' := 1-\alpha$:

$$\left\{\begin{array}{l} \Big[\big[\beta_1\mathbf{M}+\alpha\mathbf{A}\big](\boldsymbol{\phi}^{n+\theta})\Big]\mathbf{u}^{n+\theta}+\mathbf{B}^T\mathbf{p}^{n+\theta} \\ \qquad = \big[\mathbf{g}+\mathbf{f}_{\Gamma_h}\big](\boldsymbol{\phi}^{n+\theta})+\Big[\big[\beta_1\mathbf{M}-\alpha'\mathbf{A}\big](\boldsymbol{\phi}^{n})\Big]\mathbf{u}^{n}-\mathbf{C}(\boldsymbol{\phi}^n,\mathbf{u}^n)\mathbf{u}^n \\ \mathbf{B}\mathbf{u}^{n+\theta}=0 \\ \Big[\big[\beta_1\mathbf{E}+\alpha\mathbf{H}\big](\mathbf{u}^{n+\theta})\Big]\boldsymbol{\phi}^{n+\theta}=\Big[\big[\beta_1\mathbf{E}-\alpha'\mathbf{H}\big](\mathbf{u}^{n})\Big]\boldsymbol{\phi}^{n} \end{array}\right. \tag{4.8}$$

$$\left\{\begin{array}{l} \Big[\big[\beta_2\mathbf{M}+\alpha'\mathbf{A}\big](\boldsymbol{\phi}^{n+\theta'})\Big]\mathbf{u}^{n+\theta'}+\mathbf{C}(\boldsymbol{\phi}^{n+\theta'},\mathbf{u}^{n+\theta'})\mathbf{u}^{n+\theta'} \\ \qquad = \big[\mathbf{g}+\mathbf{f}_{\Gamma_h}\big](\boldsymbol{\phi}^{n+\theta'})+\Big[\big[\beta_2\mathbf{M}-\alpha\mathbf{A}\big](\boldsymbol{\phi}^{n+\theta})\Big]\mathbf{u}^{n+\theta}-\mathbf{B}^T\mathbf{p}^{n+\theta} \\ \Big[\big[\beta_2\mathbf{E}+\alpha'\mathbf{H}\big](\mathbf{u}^{n+\theta'})\Big]\boldsymbol{\phi}^{n+\theta'}=\Big[\big[\beta_2\mathbf{E}-\alpha\mathbf{H}\big](\mathbf{u}^{n+\theta})\Big]\boldsymbol{\phi}^{n+\theta} \end{array}\right. \tag{4.9}$$

$$\left\{\begin{array}{l} \Big[\big[\beta_1\mathbf{M}+\alpha\mathbf{A}\big](\boldsymbol{\phi}^{n+1})\Big]\mathbf{u}^{n+1}+\mathbf{B}^T\mathbf{p}^{n+1} \\ \qquad = \big[\mathbf{g}+\mathbf{f}_{\Gamma_h}\big](\boldsymbol{\phi}^{n+1})+\Big[\big[\beta_1\mathbf{M}-\alpha'\mathbf{A}\big](\boldsymbol{\phi}^{n+\theta'})\Big]\mathbf{u}^{n+\theta'}-\mathbf{C}(\boldsymbol{\phi}^{n+\theta'},\mathbf{u}^{n+\theta'})\mathbf{u}^{n+\theta'} \\ \mathbf{B}\mathbf{u}^{n+1}=0 \\ \Big[\big[\beta_1\mathbf{E}+\alpha\mathbf{H}\big](\mathbf{u}^{n+1})\Big]\boldsymbol{\phi}^{n+1}=\Big[\big[\beta_1\mathbf{E}-\alpha'\mathbf{H}\big](\mathbf{u}^{n+\theta'})\Big]\boldsymbol{\phi}^{n+\theta'} \end{array}\right. \tag{4.10}$$

The choice of the parameters is based on the analysis in [15]:

$$\theta = 1-\frac{1}{2}\sqrt{2}, \qquad \alpha = \frac{1-2\theta}{1-\theta} = 2-\sqrt{2}\ .$$

Note that the discretization of the level set equation is based on the same α-splitting as for the Navier-Stokes equations and thus is implicit, too. Furthermore, there is a strong coupling between the velocity field and the level set function.

5. Iterative solvers. We outline the iterative solution strategy that is used for the fractional step θ-scheme (4.8)-(4.10). The nonlinear coupling between the flow variables $(\mathbf{u},\mathbf{p})$ and the level set function $\boldsymbol{\phi}$ is linearized by a standard Picard iteration. This turns out to be satisfactory in our simulations. In (4.9) we then obtain a linear system for the level set unknowns and a nonlinear system for the velocity unknowns. To the latter system we again apply the Picard technique. The resulting linear systems for level set unknowns and velocity unknowns are solved using a preconditioned GMRES method. This is a robust and fairly efficient approach. The latter is due to the fact that the mass matrices coming from the time integration improve the conditioning of these linear systems significantly. In (4.8) and (4.10) we have to solve a linear system for the level set unknowns and a discrete Stokes type of problem for velocity and pressure. The major computational effort is needed for solving these discrete Stokes equations. For this we use an inexact Uzawa type of method that was introduced in [5] and further investigated in [29]. We briefly discuss this method.

The discrete Stokes problem has a matrix-vector representation of the form

$$\begin{pmatrix} \mathbf{K} & \mathbf{B}^T \\ \mathbf{B} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{pmatrix} \ , \quad \mathbf{K} := \mathbf{A} + \beta \mathbf{M} \ ,$$

where the parameter β is proportional to $1/\Delta t$. The Schur complement of this matrix is denoted by $\mathbf{S} = \mathbf{B}\mathbf{K}^{-1}\mathbf{B}^T$. A basic method for such a saddle point problem is the Uzawa method. This method is closely related to the block factorization

$$\mathcal{K} := \begin{pmatrix} \mathbf{K} & \mathbf{B}^T \\ \mathbf{B} & 0 \end{pmatrix} = \begin{pmatrix} \mathbf{K} & 0 \\ \mathbf{B} & -\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{K}^{-1}\mathbf{B}^T \\ 0 & \mathbf{S} \end{pmatrix}$$

Let $(\mathbf{x}^k, \mathbf{y}^k)$ be a given approximation to the solution $(\mathbf{x}, \mathbf{y})$. Note that using the block factorization we get

$$\begin{aligned} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} &= \begin{pmatrix} \mathbf{x}^k \\ \mathbf{y}^k \end{pmatrix} + \mathcal{K}^{-1} \left[\begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{pmatrix} - \mathcal{K} \begin{pmatrix} \mathbf{x}^k \\ \mathbf{y}^k \end{pmatrix} \right] \\ &= \begin{pmatrix} \mathbf{x}^k \\ \mathbf{y}^k \end{pmatrix} + \begin{pmatrix} \mathbf{I} & -\mathbf{K}^{-1}\mathbf{B}^T\mathbf{S}^{-1} \\ 0 & \mathbf{S}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{K}^{-1} & 0 \\ \mathbf{B}\mathbf{K}^{-1} & -\mathbf{I} \end{pmatrix} \left[\begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{pmatrix} - \mathcal{K} \begin{pmatrix} \mathbf{x}^k \\ \mathbf{y}^k \end{pmatrix} \right] \end{aligned}$$

We construct an iterative method based on a symmetric positive definite preconditioner $\mathbf{Q}_K$ of $\mathbf{K}$. The Schur complement is replaced by the approximation

$$\hat{\mathbf{S}} := \mathbf{B}\mathbf{Q}_K^{-1}\mathbf{B}^T$$

We use a (nonlinear) approximate inverse of $\hat{\mathbf{S}}$, which is denoted by Ψ, i. e., $\Psi(\mathbf{b})$ is an approximation to the solution of the system $\hat{\mathbf{S}}\mathbf{v} = \mathbf{b}$. With $\mathbf{K}^{-1} \approx \mathbf{Q}_K^{-1}$, $\mathbf{S}^{-1} \approx \hat{\mathbf{S}}^{-1} \approx \Psi(\cdot)$ and $\mathbf{r}_1^k := \mathbf{f}_1 - \mathbf{K}\mathbf{x}^k - \mathbf{B}^T\mathbf{y}^k$ we obtain the (nonlinear) iterative method

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{x}^k + \mathbf{Q}_K^{-1}\mathbf{r}_1^k - \mathbf{Q}_K^{-1}\mathbf{B}^T\Psi\left(\mathbf{B}(\mathbf{Q}_K^{-1}\mathbf{r}_1^k + \mathbf{x}^k) - \mathbf{f}_2\right) \\ \mathbf{y}^{k+1} &= \mathbf{y}^k + \Psi\left(\mathbf{B}(\mathbf{Q}_K^{-1}\mathbf{r}_1^k + \mathbf{x}^k) - \mathbf{f}_2\right) \end{aligned}$$

This yields the following algorithmic structure:

$$\left\{ \begin{array}{l} \begin{pmatrix} \mathbf{x}^0 \\ \mathbf{y}^0 \end{pmatrix} \text{ a given starting vector; } \mathbf{r}_1^0 := \mathbf{f}_1 - \mathbf{K}\mathbf{x}^0 - \mathbf{B}^T\mathbf{y}^0 \\ \text{for } k \geq 0 \ : \\ \quad \mathbf{w} := \mathbf{x}^k + \mathbf{Q}_K^{-1}\mathbf{r}_1^k \qquad (5.1) \\ \quad \mathbf{z} := \Psi(\mathbf{B}\mathbf{w} - \mathbf{f}_2) \qquad (5.2) \\ \quad \mathbf{x}^{k+1} := \mathbf{w} - \mathbf{Q}_K^{-1}\mathbf{B}^T\mathbf{z} \qquad (5.3) \\ \quad \mathbf{y}^{k+1} := \mathbf{y}^k + \mathbf{z} \qquad (5.4) \\ \quad \mathbf{r}_1^{k+1} := \mathbf{r}_1^k - \mathbf{K}(\mathbf{x}^{k+1} - \mathbf{x}^k) - \mathbf{B}^T\mathbf{z} \qquad (5.5) \end{array} \right.$$

This is the same method as the one presented in Section 4 in [5]. A very efficient Poisson solver is the multigrid method. Thus for the preconditioner $\mathbf{Q}_K$ we use the following:

$$\mathbf{Q}_K^{-1}\mathbf{b} = \left\{ \begin{array}{l} \text{One symmetric multigrid } V\text{-cycle iteration, using one} \\ \text{symmetric Gauss-Seidel iteration for pre- and post-} \\ \text{smoothing, with starting vector 0 applied to } \mathbf{K}\mathbf{v} = \mathbf{b}. \end{array} \right. \qquad (5.6)$$

A natural choice for $\Psi(\mathbf{Bw} - \mathbf{f}_2)$ is the following:

$$\Psi(\mathbf{Bw} - \mathbf{f}_2) = \begin{cases} \text{Result of } \ell \text{ PCG iterations with starting vector } 0 \\ \text{and preconditioner } \mathbf{Q}_S \text{ applied to } \hat{\mathbf{S}}\mathbf{v} = \mathbf{Bw} - \mathbf{f}_2. \end{cases} \tag{5.7}$$

An important issue is the choice of the Schur complement preconditioner $\mathbf{Q}_S$. We use an approach as presented in [14]. If the jump in the viscosity (or density) coefficient is very large then modifications as discussed in [25, 26] are needed. A further important issue in the algorithm is the stopping criterion for the inner PCG iteration. From the analysis presented in [29] we deduce the following stopping criterion: We stop the PCG iteration if

$$\|\mathbf{r}^\ell\|_2 \leq \sigma_{\text{acc}} \|\mathbf{r}^0\|_2 \quad \text{is satisfied, with a given } \sigma_{\text{acc}} \in [0.2, 0.6].$$

Here $\mathbf{r}^i$ denotes the residual in the PCG algorithm. The arithmetic costs per iteration of the algorithm are dominated by the evaluations of $\mathbf{Q}_K^{-1}$. The method can be implemented in such a way that per outer iteration of the algorithm (5.1)-(5.7) we need $\ell+1$ evaluations of $\mathbf{Q}_K^{-1}$, ℓ evaluations of $\mathbf{Q}_S^{-1}$, $\ell+1$ matrix-vector multiplications with $\mathbf{B}$, ℓ matrix-vector multiplications with $\mathbf{B}^T$ and one matrix-vector multiplication with $\mathbf{K}$.

A comparison of this method with other well-known Stokes solvers is presented in [29]. A convergence analysis is given in [5, 29].

6. Maintenance of the level set function. Relying only on the advection equation (2.6) for the evolution of the level set function is not enough. The level set function would degenerate over time. Strategies to prevent this undesirable behavior are discussed in this section.

6.1. Reparametrization of the level set function. During the evolution of the level set function ϕ, which is driven by the velocity field, the property of ϕ being close to a (signed) distance function is lost. This affects the treatment of the discontinuities and the refinement of the interfacial region. Moreover, the advection of ϕ becomes less accurate. Therefore, a reparametrization technique is used to reestablish this property. Important issues related to this reparametrization of ϕ are the following:

1. The 0-level of ϕ should be preserved.
2. The norm of the gradient of ϕ should be close to one: $\|\nabla\phi\| \approx 1$.
3. The reparametrization can be used to smooth ϕ (close to the interface) and thus stabilize the evolution of the level set function.

Different reparametrization techniques are known in the literature, cf. [35, 36]. The most often used method is based on a pseudo time stepping scheme for the Eikonal equation

$$\|\nabla\psi\| = 1$$

Let ϕ_h be a given approximation of the level set function, and consider the following first order partial differential equation for $\psi = \psi(x, \tau)$:

$$\begin{aligned} \frac{d\psi}{d\tau} &= S_\alpha(\phi_h)(1 - \|\nabla\psi\|) \ , \qquad \tau \geq 0, \ x \in \Omega \\ \psi(x, 0) &= \phi_h \ , \end{aligned} \tag{6.1}$$

with

$$S_\alpha(\zeta) = \frac{\zeta}{\sqrt{\zeta^2 + \alpha^2}}\ , \qquad \zeta \in \mathbb{R},$$

where α is a regularization parameter ($0 < \alpha \ll 1$). The function S_α is a smoothed sign function. It keeps the 0-level invariant (due to $S_\alpha(0) = 0$) and guarantees that the solution converges for $\tau \to \infty$ to a solution of the Eikonal equation. Thus, for sufficiently large $T > 0$ one can use the function $\psi(\cdot, T)$ as a reparametrization of ϕ_h.

The equation (6.1) can be reformulated in the more convenient form

$$\frac{d\psi}{d\tau} + \mathbf{w}(\psi) \cdot \nabla\psi = S_\alpha(\phi_h) \quad \text{with} \quad \mathbf{w}(\psi) := S_\alpha(\phi_h)\frac{\nabla\psi}{\|\nabla\psi\|} \tag{6.2}$$

The equation (6.2) can be solved numerically and then yields a reparametrization of ϕ_h. To stabilize the evolution a diffusion term can be added to the equation. For a further discussion of this reparametrization method we refer to the literature. We implemented such a method, but encountered the following two difficulties with this approach. Firstly, the algorithm is difficult to control because several parameters have to be chosen: the regularization parameter α, the diffusion parameter, the time T, the time step in the evolution. Secondly, and more important, in our simulations the 0-level was changed too much.

We then considered alternative reparametrization methods. A simple variant of the Fast Marching method (cf. [21, 34]) turned out to perform much better in our numerical simulations. Because this variant is new we give a rather detailed explanation.

Let there be given a continuous piecewise quadratic function $\phi_h \in V_k$ corresponding to the triangulation $\mathcal{T}_k$. We introduce some notation. The regular refinement of $\mathcal{T}_k$ is denoted by $\mathcal{T}_k' := \{\, T' \in \mathcal{K}(T) \mid T \in \mathcal{T}_k \,\}$. The collection of all vertices in $\mathcal{T}_k'$ is denoted by $\mathcal{V}$. Note that ϕ_h is uniquely determined by its values on $\mathcal{V}$. For $T \in \mathcal{T}_k'$, $V(T)$ is the set of the four vertices of T. Furthermore, for $v \in \mathcal{V}$, $\mathcal{T}(v)$ is the set of all tetrahedra which have v as a vertex: $\mathcal{T}(v) = \{\, T \in \mathcal{T}_k' \mid v \in V(T) \,\}$. Finally, for $v \in \mathcal{V}$, $N(v)$ is the collection of all neighboring vertices of v (i. e., for each $w \in N(v)$ there is an edge in $\mathcal{T}_k'$ connecting v and w): $N(v) = \cup_{T \in \mathcal{T}(v)} V(T) \setminus \{v\}$.

Let Γ_h be the discrete approximation of the interface as defined in (3.9) and $\mathcal{I} \subset \mathcal{T}_k'$ the collection of tetrahedra which "contains" the discrete interface as defined in (3.10), i. e. (for ease of notation we now use T instead of T'):

$$\begin{cases} \Gamma_T := \Gamma_h \cap T \quad \text{is a plane segment, for all } T \in \mathcal{I}, \\ \text{and} \quad \Gamma_h = \cup_{T \in \mathcal{I}} \Gamma_T \end{cases} \tag{6.3}$$

The plane segment Γ_T in (6.3) is either a triangle or a quadrilateral.

We first explain the initialization phase of the reparametrization algorithm. We define the set of vertices corresponding to $\mathcal{I}$:

$$\mathcal{F} := \{\, v \in V(T) \mid T \in \mathcal{I} \,\} \tag{6.4}$$

For each $v \in \mathcal{F}$ we define a discrete (approximate) distance function $d(v)$ as follows. For $v \in \mathcal{F}$ and $T \in \mathcal{T}(v) \cap \mathcal{I}$ let Γ_T be the plane segment as in (6.3), with vertices

denoted by $Q_1, \ldots, Q_m$, $m = 3$ or 4. Let W be the plane in $\mathbb{R}^3$ which contains the plane segment Γ_T and $P_W : \mathbb{R}^3 \to W$ the orthogonal projection on W. We define

$$d_T(v) := \begin{cases} \|v - P_W v\| & \text{if } \; P_W v \in T \\ \min_{1 \le j \le m} \|v - Q_j\| & \text{otherwise} \end{cases}, \quad \text{for } \; v \in \mathcal{F}, \; T \in \mathcal{T}(v) \cap \mathcal{I}$$

The quantity $d_T(v)$ is a measure for the distance between v and Γ_T. Note that if $P_W v \in T$ holds, then $d_T(v)$ is precisely this distance. Since Γ_h consists of piecewise planar segments Γ_T ($T \in \mathcal{I}$) we define as an approximate distance function:

$$d(v) := \min_{T \in \mathcal{T}(v) \cap \mathcal{I}} d_T(v) \quad \text{for } \; v \in \mathcal{F} \tag{6.5}$$

After this initialization phase the grid function $\{ (v, d(v)) \mid v \in \mathcal{F} \}$ is an approximate distance function from the interface Γ_h for the vertices $v \in \mathcal{F}$.

The second phase of the reparametrization algorithm consists of a loop in which the approximate distance function is extended to neighbor vertices of $\mathcal{F}$ and then to neighbors of neighbors, etc. To explain this more precisely we introduce a set of so-called active vertices $\mathcal{A}$, which consists of vertices $v \notin \mathcal{F}$ that have a neighboring vertex in $\mathcal{F}$:

$$\mathcal{A} := \{ v \in \mathcal{V} \setminus \mathcal{F} \mid N(v) \cap \mathcal{F} \neq \emptyset \} \tag{6.6}$$

For $v \in \mathcal{A}$ we define an approximate distance function in a similar way as in the initialization phase. Take $v \in \mathcal{A}$ and $T \in \mathcal{T}(v)$ with $V(T) \cap \mathcal{F} \neq \emptyset$. Note that such a T exists if $\mathcal{A}$ is nonempty. There are three possible cases, namely $|V(T) \cap \mathcal{F}| = 1, 2$ or 3. If $|V(T) \cap \mathcal{F}| = 1$, say $V(T) \cap \mathcal{F} = \{w\}$, we define $d_T(v) := d(w) + \|v - w\|$. For the other two cases, i. e., $V(T) \cap \mathcal{F} = \{w_i\}_{1 \le i \le m}$ with $m = 2$ or $m = 3$, we use an orthogonal projection as in the initialization phase. Let W be the line (plane) in $\mathbb{R}^3$ through the points w_1, w_2 (w_3) and $P_W : \mathbb{R}^3 \to W$ the orthogonal projection on W. We define

$$d_T(v) := \begin{cases} d(P_W v) + \|v - P_W v\| & \text{if } \; P_W v \in T \\ \min_{1 \le j \le m} \Big[d(w_j) + \|v - w_j\| \Big] & \text{otherwise} \end{cases} \tag{6.7}$$

The value $d(P_W v)$ in (6.7) is determined by linear interpolation of the known values $d(w_j)$, $1 \le j \le m$. Note that $P_W v \in T$ is satisfied if all faces of T are acute triangles. The approximate distance function at active vertices is defined by

$$d(v) := \min\{ d_T(v) \mid T \in \mathcal{T}(v) \;\; \text{with} \;\; V(T) \cap \mathcal{F} \neq \emptyset \}, \quad v \in \mathcal{A} \tag{6.8}$$

The reparametrization method is as follows:

1. Initialization: construct $\mathcal{F}$ and compute $d(\mathcal{F})$ as in (6.4), (6.5).
2. Construct active set $\mathcal{A}$ and compute $d(\mathcal{A})$ as in (6.6), (6.8).
3. Determine $v_{\min} \in \mathcal{A}$ such that $d(v_{\min}) = \min_{v \in \mathcal{A}} d(v)$.
4. Construct $\mathcal{F} := \mathcal{F} \cup \{v_{\min}\}$, $\mathcal{N} := N(v_{\min}) \setminus \mathcal{F}$, $\mathcal{A} := (\mathcal{A} \cup \mathcal{N}) \setminus \{v_{\min}\}$.
5. (Re)compute $d(v)$ for $v \in \mathcal{N}$.
6. If $|\mathcal{A}| > 0$ then go to 3.
7. For all $v \in \mathcal{V}$, set $d(v) := \text{sign}(\phi_h(v)) \cdot d(v)$

After this reparametrization we have a grid function $d(v),\ v \in \mathcal{V}$, which uniquely determines a continuous piecewise quadratic function $\chi_h \in V_k$ on the triangulation $\mathcal{T}_k$. This χ_h is the reparametrization of ϕ_h. For χ_h one can construct an approximate 0-level set $\tilde{\Gamma}_h$ as described in Section 3.3. The reparametrization procedure guarantees $\tilde{\Gamma}_h \subset \mathcal{I}$. However, in general we have $\tilde{\Gamma}_h \neq \Gamma_h$, i. e., the discrete 0-level set may be slightly changed. Since χ_h is close to a signed distance function, the variations in $\nabla\chi_h$ are usually smaller than the variations in $\nabla\phi_h$. Due to this property, the reparametrization method has a stabilizing effect.

In our simulations the time needed for the reparametrization is negligible compared to the computing times for discretization and iterative solution of the Navier-Stokes equations.

6.2. Smoothing the level set function. In the continuous model the interface is smoothed due to the surface tension force. When the surface tension coefficient τ is enlarged, this force increases and the time scale in which the smoothing effect occurs decreases. The finite resolution of the spatial discretization causes small geometric irregularities at the interface. For small τ values they are smoothed out as expected. For a large τ value, however, the time scale in which the smoothing takes place can be smaller than the time resolution of the numerical scheme. In such a situation the surface tension force is applied over a too long time period during the evolution, shifting the interface too far. This results in an oscillatory instead of a smooth interface.

This undesirable effect can be overcome by using either a higher resolution in time and/or space or some numerical interface smoothing procedure. Clearly, the first approach results in higher accuracy but in general is also more costly in terms of computing time. To allow larger time steps we implemented a strategy from [42] in which a smoothed interface is used to calculate $\mathbf{f}_{\Gamma_h}$. This prevents that spurious (nonphysical) forces spoil the computations.

The smoothing of the interface is based on an artificial diffusion equation for the level set function, i. e., we solve the following equation for some small parameter ε (we usually have $\varepsilon \approx 10^{-10}$):

$$\phi_{\text{smooth}} - \varepsilon\Delta\phi_{\text{smooth}} = \phi$$

The discretized version has the form

$$[\mathbf{M}_\phi + \varepsilon\mathbf{A}_\phi]\boldsymbol{\phi}_{\text{smooth}} = \mathbf{M}_\phi\boldsymbol{\phi}$$

with $\mathbf{M}_\phi$ the mass matrix and $\mathbf{A}_\phi$ the stiffness matrix in the finite element space V_h. This system can be solved with a PCG or multigrid method.

6.3. Conservation of mass. The algorithm (4.8)-(4.10) does not conserve mass. Due to the surface tension, we usually *lose* mass from Ω_1. This loss of mass is reduced if the grid is refined. Such finer grids, however, result in higher computational costs. Therefore we introduce another strategy to compensate for the mass loss.

After each time step, we shift the interface in normal direction such that the volume of Ω_1 at current time is the same as at time $t = 0$. To realize this we exploit the fact that the level set function is close to a signed distance function. In order to shift the interface over a distance d in outward normal direction, we only have to subtract d from the level set function.

Let $V(\phi) := \text{vol}\{\, x \in \Omega \mid \phi(x) < 0 \,\}$ denote the volume of Ω_1 corresponding to a level set function ϕ and let ϕ_h be the discrete level set function at a given time. We

have to find $d \in \mathbb{R}$ such that

$$V(\phi_h - d) - \mathrm{vol}(\Omega_1(0)) = 0$$

holds. In order to keep the number of evaluations of V low, we use a method with a high rate of convergence, namely the Anderson-Björk method [1], to solve this equation. We then set $\phi_h^{\mathrm{new}} := \phi_h - d$ and discard ϕ_h.

Note that this strategy only works if Ω_1 consists of a single component. If there are multiple components, mass must be preserved for each of them. In this case the algorithm can be modified to shift ϕ_h only locally. Discontinuities that may occur in the level set function can be removed by a reparametrization step.

Finally note that the shifting of the level set function to obtain a better mass conservation introduces a new source of discretization errors.

7. Numerical results. In this section we present results of three different experiments performed with the software package DROPS. The first two experiments concentrate on selected issues, namely the reparametrization method and the effects of the surface tension. In the third example we consider a realistic two-phase flow experiment.

7.1. Experiment 1: Reparametrization and smoothing. The first experiment shows the effect of the reparametrization and the smoothing of the level set function. For better visualization only 2D results are presented. The illustrated phenomena also occur in 3D.

We start on the unit square with a level set function ϕ_0 as shown in Figure 7.1. The interface is the intersection of ϕ_0 with the zero plane. Close to the interface we have $\|\nabla\phi_0\| \approx 3.2$.

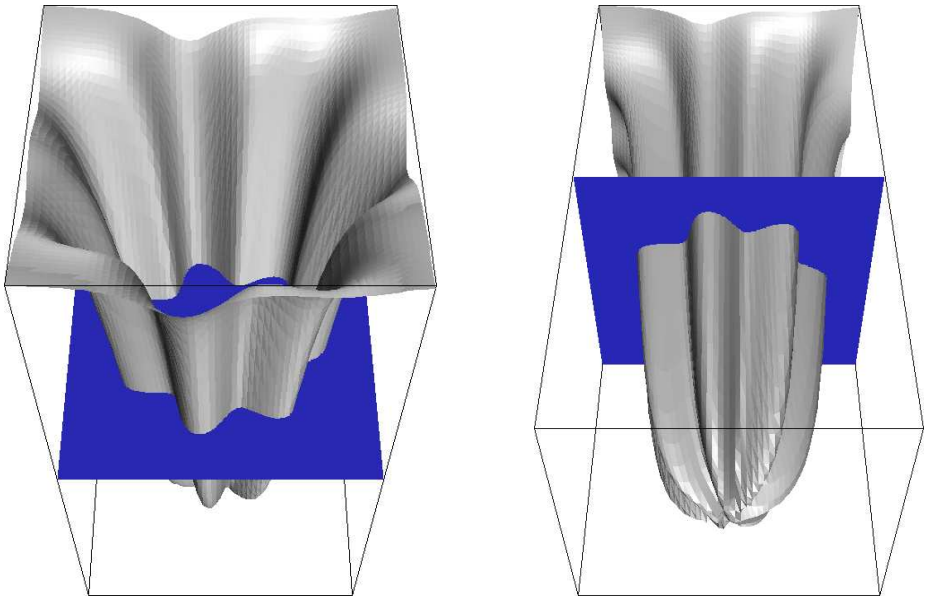

FIG. 7.1. *Initial level set function shown from above and below the zero plane.*

Figure 7.2 shows the level set function after the application of the Fast Marching method presented in Section 6.1. The interface is well preserved, but now $\|\nabla\phi\| \approx 1$.

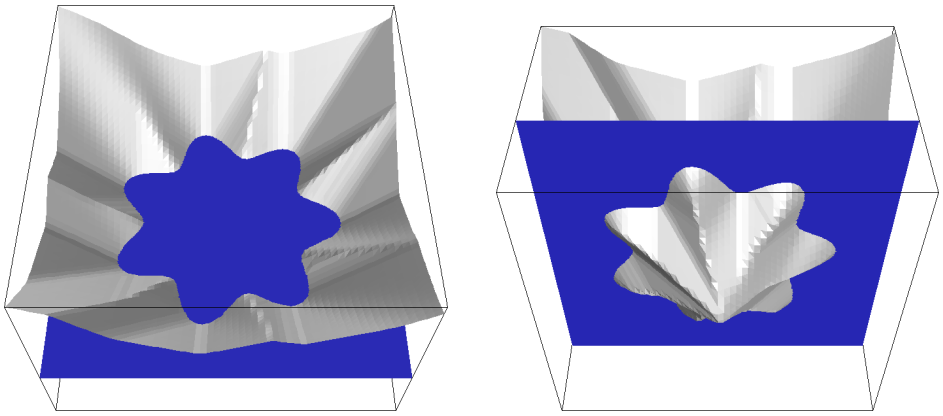

FIG. 7.2. *Reparametrized level set function.*

For the following calculations we added random noise to ϕ_0 to obtain a new initial level set function ϕ_1, shown in Figure 7.3. Note, that the noise also moves the interface.

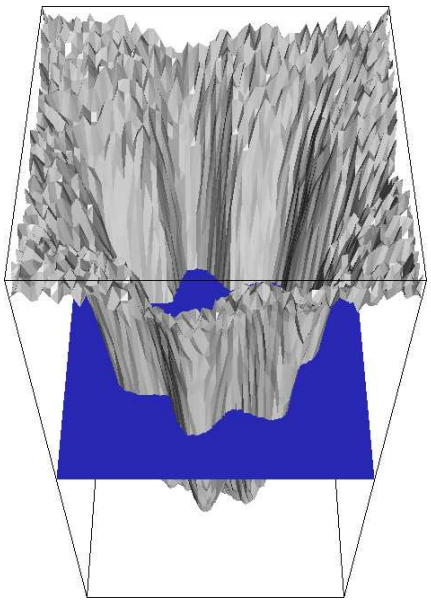

FIG. 7.3. *Noisy level set function.*

As expected, the Fast Marching method applied to ϕ_1 yields a level set function whose shape is roughly the same as in Figure 7.2. However, due to the noise there are small distortions of the 0-level as shown in the left blow up picture in Figure 7.4. To smooth the interface we apply the technique from Section 6.2 and obtain the level set function shown on the right in Figure 7.4.

7.2. Experiment 2: Effect of the surface tension. In this experiment we study the effect of the surface tension on a drop in a quiescent fluid, i. e., we concentrate on the localized force term on the right-hand side in (2.4).

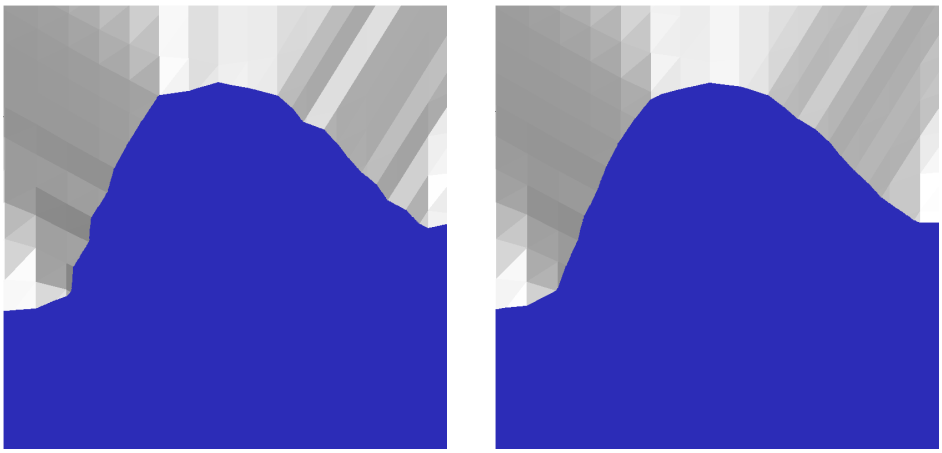

FIG. 7.4. *Detail of noisy level set function after the reparametrization and additional smoothing.*

Because $\mathbf{u} = 0$ the free boundary condition reduces to $[p\mathbf{n}]_\Gamma = \tau\kappa\mathbf{n}$ and we obtain a jump relation for the pressure:

$$\Delta_p := p_{\Omega_1} - p_{\Omega_2} = \tau\kappa \tag{7.1}$$

We take a spherical drop with radius $r_D = 2 \cdot 10^{-3}$ which is located in the middle of a cube Ω with edge length $6 \cdot 10^{-3}$. We consider the equations (2.4) – (2.6) with homogeneous Dirichlet boundary conditions for $\mathbf{u}$. As initial conditions we have $\mathbf{u}_0 = 0$ and ϕ_0 a signed distance function w. r. t. the drop. Without gravity ($\mathbf{g} = 0$) the solution of this problem is stationary with $\mathbf{u} = 0$ and $\phi = \phi_0$ for all $t \geq 0$ independent of the values of ρ and μ. For simplicity we take $\rho_1 = \rho_2 = \mu_1 = \mu_2 = 1$ in the computations.

The triangulation of the cube Ω used for the numerical simulation consists of $4 \times 4 \times 4$ subcubes, each subdivided into 6 tetrahedra, which are then locally refined three times in the neighborhood of the phase interface. Figure 7.5 shows the computed pressure distribution for two different surface tension coefficients τ along a line that is parallel to one of the edges of Ω and goes through the center of the drop. On the horizontal axis, r denotes the signed distance from the center of the drop. If the grid is further refined the oscillations at the interface in Figure 7.5 decrease and the computed pressure gets closer to a piecewise constant function.

Next we consider the difference $\tilde{\Delta}_p := p|_{r=0} - p|_{r=3\cdot 10^{-3}}$ of computed pressure values at the center of the drop and at a point outside of the drop (on the boundary of the cube Ω). Figure 7.6 shows $\tilde{\Delta}_p$ for different values of τ. With $\kappa = \frac{1}{r_D} + \frac{1}{r_D} = 10^3$ the observed pressure jump behaves as expected: $\tilde{\Delta}_p \approx \kappa\tau$. We repeated this experiment for a fixed surface tension $\tau = 10^{-3}$ and different radii of the drop r_D, thus varying the curvature κ. The results shown in Figure 7.7 are also in good agreement with (7.1).

We conclude that in this experiment the numerical treatment of the localized force term which models the effect of surface tension yields satisfactory results.

7.3. Experiment 3: Levitated drop in a measuring cell. The final experiment originates from an interdisciplinary research project, cf. [37]. A main topic in this project is the modeling of flow and mass transfer phenomena at the interface

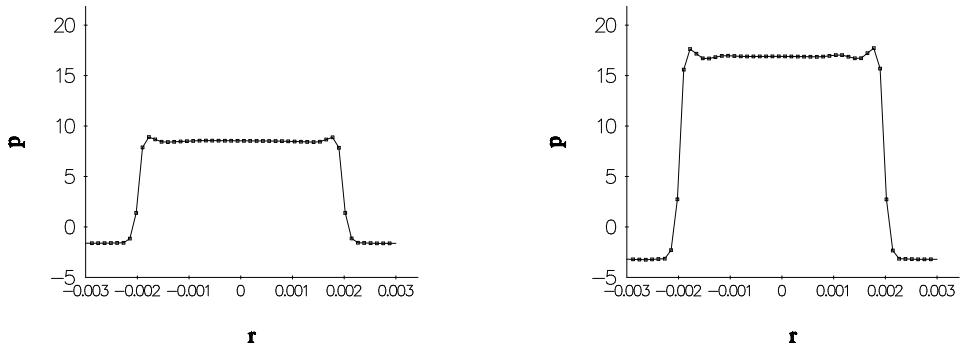

FIG. 7.5. *Pressure distribution for* $\tau = 1 \cdot 10^{-2}$ *and* $\tau = 2 \cdot 10^{-2}$ *respectively.*

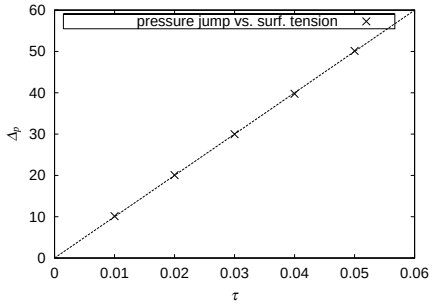

FIG. 7.6. *Computed pressure jump* $\tilde{\Delta}_p$ *versus surface tension* τ *for fixed drop radius* $r_D = 2 \cdot 10^{-3}$.

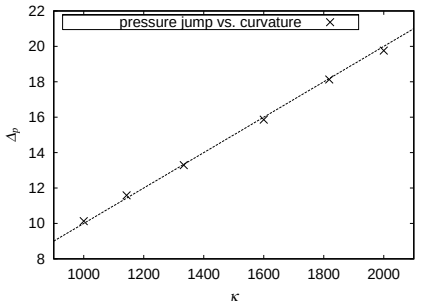

FIG. 7.7. *Computed pressure jump* $\tilde{\Delta}_p$ *versus curvature* $\kappa = 2/r_D$ *for fixed surface tension* $\tau = 10^{-3}$.

between a single droplet and a surrounding fluid. For (NMR) measurements a drop is levitated in a special device, which consists of a vertical glass tube with a narrowing in the middle. This device, in horizontal position, is illustrated in Figure 7.8. A fluid flows from the the top of the tube downwards. A drop, which is lighter than the surrounding fluid, is injected at the bottom of the tube. This drop rises up to a certain point, where its buoyancy forces are balanced by the forces induced by the flow from above. The aim of a first numerical simulation is to compute the equilibrium position and shape of the drop.

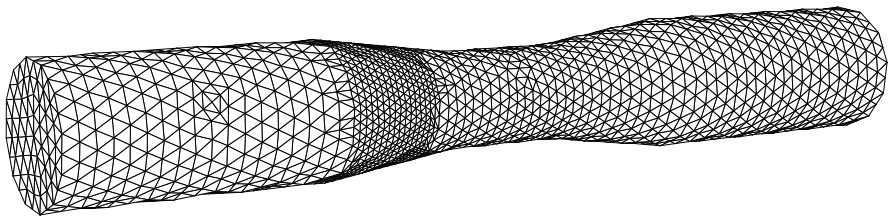

FIG. 7.8. *Triangulated measuring cell.*

The computational domain Ω is shown in Figure 7.8. The tube is $5 \cdot 10^{-2}\,[m]$ long and has a diameter of $d_{\text{in}} = 7 \cdot 10^{-3}\,[m]$ at the inlet and outlet and a diameter

of $5 \cdot 10^{-3}\,[m]$ at the narrowest part of the tube. As an initial condition the drop is assumed to be spherical with a diameter of $3.5 \cdot 10^{-3}\,[m]$. The center of the drop is located in the middle of the tube, $7.5 \cdot 10^{-3}\,[m]$ below the narrowest part, which is near the expected equilibrium position. The initial triangulation $\mathcal{T}_0$ is locally refined in the neighborhood of the drop. The boundary part of this refinement region can be seen in Figure 7.8. The finest triangulation $\mathcal{T}_2$ consists of about 11000 tetrahedra. Roughly 75% of these tetrahedra are located in the refinement region.

We consider the equations (2.4) – (2.6) with a given inflow at the inlet, outflow boundary conditions at the outlet and no-slip (homogeneous Dirichlet) boundary conditions for $\mathbf{u}$ at the remaining boundaries of Ω. The inflow velocity has a stationary parabolic profile with maximum value of $v_{\text{in}} = 25 \cdot 10^{-3}\,[m\,s^{-1}]$ in the middle of the inlet. For the initial conditions, ϕ_0 is taken as a signed distance function for the initial spherical drop and $\mathbf{u}_0$ is taken as the solution of the stationary Stokes problem

$$\begin{aligned} -\operatorname{div}\left(\mu(\phi_0)\,\mathbf{D}(\mathbf{u})\right) + \nabla p &= \rho(\phi_0)\,\mathbf{g} \\ \operatorname{div}\mathbf{u} &= 0 \end{aligned}$$

The densities have values $\rho_1 = 955\,[kg\,m^{-3}]$, $\rho_2 = 1107\,[kg\,m^{-3}]$, the dynamic viscosities are $\mu_1 = 10.4 \cdot 10^{-3}\,[N\,s\,m^{-2}]$, $\mu_2 = 4.8 \cdot 10^{-3}\,[N\,s\,m^{-2}]$ and the surface tension coefficient is $\tau = 2 \cdot 10^{-3}\,[N\,m^{-1}]$. The Reynolds number at the inlet is

$$Re_{\text{in}} := \frac{\rho_2\, d_{\text{in}} v_{\text{in}}}{\mu_2} \approx 34.4$$

At the outlet the Reynolds number is roughly the same because of similar conditions. At the narrowest part of the tube the Reynolds number is about 56.5 due to higher velocities.

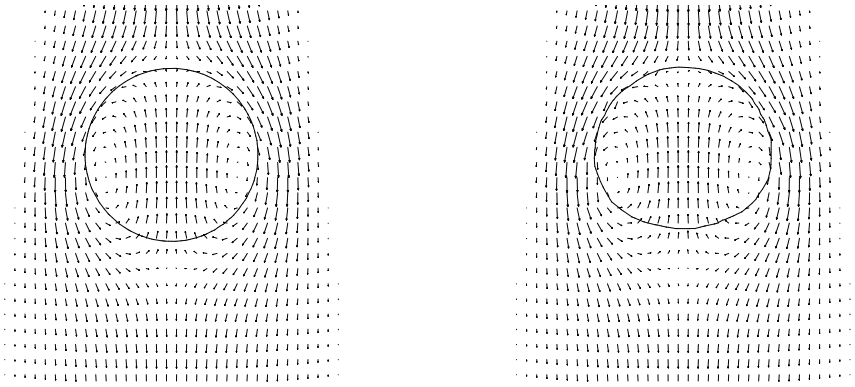

FIG. 7.9. *Interface and velocity field for* $t = 0$ *and* $t = 0.015$ *resp.*

Figures 7.9 and 7.10 show the velocity field and the interface of the drop at different times. For visualization purposes, the solution is plotted on a 2D cartesian grid intersecting the unstructured tetrahedral grid. The 3D shape of the drop at its equilibrium position is shown in Figure 7.11.

Compared to the real system, a silicon oil drop in D_2O, the viscosities used in this simulation are 4 times larger and the surface tension coefficient is about 20 times smaller. The reason why we used larger viscosities is that we first wanted to test the solver for a problem with small Reynolds numbers. In a next step the problem with

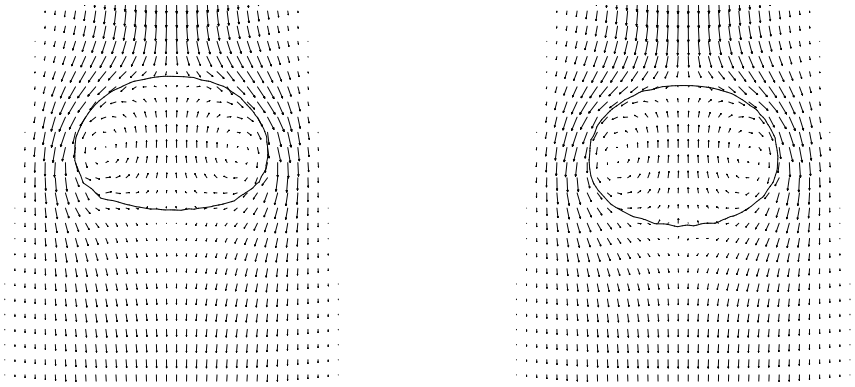

FIG. 7.10. *Interface and velocity field for* $t = 0.07$ *and* $t = 0.265$ *resp.*

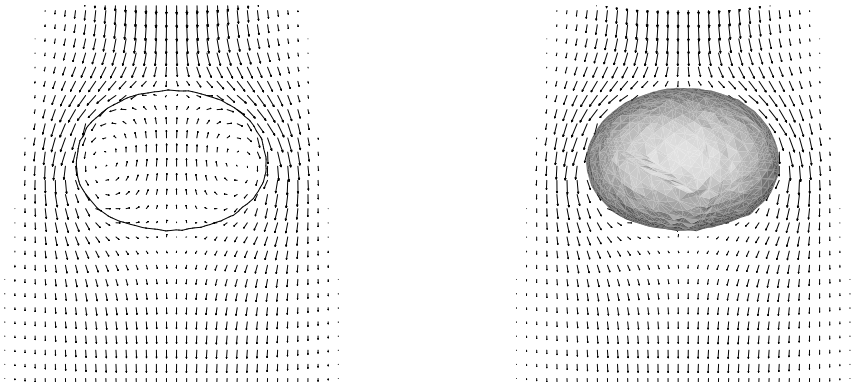

FIG. 7.11. *Interface and velocity field for* $t = 0.76$.

larger Reynolds numbers will be treated using a streamline diffusion stabilization, which is already implemented for the discretization of the level set equation, for the Navier-Stokes equation, too. The reason why we consider a smaller surface tension coefficient is twofold. Firstly, for the physically realistic value the rising droplet remains very close to spherical which is less nice for demonstrating the interface capturing behavior of the level set technique. Secondly, for (very) large surface tension coefficients we are not satisfied with our solver, yet. This is due to the fact that in this case we have (very) large forces at the interface which can be controlled either by increasing the resolution of the level set function or using some stabilization technique. This is a topic of current research.

REFERENCES

[1] N. ANDERSON, Å. BJÖRCK, *A new high Order Method of Regula Falsi Type for Computing the Root of an Equation*, BIT 13, pp. 253-264, 1973.

[2] E. BÄNSCH, *Numerical methods for the instationary Navier-Stokes equations with a free capillary surface*, Habilitation thesis, Albert-Ludwigs-University Freiburg, 1998.

[3] E. BÄNSCH, *Finite element discretization of the Navier-Stokes equations with a free capillary surface*, Numer. Math. 88, pp. 203–235, 2001.

[4] R. E. BANK, A. H. SHERMAN, A. WEISER, *Refinement algorithms and data structures for regular local mesh refinement*, in: Scientific computing (R. Stepleman, ed.), North-Holland,

Amsterdam, pp. 3–17, 1983.
[5] R. E. Bank, B. D. Welfert, H. Yserentant, *A class of iterative methods for solving saddle point problems*, Numer. Math. 56, pp. 645–666, 1990.
[6] P. Bastian, *Parallele adaptive Mehrgitterverfahren*, Teubner, Stuttgart, 1996.
[7] P. Bastian, K. Birken, K. Johannsen, S. Lang, N. Neuss, H. Rentz-Reichert, C. Wieners, *UG - A flexible software toolbox for solving partial differential equations*, Computing and Visualization in Science 1, pp. 27–40, 1997.
[8] P. Bastian, *Load balancing for adaptive multigrid methods*, SIAM J. Sci. Comput. 19, pp. 1303–1321, 1998.
[9] P. Bastian, K. Birken, K. Johannsen, S. Lang, V. Reichenberger, G. Wittum, C. Wrobel, *A parallel software-platform for solving problems of partial differential equations using unstructured grids and adaptive multigrid methods*, In: High performance computing in science and engineering (E. Krause and W. Jäger, eds.), pp. 326–339, Springer, Berlin, 1999.
[10] J. Bey, *Tetrahedral grid refinement*, Computing 55, pp. 355–378, 1995.
[11] J. Bey, *Finite-Volumen- und Mehrgitterverfahren für elliptische Randwertprobleme*, Advances in Numerical Methods, Teubner, Stuttgart, 1998.
[12] J. Bey, *Simplicial grid refinement: on Freudenthal's algorithm and the optimal number of congruence classes*, Numer. Math. 85, pp. 1–29, 2000.
[13] J. U. Brackbill, D. B. Kothe, C. Zemach, *A continuum method for modeling surface tension*, J. Comput. Phys. 100, pp. 335–354, 1992.
[14] J. H. Bramble, J. E. Pasciak, *Iterative techniques for time dependent Stokes problems*, Comput. Math. Appl. 33, No. 1-2, pp. 13–30, 1997.
[15] M. O. Bristeau, R. Glowinski, J. Periaux, *Numerical methods for the Navier-Stokes equations. Application to the simulation of compressible and incompressible flows*, Computer Physics Report 6, pp. 73–188, 1987.
[16] Y. C. Chang, T. Y. Hou, B. Merriman, S. Osher, *A level set formulation of Eulerian interface capturing methods for incompressible fluid flows*, J. Comput. Phys. 124, pp. 449–464, 1996.
[17] K. Deckelnick, G. Dziuk, *Mean curvature flow and related topics*, In: Frontiers in Numerical Analysis, Durham 2002 (J. F. Blowey, A. W. Craig, T. Shardlow, eds.), pp. 63–108, Springer, 2003.
[18] G. Dziuk, *An algorithm for evolutionary surfaces*, Numer. Math. 58, pp. 603–611, 1991.
[19] S. Gross, A. Reusken, *Parallel multilevel tetrahedral grid refinement*, IGPM report 227, RWTH Aachen, 2002. To appear in SIAM J. Sci. Comp.
[20] KASKADE, A toolbox for adaptive multilevel codes, `http://www.zib.de/Scisoft/kaskade2/`
[21] R. Kimmel, J. A. Sethian, *Computing geodesic paths on manifolds*, Proc. Natl. Acad. Sci. 95, pp. 8431–8435, 1998.
[22] P. Kloucek, F. Rys, *Stability of the fractional step θ-scheme for the nonstationary Navier-Stokes equations*, SIAM J. Numer. Anal. 31, pp. 1312–1335, 1994.
[23] S. Lang, *Parallele numerische Simulation instationärer Probleme mit adaptiven Methoden auf unstrukturierten Gittern*, Ph.D. thesis, University of Stuttgart, 2001.
[24] S. Lang, *UG - A parallel software tool for unstructured adaptive multigrids*, In: Parallel Computing: Fundamentals & Applications, Proceedings of the International conference ParCo'99 (E. H. D'Hollander, J. R. Joubert, F. J. Peters, H. Sips, eds.), Imperial College Press, Delft, 1999.
[25] M. A. Olshanskii, A. Reusken, *Analysis of a Stokes interface problem*, IGPM report 237, RWTH Aachen, 2004, submitted.
[26] M. A. Olshanskii, A. Reusken, *A Stokes interface problem: stability, finite element analysis and a robust solver*, IGPM report 239, RWTH Aachen, 2004, submitted.
[27] S. Osher, J. A. Sethian, *Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations*, J. Comp. Phys. 79, pp. 12–49, 1988.
[28] S. Osher, R. P. Fedkiw, *Level set methods: An overview and some recent results*, J. Comput. Phys. 169, pp. 463–502, 2001.
[29] J. Peters, V. Reichelt, A. Reusken, *Fast iterative solvers for discrete Stokes equations*, IGPM report 241, RWTH Aachen, 2004.
[30] S. B. Pillapakkam, P. Singh, *A level-set method for computing solutions to viscoelastic two-phase flow*, J. Comput. Phys. 174, pp. 552–578, 2001.
[31] R. Rannacher, *Finite Element Methods for the Incompressible Navier-Stokes Equations*, in: Fundamental directions in mathematical fluid mechanics (G.-P. Galdi et al., eds.), pp. 191–293, Birkhäuser, Basel, 2000.

[32] R. Rannacher, *On the numerical solution of the incompressible Navier-Stokes equations*, ZAMM 73, pp. 203–216, 1993 (V. C. Boffy and H. Neunzert, eds.), pp. 34–53, Teubner, Stuttgart, 1998.

[33] H.-G. Roos, M. Stynes, L. Tobiska, *Numerical methods for singularly perturbed differential equations: convection diffusion and flow problems*, Springer ser. in comp. math. Vol 24, Springer, Berlin, Heidelberg, 1996.

[34] J. A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci. 93, 1591, 1996.

[35] J. A. Sethian, *Theory, algorithms, and applications of level set methods for propagating interfaces*, In: Acta Numerica, Vol. 5, pp. 309–395, 1996.

[36] J. A. Sethian, *Level set methods and fast marching methods*, Cambridge University Press, 1999.

[37] SFB 540, *"Model-based Experimental Analysis of Kinetic Phenomena in Fluid Multi-phase Reactive Systems"*, `http://www.lfpt.rwth-aachen.de/SFB540/`

[38] MG, A parallel multilevel platform for unstructured grids, `http://rcswww.urz.tu-dresden.de/~jstiller/projects/mg/`

[39] M. Sussman, P. Smereka, S. Osher, *A level set approach for computing solutions to incompressible two-phase flow*, J. Comp. Phys. 114, pp. 146–159, 1994.

[40] M. Sussman, A. S. Almgren, J. B. Bell, Ph. Colella, L. H. Howell, M. L. Welcome, *An adaptive level set approach for incompressible two-phase flows*, J. Comp. Phys. 148, pp. 81–124, 1999.

[41] A.-K. Tornberg, *Interface tracking methods with application to multiphase flows*, Doctoral Thesis, Royal Institute of Technology, Department of Numerical Analysis and Computing Science, Stockholm, 2000

[42] A.-K. Tornberg, B. Engquist, *A finite element based level-set method for multiphase flow applications*, Comput. Visual. Sci. 3, pp. 93–101, 2000.

[43] S. Turek, *Efficient solvers for incompressible flow problems: An algorithmic approach in view of computational aspects*, LNCSE Vol 6, Springer, Berlin, Heidelberg, 1999.

[44] UG, `http://cox.iwr.uni-heidelberg.de/~ug/`

[45] S. Zhang, *Successive subdivisions of tetrahedra and multigrid methods on tetrahedral meshes*, Houston J. Math. 21, pp. 541–556, 1995.