Numerical Simulation of Droplets with Dynamic Contact Angles

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch–Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich–Wilhelms–Universität Bonn

vorgelegt von

Margrit Klitz

aus

Husum

Bonn 2014

Angefertigt mit Genehmigung der Mathematisch–Naturwissenschaftlichen Fakultät der Rheinischen Friedrich–Wilhelms–Universität Bonn

1. Gutachter: Prof. Dr. Michael Griebel

2. Gutachter: Prof. Dr. Marc Alexander Schweitzer

Tag der Promotion: 18.12.2014

Erscheinungsjahr: 2015

Summary

The numerical simulation of droplet impact is of interest for a vast variety of industrial processes, where practical experiments are costly and time-consuming. In these simulations, the dynamic contact angle is a key parameter, but the modeling of its behavior is poorly understood so far.

One of the few models, which considers the overall physical context of the involved ‘moving contact line problem’ is Shikhmurzaev’s interface formation model [Shi07]. In addition to keeping the problem well-posed, all surface and bulk parameters, such as the contact angle, are determined as part of the solution rather than being prescribed functions of contact line speed. Furthermore, this model is able to describe a large variety of other flows with singularities including coalescence, break-up and formation of cusps.

In this thesis, we couple an asymptotic version of the interface formation model with our three-dimensional incompressible two-phase Navier-Stokes solver NaSt3DGPF developed at the Institute for Numerical Simulation, Bonn University. This model has never been implemented in a three-dimensional solver before. Additionally, we employ a model by Yokoi et al. [YVHH09] based on the conventional Tanner’s law [Tan79] and a constant contact angle approach for comparative simulations of droplet impact on substrates with a variety of wetting characteristics. With the sophisticated asymptotic interface formation model, the droplet shapes, heights and diameters compare very well with those from a range of practical experiments. Of all three approaches, this model proves to be invaluable due to its accurate contact angle predictions and its general applicability.

For our flow solver NaSt3DGPF, we employ a standard finite difference discretization on uniform Cartesian staggered grids and use Chorin’s projection approach. The free surface between the two fluid phases is tracked with the level-set approach. Here, the interface conditions are implicitly incorporated into the momentum equations by the Continuum Surface Force method [BKZ92]. Surface tension is evaluated using a smoothed delta function and third order interpolation. The parallelization of the code is based on conventional domain decomposition techniques using MPI. This allows us to deal with reasonably fine mesh resolutions in three dimensions.

For dynamic wetting simulations and droplet impact in particular, the contact angle needs to determine the shape of the level-set function at the boundary. Therefore, we present a straightforward yet innovative approach to include dynamic contact angle models into three-dimensional flow solvers where the dynamic contact angle provides a Neumann boundary condition for the level-set function. In addition, dynamic wetting simulations require excellent mass conservation properties of our flow solver. Our in-house flow solver NaSt3DGPF employs the level-set method; a method rather renowned for its inability to conserve mass. In a first step, we therefore apply a local [SF99] and a global volume correction [Cro10] to remedy the loss of mass. We compare both methods, study their mass convergence properties and discuss their numerical complexity. However, both of these methods yield unphysical results even for the most simple advection tests.

Due to the failure of the volume correction methods, we implement the complex Coupled Level-Set and Volume-Of-Fluid (CLSVOF) method [MTB07, SP00, Son03], which is faster than our previous higher-order level-set method and conserves mass excellently even on coarse grids. In our description of the CLSVOF method, we include all details of the implementation. Thus, our section on this method can be used as a comprehensive guide for an implementation into existing level-set Navier-Stokes solvers, which has not been available so far. Additionally, we present a new technique for the reinitialization of the level-set function within the CLSVOF method. Furthermore, we also address the details of parallelization, which is neglected in the standard literature. The CLSVOF method will prove to be mass conserving, computationally efficient and highly reliable.

All numerical methods are thoroughly validated. Starting with basic advection tests in two and three dimensions, we then include the whole flow solver for the rising bubble benchmark. Finally, our methods are also tested for droplets impacting on substrates with various wetting characteristics.

To summarize, the implementation of Shikhmurzaev's asymptotic interface formation model combined with the CLSVOF method for the treatment of the free surface yields an extraordinary tool for the simulation of droplet impact in three spatial dimensions, which we put to the test for a large range of droplet impaction experiments.

Contents

1. Introduction

The numerical simulation of dynamic wetting processes is of critical importance for a number of industrial applications such as coating, lamination, lubrication or oil-recovery. In particular, many wetting processes involve droplets impacting and spreading on solid surfaces. These range from spray painting or coating and ink-jet printing to the delivery of agricultural chemicals and the coating of tablets in the pharmaceutical industry [Bol11]. Spray painting, for example, is a technique where a distribution of droplets is sprayed through the air onto a surface. Thereby, protective coatings as well as decorative finishes are applied to manufactured products. A simulation example of a water droplet impacting on a solid substrate is shown in Figure 1.1 and is animated in the flip-book starting at the bottom of this page.

The quality of the wetting highly controls the quality of the industrial end products and, therefore, needs to be optimized to reduce wetting defects and instabilities such as air entrainment or ribbing. In general, numerical simulation and optimization is a cheaper alternative to the traditional time-consuming adjustment of machines and the expensive waste of raw materials. A key parameter for the simulation of dynamic wetting and in particular of droplet impact is the contact angle θ: All wetting applications have in common that liquid comes into contact with a solid surface and that a liquid-gas phase boundary is in motion. Thereby, a moving contact line is produced along the substrate, which is the line where the air is displaced by the liquid; see Figure 1.2. Then, θ is measured through the liquid at the contact line and quantifies the wettability of the substrate. Due to the contact line motion, θ becomes dynamic as well and can change considerably from its well-understood static value.

Despite the industrial interest in the numerical simulation of dynamic wetting processes, the existing theoretical and numerical approaches so far often fail to correctly predict the results of practical experiments. This is due to two fundamental challenges which constitute the so-called 'moving contact line problem': First, within the classical theory of continuum fluid mechanics (the Navier-Stokes equations with the no-slip condition for the velocity), the mathematical problem describing the process of displacement of one fluid by another from a solid surface possesses no solution in the physically relevant class of

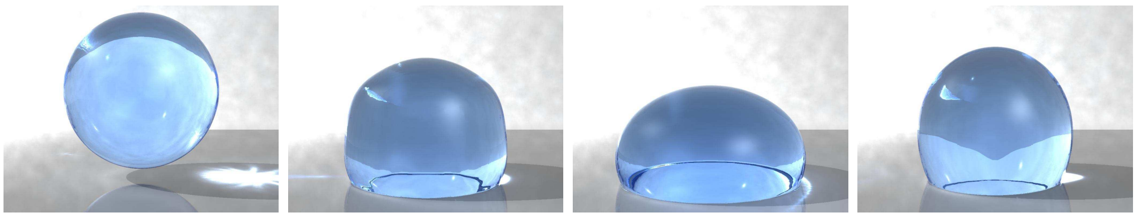
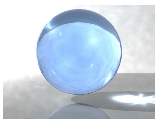


Figure 1.1.: Simulation of dynamic wetting with a water droplet.



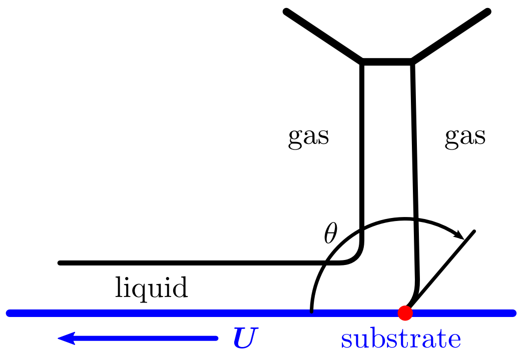


Figure 1.2.: The dynamic contact angle problem for curtain coating and dip coating processes. Here, $\boldsymbol{U}$ denotes the velocity of the moving substrate and θ is the contact angle. The red dot marks the contact line.

smooth bounded functions [Shi07].[1] Second, the modeling of the dynamic contact angle, which determines the shape of the free surface at the contact line, is poorly understood so far [Bla06, BEI$^+$09, Shi07]. Both sides of the problem indicate that some essential physical mechanism is not covered by the standard mathematical models. Therefore, an extended model is required, which predicts a smooth bounded solution of the moving contact line problem and prescribes the contact angle as part of the model and not separate from it.

So far, most mathematical models are unable to describe the contact angle and flow behavior as observed in practical experiments. For instance, many of them allow the velocity to slip at the contact line, while the contact angle is found empirically. An example of these models is the approach by Yokoi et al. [YVHH09]. One of the few models, which considers the overall physical context of the moving contact line problem, is Shikhmurzaev's interface formation model [Shi07]. In addition to keeping the problem well-posed, all surface and bulk parameters, such as the contact angle, are determined as part of the solution rather than being prescribed functions of contact line speed. Furthermore, this model is able to describe a large variety of other flows with singularities including coalescence, break-up and formation of cusps. So far, the full interface formation model has only been implemented in a two-dimensional finite element framework [SS13] to study axisymmetric drop impact [SS12b] and coalescence [SS12a]. Due to its complexity, there is no implementation in three dimensions yet.

In addition to the modeling side of the moving contact line problem, the numerical challenges for the simulation of impacting droplets are at least two-fold. First, we need to couple the discretized contact angle model to a two-phase Navier-Stokes solver. For instance, this can be achieved by prescribing the contact angle, which determines the shape of the free surface at the contact line. Second, the numerical scheme for the treatment of the free surface has to be mass-conserving. Otherwise, the comparison of numerically evaluated droplet diameters with those from experiments is useless. In Figure 1.3, a typical droplet impact simulation is shown, where after about $t = 0.015\,\mathrm{s}$ the droplet simply vanishes.

In this thesis, we couple an asymptotic version of the interface formation model with our

[1] Historically, the moving contact line problem was analyzed in a slightly simplified manner, which lead to a non-integrable shear stress singularity at the moving contact line [HS71].

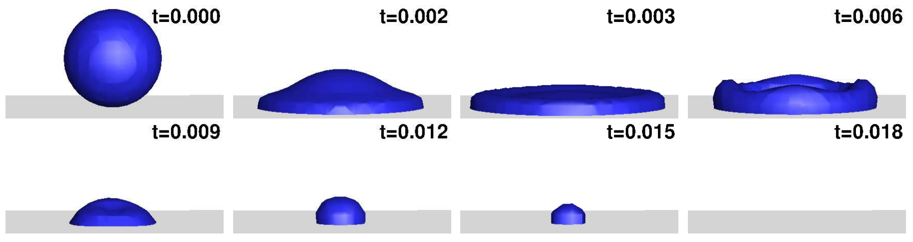
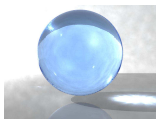


Figure 1.3.: Typical droplet impact simulation demonstrating the mass loss of the standard level-set method over time t.

three-dimensional incompressible two-phase Navier-Stokes solver NaSt3DGPF [GDN98, NaS] combined with the Coupled Level-Set and Volume-Of-Fluid (CLSVOF) method for mass conservation. For our flow solver, we employ a standard finite difference discretization on uniform Cartesian staggered grids and use Chorin's projection approach. The interface conditions are implicitly incorporated into the momentum equations by the Continuum Surface Force (CSF) method [BKZ92]. Surface tension is evaluated using a smoothed delta function and third order interpolation. The parallelization of the code is based on conventional domain decomposition techniques using Message Passing Interface (MPI). This allows us to deal with reasonably fine mesh resolutions in three dimensions.

The contributions of this thesis are as follows:

- We couple the approach by Yokoi et al. [YVHH09] as well as an asymptotic version of Shikhmurzaev's interface formation model with our three-dimensional incompressible two-phase Navier-Stokes solver NaSt3DGPF. Due to its origin in the complex full interface formation model, the asymptotic approach has never before been implemented in a three-dimensional Navier-Stokes solver and has been scarcely studied so far [Dec06, MW04, SSK12]. With this sophisticated model, the droplet shapes and diameters in impact simulations compare very well with those from a range of practical experiments.
- For dynamic wetting simulations with the above models, the contact angle needs to determine the shape of the level-set function at the boundary. Therefore, we present a straightforward yet innovative way to include dynamic contact angle models into three-dimensional flow solvers where the dynamic contact angle provides a Neumann boundary condition for the level-set function. To our knowledge, this condition has not been used in conjunction with the level-set method so far. We prove the consistency of the boundary condition in Proposition 3.1 of this thesis and show its connection to the velocity extension method proposed by Sussman [Sus01].
- In addition, dynamic wetting simulations require excellent mass conservation properties of our flow solver. Our in-house flow solver NaSt3DGPF employs the level-set method; a method rather renowned for its inability to conserve mass. In a first step, we therefore apply a local [SF99] and a global volume correction [Cro10] to remedy the loss of mass. We compare both methods, study their mass convergence properties and discuss their numerical complexity. However, both of these methods yield unphysical results even for the most simple advection tests.
- Due to the failure of the volume correction methods, we implement the complex



CLSVOF method [MTB07, SP00, Son03], which is faster than our previous higher-order level-set method and conserves mass excellently even on coarse grids. In our description of the CLSVOF method, we include all details of the implementation. Thus, our section on this method can be used as a comprehensive guide for an implementation into existing level-set Navier-Stokes solvers, which has not been available so far. Additionally, we present a new technique for the reinitialization of the level-set function within the CLSVOF method and show that an existing approach by Wang et al. [WYKS09] fails in three dimensions. Furthermore, we also address the details of parallelization, which is neglected in the standard literature.

- The pure level-set method, the level-set method with both volume corrections, and the CLSVOF method are thoroughly validated. Starting with basic advection tests in two and three dimensions, we then include the whole flow solver for the rising bubble benchmark. Finally, all of these methods are also tested for a droplet impact example. All methods are judged by their accuracy, mass convergence and by their computational complexity. For the convergence of mass, we obtain a convergence order of 1.5–2 for the CLSVOF and an order of 0.5–2 for the level-set method, depending on the problem.
- For the first time, we present simulation results with the reduced interface formation model coupled to a three-dimensional Navier-Stokes solver. This combination yields excellent results for a range of droplet impact simulations compared with practical experiments. Note that Yokoi's approach cannot be easily adapted for such a large number of experiments since it requires further knowledge about experimental parameters which are not always available.

The remainder of this thesis is organized as follows:

In the second chapter, we deal with the state of the art of the solution of the moving contact line problem, of numerical methods for interface localization and of the coupling between contact angle models and the two-phase Navier-Stokes equations, and we place the models used in this thesis into this wide context.

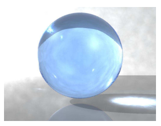
In the third chapter, we discuss the mathematical details of our approach. We derive the standard interface conditions for the Navier-Stokes equations and shortly discuss how these are implicitly incorporated into the momentum equations by the CSF method [BKZ92]. Here, we also present the mathematical details of the level-set method implemented in NaSt3DGPF. We then explain how the contact angle can be included as a boundary condition for the level-set function and prove the validity of this condition. Furthermore, we show the connection of our method and Sussman's velocity extension approach [Sus01]. Then, we describe the mathematical details of the approach by Yokoi [YVHH09] and of the contact angle model by Shikhmurzaev [Shi07]. For the sake of completeness, we also give an insight into the full interface formation model by Shikhmurzaev.

In the fourth chapter, we address our numerical model. Here, we describe the discretization of our two-phase level-set Navier-Stokes solver NaSt3DGPF in space and time. Additionally, we focus on the implementation of the contact angle boundary condition and present the local and global volume correction to improve the mass conservation properties of our level-set approach. The discretization details of the reduced interface formation model and Yokoi's approach are discussed as well. Last, we present the parallelization of our flow solver with special emphasis on its latest additions, i.e. the local volume correction, the contact angle boundary condition and both contact angle models.

Due to its complexity, the whole fifth chapter is dedicated to the CLSVOF method. We describe the required geometric reconstruction of the interface, the discretization in time by operator splitting and the discretization in space. Especially, we focus on those details of the CLSVOF method which are left out in most articles so far but are necessary for a thorough understanding and a fast and reliable implementation of the method. Thus, we explain how the reduction of the number of interface configurations works and present a new algorithm for the reinitialization of the level-set function within the CLSVOF approach. Additionally, we discuss the parallelization of the CLSVOF method.

Numerical results are presented in the sixth chapter. In its first part, the focus is on benchmark experiments which do not require the computation of dynamic contact angles. Here, we consider 2D and 3D vortex shearing and the rise of bubbles with high density jumps. We compare the numerical results of the CLSVOF method with those of the level-set method and of the two volume correction methods. After the validation of our numerical models, we simulate the droplet impact behavior on dry surfaces and compare the evolving droplet shapes and diameters to those obtained from practical experiments. These numerical results rely heavily on the contact models implemented into our flow solver. In a first step, we consider the impact of water droplets in a purely qualitative scenario where we choose the dynamic contact angle to be constant. Second, we simulate the impact of water on silicon rubber with Shikhmurzaev's asymptotic interface formation model and with Yokoi's approach. Note that this is the specific experiment Yokoi's approach has been designed for [YVHH09]. Third, we simulate micron scale drop impact of water with the asymptotic interface formation model only. Fourth, we compare the impact of a micron drop with the impact of a millimeter drop to study in how far the numerical simulation is able to capture the effect of drop size on the drop impaction process as observed in the experiments [DCBM07].

Finally, we give some concluding remarks in the seventh chapter.



2. State of the Art

This chapter presents the state of the art of the moving contact line problem, of the numerical methods for interface localization, and of the coupling between contact angle models and the two-phase Navier-Stokes equations.

2.1. The moving contact line problem

In this section, we describe the moving contact line problem and some of the current mathematical models for its solution. In this discussion, we include the approach by Yokoi et al. [YVHH09] and the asymptotic interface formation model by Shikhmurzaev [Shi07], which we both use in this thesis.

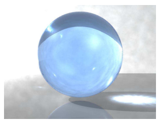
Before we address the moving contact line problem, we have to shortly address the physical reliability of the Navier-Stokes equations on their own. Partly, this reliability refers to the fundamental problem of the existence of a smooth bounded solution with an arbitrary initial solenoid velocity field as initial condition, which has been made one of the seven Millennium Prize problems in mathematics by the Clay Mathematics Institute [Fef00]. In their problem statement, a solution is accepted as 'physically reasonable' only if it satisfies $\mathbf{u}(\boldsymbol{x},t) \in \left[C^\infty(\mathbb{R}^3 \times [0,\infty))\right]^3$ for the 3D velocity and $p(\boldsymbol{x},t) \in C^\infty(\mathbb{R}^3 \times [0,\infty))$ for the pressure. Additionally, there must exist a constant $C \in (0,\infty)$ such that $\int_{\mathbb{R}^3} |\mathbf{u}(\boldsymbol{x},t)|^2 dx < C$ for all $t \geq 0$.[1] Note that in 2013 Otelbaev proposed a solution but at least one serious flaw was found in his proof [Mos14].

So far, the problem has not been solved and remains a subject of substantial research [Wik03a]. Partial results concern the 2D Navier-Stokes equations, which possess smooth and globally defined solutions [LS69]. Furthermore, in 3D, there exists a finite 'blow-up' time T, depending on the initial velocity field, such that the Navier-Stokes equations on $\mathbb{R}^3 \times (0,T)$ have smooth solutions $\mathbf{u}(\boldsymbol{x},t)$ and $p(\boldsymbol{x},t)$. Additionally, if the initial velocity field is sufficiently small, smooth and globally defined, solutions can be found [Fef00]. The existence of weak solutions for the Navier-Stokes equations in three spatial dimensions has been shown by Leray [Ler34], but it is not known whether these are unique.

Despite these unresolved issues, there are "no experiments indicating discrepancies between what the Navier-Stokes equations predict for unsteady flows and what is actually measured" [Shi07, p. 89]. Of course, this argument is purely empiric in quality. Therefore, in the following, we can only postulate that the Navier-Stokes equations, and the two-phase Navier-Stokes equations in particular, remain physically reliable for the problems considered in this thesis.

Let us come back to the moving contact line problem, which describes the failure of classical fluid mechanics to describe processes of liquid-liquid or liquid-gas displacement on

[1]Naturally, the problem is also considered to be solved if a counterexample for the existence of such solutions can be found.



solid surfaces. As for the Navier-Stokes equations in the millennium prize problem, we consider a solution to be physically adequate if the flow parameters remain finite everywhere, and we must reject or challenge those solutions where they become infinite.

Thus, in [Shi07], Shikhmurzaev shows that there is no solution to the moving contact line problem within the standard theory, if we assume no-slip at the boundary and describe the fluid flow with the Navier-Stokes equations.[2] This failure can be exemplified for the spreading of a viscous liquid over a smooth solid surface which displaces an inviscid gas. Here, the bulk flow is described by the Stokes equations subject to the no-slip condition on the solid boundary and the usual conditions of zero normal velocity in the frame moving with the contact line. Furthermore, zero tangential stress and balance of normal stress and capillary pressure hold at the free surface. Additionally, the contact angle has to be prescribed since it is independent of the flow field within this formulation. For this simple problem, Shikhmurzaev finds that the solution does not exist in the class C^∞ of smooth bounded functions. Qualitatively in this case, the balance of capillary pressure and normal stress can never be fulfilled since the no-slip condition removes the liquid in the vicinity of the contact line. This removal causes a pressure difference between the constant pressure in the gas and the reduced pressure in the liquid, which the capillary pressure simply cannot compensate.

Historically, the moving contact line problem was known as the non-integrability of shear stress, which arises if the assumptions by Shikhmurzaev are slightly altered. Thus, in the work of Huh and Scriven [HS71], the troublesome interface condition of normal stress and capillary pressure balance is replaced by a prescription of the free surface shape. Then, although a solution can be found in this case, a non-integrable $1/r$ singularity occurs in the stress suggesting that the tangential force acting on the solid adjacent to the contact line becomes infinite. This means that we would need an infinite force to immerse a solid into a liquid, which is clearly unphysical and does not correspond to any experimental evidence. Or, as Huh and Scriven [HS71] write: “not even Herakles could sink a solid if the physical model [no slip] were entirely valid, which it is not.” Note that the arising infinite tangential stress can also be interpreted as a multivalued velocity at the contact line [DD74].

If we speak of the ‘removal’ of the shear stress singularity, this refers to the influence of the boundary conditions on the global solutions of the ordinary differential equation obtained when a separable stream function is substituted in the Stokes equations. With no-slip at the wall, one obtains the $1/r$ singularity of the shear stress, while in the case of Navier-slip (which ‘removes’ the singularity, see below) only an integrable logarithmic singularity in the pressure remains, which is “not as worrying” [KNP$^+$13] .

However, as Shikhmurzaev [Shi07] stresses, the non-integrability of the shear stress and multivaluedness of the velocity field is found in Huh’s and Scriven’s simplified analysis of the moving contact line problem (where the standard interfacial balance of capillary pressure and normal stress is dropped). In the general formulation analyzed by Shikhmurzaev,

[2]Note that the no-slip condition is almost always used in the modeling of viscous flows and, therefore, presents a natural choice for many wetting problems. Instead, a slip boundary conditions *allows* for a solution. However, such a boundary condition can still not repair the other side of the moving contact line problem, which is the modeling of the dynamic contact angle. Furthermore, a mere slip condition is unable to account for the vast variety of wetting phenomena observed in experiments; see our discussion on ‘slip models’ below.

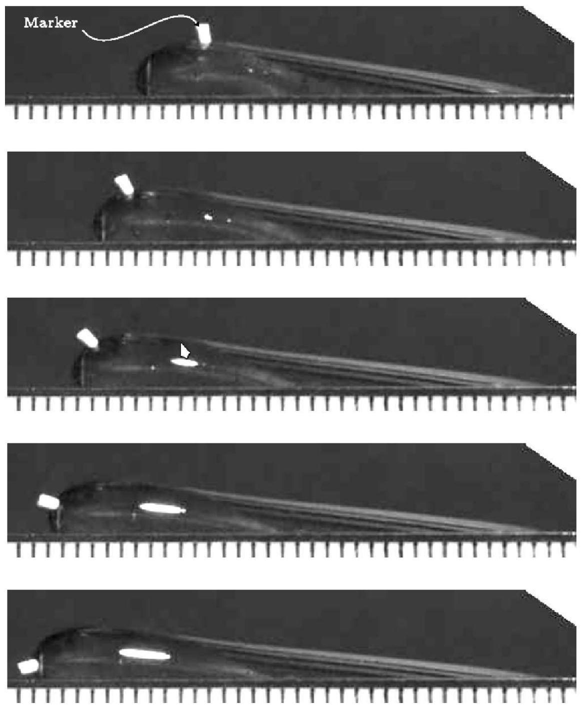


Figure 2.1: A marker particle is placed on the liquid which demonstrates the rolling motion of the interface. With permission from C. Chiquete [Chi07].

the solution is nonexistent in the physically relevant class of smooth bounded functions. Therefore, instead of 'removing' the non-integrability, one should rather find the missing physics in the classical formulation and appropriately incorporate them into the model.

2.1.1. Observations from dynamic wetting experiments

There are many experiments which are relevant to the moving contact line problem, of which a comprehensive overview is given in [Shi07]. Here, we only focus on those results, which are observed in dynamic wetting experiments, where a gas is displaced by a liquid from a solid surface. Note that this is also the case relevant for droplet impaction.

The first very interesting result from dynamic wetting experiments is that the motion in the vicinity of the contact line is 'rolling', which is more akin to a rigid body than a fluid. This means that material fluid points that make up the interface between liquid and gas reach the contact line in finite time. Subsequently, these material points become part of the solid-fluid interface, which was originally observed by Dussan and Davis [DD74] and is exemplified in an experiment conducted by Chiquete [Chi07], whose results we reproduce in Figure 2.1.

Second, instead of being a stagnation point, the region near the contact line is a region of intense flow, with larger rates of strain than in the bulk of the fluid. This was observed in an experiment by Clarke [Cla95], who employed a flow visualization and particle tracking velocimetry to a curtain coating experiment (Fig. 1.2). Then, the resulting images were analyzed to obtain the rate of strain and vorticity of the flow field.

Third, the contact angle depends on the flow rate and not only on the contact line speed: Instead, even for the same contact line speed, the dynamic contact angle can be changed by changing the flow field or the geometry near the contact line. For example, in the curtain coating context, Blake et al. [BBS99] have shown that for a fixed substrate speed, the observed contact angle varies with the flow rate and curtain height so that air entrainment can be delayed to higher coating speeds. This effect has been termed the *hydrodynamic assist of wetting*.

2.1.2. State of the art

In this section, we discuss models which have been proposed to resolve the moving contact line problem. Due to the large variety of models which range from molecular dynamics or continuum models with intermolecular forces to diffuse interface models and slip models, there are already a number of reviews which discuss the different approaches and their specific subtleties. In [Bla06], Blake gives a comprehensive description of the different approaches and how these models compare with experiments and simulation. Furthermore, in [BEI$^+$09], Bonn et al. describe the recent theoretical, experimental, and numerical progresses in the description of moving contact line dynamics. Additionally, the special topic of dynamic wetting on rough surfaces is reviewed.

In the following, we mainly follow the monograph [Shi07, pp. 145–173], which describes the different models dealing with the moving contact line problem in detail. This work is especially interesting since Shikhmurzaev tests all the models against his ‘ABC’-criterion, which all reasonable models should fulfill: Criterion A checks if the model is mathematically well-defined. Criterion B tests if the solution remains finite, i.e. if there are no singularities in the velocity or pressure. Finally, criterion C makes sure that the solution is at least qualitatively able to produce the same results as observed in experiments.

Slip models

Most moving contact line models so far neither focus on finding the missing physics in the classical formulation nor do they focus on the development of a comprehensive self-consistent physical model, which deals with the problem as a whole. In the simplified setting, the key to the moving contact line problem is at least twofold: On the one hand, the stress singularity has to be removed and, on the other hand, the contact-angle behavior has to be modeled accurately since it defines the shape of the free surface and thereby has a large effect on the whole flow field. In the framework of the so-called slip models both problems are addressed independently: First, the no-slip condition for the velocity is relaxed and the fluid is allowed to slip at the contact line [ES04, GH91, Hoc77, SB00], which eliminates the stress singularity. Then, the contact angle is modeled separately.

Slip models form a very large and popular group since they seemingly offer the most natural approach for the removal of the stress singularity. Thus, all of them share the key idea of a new length scale l_s which is the so-called slip length of the liquid-solid interface at the contact line (Fig. 2.2).

A very common variation of the slip condition is the so-called Navier-slip condition [Nav27], which states that the slip length is the proportionality factor which relates the slip velocity at the liquid-solid interface to the tangential stress acting on it. The slip

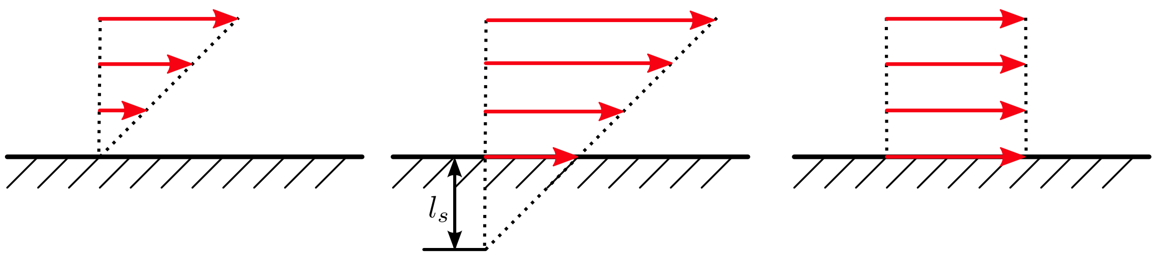


Figure 2.2.: Schematic representation of the slip length l_s with $l_s = 0$ on the left (no-slip), $0 < l_s < \infty$ in the middle (partial slip) and $l_s = \infty$ on the right (perfect slip). For partial or Navier-slip, the slip length is defined as the distance from the liquid-solid interface to the vanishing point of the linear extrapolation of the tangential velocity. The slip length is the proportionality factor which relates the slip velocity $\boldsymbol{v}_{\text{slip}}$ at the liquid-solid interface to the shear rate.

velocity $\boldsymbol{v}_{\text{slip}}$ is the projection of the fluid velocity $\boldsymbol{u}$ onto the tangent of the surface, i.e.

$$\boldsymbol{v}_{\text{slip}} = \boldsymbol{u} \cdot (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T})$$

with the outward geometry normal $\boldsymbol{n}$. Then, the Navier-slip condition reads

$$\boldsymbol{v}_{\text{slip}} = \boldsymbol{u} \cdot (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T}) = l_s \boldsymbol{n}^\mathsf{T} \cdot \boldsymbol{T} \cdot (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T}), \tag{2.1}$$

with the stress tensor is $\boldsymbol{T} = -p\boldsymbol{I} + \mu\boldsymbol{S}$, the rate of strain tensor $\boldsymbol{S} = \nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^\mathsf{T}$ and the dynamic viscosity μ.[3] This boundary condition removes the singularity at the contact line, and many of the slip models only differ in their definition of the slip length l_s [Shi07, pp. 147–150].

In addition, the contact angle θ_d has to be modeled. The simplest possible assumption is that of a constant 'dynamic' contact angle[4] equal to the equilibrium contact angle θ_e [Hoc77], i.e.

$$\theta_d = \theta_e.$$

Or, θ_d is chosen as a function

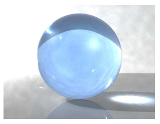
$$\theta_d = f(Ca, \theta_e, k_1, k_2, \ldots) \tag{2.2}$$

of the capillary number Ca, the equilibrium angle θ_e, and material-related parameters k_i, which are used to fit the numerical results to experiments [ES04, GH91]. Here, the capillary number is defined as $Ca = \mu u_{\text{cl}}/\sigma$ with the viscosity μ, the contact line velocity u_{cl}, and the surface tension σ. Thereby, θ_d only depends on the speed of the contact line if all the other parameters are assumed to be constant.

A fundamental difficulty with these models is, for example, that they fail to describe the dependency of the contact angle on the flow rate which is, however, observed in experiments (cf. Section 2.1.1). Thus, the dynamic contact angle cannot simply be function of the

[3]This is the tangential projection of the more general formulation $\boldsymbol{u} = l_s \boldsymbol{n}^\mathsf{T} \cdot \boldsymbol{T} \cdot (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T})$, which relates the fluid velocity to the tangential stresses acting from the liquid on the liquid-solid interface.

[4]This approach is also employed in Chapter 6 for the purpose of comparison with more sophisticated models.



contact line speed and the material properties of the contacting media as assumed in equation (2.2). Instead, even for the same contact line speed, the dynamic contact angle can be changed by changing the flow field or the geometry near the contact line. This hydrodynamic assist of wetting cannot be described by an equation such as (2.2), where the contact angle depends foremost on the speed of the contact line. Additionally, the slip models predict a stagnation point at the contact line and are therefore unable to reproduce the rolling motion of the contact line seen in experiments.

Homogenization of surface roughness

Another similar approach is proposed by Glasner [Gla06], who uses homogenization to study the effect of surface roughness on the contact line. Unfortunately, his analysis is restricted to the small angle limit, which is irrelevant for most systems such as drop impact dynamics. Furthermore, a contact angle formula similar to (2.2) is assumed to hold, which makes his approach as equally unable to describe the hydrodynamic assist of wetting as the slip models discussed above.

Thin film models

In the theory of thin films, the moving contact line problem is circumvented by assuming that the surface is already covered by a thin film of fluid [Tan79]. Then, with a scaled lubrication approximation, one can derive Tanner's law

$$U = A\theta^3, \tag{2.3}$$

where θ is now an apparent contact angle, which can be defined anywhere on the free surface. Here, U is a dimensionful velocity and A depends on the fluid properties. This law is often used for the modeling of the contact angle behavior in the framework of slip models. Then, it is also used as a variant of equation (2.2) and becomes

$$Ca = k(\theta_d - \theta_e)^3. \tag{2.4}$$

This use of Tanner's law is conceptually questionable [Shi07, p. 165] since there is no actual contact angle involved in the derivation of Tanner's law in the first place. In this thesis, we test a related type of slip model which has been improved and extended by Yokoi et al. [YVHH09].

Diffuse interface models

The phase-field or diffuse-interface models are more commonly known for their ability to capture the interface for the simulation of two-phase flow problems; see Section 2.2.1. However, they also tackle the moving contact line problem in their own right [Sep96]. In these models the sharp fluid interface is replaced by a thin transition region. Thus, the boundary conditions at the interface are replaced by equations for the evolution of an auxiliary conserved order parameter (the phase field), which takes a distinct uniform value in each of the phases and is smoothly distributed in the transition region. Then, the phase field, such as mass concentration, can be described by the Cahn-Hilliard equation.

The Cahn-Hilliard-van der Waals model originally suggested by Seppecher [Sep96] constitutes a solution to the moving contact line problem while the no-slip boundary condition is retained. In this case the contact line can move through diffusion rather than convection [QWS06], or, as Shikhmurzaev [Shi07] puts it, there is no contact line in the first place since the receding gas is allowed to flow into the advancing liquid near the contact line.

Moreover, the thickness of the interfacial layer exists on a macroscopic, not molecular length scale, which Shikhmurzaev [Shi07, p. 168] finds to be fundamentally flawed when applied to immiscible fluids. Thus, the thickness of the interface should be determined by its underlying physics, i.e. by the length scale associated with molecular motion. Such an interface, however, would always become a mathematical surface of zero thickness when the continuum approximation is taken.

Navier-slip conditions motivated by molecular dynamics simulations

From molecular dynamics (MD) studies and analysis Qian et al. [QWS06] propose a generalized Navier boundary condition which relates the tangential slip velocity to the sum of tangential viscous and uncompensated Young stress. This uncompensated Young stress arises only in the dynamic case, when the fluid-gas interface deviates from its static configuration and is the main addition to the classical Navier boundary condition (2.1). Further away from the moving contact line, the Young stress vanishes and the classical condition is retained.

Similiarly, Ren and E [WW07] propose a generalized Navier condition, which is also derived from force balances and contains the uncompensated Young stress. Contrary to Qian et al. [QWS06], they focus on a sharp interface model and do not resort to a phase-field approach. The main differences between both models are the exact location of the definition of the contact angle (at the upper boundary of the fluid-solid interfacial region), the definition of the friction force (an effective one which contains the friction force from the solid as well as the normal stress jump in the fluid-solid interfacial region) and the resolution of the interfacial region (a sharp region).

Both models have in common that the Navier-slip boundary condition is motivated by MD simulations, and that this Navier boundary condition becomes part of a continuum model along with the Navier-Stokes equations and the usual interface conditions.

However, by using MD simulations in a continuum model they are using two different frameworks which together are not self-consistent. Also, there is no universal rule for the relationship between the computational data obtained by MD simulations and macroscopic quantities [Shi11].

As an example of the incoherency between MD and continuum models, let us look at the near perfect slip, revealed by MD simulations near the contact line. According to Qian et al. [QWS06], there is slip in the vicinity of a couple of atomic distances from the contact line on the solid facing side of the liquid-solid interface. Elsewhere, the no-slip condition holds. In this study, slip occurs in about $10\,\sigma$ of the moving contact line, where σ is the length scale in the Lennard-Jones potential. Furthermore, also the work of Ren and E [WW07, p. 7] in 2007 confirms that “significant slip occurs in a region which extends about $20\,\sigma$ in the advancing fluid and $30\,\sigma$ in the receding fluid. Outside this region, slip is negligible. For Lennard-Jones argon, 1σ corresponds to 3.4×10^{-10} m [. . .]”. However, these couple of atomic distances in which slip occurs become a point as soon as we take the continuum limit [Shi07]: Then, $\frac{l}{L} \to 0$ for the length scale l associated with molecular distances and the macroscopic length scale L associated with the averaged continuum model. Essentially, from the point of view of continuum mechanics, we should have the no-slip condition everywhere.

Additionally, both MD studies measure the slip boundary condition at the bottom of the fluid-solid interfacial layer, i.e. on the solid-facing side of the liquid-solid interface, whereas the boundary condition for the bulk flow holds on the liquid-facing side of the interface, which renders this kind of motivation by MD simulations even more questionable.

Note that Shikhmurzaev is not opposed to MD studies in general. Instead, only the combination with a continuum approach is considered problematic as presented convincingly in [Shi11].

Hybrid MD-continuum models

A hybrid molecular-dynamics-continuum model is used by Hadjiconstantinou [Had99] for the simulation of the moving contact line problem. Here, MD are employed close to the walls for the capturing of molecular effects while continuum fluid mechanics are used to describe the far-field. In an intermediate region, both simulations have to be matched.

As with the phase-field method and the boundary conditions motivated by molecular dynamics simulation, we are dealing with two incompatible approaches in this case as criticized in [Shi07, Shi11]. Namely, from the point of view of the continuum limit the scale of the MD is zero. Either we have to use MD for the simulation on the whole domain or the region where MD simulations operate are a point in the continuum simulation. Since we are dealing with two fundamentally different concepts in each case, it becomes impossible to match or couple both simulations in the overlapping region.

2.1.3. The models used in this thesis

In this section, we place the two contact line models used in this thesis into the context of the models described above. Thus, the conventional contact angle model used by Yokoi [YVHH09] basically fits in with the slip models and the thin film theory, while Shikhmurzaev's interface formation model presents a self-consistent continuum approach.

Yokoi's approach

Interestingly, Shikhmurzaev concludes the review of contact angle models with the statement that only the slip models 'address the moving contact line problem as it stands and remain strictly within the methodology and conceptual framework of continuum mechanics'. As already mentioned, however, the slip models have their very own drawbacks: They are unable to reproduce the rolling motion of the contact line since they predict a stagnation point there. And the models cannot describe the hydrodynamic assist of wetting, i.e. the influence of the flow field on the dynamic contact angle.

In this thesis, we use an approach by Yokoi [YVHH09], which is a slip model combined with an extended and refined Tanner's law (2.4) for the modeling of the contact angle. Here, slip is induced by the so-called numerical slip. Due to the grid arrangement of our variables,

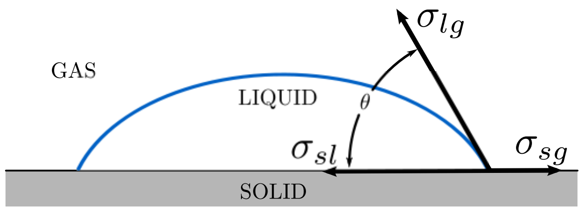


Figure 2.3: Young's force diagram: $\sigma_{sg} = \sigma_{sl} + \sigma_{lg}\cos\theta$. Here, σ_{sg} is the solid-gas, σ_{sl} the liquid-solid and σ_{lg} the liquid-gas interfacial tension.

the no-slip condition is never exactly fulfilled at the boundary, which causes enough slip to remove the singularity at the contact line. As an undesirable by-effect, the slip length becomes mesh-dependent and our numerical solution as well. Then, if our computational mesh is refined, the contact line motion slows down due to the no-slip condition, which also influences the computed contact angle. In the limit, the contact line motion would stop altogether.

Another model that we consider in this thesis is the asymptotic version of Shikhmurzaev's interface formation model, which we describe in the next section, followed by a qualitative comparison between Yokoi's and Shikhmurzaev's approaches to remedy the moving contact line problem. The mathematical details can be found in Section 3.6.

The interface formation model by Shikhmurzaev

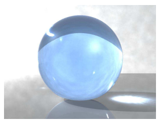
Currently, the interface formation model by Shikhmurzaev [Shi07] is the only model which is able to describe the dependence of the contact angle on the flow field, i.e. the hydrodynamic assist of wetting. Instead of an ad-hoc and separable treatment of the moving contact line problem, Shikhmurzaev considers its overall physical context in the derivation of his model for the formation and disappearance of interfaces. His continuum approach is based on non-equilibrium thermodynamics for the description of dissipation due to interface formation and thereby couples the contact angle directly to the flow field, instead of treating it as an independent quantity [Bla06].

Let us shortly state its idea: During the interface formation process fluid particles are transferred from one interface to another interface. If, for example, liquid advances across a solid surface, liquid particles are advected from the liquid-gas interface to the liquid-solid interface. Of course, both interfaces have different equilibrium properties such as surface tension, material density, or structure. Therefore, the molecules comprising the interfacial regions must undergo some reorganization when they are transferred to the new interface. This reorganization or relaxation from one state to the other takes a finite time to complete, before the fluid particles disappear from the liquid-gas interface and relax to their new equilibrium values in the liquid-solid interface, and this is what is captured by Shikhmurzaev's interface formation model.

During the relaxation process, the surface tension is disturbed from its equilibrium value and becomes a dynamic quantity. The balance of interfacial tensions at the contact line is modeled by the Young equation

$$\sigma_{sg} = \sigma_{sl} + \sigma_{lg}\cos\theta,$$

where θ is the contact angle, and σ_{sg} is the solid-gas, σ_{sl} the liquid-solid, and σ_{lg} the liquid-



gas interfacial tension; see Figure 2.3.[5] Since the surface tensions become dynamic, the contact angle θ will also attain a non-equilibrium dynamic value as observed in experiments.

The interface formation model is described in great detail in [Shi07], including an analysis of the mathematical model, its analytical solutions for steady motion at low Reynolds and capillary numbers, as well as comparisons with experiments.

Note that while the conventional slip model induces a stagnation point at the contact line, the interface formation model predicts material flux through the contact line. For example, in the process of dynamic wetting, liquid from the liquid-gas interface is transferred to the newly created liquid-solid interface. Thus, in contrast to other approaches, the interface formation model not only removes the singularity but is able to predict the experimentally observed rolling motion of the interface, as well as the dependence of the contact angle on the flow field.

Within the literature, Shikhmurzaev's interface formation model has been considered scarcely so far, which is mostly due to its complexity (see [Shi07] and the references therein and [SS13]). On the one hand, this complexity stems from the model itself. Here, lots of additional phenomenological coefficients are introduced which link the flow field to quantities such as the surface tension gradient or the shear stress at the interface, and their a-priori determination is not easy to see through [Bla06]. However, in [Shi07, p. 195–198], estimates for these parameters are given under the assumption that the thickness of the interfacial layer is about 1 nm to 3 nm – a typical length scale for most simple fluids. A further simplification occurs in the asymptotic solution at low capillary and Reynolds numbers, where most of the coefficients can be combined to a single scaling factor.

The complexity continues from a numerical point of view since the extension of conventional flow solvers to incorporate the full interface formation model is far from trivial [SS13]: We have to solve the Navier-Stokes equations for the bulk flow along with the interface formation equations at the free surface and liquid-solid interface. These equations are themselves partial differential equations and have to be complemented with boundary conditions at the contact line. By the solution of these equations, the dynamic contact angle is determined. In turn, the dynamic contact angle defines the shape of the free surface, which has a large influence on the bulk flow. Thus, as Sprittles and Shikhmurzaev [SS13] write,

> [. . .] the bulk flow, the distribution of the surface parameters along the interfaces and the dynamic contact angle that these distributions 'negotiate' become interdependent, with the dynamic contact angle being an outcome of the solution as opposed to conventional models where it is an input. This interdependency is, on the one hand, the physical essence of the experimentally observed effect of hydrodynamic assist to be described but, on the other hand, it is this very interdependency that, coupled with the nonlinearity of the bulk equations and the flow geometry, is the reason why the model is difficult to implement robustly into a numerical code.

[5] In this thesis, we use σ instead of σ_{lg} for the liquid-gas interfacial tension if there can be no confusion about which of the three tensions is meant. If we use σ, we always mean the equilibrium value and not a potentially dynamic one.

Although there have been made some advances to implement the interface formation model, the first real introduction to the model's implementation into finite element codes has only been published very recently in [SS13], including benchmark results for axisymmetric or two-dimensional coordinate systems. This work focuses on the two-dimensional numerical implementation of the interface formation model while a detailed discussion and comparison to experiments is promised to follow. The full model has also been used to study axisymmetric drop impact [SS12b] and coalescence [SS12a]. Note that this model has not yet been implemented in 3D.

Due to the complexity of the full interface formation model and its numerical implementation, we consider the asymptotic version of the interface formation model for small capillary and Reynolds numbers in this thesis, which has also not been implemented in a 3D flow solver so far. Within this approach, the flow domain is split into two asymptotic regions, and in both the limit $Ca \to 0$ of the interface formation model is studied analytically. Complemented with a slip boundary condition at the contact line, the asymptotic model can account for the influence of the flow field on the contact angle and proves to be a valuable alternative to conventional contact angle models. In two dimensions, the asymptotic approach performs very well compared to the numerical results of the full interface formation model [SS13].

A qualitative comparison

Obviously, there are fundamental differences between both, the conventional approach by Yokoi et al. and the model by Shikhmurzaev, which are considered in this thesis: The first lacks a thorough underlying mathematical theory. It is developed for and based on a single droplet impact experiment only, i.e. the impact of a droplet of distilled water on a silicon wafer onto which hydrophobic silane has been grafted. A straightforward application of this model to the numerical simulation of other wetting experiments is difficult. Rather, this model is a prescription of contact angle values which fit well with a specific practical experiment. Since the prescribed dynamic contact angle values are very close to the ones observed in the experiment, the modeling error for that specific experiment is very small. Therefore, it gives us the opportunity to test our contact angle implementation as well as our methods for mass conservation and compare them to the results of the experiment.

On the other hand, behind Shikhmurzaev's model, there is a whole theoretical framework to explain the formation and disappearance of interfaces. This model is able to describe a vast variety of dynamic wetting phenomena. However, we use the asymptotic interface formation model, which is derived for small capillary numbers only. Therefore, we expect to obtain an approximate and smoothed contact line speed–contact angle relationship compared to the practical experiments and Yokoi's results in [YVHH09]. Still, the reduced model will prove to be an excellent trade-off between the complex full interface formation model and a reasonably accurate dynamic contact angle model. Furthermore, this model is generally applicable, estimates of empirical parameters can be found in the literature [BS02], and its results do not drastically change with slight changes of these parameters [Shi07]. In Subsection 6.4.3, we perform a range of droplet impact simulations with this model only, and the numerical results compare very well with those from practical experiments.

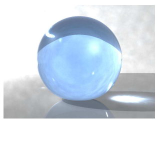


Figure 2.4.: Eulerian vs. Lagrangian point of view in fluid dynamics: The Eulerian point of view is similar to looking at the fluid while standing on the bank of the river and the Lagrangian point of view corresponds to standing in a boat, which follows the movement of the fluid.

2.2. Numerical methods for interface localization

The simulation of fluid flow with moving contact lines requires the consideration of two immiscible fluids and the treatment of the free surface between the two phases. This chapter is not intended to give a comprehensive survey of interface localization methods, which would clearly go beyond the scope of this thesis. Instead, we would like the reader to gain a certain insight into the various possible methods and place the level-set (LS) method, the volume-of-fluid (VOF) method, and the CLSVOF method into this context. The mathematical and numerical details of our LS and CLSVOF method can then be found in Chapters 3, 4, and 5.

When we looked at the different categories for interface localization methods in the standard literature, we soon found that these categories are inconsistent and even contradictory. Therefore, our last section is dedicated to a discussion of the usefulness of the prevalent categories of interface localization techniques.

2.2.1. State of the art

In the following, we do not use the standard categories of 'front tracking', 'front capturing' and 'particle' methods. Instead we present the different interface localization methods one by one, state whether they represent the interface explicitly or implicitly, and use the term 'front tracking' (or 'surface tracking') only for the special approach developed by Glimm et al. [GGG+98] and methods very closely related to that approach.

Moreover, we resort to a very natural categorization of interface localization methods and differ between Eulerian and Lagrangian approaches. In a Lagrangian method the mesh itself is moved with the flow field in space and time, while in an Eulerian method, the mesh is fixed throughout the calculation (Fig. 2.4). Additionally, there exist combinations of both methods like the Arbitrary Lagrangian Eulerian (ALE) methods.

All presented methods are judged by their applicability for the simulation of droplet impact, i.e. by their ability to handle topological changes of the interface, their mass conservation properties, and their computational complexity. However, there is no universally superior method, which is more efficient, stable, and robust than the rest. All presented techniques have their advantages and drawbacks and all of them have been successfully applied to diverse multiphase flow problems [Fra02].

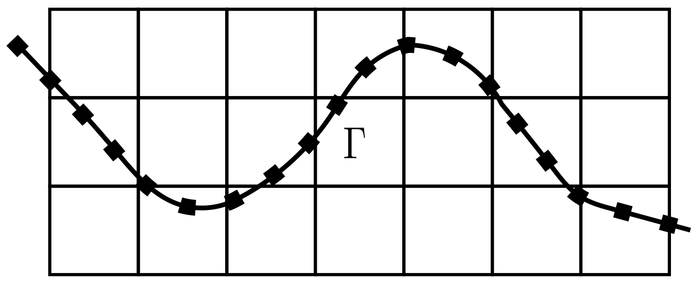


Figure 2.5.: Surface tracking: The interface Γ is represented by an ordered set of marker points. The underlying grid is fixed.

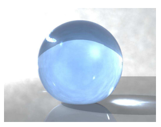
In what follows, we focus on interface localization techniques only. In general, coupled to the Navier-Stokes equations, all of the approaches can benefit from a 'suitable' spatial and temporal discretization. Especially, if additional PDEs on the interface have to be solved for the full interface formation model by Shikhmurzaev, there are various high-quality approaches in the framework of finite elements. For example, the evolving surface finite element method (ESFEM) introduced by Dziuk and Elliott [DE07] can treat conservation laws on moving surfaces. The extended finite element method (XFEM) can be used to properly approximate the viscosity, density and pressure at the interface [GR11]. Similarly, the Legendre spectral element method used in the framework of ALE methods accurately represents moving boundaries [HP90].

Surface tracking methods

In surface tracking methods, two grids are employed [GGG$^+$98]. A standard grid is used for the solution of the smooth bulk flow equations, while the discontinuities and jumps in the bulk flow variables are explicitly tracked by a lower-dimensional moving grid (the front or interface), which is specified by an ordered set of marker points located on the interface (Fig. 2.5).

In surface tracking methods [Hym84], the propagation of the interface relies on a Lagrangian approach, which is separate from the smooth solution of the Navier-Stokes equations on the fixed underlying grid. Thus, one has to derive separate lower-order differential equations for the evolution of the surface, which are linked to the underlying physical model. Furthermore, for the reconstruction of the interface in-between the marker particles, an interpolation is required, which strongly influences the accuracy of the solution.

Surface tracking methods are very accurate since a large number of grid points can be used on the interface. Thereby, the exact location and shape of the interface is tracked at all times, which makes them an ideal method for treating discontinuities in the solution. They are also able to accurately simulate shock waves or flame fronts [GGG$^+$98], where many other methods fail. Furthermore, in contrast to the use of a marker function (as for LS and CLSVOF methods), the explicit tracking of the interface reduces errors associated with advection and surface tension computation. However, one of the main disadvantages of surface tracking methods is their inability to handle topological changes of the interface automatically. For example, the break-up or merging of droplets is especially difficult to



simulate with surface tracking methods in three spatial dimensions.

Since the impact of droplets very much depends on capillary effects and the movement of the interface, surface tracking methods seem to present an interesting choice for their simulation. Thus, if the full interface formation model by Shikhmurzaev [Shi07] is considered, the physics of the interface become even more complex, which surface tracking methods are well-equipped for. Also, in three dimensions, a general purpose software package for the geometry and dynamics of an interface is publicly available [DFG$^+$06] and has been shown to be an accurate method in handling the propagation of interfaces with regions of high curvature.

However, the application of surface tracking methods for the simulation of droplet impact requires careful consideration of the trade-off between the method's complexity and its potential benefits. Thus, we have to consider the additional costs for the management of the interface data structure. Furthermore, the break-up of droplets cannot be handled straightforwardly, which is crucial for the simulation of effects such as the famous 'corona splash', where a liquid layer detaches from the impacting droplet.

Moving-mesh methods

Moving-mesh methods track the interface location explicitly by employing a mesh which moves with the interface in a Lagrangian manner. For example, the underlying mesh can be initially distributed in such a way that the edges of the local cells align with the interface location. Then, the Navier-Stokes equations are solved cell-wise and continue to track the interface. Depending on the movement of the interface, cells have to be added or deleted. However, simple moving-mesh methods suffer from a distortion of the mesh unless distinctive measures are taken to prevent this [Hym84].

Moreover, in ALE methods [BPS13], the interface mesh is moved in a Lagrangian manner with the flow velocity, while the underlying mesh is adapted and moved in such a way that mesh distortions are avoided. Then, the Navier-Stokes equations are often formulated with a relative velocity which is the difference between the actual flow velocity and the velocity of the underlying mesh; see the overview in [GR11] for more details. The ALE method suffers from two disadvantages [Mai06]: First, for the computation of the spatial derivative, the change in the underlying mesh has to be taken into account in every time step. Second, the time-derivative can no longer be simply computed by the difference of a quantity in two consecutive time steps due to the movement of the fluid domain.

For the simulation of droplet impact, the ALE methods offer considerable advantages: The free moving interface coincides with one line of the numerical grid so that the associated interface and boundary conditions can be resolved easily and accurately. However, for general three-dimensional applications involving moving contact lines, topological changes of the interface can render the re-meshing of the underlying grid very complex and very costly, and a renewed computation of the spatial derivative, e.g. in a finite element framework, requires the reassembly of the elementary matrix at each time step.

Immersed interface and immersed boundary methods

Both the immersed interface and the immersed boundary method use a simple Cartesian grid for the solution of interface problems. However, in immersed interface methods, the jump conditions across the interface are incorporated into the finite difference scheme, while the immersed boundary method uses Lagrangian control points on the interface, where force densities are computed and transferred back to the Cartesian grid by a properly chosen smoothed approximation of the Dirac delta function; see [Le05] and the references therein. Note that the immersed boundary method is a mixture between an Eulerian and Lagrangian approach since the markers which represent the immersed boundary are Lagrangian, which are embedded in a fixed Eulerian fluid domain.

The term ‘immersed boundary method’ was first used to describe the work by Peskin [Pes72], who simulated blood flow of the heart. The entire simulation was performed in an Eulerian coordinate system, while the Lagrangian variables are defined on a curvilinear mesh. Both solutions are related by interactions equations through the Dirac delta function. Immersed boundary methods are very popular for the simulation of fluid-structure-interaction problems.

Tryggvason et al. adapted the immersed boundary method (sometimes called embedded boundary method) to viscous incompressible multi-phase flow problems [TBE$^+$01, UT92].

The advantage of the Cartesian grid-based immersed boundary methods becomes most obvious when moving boundaries are involved, where a re-generation of a body-conformal grid at each time step would simply be too cumbersome [MI05]. One drawback of the immersed boundary method is the application of boundary conditions, which is not straightforward compared to traditional methods. This difficulty, however, does not rule out their application for droplet impact simulations [Fra02].

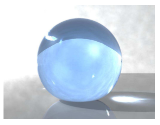
Phase field models

In phase field or diffuse-interface models the sharp fluid interface is replaced by a thin transition region [Sep96]. Thus, the boundary conditions at the interface are replaced by equations for the evolution of an auxiliary conserved order parameter (the phase field), which takes a distinct uniform value in each of the phases and is smoothly distributed in the transition region. The phase field, such as the mass concentration, is described by the Cahn-Hilliard equation.

Phase field methods are able to handle topological changes in the interface automatically and the method can be easily extended to account for further complex physics such as multicomponent systems.

One drawback of phase field methods is the computational effort which has to be invested for accurate simulations. Thus, interfacial dynamics are only captured correctly when the interface width is vanishingly small. Nevertheless, for sufficient numerical accuracy, the interfacial width must be at least a few grid lengths in size, which renders computations of accurate phase field models highly expensive [KL04].

Of course, the phase field approach inherently offers considerable advantages for the simulation of droplet impact. However, next to its computational complexity, the ansatz of phase field methods itself is conceptually questionable. Normally, within the continuum



approximation, the small scale transition region would become a mathematical surface of zero thickness instead of remaining finite as assumed for these models [Shi07].

From the volume tracking idea to MAC, VOF and LS

The Volume-Of-Fluid (VOF), the Marker And Cell (MAC) and the Level-Set (LS) method are often referred to as volume tracking methods; a classification which is correct but misleading as we will see in Subsection 2.2.3. In the following, we show how this affiliation comes about.

The idea of volume tracking methods is as follows: We first divide the solution region Ω into a union of disjoint regions $\Omega = \Omega_1 \cup \Omega_2 \cup \Gamma_f \subset \mathbb{R}^3$, where Ω_1 and Ω_2 denote the two phases and Γ is the interface. Of course, all of these subdomains depend on time $t \geq 0$. Then, we can define the characteristic function

$$\chi_1(x,t) = \begin{cases} 1, & \text{if } \boldsymbol{x} \in \Omega_1(t) \\ 0, & \text{else.} \end{cases} \tag{2.5}$$

By the simple observation that $\partial\Omega_1(t) = \partial \operatorname{supp} \chi_1(\cdot, t)$ we see that the interface $\Gamma(t)$ can be defined by the boundary of the support of χ_1. Thus, by tracking the characteristic function, we can track the interface itself; see [GR11] for further details.

This very general approach leads to the very important class of VOF methods which are based on the transport equation

$$\frac{\partial \chi_1}{\partial t} + \boldsymbol{u} \cdot \nabla \chi_1 = 0,$$

where $\boldsymbol{u}$ denotes the fluid velocity. Following [GR11], we integrate this equation over an arbitrarily small fluid volume $W \subset \Omega$ and with partial integration obtain

$$\frac{\partial}{\partial t} \int_W \chi_1 \, d\boldsymbol{x} + \int_{\partial W} \chi_1 \boldsymbol{u} \cdot \boldsymbol{n} \, ds = 0, \tag{2.6}$$

where $\boldsymbol{n}$ denotes the outward normal on ∂W. Thus, the change of volume in fluid 1 contained in W equals the volume flux across the boundary of W, which simply describes mass conservation in Ω.

Then, the VOF method is based on the idea to approximate the characteristic function χ_1 by fractional volumes in each computational cell, which still fulfill the conservation law (2.6). Furthermore, with the help of these fractional volumes the approximate interface location can be reconstructed at any time. In MAC methods, the characteristic function is replaced by marker particles and the interface can be reconstructed from the density distribution of the particles across the interfacial cells. The idea of LS methods is to use a smooth initial function ϕ instead of the discontinuous volume fractions. Additionally, ϕ is often chosen as a signed distance function, whose zero level-set represents the interface Γ.

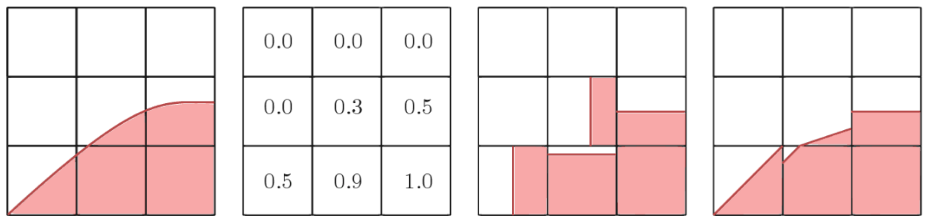


Figure 2.6.: Illustration from left to right: The interface, the color function, the simple line interface calculation (SLIC), and the piecewise linear interface calculation (PLIC). With permission from [GR11].

The Marker and Cell method

A prominent example for an interface capturing method with particles is the MAC method. The original approach by Harlow and Welch [HW65] is one of the earliest attempts to compute solutions of fluid-dynamics problems with free boundaries. Note that the famous staggered-grid arrangement of velocity and pressure variables for a stable solution of the Navier-Stokes equations is a by-product of this approach. The MAC method tracks the free surface in an implicit way.

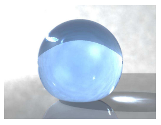
Within this approach, massless marker particles are scattered initially to identify each fluid phase and thereby its free surface, and the particles are tracked by a Lagrangian approach with the fluid. Then, the free surface can be reconstructed by the density distribution of the marker particles in the mixed cells surrounding the interface, which consist of particles of both fluid phases. Therefore, a certain number of particles is required [Hym84], or the interface location is distorted.

By tracking the bulk phases instead of the interface itself, topological changes where many interfaces interact can be easily handled. This works well under the assumption that the interface is well-resolved, which sometimes even requires a larger number of particles than grid cells. These particles have to be stored and relocated, as well as continually deleted and added during the calculation, which requires a great amount of storage demand and computational effort [Jak08].

Due to its computational complexity, applying the MAC method for three-dimensional simulations of drop impact, would be a very costly task. However, over the years, the original MAC scheme was further developed and improved. Originally, the scheme dealt with one fluid only. Then, Daly [Dal69] extended it to two bulk phases. After further refinements by Amsden and Harlow [AH70], Hirt and Nichols [HN81] proposed the use of a marker function instead of marker particles. This marker function is advected by the flow field and used for the front reconstruction in every time step, which resulted in the VOF method.

Volume-of-fluid methods

The VOF method was first introduced by DeBar [DeB74] and got subsequently improved by Noh and Woodward [NW76], Hirt and Nichols [HN81], Lafaurie et al. [LNS$^+$94], and Zaleski et al. [GLN$^+$99]. Within this method, a piecewise constant scalar field (the ‘color field’)



is employed to locate the position of both fluid phases (2.6). This scalar field describes the volume fraction of one fluid phase for each discretization cell and is transported over time with the fluid flow. Similar to the MAC method, the VOF method tracks the interface implicitly.

The VOF algorithm requires a geometric reconstruction of the interface. Here, the earliest method was the 'simple line interface calculation' (SLIC) introduced by Noh and Woodward [NW76]. As shown in Figure 2.6, the SLIC method reconstructs the interface by line segments which are parallel to the coordinate axes and is therefore at best first-order accurate only [GR11].

A well-known extension of this method is the 'piecewise linear interface calculation' (PLIC) [HN81], where the interface is approximated by straight lines perpendicular to the surface normal vector of the interface in each cell. Here, the surface normal is determined from the volume fraction gradients of neighboring cells. Additionally, the position of the line can be determined in such a way that the area beneath the line equals a given volume fraction. For small curvature κ, the jumps in the interface position at cell boundaries is of order $\mathcal{O}(\kappa h^2)$ [SZ99]. Details that fall below the mesh-width are lost, which is of particular significance when the curvature is large. Note that increased accuracy can be achieved by piecewise parabolic interface reconstruction [RR02].

Due to the explicit reconstruction of the interface, the fluxes computed for the advection of the interface are mass-conservative and introduce no diffusion [KL04]. Furthermore, topological changes can be handled easily.

Although the VOF method is inherently mass-conserving, the creation of flotsam and jetsam (i.e. spurious bubbles) can occur. Furthermore, the computation of the normal and of the curvature is difficult and not necessarily accurate since the volume fraction is a discontinuous function and the surface normal a derivative of that function.

For us, the most important advantage of the VOF method is its ability to conserve mass as opposed to our LS method. However, for droplet impact simulations, implementing a VOF method instead is not advisable due to complexity, the less accurate computation of the surface normal and curvature, as well as the unphysical occurrence of spurious bubbles.

2.2.2. The methods used in this thesis

In the following, we present the level-set method and the CLSVOF method which are both used for interface localization in this thesis. We explain their idea, their drawbacks, and our motivation for considering the CLSVOF method instead of the LS method in the first place. The mathematical and numerical details of both methods can be found in Chapter 3, 4, and 5.

Level-set methods

A popular method for implicitly capturing free surface motion is the LS method introduced by Osher and Sethian [OS88], where a smooth scalar field is advected with the flow, and the zero level-set of this field represents the interface; see Figure 2.7.

Similar to the VOF method, the LS technique greatly reduces the complexity required for the capturing of the interface and can handle topological changes such as pinching and

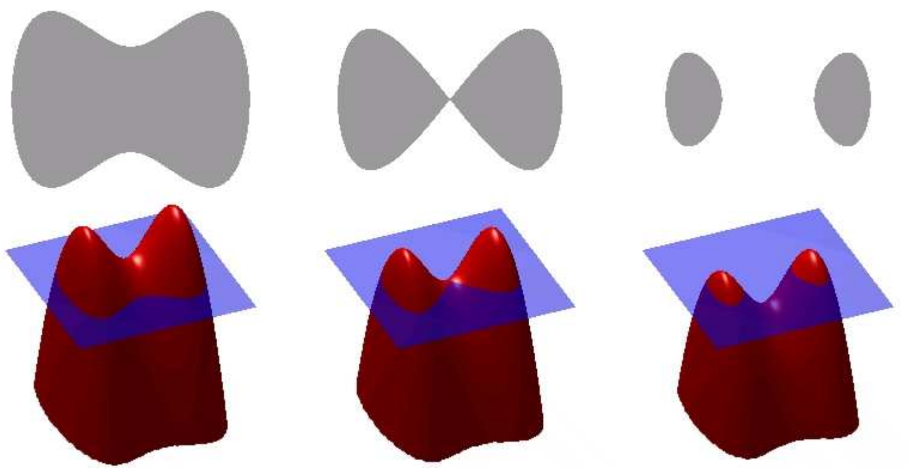


Figure 2.7.: Based on [Wik03b]: Illustration of the LS method.

merging automatically. Both methods are Eulerian in character. In contrast to the VOF method however, the LS method does not need an explicit reconstruction of the interface. Instead, the interface can remain implicit at all times.

However, the LS method also suffers from several drawbacks. Topological changes simulated with the LS method are often under-resolved. Then, these changes only occur due to the diffusion introduced by the LS method and not because of physical necessities. Additionally, the LS function should remain a signed distance function at all times, but the LS advection distorts the interface. Therefore, a so-called reinitialization step must be performed, where the LS function is replaced by a smoother, less distorted function which has the same zero level-set. Although there are several techniques for the reinitialization of the LS function, simple reinitialization techniques introduce numerical diffusion to the solution. This leads to difficulties with volume conservation [CGS04]. Therefore, several modified reinitialization methods have been developed to improve mass conservation [RS00, SP00, PMO$^+$99] including the local volume correction method by Sussman and Fatemi [SF99], which we implement as part of this thesis. Unfortunately, volume correction methods can in turn become a source of numerical errors (cf. Chapter 6) and can introduce undesirable noise in the curvature [Her05, Kec98], which renders this method rather unreliable for most simulation purposes. This includes droplet impact, where surface tensions effects and the shape of the interface at the boundary, play a dominant role.

A further way to improve on the mass conservation of the LS function is the use of hybrid methods. Thus, in [EFFM02] the LS method is coupled with a Lagrangian particle method, where the marker particles are used to rebuild the level-set in regions which are under-resolved. Another hybrid approach is the coupling of the LS with the volume-of fluid method, which results in the CLSVOF method described next.

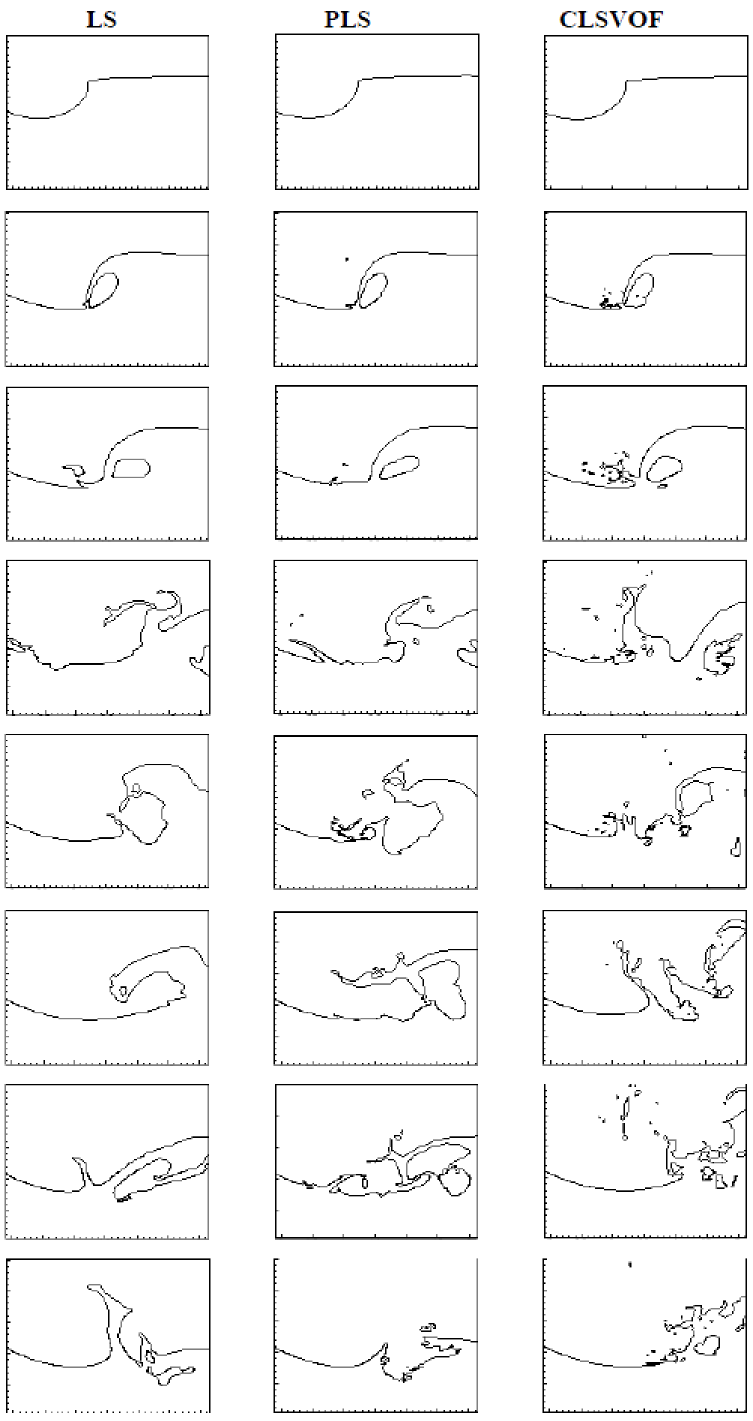


Figure 2.8.: Instantaneous free surface profiles of the wave breaking process with the level-set method, the particle level-set method, and the coupled level-set volume-of-fluid method. Time evolution is from top to bottom. With permission from [WYS08].

The CLSVOF method

The main idea of the CLSVOF method is to exploit the advantages of both the LS and of the VOF method. Thus, the mass-conservative VOF function is used to correct the mass enclosed by the zero level-set of ϕ. Furthermore, the LS function is used to truncate the VOF function to avoid the aforementioned flotsam and jetsam. Additionally, we do not have to reconstruct the curvature or the surface normal from the discontinuous volume fractions. Instead, we use the smooth LS function for their computation. Due to the combination of the advantages of both methods, the CLSVOF method is ideally suited for our droplet impact simulations.

Let us give a few more details on this approach. We transport two scalar indicator functions over time with the flow field, i.e. the continuous LS function ϕ and the discontinuous VOF function, so that

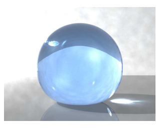
$$\frac{\partial\{F,\phi\}}{\partial t} + \boldsymbol{u}\cdot\nabla\{F,\phi\} = 0.$$

For the discretization of this equation, we have to compute both the LS fluxes and the VOF fluxes across computational cells. Since the LS function is continuous, the respective fluxes can be computed by interpolation from neighboring cells. This, however, is not possible for the discontinuous VOF function. Thus, the computation of the VOF fluxes requires a geometric reconstruction of the interface.

For the geometric reconstruction of the interface, we opt for the PLIC approach where the interface is approximated by straight planes perpendicular to the surface normal vector of the interface in each cell; compare Figure 2.6. Therefore, we compute a piecewise linear reconstructed LS function ϕ^R in such a way that it is as close as possible to the real LS function ϕ. Specifically, we determine the normal $\boldsymbol{n}^R$, which is perpendicular to ϕ^R, as well its distance d to the origin in each computational cell. With the help of the VOF function, the position of the plane is newly computed so that the area beneath the plane equals the given volume fraction. Due to the reconstructed interface, we can then compute the VOF fluxes geometrically. In a last step we reinitialize the LS function by assigning it its exact signed normal distance to the reconstructed interface. All of these interrelated steps are explained in great detail in Chapter 5.

In this thesis we were inspired to implement the CLSVOF method instead of the particle level-set method (PLS) due to a study by Wang, Yang and Stern [WYS08], where both methods are implemented and compared with each other. Here, the PLS method performs better for the standard test cases such as Zalesak's rotating disk, single vortex flow, or the deformation of a sphere. For these test cases, the CLSVOF and PLS methods are not coupled to the full Navier-Stokes solver since only the transport equation is solved. After the flow field is reserved, the initial shape of the disk, vortex or sphere should be recovered as closely as possible.

In these simple tests, the PLS method is superior to the CLSVOF method since the initially prescribed particles can simply return back to their initial position after the flow reverses. Therefore, without any reseeding of the particles, the initial shapes can be recovered exactly. However, in practice, the reseeding is dependent on the LS function and frequently required for situations with complicated interface interactions. Thus, for realistic simulations such as drop impact or plunging breaking waves, the CLSVOF method shows more realistic and



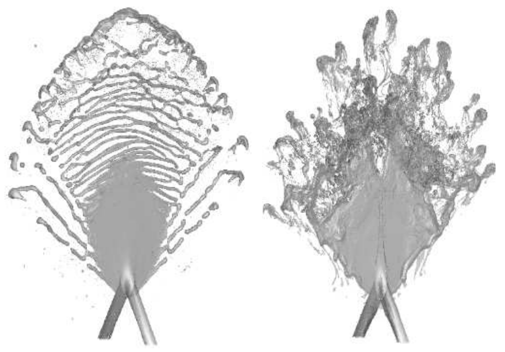


Figure 2.9: Interface profile for impinging jets with the CLSVOF method on the left and the CLSMOF method on the right. With permission from [JLS+13].

reasonable results than the PLS method as exemplified in Figure 2.8. Here, the CLSVOF method reproduces many of the details observed in practical experiments.

Note that the idea of coupling LS with VOF methods is not new and there are many articles devoted to this topic; see [Mén07, MTB07, SH02, Son03, SP00, Sus03, SSH+07, WYKS09]. However, the vast majority skip the details of the involved complex computational algorithms or focus on single parts of them, which makes a fast and straightforward implementation of the CLSVOF method impossible.

When we describe the CLSVOF method in Chapter 5 of this thesis, we are therefore devoted to a comprehensive description of the CLSVOF method. We especially focus on those parts of the method, which are neglected in the articles above. Thus, we explain how the reduction of the number of interface configurations works and present a new algorithm for the reinitialization of the LS function. Additionally, we also discuss the parallelization of the CLSVOF method.

Outlook on the CLSMOF method

A further development of the CLSVOF method is the coupled level-set and moment-of-fluid (CLSMOF) method, which uses next to the LS and the VOF function, also a reference centroid in order to produce a slope and an intercept for the local reconstruction of the interface [JLS+13]. Jemison et al. claim similar accuracy of the CLSMOF and the CLSVOF method for many test problems and a better preserved interface with the CLSMOF method for 3D problems with deforming, stretching or disintegrating interfaces; see Figure 2.9.

In parts, the CLSMOF implementation is based on the CLSVOF method, which we describe extensively in this thesis. Therefore, our description also offers a natural access to the CLSMOF method, which, additionally, requires the consideration of the centers of mass in each computational cell.

2.2.3. Assessment of interface localization categories

Our last section is dedicated to a discussion of the usefulness of the prevalent categories of interface localization in the literature, namely ‘front tracking’, ‘front capturing’ and ‘particle’ methods. For example, Jakobsen [Jak08, p. 345] writes:

> In the early *particle methods* the flow fields of the bulk phases are solved on a fixed grid, and the position of the interface is identified using passive marker

> particles, which are following the bulk flow. In the modern *front capturing* methods, the interface or front separating the bulk phases appears directly on a fixed grid as a region with steep gradients in the primary variables [228]. The *front tracking* methods are usually constructed by explicitly tracking the interface by use of a moving grid, a front, determining a material boundary between the fluid phases which are thus treated separately on a fixed grid [224] [228].
>
> Following these classifications, the MAC, SMAC, VOF and LS techniques are all front capturing methods. The MAC and SMAC techniques are also referred to as particle methods. In the VOF and LS methods a similar marker function is used to identify the phases and reconstruct the interface, hence they are sometimes called volume tracking methods [111] [227].

Refer to [Jak08] for further details and the references in this quotation. Here, MAC is the Marker and Cell method and SMAC the Simplified Marker and Cell method.

Jakobsen also writes that the above ‘compound’ characterization has only lately been invented to distinguish the different interface localization techniques since the involved methods have become more and more complex.

However, we already see that within this single work the categorization is ambiguous. Thus, we have a single simple category for the particle methods, where the MAC method would naively fit in – but it is still foremost referred to as a front capturing method by Jakobsen. Furthermore, another source of ambiguity is the definition of VOF andLS as front capturing methods while they are ‘sometimes called volume tracking methods’, which, historically, is a front tracking approach.

These historical subcategories of front tracking can be found in [Hym84]. Here, the front tracking approach is subdivided into surface tracking methods, moving mesh methods and volume tracking methods. Furthermore, in this work, both the MAC and the VOF method, are mentioned as examples of volume tracking methods. Thus, the MAC method not only fits in with particle methods and front capturing methods but also with front tracking methods, i.e. into *all* three interface localization categories as defined by Jakobsen.

Note that Jakobsen’s work is only an example for the prevalent tries to categorize interface localization methods within the community of researchers dealing with two-phase flow problems. Let us give a few more examples on this point.

In a recent book by Gross and Reusken [GR11, pp. 169–180], the classification described by Jakobsen is already abandoned to a certain degree. Still, immersed interface methods and ALE methods are referred to as explicit front tracking methods, while VOF and LS methods are defined as volume tracking methods. To a certain degree, this corresponds to the definition in [Gro08, p. 21], where both the LS and the VOF method are called front capturing, and the VOF method later on becomes an example for the volume tracking approach. However, if we follow the historical definition by Hyman [Hym84] above, the volume tracking method is a front tracking approach itself, i.e. *not* a front capturing one, which seems to be in direct opposition to what is written in [GR11] and [Gro08].

All in all, even if the ambiguous classification of ‘front tracking’ and ‘front capturing’ is dropped by using ‘explicit’ and ‘implicit’ interface representation instead, the confusion cannot be circumvented:

> Methods using an implicit representation of the interface such as VOF or

level-set (LS) can robustly and efficiently represent evolving, topologically complex interfaces but generally suffer from an inaccurate representation of surface tension. Methods using an explicit representation of the interface such as ALE discretizations or front-tracking (FT) can provide an accurate representation of surface tension but have difficulty dealing with complex and evolving interface topologies.

Refer to [FAJ+09, pp. 2–3] for further details and the references in this quotation, which we have omitted for a better readability. Although, the classification between explicit and implicit representation is very useful, now the original front tracking method appears as a method in its own right – which might again confuse those who mainly use it in terms of categorization.

Then, there are hybrid approaches, which do not simplify the classification of interface localization methods. Thus, Unverdi and Tryggvason [TBE+01, UT92] call their immersed interface method a hybrid between a front tracking and a front capturing approach. On the one hand, a moving interface grid and a fixed fluid grid is used – which is a trademark of surface tracking methods, while on the other hand the same set of governing equations is solved on both grids – which follows most front capturing approaches [Jak08]. Additionally, Tryggvason proposes four classes of interface localization techniques, namely front capturing methods that use separate boundary-fitted grids for each fluid phase, Lagrangian methods where the grid follows the fluid, and a special class of front tracking methods as mainly developed by Glimm et al. [GGG+98].

In our opinion, the source of all the ambiguity is due to the historical development of interface localization methods. Thus, in 1965, the MAC method was one of the earliest attempts to compute solutions to two-phase fluid dynamic problems. From this method, the volume-of fluid method arose about ten years later. Both of these methods do not explicitly track the interface and, therefore, nowadays would be classified as interface capturing methods.

However, until the LS method was developed in 1988, there was simply no need to differentiate between explicit vs. implicit or front tracking vs. capturing approaches. Therefore, the MAC method, as well as the VOF method, fell under the category of 'volume tracking methods' within the then known front tracking methods. As already mentioned, in [Hym84], only the front tracking approach is mentioned, subdivided into surface tracking methods, moving mesh methods and volume tracking methods, and both the MAC and the VOF method are categorized as volume tracking methods, while the LS method had not yet been developed.

There is another reason why the MAC method is even nowadays categorized as a front tracking method: A further development of the MAC method were the surface tracking methods with particles. Here, ordered and connected particles are used to track the interface explicitly. This method is then 'clearly' a front tracking method, which probably explains why some authors classify the very first particle method, the MAC method, as a front tracking approach as well.

In conclusion, we can only suggest to drop the categories front tracking, front capturing and particle methods since they are more a source of confusion than a contribution to a deeper understanding of the different methods for the localization of interfaces. As already partly done in [GR11] and [TBE$^+$01], it makes a lot more sense to speak of methods which represent the interface explicitly or implicitly – and to use the term 'front tracking' only for the special approach developed by Glimm et al. [GGG$^+$98] and methods very closely related to that approach. Furthermore, we propose to differ interface localization methods by whether they track the interface in an Eulerian or Lagrangian manner or by a mixture of the two.

2.3. Coupling of contact angle models and Navier-Stokes equations

In this section, we address the implementation of the contact angle, which is needed as a boundary condition for the LS function ϕ.

In the literature, a number of different approaches for the contact angle boundary condition of the LS function can be found: Here, the simplest model is the zero Neumann boundary condition, which effectively fixes the contact angle to be 90°. If θ is variable, the implementation is less clear. One of the main approaches [YVHH09, YCZ11] was developed by Sussman [Sus01]. Here, the contact angle is taken into account by extrapolating the liquid interface, represented by the LS function, into the solid. This approach requires the construction of an appropriate extension velocity, but the exact location of the position of the contact line is not needed. Moreover, in a method by Spelt [Spe05], contact line position and contact angle or contact line velocity are determined iteratively.

In [ZGK09], the movement of the contact line is induced by diffusion. Instead of using a direct relation between the gradient of the LS function and the normal of the interface, a regularized normal vector field is constructed to avoid flux of ϕ over the boundary. Thereby, two additional regularization parameters appear, which influence the shape of the free surface at the contact line.

In [LNY11, MK07, YS07], a Neumann boundary condition for the LS function is derived as follows: Let our fluid flow domain Ω be a box and $\boldsymbol{x} = (x, y, z) \in \Omega$. For example, at the wall at $y = 0$, the geometric relation

$$\boldsymbol{n} \cdot \boldsymbol{n}_w = \cos\theta \tag{2.7}$$

holds for the contact angle θ. Here, $\boldsymbol{n}$ is the outward surface normal and $\boldsymbol{n}_w = (0, -1, 0)^t$ is the outward normal at $y = 0$. Now, a Neumann boundary condition for the LS function can be prescribed by rewriting relation (2.7) as

$$\phi_y = -\cos\theta \text{ for } |\phi| = 1, \tag{2.8}$$

since $\boldsymbol{n} = \phi/|\phi|$ and $|\phi| = \sqrt{\phi_x^2 + \phi_y^2 + \phi_z^2}$. However, the condition $|\phi| = 1$ is not always fulfilled. Thus, the recovery of this property of the LS function is achieved by some reinitialization equation in the first place. In this thesis, we extend this approach and show that

$$\phi_y = -\cot\theta \sqrt{\phi_x^2 + \phi_z^2} \text{ for } 0 < \theta < \pi \tag{2.9}$$

without further assumptions on ϕ. Similar techniques have already been used by van

Mourik [vM02], Fang et al. [FHW$^+$08], and Dupont and Legendre [DL10] within the VOF approach. For example, in [DL10], the discretization of the contact angle boundary condition for a single wall is exemplified and is directly linked to the underlying discretization and the VOF approach. Instead our Proposition 3.1 holds for any wall and is not restricted to the LS approach. Instead, it holds for any function ϕ for which the surface normal can be evaluated as $\boldsymbol{n} = \phi/|\phi|$.

Note here that our approach is consistent with the more general extension technique introduced by Sussman [Sus01] for the case that the geometry is a box. Sussman's approach, however, requires the solution of the partial differential 'extension equation'. Like in his method, we do not need to locate the exact contact line position. Additionally, relation (2.9) allows us to set contact angle boundary conditions for complex geometrical objects in our fluid flow domain. [6]

[6]Then, contact angles at corner cells of the geometry have to fulfill further restrictions as described in [FHW$^+$08].

3. Mathematical Modeling of Free Surface Flows with Dynamic Contact Angles

In this chapter, we discuss our mathematical model for the three-dimensional flow of two immiscible incompressible fluids. We show how the contact angle can be modeled as a boundary condition for the level-set function and present the details of the two dynamic contact angle models as given by Yokoi et al. [YVHH09] and Shikhmurzaev [Shi07].

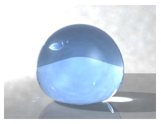
In Section 3.1, we present the standard two-phase Navier-Stokes equations, which have to be complemented with boundary conditions at the interface. These boundary conditions are derived from mass and momentum conservation in Section 3.2, where we mainly follow the descriptions by Shikhmurzaev [Shi07]. In Section 3.3, we start with an integral formulation for the coupling of the Navier-Stokes equations with the boundary conditions at the interface. Within this formulation, we use the continuum surface force approach [BKZ92] to convert the boundary integral into a volume integral. This is by now a common mathematical method and the details that we skip in this section can be found in [CGS10, Gro08]. Additionally, in Section 3.4, we discuss the possible boundary conditions for the velocity and the pressure in the Navier-Stokes equations. In Section 3.5, we present our Neumann boundary condition for the level-set function which includes the contact angle at the boundary. We prove the validity of this boundary condition in Proposition 3.1 and show its connection to Sussman's extension technique [Sus01]. Finally, in Section 3.6, we discuss the models for the dynamic contact angle by Yokoi et al. [YVHH09] and Shikhmurzaev [Shi07]. Since Shikhmurzaev's interface formation model is able to account for a whole variety of flows, where interfaces are formed or destroyed, we provide an insight into his full model before we discuss the simplified version used in this thesis.

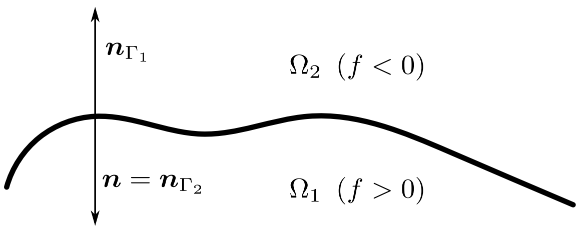
3.1. The two-phase Navier-Stokes equations

The behavior of the fluids is governed by the incompressible Navier-Stokes equations defined on an open domain $\Omega \subset \mathbb{R}^3$ with Lipschitz boundary $\Gamma := \partial\Omega$. The two fluid phases fill the two open subdomains Ω_1 and Ω_2 defined by $\overline{\Omega} = \overline{\Omega}_1 \cup \overline{\Omega}_2$ and $\Omega_1 \cap \Omega_2 = \emptyset$. Then, the free interface separating these phases is given by $\Gamma_f := \overline{\Omega}_1 \cap \overline{\Omega}_2$. The two fluid domains Ω_1 and Ω_2 and the free interface Γ_f depend on time.

The temporal evolution of the fluids is described by the Navier-Stokes equations

$$\begin{aligned} \rho_i \left(\partial_t \boldsymbol{u}_i + (\boldsymbol{u}_i \cdot \nabla)\boldsymbol{u}_i\right) &= \nabla \cdot \boldsymbol{T}_i + \rho_i \boldsymbol{g} && \text{in } \Omega_i \times [0,T] \\ \nabla \cdot \boldsymbol{u}_i &= 0 && \text{in } \Omega_i \times [0,T] \\ \boldsymbol{u}_i|_\Gamma &= 0 && \text{in } [0,T] \\ \boldsymbol{u}_i|_{t=0} &= u_{0,i} && \text{in } \Omega_i \end{aligned}$$





$\boldsymbol{n}_{\Gamma_1}$
$\Omega_2 \ (f < 0)$
$\boldsymbol{n} = \boldsymbol{n}_{\Gamma_2}$
$\Omega_1 \ (f > 0)$

Figure 3.1.: Illustration of the surface normal $\boldsymbol{n}$ and the respective outward surface normals $\boldsymbol{n}_{\Gamma_1}$ and $\boldsymbol{n}_{\Gamma_2}$. The function $f(\mathbf{r}, t) = 0$ describes the smooth moving interface. Fluid 2 is defined by $f < 0$ and fluid 1 by $f > 0$.

with $i = 1, 2$, time $t \in [0, T]$, fluid velocity $\boldsymbol{u}_i$ and volume forces $\boldsymbol{g}$. The stress tensor is $\boldsymbol{T}_i = -p_i \boldsymbol{I} + \mu_i \boldsymbol{S}_i$, where p_i is the pressure. The viscous stress tensor is given by $\boldsymbol{S}_i = \nabla \boldsymbol{u}_i + (\nabla \boldsymbol{u}_i)^{\mathsf{T}}$, and the dynamic viscosity μ_i and density ρ_i are assumed to be constant in each phase. Note that the homogeneous Dirichlet boundary condition $\boldsymbol{u}_i|_\Gamma = 0$ can be replaced by more sophisticated domain dependent boundary conditions as discussed in Section 3.4. To complete this system of equations, we require boundary conditions at the interface Γ_f.

3.2. Boundary conditions at the interface

In continuum mechanics Γ_f is usually regarded as a sharp interface, i.e. a geometric surface of zero thickness which separates the two flow regions. The physics associated with this interfacial layer have to be accounted for by appropriate boundary conditions. These have to incorporate conversation laws on the one hand, as well as specific physical processes of the interfacial layer on the other hand.

Following [Shi07], we describe the smooth moving interface by the function $f(\mathbf{r}, t) = 0$, where $f : \Omega \to \mathbb{R}$, and $\mathbf{r}$ is a position vector located in the interface. Then the outward normal $\boldsymbol{n} = \nabla f / |\nabla f|$ points from fluid 2 ($f < 0$) to fluid 1 ($f > 0$), where $|\cdot|$ denotes the Euclidean norm; the surface normal is illustrated in Figure 3.1. Thus, the movement of the interface can be described by the transport equation

$$\frac{\partial f}{\partial t} + \mathbf{v}_s \cdot \nabla f = 0, \tag{3.1}$$

where $\mathbf{v}_s$ denotes the interface velocity.

Conservation of mass

We assume that the density in the thin interfacial layer is of the same order as in the bulk. Then, the interface's role as a source or sink of mass becomes negligible compared to the mass flux across its boundary

$$\rho_1(\boldsymbol{u}_1 - \mathbf{v}_s) \cdot \boldsymbol{n} = \rho_2(\boldsymbol{u}_2 - \mathbf{v}_s) \cdot \boldsymbol{n} \text{ at } f(\mathbf{r}, t) = 0. \tag{3.2}$$

In addition to this equation, we have to prescribe the mass flux itself by

$$\rho_1(\boldsymbol{u}_1 - \mathbf{v}_s) \cdot \boldsymbol{n} = c, \tag{3.3}$$

where c is the result of physical mechanisms responsible for the mass transfer, such as chemical reactions, evaporation-condensation, etc. [Shi07, p.68].

For an impermeable surface, as in our setting, c is zero and the combination of equations (3.2) and (3.3) yields

$$\begin{aligned} &\rho_i(\boldsymbol{u}_i - \mathbf{v}_s) \cdot \boldsymbol{n} = 0 \\ \Leftrightarrow\ &(\boldsymbol{u}_i - \mathbf{v}_s) \cdot \boldsymbol{n} = 0 \\ \Leftrightarrow\ &\boldsymbol{u}_i \cdot \boldsymbol{n} = \mathbf{v}_s \cdot \boldsymbol{n} \end{aligned} \tag{3.4}$$

Thus, the normal component of $\mathbf{v}_s$ can be replaced by $\boldsymbol{u}_i$, and we obtain

$$\boldsymbol{u}_1 \cdot \boldsymbol{n} = \boldsymbol{u}_2 \cdot \boldsymbol{n}. \tag{3.5}$$

Furthermore, replacing $\boldsymbol{n} = \nabla f / |\nabla f|$ in (3.4) and combination with the transport equation (3.1) yields

$$\frac{\partial f}{\partial t} + \boldsymbol{u}_i \cdot \nabla f = 0. \tag{3.6}$$

The kinematic equation (3.5) prescribes a relation of the normal projection of the two bulk velocities $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ at the interface while (3.6) determines the evolution of the interface's shape.

Conservation of momentum

If the interface's source or sink is negligible compared to the momentum flux across the boundary, we obtain

$$\boldsymbol{n} \cdot \boldsymbol{\Pi}_1 = \boldsymbol{n} \cdot \boldsymbol{\Pi}_2$$

with $\boldsymbol{\Pi}_i = \rho_i(\boldsymbol{u}_i - \mathbf{v}_s)(\boldsymbol{u}_i - \mathbf{v}_s)^\mathsf{T} - \boldsymbol{T}_i$ the momentum flux tensor. However, there are situations when this assumption is not valid. In such cases, there are two components which describe the source/sink of momentum due to an interface [Shi07]. First, there may be external forces $\mathbf{F}_s$ which act on the interface and cannot be neglected despite the interface's negligible thickness. For example, there can be an electrically charged interface while the fluid in the bulk is neutral. Second, the interface possesses intrinsic properties, which can contribute to the overall dynamics of the fluid. For example, the surface tension σ results from intermolecular forces from the bulk phases which are singularly strong compared to other forces on the fluid and, therefore, cannot be neglected. Then, we have

$$\boldsymbol{n} \cdot (\boldsymbol{\Pi}_1 - \boldsymbol{\Pi}_2) = \nabla \cdot \boldsymbol{T}_s + \mathbf{F}_\mathbf{s}$$

with the surface stress tensor $\boldsymbol{T}_s = \sigma(\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T})$, where $(\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T})$ projects any vector onto the interface (i.e. it singles out the components of the vector which are tangential to the interface).

For an impermeable interface and negligible external surface forces we obtain the stress balance equation

$$\boldsymbol{n} \cdot (\boldsymbol{T}_1 - \boldsymbol{T}_2) = \sigma \boldsymbol{n} \, (\nabla \cdot \boldsymbol{n}) - \nabla \sigma. \tag{3.7}$$

The normal projection of equation (3.7) yields

$$\boldsymbol{n} \cdot (\boldsymbol{T}_1 - \boldsymbol{T}_2) \cdot \boldsymbol{n} = \sigma \left(\nabla \cdot \boldsymbol{n}\right), \tag{3.8}$$

which means that the normal stress across the interface must balance the curvature force. Here, we have already used that the surface tension gradient is directed along the interface, which means that $\nabla\sigma \cdot \boldsymbol{n} = 0$ and $\nabla\sigma \cdot (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T}) = \nabla\sigma$. Then, the tangential projection of (3.7) is

$$\boldsymbol{n} \cdot (\boldsymbol{T}_1 - \boldsymbol{T}_2) \cdot (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T}) = \nabla\sigma \tag{3.9}$$

The right hand side of this equation is zero, if we assume the surface tension to be constant.

The missing boundary condition

In order to obtain a closed system of equations, a further boundary condition on the interface is needed. Conventionally, for viscous fluid phases, the tangential component of the bulk velocity is assumed to be continuous over the interface, i.e.

$$\boldsymbol{u}_1(\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T}) = \boldsymbol{u}_2(\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T}) \tag{3.10}$$

A violation of this condition would lead to very large stresses at the interface, which is not expected for viscous fluids; see [Shi07, p.74] for further details.

3.3. Coupling the Navier-Stokes equations with the boundary conditions

All in all, the complete model for two-phase flow problems including the boundary conditions and equations (3.5), (3.6), (3.7) and (3.10) at the interface is given by

$$\begin{aligned}
\rho_i \left(\partial_t \boldsymbol{u}_i + (\boldsymbol{u}_i \cdot \nabla)\boldsymbol{u}_i\right) &= \nabla \cdot \boldsymbol{T}_i + \rho_i \boldsymbol{g} && \text{in } \Omega_i \times [0,T] \\
\nabla \cdot \boldsymbol{u}_i &= 0 && \text{in } \Omega_i \times [0,T] \\
\boldsymbol{u}_i|_\Gamma &= 0 && \text{in } [0,T] \\
\boldsymbol{u}_i|_{t=0} &= u_{0,i} && \text{in } \Omega_i \\
\boldsymbol{u}_1 &= \boldsymbol{u}_2 && \text{on } \Gamma_f \times [0,T] \\
\boldsymbol{n} \cdot (\boldsymbol{T}_1 - \boldsymbol{T}_2) &= \sigma \boldsymbol{n} \left(\nabla \cdot \boldsymbol{n}\right) - \nabla\sigma && \text{on } \Gamma_f \times [0,T] \\
\frac{\partial f}{\partial t} + \boldsymbol{u}_i \cdot \nabla f &= 0 && \text{on } \Gamma_f \times [0,T].
\end{aligned} \tag{3.11}$$

In order to couple the Navier-Stokes equations with the free surface boundary conditions, we start with the integral form of the Navier-Stokes equations, i.e.

$$\rho_i \int_{\Omega_i} \partial_t \boldsymbol{u}_i + (\boldsymbol{u}_i \cdot \nabla)\boldsymbol{u}_i \, d\boldsymbol{x} = \int_{\partial\Omega_i} \boldsymbol{T}_i \cdot \boldsymbol{n} \, ds + \rho_i \int_{\Omega_i} \boldsymbol{g} \, d\boldsymbol{x}.$$

for $i \in \{1, 2\}$. Summation yields

$$\sum_{i=1,2} \rho_i \int_{\Omega_i} \partial_t \boldsymbol{u}_i + (\boldsymbol{u}_i \cdot \nabla)\boldsymbol{u}_i - \boldsymbol{g}\, d\boldsymbol{x} = \int_{\partial\Omega_1} \boldsymbol{T}_1 \cdot \boldsymbol{n}_w^1\, ds + \int_{\partial\Omega_2} \boldsymbol{T}_2 \cdot \boldsymbol{n}_w^2\, ds \\ + \int_{\Gamma_f} \boldsymbol{T}_1 \cdot \boldsymbol{n}_{\Gamma_1}\, ds + \int_{\Gamma_f} \boldsymbol{T}_2 \cdot \boldsymbol{n}_{\Gamma_2}\, ds.$$

Here, $\boldsymbol{n}_w^i$ denote the outward normals on $\partial\Omega_i$, and $\boldsymbol{n}_{\Gamma_i}$ denote the unit surface normals on Γ_f from the perspective of Ω_i.

With our definition of $\boldsymbol{n} = \nabla f / |\nabla f|$, which points from fluid 2 ($f < 0$) to fluid 1 ($f > 0$), we have

$$\boldsymbol{n} = \boldsymbol{n}_{\Gamma_2} = -\boldsymbol{n}_{\Gamma_1},$$

see Figure 3.1. Thus,

$$\sum_{i=1,2} \rho_i \int_{\Omega_i} \partial_t \boldsymbol{u}_i + (\boldsymbol{u}_i \cdot \nabla)\boldsymbol{u}_i - \boldsymbol{g}\, d\boldsymbol{x} = \int_{\partial\Omega_1} \boldsymbol{T}_1 \cdot \boldsymbol{n}_1\, ds + \int_{\partial\Omega_2} \boldsymbol{T}_2 \cdot \boldsymbol{n}_2\, ds \\ - \int_{\Gamma_f} (\boldsymbol{T}_1 - \boldsymbol{T}_2) \cdot \boldsymbol{n}\, ds.$$

From now on we assume that the surface tension is constant, i.e. $\nabla\sigma = 0$ in (3.11). We define $T = T_1\chi_{\Omega_1} + T_2\chi_{\Omega_2}$ with the help of characteristic functions on Ω_1 and Ω_2, respectively. Then, with the Gauss Theorem and the stress balance equation (3.7), we find

$$\sum_{i=1,2} \rho_i \int_{\Omega_i} \partial_t \boldsymbol{u}_i + (\boldsymbol{u}_i \cdot \nabla)\boldsymbol{u}_i - \boldsymbol{g}\, d\boldsymbol{x} = \int_{\Omega} \nabla \cdot \boldsymbol{T}\, d\boldsymbol{x} - \int_{\Gamma_f} \sigma \boldsymbol{n}\, (\nabla \cdot \boldsymbol{n})\, ds. \tag{3.12}$$

Due to the continuity of the velocity field at the interface, we write the velocity field as $\boldsymbol{u} = \boldsymbol{u}_1\chi_{\Omega_1} + \boldsymbol{u}_2\chi_{\Omega_2}$ and obtain

$$\sum_{i=1,2} \rho_i \int_{\Omega_i} \partial_t \boldsymbol{u}_i + (\boldsymbol{u}_i \cdot \nabla)\boldsymbol{u}_i - \boldsymbol{g}\, d\boldsymbol{x} = \int_{\Omega} \rho(\boldsymbol{x})\partial_t \boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} - \rho(\boldsymbol{x})\boldsymbol{g}\, d\boldsymbol{x}, \tag{3.13}$$

where

$$\rho(\boldsymbol{x}) = \begin{cases} \rho_1 & \text{if } \boldsymbol{x} \in \Omega_1 \\ \rho_2 & \text{if } \boldsymbol{x} \in \Omega_2. \end{cases}$$

Finally, inserting (3.13) into (3.12) and with the definition of the curvature $\kappa = \nabla \cdot \boldsymbol{n}$, we obtain

$$\int_{\Omega} \rho\, (\partial_t \boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u})\, d\boldsymbol{x} = \int_{\Omega} \nabla \cdot \boldsymbol{T}\, d\boldsymbol{x} - \int_{\Gamma_f} \sigma\kappa\boldsymbol{n}\, ds + \int_{\Omega} \rho\boldsymbol{g}\, d\boldsymbol{x}. \tag{3.14}$$

for the incompressible two-phase flow problem with surface tension; see [CGS10, Gro08] for more details.

Note that the surface tension term $-\int_{\Gamma_f} \sigma\kappa\boldsymbol{n}\, ds$ in equation (3.14) is often encountered with a positive sign in the literature, for example in [Gro08]. This sign depends on the

definition of the curvature and of the surface normal. Suppose the surface normal $\tilde{\boldsymbol{n}}$ is chosen to point from Ω_1 to Ω_2 and the curvature is defined as $\tilde{\kappa} = -\nabla \cdot \tilde{\boldsymbol{n}}$. Then, the positive term

$$\begin{aligned} +\int_{\Gamma_f} \sigma\tilde{\kappa}\tilde{\boldsymbol{n}}\, ds &= +\int_{\Gamma_f} \sigma(-\nabla\cdot\tilde{\boldsymbol{n}})\tilde{\boldsymbol{n}}\, ds \\ &= \int_{\Gamma_f} \sigma(\nabla\cdot\boldsymbol{n})(-\boldsymbol{n})\, ds \quad = -\int_{\Gamma_f} \sigma\kappa\boldsymbol{n}\, ds \end{aligned}$$

To summarize, there is no consistent notation in the literature, and one has to be very careful with the precise definition of the curvature and the surface normal.

By (3.14), the boundary condition for the surface tension is implicitly contained in the momentum equations. However, the boundary integral $\int_{\Gamma_f} \sigma\kappa\boldsymbol{n}\, ds$ still requires the knowledge of the interface's exact location for each time instant $t \in [0,T]$. Therefore, in a next step, the continuum surface force (CSF) approach is applied to convert the boundary integral into a volume integral. To this end, we now choose f as a level-set function ϕ to describe the free surface.

3.3.1. The level-set method

We define ϕ as a signed-distance function such that

$$\phi(\boldsymbol{x},t) \begin{cases} < 0 & \text{if } \boldsymbol{x} \in \Omega_1 \\ = 0 & \text{if } \boldsymbol{x} \in \Gamma_f \\ > 0 & \text{if } \boldsymbol{x} \in \Omega_2 \end{cases} \tag{3.15}$$

holds and the Eikonal equation $|\nabla\phi| = 1$ is fulfilled, where $|\cdot|$ still denotes the Euclidean norm. The interface between the two fluids is then given by the zero level-set of ϕ as

$$\Gamma_f(t) = \{\boldsymbol{x} : \phi(\boldsymbol{x},t) = 0\}$$

for all times $t \in [0,T]$. The level-set function is advected by the pure transport equation

$$\phi_t + \boldsymbol{u}\cdot\nabla\phi = 0 \tag{3.16}$$

with initial value $\phi_0(\boldsymbol{x}) = \phi(\boldsymbol{x},0)$.

With the help of ϕ, we define the density ρ and the viscosity μ on the whole domain, i.e. on both fluid-phases. To this end, we set

$$\rho(\phi) := \rho_2 + (\rho_1 - \rho_2)H(\phi) \quad \text{and} \quad \mu(\phi) := \mu_2 + (\mu_1 - \mu_2)H(\phi), \tag{3.17}$$

where $H(\phi)$ denotes the Heaviside function which is defined as

$$H(\phi) := \begin{cases} 0 & \text{if } \phi < 0 \\ \frac{1}{2} & \text{if } \phi = 0 \\ 1 & \text{if } \phi > 0. \end{cases}$$

With the above definitions, our integral formulation of the Navier-Stokes equations (3.14) becomes

$$\int_\Omega \rho(\phi)\left(\partial_t \boldsymbol{u} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\right)\,d\boldsymbol{x} = \int_\Omega \nabla\cdot\boldsymbol{T}\,d\boldsymbol{x} - \int_{\Gamma_f}\sigma\kappa\boldsymbol{n}\,ds + \int_\Omega \rho(\phi)\boldsymbol{g}\,d\boldsymbol{x}. \tag{3.18}$$

3.3.2. The continuum surface force method

In order to convert the boundary integral in equation (3.18) into a volume integral we use the CSF approach [BKZ92].

Here, we need the delta distribution which is a continuous linear mapping from the space of infinitely differentiable functions with compact support onto $\mathbb{R}$, i.e.

$$\delta\colon\ C_c^\infty \to \mathbb{R}\,,\ \varphi \mapsto f(0).$$

In the theory of distributions, δ is defined as

$$\langle\delta,\varphi\rangle = \varphi(0),$$

and is the derivative of the distribution corresponding to the Heaviside step function H. Thus,

$$\begin{aligned}\langle H',\varphi\rangle &= -\langle H,\varphi'\rangle\\ &= -\int_{-\infty}^{\infty} H(x)\varphi'(x)dx\\ &= -\int_0^\infty \varphi'(x)dx \text{ almost everywhere}\\ &= \varphi(0)-\varphi(\infty) = \varphi(0) = \langle\delta,\varphi\rangle \ \text{ a.e.,}\end{aligned}$$

for any test function φ (cf. [Wik03c]).

In [CHMO96] it is shown that that

$$\int_{\Gamma_f}\sigma\kappa\boldsymbol{n}\,ds = \int_\Omega \sigma\kappa(\phi(\boldsymbol{x}))\delta(\phi(\boldsymbol{x}))\nabla\phi(\boldsymbol{x})\,d\boldsymbol{x}, \tag{3.19}$$

which we write with the help of the identities $\boldsymbol{n} = \nabla\phi/\left|\nabla\phi\right|$ and $|\nabla\phi| = 1$. Here, $\kappa(\phi) = \nabla\cdot\nabla\phi/|\nabla\phi|$. Substitution into equation (3.18) yields a volume integral over the whole domain Ω, which holds for any arbitrarily small volume Ω. Therefore, we can pass from (3.18) and (3.19) to the associated differential equation and obtain

$$\begin{aligned}\rho(\phi)\left(\partial_t\boldsymbol{u} + (\boldsymbol{u}\cdot\nabla\boldsymbol{u})\right) &= \nabla\cdot\boldsymbol{T} - \sigma\kappa(\phi)\delta(\phi)\nabla\phi + \rho(\phi)\boldsymbol{g}\\ \nabla\cdot\boldsymbol{u} &= 0.\end{aligned}$$

Furthermore, with $\boldsymbol{T} = -p\boldsymbol{I} + \mu(\phi)\boldsymbol{S}$, we have

$$\begin{aligned}\rho(\phi)\left(\partial_t\boldsymbol{u} + (\boldsymbol{u}\cdot\nabla\boldsymbol{u})\right) + \nabla p &= \nabla\cdot(\mu(\phi)\boldsymbol{S}) - \sigma\kappa(\phi)\delta(\phi)\nabla\phi + \rho(\phi)\boldsymbol{g}\\ \nabla\cdot\boldsymbol{u} &= 0\end{aligned} \tag{3.20}$$

again with time $t\in[0,T]$, fluid velocity $\boldsymbol{u}$, pressure p, stress tensor $\boldsymbol{S} = \nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^\mathsf{T}$ and

a volume force $\boldsymbol{g}$. Here, μ is the viscosity and ρ the density, both defined by (3.17).

Next, we complement the Navier-Stokes equations (3.20) with boundary conditions for velocity and pressure.

3.4. Boundary conditions for velocity and pressure

At $t = 0$ we impose initial conditions on the velocity $\boldsymbol{u}$ which satisfy the continuity equation of the Navier-Stokes equations. In addition, conditions at the domain's boundary $\Gamma = \partial\Omega$ are required so that a well-defined initial boundary value problem results [GDN98]. Let $\boldsymbol{u}\cdot\boldsymbol{n}$ denote the component of the velocity orthogonal to the boundary, $\boldsymbol{u}\cdot\boldsymbol{\tau}$ the component of the velocity tangential to the boundary and $\partial\boldsymbol{u}/\partial\boldsymbol{n}$ the velocity's derivative in normal direction. We consider the following boundary conditions:

No-Slip Condition: The fluid adheres to the boundary, and no fluid penetrates the boundary. Hence the velocity vector has to vanish there,

$$\boldsymbol{u}|_\Gamma = 0.$$

Slip Condition: Again, no fluid penetrates the boundary but, as opposed to the no-slip condition, there are no friction losses at the boundary, i.e.

$$(\boldsymbol{u}\cdot\boldsymbol{n})|_\Gamma = 0, \quad \partial_{\boldsymbol{n}}(\boldsymbol{u}\cdot\boldsymbol{\tau})|_\Gamma = 0.$$

In/Outflow Condition: A fixed velocity $\boldsymbol{u}_0$ is given, i.e.

$$\boldsymbol{u}|_\Gamma = \boldsymbol{u}_0.$$

Natural Outflow Condition: The fluid velocity does not change in the normal direction of the boundary,

$$(\partial_{\boldsymbol{n}}\boldsymbol{u})|_\Gamma = 0.$$

Periodic Boundary Condition: For periodic problems along one or more coordinate directions the velocities and pressure values must coincide at opposite boundaries.

Following Gauss's Theorem, we note that the velocity at the boundaries has to fulfill the restriction

$$0 = \int_\Omega \nabla\cdot\boldsymbol{u}\,d\boldsymbol{x} = \int_\Gamma \boldsymbol{u}\cdot\boldsymbol{n}\,d\Gamma.$$

This means that the boundary integral of the normal components of the given velocities must vanish.

Now, in addition to the standard equations of fluid dynamics and the boundary conditions described above, we require boundary conditions for the level-set function ϕ. At the contact line, where the free surface meets the solid substrate, these boundary conditions are determined by the dynamic contact angle. In Section 3.5, we therefore describe how the contact angle can be modeled as a boundary condition for the level-set function. Then, in Section 3.6, we present two theories for the proper modeling of the dynamic contact angle.

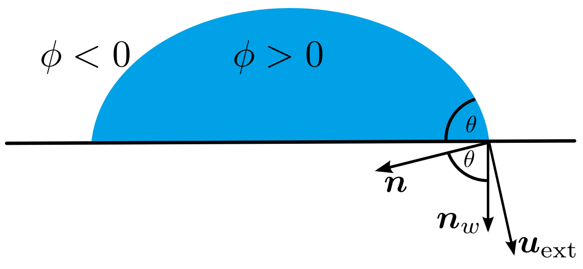


Figure 3.2: Boundary conditions for ϕ. $\boldsymbol{n}$ is normal to the interface and points from lower to higher level set values and $\boldsymbol{n}_w$ is the outward normal drawn from the active flow region into the geometry region. Further, $\boldsymbol{u}_{\text{ext}}$ is the extension velocity of Sussman's approach [Sus01].

3.5. Contact angle boundary condition for the level-set method

In this section, we present a boundary condition for the level-set function, which is required for the transport equation (3.16), the level-set reinitialization and the computation of the curvature κ. This condition determines the shape of the free surface at the contact line and, therefore, depends on the dynamic contact angle θ_d as soon as the contact line is moving.

As already exemplified in Subsection 2.3, we now formulate a Neumann boundary condition for the level-set function, which incorporates the dynamic contact angle. Thus, at the boundary of Ω the contact angle is defined by the geometric relation

$$\boldsymbol{n} \cdot \boldsymbol{n}_w = \cos(\theta), \tag{3.21}$$

where θ is the contact angle (static or dynamic), $\boldsymbol{n}_w$ is the outward normal drawn from the flow region into the boundary, and $\boldsymbol{n}$ is normal to the zero contour of the level-set function which points from the fluid phase with lower level-set values to the one with higher values, i.e.

$$\boldsymbol{n} = \frac{\nabla\phi}{|\nabla\phi|}; \tag{3.22}$$

see Figure 3.2. Then, the boundary condition for the level-set function is given in the following proposition.

Proposition 3.1
At any wall of Ω, whose outward normal is given by $\boldsymbol{n}_w^i = \pm\boldsymbol{e}^i$ for $i \in \{1,2,3\}$, the level-set's i-th derivative ϕ_{x_i} can be related to θ by

$$\phi_{x_i} = \pm\cot(\theta)\sqrt{\sum_{j=1,\, j\neq i}^{3} \phi_{x_j}^2} \tag{3.23}$$

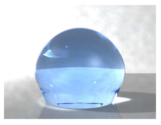
for any angle $0 < \theta < \pi$.

Proof. We prove this proposition for two walls with outward normals $\boldsymbol{n}_w^2 = -\boldsymbol{e}^2$ and $\boldsymbol{n}_w^2 = \boldsymbol{e}^2$, since the cases $\boldsymbol{n}_w^i = \pm\boldsymbol{e}^i$ for $i = 1$ or $i = 3$ can be treated in the very same way. For both boundaries, we have to distinguish between the cases $0 < \theta \leq \frac{\pi}{2}$ and $\frac{\pi}{2} < \theta < \pi$.

Let $\boldsymbol{n}_w^2 = -\boldsymbol{e}^2 = (0,-1,0)^\mathsf{T}$. Then, from equation (3.21) and (3.22), we have

$$\boldsymbol{n} \cdot \boldsymbol{n}_w^2 = \cos(\theta) \Leftrightarrow -\phi_{x_2} = \cos(\theta)|\nabla\phi|. \tag{3.24}$$

First, let $0 < \theta \leq \frac{\pi}{2}$. Then, $0 \leq \cos\theta < 1$ and $\sin^2(\theta) = 1 - \cos^2(\theta) > 0$. Since $\cos(\theta) \geq 0$,



we conclude that $-\phi_{x_2} \geq 0$ as well, and define the positive function $\tilde{\phi} := -\phi_{x_2}$ with $\tilde{\phi}^2 = \phi_{x_2}^2$. Inserting $\tilde{\phi}$ into equation (3.24), we obtain

$$\tilde{\phi} = \cos(\theta)\sqrt{\phi_{x_1}^2 + \tilde{\phi}^2 + \phi_{x_2}^2}.$$

Now, only positive variables constitute both sides of the equation. Thus, we are allowed to square both sides and still obtain the equivalent relation

$$\begin{aligned}
\tilde{\phi}^2 &= \cos^2(\theta)(\phi_{x_1}^2 + \tilde{\phi}^2 + \phi_{x_3}^2) \\
\Leftrightarrow \tilde{\phi}^2 \left(1 - \cos^2(\theta)\right) &= \cos^2(\theta)(\phi_{x_1}^2 + \phi_{x_3}^2) \\
\Leftrightarrow \tilde{\phi}^2 &= \frac{\cos^2(\theta)}{\sin^2(\theta)}(\phi_{x_1}^2 + \phi_{x_3}^2) \\
\Leftrightarrow \tilde{\phi} &= \frac{\cos(\theta)}{\sin(\theta)}\sqrt{\phi_{x_1}^2 + \phi_{x_3}^2}.
\end{aligned}$$

Again, taking the root to obtain the last equivalency relation is only allowed since all parts of the equation (including $\sin^2(\theta)$) are non-negative. Then, for $0 < \theta \leq \frac{\pi}{2}$, we have $\sin(\theta) = \sqrt{1 - \cos^2(\theta)}$. Resubstituting $\tilde{\phi} = -\phi_{x_2}$, we obtain the desired result

$$\phi_{x_2} = -\cot(\theta)\sqrt{\phi_{x_1}^2 + \phi_{x_3}^2}.$$

Now, let $\frac{\pi}{2} < \theta < \pi$. Then, $-1 < \cos\theta < 0$ and $\sin^2(\theta) = 1 - \cos^2(\theta) > 0$. From equation (3.24), we know that $\phi_{x_2} > 0$, since $\cos\theta < 0$. We define the positive function $\tilde{c} := -\cos(\theta)$ with $\tilde{c}^2 = \cos^2(\theta)$ and obtain likewise

$$\phi_{x_2} = \tilde{c}\sqrt{\phi_{x_1}^2 + \phi_{x_2}^2 + \phi_{x_3}^2} \Leftrightarrow \phi_{x_2} = -\cot(\theta)\sqrt{\phi_{x_1}^2 + \phi_{x_3}^2}.$$

In the second part of this proof, the outward normal of the boundary is given by $\boldsymbol{n}_w^2 = \boldsymbol{e}^2$. Then, from equation (3.21) and (3.22), we have

$$\boldsymbol{n} \cdot \boldsymbol{n}_w^2 = \cos(\theta) \Leftrightarrow \phi_{x_2} = \cos(\theta)|\nabla\phi|. \tag{3.25}$$

Again, we consider the two cases $0 < \theta \leq \frac{\pi}{2}$ and $\frac{\pi}{2} < \theta < \pi$. First, let $0 < \theta \leq \frac{\pi}{2}$. Then, $0 \leq \cos\theta < 1$ and $\sin^2(\theta) = 1 - \cos^2(\theta) > 0$. From equation (3.25), we know that $\phi_{x_2} > 0$, since $\cos\theta > 0$. Similar to the first case above, we can then show that

$$\phi_{x_2} = \cos(\theta)\sqrt{\phi_{x_1}^2 + \phi_{x_2}^2 + \phi_{x_3}^2} \Leftrightarrow \phi_{x_2} = \cot(\theta)\sqrt{\phi_{x_1}^2 + \phi_{x_3}^2}.$$

Now, let $\frac{\pi}{2} < \theta < \pi$. Then, $-1 < \cos\theta < 0$ and $\sin^2(\theta) = 1 - \cos^2(\theta) > 0$. From equation (3.25), we know that $-\phi_{x_2} > 0$, since $\cos\theta < 0$. We define the positive function $\tilde{c} := -\cos(\theta)$ with $\tilde{c}^2 = \cos^2(\theta)$ and the positive function $\tilde{\phi} := -\phi_{x_2}$ with $\tilde{\phi}^2 = \phi_{x_2}^2$. With these definitions and similar equivalency relations as in all the other cases, we obtain

$$\tilde{\phi} = \tilde{c}\sqrt{\phi_{x_1}^2 + \tilde{\phi}^2 + \phi_{x_3}^2} \Leftrightarrow \phi_{x_2} = \cot(\theta)\sqrt{\phi_{x_1}^2 + \phi_{x_3}^2}.$$

□

Note here, that our approach is consistent with Sussman's extension technique [Sus01] for the case that the geometry is a box, which we shall exemplify in the next subsection.

3.5.1. Connection to Sussman's extension method

The relation between the level-set function and the contact angle as given in Proposition 3.1 can be considered as a special case of the extension velocity approach by Sussman [Sus01], where the contact angle is taken into account by extrapolating the liquid interface, represented by the level-set function, into the solid. This approach requires the construction of an appropriate extension velocity $\boldsymbol{u}_{\text{ext}}$ (Fig. 3.2).

To this end, we extend ϕ by solving the equation

$$\phi_\tau + \boldsymbol{u}_{\text{ext}} \cdot \nabla\phi = 0 \tag{3.26}$$

in the solid part of the boundary. Here, $\boldsymbol{u}_{\text{ext}}$ is defined as

$$\boldsymbol{u}_{\text{ext}} = \begin{cases} \frac{\boldsymbol{n}_w + \cot(\theta)\boldsymbol{n}_2}{|\boldsymbol{n}_w + \cot(\theta)\boldsymbol{n}_2|} & \text{if } c < 0 \\ \frac{\boldsymbol{n}_w - \cot(\theta)\boldsymbol{n}_2}{|\boldsymbol{n}_w - \cot(\theta)\boldsymbol{n}_2|} & \text{if } c > 0 \\ \boldsymbol{n}_w & \text{if } c = 0 \end{cases}$$

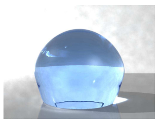
with

$$\begin{aligned} \boldsymbol{n}_1 &= -\frac{\boldsymbol{n} \times \boldsymbol{n}_w}{|\boldsymbol{n} \times \boldsymbol{n}_w|}, \\ \boldsymbol{n}_2 &= -\frac{\boldsymbol{n}_1 \times \boldsymbol{n}_w}{|\boldsymbol{n}_1 \times \boldsymbol{n}_w|}, \\ c &= \boldsymbol{n} \cdot \boldsymbol{n}_2, \end{aligned}$$

where again $\boldsymbol{n} = \nabla\phi / |\nabla\phi|$.

Let us exemplify the connection to Proposition 3.1 by the case $\boldsymbol{n}_w = (0, -1, 0)^{\mathsf{T}}$. Then, we have

$$\boldsymbol{n}_1 = -\frac{\boldsymbol{n} \times \boldsymbol{n}_w}{|\boldsymbol{n} \times \boldsymbol{n}_w|} = -\frac{\begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} \times \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}}{|\boldsymbol{n} \times \boldsymbol{n}_w|} = \frac{\begin{pmatrix} -n_3 \\ 0 \\ n_1 \end{pmatrix}}{\sqrt{(n_1^2 + n_3^2)}}$$



$$\boldsymbol{n}_2 = -\frac{\boldsymbol{n}_1 \times \boldsymbol{n}_w}{|\boldsymbol{n}_1 \times \boldsymbol{n}_w|} = -\frac{\frac{1}{\sqrt{n_1^2+n_3^2}}\begin{pmatrix}-n_3\\0\\n_1\end{pmatrix} \times \begin{pmatrix}0\\-1\\0\end{pmatrix}}{|\boldsymbol{n}_1 \times \boldsymbol{n}_w|} = -\frac{\frac{1}{\sqrt{n_1^2+n_3^2}}\begin{pmatrix}n_1\\0\\n_3\end{pmatrix}}{|\boldsymbol{n}_1 \times \boldsymbol{n}_w|} = -\frac{\begin{pmatrix}n_1\\0\\n_3\end{pmatrix}}{\sqrt{n_1^2+n_3^2}}.$$

Then,

$$c = \boldsymbol{n} \cdot \boldsymbol{n}_2 = \frac{(-n_1^2 - n_3^2)}{\sqrt{n_1^2+n_3^2}} < 0,$$

and

$$\begin{aligned}
\boldsymbol{u}_{\text{ext}} &= \frac{\boldsymbol{n}_w + \cot(\theta)\boldsymbol{n}_2}{|\boldsymbol{n}_w + \cot(\theta)\boldsymbol{n}_2|} = \frac{\begin{pmatrix}0\\-1\\0\end{pmatrix} - \frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}\begin{pmatrix}n_1\\0\\n_3\end{pmatrix}}{|\boldsymbol{n}_w + \cot(\theta)\boldsymbol{n}_2|} = \frac{\begin{pmatrix}-\frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}n_1\\-1\\-\frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}n_3\end{pmatrix}}{|\boldsymbol{n}_w + \cot(\theta)\boldsymbol{n}_2|} \\
&= \frac{\begin{pmatrix}\vdots\end{pmatrix}}{\sqrt{\frac{\cot^2(\theta)n_1^2}{n_1^2+n_3^2} + 1 + \frac{\cot^2(\theta)n_3^2}{n_1^2+n_3^2}}} = \frac{\begin{pmatrix}\vdots\end{pmatrix}}{\sqrt{\frac{\cot^2(\theta)n_1^2+n_1^2+n_3^2+\cot^2(\theta)n_3^2}{n_1^2+n_3^2}}} \\
&= \sqrt{n_1^2+n_3^2}\frac{\begin{pmatrix}\vdots\end{pmatrix}}{\sqrt{(1+\cot^2(\theta))n_1^2 + (1+\cot^2(\theta))n_3^2}} = \sqrt{n_1^2+n_3^2}\frac{\begin{pmatrix}\vdots\end{pmatrix}}{\sqrt{(1+\cot^2(\theta))(n_1^2+n_3^2)}} \\
&= \frac{1}{\sqrt{1+\cot^2(\theta)}}\begin{pmatrix}-\frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}n_1\\-1\\-\frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}n_3\end{pmatrix}.
\end{aligned}$$

We compute

$$\begin{aligned}
\boldsymbol{u}_{\text{ext}} \cdot \nabla\phi = \boldsymbol{u}_{\text{ext}} \cdot \boldsymbol{n}|\nabla\phi| \\
&= -\frac{|\nabla\phi|}{\sqrt{1+\cot^2(\theta)}}\left(\frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}n_1^2 + n_2 + \frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}n_3^2\right) \\
&= -\frac{|\nabla\phi|}{\sqrt{1+\cot^2(\theta)}}\left(\frac{\cot(\theta)}{\sqrt{n_1^2+n_3^2}}(n_1^2+n_3^2) + n_2\right) \\
&= -\frac{|\nabla\phi|}{\sqrt{1+\cot^2(\theta)}}\left(\cot(\theta)\sqrt{n_1^2+n_3^2} + n_2\right) \\
&= -\frac{|\nabla\phi|}{\sqrt{1+\cot^2(\theta)}}\left(\cot(\theta)\frac{\sqrt{\phi_{x_1}^2+\phi_{x_3}^2}}{|\nabla\phi|} + \frac{\phi_{x_2}}{|\nabla\phi|}\right)
\end{aligned}$$

$$= -\frac{1}{\sqrt{1+\cot^2(\theta)}} \left(\cot(\theta)\sqrt{\phi_{x_1}^2 + \phi_{x_3}^2} + \phi_{x_2}\right).$$

Thus, at steady state $\phi_\tau = 0$, the extension equation (3.26) is fulfilled if $\boldsymbol{u}_{\text{ext}} \cdot \nabla\phi = 0$, so that

$$\phi_{x_2} = -\sqrt{\phi_{x_1}^2 + \phi_{x_3}^2}\cot(\theta),$$

which is exactly what Proposition 3.1 suggests for this case. This shows the connection of our approach to Sussman's extension velocity method.

All in all, Proposition 3.1 allows us to write the contact angle θ as a boundary condition of the level-set function. Now, the challenging part is the model for the dynamic contact angle $\theta = \theta_d$. In the next section, we present the two approaches for the modeling of the dynamic contact angle, which we use in this thesis.

3.6. Models for the dynamic contact angle

In this section we present two models for the dynamic contact angle. The first is a model by Yokoi et al. [YVHH09], which is an extension of the well-known Tanner's law [Tan79]. This model is developed for and based on a single droplet impact experiment only. The second model is a part of Shikhmurzaev's interface formation theory [Shi07], which offers a whole framework to explain the formation and disappearance of interfaces.

3.6.1. The dynamic contact angle model by Yokoi et al.

Yokoi et al. [YVHH09] propose a dynamic contact angle model, which combines Tanner's law (2.4) with a static advancing and receding contact angle, since Tanner's law holds for small capillary numbers only. In the combined model, similar to what is observed in experiments, the contact angle tends to both the limit of a maximum dynamic advancing angle θ_{mda} as the dimensionful contact line speed u_{cl} increases and the limit of a minimum dynamic receding angle θ_{mdr} as u_{cl} decreases:

$$\theta_d = \begin{cases} \min\left\{\theta_e + \left(\frac{\mu u_{\text{cl}}}{\sigma k_a}\right)^{\frac{1}{3}}, \theta_{\text{mda}}\right\} & \text{if } u_{\text{cl}} \geq 0 \\ \max\left\{\theta_e + \left(\frac{\mu u_{\text{cl}}}{\sigma k_r}\right)^{\frac{1}{3}}, \theta_{\text{mdr}}\right\} & \text{if } u_{\text{cl}} < 0, \end{cases} \tag{3.27}$$

where θ_e denotes the equilibrium contact angle. The material-related parameters k_a and k_r are adjusted to fit the numerical results to the results obtained from measurements. Furthermore, the stress singularity is circumvented by inducing numerical slip for the velocity at the contact line.

3.6.2. The dynamic contact angle model by Shikhmurzaev

The second model for the dynamic contact angle at small capillary number is a reduced version of Shikhmurzaev's interface formation model [Shi07]. The full model accounts for

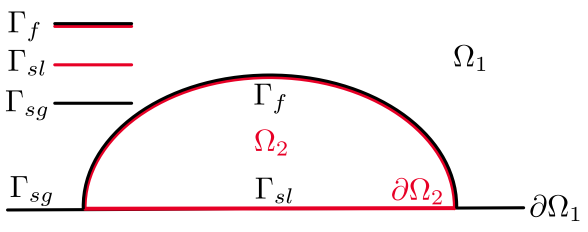


Figure 3.3: Sketch of the boundary definitions for the interface formation model: The black line marks $\partial\Omega_1$, the red line $\partial\Omega_2$. Then, we have the free liquid-gas interface $\Gamma_f := \partial\Omega_1 \cap \partial\Omega_2$, the solid-liquid interface $\Gamma_{sl} := \partial\Omega_2 \setminus \Gamma_f$ and the solid-gas interface $\Gamma_{sg} := \partial\Omega_1 \setminus \Gamma_f$.

different classes of flows, where interfaces are formed or destroyed. The equations, which capture the surface tension relaxation process and have to be solved on the surface itself, are derived from mass, momentum and energy conversation. For the case of small capillary and Reynolds numbers, we can analyze them as a local problem whose solution can be incorporated into various types of global flow problems. Here, lots of experimental works simplify the verification of numerical results.

We will first describe the full interface formation model for the sake of completeness and then consider its simplified counterpart.

The full interface formation model

In the following we consider the full interface formation model in its simplest irreducible form, which is derived from the macroscopic irreversible thermodynamics of systems with interfaces. These equations are described in great detail in [Shi07], and we refer the reader to this book for generalizations of the model.

Let us define an open set $\Omega = \Omega_1 \cup \Omega_2 \cup \Gamma_f \subset \mathbb{R}^3$. The gas domain Ω_1 and the liquid domain Ω_2 as well as the free liquid-gas interface $\Gamma_f := \partial\Omega_1 \cap \partial\Omega_2$ depend on time. Furthermore, we define the solid-liquid interface $\Gamma_{sl} := \partial\Omega_2 \setminus \Gamma_f$ and the solid-gas interface $\Gamma_{sg} := \partial\Omega_1 \setminus \Gamma_f$; compare Figure 3.3.

Then, we write the Navier-Stokes equations for an inviscid gas at constant pressure

$$p = \begin{cases} p_g & \text{in } \Omega_1 \\ p & \text{in } \Omega_2 \end{cases}$$

as

$$\begin{aligned} \rho\left(\partial_t \boldsymbol{u} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\right) &= -\nabla p + \mu\Delta\boldsymbol{u} + \rho\boldsymbol{g} && \text{in } \Omega_2 \\ \nabla\cdot\boldsymbol{u} &= 0 && \text{in } \Omega_2, \end{aligned} \tag{3.28}$$

which only have to be solved in the liquid part $\Omega_2 \subset \Omega$.

Due to the dynamically passive gas, which results in a constant prescribed stress on the gas-facing side of Γ_f, we have to consider the interface formation model for the liquid-facing side of Γ_f only. Here, the interface formation equations act as boundary conditions for the bulk equations (3.28). These consist in part of the standard interface conditions

$$\frac{\partial \boldsymbol{f}}{\partial t} + \boldsymbol{v}^s \cdot \nabla \boldsymbol{f} = 0 \tag{3.29}$$

$$p_g - p + \mu\boldsymbol{n}\cdot\left[\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^\mathsf{T}\right]\cdot\boldsymbol{n} = \sigma\nabla\cdot\boldsymbol{n} \tag{3.30}$$

$$\mu\boldsymbol{n}\cdot\left[\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^\mathsf{T}\right]\cdot(\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T}) + \nabla\sigma = 0, \tag{3.31}$$

which we already considered in Section 3.1 of this chapter. The first equation describes the movement of the interface, while the last two equations are the normal and tangential projection of the stress balance equation (3.7). However, the difference to our standard model (3.11) is that the normal velocity is no longer assumed to be equal to the normal velocity at the interface, i.e. we no longer have

$$(\boldsymbol{u} - \boldsymbol{v}_s) \cdot \boldsymbol{n} = 0$$

as derived in equations (3.2) and (3.4). This derivation had been done under the assumption that the interface's role as a source of sink of mass could be neglected. Instead, we now introduce a further equation, which is responsible for the exchange of mass between the bulk and the surface phase, i.e.

$$\rho(\boldsymbol{u} - \boldsymbol{v}^s) \cdot \boldsymbol{n} = \frac{\rho^s - \rho^s_e}{\tau}, \tag{3.32}$$

where ρ^s is the surface density, ρ^s_e is the equilibrium surface density and τ is a phenomenological constant. This equation can be interpreted as a boundary condition for the normal component of the bulk velocity

$$\boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{v}^s \cdot \boldsymbol{n} + \frac{\rho^s - \rho^s_e}{\tau\rho}.$$

Additionally, we formulate a mass balance equation for the surface density,

$$\frac{\partial \rho^s}{\partial t} + \nabla \cdot (\rho^s \boldsymbol{v}^s) = -\frac{\rho^s - \rho^s_e}{\tau}, \tag{3.33}$$

which accounts for the mass exchange at the interface. The difference between the tangential component of the velocity $\boldsymbol{u}_\parallel = \boldsymbol{u}(\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T})$ and the velocity at the interface $\boldsymbol{v}^s_\parallel = \boldsymbol{v}^s(\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^\mathsf{T})$ is proportional to the surface tension gradient, i.e.

$$(1 + 4\alpha\beta)\nabla\sigma = 4\beta(\boldsymbol{v}^s_\parallel - \boldsymbol{u}_\parallel), \tag{3.34}$$

where α and β are phenomenological constants. In order to close the model, we need the further equation

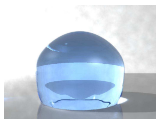
$$\gamma\,(\rho^s_0 - \rho^s) = \sigma, \tag{3.35}$$

which Shikhmurzaev terms the 'surface equation of state'. This equation connects the compression of the surface phase γ with a decrease in surface tension.[1] Again, γ and ρ^s_0 are phenomenological material constants.

In addition to the equations describing the state of the interface (3.29)-(3.35), boundary conditions for the Navier-Stokes equations on the liquid-facing side of the solid-liquid interface Γ_{sl} are needed,

$$\boldsymbol{v}^s \cdot \boldsymbol{n} = 0, \tag{3.36}$$

[1]This is the simplest linear form of the surface equation of state for processes where the surface density variation is bounded by the equilibrium surface densities [Shi07, p.200].



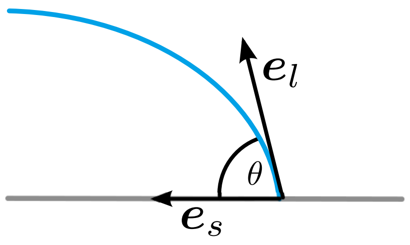


Figure 3.4: Sketch of the moving contact line with the definition of the contact line's unit normals $\boldsymbol{e}_l$ and $\boldsymbol{e}_s$.

$$\mu \boldsymbol{n} \cdot \left[\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^{\mathsf{T}}\right] \cdot (\boldsymbol{I} - \boldsymbol{n}\boldsymbol{n}^{\mathsf{T}}) + \frac{1}{2}\nabla\sigma = \beta_{sl}\boldsymbol{u}_{\|}, \tag{3.37}$$

$$\rho(\boldsymbol{u} - \boldsymbol{v}^s) \cdot \boldsymbol{n} = \frac{\rho^s - \rho^s_{sl,e}}{\tau_{sl}}, \tag{3.38}$$

$$\frac{\partial \rho^s}{\partial t} + \nabla \cdot (\rho^s \boldsymbol{v}^s) = -\frac{\rho^s - \rho^s_{sl,e}}{\tau_{sl}}, \tag{3.39}$$

$$\boldsymbol{v}^s_{\|} = \frac{1}{2}\boldsymbol{u}_{\|} + \alpha_{sl}\nabla\sigma, \tag{3.40}$$

$$\gamma\,(\rho^s_0 - \rho^s) = \sigma. \tag{3.41}$$

Here, $\rho^s_{sl,e}$ is the equilibrium surface density in the solid-liquid interface and α_{sl}, β_{sl} and τ_{sl} are phenomenological constants. For a simple start, Shikhmurzaev [Shi07, p.201] proposes to set them equal to their counterparts α, β and τ in the liquid-gas interface.

Note that the boundary and interface conditions formulated above are partial differential equations, which have to be solved for the surface variables on the interface itself. For their solution boundary conditions are required. At a contact line where a free surface meets the solid surface, we make the following definition: Since we can approach the contact line along the free surface Γ_f on the one hand and along the solid-liquid interface Γ_{sl} on the other hand, we denote the limiting values of these approaches by G and S, respectively. Then, we have a surface mass balance equation at the contact line

$$(\rho^s \boldsymbol{v}^s_{\|} - \boldsymbol{u}_{cl})|_G \cdot \boldsymbol{e}_l + (\rho^s \boldsymbol{v}^s - \boldsymbol{u}_{cl})|_S \cdot \boldsymbol{e}_s = 0, \tag{3.42}$$

along with the Young equation

$$\sigma|_S = \sigma_{sg} + \sigma|_G \cos\theta_d, \tag{3.43}$$

which balances the tangential projections of the forces due to surface tensions acting on the contact line [SS12b]. Here, $\boldsymbol{e}_l$ and $\boldsymbol{e}_s$ denote the unit normals of the contact line, which are tangential to the liquid or the solid surface (Fig. 3.4), and $\boldsymbol{u}_{cl}$ is the contact line velocity.

In equation (3.43), the solid-gas surface tension does not have much influence on the dynamics of droplet impact and becomes relevant at contact angles close to 180° only [Shi07, SS12b]. Note also, that the involved surface tensions are no longer constant, thereby are coupled to the distribution of surface parameters along the interfaces and hence to the bulk flow itself [Shi07, p.209].

The reduced interface formation model

For a reduction of the full interface formation model at small capillary numbers Ca, we again follow the description in [Shi07]. There, the flow domain is split into two asymptotic regions, and in both the limit $Ca \to 0$ is studied analytically. In the inner asymptotic region to leading order in Ca, the dynamic contact angle and the dimensionless contact-line speed V are related by

$$\cos(\theta_e) - \cos(\theta_d) = \frac{2V\left[\cos(\theta_e) - \tilde{\sigma}_{sg} + (1-\rho^s_G)^{-1}(1+\rho^s_G u_{(12)}(\theta_d, k_\mu))\right]}{V + [V^2 + 1 + (\cos(\theta_e) - \tilde{\sigma}_{sg})(1-\rho^s_G)]^{\frac{1}{2}}} \tag{3.44}$$

with θ_e the equilibrium contact angle, k_μ the gas-to-liquid viscosity ratio, and $\rho^s_G = \rho^s_e/\rho^s_{(0)}$ the dimensionless surface density. Here, ρ^s_e is the equilibrium surface density and $\rho^s_{(0)}$ the phenomenological constant describing the surface density for zero surface tension. Furthermore, $\tilde{\sigma}_{sg} = \sigma_{sg}/\sigma$ is the dimensionless surface tension in the gas-solid interface obtained by division with the equilibrium liquid-gas surface tension σ.

The dimensionless contact line speed in equation (3.44) is given by

$$V = u_{\mathrm{cl}}\sqrt{\frac{\tau\beta}{\gamma\rho^s_0(1+4\alpha\beta)}},$$

and we introduce the scaling factor Sc by

$$Sc = \sqrt{\frac{\sigma^2\tau\beta}{\mu^2\gamma\rho^s_0(1+4\alpha\beta)}}. \tag{3.45}$$

Therefore,

$$V = \frac{u_{\mathrm{cl}}\mu}{\sigma}Sc. \tag{3.46}$$

Here, σ is the equilibrium surface tension, α and β are phenomenological constants depending on the 'state of the interface', γ is a phenomenological constant describing the compressibility of the fluid, and τ is the surface tension relaxation time which can be treated as a material constant. Thus, Sc depends on the material properties of the fluid and the interface.

The radial velocity $u_{(12)}(\theta_d, k_\mu)$ in equation (3.44) must be derived from the solution in the outer region. In particular, in the outer region to leading order in Ca the free-surface curvature becomes zero and we obtain a flow problem in a wedge [Shi07]. The solution to this problem is given by Moffatt in [Mof64] as

$$u_{(12)}(\theta_d, 0) = \frac{\sin\theta_d - \theta_d\cos\theta_d}{\sin\theta_d\cos\theta_d - \theta_d}. \tag{3.47}$$

If the viscosity of the gas phase is taken into account, Moffatt's solution becomes

$$u_{(12)}(\theta_d, k_\mu) = \frac{(\sin\theta_d - \theta_d\cos\theta_d)K(\theta_2) - k_\mu(\sin\theta_2 - \theta_2\cos\theta_2)K(\theta_d)}{(\sin\theta_d\cos\theta_d - \theta_d)K(\theta_2) + k_\mu(\sin\theta_2\cos\theta_2 - \theta_2)K(\theta_d)}, \tag{3.48}$$

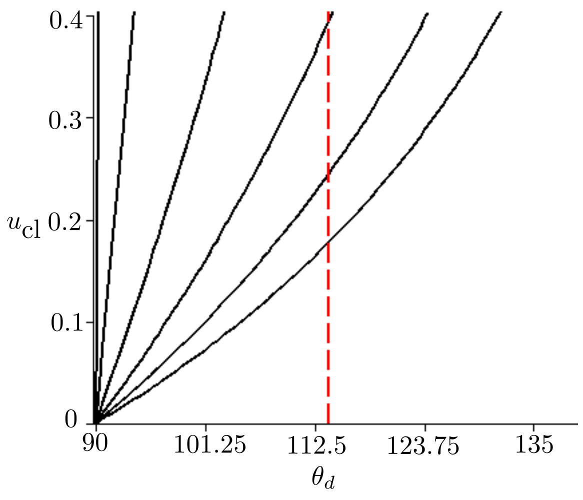


Figure 3.5: Dynamic contact angle [°] vs. dimensionful contact-line speed [m/s] for different values of Sc. From left to right: $Sc =$ 0.1, 1.5, 5.5, 11.5, 18.5, 25.5. The red dashed line corresponds to the maximum of the dynamic advancing contact angle of 114° observed in the experiments.

with $\theta_2 = \pi - \theta_d$ and $K(\theta) = \theta^2 - \sin^2\theta$ as described in [Shi07].

Alternatively, $u_{(12)}(\theta_d, k_\mu)$ in (3.44) can be replaced by the inner limit of the outer solution, i.e. by a numerically computed far field velocity sufficiently close to the contact line. This alters the dynamic contact angle for the same contact line speed and is exactly what is observed in laboratory experiments as the non-local influence of the flow field/geometry on the dynamic contact angle, i.e. the hydrodynamic assist of wetting.

An example for the choice of Sc

Let us demonstrate how the dimensionless parameter Sc influences the results of equation (3.44). As an example, we consider a droplet of distilled water which impacts on a silicon wafer onto which hydrophobic silane has been grafted; an experiment which we consider in detail in Subsection 6.4.2. The equilibrium contact angle of the substrate with distilled water is 90°. The remaining parameters are chosen according to [Shi07] as $\rho^s_G = 0.9$ and $\sigma_{sg} = 0$. Furthermore, $u_{(12)}(\theta_d, 0)$ is determined by Moffatt's solution (3.47).

Then, equation (3.44) can be resolved with respect to positive V as given in [Shi07]. Since we are interested in the relationship between the dynamic contact angle and the dimensionful contact line velocity, we use equation (3.46) to plot the dimensionful speed-angle relationship for different values of Sc. The effect of the variation of Sc is shown in Figure 3.5: We see that the contact angle increases faster from its static value $\theta_e = 90°$ the more we increase Sc. The horizontal straight red line indicates the maximum dynamic advancing contact angle of 114° determined from the experiments. For $Sc = 11.5$ this value is reached at a maximum contact line speed of about $u_{\mathrm{cl}} = 0.4\,\mathrm{m/s}$ which explains our choice of Sc in Subsection 6.4.2.

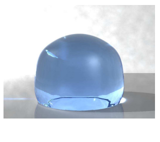
4. Numerical Modeling of Free Surface Flows with Dynamic Contact Angles

So far we have addressed and sufficiently answered the question of mathematical modeling. This chapter deals with the transfer of the mathematical model onto a discrete grid, which is already a demanding task. Additionally, the discretized model provides us with further challenges on its own, which, in our case, is most prominently the loss of mass of our currently implemented level-set (LS) method. In this chapter, the problem of mass loss is tackled with two volume correction methods. However, both methods are an additional source of possible numerical errors. This will become apparent in Chapter 6, where both correction methods fail in a simple advection test. Therefore, we discuss the CLSVOF method in the next chapter, whose demanding implementation presents us with much more reliable and faster numerical results. Note, that the discussion of the CLSVOF method is done in a separate chapter due to its complexity. Furthermore, its discretization presupposes a discretization of the Navier-Stokes equations and of the LS method.

This chapter starts with the discretization of the Navier-Stokes equations (3.20) in space and time with special emphasis on the LS method. Then, we discuss the implementation of the contact angle boundary condition and present two methods for a better conservation of mass within the LS method. Furthermore, we present the discretization of the two contact angle models. In a last step, we discuss the parallelization strategy of NaSt3DGPF in general and of our new additions to the code in particular.

Our flow solver NaSt3DGPF has been developed over many years and many people have been involved in its development. Therefore, many aspects of the flow solver that are described in this chapter for the sake of completeness are not my own work, but belong in parts to others. Thus, the implemented LS method as well as the global volume correction are due to R. Croce [Cro02, CGS04, CGS10], whereas the local volume correction and the implemented contact angle models are new additions to NaSt3DGPF.

Our discretization strategy in space and time can be summarized as follows. We discretize the Navier-Stokes equations with finite differences on a staggered uniform grid and use an explicit second-order Adams-Bashforth time integration scheme. The solution process is based on the well-known projection method: First, an intermediate velocity field $\boldsymbol{u}^*$, which may not be divergence free, is advanced by the Adams-Bashforth time scheme; second, we compute a correction ∇p^{n+1} of the intermediate velocity field by the pressure Poisson equation which leads to a divergence free velocity field $\boldsymbol{u}^{n+1}$. Thus, we treat the pressure implicitly and solve the Poisson equation by a Jacobi-preconditioned conjugate gradient method. A fifth-order weighted essentially non-oscillatory (WENO) scheme is used for the discretization of the convective transport of the Navier-Stokes equations (3.20) as well as for the LS transport (3.16). The diffusion term is computed by using second-order central differences.



4.1. Discretization in time

We write the Navier-Stokes equations (3.20) in a time-discrete notation as

$$\begin{aligned}\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^n}{\delta t}+\frac{\nabla p^n}{\rho(\phi^n)} &= -(\boldsymbol{u}^n\cdot\nabla)\boldsymbol{u}^n+\boldsymbol{g}+\frac{1}{\rho(\phi^n)}\left(\nabla\cdot(\mu(\phi^n)\boldsymbol{S}^n)-\sigma\kappa(\phi^n)\delta(\phi^n)\nabla\phi^n\right)\\ \nabla\cdot\boldsymbol{u}^{n+1} &= 0,\end{aligned}$$

where n indicates the time step. Then, the Chorin projection method is given by two successive steps [CGS04].

1. Solve the momentum equations for an intermediate velocity field $\boldsymbol{u}^*$

$$\frac{\boldsymbol{u}^*-\boldsymbol{u}^n}{\delta t}=-(\boldsymbol{u}^n\cdot\nabla)\boldsymbol{u}^n+\boldsymbol{g}+\frac{1}{\rho(\phi^n)}\left(\nabla\cdot(\mu(\phi^n)\boldsymbol{S}^n)-\sigma\kappa(\phi^n)\delta(\phi^n)\nabla\phi^n\right). \quad (4.1)$$

2. Project the vector field $\boldsymbol{u}^*$ onto a divergence-free vector field $\boldsymbol{u}^{n+1}$

$$\boldsymbol{u}^*=\boldsymbol{u}^{n+1}+\delta t\frac{\nabla p^{n+1}}{\rho(\phi^{n+1})} \quad (4.2)$$

$$\nabla\cdot\boldsymbol{u}^{n+1}=0. \quad (4.3)$$

   Application of the divergence operator to (4.2) results in a Poisson equation for the pressure from which we can obtain the pressure implicitly by solving

$$\nabla\cdot\left(\frac{1}{\rho(\phi^{n+1})}\nabla p^{n+1}\right)=\frac{1}{\delta t}\nabla\cdot\boldsymbol{u}^*. \quad (4.4)$$

3. Compute the velocity field $\boldsymbol{u}^{n+1}$ of the next time step by equation (4.2).

For the solution of the Poisson equation, boundary conditions for the pressure are required. Therefore, we project equation (4.2) onto the outer unit normal of the domain's boundary

$$\left.\frac{\partial p^{n+1}}{\partial\boldsymbol{n}}\right|_\Gamma=\frac{\rho}{\delta t}\left(\boldsymbol{u}^*_\Gamma-\boldsymbol{u}^{n+1}_\Gamma\right)\cdot\boldsymbol{n}.$$

Thereby, homogeneous Neumann boundary conditions for the pressure are induced if we require $\boldsymbol{u}^*|_\Gamma=\boldsymbol{u}^{n+1}|_\Gamma$ for the intermediate velocity field.

For the discretization of time, our flow solver provides explicit Euler as well as second-order Adams-Bashforth or Runge-Kutta schemes. The extension of the projection method towards these higher order methods is straightforward. If not stated otherwise, we employ the Admas-Bashforth time-integration scheme in this thesis.

For the solution of the pressure Poisson equation (4.4), we use a Jacobi-preconditioned conjugate gradient method in most applications. Additionally, via a PETSc interface, we are able to use a sophisticated algebraic multigrid method (AMG) implemented by Metsch [GMOS06, GMS08]. The AMG is especially valuable for challenging highly resolved flow problems, where the solution of the Poisson equation takes up a large amount of computing time. Here, the AMG is taken as a preconditioner for the conjugate gradient method, which reduces the number of iterations considerably and works in parallel. However, the AMG is not always the method of choice for free surface flows, since its setup takes time and has to be redone every so often when the interface position deviates too much from its position during the previous setup.

4.1.1. Level-set transport and reinitialization equation

Additionally to the Navier-Stokes equations, we have to discretize the LS transport equation (3.16) in time, for which we employ a second-order Adams-Bashforth or third-order Runge-Kutta scheme. In the following, we describe a discretization with the Adams-Bashforth scheme only and refer to [Cro10] for details on the Runge-Kutta one.

In general, a straightforward discretization of the transport equation will result in a LS function which no longer fulfills the signed distance property. However, this property considerably simplifies a stable implementation of surface tension and curvature. Therefore, our transport step

$$\phi^* = \phi^n + \frac{\delta t}{2}\left(3\boldsymbol{u}^n \cdot \nabla\phi^n - \boldsymbol{u}^{n-1} \cdot \nabla\phi^{n-1}\right)$$

only leads to a preliminary function ϕ^*, which needs to be reinitialized after each transport step to recover its signed distance property $|\nabla\phi^{n+1}| = 1$ without disturbing the zero level-set. In order to generate the appropriate signed-distance function $\phi^{n+1}(\boldsymbol{x})$ with the same zero level-set as $\phi^*(\boldsymbol{x})$, we solve the pseudo-transient Hamilton–Jacobi problem

$$\phi^*_\tau = \operatorname{sgn}(\phi_0)(1 - |\nabla\phi^*|) \tag{4.5}$$

with initial value $\phi_0 = \phi^*(\boldsymbol{x})$ and pseudo time τ. Again, this equation is discretized in time by a third-order Runge-Kutta scheme; see [CGS04, CGS10, Cro10] for details.

For reasons of numerical stability, we employ a regularized signum function $\mathrm{S}(\cdot)$ defined by

$$\mathrm{S}(\phi^*) = \frac{\phi^*}{\sqrt{(\phi^*)^2 + |\nabla\phi^*|^2(\delta x^2)}}. \tag{4.6}$$

In this case, the Hamilton–Jacobi equation changes to

$$\phi^*_\tau = \mathrm{S}(\phi^*)(1 - |\nabla\phi^*|), \tag{4.7}$$

so that the signum function needs to be newly computed in every pseudo time step of the reinitialization equation.

4.2. Discretization in space

Here and in the following, we denote by cells the rectangular subdomains $[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}] \times [y_{j-\frac{1}{2}}, y_{j+\frac{1}{2}}] \times [z_{k-\frac{1}{2}}, z_{k+\frac{1}{2}}]$, and we define the discrete computational domain Ω^h as a union of such cells. For $\{i, j, k\} \in \mathbb{Z}$ we use the notation

$$\Delta x_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}},$$

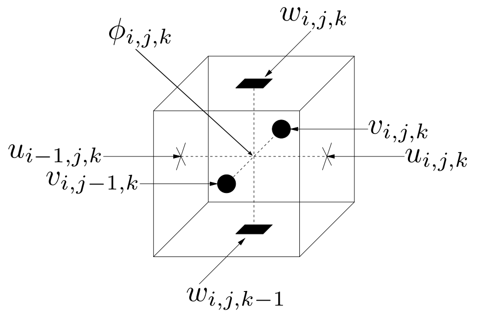


Figure 4.1: On the staggered grid, the LS function ϕ is discretized at the cell centers and the velocity is discretized at the face centers of the grid.

and Δy_j as well as Δz_k are defined analogously. Furthermore, 'ghost cells' or 'boundary cells' denote the up to three additional strips of cells attached to Ω_h, which are needed for the discretization of large finite difference stencils and boundary conditions.

The Navier-Stokes equations are resolved on a regular staggered grid by a finite difference discretization. On the staggered grid the pressure p and the LS function ϕ are discretized at the center of the cells, while the velocities u, v and w are discretized at the center of the cell faces (Fig. 4.1). This discretization leads to a strong coupling between pressure and velocities and therefore avoids the occurrence of unphysical oscillations in the pressure. The velocity, pressure and LS values are defined at the following nodes:

- $[u_{i,j,k}]$ is defined on $[x_{i+\frac{1}{2}}, y_j, z_k]$ at the right side surface center.
- $[v_{i,j,k}]$ is defined on $[x_i, y_{j+\frac{1}{2}}, z_k]$ at the back side surface center.
- $[w_{i,j,k}]$ is defined on $[x_i, y_j, z_{k+\frac{1}{2}}]$ at the upper side surface center.
- $[p_{i,j,k}$ and $\phi_{i,j,k}]$ are defined on $[x_i, y_j, z_k]$ at the cell center.

As described in detail in [Cro02, CGS04], the first and second-order derivatives of the viscous terms $\nabla \cdot (\mu(\phi^n)\boldsymbol{S}^n$ are discretized with central differences. Therefore, the LS function needs to be evaluated at the velocity nodes for which a third-order Lagrange interpolation scheme is used.

NaSt3DGPF provides several schemes for the spatial discretization of the convective terms, including Donor-Cell (1st/2nd order), QUICK (2nd-order), HLPA (2nd-order), SMART (2nd-order) and VONOS (2nd/3rd-order) [Cro02]. Throughout this thesis, we employ a fifth-order WENO scheme for the treatment of all convective terms, including the transport of the LS function; see [Cro02, CGS04] for details.

4.2.1. Smoothing and discretization of surface variables

Within the flow solver NaSt3DGPF, the discontinuity of the density at the interface is not treated directly. Instead, a smoothing scheme is applied, where the interface is considered to have a fixed thickness ϵ proportional to the spatial mesh size h. This thickness is a tunable computational parameter and can be adapted to the specific numerical problem.

With this parameter, the discrete density $\rho(\phi_h)$ and the discrete viscosity $\mu(\phi_h)$ can be replaced by

$$\rho^\varepsilon(\phi_h) = \rho_2 + (\rho_1 - \rho_2)H^\varepsilon(\phi_h) \quad \text{and} \quad \mu^\varepsilon(\phi_h) = \mu_2 + (\mu_1 - \mu_2)H^\varepsilon(\phi_h) \tag{4.8}$$

with a smoothed Heaviside function

$$H^\varepsilon(\phi_h) = \begin{cases} 0 & \text{if } \phi_h < -\varepsilon \\ \frac{1}{2}(1 + \frac{\phi_h}{\varepsilon} + \frac{1}{\pi}\sin(\frac{\pi\phi_h}{\varepsilon})) & \text{if } |\phi_h| \leq \varepsilon \\ 1 & \text{if } \phi_h > \varepsilon. \end{cases} \tag{4.9}$$

The associated smoothed delta functional is given by

$$\delta^\varepsilon(\phi_h) = \partial_{\phi_h} H^\varepsilon = \begin{cases} \frac{1}{2\varepsilon}(1 + \cos(\frac{\pi\phi_h}{\varepsilon})) & \text{for } |\phi_h| < \varepsilon \\ 0 & \text{else.} \end{cases} \tag{4.10}$$

For further details on our smoothing approach and its order of approximation see [CGS10].

Since the unit normal on the interface is defined by $\boldsymbol{n} = \frac{\nabla\phi_h}{|\nabla\phi_h|}$, the curvature can be directly computed by

$$\begin{aligned} \kappa &= \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \\ &= \frac{\phi_{xx}(\phi_y^2 + \phi_z^2) + \phi_{yy}(\phi_x^2 + \phi_z^2) + \phi_{zz}(\phi_x^2 + \phi_y^2) - (2\phi_x\phi_y\phi_{xy} + 2\phi_x\phi_z\phi_{xz} + 2\phi_y\phi_z\phi_{yz})}{\left(\phi_x^2 + \phi_y^2 + \phi_z^2\right)^{\frac{3}{2}}}, \end{aligned}$$

where we write ϕ instead of ϕ_h for readability. For the discretization of the curvature, we employ central differences for the first order as well as for the second order derivatives.

4.2.2. Level-set transport and reinitialization equation

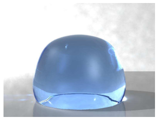
We discretize the convective terms in the transport equation (3.16) as well as in the Hamilton-Jacobi equation (4.5) by a a fifth-order WENO scheme in space. We do not need boundary conditions for the solution of the Hamilton-Jacobi equation, since its characteristics point outward from the interface and carry information near the interface along the normal direction away from the zero level-set to the rest of the domain. A boundary condition is only needed at the contact line, where the interface is in direct contact with the substrate.

For the solution of the Hamilton-Jacobi equation (4.5), we compute a solution in the ε-neighborhood of the zero level-set only. Due to the motion of the characteristics, the number of artificial time steps can be chosen approximately of the order of ε.

4.2.3. Discretization of the contact angle boundary condition

The discretization of the contact angle boundary condition (3.23) is very similar to the discretization of the standard Neumann boundary condition for the LS function. We exemplify this at the wall $y = 0$, where equation (3.23) becomes

$$\phi_y = -\cot(\theta)\sqrt{\phi_x^2 + \phi_z^2}. \tag{4.11}$$



On the staggered grid (Fig. (4.1)), the LS values are discretized at the cell center. Then, with grid cells denoted by integers (i, j, k),

$$\frac{\phi_{i,j,k} - \phi_{i,j-1,k}}{\delta y_j} = -\cot(\theta)\sqrt{\phi^2_{x_{i,j,k}} + \phi^2_{z_{i,j,k}}}, \tag{4.12}$$

where δy_j is the mesh width. The derivatives $\phi_{x_{i,j,k}}$ and $\phi_{z_{i,j,k}}$ can be discretized by central differences. This equation can be solved for the staggered grid's ghost cell value $\phi_{i,j-1,k}$, which gives the required boundary condition for ϕ.

The values for the contact angle θ are computed by the discretized dynamic contact angle models of Yokoi et al. (3.27) or Shikhmurzaev (3.44), which we will discuss subsequently.

4.3. Time step control

In general, the Courant-Friedrichs-Lewy (CFL) condition is necessary for the stability of the numerical solution of certain partial differential equations and arises in the numerical analysis of explicit time integration schemes. In our case, the explicit treatment of the time-advancement of the momentum equations (4.1) yields a time step restriction for the convective terms, as well as for the diffusive terms, surface tension and volume forces. This restriction guarantees the numerical stability of the method by forcing discrete information to travel no further than one grid cell per time step.

As in [CGS04], we define $u_{\max} = \max_{i,j,k}|u_{i,j,k}|$, $\kappa_{\max} = \max_{i,j,k}|\kappa_{i,j,k}|$ and

$$V := \max\left\{\frac{\mu_1}{\rho_1}, \frac{\mu_2}{\rho_2}\right\} \cdot \left(\frac{2}{(\delta x)^2} + \frac{2}{(\delta y)^2} + \frac{2}{(\delta z)^2}\right).$$

Then, the time step restriction δt^u for the velocity component u is

$$\delta t^u \leq 2C_u$$

with

$$C_u := \left(\left(\frac{u_{\max}}{\delta x} + V\right) + \sqrt{\left(\frac{u_{\max}}{\delta x} + V\right)^2 + \frac{4|g_x|}{\delta x} + \frac{8\kappa_{\max}\sigma}{\alpha(\rho_2 + \rho_1)(\delta x)^2}}\right)^{-1},$$

where we refer to [Cro02, CGS04, CGS10] for further details on the derivation of this restriction. Here, $\alpha = \frac{\varepsilon}{\delta x}$ is a number associated with the amount of smoothing near the interface; see Section 4.2.1.

Analogous definitions for δt^v and δt^w for the velocities in the other directions amount to the overall time step restriction

$$\delta t \leq 2\xi \min_{\Omega}(C_u, C_v, C_w) \tag{4.13}$$

with a safety factor $\xi \in (0, 1]$.

4.4. Volume correction

An important issue – especially for LS methods – is mass conservation. In the reinitialization step, the LS function ϕ^* is replaced by a smoother, less distorted function ϕ^{n+1} with the same zero level-set. Thereby, we introduce numerical diffusion to the solution which leads to difficulties with mass conservation. In order to remedy this problem, we implement a global and a local volume correction method.

In this section, we write ϕ instead of ϕ_h for readability. However, we always mean the discretized LS function.

4.4.1. Global volume correction

The global volume correction described in [CGS10, Cro10] aims at correcting the loss of mass after the reinitialization of the LS function by finding a new contour line of ϕ, which is different from the zero contour, but fits better with the initial mass of both fluid phases.

Let ϕ^{n+1} denote the LS function at time step $t = n+1$ after the reinitialization procedure. At that time, we define the volume of fluid phase $\Omega_2^h(t^{n+1})$ as $V(\phi^{n+1}) := \int_{\Omega^h} H_\varepsilon(\phi^{n+1}) d\boldsymbol{x}$ and the initial volume of $\Omega_2^h(0)$ as $V(\phi^0) := \int_{\Omega^h} H_\varepsilon(\phi^0) d\boldsymbol{x}$

Let $l = 1, \dots, \infty$ denote the steps of our iterative procedure. Now, the main idea is to add a constant c_l to the zero contour value of the LS function of the previous iteration step, so that $|c_l| = \left|\phi_{l+1}^{n+1} - \phi_l^{n+1}\right|$ becomes the distance between the old and the new zero contour lines in the iteration. This c_l must be found in such a way as to preserve the volume of ϕ, i.e. $V(\phi_l^{n+1}) \to V(\phi^0)$ for $l \to \infty$. Therefore, our Picard iteration is defined as follows: For $l = 0, \dots, \infty$ or until $\frac{|V(\phi_l^{n+1}) - V(\phi^0)|}{|V(\phi^0)|} < \beta$ do

$$\begin{aligned} c_l &= \omega(V(\phi^0) - V(\phi_l^{n+1})) \\ \phi_{l+1}^{n+1} &= \phi_l^{n+1} + c_l \end{aligned} \tag{4.14}$$

Here, β is a predefined value of accuracy and ω is an arbitrary transformation parameter, which converts volumetric quantities into distance quantities.

In [Cro10] and in this thesis, ω is chosen as follows: We approximate $V(\phi_l^{n+1})$ as the volume of a ball with radius

$$r_l = \sqrt[3]{\frac{3V(\phi_l^{n+1})}{4\pi}}.$$

If we define $r_0 = \sqrt[3]{\frac{3V(\phi^0)}{4\pi}}$, then $c_l = r_l - r_0$ offers an approximation of the correction term, which minimizes the number of iteration steps in the relaxation process of the Picard iteration (4.14).

Note that we add a constant value c_l to the LS function in the whole computational domain resulting in a uniform blow-up of the zero contour. However, mass loss does not occur uniformly in space, wherefore the global volume correction is a source of numerical errors. This is further investigated in Subsection 6.2.1; confer also [Cro10].

For the discretization of the global volume correction, we need to discretize the integrals

$V(\phi^n) := \int_{\Omega^h} H_\varepsilon(\phi^n) d\boldsymbol{x}$, which we do by a simple summation rule, i.e.

$$|V(\phi^n)| = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \sum_{k=1}^{n_z} H_\varepsilon(\phi)\, dx_i\, dy_i\, dz_i, \tag{4.15}$$

where n_x, n_y and n_z are the number of grid cells in the respective space directions. This summation is an approximation of the volume of Ω_2^h, since $H_\varepsilon \neq 0$ if $\phi_h > -\varepsilon$, i.e. if $\boldsymbol{x} \in \Omega_2^h$ or in the smoothing region of order ε around Ω_2^h. Note that this discretization can be another large source of numerical errors, especially if the zero contour of the LS function undergoes large deformations; see Subsection 6.2.1.

4.4.2. Local volume correction

For the local volume correction, we follow [SF99] in improving the re-distancing algorithm of the LS function by formulating a constraint which conserves the volume of the domain and prevents the straying of the LS function from its initial position. In numerical computations, the Hamilton-Jacobi problem (4.5) is no longer able to conserve the volume of the domain bounded by the curve $\phi_{\tau=0}$ since numerical diffusion allows the implicitly defined boundary to move.

Instead, we now require that

$$\partial_t \int_{\Omega^h} H_\varepsilon(\phi^*) = 0, \tag{4.16}$$

where H_ε is the smoothed approximation of the Heaviside function defined by (4.9). This requirement is formulated as a constraint for the Hamilton-Jacobi problem, which we modify by

$$\phi^*_\tau = \operatorname{sgn}(\phi_0)(1 - |\nabla\phi^*|) + \lambda f(\phi^*). \tag{4.17}$$

Then we determine the time-dependent function λ by

$$\partial_\tau \int_{\Omega^h} H_\varepsilon(\phi^*) = \int_{\Omega^h} H'_\varepsilon(\phi^*)\phi^*_\tau = \int_{\Omega^h} H'_\varepsilon(\phi^*)\left(\operatorname{sgn}(\phi_0)(1 - |\nabla\phi^*|) + \lambda f(\phi^*)\right), \tag{4.18}$$

i.e.

$$\lambda = \frac{-\int_{\Omega^h} H'_\varepsilon(\phi^*)\operatorname{sgn}(\phi_0)(1 - |\nabla\phi^*|)}{\int_{\Omega^h} H'_\varepsilon(\phi^*) f(\phi^*)}. \tag{4.19}$$

The choice of

$$f(\phi^*) = H'_\varepsilon(\phi^*)\,|\nabla\phi^*| \tag{4.20}$$

ensures that the correction takes place at the interface only.

The discretization of the local mass correction in two dimensions is described in [SF99]. In three dimensions, the numerical integration of some function g over the domain

$$\Omega_{ijk} = \{(x,y,z) \in \Omega :\, x_{i-\frac{1}{2}} < x < x_{i+\frac{1}{2}},\, y_{j-\frac{1}{2}} < y < j_{j+\frac{1}{2}},\, z_{k-\frac{1}{2}} < z < z_{k+\frac{1}{2}}\},$$

changes to

$$\int_{\Omega_{ijk}} g_{ijk} \quad d\boldsymbol{x} \approx \frac{1}{78}\Bigg[52 g_{ijk}(\delta x_i \delta y_j \delta z_k) + \sum_{\substack{p,q,r=-1\\(p,q,r)\neq(0,0,0)}}^{1} \left(g_{i+p;j+q;k+r}(\delta x_i \delta y_j \delta z_k)\right)\Bigg]. \tag{4.21}$$

Furthermore, in the original article [SF99], a non-smooth signum-function is employed. For better conservation properties and numerical stability, we again choose a smooth variant and replace $\operatorname{sgn}(\phi_0)$ by $\mathrm{S}(\phi^*)$ as given in equation (4.6).

This volume correction is 'local' since the mass should remain unchanged in any subdomain of Ω^h, so that $\int H(\phi^*)$ is preserved in every grid cell. It is also 'local' in a negative sense, since it only prevents the straying of the LS function, but does not correct mass errors which occur due to the numerical diffusion introduced when solving the transport equation (3.16).

Note that due to this individual addition of mass, the local volume correction introduces noise in the curvature [Her05, Kec98], which might become an issue on very fine grids; see Subsection 6.3.1. Furthermore, in [SF99] the time step of the reinitialization equation is chosen larger if the local volume correction is applied, but a rigorous analysis is lacking. Therefore, in this thesis, we do not adapt the time step, which in turn renders the reinitialization with local volume correction computationally expensive.

4.5. Discretization of the contact angle models

In this section, we describe how the contact angle models are incorporated into our two-phase Navier-Stokes solver. For this, both models require the computation of the discrete contact line velocity $u_{\mathrm{cl},h}$. Additionally, for Shikhmurzaev's model, we also need the radial velocity $u_{(12),h}$. In the following, we focus on the example of drop impact, and we assume that the drop spreads symmetrically (cf. Fig. 4.2). Therefore, the discrete contact angle $\theta_{d,h}$, the contact point $x_{p,h}$, the contact line velocity $u_{\mathrm{cl},h}$, and the radial velocity $u_{(12),h}(\theta_{d,h}, 0)$ are evaluated at one grid point only, and for these we omit the subscript h for better readability.

Yokoi's approach

We discretize Yokoi's approach (3.27) for the computation of the dynamic contact angle by

$$\theta_d^{n+1} = \begin{cases} \min\left\{\theta_e + \left(\frac{\mu u_{\mathrm{cl}}^{n+1}}{\sigma k_a}\right)^{\frac{1}{3}}, \theta_{\mathrm{mda}}\right\} & \text{if } u_{\mathrm{cl}}^{n+1} \geq 0 \\ \max\left\{\theta_e + \left(\frac{\mu u_{\mathrm{cl}}^{n+1}}{\sigma k_r}\right)^{\frac{1}{3}}, \theta_{\mathrm{mdr}}\right\} & \text{if } u_{\mathrm{cl}}^{n+1} < 0, \end{cases} \tag{4.22}$$

where θ_e denotes the equilibrium contact angle, θ_{mda} the maximum dynamic advancing angle and θ_{mdr} the minimum dynamic receding angle. The material-related parameters k_a and k_r can be adjusted to fit the numerical results to the results obtained from measurements. Here, u_{cl}^{n+1} is approximated by the velocity value u_h^n in x-direction which is closest to the contact point x_p at the line $z_{\max}/2$ and still lies in the droplet's fluid phase; see

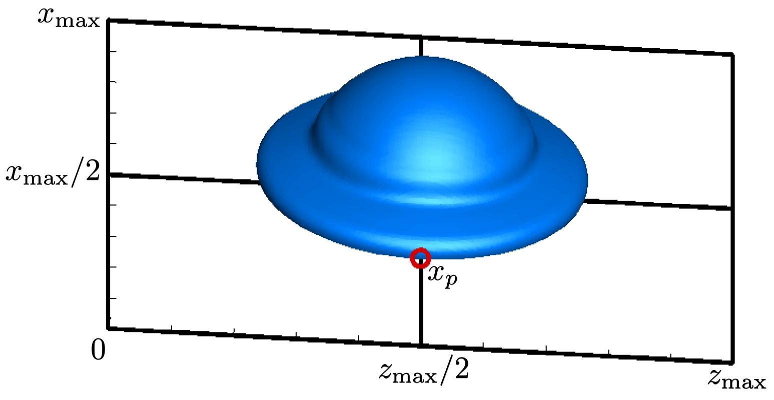


Figure 4.2.: The contact line velocity is evaluated at the contact point x_p at the intersection with the line $z = z_{\max}/2$.

Figure 4.2. This simplified computation of the contact line velocity is also done by Yokoi et al. [YVHH09] and we stick to it for the sake of comparison.

Asymptotic interface formation model

We discretize the asymptotic interface formation model (3.44) by

$$\cos(\theta_e) - \cos(\theta_d^{n+1}) = \frac{2V^{n+1}\left[\cos(\theta_e) - \tilde{\sigma}_{sg} + (1-\rho_G^s)^{-1}(1+\rho_G^s u_{(12)}^{n+1}(\theta_d, k_\mu))\right]}{V^{n+1} + [(V^{n+1})^2 + 1 + (\cos(\theta_e) - \tilde{\sigma}_{sg})(1-\rho_G^s)]^{\frac{1}{2}}} \tag{4.23}$$

with θ_e the equilibrium contact angle, k_μ the gas-to-liquid viscosity ratio, and ρ_G^s and $\tilde{\sigma}_{sg}$ phenomenological material constants; compare Subsection 3.6.2. The dimensionless contact line speed in equation (4.23) is given by

$$V^{n+1} = \frac{u_{\text{cl}}^{n+1}\mu}{\sigma} Sc.$$

with the scaling factor Sc described by equation (3.45), viscosity μ and equilibrium surface tension σ. Here, the contact line velocity u_{cl}^{n+1} is determined by the motion of the contact line position, i.e.

$$u_{\text{cl}}^{n+1} = \frac{x_p^{n+1} - x_p^n}{\delta t}, \tag{4.24}$$

and we compute the contact point x_p at the line $z = z_{\max}/2$ as follows: We check the values of ϕ_h along this line. As soon as ϕ_h changes its sign, we store the values ϕ_{in} and ϕ_{out} of the LS function inside and outside the droplet, along with the absolute position of the cell centers x_{in}^c and x_{out}^c. Then, we approximate the position of the contact point by the weighted average

$$x_p = \frac{|\phi_{\text{in}}| x^c_{\text{out}} + |\phi_{\text{out}}| x^c_{\text{in}}}{|\phi_{\text{in}}| + |\phi_{\text{out}}|}. \tag{4.25}$$

Thereby, the position of the cell center nearer to the contact line contributes more to the computation of x_p than the one further away.

The radial velocity $u^{n+1}_{(12)}$ in equation (4.23) must be derived from the solution in the outer region. In particular, in the outer region to leading order in Ca the free-surface curvature becomes zero and we obtain a flow problem in a wedge [Shi07]. The solution to this problem is given by Moffatt in [Mof64] as

$$u^{n+1}_{(12)}(\theta^{n+1}_d, 0) = \frac{\sin\theta^{n+1}_d - \theta^{n+1}_d \cos\theta^{n+1}_d}{\sin\theta^{n+1}_d \cos\theta^{n+1}_d - \theta^{n+1}_d}. \tag{4.26}$$

With Moffatt's solution, the contact angle equation (4.23) becomes nonlinear and we invoke a Newton iteration method to solve for θ^{n+1}_d. In the following, we refer to this variant of the asymptotic interface formation model by AIFM_{MOF}.

Alternatively, $u^{n+1}_{(12)}$ in (4.23) can be replaced by the inner limit of the outer solution, i.e. by a numerically computed far field velocity sufficiently close to the contact line. This far field velocity value is arbitrarily chosen to be about two grid cells away from the contact line. Thus, if e.g. at $y = 0$ at the intersection with the line $z = z_{\max}/2$, $\phi_{i,1,k} \cdot \phi_{i+1,1,k} < 0$ and $\phi_{i,1,k}$ in the liquid phase, we set $u^{n+1}_{(12)} = u^n_{i-1,2,k}$. In contrast to the AIFM_{MOF}, we can solve eq. (4.23) directly by evaluating the arccos-function. If the argument of the arccos is not in $[-1, 1]$, we use Moffatt's solution instead. In the following, we refer to this variant of the asymptotic interface formation model by AIFM_{FAR}.

Inclusion of both contact angle models

All in all, the contact line models fit into our flow solver as follows:

1. Let θ^n and the contact line position x^n_p be given from the previous time step.
2. Solve the LS advection equation (3.16) in conjunction with the contact angle boundary condition (3.23) and θ^n.

   Yokoi's approach: Use the velocity value closest x^{n+1}_p for the computation of the contact line velocity u^{n+1}_{cl} and compute θ^{n+1} from Yokoi's contact angle equation (4.22).

   Shikhmurzaev's model: Determine x^{n+1}_p from (4.25) and use equation (4.24) for u^{n+1}_{cl}.

   - AIFM_{MOF}: Use Moffatt's solution (4.26) for the radial velocity and compute θ^{n+1} by a Newton iteration of (4.23).
   - AIFM_{FAR}: Use an arbitrary far field velocity value and solve (4.23).
3. Solve the LS reinitialization (4.7) with the contact angle boundary condition (3.23) and the new $\theta = \theta^{n+1}$.

Finally, note that we use the no-slip condition for the velocity for both contact angle models. On the staggered grid, as depicted in Figure 4.1, the no-slip condition is never

exactly fulfilled, which introduces enough numerical slip to eliminate the stress singularity at the contact line.

Let us stress the potential problems of our implementation of the contact angle models: The use of the no-slip condition, as well as our choice of the radial velocity value near the contact line for the AIFM$_{\text{FAR}}$, make our numerical solution undesirably dependent on the choice of the underlying grid: For a very fine grid, we have less slip and $u_{(12)}$ is evaluated closer to the contact line than on a coarser grid. Note that the no-slip condition is also used by Yokoi et al. in [YVHH09], and so far we applied it for the sake of comparison. An alternative would be the use of a slip or Navier-slip condition (2.1), which are both already implemented in our flow solver. Concerning the radial velocity, this should have a fixed distance to the contact line. However, the choice remains problematic, since this value should be both sufficiently near the contact line *and* a part of the far field solution. Therefore, the position of $u_{(12)}$ will at least remain a problem-dependent parameter and cannot be finally determined.

An additional constraint is our assumption of symmetric drop spreading. This symmetry underlies many droplet impaction processes, but for a wider range of applications, the dynamic contact angle will have to be computed at each point along our contact line – not at a single reference point only.

4.6. Parallelization

NaSt3DGPF [NaS] already works completely in parallel and employs a natural parallelization strategy for the Navier-Stokes equations: the Cartesian grid is decomposed into q overlapping subdomains Ω^q, which are in turn treated by one of the q processors. Therefore, individual processes no longer require access to the entire data structure and the solution of iterative algorithms can be divided among them. In addition, memory requirements for each processor reduce to the memory requirements of the processes running on it.

Furthermore, we can assure the convergence of the parallelized algorithm by an exchange of relevant data between processes treating adjacent subdomains (neighboring processes). Each of these subdomains is extended by an artificial boundary, which guarantees the well-definedness of the equations on every process [Cro02, GDN98]. Since we employ finite difference stencils of variable width, which can range from three to seven grid cells, we need to attach up to three slices of boundary ghost cells to each neighboring subdomain. These boundary values are not directly available, since they are computed by processes assigned to the neighboring subdomains. Therefore, they must be sent by the neighboring processes in a communication phase inside the time-stepping loop. The velocity and pressure values in these boundary strips are updated by communication with neighbors in every time step. Additionally, the pressure has to be communicated for each iteration of the Poisson solver. Unfortunately, the increased communication of a parallelized algorithm reduces the increase in speed provided by the simultaneous execution of the code.

Optimal speed-up can only be achieved if the computing load is distributed as evenly as possible among the processors. This can be done by dividing the domain into subdomains of nearly equal size, so that each processor treats approximately the same number of unknowns. Thus, to keep the communication costs as low as possible, the discrete domain Ω^h

is decomposed by minimization of the cost functional

$$C(q^x, q^y, q^z) = \frac{N}{q^x}\frac{M}{q^y} + \frac{N}{q^x}\frac{P}{q^z} + \frac{M}{q^y}\frac{P}{q^z}$$

restricted by the side condition

$$p = q^x \cdot q^y \cdot q^z,$$

for $q^x, q^y, q^z \in \mathbb{N}$ [CGS04]. Here, N, M, P denote the number of grid cells in x-, y- and z-direction and q^x, q^y, q^z the number of processes in these directions. If q^x, q^y, q^z are factors of N, M, P, a domain decomposition into uniform cuboids results.

For the communication between processes, the computer communications protocol Message Passing Interface (MPI) is employed [MPI14]. MPI was designed for high performance on both massively parallel machines and on workstation clusters and is a standard for communication among nodes in a parallel program. The implementation consists of a library of routines that can be called from C, C++ and other programs. MPI is both portable and fast, since it has been implemented on almost every distributed memory architecture and is optimized for each hardware on which it runs.

The details of our parallelization approach have already been extensively described in [Cro02, Kli06]. In particular, the Institute for Numerical Simulation, Bonn University, aims at the development of a massively parallel, completely multi-GPU based high performance two-phase fluid solver of our NaSt3DGPF software. Preliminary results show a speedup by an order of magnitude for the whole fluid solver on one GPU and a good scaling for multi-GPU [GZ10]. However, in this thesis, our focus is on standard, not on GPU parallelization.

Therefore, all our additions to the flow solver need to work in parallel, i.e. the local volume correction, the contact angle boundary condition and the implemented contact angle models. The parallelization of the local volume correction is unproblematic, since the evaluation of the integral (4.21) can be done locally by each process. Beforehand, the values of g in the boundary strips need to be updated by communication. This communication step concerns the LS function as well as the smooth sign-function S.

Furthermore, implementing the contact angle boundary condition in parallel is straightforward, since it is an extension of the already parallelized Neumann boundary condition only, which can be handled by each process individually.

For the parallelization of the contact angle model, the most problematic part is the localization and communication of the contact point or contact line velocity for which we employ the following procedure, which we exemplify at the boundary $y = 0$:

1. Each process: At the intersection with the line $z = z_{\max}/2$, find the point where $\phi_{i,1,k} \cdot \phi_{i+1,1,k} < 0 \, \forall \, i \in \Omega^q$. Save the integer $i_s = i$.
2. With MPI, find the maximum i_s of all processes. Then, the corresponding grid point denotes the outer surface of the droplet.
3. The process containing i_s: Compute the contact line velocity u_{cl} for Yokoi's approach, the far field velocity $u_{(12)}$ and the contact point x_p. Use Yokoi's approach (3.27) or Shikhmurzaev's model (3.44) for the computation of θ_d.
4. With MPI, communicate θ_d to the remaining processes.

Thus, the computation of θ_d (e.g. with a Newton iteration) can be handled by a single process, which is optimally the one containing the reference contact point needed for the computation of all input parameters for the contact angle approaches.

With the description of parallelization, we conclude the chapter on our numerical model of the two-phase Navier-Stokes equations and the specific extensions implemented in this thesis. Our very last extension of NaSt3DGPF, the CLSVOF method, is described in the next chapter.

5. The CLSVOF Method

This chapter presents the coupled level-set and volume-of-fluid method (CLSVOF) that was implemented in this thesis. In the previous chapter, we discretized the Navier-Stokes equations (3.20) in space and time with special emphasis on the LS method, discussed the implementation of the contact angle boundary condition and presented two methods for a better conservation of mass within the LS method. Here, we mainly focus on the addition of the VOF function and its coupling with the already discussed LS method. The CLSVOF method is able to conserve mass and will turn out to be faster and much more stable than the volume correction methods described in Section 4.4.

The main motivation of the CLSVOF method is to exploit the advantages of both the LS and of the VOF method. Thus, the mass-conservative VOF function is used to correct the mass enclosed by the zero level-set of ϕ. Furthermore, the LS function is used to truncate the VOF function to avoid flotsam and jetsam. Additionally, we do not have to reconstruct the curvature or the surface normal from the discontinuous volume fractions. Instead, we use the smooth LS function for their computation.

Although the idea of coupling LS with VOF methods is not new and there are many articles devoted to this topic, e.g. [Mén07, MTB07, SH02, Son03, SP00, Sus03, SSH$^+$07, WYKS09], the vast majority of them skips the details of the involved complex computational algorithms or focus on single parts of them, which makes a fast and straightforward implementation of the CLSVOF method impossible. One of the most helpful sources for the implementation of the CLSVOF method in 3D is the article by Son [Son03]. However, Son's reinitialization strategy is quite complicated since it requires the back and forth rotation of interface cells and depends on the different possibilities of how the interface cuts the 3D numerical grid cell.

Here, we are devoted to a comprehensive description of the CLSVOF method. We especially focus on those parts of the method, which are neglected in the articles above. Thus, we explain how the reduction of the number of interface configurations works and present a new algorithm for the reinitialization of the LS function. Additionally, we also discuss the parallelization of the CLSVOF method.

5.1. Conceptual framework

Let us begin with the basic idea of the CLSVOF method: We transport two scalar indicator functions over time with the flow field, i.e. the continuous LS function ϕ and the discontinuous VOF function F, so that

$$\frac{\partial\{F,\phi\}}{\partial t} + \boldsymbol{u}\cdot\nabla\{F,\phi\} = 0.$$

This equation is equivalent to

$$\frac{\partial\{F,\phi\}}{\partial t} + \nabla \cdot \boldsymbol{u}\{F,\phi\} = \{F,\phi\}(\nabla \cdot \boldsymbol{u}), \tag{5.1}$$

and, since $\nabla \cdot \boldsymbol{u} = 0$, we have

$$\frac{\partial\{F,\phi\}}{\partial t} + \nabla \cdot \boldsymbol{u}\{F,\phi\} = 0.$$

For the discretization of this equation, we have to compute both the LS fluxes and the VOF fluxes across computational cells. Since the LS function is continuous, the respective fluxes can be computed by interpolation from neighboring cells. This, however, is not possible for the discontinuous VOF function. Thus, the computation of the VOF fluxes requires a geometric reconstruction of the interface.

For the geometric reconstruction of the interface, we opt for the PLIC approach where the interface is approximated by straight planes perpendicular to the surface normal vector of the interface in each cell. Therefore, we compute a piecewise linear reconstructed LS function ϕ^R in such a way that it is as close as possible to the real LS function ϕ. Specifically, we determine the normal $\boldsymbol{n}^R$, which is perpendicular to ϕ^R, as well as its distance d to the origin $(0, 0, 0)$ of our computational domain in each grid cell. With the help of the VOF function, the position of the plane is newly computed, so that the area beneath the plane equals the given volume fraction. Due to the reconstructed interface, we can then compute the VOF fluxes geometrically. In a last step, we reinitialize the LS function by the exact signed normal distance of each cell center to the reconstructed interface.

In what follows, we describe the necessary additions to our flow solver NaSt3DGPF which employs a LS method already but lacks the VOF function. In Section 5.2, we discuss the space-time discretization of the transport equation (5.1) for which we use an operator splitting (or fractional step) method. This discussion will already indicate the two main requirements for our CLSVOF method, namely the need for a geometric reconstruction of the interface (Sec. 5.3) to compute the VOF fluxes (Sec. 5.5), as well as the need for a correction of said interface to become mass conservative, which we discuss in Section 5.4. Furthermore, Section 5.6 deals with the reinitialization of the LS function, and in Section 5.7, we describe the discretization of contact angle boundary conditions, which is different than for the LS method. We conclude this chapter with our parallelization strategy for the CLSVOF method in Section 5.8.

5.2. Discretization of the CLSVOF method

In this section we discuss the discretization of the CLSVOF method. Basically, the discretization of the Navier-Stokes equations and parts of the discretization of the LS function remain untouched. Thus, only the space-time discretization of the transport equation (5.1) has to be discussed as well as the new reinitialization scheme. However, as mentioned above, the devil is always in the details, which seems to be especially true for the hybrid CLSVOF method.

Note that the implementation of the CLSVOF method has to be done with special care concerning floating point comparisons, which arise in the computation of the volume fractions and within the reinitialization equation. Additionally, all divisions have to be carefully controlled since the denominators can become tiny. Although these considerations hold for every computer program, the sheer number of such occurrences within the implementation of the CLSVOF method necessitate this remark.

5.2.1. Discretization in time: Operator splitting

For the advection of the LS function and the liquid volumes in the x, y and z-directions we use an operator splitting algorithm, i.e. we solve equation (5.1) for one direction at a time. This splitting can be done in different ways and we opt for the algorithm by Son [Son03]. With $s = F$ or $s = \phi$, the transport equation reads

$$\frac{\partial s}{\partial t} + \nabla \cdot (\boldsymbol{u} s) = s(\nabla \cdot \boldsymbol{u}),$$

which is discretized as

$$\frac{s^* - s^n}{\delta t} + \frac{\partial u^n s^n}{\partial x} = s^* \frac{\partial u^n}{\partial x} \tag{5.2}$$

$$\frac{s^{**} - s^*}{\delta t} + \frac{\partial v^n s^*}{\partial y} = s^* \frac{\partial v^n}{\partial y} \tag{5.3}$$

$$\frac{s^{n+1} - s^{**}}{\delta t} + \frac{\partial w^n s^{**}}{\partial z} = s^* \frac{\partial w^n}{\partial z}. \tag{5.4}$$

Here, s^* and s^{**} are functions of intermediate time in between time steps n and $n+1$. Equations (5.2)–(5.4) are written in conservation form, so that F^{n+1} is conserved under the incompressibility condition $\nabla \cdot \boldsymbol{u}^n = 0$. Additionally, this discretization has the potential to leave F invariant in the single phase region in each of the fractional steps, which is the main reason why Son's operator splitting was chosen in this thesis: If $F^n = 1$ and $\frac{\partial u}{\partial x} < 0$, we also obtain $F^{n+1} = 1$ in equation (5.4). Note that in the works of Sussman et al. [SP00, Sus03, SSH+07], one can instead find

$$\begin{aligned}
\frac{s^* - s^n}{\delta t} + \frac{\partial u^n s^n}{\partial x} &= s^* \frac{\partial u^n}{\partial x} \\
\frac{s^{**} - s^*}{\delta t} + \frac{\partial v^n s^*}{\partial y} &= s^{**} \frac{\partial v^n}{\partial y} \\
\frac{s^{***} - s^{**}}{\delta t} + \frac{\partial w^n s^{**}}{\partial z} &= s^{***} \frac{\partial w^n}{\partial z} \\
\frac{s^{n+1} - s^{***}}{\delta t} &= -s^* \frac{\partial u^n}{\partial x} - s^{**} \frac{\partial v^n}{\partial y} - s^{***} \frac{\partial w^n}{\partial z},
\end{aligned}$$

which is also in conservation form and can leave F invariant in the single-phase region.

The operator splitting is of second order in time if we alternate between the sweep directions in every time step.

5.2.2. Discretization in space

Again, we denote by cells the rectangular subdomains

$$[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}] \times [y_{j-\frac{1}{2}}, y_{j+\frac{1}{2}}] \times [z_{k-\frac{1}{2}}, z_{k+\frac{1}{2}}],$$

and we define the discrete computational domain Ω_h as a union of such cells. For $\{i,j,k\} \in \mathbb{Z}$ we use the notation

$$\delta x_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}},$$

and δy_j as well as δz_k are defined analogously. Furthermore, 'ghost cells' or 'boundary cells' denote the up to three additional strips of cells attached to Ω_h, which are needed for the discretization of large finite difference stencils and boundary conditions.

Both the LS and the VOF function are discretized by finite differences in the cell centers of our staggered grid, i.e. at the discrete coordinates (x_i, y_j, z_k) with $\{i,j,k\} \in \mathbb{Z}$. Integration of equations (5.2)–(5.4) over the computational cell (i,j,k) yields

$$s^*_{i,j,k} = \frac{s^n_{i,j,k} - \frac{\delta t}{\delta x}\left(G_{i+\frac{1}{2},j,k} - G_{i-\frac{1}{2},j,k}\right)}{1 - \frac{\delta t}{\delta x}\left(u_{i+\frac{1}{2},j,k} - u_{i-\frac{1}{2},j,k}\right)} \tag{5.5}$$

$$s^{**}_{i,j,k} = s^*_{i,j,k}\left(1 + \frac{\delta t}{\delta y}\left(v_{i,j+\frac{1}{2},k} - v_{i,j-\frac{1}{2},k}\right)\right) - \frac{\delta t}{\delta y}\left(G^*_{i,j+\frac{1}{2},k} - G^*_{i,j-\frac{1}{2},k}\right) \tag{5.6}$$

$$s^{n+1}_{i,j,k} = s^{**}_{i,j,k} + s^*_{i,j,k}\frac{\delta t}{\delta z}\left(w_{i,j,k+\frac{1}{2}} - w_{i,j,k-\frac{1}{2}}\right) - \frac{\delta t}{\delta z}\left(G^{**}_{i,j,k+\frac{1}{2}} - G^{**}_{i,j,k-\frac{1}{2}}\right), \tag{5.7}$$

where $G_{i+\frac{1}{2},j,k} = s_{i+\frac{1}{2},j,k} u_{i+\frac{1}{2},j,k}$. This is the flux of s across the face $(i+\frac{1}{2}, j, k)$ of the (i,j,k)-th computational cell. Similarly, we write $G^*_{i,j+\frac{1}{2},k} = s^*_{i,j+\frac{1}{2},k} v_{i,j+\frac{1}{2},k}$ and $G^{**}_{i,j,k+\frac{1}{2}} = s^{**}_{i,j,k+\frac{1}{2}} w_{i,j,k+\frac{1}{2}}$. Similarly, $G_{i-\frac{1}{2},j,k}$, $G^*_{i,j-\frac{1}{2},k}$ and $G^{**}_{i,j,k-\frac{1}{2}}$ are defined as the fluxes across the face $(i-\frac{1}{2}, j, k)$, $(i, j-\frac{1}{2}, k)$ and $(i, j, k-\frac{1}{2})$ of the (i,j,k)-th computational cell.

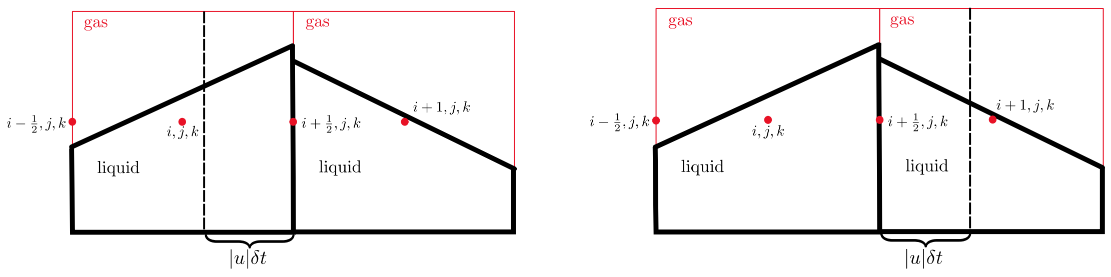
For the solution of these equations we have to determine $s_{i\pm\frac{1}{2},j,k}$, $s_{i,j\pm\frac{1}{2},k}$ and $s_{i,j,k\pm\frac{1}{2}}$. Therefore, we have to distinguish between the case that s denotes the LS fluxes or the VOF fluxes. In the following, we focus on $s_{i+\frac{1}{2},j,k}$, $s_{i,j+\frac{1}{2},k}$ and $s_{i,j,k+\frac{1}{2}}$ only since the computation of the fluxes across the downward cell faces works in an equivalent fashion.

Computation of the LS fluxes $s = \phi$

Since the LS function is smooth, $s_{i+\frac{1}{2},j,k}$, $s_{i,j+\frac{1}{2},k}$ and $s_{i,j,k+\frac{1}{2}}$ are obtained by extrapolation of s in space and time [Mén07, SP00]:

$$s^n_{i+\frac{1}{2},j,k} = s^n_{i,j,k} + \frac{\delta x}{2}\begin{cases}\left(1 - u_{i+\frac{1}{2},j,k}\frac{\delta t}{\delta x}\right)\frac{s^n_{i+1,j,k} - s^n_{i-1,j,k}}{2\delta x} & \text{if } u_{i+\frac{1}{2},j,k} > 0\\ \left(1 + u_{i+\frac{1}{2},j,k}\frac{\delta t}{\delta x}\right)\frac{s^n_{i+2,j,k} - s^n_{i,j,k}}{2\delta x} & \text{if } u_{i+\frac{1}{2},j,k} < 0\end{cases}$$

$$s^*_{i,j+\frac{1}{2},k} = s^*_{i,j,k} + \frac{\delta y}{2}\begin{cases}\left(1 - v_{i,j+\frac{1}{2},k}\frac{\delta t}{\delta y}\right)\frac{s^*_{i,j+1,k} - s^*_{i,j-1,k}}{2\delta y} & \text{if } v_{i,j+\frac{1}{2},k} > 0\\ \left(1 + v_{i,j+\frac{1}{2},k}\frac{\delta t}{\delta y}\right)\frac{s^*_{i,j+2,k} - s^*_{i,j,k}}{2\delta y} & \text{if } v_{i,j+\frac{1}{2},k} < 0\end{cases}$$



(a) $u_{i+\frac{1}{2},j,k} > 0$ **(b)** $u_{i+\frac{1}{2},j,k} < 0$

Figure 5.1.: If $u_{i+\frac{1}{2},j,k} > 0$, there is flux from cell (i,j,k) to $(i+1,j,k)$. If $u_{i+\frac{1}{2},j,k} < 0$, there is flux from cell $(i+1,j,k)$ to (i,j,k).

$$s^{**}_{i,j,k+\frac{1}{2}} = s^{**}_{i,j,k} + \frac{\delta z}{2}\begin{cases}\left(1 - w_{i,j,k+\frac{1}{2}}\frac{\delta t}{\delta z}\right)\frac{s^{**}_{i,j,k+1}-s^{**}_{i,j,k-1}}{2\delta z} & \text{if } w_{i,j,k+\frac{1}{2}} > 0\\ \left(1 + w_{i,j,k+\frac{1}{2}}\frac{\delta t}{\delta z}\right)\frac{s^{**}_{i,j,k+2}-s^{**}_{i,j,k}}{2\delta z} & \text{if } w_{i,j,k+\frac{1}{2}} < 0.\end{cases}$$

Note here, however, that our solution $s^{n+1} = \phi^{n+1}$ is not mass-conservative. Therefore, one of the main tasks of our CLSVOF method is a correction of the interface through the volume fractions for a better conservation of mass.

Computation of the VOF fluxes $s = F$

Since F is not distributed smoothly, arithmetic interpolations from neighbor cells (as done for the LS function) are not feasible. Instead, the volume fluxes $s_{i+\frac{1}{2},j,k}$, $s_{i,j+\frac{1}{2},k}$ and $s_{i,j,k+\frac{1}{2}}$ are computed geometrically as the liquid volume fraction that is advected across a given cell face during a certain time step. This geometric computation is exemplified in Figure 5.1(a). Here, since $u_{i+\frac{1}{2},j,k} > 0$, volume from cell (i,j,k) (the donor cell) is transported to cell $(i+1,j,k)$. Then, we define $s^n_{i+\frac{1}{2},j,k}$ as the liquid volume fraction

$$s^n_{i+\frac{1}{2},j,k} = \frac{(\delta V_F)^{\rightarrow}_{i+\frac{1}{2},j,k}}{|u|_{i+\frac{1}{2},j,k}\delta t \delta y_j \delta z_k},$$

i.e. as the liquid volume $(\delta V_F)^{\rightarrow}_{i+\frac{1}{2},j,k}$ in relation to the total volume $|u|_{i+\frac{1}{2},j,k}\delta t\delta y_j\delta z_k$, which is advected in the x-direction during the time step δt. Furthermore, the liquid volume is defined as

$$(\delta V_F)^{\rightarrow}_{i+\frac{1}{2},j,k} := \int_{x_{i+\frac{1}{2}}-|u|_{i+\frac{1}{2},j,k}\delta t}^{x_{i+\frac{1}{2}}} \int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} \int_{z_{k-\frac{1}{2}}}^{z_{k+\frac{1}{2}}} H\left(\phi^{R,n}_{i,j,k}(x,y,z)\right)\, dx\, dy\, dz,$$

where $\phi^{R,n}_{i,j,k}$ denotes the piecewise linear reconstruction of the interface at time $t^n = n \cdot \delta t$. This, however, requires the reconstruction of the interface $\phi^{R,n}_{i,j,k}$ in the first place, which is explained in Section 5.3.

In the equation above and in the following, the superscript arrows $\leftrightarrows$ of (δV_F) indicate the donor cell. If the velocity is positive, there is flux from the donor cell (i, j, k) to $(i+1, j, k)$, which we depict by the arrow from left to right, because of the flux direction $(i, j, k) \to (i+1, j, k)$. On the other hand, if the velocity is negative we have flux from the donor cell $(i+1, j, k)$ to (i, j, k), which we indicate by the arrow from right to left, because of the flux direction $(i, j, k) \leftarrow (i+1, j, k)$. Both situations are depicted in Figure 5.1.

For the sake of shortness we write $u = u_{i+\frac{1}{2},j,k}$, $v = v_{i,j+\frac{1}{2},k}$, $w = w_{i,j,k+\frac{1}{2}}$, $\delta x = \delta x_i$, $\delta y = \delta y_j$, $\delta_z = \delta z_k$ and compute the volume fractions as

$$\begin{aligned}
s^n_{i+\frac{1}{2},j,k} &= \frac{1}{|u|\delta t \delta y \delta z} \begin{cases} (\delta V_F)^{\rightarrow}_{i+\frac{1}{2},j,k} & \text{if } u > 0 \\ (\delta V_F)^{\leftarrow}_{i+\frac{1}{2},j,k} & \text{if } u < 0 \end{cases} \\
s^*_{i,j+\frac{1}{2},k} &= \frac{1}{|v|\, \delta t \delta x \delta z} \begin{cases} (\delta V_F)^{\rightarrow}_{i,j+\frac{1}{2},k} & \text{if } v > 0 \\ (\delta V_F)^{\leftarrow}_{i,j+\frac{1}{2},k} & \text{if } v < 0 \end{cases} \\
s^{**}_{i,j,k+\frac{1}{2}} &= \frac{1}{|w|\, \delta t \delta x \delta y} \begin{cases} (\delta V_F)^{\rightarrow}_{i,j,k+\frac{1}{2}} & \text{if } w > 0 \\ (\delta V_F)^{\leftarrow}_{i,j,k+\frac{1}{2}} & \text{if } w < 0 \end{cases} .
\end{aligned} \tag{5.8}$$

The computation of the liquid volumes such as $(\delta V_F)^{\leftrightarrows}_{i+\frac{1}{2},j,k}$ with the help of the reconstructed interface is explained in Section 5.5.

5.2.3. Summary and further tasks

We have seen that the basic idea of the CLSVOF method is the construction of a new interface ϕ^R, which will be done with the help of both the VOF function F and the LS function ϕ. This interface construction is done in each operator splitting step and is then employed for the evaluation of the advected liquid volumes on the one hand and, finally, for the reinitialization of the LS function on the other hand.

For the reconstruction and advection of the interface we use the following procedure by Son [Son03]:

1. Use F^n and ϕ^n to reconstruct the interface in (i, j, k) for the computation of the volume flux $(\delta V_F)^n_{i\pm\frac{1}{2},j,k}$. Solve equation (5.2) with $s^n = F^n$ and $s^n = \phi^n$ for F^* and ϕ^*.
2. Use F^* and ϕ^* to reconstruct the interface in (i, j, k) for the computation of the volume flux $(\delta V_F)^*_{i,j\pm\frac{1}{2},k}$. Solve equation (5.3) with $s^* = F^*$ and $s^* = \phi^*$ for F^{**} and ϕ^{**}.
3. Use F^{**} and ϕ^{**} to reconstruct the interface in (i, j, k) for the computation of the volume flux $(\delta V_F)^{**}_{i,j,k\pm\frac{1}{2}}$. Solve equation (5.4) with $s^{**} = F^{**}$ and $s^{**} = \phi^{**}$ for F^{n+1} and ϕ^{n+1}.
4. Truncate the volume fractions by equation (5.28).
5. Reconstruct the interface from F^{n+1} and ϕ^{n+1} to reinitialize ϕ^{n+1}, i.e. the LS function is set to be the exact signed normal distance to the reconstructed interface.

All in all, this procedure requires three basic steps:

1. Reconstruction of an interface from a given VOF and LS function; see Section 5.3.
2. Computation of the advected liquid volumes $F_{i\pm\frac{1}{2},j,k}$, $F_{i,j\pm\frac{1}{2},k}$ and $F_{i,j,k\pm\frac{1}{2}}$ with the reconstructed interface; see Section 5.5.
3. Reinitialization of the LS function with the help of the reconstructed interface; see Section 5.6. In this section, we also describe the truncation of the volume fractions.

5.3. Reconstruction of the interface

The main parts of our interface reconstruction algorithm are as follows:

- Check in which computational cells the interface needs to be reconstructed.
- In each of these computational cells, construct a linear function ϕ^R_{ijk} which is as close as possible to the real zero LS function ϕ_{ijk}.
- Shift the interface so that the area beneath the plane equals the volume fraction F_{ijk} in that cell.

Since the reconstruction of the interface is of linear type, we first introduce some basic definitions and concepts about planes in three dimensions.

5.3.1. Plane, distance and projection in three dimensions

Let $(x, y, z) \in \mathbb{R}^3$ and $a, b, c, d \in \mathbb{R}$. A plane is defined by

$$ax + by + cz + d = 0 \tag{5.9}$$

with its normal (a, b, c) and the plane's distance d to the origin. We can *normalize* this equation by dividing with $\sqrt{1/(a^2 + b^2 + c^2)}$. Then,

$$\boldsymbol{n} = \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} = \begin{pmatrix} a/\sqrt{(a^2 + b^2 + c^2)} \\ b/\sqrt{(a^2 + b^2 + c^2)} \\ c/\sqrt{(a^2 + b^2 + c^2)} \end{pmatrix}$$

is the unit normal of the plane.

The *signed distance* of a point $\boldsymbol{p} = (p_x, p_y, p_z) \in \mathbb{R}^3$ to the plane is

$$D(\boldsymbol{p}) = \boldsymbol{n} \cdot \boldsymbol{p} + d = n_x p_x + n_y p_y + n_z p_z + d. \tag{5.10}$$

This distance is positive, if $\boldsymbol{p}$ lies on the same side of the plane towards which the unit normal points and negative otherwise.

The projection of a point $\boldsymbol{p} \in \mathbb{R}^3$ onto the plane is defined by

$$P(\boldsymbol{p}) = \boldsymbol{p} - D(\boldsymbol{p}) \cdot \boldsymbol{n}.$$

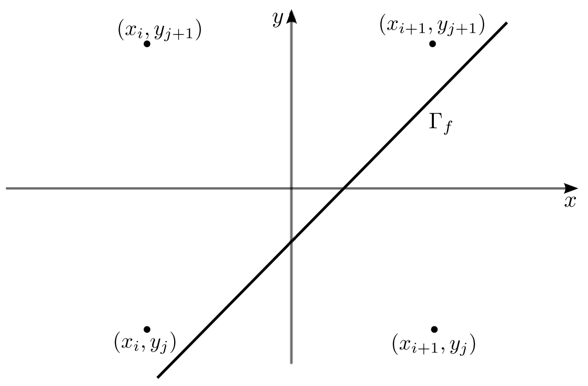


Figure 5.2.: Example for finding the cells in which to reconstruct the interface Γ_f.

5.3.2. Where to reconstruct the interface

In order to check in which cells we want to reconstruct the interface, the procedure described in [SP00, Sus03] is as follows:

- For all computational cells (i, j, k): Check if $0 < F_{i,j,k} < 1$.
- If additionally, $\phi_{i,j,k}(\phi_{i,j,k} + \phi_{i',j',k'}) \leq 0$ for some $|i - i'| \leq 1$, $|j - j'| \leq 1$ and $|k - k'| \leq 1$ reconstruct the linear interface $\phi^R_{i,j,k}$ in cell (i, j, k).

These conditions can be understood as follows:

$$\begin{aligned} & \phi_{i,j,k}(\phi_{i,j,k} + \phi_{i',j',k'}) \leq 0 \\ \Leftrightarrow \qquad & \phi^2_{i,j,k} \leq -\phi_{i,j,k}\phi_{i',j',k'} \end{aligned}$$

This inequality can only be fulfilled if its right hand side is positive. Therefore, the LS function in cell (i', j', k') must have opposite sign than in cell (i, j, k), and the interface has to lie in between these two cells. Specifically, we know that it has to be located in cell (i, j, k): If $\phi_{i,j,k} > 0$, then $\phi_{i,j,k} \leq -\phi_{i',j',k'}$, which is equivalent to $|\phi_{i,j,k}| \leq |\phi_{i',j',k'}|$ and means that cell i, j, k is nearer to the interface than cell (i', j', k'), and we have to reconstruct the interface there. Else, if $\phi_{i,j,k} < 0$, we have $-\phi_{i,j,k} \leq \phi_{i',j',k'}$ by division with $-\phi_{i,j,k}$. Again, this is equivalent to $|\phi_{i,j,k}| \leq |\phi_{i',j',k'}|$.

Note that the restriction '$|i-i'| \leq 1$, $|j-j'| \leq 1$ and $|k-k'| \leq 1$' makes sure that we not only look at cells which are direct neighbors of each other, but also at diagonal neighbors.

However, in this thesis we demand that additionally $(i - i')(j - j')(k - k') < 1$, i.e. we only look at cells which are direct neighbors of each other (not diagonal neighbors). Then, the above procedure is altered as follows (cf. [Mén07]):

- For all computational cells (i, j, k): Check if $0 < F_{i,j,k} < 1$.
- If additionally, $\phi_{i,j,k} \cdot \phi_{i',j',k'} < 0$ or $\phi_{i,j,k} = 0$ for some $|i - i'| \leq 1$, $|j - j'| \leq 1$ and $|k - k'| \leq 1$ with $(i - i')(j - j')(k - k') \leq 1$, reconstruct the linear interface $\phi^R_{i,j,k}$ in cell (i, j, k).

Theoretically, in combination with our restriction $0 < F_{i,j,k} < 1$, both procedures should produce identical results. Let us exemplify this in two dimensions. Figure 5.2 shows an interface which crosses three cells. Then, Sussman's procedure would reconstruct the interface in cell (i, j) since the LS function has opposite sign in cells (i, j) and $(i+1, j)$, and the cell center of cell (i, j) is nearer to the interface. With similar arguments, the interface is reconstructed in cell $(i + 1, j + 1)$. Furthermore, the interface is also reconstructed in cell $(i + 1, j)$ since Sussman's procedure allows for comparison with the diagonal neighbor $(i, j + 1)$. In our procedure, the interface would also be reconstructed in all of these three cells since we simply check if the LS function changes its sign from one cell to its direct neighbor.

5.3.3. How to reconstruct the interface

For the geometric reconstruction of the interface, we opt for the PLIC approach where the interface is approximated by straight planes perpendicular to the surface normal vector of the interface in each cell. Therefore, we compute a piecewise linear reconstructed LS function ϕ^R in such a way that it is as close as possible to the real LS function ϕ.

Our linear reconstruction of the interface in cell (i, j, k) results in a planar equation, i.e.

$$\phi^R_{ijk}(x, y, z) = a_{i,j,k}(x - x_i) + b_{i,j,k}(y - y_j) + c_{i,j,k}(z - z_k) + d_{i,j,k}. \tag{5.11}$$

Here, $a_{i,j,k}$, $b_{i,j,k}$, $c_{i,j,k}$ are the coordinates of the vector which is normal to the plane given by

$$a_{i,j,k}(x - x_i) + b_{i,j,k}(y - y_j) + c_{i,j,k}(z - z_k) + d_{i,j,k} = 0, \tag{5.12}$$

and the intercept $d_{i,j,k}$ is the distance of the interface to the origin.

Now, the coefficients $a_{i,j,k}$, $b_{i,j,k}$, $c_{i,j,k}$ and $d_{i,j,k}$ are determined so that the reconstruction ϕ^R_{ijk} is as close as possible to the real zero LS function ϕ [Mén07, MTB07, SP00]. Therefore, they minimize the error functional

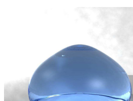
$$\begin{aligned} E_{i,j,k} = &\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} \int_{z_{k-\frac{1}{2}}}^{z_{k+\frac{1}{2}}} H'(\phi_{i,j,k}) \\ &(\phi_{i,j,k} - a_{i,j,k}(x - x_i) - b_{i,j,k}(y - y_j) - c_{i,j,k}(z - z_k) - d_{i,j,k})^2 \, dx \, dy \, dz. \end{aligned}$$

In order to determine a, b, c and d we minimize the discretized error

$$\begin{aligned} E_{i,j,k} = \sum_{i'=i-1}^{i+1} \sum_{j'=j-1}^{j+1} \sum_{k'=k-1}^{k+1} &\Big[w_{i'-i,j'-j,k'-k} H'_\varepsilon(\phi_{i',j',k'}) \\ &(\phi_{i',j',k'} - a_{i,j,k}(x_{i'} - x_i) - b_{i,j,k}(y_{j'} - y_j) - c_{i,j,k}(z_{k'} - z_k) - d_{i,j,k})^2 \Big], \end{aligned}$$

which leads to the conditions

$$\frac{\partial E_{i,j,k}}{\partial a_{i,j,k}} = \frac{\partial E_{i,j,k}}{\partial b_{i,j,k}} = \frac{\partial E_{i,j,k}}{\partial c_{i,j,k}} = 0,$$



and requires the solution of the system of equations

$$\begin{bmatrix} \sum whX^2 & \sum whXY & \sum whXZ & \sum whX \\ \sum whXY & \sum whY^2 & \sum whYZ & \sum whY \\ \sum whXZ & \sum whYZ & \sum whZ^2 & \sum whZ \\ \sum whX & \sum whY & \sum whZ & \sum wh \end{bmatrix} \begin{bmatrix} a_{i,j,k} \\ b_{i,j,k} \\ c_{i,j,k} \\ d_{i,j,k} \end{bmatrix} \begin{bmatrix} \sum wh\phi X \\ \sum wh\phi Y \\ \sum wh\phi Z \\ \sum wh\phi \end{bmatrix}. \tag{5.13}$$

Here, similar to [MTB07], we use the notation

$$\begin{aligned} \sum &\to \sum_{i'=i-1}^{i+1} \sum_{j'=j-1}^{j+1} \sum_{k'=k-1}^{k+1} \\ wh &\to w_{i'-i,j'-j,k'-k} H'_\varepsilon(\phi_{i',j',k'}) \\ X &\to (x_{i'} - x_i) \\ Y &\to (y_{j'} - y_j) \\ Z &\to (z_{k'} - z_k) \\ \phi &\to \phi_{i',j',k'}. \end{aligned}$$

We solve these equations by a simple Cholesky decomposition [PTVF07].

If the equation system is not solvable by a Cholesky decomposition, the matrix on the left hand side of (5.13) is not positive definite, which can be the result of roundoff errors. In this case we compute $a_{i,j,k}$, $b_{i,j,k}$, $c_{i,j,k}$ by central differences

$$\begin{aligned} a_{i,j,k} &= \frac{\phi_{i+1,j,k} - \phi_{i-1,j,k}}{\frac{1}{2}(\delta x_{i-1} + \delta x_{i+1}) + \delta x_i} \\ b_{i,j,k} &= \frac{\phi_{i,j+1,k} - \phi_{i,j-1,k}}{\frac{1}{2}(\delta y_{j-1} + \delta y_{j+1}) + \delta y_j} \\ c_{i,j,k} &= \frac{\phi_{i,j,k+1} - \phi_{i,j,k-1}}{\frac{1}{2}(\delta z_{k-1} + \delta z_{k+1}) + \delta z_k} \end{aligned}$$

and $d_{i,j,k} = \phi_{i,j,k}$. At the boundary, one-sided differences are employed. All in all, this simpler computation of the normal proved to be less accurate in numerical tests than the solution of (5.13) [SP00].

After we have determined $a_{i,j,k}$, $b_{i,j,k}$, $c_{i,j,k}$ and $d_{i,j,k}$, we normalize the interface equation (5.12) by multiplication with $1/\sqrt{a_{i,j,k}^2 + b_{i,j,k}^2 + c_{i,j,k}^2}$. Therefore in the following, we write n_x, n_y, n_z, d instead of $a_{i,j,k}$, $b_{i,j,k}$, $c_{i,j,k}$, $d_{i,j,k}$, if the components are already normalized and if there can be no confusion concerning the indices.

For the above reconstruction we have used the LS function only. In the next step, we use the VOF function for the correction of the intercept d: we correct d in such a way that the plane represented by (5.12) cuts out the same volume in the cell (i, j, k) as specified by $F_{i,j,k}$.

5.4. Shifting the interface for mass conservation

Since we want to shift the interface in such a way that the area beneath the plane equals the given volume fraction, we have to formulate an algorithm which determines d from given $F = F_{i,j,k}$ and $\boldsymbol{n} = (n_x, n_y, n_z)^\mathsf{T}$ in each grid cell. Here, the main difficulty lies in the vast number of interface configurations which are possible both in two and in three dimensions. Therefore, many authors [MTB07, SH02, Son03, SP00] first reduce the number of considered interface configurations as follows [SH02, p. 530]:

> The interface location is determined effectively by introducing s which denotes the distance from the interface to the corner of an interface cell [20]. The corner is chosen to be inside the liquid region and the farthest from the interface.

Here, the source given by '[20]' is 'M. Sussman, personal communication, 2001'. Fortunately, some more details can be found in Ménard's thesis [Mén07] and his subsequent article [MTB07].

Note that we follow the notation in the standard literature, so that from hereon, s no longer stands for the VOF or LS function, but for the distance from the interface to the corner of an interface cell.

Since the reduction of the number of interface configurations is a crucial part of the CLSVOF algorithm, our first subsection is merely devoted to a rigorous explanation of the above quotation.

5.4.1. Reduction of the number of interface configurations

In order to reduce the number of possible interface configurations, we make the following two transformations, which are illustrated in Figure 5.3 for the 2D case:

1. We reflect the plane in such a way, that all components of its unit normal $\boldsymbol{n} = (n_x, n_y, n_z)^\mathsf{T}$ become negative.
2. We shift the plane's origin in our computational cell from the cell center to the lower left hand corner of the cell. This corner will lie in the liquid region and will be farthest away from the interface.

Our original plane as written in equation (5.12) is given by

$$n_x(x - x_i) + n_y(y - y_j) + n_z(z - z_k) + d = 0. \tag{5.14}$$

Here, we introduce a new coordinate system (x', y', z') with its origin in the cell center (x_i, y_j, z_k) by the following translation:

$$\begin{aligned} x' &= x - x_i \\ y' &= y - y_j \\ z' &= z - z_k. \end{aligned}$$

Then our plane in the new coordinate system simply has the form

$$n_x x' + n_y y' + n_z z' + d = 0 \tag{5.15}$$

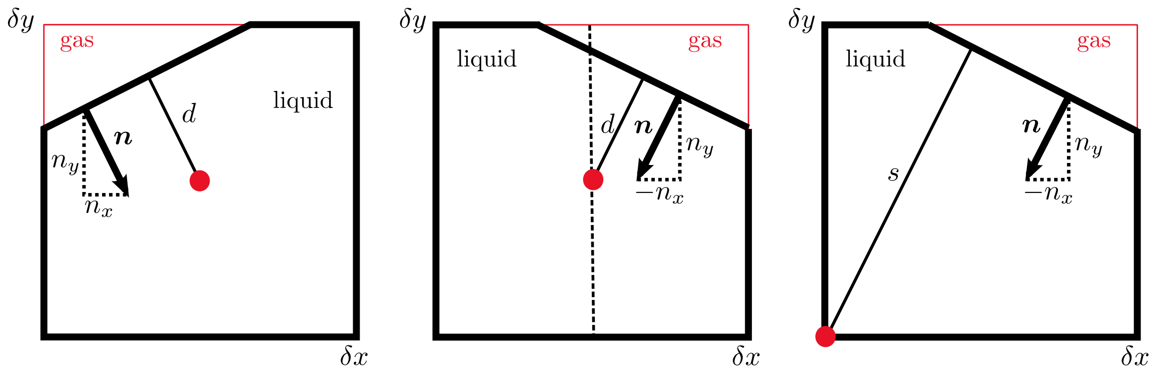


Figure 5.3.: Computing the reflected line and choosing the origin of the computational cell in 2D. The gas volume is surrounded by red solid lines and the liquid volume by thick black lines. In the first step, we reflect the line (i.e. the interface between gas and liquid) in such a way that all components of the normal vector become negative. In the above case, only $n_x > 0$ and we compute the reflection across the dashed vertical line. In the second step, we shift the origin from the cell center to the lower left hand cell edge, which then lies in the liquid region and as far from the interface as possible.

In the first step, we reflect the plane across our computational cell in such a way that all components of the unit normal become negative (see Fig. 5.3). This requires the construction of a reflection $\mathrm{Ref} = (\mathrm{Ref}^x, \mathrm{Ref}^y, \mathrm{Ref}^z)$ with the following properties:

$$\begin{aligned}
\mathrm{Ref}^x(x') &= \begin{cases} -x' & \text{if } n_x > 0 \\ x' & \text{else} \end{cases} \\
\mathrm{Ref}^y(y') &= \begin{cases} -y' & \text{if } n_y > 0 \\ y' & \text{else} \end{cases} \\
\mathrm{Ref}^z(z') &= \begin{cases} -z' & \text{if } n_z > 0 \\ z' & \text{else.} \end{cases}
\end{aligned} \tag{5.16}$$

In general, a reflection across the hyperplane which lies in the origin and is orthogonal to an arbitrary vector $\boldsymbol{e}$ is given by

$$\mathrm{Ref}_{\boldsymbol{e}}(\boldsymbol{x}') = \boldsymbol{x}' - 2\frac{\boldsymbol{x}' \cdot \boldsymbol{e}}{\boldsymbol{e} \cdot \boldsymbol{e}}\boldsymbol{e}.$$

We choose the hyperplane in such a way that

$$\begin{aligned}
\boldsymbol{e}^1 &= \begin{cases} (1,0,0)^\top & \text{if } n_x > 0 \\ (0,0,0)^\top & \text{else} \end{cases} \\
\boldsymbol{e}^2 &= \begin{cases} (0,1,0)^\top & \text{if } n_y > 0 \\ (0,0,0)^\top & \text{else} \end{cases}
\end{aligned}$$

$$\boldsymbol{e}^3 = \begin{cases} (0,0,1)^\mathsf{T} & \text{if } n_z > 0 \\ (0,0,0)^\mathsf{T} & \text{else} \end{cases}$$

The successive computation of $\mathrm{Ref}_{\boldsymbol{e}^3}(\mathrm{Ref}_{\boldsymbol{e}^2}(\mathrm{Ref}_{\boldsymbol{e}^1}(\boldsymbol{x}')))$ yields the desired result (5.16). Then, the reflected plane can be written as

$$n_x \mathrm{Ref}^x(x') + n_y \mathrm{Ref}^y(y') + n_z \mathrm{Ref}^y(y') + d = 0, \tag{5.17}$$

which is equal to

$$-\left|n_x\right| x' - \left|n_y\right| y' - \left|n_z\right| z' + d = 0, \tag{5.18}$$

since e.g. for the x-component

$$n_x \mathrm{Ref}^x(x') = \left|n_x\right| x' \text{ for all } n_x.$$

In the second step, we place the origin in the lower left hand corner of the computational cell (see Fig. 5.3). Therefore, we introduce the new coordinates

$$\begin{aligned} x'' &= x' - 0.5\Delta x \\ y'' &= y' - 0.5\Delta y \\ z'' &= z' - 0.5\Delta z, \end{aligned}$$

so that our plane (5.18) is given by

$$-\left|n_x\right| x'' - \left|n_y\right| y'' - \left|n_z\right| z'' + s = 0. \tag{5.19}$$

The distance s to the origin is then computed by the distance of $(-0.5\Delta x, -0.5\Delta y, -0.5\Delta z)$ to the plane defined by (5.18), i.e.

$$\begin{aligned} s &= D_{(5.18)}(-0.5\Delta x, -0.5\Delta y, -0.5\Delta z) \\ &= \left|n_x\right| 0.5\Delta x' + \left|n_y\right| 0.5\Delta y' + \left|n_z\right| 0.5\Delta z' + d \end{aligned}$$

The so computed origin lies in the liquid region and as far from the interface as possible.

All in all, we now can always consider $|n_x|, |n_y|, |n_z|$ instead of the signed components, which simplifies the computation of the liquid volumes considerably as explained in the following two subsections for the two-dimensional and three-dimensional case. Furthermore, because of the shift of the interface's origin, the number of interface configurations becomes manageable and reduces to four in two dimensions as shown in Figure 5.4.

In the following, we suppose that the above coordinate transformations have already taken place. Thus, we always use (5.19) to describe our plane, which we write as

$$\left|n_x\right| x + \left|n_y\right| y + \left|n_z\right| z = s \tag{5.20}$$

for better readability.

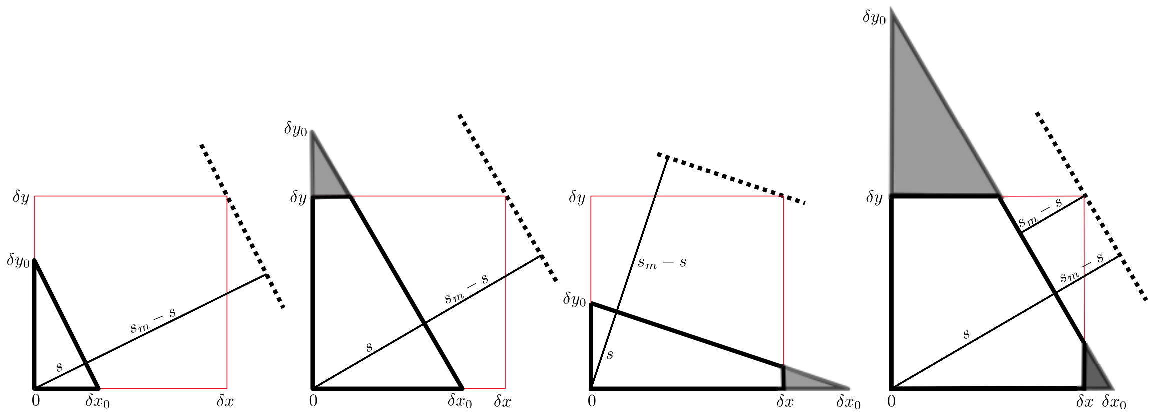


Figure 5.4.: Four cases of possible interface configurations in 2D as given in [Son03]. The gas volume is surrounded by red solid lines and the liquid volume by thick black lines. The line $s = |n_x|x+|n_y|y$ has the distance s to the origin, while the dashed line $s_m = \delta x_1+\delta x_2 = |n_x|x+|n_y|y$ has the distance s_m to the origin. The volume is computed in such a way that in the last case the gas volume would be computed (with distance $s_m - s$ to the origin). δx_0 and δy_0 mark the x- and y-intercepts of the line. δx and δy are the widths of the cell. Overlapping parts of the triangles are shaded gray.

5.4.2. Computation of distance from normal and volume fraction in 2D

Before we describe the three-dimensional case, we learn as much as we can from the simpler two-dimensional problem as described in [Son03]. Let δx_0 and δy_0 denote the x- and y-intercept of the line (which may lie inside or outside of the computational cell). Then, the liquid volume for all possible interface configurations can be expressed by

$$\underbrace{F\delta x\delta y}_{\text{fluid volume in the cell}} = \underbrace{\frac{1}{2}\delta x_0\delta y_0}_{\text{area of triangle}} - \underbrace{\frac{1}{2}\frac{\delta y_0}{\delta x_0}\left\langle \delta x_0 - \delta x\right\rangle^2}_{\text{overlap of triangle in } x\text{-direction}} - \underbrace{\frac{1}{2}\frac{\delta x_0}{\delta y_0}\left\langle \delta y_0 - \delta y\right\rangle^2}_{\text{overlap of triangle in } y\text{-direction}} \tag{5.21}$$

with $\langle\cdot\rangle := \max(\cdot, 0)$, which is a summation over all triangles depicted in Figure 5.4.

Naturally, all vectors (x, y) which lie on the line fulfill its equation

$$s = |n_x|x + |n_y|y. \tag{5.22}$$

Since this is true for the intercepts $(\delta x_0, 0)$ and $(0, \delta y_0)$, their substitution into the line equation yields

$$s = |n_x|\delta x_0 = |n_y|\delta y_0.$$

Then, equation (5.21) can be rewritten as

$$\begin{aligned} & 2F\delta x\delta y && = \delta x_0\delta y_0 - \frac{\delta y_0}{\delta x_0}\left\langle \delta x_0 - \delta x\right\rangle^2 - \frac{\delta x_0}{\delta y_0}\left\langle \delta y_0 - \delta y\right\rangle^2 \\ \Leftrightarrow\ & 2F\delta x\delta y && = \frac{s}{|n_x|}\frac{s}{|n_y|} - \frac{s}{|n_y|}\frac{|n_x|}{s}\left\langle \frac{s}{|n_x|} - \delta x\right\rangle^2 - \frac{s}{|n_x|}\frac{|n_y|}{s}\left\langle \frac{s}{|n_y|} - \delta y\right\rangle^2 \end{aligned}$$

$$\begin{aligned}
\Leftrightarrow \quad 2F\delta x|n_x|\delta y|n_y| \quad &= s^2 - |n_x|^2 \left\langle \frac{s}{|n_x|} - \delta x \right\rangle^2 - |n_y|^2 \left\langle \frac{s}{|n_y|} - \delta y \right\rangle^2 \\
\Leftrightarrow \quad 2F\delta x_1 \delta y_1 \quad &= s^2 - \langle s - \delta x_1 \rangle^2 - \langle s - \delta y_1 \rangle^2 \qquad (5.23)
\end{aligned}$$

with $\delta x_1 = |n_x|\delta x$ and $\delta y_1 = |n_y|\delta y$.

On the other hand, the gas phase consists of the opposite triangles than the liquid volume. Therefore, we define $s_m = \delta x_1 + \delta x_2$ together with the line equation

$$s_m = |n_x|x + |n_y|y.$$

This line intersects the point $(\delta x, \delta y)$ since

$$s_m = \delta x_1 + \delta x_2 = |n_x|\delta x + |n_y|\delta y.$$

Hence, we have defined a line parallel to (5.22). Only its distance to the origin has changed from s to s_m so that it intersects $(\delta x, \delta y)$; compare Figure 5.4.

Then the vapor volume is

$$2(1-F)\delta x_1 \delta y_1 = (s_m - s)^2 - \langle s_m - s - \delta x_1 \rangle^2 - \langle s_m - s - \delta y_1 \rangle^2 . \qquad (5.24)$$

In order to find a unified expression for equations (5.23) and (5.24) we write

$$2F_c \delta x_c \delta y_c = s_c^2 - \langle s_c - \delta y_c \rangle^2 \qquad (5.25)$$

with $F_c = \min(F, 1-F)$, $s_c = \min(s, s_m - s)$, $\delta x_c = \max(\delta x_1, \delta y_1)$ and $\delta y_c = \min(\delta x_1, \delta y_1)$.

These definitions can be explained as follows: We always choose to compute the smaller of the volumes F (be it liquid or vapor) and set the parameter s_c correspondingly. Since $F_c \leq 0.5$ and $s_m = \delta x_1 + \delta x_2 \leq 2\max(\delta x_1, \delta x_2)$, we know $s_c \leq 0.5 s_m \leq \delta x_c$.

Let us further explain why the unified equation (5.25) equals equations (5.23) and (5.24). On the left hand of (5.25), we have $2F_c \delta x_c \delta y_c = 2F_c \delta x_1 \delta y_1$. And on the right hand, the subtraction of $\langle s_c - \delta y_c \rangle$ now suffices, since for the computation of the smaller piece of volume in cases (1–4) of Figure 5.4, we have at most one part of the triangle which overlaps the computational cell and therefore, has to be subtracted from the big triangle.

We learn from the two-dimensional case that a unified expression for the volume can be derived and, instead of the four interface configurations, only two cases are really relevant since we always compute the smaller of the liquid and vapor volume. In three dimensions, where there are many more possible configurations, these considerations can be of great help.

5.4.3. Computation of distance from normal and volume fraction in 3D

Again, the following algorithm is taken from [Son03]. It requires that $\delta x_1 = |n_x|\delta x \geq \delta y_1 = |n_y|\delta y \geq \delta z_1 = |n_z|\delta z$. In order to achieve this, we do not really rotate the cell, but rename the coordinate axes if necessary. In three dimensions the unified formula for the

computation of the liquid or vapor volume is

$$6F_c\delta x_1\delta y_1\delta z_1 = s_c^3 - \langle s_c - \delta z_1\rangle^3 - \langle s_c - \delta y_1\rangle^3 - \langle s_c - \delta x_1\rangle^3 + \langle s_c - \delta y_1 - \delta z_1\rangle^3 \qquad (5.26)$$

with $F_c = \min(F, 1-F)$, $s_c = \min(s, s_m - s)$, $s = |n_x|\delta x_0 = |n_y|\delta y_0 = |n_z|\delta z_0$, $s_m = |n_x|\delta x = |n_y|\delta y = |n_z|\delta z$. Here, $s_c \leq 0.5 s_m \leq \max(\delta x_1, \delta y_1 + \delta z_1)$ since $F_c \leq 0.5$ and $s_m = \delta x_1 + \delta y_1 + \delta z_1 \leq 2\max(\delta x_1, \delta y_1 + \delta z_1)$. The five possible interface configurations are given in Figure 5.5. For a more detailed figure with descriptions of the tetrahedrons which comprise F_c in equation (5.26), we refer to [Son03, p. 556]. Then, the algorithm for the solution of equation (5.26) as proposed in [Son03] reads:

1. Set $F_c = \min(F, 1-F)$ and rotate the interface cell so that $\delta x_1 \geq \delta y_1 \geq \delta z_1$.
2. Set
$$s_c = (6F_c\delta x_1\delta y_1\delta z_1)^{\frac{1}{3}}.$$
If $s_c < \delta z_1$ go to step 6.
3. Set
$$s_c = 0.5\delta z_1 + \sqrt{2F_c\delta x_1\delta y_1 - \delta z_1^2/12}.$$
If $s_c < \delta y_1$ go to step 6.
4. If $\delta x_1 \geq \delta y_1 + \delta z_1$, then
$$s_c = F_c\delta x_1 + \frac{\delta y_1 + \delta z_1}{2}.$$
If $s_c \geq \delta y_1 + \delta z_1$ go to step 6.
5. With the initial value s_c set by one of the above steps, solve the following equation iteratively using the Newton iteration method:
$$6F_c\delta x_1\delta y_1\delta z_1 - s_c^3 + (s_c - \delta z_1)^3 + (s_c - \delta y_1)^3 + \langle s_c - \delta x_1\rangle^3 = 0.$$
Since $\langle s_c - \delta x_1\rangle^3 = \max(s_c - \delta x_1, 0)^3$, we differentiate in our implementation between the cases:
    - $\delta x_1 \leq s_c$: Solve
    $$6F_c\delta x_1\delta y_1\delta z_1 - s_c^3 + (s_c - \delta z_1)^3 + (s_c - \delta y_1)^3 + (s_c - \delta x_1)^3 = 0$$
    iteratively and go to step 6.
    - $\delta x_1 > s_c$: Solve
    $$6F_c\delta x_1\delta y_1\delta z_1 - s_c^3 + (s_c - \delta z_1)^3 + (s_c - \delta y_1)^3 = 0$$
    iteratively and go to step 6.
6. If $F \leq 0.5$ set $s = s_c$ else set $s = s_m - s_c$.

This algorithm concludes the geometric reconstruction of the interface. In summary, we have used the LS function to obtain a linear reconstruction which is very close to the real zero level-set of ϕ, and we have used the VOF function for a shift of the reconstructed interface for a better conservation of mass. With the help of the reconstructed interface, we are now able to compute the VOF fluxes in equation (5.8) since we can now evaluate the advected liquid volumes (δV_F) geometrically.

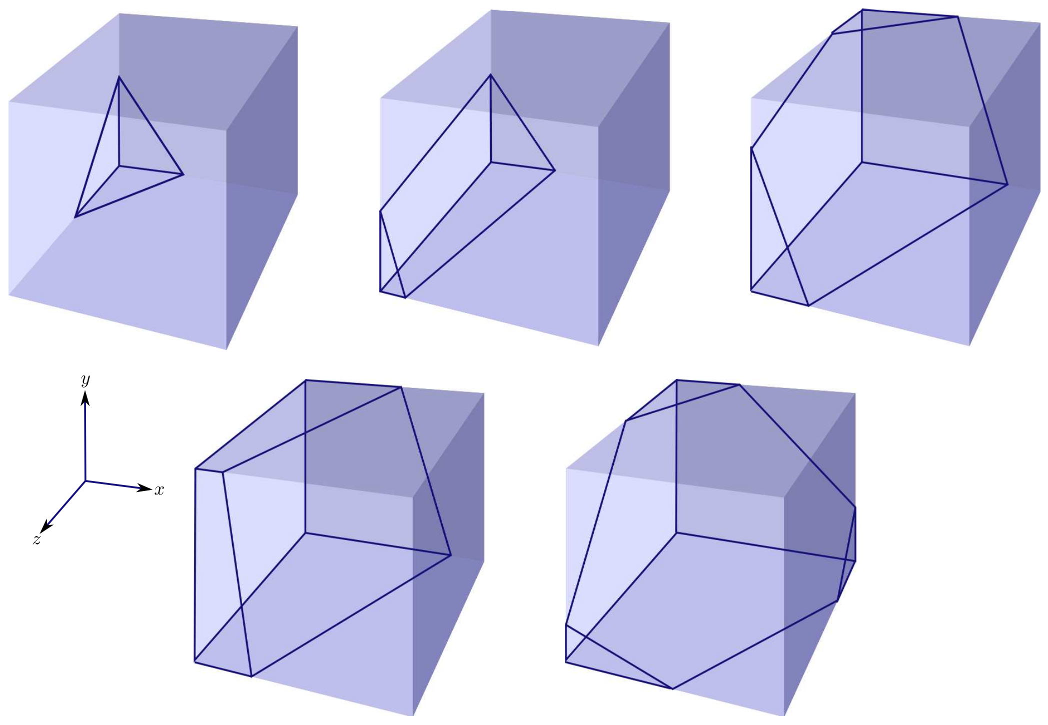


Figure 5.5.: The five possible interface configuration in three dimensions.

5.5. Computation of the advected liquid volumes from the reconstructed interface

We define the advected liquid volumes (δV_F), which we need in equation (5.8), by

$$
\begin{aligned}
(\delta V_F)^{\rightarrow}_{i+\frac{1}{2},j,k} &= \left\{\begin{array}{ll} (\delta V_F)_0(\boldsymbol{n}, s, |u|\,\delta t, \delta y, \delta z) & \text{if } n_x u \geq 0 \\ F\delta x\delta y\delta z - (\delta V_F)_0(\boldsymbol{n}, s, \delta x - |u|\delta t, \delta y, \delta z) & \text{if } n_x u < 0 \end{array}\right\} \text{in } (i,j,k) \\
(\delta V_F)^{\leftarrow}_{i+\frac{1}{2},j,k} &= \left\{\begin{array}{ll} (\delta V_F)_0(\boldsymbol{n}, s, |u|\,\delta t, \delta y, \delta z) & \text{if } n_x u \geq 0 \\ F\delta x\delta y\delta z - (\delta V_F)_0(\boldsymbol{n}, s, \delta x - |u|\delta t, \delta y, \delta z) & \text{if } n_x u < 0 \end{array}\right\} \text{in } (i+1,j,k) \\
(\delta V_F)^{\rightarrow}_{i,j+\frac{1}{2},k} &= \left\{\begin{array}{ll} (\delta V_F)_0(\boldsymbol{n}, s, \delta x, |v|\delta t, \delta z) & \text{if } n_y v \geq 0 \\ F\delta x\delta y\delta z - (\delta V_F)_0(\boldsymbol{n}, s, \delta x, \delta y - |v|\delta t, \delta z) & \text{if } n_y v < 0 \end{array}\right\} \text{in } (i,j,k) \\
(\delta V_F)^{\leftarrow}_{i,j+\frac{1}{2},k} &= \left\{\begin{array}{ll} (\delta V_F)_0(\boldsymbol{n}, s, \delta x, |v|\delta t, \delta z) & \text{if } n_y v \geq 0 \\ F\delta x\delta y\delta z - (\delta V_F)_0(\boldsymbol{n}, s, \delta x, \delta y - |v|\delta t, \delta z) & \text{if } n_y v < 0 \end{array}\right\} \text{in } (i,j+1,k) \\
(\delta V_F)^{\rightarrow}_{i,j,k+\frac{1}{2}} &= \left\{\begin{array}{ll} (\delta V_F)_0(\boldsymbol{n}, s, \delta x, \delta y, |w|\delta t) & \text{if } n_z w \geq 0 \\ F\delta x\delta y\delta z - (\delta V_F)_0(\boldsymbol{n}, s, \delta x, \delta y, \delta z - |w|\delta t) & \text{if } n_z w < 0 \end{array}\right\} \text{in } (i,j,k) \\
(\delta V_F)^{\leftarrow}_{i,j,k+\frac{1}{2}} &= \left\{\begin{array}{ll} (\delta V_F)_0(\boldsymbol{n}, s, \delta x, \delta y, |w|\delta t) & \text{if } n_z w \geq 0 \\ F\delta x\delta y\delta z - (\delta V_F)_0(\boldsymbol{n}, s, \delta x, \delta y, \delta z - |w|\delta t) & \text{if } n_z w < 0 \end{array}\right\} \text{in } (i,j,k+1).
\end{aligned}
\tag{5.27}
$$

In the following, we explain the meaning of the arrows, the definition of $(\delta V_F)_0$ for a subregion and the distinction between positive and negative outcomes of the velocity's and

normal component's product in equation (5.27). Then we proceed with an algorithm for the computation of $(\delta V_F)_0$, which again relies on relation (5.26).

As already mentioned, the superscript arrows $\leftrightarrows$ of (δV_F) indicate from which cell we take the values F, $\boldsymbol{n}$, s, δx, δy and δz. This cell is the so-called donor cell. For example, if the velocity component u is positive, there is flux from the donor cell (i, j, k) to $(i + 1, j, k)$, which we indicate by the arrow from left to right, because of the flux direction $(i, j, k) \to (i + 1, j, k)$. On the other hand, if the velocity is negative, we have flux from the donor cell $(i + 1, j, k)$ to (i, j, k), which we indicate by the arrow from right to left, because of the flux direction $(i, j, k) \leftarrow (i + 1, j, k)$. Both situations are depicted in Figure 5.1.

Additionally, we now define $(\delta V_F)_0 = (\delta V_F)_0(\boldsymbol{n}, s, \delta\tilde{x}, \delta\tilde{y}, \delta\tilde{z})$ to be the liquid volume in the subregion $\delta\tilde{x} \times \delta\tilde{y} \times \delta\tilde{z}$, where $0 \le \delta\tilde{x} \le \delta x$, $0 \le \delta\tilde{y} \le \delta y$ and $0 \le \delta\tilde{z} \le \delta z$. This subregion must always include the origin of s since the algorithm for the computation of F, which shall be given in Subsection 5.5.1, relies on this property.

Let us try to understand what this requirement means in two dimensions. Therefore, let $u > 0$ and go back to the 2D example illustrated in Figure 5.6. Recall that in Subsection 5.4.1, the number of interface configurations is reduced by the following two criteria:

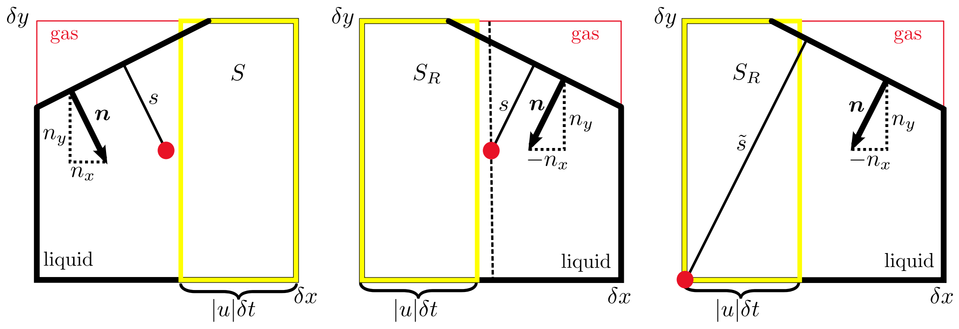
1. We reflect the plane in such a way, that all components of its unit normal $\boldsymbol{n} = (n_x, n_y, n_z)^T$ become negative.
2. We shift the plane's origin in our computational cell from the cell center to the lower left hand corner of the cell. This corner will lie in the liquid region and will be farthest away from the interface.

We define the subregion S with the vertices $(\delta x - |u|\delta t, 0)$, $(\delta x, 0)$, $(\delta x - |u|\delta t, \delta y)$ and $(\delta x, \delta y)$ as illustrated in Figure 5.6. First, let $n_x \ge 0$ as shown in Figure 5.6(a). In this case, the origin lies in the reflected image S_R of the subregion S after the computation of the reflected plane and the computation of the new position of the origin. Then, we may compute the advected liquid volume directly in the subregion S_R. Now, let $n_x < 0$ as shown in Figure 5.6(b). Then, the origin is not in the subregion S after the computation of the new position of the origin (here, no reflection is necessary). Then, we simply compute the volume of the subregion with the vertices $(0, 0)$, $(\delta x - |u|\delta t, 0)$, $(0, \delta y)$ and $(\delta x - |u|\delta t, \delta y)$ which *includes* the origin and subtract this volume from the overall liquid volume $F\delta x\delta y$ to obtain the volume that is advected across the cell boundary. That is why the distinction between the cases $n_x u \ge 0$ and $n_x u < 0$ is necessary in equation (5.27).

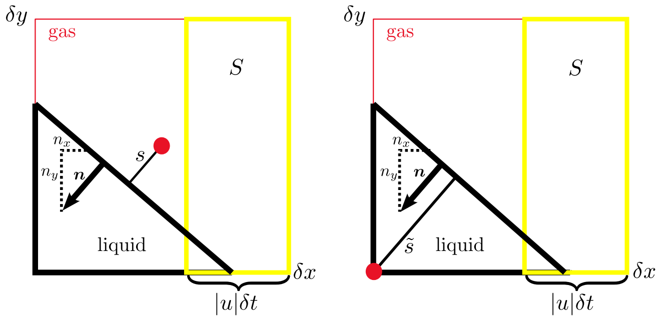
5.5.1. Computation of volume fraction from normal and distance in 3D

Finally, we describe the computation of $(\delta V_F)_0$ as done in [Son03]. Thus, if the cell in which we wish to compute the volume flux is a single-phase cell, where $F = 0$ or 1, we simply set $(\delta V_F) = F\delta\tilde{x}\delta\tilde{y}\delta\tilde{z}$ in (5.27). Otherwise we use relation (5.26) and obtain the following algorithm for the computation of $(\delta V_F)_0$ in (5.27):

1. Rename the axes so that $\delta\tilde{x}_1 = |n_x|\delta\tilde{x} \ge \delta\tilde{y}_1 = |n_y|\delta\tilde{y} \ge \delta\tilde{z}_1 = |n_z|\delta\tilde{z}$.
2. Set $\tilde{s}_m = \delta\tilde{x}_1 + \delta\tilde{y}_1 + \delta\tilde{z}_1$ and $\tilde{s}_c = \min(s, \tilde{s}_m - s)$.
3. If $\tilde{s}_m \le s$, the whole subregion is filled with liquid and we set $F_c = 1$. If $s < 0$, set $F_c = 0$.



(a) $n_x u \geq 0$



(b) $n_x u < 0$

Figure 5.6.: Placement of origin depending on the sign of $n_x u$. For the computation of the volume fractions, the origin needs to be in the considered subregion S marked by yellow boxes. For $n_x u \geq 0$, the picture in the middle shows the reflected plane which is computed across the vertical dashed line. Then, the third picture shows the placement of the origin in the liquid region as far from the interface as possible. Still, in this case the origin remains in the reflected image S_R of the subregion. If $n_x u < 0$, a reflection is unnecessary since $n_x < 0$ already, but the newly computed origin remains outside the subregion S.

4. We obtain the fluid volume fraction in the subregion by the following distinctions:

   (a) If $s_c < \delta\tilde{z}_1$,
   $$F_c = \frac{s_c^3}{6\delta\tilde{x}_1\delta\tilde{y}_1\delta\tilde{z}_1}.$$

   (b) If $\delta\tilde{z}_1 \leq s_c < \delta\tilde{y}_1$,
   $$F_c = \frac{s_c^2 - \delta\tilde{z}_1 s_c + \delta\tilde{z}_1^2/3}{2\delta\tilde{x}_1\delta\tilde{y}_1},$$
   which is the solution of $6F_c\delta\tilde{x}_1\delta\tilde{y}_1\delta\tilde{z}_1 = s_c^3 - (s_c - \delta\tilde{z}_1)^3$.

   (c) If $\delta\tilde{y}_1 + \delta\tilde{z}_1 \leq s_c \leq \delta\tilde{x}_1$,
   $$F_c = \frac{2s_c - \delta\tilde{y}_1 - \delta\tilde{z}_1}{2\delta\tilde{x}_1},$$
   which is the solution of
   $$6F_c\delta\tilde{x}_1\delta\tilde{y}_1\delta\tilde{z}_1 = s_c^3 - (s_c - \delta\tilde{z}_1)^3 - (s_c - \delta\tilde{y}_1)^3 + (s_c - \delta\tilde{y}_1 - \delta\tilde{z}_1)^3$$

.

(d) Otherwise,

$$F_c = \frac{s_c^3 - (s_c - \delta\tilde{z}_1)^3 - (s_c - \delta\tilde{y}_1)^3 - \langle s_c - \delta\tilde{x}_1\rangle^3}{6\delta\tilde{x}_1\delta\tilde{y}_1\delta\tilde{z}_1}.$$

5. If $s \leq 0.5\tilde{s}_m$, set $(\delta V_F)_0 = F_c\delta\tilde{x}\delta\tilde{y}\delta\tilde{z}$. Otherwise, set $(\delta V_F)_0 = (1 - F_c)\delta\tilde{x}\delta\tilde{y}\delta\tilde{z}$.

With the help of this algorithm, we are able to compute the volume fluxes $(\delta V_F)_0$ which are required for the computation of the advected liquid volumes in equation (5.27).

Remark: This algorithm is also used for the initialization of F at the beginning of the computation. Since the LS values are given by an analytic distance function or by an initial reinitialization step over all grid cells, we can use these values to compute the volume fractions. Then, in the above algorithm, $\delta\tilde{x} = \delta x$, $\delta\tilde{y} = \delta y$ and $\delta\tilde{z} = \delta z$. In the same way, this algorithm is used for the computation of boundary values for the VOF function if a boundary cell contains a reconstructed interface; see Section 5.7.

5.6. Reinitialization of the LS function

The last step of our CLSVOF method is the reinitialization of the LS function with the help of the reconstructed interface, i.e. we want the LS function to be the exact signed normal distance to the reconstructed interface.

Before the reinitialization of the LS function and after the final reconstruction of the interface, the volume fractions are truncated to avoid unphysical flotsam and jetsam, which are generally created by round-off errors, i.e.

$$F_{i,j,k} = \begin{cases} 0 & \text{if } \phi_{i,j,k} < -\varepsilon \text{ or } F_{i,j,k} < 0 \\ 1 & \text{if } \phi_{i,j,k} > \varepsilon \text{ or } F_{i,j,k} > 1. \end{cases} \tag{5.28}$$

By restricting F to lie between 0 and 1, we introduce a truncation error and lose a negligible amount of mass in the process. However, this mass loss is typically an order of magnitude less than that lost in LS methods [LFO06].

The reconstructed interface consists of piecewise linear plane segments as well as the plane segments at the cell boundaries that connect a plane in one cell with the plane in the cell next to it. The sign of the LS function can be determined by the sign of the VOF function $\text{sgn}(F - 0.5)$. The magnitude of ϕ is calculated as the shortest distance from the cell center to any of the reconstructed interface segments. Thus, in the cell (i, j, k), we have to locate the point $\boldsymbol{x}_s$ on the interface which has the shortest distance to a neighboring cell center $\boldsymbol{x}'$ of a cell (i', j', k').

In summary the reinitialization algorithm consists mainly of the following two parts, which are illustrated in Figure 5.7 (a)–(c):

1. Try the simple cases: The point $\boldsymbol{x}_s$ can be found at the face center or face corner of cell (i, j, k) as in (c). Or, the projection of $\boldsymbol{x}'$ onto the interface lies in cell (i, j, k) as in (a).

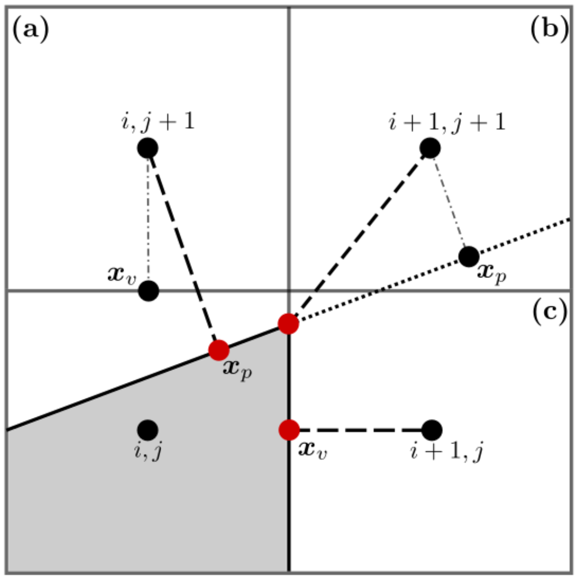


Figure 5.7.: Level-set redistancing: The gray area marks one phase and the white area marks the other one, while the thick black lines denote the reconstructed interface. The red points, which are connected to their cell cell centers by dashed lines, are those with the shortest distance to the interface. Possible candidates for these points are projection points $\boldsymbol{x}_p$ from the cell center onto the interface or points $\boldsymbol{x}_v$ on the boundary of cell (i, j, k); compare Table 5.1.

2. Do the difficult cases: The point $\boldsymbol{x}_s$ is one of the vertices of the polygon in cell (i, j, k) as in (b). In 3D, $\boldsymbol{x}_s$ could also be the projection of $\boldsymbol{x}'$ onto one of the line segments of the polygon.

For the computation of $\boldsymbol{x}_s$ we have mainly consulted the works of Sussman and Puckett [SP00] (2D/3D), Sussman [Sus03] (2D/3D), Son and Hur [SH02] (2D), Son [Son03] (3D) and Wang et al. [WYKS09] (2D/3D). Most of the algorithms given in these articles describe the distance computation well for the simple interface configurations. However in three dimensions, the most difficult case is the one where we need to compute the vertices of the interface, which is not dealt with in detail in the works of Sussman [SP00] and [Sus03]. Nevertheless, the technicalities of the vertex computation can be found in Son's article [Son03]. Then again, Wang et al. [WYKS09, p. 234] criticize Son's 3D algorithm for its complexity and state that:

> Moreover, for a 3D case, the projection point of point (i', j', k') onto the side of the polygon is also the possible closest point on the polygon boundary, which is not mentioned in the reference [i.e. Son's article [Son03], author's note]. Obviously, if all the possible closest points are determined and the distances are calculated, the procedure will be complicated and computationally expensive. In this study, a much simple [*sic*] and efficient procedure to locate the closest point on the interface boundary is presented below.

In what follows, Wang et al. describe their own new distance computation procedure in two and three dimensions. Unfortunately, while this procedure might successfully work in

Point	Description
$\boldsymbol{x}'$	cell center of (i', j', k') within the narrow band about cell (i, j, k)
$\boldsymbol{x}_s$	the sought point on the interface in cell (i, j, k) with shortest distance to $\boldsymbol{x}'$
$\boldsymbol{x}_v$	point on the boundary of cell (i, j, k) with shortest distance to $\boldsymbol{x}'$, candidate for $\boldsymbol{x}_s$
$\boldsymbol{x}_p$	projection of $\boldsymbol{x}'$ onto the interface, candidate for $\boldsymbol{x}_s$

Table 5.1.: Description of points used in the reinitialization algorithm.

2D, we can show it to fail in 3D, which we will do in Subsection 5.6.1. Furthermore, we cannot really share Wang et al.'s criticism for Son's algorithm: Although his procedure is quite complicate to implement, at least all the right vertices of the plane segments are found reliably in three dimensions. Moreover, while the projection point itself is not mentioned in Son's work of 2003 [Son03], the algorithm here is only an extension to the original one of 2002 [SH02], where the projection point on the interface segment is indeed mentioned as a possibility for being the closest point to $\boldsymbol{x}'$.

In this thesis, we propose another simple procedure for computing the interface vertices in 3D. In contrast to Son's approach, we do not rotate the interface to work with the reduced interface configurations (Fig. 5.5), which would render the computation more complicated since we also need to transform the coordinates of the vertices back into the original system. Instead, we work directly in the original system with a larger amount of interface configurations and still compute the vertices efficiently by taking into account the distances of the *vertices of the computational cell* to the interface. In addition, we describe the more simple cases of the reinitialization procedure in detail by following the aforementioned articles and [WYKS09] in particular.

In the following, let (i, j, k) be a given cell with a reconstructed interface as described in Subsection 5.3.2. Now, (i', j', k') with $|i' - i| \leq K$, $|j' - j| \leq K$ and $|k' - k| \leq K$ are the computational cells within a narrow band about (i, j, k), where K denotes the width of the band. In all computational cells of our computational domain we initialize the auxiliary function ϕ' with a very large value. Further, we again denote the cell center of (i', j', k') by $\boldsymbol{x}'$ and the point on the interface with the shortest distance to $\boldsymbol{x}'$ by $\boldsymbol{x}_s$. In Table 5.1, we give an overview and description of all points used in the reinitialization algorithm.

1. If (i, j, k) is a cell with a reconstructed interface, the value of the auxiliary function ϕ' simply becomes the distance of the cell center to the reconstructed interface, i.e.
$$\phi'_{i,j,k} = \operatorname{sgn}(F_{i,j,k} - 0.5)|\phi^R_{i,j,k}| = \operatorname{sgn}(F - 0.5)\left|n_x x' + n_y y' + n_z z' + d\right|, \qquad (5.29)$$
where d is now the distance of the computational domain's origin to the plane (and no longer from the corner or center of the computational cell).
2. As a candidate for $\boldsymbol{x}_s$, determine the point $\boldsymbol{x}_v$ on the boundary of cell (i, j, k) with the shortest distance to the cell center of (i', j', k'). This point will always be found at the corner of a cell face of (i, j, k) or at the center of a cell face of (i, j, k). Its coordinates are given by
$$\boldsymbol{x}_v = (x_{i+\frac{l}{2}}, y_{j+\frac{m}{2}}, z_{k+\frac{n}{2}})$$

with[1]

$$\begin{aligned} l &= \max(-1, \min(1, i' - i)) \\ m &= \max(-1, \min(1, j' - j)) \\ n &= \max(-1, \min(1, k' - k)). \end{aligned}$$

Then, we compute the signed distance of the point $\boldsymbol{x}_v$ to the reconstructed interface by

$$D_{\phi^R_{i,j,k}}(\boldsymbol{x}_v) = d + n_x x_v + n_y y_v + n_z z_v.$$

This distance is positive if the interface's normal vector points to the same side of the interface, where $\boldsymbol{x}_v$ lies.
If the sign of this distance is different from the sign of the LS function in the cell center $\boldsymbol{x}'$ of cell (i', j', k'), i.e. $D_{\phi^R_{i,j,k}}(\boldsymbol{x}_v)\,\mathrm{sgn}(\phi_{i',j',k'}) \le 0$, we know that $\boldsymbol{x}_v$ and $\boldsymbol{x}'$ belong to different phases (Fig. 5.7 (c)). Then, we compute the distance $d(\boldsymbol{x}_v, \boldsymbol{x}')$ between these two points by

$$d(\boldsymbol{x}_v, \boldsymbol{x}') = \sqrt{(x_v - x')^2 + (y_v - y')^2 + (z_v - z')^2}.$$

After that, we update ϕ' by

$$\phi'_{i',j',k'} = \mathrm{sgn}(F_{i',j',k'} - 0.5) \min\left(d(\boldsymbol{x}_v, \boldsymbol{x}'), |\phi'_{i',j',k'}|\right).$$

3. $D_{\phi^R_{i,j,k}}(\boldsymbol{x}_v)\,\mathrm{sgn}(\phi_{i',j',k'}) > 0$, both points $\boldsymbol{x}_v$ and $\boldsymbol{x}'$ belong to the same phase; see Figure 5.7(a)). Then, the next likely candidate for $\boldsymbol{x}_s$, is the projection of $\boldsymbol{x}'$ onto the interface, i.e. the point

$$\boldsymbol{x}_p = P(\boldsymbol{x}') = \boldsymbol{x}' - D_{\phi^R_{i,j,k}}(\boldsymbol{x}')\boldsymbol{n}.$$

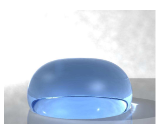
If $\boldsymbol{x}_p$ is inside cell (i, j, k), this point must lie directly on the interface and, by definition, is the one with the smallest distance to $\boldsymbol{x}'$. In this case we set

$$\phi'_{i',j',k'} = \mathrm{sgn}(F_{i',j',k'} - 0.5) \min\left(d(\boldsymbol{x}_p, \boldsymbol{x}'), |\phi'_{i',j',k'}|\right).$$

If the projection point is not in cell (i, j, k), we proceed with the next step.

4. With the above part of the algorithm all the simple standard cases are dealt with. Indeed, we now know that $\boldsymbol{x}_s$ must lie on one of the intersection points of the plane in cell (i, j, k) with the boundary of cell (i, j, k) (i.e. on one of the vertices of the at most 6-sided polygons drawn in Figure 5.5). Additionally, we have to take into account that $\boldsymbol{x}_s$ could be the projection point of a neighboring cell onto the side of the polygon.

[1]This definition of $\boldsymbol{x}_v$ can be understood as follows (cf. Fig. 5.7): if (i', j', k') is a direct neighbor of (i, j, k) (e.g. $i' = i + 1, j = j, k = k$), then $l = \max(-1, \min(1, i + 1 - i)) = \max(-1, 1) = 1$, $m = \max(-1, \min(1, j - j)) = 0$ and $n = 0$ so that $\boldsymbol{x}_v = x_{i+\frac{1}{2},j,k}$, which is a face center of cell (i, j, k). If (i', j', k') is a diagonal neighbor (e.g. $i' = i + 1, j = j + 1, k = k$), then $l = 1$, $j = 1$, $k = 0$ and $\boldsymbol{x}_v = x_{i+\frac{1}{2},j+\frac{1}{2},k}$, which is a face corner.



In order to find $\boldsymbol{x}_s$, we propose the following algorithm: Set $\min_{\text{dist}}$ to a very large value and consider the vertices of the computational cell which are connected by 12 edges. For these edges $s = 1, ..., 12$:

a) Consider the pair of vertices $(\boldsymbol{p}_{s_1}, \boldsymbol{p}_{s_2})$ of the computational cell which are connected by this edge.

b) Compute the distances $D(\boldsymbol{p}_{s_1})$ and $D(\boldsymbol{p}_{s_2})$ of these points to the interface.[2]

c) If $D(\boldsymbol{p}_{s_1})D(\boldsymbol{p}_{s_2}) > 0$, the interface does not cut the edge (since both points lie on the same side of the interface). In this case, go back to (a) and consider the next edge. Otherwise, the interface does cut the edge and we proceed with the next step.

d) Compute the intersection point $\boldsymbol{p}^{\text{is}}_{s_1 s_2}$ of the interface with the edge. Set $\min_{\text{dist}} = \min\left(d\left(\boldsymbol{p}^{\text{is}}_{s_1 s_2}, \boldsymbol{x}'\right), \min_{\text{dist}}\right)$.

e) Compute the projection of $\boldsymbol{x}'$ onto both of the polygon's interface segments which have $\boldsymbol{p}^{\text{is}}_{s_1 s_2}$ as a common vertex. We denote the projection points by $\boldsymbol{p}^{\text{pr}_1}_{s_1 s_2}$ and $\boldsymbol{p}^{\text{pr}_2}_{s_1 s_2}$. If $\boldsymbol{p}^{\text{pr}_1}_{s_1 s_2}$ is in cell (i, j, k), set $\min_{\text{dist}} = \min\left(d\left(\boldsymbol{p}^{\text{pr}_1}_{s_1 s_2}, \boldsymbol{x}'\right), \min_{\text{dist}}\right)$. If $\boldsymbol{p}^{\text{pr}_2}_{s_1 s_2}$ is in cell (i, j, k), set $\min_{\text{dist}} = \min\left(d\left(\boldsymbol{p}^{\text{pr}_2}_{s_1 s_2}, \boldsymbol{x}'\right), \min_{\text{dist}}\right)$.

f) Go to (a) and consider the next edge. If all 12 edges have been taken care of, go to the last step.

g) Set $\phi'_{i',j',k'} = \operatorname{sgn}(F_{i',j',k'} - 0.5) \min\Big(|\min_{\text{dist}}|, |\phi'_{i',j',k'}|\Big)$

5. Compute the maximum diagonal length $d_{\max}$ of all computational cells. In cells which lie outside the narrow band of K cells, set

$$\phi_{i,j,k} = \begin{cases} -Kd_{\max} - d_{\max} & \text{if } \phi_{i,j,k} < 0 \\ Kd_{\max} + d_{\max} & \text{if } \phi_{i,j,k} > 0, \end{cases} \tag{5.30}$$

i.e. these cells are filled with a constant distance to the interface. Our computation works well on uniform and most anisotropic grids, as long as the difference in mesh size in the respective space directions is not too large. However, this is just an implementation detail which could be remedied by adapting $d_{\max}$ accordingly.

Remark: Sussman [SP00] mentions a further method for the reinitialization of the LS function, which can be used as follows: After the transport step (5.8) and the final interface reconstruction, we set the LS function ϕ_{n+1} to be the intercept $d_{i,j,k}$ of the reconstructed interface $\phi^{R,n+1}_{i,j,k}$. In cells, where no reconstructed interface exists, we use the Hamilton-Jacobi-reinitialization of the standard LS method as described in Subsection 4.1.1.

What makes the algorithm described in step 4 fast is that by comparing the sign of the distance of the cell's vertices to the interface, we know very quickly which edges of the computational cell intersect the polygon. Then, since the coordinates of the vertices of

[2]See equation (5.10) for the definition of the *signed distance* of a point $\boldsymbol{p}$ to a plane. This distance is positive, if $\boldsymbol{p}$ lies on the same side of the plane towards which the unit normal points and negative otherwise.

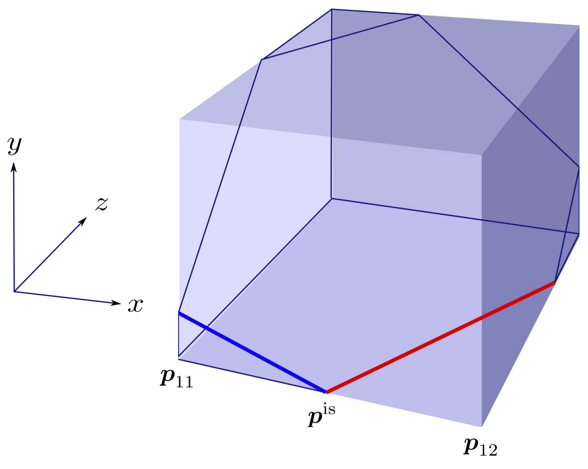


Figure 5.8.: Illustration for finding the vertices of the polygon and the projection points onto the segments of the polygon. The vertices $\boldsymbol{p}_{11} = (p_1, p_2, p_3)$ and $\boldsymbol{p}_{12} = (p_1 + \delta x, p_2, p_3)$ of the computational cell are connected by the same edge which is intersected by the 6-sided polygon. This intersection point is denoted by $\boldsymbol{p}^{\text{is}}$. The segments of the polygon which are connected to $\boldsymbol{p}^{\text{is}}$ lie in the $y = p_2$-face. (thick red line) and the $z = p_3$-face (thick blue line) of the cell.

the edges are already stored as grid information, we can also compute the vertices of the polygon and the projection points onto the line segments very efficiently.

Let us exemplify their computation for a pair of vertices $\boldsymbol{p}_{11} = (p_1, p_2, p_3)$ and $\boldsymbol{p}_{12} = (p_1 + \delta x, p_2, p_3)$ whose edge intersects the polygon (cf. Fig. 5.8). Then, $\boldsymbol{p}^{\text{is}}$ can simply be computed as:

$$\boldsymbol{p}^{\text{is}} = \begin{pmatrix} \frac{n_y p_2 + n_z p_3 + d}{-n_x} \\ p_2 \\ p_3 \end{pmatrix}$$

Furthermore, we know that the polygon segments which intersect this vertex must lie within the faces $y = p_2$ and $z = p_3$ of the computational cell.[3] They obey the linear equations

$$n_x x + n_z z + (d + n_y p_2) = 0 \tag{5.31}$$

and

$$n_x x + n_y y + (d + n_z p_3) = 0. \tag{5.32}$$

We normalize these equations by division with $\sqrt{n_x^2 + n_y^2}$ in the first and by division with

[3] Here, we define the x_f-face of cell (i, j, k) as

$$\left\{(x, y, z) : x = x_f,\ y_{j-\frac{1}{2}} \leq y \leq y_{j+\frac{1}{2}},\ z_{k-\frac{1}{2}} \leq z \leq z_{k+\frac{1}{2}}\right\},$$

and the other faces analogously.

$\sqrt{n_x^2+n_z^2}$ in the second case. Thereby we obtain the new parameters

$$\begin{aligned}\tilde{n}_x &= n_x/\sqrt{n_x^2+n_y^2}\\ \tilde{n}_z &= n_z/\sqrt{n_x^2+n_y^2}\\ \tilde{d} &= (d+n_y p_2)/\sqrt{n_x^2+n_y^2}\end{aligned}$$

for the first polygon segment and

$$\begin{aligned}\hat{n}_x &= n_x/\sqrt{n_x^2+n_z^2}\\ \hat{n}_y &= n_y/\sqrt{n_x^2+n_z^2}\\ \hat{d} &= (d+n_z p_3)/\sqrt{n_x^2+n_z^2}.\end{aligned}$$

for the second polygon segment. Then equations (5.31) and (5.32) become

$$\tilde{n}_x x + \tilde{n}_z z + \tilde{d} = 0 \tag{5.33}$$

and

$$\hat{n}_x x + \hat{n}_y y + \hat{d} = 0. \tag{5.34}$$

In the next step, we compute the projection of $\boldsymbol{x}'$ onto both of the polygon's interface segments (5.33) and (5.34). Since we know that these segments lie in the faces $y = p_2$ and $z = p_3$, respectively, these entries of the coordinates of the projection point are fixed, while the remaining two entries can be computed by the standard projection onto the normalized line equations (5.33) and (5.34). Thus,

$$\boldsymbol{p}^{\mathrm{pr}_1} = \begin{pmatrix} x_1' - (\tilde{n}_x x_1' + \tilde{n}_z x_3' + \tilde{d})\tilde{n}_x \\ p_2 \\ x_3' - (\tilde{n}_x x_1' + \tilde{n}_z x_3' + \tilde{d})\tilde{n}_z \end{pmatrix}$$

and

$$\boldsymbol{p}^{\mathrm{pr}_2} = \begin{pmatrix} x_1' - (\hat{n}_x x_1' + \hat{n}_y x_2' + \hat{d})\hat{n}_x \\ x_2' - (\hat{n}_x x_1' + \hat{n}_y x_2' + \hat{d})\hat{n}_y \\ p_3 \end{pmatrix}.$$

Incidentally, the computation of the projection point can be handled by a single function in our computer program, which is exemplified in Algorithm 1. This function would receive the parameters

$$(x_1', x_2', x_3', x_1', p_2, x_3', n_x, 0.0, n_z, d + n_y p_2, i, j, k)$$

in the first and

$$(x_1', x_2', x_3', x_1', x_2', p_3, n_x, n_y, 0.0, d + n_z p_3, i, j, k)$$

in the second case and return the distance from $\boldsymbol{x}'$ to the projection point, if it is in cell (i, j, k).

With this example we conclude the description of the reinitialization of the LS function

Algorithm 1: C++ function for the computation of the projection point onto the line segment.

```
double DistanceProjectionPointOnLineSegment(double x1, double x2,
double x3, double x, double y, double z,
double nx, double ny, double nz, double d,
int i, int j, int k){
  double root=sqrt(nx*nx+ny*ny+nz*nz);
  nx/=root;
  ny/=root;
  nz/=root;
  d/=root;
  double dist=CompDistToPlane(x,y,z,nx,ny,nz,d);
  if(IsPointInCell(x-nx*dist,y-ny*dist,z-nz*dist,i,j,k))
    return (EuclideanDistance(x1,y2,z2,
            x-nx*dist,y-ny*dist,z-nz*dist));
  else return 1e15;
}
```

with the help of the reconstructed interface. In the following, we present a further vertex finding algorithm by Wang et al. [WYKS09] and show its failure in three dimensions.

5.6.1. The vertex finding algorithm by Wang et al.

In this subsection we present the vertex finding algorithm by Wang et al. [WYKS09]. We explain how this algorithm works in 2D and prove that it cannot be easily extended to 3D as claimed by the authors. The algorithm in [WYKS09] reads as follows: Determine the auxiliary quantity $\boldsymbol{dx}_p = (dx_p, dy_p, dz_p)$, which is the distance from the projection point $\boldsymbol{x}_p$ to the nearest boundary of cell (i, j, k) in each coordinate direction

$$\begin{aligned} dx_p &= \max(|x_p - x_i| - 0.5\Delta x, 0) \\ dy_p &= \max(|y_p - y_j| - 0.5\Delta y, 0) \\ dz_p &= \max(|z_p - z_k| - 0.5\Delta z, 0). \end{aligned}$$

The auxiliary quantities x_f, y_f, and z_f are the three faces of cell (i, j, k) which are nearest to the projection point given by

$$\begin{aligned} x_f &= x_i + \operatorname{sgn}(x_p - x_i)0.5\Delta x_i \\ y_f &= y_j + \operatorname{sgn}(y_p - y_j)0.5\Delta y_j \\ z_f &= z_k + \operatorname{sgn}(z_p - z_k)0.5\Delta z_k. \end{aligned}$$

Now, we determine the nearest of these faces that is *intercepted* by the interface by computing

$$\max(dx_p|n_x|, dy_p|n_y|, dz_p|n_z)$$

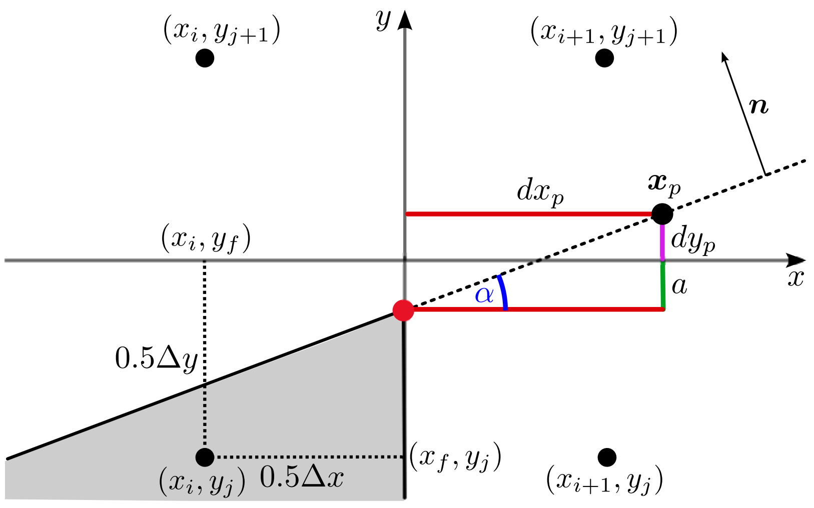


Figure 5.9.: Example 1 for the vertex finding algorithm by Wang et al. in 2D. The shortest distance from $\boldsymbol{x}' = (x_{i+1}, y_{j+1})$ to the interface (black line) in cell (i, j, k) is to be found. This point is marked with a red dot.

and set

$$\begin{aligned} x_s &= x_f \quad \text{if } dx_p|n_x| = \max(dx_p|n_x|, dy_p|n_y|, dz_p|n_z) \\ y_s &= y_f \quad \text{if } dy_p|n_y| = \max(dx_p|n_x|, dy_p|n_y|, dz_p|n_z) \\ z_s &= z_f \quad \text{if } dz_p|n_z| = \max(dx_p|n_x|, dy_p|n_y|, dz_p|n_z). \end{aligned}$$

If we have so found the nearest face to $\boldsymbol{x}_p$ which is cut by the interface, we know one of the coordinates of $\boldsymbol{x}_s$, which we insert into the interface equation to obtain the remaining coordinates. Thus, in the 2D example in Figure 5.9, we have $dx_p|n_x| \geq dy_p|n_y|$. Therefore, $x_s = x_f$, which we use in the line equation to obtain $y_s = (-d - n_x x_s)/n_y$. In 3D, if e.g. $z_d|n_z| = \max(x_d|n_x|, y_d|n_y|, z_d|n_z)$, we have $z_s = z_f$ and obtain the line equation

$$n_x x + n_y y = -d - n_z z_s = -d'.$$

Thereby, we reduce the three-dimensional problem to a 2D one and can go back to the previous steps to determine x_s and y_s correspondingly.

Note, that for the implementation of this algorithm special care has to be taken of the cases when the plane or line is parallel to one or two of the axes and the corresponding components of the unit normals become zero or close to zero.

How this algorithm works in 2D

In [WYKS09], Wang et al. do not give a proof for why their algorithm should work in two or even in three dimensions. In the following we will exemplify how the algorithm works in 2D and give a counterexample to prove why it fails in 3D. Although the 2D variant seems to work for lots of different interface configurations, we have not rigorously proved its integrity.

In order to see how this algorithm works in 2D, let us again look at the example drawn

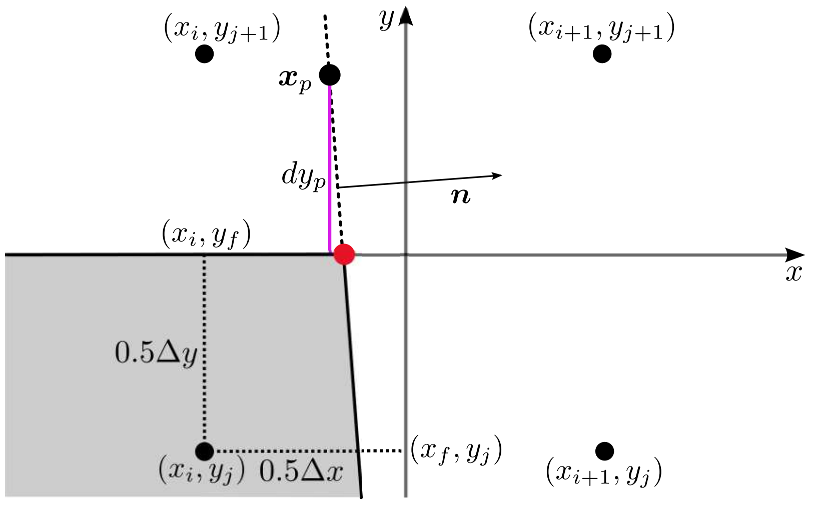


Figure 5.10.: Example 2 for the vertex finding algorithm by Wang et al. in 2D. The shortest distance from $\boldsymbol{x}' = (x_{i+1}, y_{j+1})$ to the interface (black line) in cell (i, j, k) is to be found. This point is marked with a red dot.

in Figure 5.9. We are interested in the point $\boldsymbol{x}_s$ on the interface in cell (i, j, k) with the shortest distance to $\boldsymbol{x}' = (x_{i+1}, y_{j+1})$. In this case, the algorithm should confirm that $dx_p|n_x| = \max(dx_p|n_x|, dy_p|n_y|)$, so that $x_f = x_{i+\frac{1}{2}}$ is found as the nearest face of cell (i, j, k) that is cut by the interface.

First, the angle α between $|n_x|$ and $|n_y|$ can be evaluated as

$$\tan(\alpha) = \frac{|n_y|}{|n_x|}.$$

Let a denote the distance from the intersection of the x_f-face with the interface to the y_f-face of cell (i, j, k). Then, the same angle α can be found between the lines dx_p and $dy_p + a$ with

$$\tan(\alpha) = \frac{dx_p}{dy_p + a}.$$

Thus,

$$\begin{aligned} \frac{|ny|}{|nx|} &= \frac{dx_p}{dy_p + a} \\ \Leftrightarrow \quad dx_p|n_x| &= (dy_p + a)|n_y| \\ \Rightarrow \quad dx_p|n_x| &\geq dy_p|n_y|, \end{aligned}$$

which shows that indeed $dx_p|n_x| = \max(dx_p|n_x|, dy_p|n_y|)$, and $\boldsymbol{x}_s = (x_f, -\frac{x_f n_x + d}{n_y})$ would be correctly found as the point on the interface nearest to (x_{i+1}, y_{j+1}).

Another example is given in Figure 5.10. Again, we are interested in the point $\boldsymbol{x}_s$ on the interface in cell (i, j, k) with the shortest distance to $\boldsymbol{x}' = (x_{i+1}, y_{j+1})$. Here, the $y_f = y_{j+\frac{1}{2}}$-face must be found as the nearest face of cell (i, j, k) that is cut by the interface.

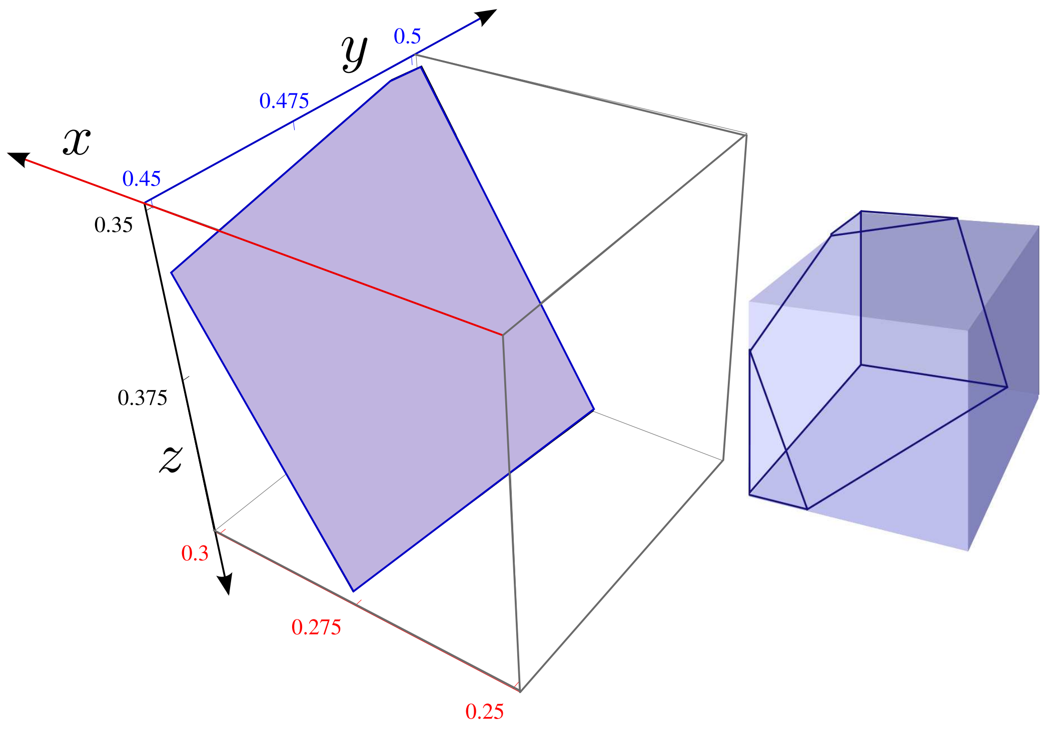


Figure 5.11.: The plane given by $n_x = -0.861697$, $n_y = -0.098878$, $n_z = -0.497696$ and $d = 0.481543$ in the cell (i, j, k) for which the algorithm by Wang et al. [WYKS09] fails. On the right, we show the plane's interface configuration type (cf. Fig. 5.5). We clearly see that the interface does not cut the $x = 0.25$-face as Wang's algorithm would predict.

In this case,

$$dx_p = \max(|x_p - x_i| - 0.5\Delta x, 0) = 0,$$

since $x_{i-\frac{1}{2}} \leq x_p \leq x_{i+\frac{1}{2}}$ for the x-component of the projection point, so that $dy_p|n_x| = \max(dx_p|n_x|, dy_p|n_y|)$. Therefore, the point $\boldsymbol{x}_s = (-\frac{y_f n_y + d}{n_x}, y_f)$ is correctly found by the algorithm.

How this algorithm fails in 3D

Unfortunately, this approach does not work in three dimensions. Let us consider the following counterexample as illustrated in Figure 5.11. We work on a computational grid where $\delta x = \delta y = \delta z = 0.05$. We have

- $x_i = 0.275$, $y_j = 0.475$, $z_k = 0.375$.
- $x_{i'} = 0.175$, $y_{j'} = 0.675$, $z_{k'} = 0.425$.
- The plane is defined by $n_x = -0.861697$, $n_y = -0.098878$, $n_z = -0.497696$ and $d = 0.481543$.
- The distance from $\boldsymbol{x}'$ to the plane is computed by $D(\boldsymbol{x}') = d + n_x x_{i'} + n_y y_{j'} + n_z z_{k'} =$

0.0524826. The projection point is then

$$\begin{aligned} x_p &= x_{i'} - n_x \operatorname{dist}(\boldsymbol{x'}) = 0.220224 \\ y_p &= y_{j'} - n_y \operatorname{dist}(\boldsymbol{x'}) = 0.680189 \\ z_p &= z_{k'} - n_z \operatorname{dist}(\boldsymbol{x'}) = 0.451120. \end{aligned}$$

- We compute the offset from the axes by

$$\begin{aligned} dx_p &= \max(|x_p - x_i| - 0.5\Delta x, 0) = 0.029776 \\ dy_p &= \max(|y_p - y_j| - 0.5\Delta y, 0) = 0.180189 \\ dz_p &= \max(|z_p - z_k| - 0.5\Delta z, 0) = 0.05112, \end{aligned}$$

  and

$$\begin{aligned} dx_p|n_x| &= \max(dx_p|n_x|, dy_p|n_y|, dz_p|n_z) \\ &= \max(0.0256579, 0.0178167, 0.0254422). \end{aligned}$$

- The three faces of cell (i, j, k) which are nearest to the projection point are given by

$$\begin{aligned} x_f &= x_i + \operatorname{sgn}(x_p - x_i) 0.5\Delta x_i = 0.25 \\ y_f &= y_j + \operatorname{sgn}(y_p - y_j) 0.5\Delta y_j = 0.5 \\ z_f &= z_k + \operatorname{sgn}(z_p - z_k) 0.5\Delta z_k = 0.4. \end{aligned}$$

In Figure 5.11 we see that the interface does not even cut the $x_f = 0.25$-face of cell (i, j, k). This contradicts Wang's algorithm which states that the $x_s = x_f = 0.25$-face must be cut by the interface since $dx_p|n_x| = \max(dx_p|n_x|, dy_p|n_y|, dz_p|n_z)$. Indeed, the correct point on the interface nearest to $\boldsymbol{x'}$ is $\boldsymbol{x}_s = (0.270426, 0.5, 0.4)$.

5.7. Contact angle boundary conditions in the CLSVOF method

Generally, the boundary conditions for the VOF function F can be derived from those of the LS function. Thus, at the end of the interface reconstruction explained in Subsection 5.3.3, if we do *do not* reconstruct the interface in a ghost cell (i, j, k), we set

$$F_{i,j,k} = \begin{cases} 1 & \text{if } \phi_{i,j,k} > 0 \\ 0 & \text{else.} \end{cases}$$

An interface is reconstructed in the ghost cell (i, j, k), if the interface is also reconstructed in the fluid cell next to it. Let us exemplify this at the boundary $y = 0$, and let cell $(i, 0, k)$ denote a ghost cell. Then, the interface is reconstructed in this cell, if $0 < F_{i,0,k} < 1$ and if additionally, $\phi_{i,0,k} \cdot \phi_{i',j',k'} < 0$ or $\phi_{i,0,k} = 0$ for some $|i - i'| \leq 1$, $|0 - j'| \leq 1$ and $|k - k'| \leq 1$ with $(i - i')(0 - j')(k - k') \leq 1$, which is the standard condition described in Subsection 5.3.2. Furthermore, we allow the interface to be reconstructed in cell $(i, 0, k)$, if $\phi_{i,1,k} \cdot \phi_{i',j',k'} < 0$ or $\phi_{i,1,k} = 0$ for some $|i - i'| \leq 1$, $|1 - j'| \leq 1$ and $|k - k'| \leq 1$ with $(i - i')(1 - j')(k - k') \leq 1$, i.e. if cell $(i, 1, k)$ also contains a reconstructed interface.

In the ghost cells, we do not have any meaningful values of the VOF function. There-

fore, in these cells, we do not recompute the position of the reconstructed interface, which is normally shifted along its unit normal, so that the area beneath the plane equals the given volume fraction; see Subsection 5.4.3. Instead, we use the reconstructed interface in the ghost cell to compute a meaningful value of F as described in the Remark in Subsection 5.5.1. Thereby the VOF function inherits its boundary values from the LS function, and we do not have to prescribe a direct dependency on the contact angle.

Since we are using an operator split transport scheme and a different reinitialization of the LS function, the contact angle boundary condition is included in the CLSVOF methods as follows:

- We set the contact angle boundary condition with $\theta = \theta^n$ for the LS function as described in Subsection 4.2.3 before each interface reconstruction step. This boundary condition is only set in the ε-environment containing the interface. Note that the remaining boundary cells also contain meaningful LS values due to our reinitialization, where even the LS values in the boundary are assigned to the exact signed distance of the cell centers to the reconstructed interface.
- After the transport in all three space directions, we compute the new $\theta = \theta^{n+1}$ by our contact angle models; confer Subsection 4.2.3. This new θ is used in the contact angle boundary condition for the LS function before the last interface reconstruction step and the for the subsequent reinitialization of the LS function.

Note that our implementation of the contact angle boundary condition in the CLSVOF method contains two potential problems. First, we simply decide to reconstruct the interface in a boundary cell, if its direct fluid neighbor also contains a reconstructed interface. Depending on the position of the interface, the interface of the fluid cell does not necessarily cut the underlying boundary cell. Second, we no longer have a pseudo time-stepping procedure in the reinitialization. Thus, if θ^{n+1} differs considerably from θ^n, we force both the VOF and the LS function to adapt quite suddenly to these changes. To our knowledge, it is unclear if a changing contact angle should influence the overall time step size to account for a stable numerical method. Such an analysis would go beyond the scope of this thesis. In all of our numerical experiments, the condition performed well, but some of the developing asymmetries and air inclusions observed in Subsection 6.4.3 might be attributed in part to this problem.

5.8. Parallelization of the CLSVOF method

In Section 4.6, we have already explained the parallelization strategy employed in our flow solver NaSt3DGPF. In this section, we focus on the parallelization of the CLSVOF method, which requires additional communication between processes on adjacent subdomains. To our knowledge, the parallelization of the reinitialization procedure (cf. Section 5.6) is described nowhere else in the literature.

Again, our Cartesian grid is decomposed into q overlapping subdomains which are in turn treated by one of the q processors. Therefore, we denote by Ω_q the subdomain on which each process works. Furthermore, we define the domain Ω_q^+ which equals the domain Ω_q plus an additional strip of ghost cells in each space direction. Then our CLSVOF algorithm, embedded in the Navier-Stokes solver NaSt3DGPF, reads as follows:

```
1. Initialize $\delta x$, $\delta y$, $\delta z$, $T$, $\alpha$, $\phi_0$ and $\Omega^h$.
2. Decompose $\Omega^h$ into local domains $\Omega_q$.
3. Set $t := 0$, $n := 0$, $h := \max(\delta x, \delta y, \delta z)$, and $\epsilon := \alpha h$.
4. Enforce $\phi_0$ to be a global signed-distance function: Solve (4.7) with $\tau = 1$. If possible,
   the LS function can also be initialized with an analytic solution.
5. INITIALIZATION
       Set initial values $\boldsymbol{u}^n := \boldsymbol{u}_0$, $p^n := p_0$, and $\phi^n := \phi_0$ on local domain $\Omega_q$.
       Exchange ghost cell values for $\phi^n$.
       Reconstruct the interface and compute $F^n := F_0$ from the reconstructed inter-
       face as explained in Subsection 5.5.1. In all other cells:
           $$F^n := \begin{cases} 1 & \text{if } \phi^n > 0 \\ 0 & \text{else.} \end{cases}$$
6. Set boundary values for $\boldsymbol{u}^n$ on the boundary $\partial\Omega_q \cap \partial\Omega^h$.
7. TIME-LOOP: While $t \leq T$:
   a) Compute time step $\delta t$ by (4.13).
   b) MOMENTUM EQUATION I:
          Exchange ghost cell values for $\boldsymbol{u}^n$ and $\phi^n$.
          Compute intermediate velocity field $\boldsymbol{u}^*(\boldsymbol{u}^n, p^n, \phi^n)$ on local domain $\Omega_q$ ac-
          cording to Section 4.1.
          Set boundary values for $\boldsymbol{u}^*$ on boundary $\partial\Omega_q \cap \partial\Omega^h$.
   c) PRESSURE POISSON EQUATION I:
          Exchange ghost cell values for $u^*$.
          Set right-hand side $\boldsymbol{b} := \nabla \cdot \boldsymbol{u}^*$ on local domain $\Omega_q$ for pressure Poisson
          equation.
   d) TRANSPORT EQUATION:
        i. Set contact angle boundary condition for $\phi$ within the $\epsilon$-strip of the interface
           on the boundary $\partial\Omega_q \cap \partial\Omega^h$.
       ii. Exchange ghost cell values for $\phi^n$ and $F^n$.
      iii. Use $F^n$ and $\phi^n$ to reconstruct the interface in $(i, j, k)$ for the computation
           of $(\delta V_F)^n_{i\pm\frac{1}{2},j,k}$ on $\Omega_q^+$.
       iv. Set boundary conditions for $F^n$ on $\Omega_q \cap \partial\Omega^h$ as explained in Section 5.7.
        v. Exchange ghost cell values for $\phi^n$ and $F^n$.
       vi. Solve equation (5.2) with $s^n = F^n$ and $s^n = \phi^n$ for $F^*$ and $\phi^*$.
      vii. Repeat all of these steps for the transport in the remaining space directions.
           Alternate the transport directions in every time step $\delta t$.
   e) Compute contact angle $\theta = \theta^{n+1}$ as explained in Section 4.5.
   f) Set contact angle boundary condition for $\phi$ within the $\epsilon$-strip of the interface on
      the boundary $\partial\Omega_q \cap \partial\Omega^h$.
   g) Exchange ghost cell values for $\phi^{n+1}$ and $F^{n+1}$.
   h) Truncate the volume fractions by eq. (5.28).
   i) Exchange ghost cell values for $\phi^{n+1}$ and $F^{n+1}$.
```

j) Reconstruct the interface from F^{n+1} and ϕ^{n+1} to reinitialize ϕ^{n+1} on Ω_q^+ and in all boundary cells of Ω^h as in 5.6.

k) Exchange ghost cell values for ϕ^{n+1}

l) PRESSURE POISSON EQUATION II:
Solve for $p^{n+1}\left(\frac{1}{\delta t}\boldsymbol{b}, \phi^{n+1}\right)$ by preconditioned Krylov-method.
Exchange ghost cell values for p_{it} per iteration it as required.

m) MOMENTUM EQUATION II:
Set $\boldsymbol{u}^{n+1}$ according to section 4.1 on local domain Ω_q.
Set boundary values for $\boldsymbol{u}^{n+1}$ on local boundary $\partial\Omega_q \cap \partial\Omega^h$.

n) Set $t = t + \delta t$ and $n = n + 1$.

Let us shortly focus on the transport part of our algorithm. In the transport steps (5.2), (5.3) and (5.4) we not only need boundary values for ϕ and F but also boundary values for the interface normals n_x^n, n_y^n n_z^n and the interface's distance to the origin d^n, which are all computed in the interface reconstruction process. These values do not need to be communicated since we do the interface reconstruction on the domain Ω_q^+, which includes one strip of ghost cells.

The most challenging part of the parallelization of the CLSVOF method is the reinitialization, in which we assign the LS function to the exact distance of the cell centers to the reconstructed interface. Thus, if we have found a cell with a reconstructed interface, we compute its distance from all cell centers $\boldsymbol{x}'_{i',j',k'}$ in a strip of K cells about the interface. Then, in each cell center the LS value is the minimum distance to all considered interface segments, i.e.

$$\phi_{i',j',k'} := \min_{i,j,k}(d(\boldsymbol{x}'_{i',j',k'}, \boldsymbol{x}^s_{i,j,k})), \tag{5.35}$$

for $(i - i') \geq K$, $(j - j') \geq K$ and $(k - k') \geq K$. Here, $d(\boldsymbol{x}'_{i',j',k'}, \boldsymbol{x}^s_{i,j,k})$ is the Euclidean distance of the cell center $\boldsymbol{x}'_{i',j',k'}$ to each interface segment, i.e. $\boldsymbol{x}^s_{i,j,k}$ is the point on the interface with shortest distance to $\boldsymbol{x}'_{i',j',k'}$.

There are two strategies for the parallelization of the reinitialization in the CLSVOF method. The first and seemingly more natural strategy is for each process to check only its own domain Ω_q for cells with an interface. Then, the distances from the cell centers of neighboring processes have to be computed and communicated. The second strategy is for each process to check its own domain and the K cells of neighboring processes for interface segments (requiring communication). Then, each process can compute the distances to these segments in its own domain Ω_q, for which no communication is necessary.

Let us exemplify in 2D how the first strategy works and why we chose the second: Suppose each process considers only cells with interface segments which lie in its local domain Ω_q. In the example drawn in Figure 5.12, q_2 does not know anything about the interface in cell (i, j) and can only compute the distance from $\boldsymbol{x}'_{i+1,j+1}$ to its own interface in $(i + 1, j)$. Then, q_1 would compute the distances from $\boldsymbol{x}'_{i+1,j+1}$, $\boldsymbol{x}'_{i+1,j}$ and $\boldsymbol{x}'_{i,j+1}$ to the interface segment in (i, j). Furthermore, each process computes the minimum distance

$$\phi_{i',j'} := \min_{i,j}(d(\boldsymbol{x}'_{i',j'}, \boldsymbol{x}^s_{i,j})), \tag{5.36}$$

for $(i - i') \geq K$, $(j - j') \geq K$ with the further restriction that the considered interface

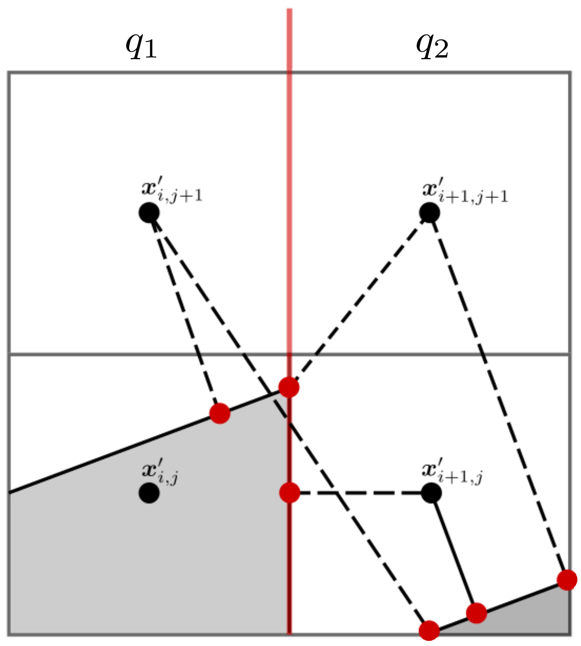

q_1
q_2
$\boldsymbol{x}'_{i,j+1}$
$\boldsymbol{x}'_{i+1,j+1}$
$\boldsymbol{x}'_{i,j}$
$\boldsymbol{x}'_{i+1,j}$

Figure 5.12: Example for the parallelization of the reinitialization within the CLSVOF method. The red line denotes the boundary between the processes q_1 and q_2. All cell centers are marked by black dots, and the distance from the cell centers to the interface segments are drawn by dashed black lines. Then, the red dot denotes the point with the shortest distance from the respective cell center to the interface segment.

segments lie in Ω_q only, i.e. $\boldsymbol{x}^s_{i,j,k} \in \Omega_q$. In order to compute the global minimum, communication between the processes is necessary. In our 2D example, communication between processes q_1 and q_2 would ensure that $\phi_{i+1,j+1} = d(\boldsymbol{x}'_{i+1,j+1}, \boldsymbol{x}^s_{i,j})$ (as computed by q_1) and not $\phi_{i+1,j+1} = d(\boldsymbol{x}'_{i+1,j+1}, \boldsymbol{x}^s_{i+1,j})$ (as computed by q_2). A further difficulty is presented by cell $(i+1, j)$ which contains an interface segment. In such a cell, the LS function can be directly evaluated by eq. (5.29). However, q_1 does not know so and unnecessarily computes distances from $\boldsymbol{x}'_{i+1,j}$ anyway. Or, q_2 would have to tell q_1 beforehand that cell $(i+1, j)$ contains an interface segment, which would require further communication.

In order to avoid these unnecessary computations or communications within the already complex reinitialization equation, we apply the second strategy, i.e. each process checks its own domain and the K cells of neighboring processes for interface segments and computes distances to these interfaces in its own domain Ω_q only. The first part of our strategy requires communication among the processes since for the K strips of boundary cells of Ω_q each process needs to know

1. if there is a cell containing an interface,
2. the three values of its normal and its distance to the origin,
3. the value of ϕ in each cell,
4. and the value of F in each cell.

This is a standard communication step, which is more costly due to the larger number of functions and strips of boundary cells that have to be considered. However, it has to be done only once for every time step and could be further reduced, if the position of the interface was taken into account. Furthermore, the minimum distance can be computed by

each process individually, which renders our strategy less complex.[4]

Let us further exemplify our approach by the 2D example in Figure 5.12. Here, q_1 would compute the minimum of the distances from $\boldsymbol{x}'_{i,j+1}$ to both interface segments in cell (i,j) and $(i+1,j)$, while q_2 could do the same in cell $(i+1,j+1)$. Furthermore, both q_1 and q_2 know about the interface segments in cells (i,j) and $(i+1,j)$, so that no unnecessary computations are performed. Thus, each process performs the minimum computation (5.36) with the restriction that $\boldsymbol{x}'_{i',j'} \in \Omega_q$. In contrast to the first approach, each process can do this computation individually and no further communication is necessary.

The details of our implementation are as follows:

1. Each process checks its own domain for interface segments as well as the K strips of boundary cells of Ω_q. If the process reaches the boundary of the domain Ω^h, we only check its first layer of boundary cells, not all three rows of ghost cells.
2. If a cell with a reconstructed interface is found, each process only computes the minimal distance to the interface in its own domain Ω_q and in the ghost cells rows of Ω^h.

Let

$$\Omega^h := [i_{\min}, i_{\max}] \times [j_{\min}, j_{\max}] \times [k_{\min}, k_{\max}] \text{ and } \Omega_q := [i_l, i_u] \times [j_l, j_u] \times [k_l, k_u]$$

without ghost cells. Then, we define the lower and upper boundaries in which each process computes distances by

$$p_l := \begin{cases} i_{\min} - 3 & \text{if } i - K < i_{\min} - 1, \text{ i.e. we are below the lower boundary of } \Omega^h \\ i_l & \text{if } i - K < i_l, \text{ i.e. we are at the lower boundary of } \Omega_q \\ i - K & \text{else,} \end{cases} \tag{5.37}$$

and

$$p_u := \begin{cases} i_{\max} + 3 & \text{if } i + K > i_{\max} + 1, \text{ i.e. we are above the upper boundary of } \Omega^h \\ i_u & \text{if } i + K > i_u, \text{ i.e. we are at the upper boundary of } \Omega_q \\ i + K & \text{else,} \end{cases} \tag{5.38}$$

and we define q_u, q_o, r_u and r_o for the indices j and k analogously. Note that the '-3' or '+3' indicates that we have three rows of ghost cells for the boundary of Ω^h. If more or less rows are needed, this number has to be adapted accordingly.

The nesting of loops for the parallelization of the reinitialization equation is then given in Algorithm 2. Thus, our first three loops make sure that each processor checks its own domain for cells which contain an interface as well as the K boundary cells. Then, the 'if statement' lets each process consider one strip of boundary cells of Ω^h only. If a cell with a reconstructed interface is found, each process looks at the K neighboring cells and computes

[4]Probably, if parallelized effectively, the first strategy might render at most as costly as the second strategy. However, the reinitialization is very complex on its own, and we strongly recommend to keep the parallelization easy and consider its improvement only when all other parts of the code have been well-tested.

Algorithm 2: Nesting of loops for the parallelization of the reinitialization equation.

```
for(i=il-K; i<=iu+K; i++)
for(j=jl-K; j<=ju+K; j++)
for(k=kl-K; k<=ku+K; k++){
  if(i>=0 && i<=imax+1 && j>=0 && j<=jmax+1 && k>=0 && k<=kmax+1)
  if(IsCellWithReconstructedInterface(i, j, k)){
      for(p=pl; p<=pu; p++)
      for(q=ql; q<=qu; q++)
      for(r=rl; r<=ru; r++)
      if(!IsCellWithReconstructedInterface(p, q, r)){
      \\find minimal distance of point in cell p,q,r
      \\to interface in i,j,k
      }
  }
}
```

their distance to the interface cell. However, the next three loops and the conditions for their lower (5.37) and upper bound (5.35) make sure that each process only computes distances its own domain Ω_q.

With the parallelization of the CLSVOF method, we conclude this chapter. By now, our mathematical model is defined and discretized. We proposed three methods with the potential power to conserve the mass normally lost with the LS method, and we are able to compute meaningful dynamic contact angle boundary conditions at contact lines. Thus, we are well-equipped to proceed with a range of numerical experiments in the subsequent chapter, whose aim will be twofold: On the one hand, we need to thoroughly validate our two volume correction methods and the CLSVOF approach. On the other hand, we want to apply and validate our contact angle models for the example of droplets impacting on substrates with different wetting characteristics.

6. Numerical Results

In this chapter, we present our results for the numerical simulation of droplets impacting on substrates with different wetting characteristics. We show that both the implemented reduced interface formation model by Shikhmurzaev [Shi07] as well as Yokoi's approach [YVHH09] are able to produce droplet shapes which are extremely close to results from practical experiments. However, Yokoi's approach requires the knowledge of advancing and receding contact angles, which is not always available beforehand. Furthermore, this approach is very sensitive to its material-related parameters as described in [YVHH09]. In contrast, the asymptotic interface formation model is generally applicable, estimates of its empirical parameters can be found in the literature [BS02] and its results do not change drastically with slight changes of these parameters [Shi07].

Before we study droplet impact with the CLSVOF method or any of the volume corrections methods, we have to ensure their numerical stability. Therefore, we first validate our implemented CLSVOF method, LS method, level-set method with local volume correction and level-set method with global volume correction by a range of numerical benchmarks.

The structure of this chapter is as follows. First, we are concerned with two test cases without wetting in which we prescribe the velocity field. Thereby, a direct comparison between the interface capturing and volume correction methods is possible – without involvement of the flow solver.

Second, we investigate rising bubbles in two and three dimensions. The 2D case is tested with various flow solvers in [HTK$^+$09] which allows for a comparison with our flow solver NaSt3DGPF. In [AEG$^+$14], we published the results for the 3D case with the flow solvers NaSt3DGPF [CGS10], DROPS [DRO08, RFG$^+$] and OpenFOAM [Ope13]. In that work, however, we employ the level-set method with local volume correction only – not the CLSVOF method. Since the results computed with the different flow solvers are not entirely consistent in [AEG$^+$14], we are very much interested in how the CLSVOF method fares with these problems. Furthermore, in what follows, the level-set method with local volume correction will prove to be unreliable, displaying the unphysical oscillations already described in the literature [Her05, Kec98]. Therefore, the CLSVOF method presents an invaluable and computationally very fast alternative.

Third, we evaluate our mathematical and numerical models for the simulation of dynamic wetting. Here, we are first interested in a purely qualitative comparison of our numerical simulation and physical experiments to show that our numerical code is able to cope with droplet impact dynamics, that our implementation of the contact angle is valid for a wide range of applied angles and that with these prerequisites and without any sophisticated contact angle models, we are already able to predict droplet impact behavior which is very close to what is observed in experiments. Next, we simulate the impact of water on silicon rubber, which is the specific experiment Yokoi's approach has been designed for. Additionally, we employ the asymptotic interface formation model by Shikhmurzaev,

which we compare with Yokoi's approach and the practical experiment. Furthermore, we simulate the micron-scale impact of water droplets with two different impact speeds on two different substrates. Here, we employ the asymptotic interface formation model only since Yokoi's approach requires knowledge of the dynamic advancing and receding angles as well as adaption of the involved empirical parameters which we cannot provide. The droplets' heights and diameters are compared to practical experiments by Dong et al. [DCBM07] and to simulation results with a constant contact angle. Last, we simulate the micron-scale drop impact of water compared to the millimeter-scale drop impact of a water-glycerin mixture.

Before we present our numerical results, we wish to discuss some preliminaries which hold for all subsequent experiments: We address the numerical methods employed in NaSt3DGPF, our cluster architecture for parallelization and our tools for the post-processing of data.

6.1. Preliminaries

In this chapter, if not otherwise noted, all simulations are conducted with our flow solver NaSt3DGPF which employs the following features: For temporal discretization, Chorin's projection method is used to decouple the fluid velocities and the pressure field. The velocities are treated explicitly with a 2nd-order Adams-Bashforth scheme. The pressure field is computed by solving a Poisson-type equation in each time step. For the iterative solution of the pressure Poisson problem, we use a Jacobi preconditioned conjugate-gradient method with an absolute residual tolerance of 10^{-15}. The convective terms in the momentum equations are treated with a 5th-order WENO scheme.

Our parallel simulations were computed on three different clusters of the Institute for Numerical Simulation [INS13], Bonn University. These systems have special high-performance networking hardware, allowing fast communication between the nodes separated from the service ethernet and without the TCP/IP protocol overhead. The older *Himalaya* has 256 Intel Xeon EM 64 T 3.2 GHz CPUs, a main memory of 640 GB and performs 1269 GFlop/s in the Linpack benchmark test. In addition, *Siebengebirge* features 160 Intel Xeon X7560 2.226 GHz CPU cores and a main memory of 2560 GB in total. This system has a Linpack performance of 1349 GFlops/s with a parallel efficiency of 93%. The recent *Atacama* cluster has 78 Dell PowerEdge M620 nodes and 1264 cores and a Linpack performance of 20630 GFlop/s, while maintaining a total memory of 4992 GB.

Note that all surface and volume integrals for the analysis of results are computed by the ParaView [Par11] or VisIt software [CBW$^+$12] in a post-processing step. Both software tools are able to reconstruct the iso 0 level-set surface. Then, an integration over the interface Γ_f can be done without the use of the δ-function, which always comes with a thickness of several cells across the interface and introduces a constant error in the integration. Here, ParaView and VisIt rely on the Marching Cubes algorithm [LC87] for isosurface extraction. Basically, both software tools are able to read the Visualization Tool Kit (VTK) format of data files and compute identical results for the test cases considered in this thesis.

We use the ParaView versions from 3.14.1 onwards and the two VisIt software versions 2.4.1 and 2.8. We do not use the newest version 4.1.0 of ParaView, which tended to crash when drawing the zero level-set. In sequential computing, the VisIt software is able to handle larger amounts of data than ParaView which is why both software tools were used in this thesis.

Let us shortly explain how the ParaView and the VisIt software can be used for the computation of surface and volume integrals. For both tools, we provide the level-set data in a VTK format. In ParaView, we apply the 'contour' filter with the value 0 for the level-set function. The surface integral can then be computed by applying the 'IntegrateVariables' filter to the contour. If we choose 'cell data' in the according dialog, the result appears. If the volume integral needs to be computed, the filter 'IsoVolume' is applied to the original data set (not the contour). Then, we can choose the lower and upper threshold of level-set values, e.g., all values less than zero if the volume of a gas bubble is computed. Next, we apply the 'IntegrateVariables' filter, choose 'cell data' and obtain the required result. Note that these ParaView classes can be wrapped into a python callable format, which enables us to use scripts for the computation of the required quantities over time.

Within the VisIt software, we add the zero contour to the plot dialog. Then, we choose '2D area' from the menu 'Controls' $\rightarrow$ 'Query', which computes the desired surface integral. For the computation of the volume integral, add 'volume' to the plot dialog. Then, 'OpAtts' $\rightarrow$ 'Slices' $\rightarrow$ 'Isosurfaces' yields a menu, where you choose the level-set data as 'Variable' and the 'min' and 'max' values according to the problem. After that, we use the 'Query' dialog to compute the required 'Volume'.

For the volume computation of spherical shapes such as the droplets considered in this thesis, the combination of NaSt3DGPF and ParaView is at best second order accurate: If we initialize a sphere with the level-set function in NaSt3DGPF on successively refined grids, write VTK structured data files and compute the volume in ParaView in comparison with the analytical volume, this computation is second-order accurate.

Additionally, for visualization purposes only, we use the Tecplot 360 software (Release 1, 2013) [Tec13]. This software also uses as Marching Cubes-style algorithm for the extraction of the isosurface.

In what follows, we use the following abbreviations:

- CLSVOF: Coupled level-set and volume-of-fluid.
- LS: Level-set: the level-set method without any volume correction method.
- LSG: Level-set with global volume correction: the level-set method augmented with the Global Picard iteration for volume correction as described in Section 4.4.
- LSL: Level-set with local volume correction: the level-set method where an additional constraint is introduced in the Hamilton-Jacobi equation (4.18) which conserves the volume in every subdomain of Ω and prevents the straying of the level-set function from its initial position.

Note that a list of all abbreviations can also be found in the Appendix A.

6.2. Advection tests

In this section, we introduce two test cases in which we prescribe the velocity field. First, we solve the stretching of a circle in a shear velocity field as proposed by Rider and

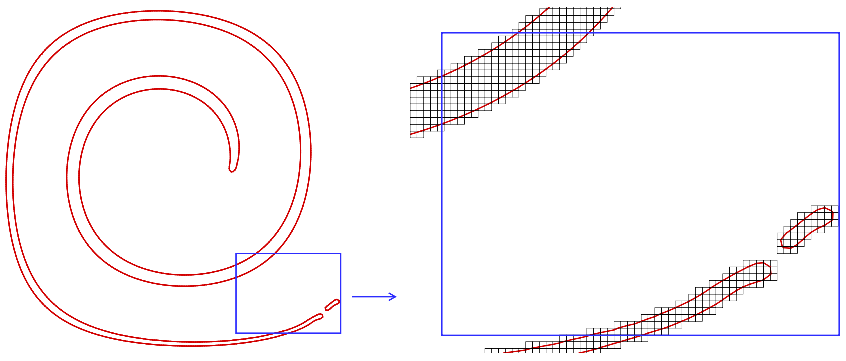


Figure 6.1.: Circle at maximum stretching. On the left, the contour computed by the CLSVOF method is shown. The blue rectangle is magnified on the right, where grid cells are drawn. The very thin filaments are resolved by 2–3 grid cells only.

Kothe [RK95], which is a two-dimensional example. Our second advection case is the fully three-dimensional deformation of a sphere. Both of these test cases allow for a direct comparison between the different interface capturing methods, without involvement of the flow solver and without the costly solution of the Poisson equation.

Note that in both of these test cases a particle level-set method performs superior to the CLSVOF method [WYS08, EFFM02]. In these special cases,the initially prescribed particles can return to their initial location after the flow reverses. If the particles in the reseeding process do not deviate much from the initial ones and if the reseeding operation is done scarcely or not at all, the initial shape of the circle (in 2D) or the sphere (in 3D) can be recovered almost exactly. Thus, one of the main sources of error for the particle level-set method is the new addition or deletion of particles during the reseeding process. If no reseeding is necessary, all particles return almost exactly to their initial position [WYS09].

6.2.1. Two-dimensional stretching of a circle in a shear velocity field

Our first test case is the two-dimensional stretching of a circle in a shear velocity field as proposed by Rider and Kothe [RK95]. This advection test is a realistic problem since interfaces undergo strong topological changes, including merging and fragmentation. Our here presented test takes such shear effects into account and is further challenging due to severe topological changes with very thin filaments on the scale of the mesh size (Fig. 6.1). The LS, the CLSVOF method and both volume correction methods are evaluated by their ability to cope with these changes and their ability to preserve the initial mass of the sphere. In addition, we are interested in the computational efficiency of all four methods in order to find an adequate trade-off between the most accurate and computationally most efficient method.

Setup of the numerical experiment

We choose a flow domain Ω with size $\Omega = [0, 1.0] \times [0, 1.0] \times [0, 0.5]$ which is quasi-periodic with respect to the z-axis. A cylinder with radius 0.15 is centered at at $\bar{\boldsymbol{x}}(t = 0) = (0.5, 0.75, 0.25)^{\mathsf{T}}$. In the following, we denote the area occupied by the cylinder by $\Omega_1 =$

u:	$-\sin^2(\pi x)\sin(2\pi y)\cos\left(\frac{\pi t}{T}\right)$
v:	$\sin^2(\pi y)\sin(2\pi x)\cos\left(\frac{\pi t}{T}\right)$
w:	0
final time:	$T = 8$
flow domain:	$\Omega = [0, 1.0] \times [0, 1.0] \times [0, 0.5]$
circle radius:	0.15
circle center:	$(0.5, 0.75, 0.25)$
interface thickness:	$\epsilon = 1.75h$
grid resolution:	$128 \times 128 \times 5$ and $256 \times 256 \times 10$
employed methods:	LS, LSL, LSG, CLSVOF
quasi 2D:	periodic boundary conditions in z-direction
boundary conditions:	no-slip except for z-direction

Table 6.1.: Simulation parameters for the two-dimensional stretching of a circle in a shear velocity field.

$\Omega_1(t) \subset \Omega$. Due to the quasi-periodic setup, we then have a circle on each slice in the periodic z-direction. Hence, in the following, we always use the term 'circle' – although we are talking about a cylinder from the three-dimensional point of view.

The prescribed shearing field is given by

$$\begin{aligned} u &= -\sin^2(\pi x)\sin(2\pi y)\cos\left(\frac{\pi t}{T}\right) \\ v &= \sin^2(\pi y)\sin(2\pi x)\cos\left(\frac{\pi t}{T}\right) \\ w &= 0 \end{aligned} \tag{6.1}$$

with time t and $T = 8$ the time when the flow returns to its initial position. Due to the shearing field, the circle deforms to a vortex with maximum stretching at $t = 4$. Then the velocity field is inverted and the vortex is compressed back to its initial position and circular shape. Note that a multiplication by $\cos(\pi t/T)$ causes any incompressible flow field to return to its initial state at $t = T$ [LeV96, RK95], a feature employed in this section as well as for the three-dimensional deformation field in the next section. Then, at $t = 0$ and $t = T$ the differences in data can be used for error measurements. All numerically and physically relevant parameters are summarized in Table 6.1.

We compute the solution on two grids with 128×128 and 256×256 grid cells in the x- and y-direction. Since we are computing a quasi-2D solution with our three-dimensional Navier-Stokes solver, we resolve the domain of length 0.5 in z-direction by 5 or 10 grid cells for the lower and higher resolution, respectively.[1] We then have periodic boundary conditions in the z-direction and no-slip boundary conditions hold at all remaining boundary faces. On both grids the time step is restricted by

$$\delta t = \text{cfl} \cdot \frac{h_{\min}}{\max_{(i,j,k)}\left(u_{i,j,k}, v_{i,j,k}, w_{i,j,k}\right)} \tag{6.2}$$

[1]Normally, we would not apply two different resolutions in the periodic direction. However, the CLSVOF code cannot yet deal with too anisotropic grids which is due to the implementation – not the method itself.

with **CFL!** number cfl $= 0.25$ and $h_{\min}$ the smallest mesh size.

Our simulations are conducted in parallel on one 16-core node of the *Atacama* cluster. Since the velocity field is prescribed, we do not solve the full Navier-Stokes equations. We only have to concern ourselves with the solution of the transport equation and with the reinitialization of the LS function.

We compute results on both grids with all four possible methods, i.e. the LS, the LSL, the LSG and the CLSVOF method. Additionally, we measure the computing time of all methods.

Expected behavior

We already indicated that the two-dimensional stretching of a circle in a shear velocity field is a very challenging test case concerning the conservation of mass, which we hope to improve with the CLSVOF and both volume correction methods. Since very thin filaments on the scale of the mesh size evolve (Fig. 6.1), all methods have to demonstrate their ability to conserve mass despite these topological changes. Furthermore, they have to show their ability to return the circle to its initial position.

Certainly, for this test, we expect better mass conservation with the CLSVOF method compared to our LS method at least on coarse grids. If we choose a fine enough grid, both the LS method and the CLSVOF method should be able to conserve a satisfactory amount of mass and to recover the initial circular configuration.

The application of the LSG method might not yield the best results in this case. With this volume correction, the LS function is blown up everywhere to maintain the mass of the initial circular configuration. However, this 'blowing up' mechanism entails two problems. First, the loss of mass does not occur uniformly in space and time. More mass is lost, where thin filaments or droplets on the scale of the mesh size evolve. These might then still be lost with the LSG method, while an artificially large amount of mass is added elsewhere.

The second problem of the LSG is only due to our implementation of the method. Thus, we approximate the mass of the circle by a simple summation rule, i.e.

$$|\Omega_1| \approx \sum_{i,j,k=1}^{n} H_\varepsilon(\phi)\, dx_i\, dy_j\, dz_k \tag{6.3}$$

with ε the interface thickness. Since the Heaviside function is smeared out over several grid cells across the interface, we introduce a constant error in the computation of the mass. This error depends heavily on the shape of the zero level-set, i.e. on the number of cells which contain an interface. For example, at the beginning of this test case, the LS function has the shape of a circle. Then, at maximum stretching, severe topological changes with very thin filaments on the scale of the mesh size evolve which include a much larger number of interface cells. With equation (6.3), we then automatically compute a larger amount of mass than for the initial circle, only because we have a larger amount of interface cells. Thus, the Picard iteration (4.14) would assume that no mass needs to be corrected, although a vast amount already might have been lost.

Note again, that this second drawback concerns our implementation of the method and is not a drawback of the method itself. In order to remedy this problem, we would need a better knowledge of the interface position. This could be achieved by adaptive refinement or the reconstruction of the interface by, e.g., Marching Cubes [LC87]. However, even if we could compute the mass of $|\Omega_1|$ exactly, the first problem concerning the LSG would still remain: Mass loss does not occur uniformly in space as the LSG inherently assumes. Therefore, instead of refining this simple but rather unphysical method, we opted for the implementation of the CLSVOF method.

In contrast, the LSL method should perform well for this problem. With this method the mass should remain unchanged in any sub-domain of Ω, so that $\int H(\phi^*)$ is preserved in every grid cell. Therefore, we expect that the LSL method is well-equipped to deal with the evolving thin filaments of our two-dimensional test case – minus the mass that is lost due to numerical diffusion when solving the transport equation.

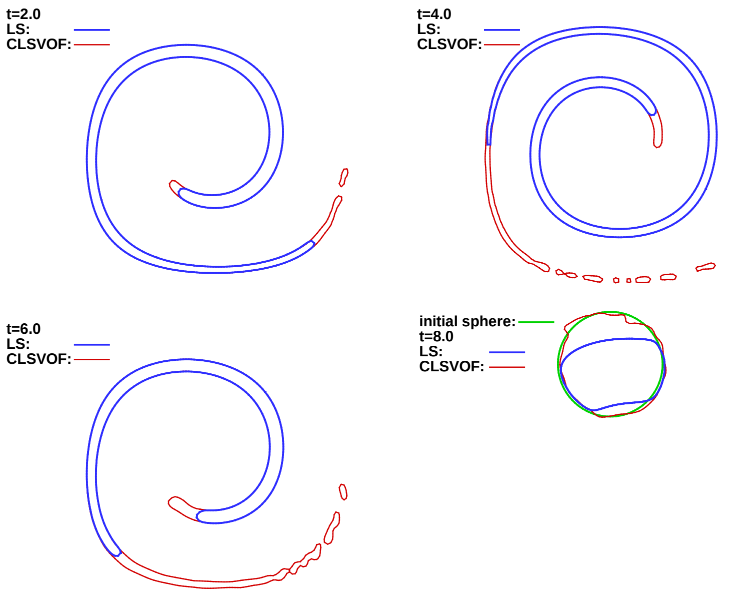
Concerning the computing time, we expect both the CLSVOF and the LS method to behave similarly. The bottleneck for the LS method is the reinitialization of the LS function, which takes up most computing time. In the CLSVOF method, the LS function ϕ^{n+1} needs to be exact within a narrow band of K cells about the interface, where we usually set $K = 4$. With N the number of cells which contain an interface reconstruction, the speed of the algorithm is $\mathcal{O}(K^3N)$ [SP00], which is also that of the reinitialization of the LS method in a tube of K cells about the zero level-set. In addition to the LS method, we have to transport the VOF function and compute the reconstruction of the interface for the CLSVOF method. Instead for the LS method, we employ several costly features in the reinitialization equation like a high-order WENO method in space and a Runge-Kutta scheme for time discretization. All in all, we therefore expect a similar computational efficiency of both methods.

The computing time of the LSG method should also be very close to that of the LS method. Here, we only have to solve the additional Picard iteration (6.3), which does not substantially contribute to the overall run-time. However, on very coarse grids, due to the volume correction, we have a larger number of interface cells compared to the LS method. Then, these interface cells would add to the costs of the reinitialization equation of $\mathcal{O}(K^3N)$ with N the number of interface cells. However, this effect should become negligible on finer grids.

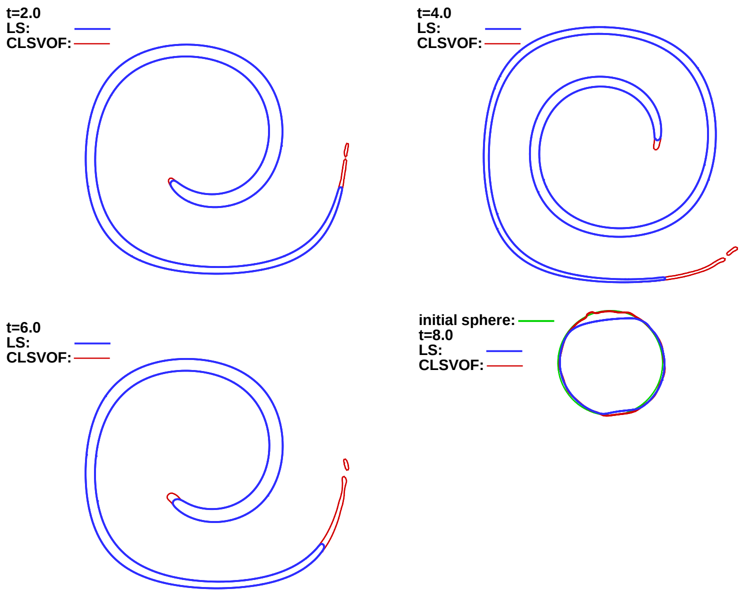
The LSL method performs the same reinitialization as the LS method with an added constraint (4.17) for every interface cell. Note that in the original article [SF99], the complexity of the LSL method is not quantified. Only in the presented numerical results, the time-step τ of the reinitialization equation (4.18) with the constraint (i.e. with the local volume correction) is chosen 5 times that of the LS method since 'the constraint allows us to iterate with a larger time-step and with less error' [SF99, p.1182]. Then, with τ chosen this large, the LS method still outperforms the LSL method with respect to computing time. For the moment, we do not adapt the time-step of the reinitialization in the LSL method. This would require further analysis of the method, which goes beyond the scope of this thesis. All in all, the LSL method will therefore be considerably slower than the LS method.

Discussion of results

The development of the circle in the shear flow over time is shown in Figures 6.2 and 6.3. In Figure 6.2, we compare the results of the CLSVOF method with the LS method without any volume correction, while we focus on the results of the LS method compared to the LSL and the LSG method in Figure 6.3. For all involved methods we draw the zero level-set

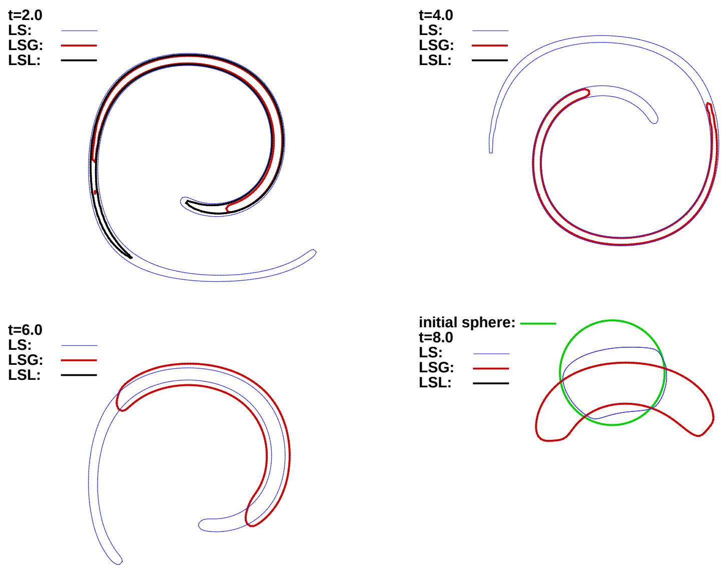


(a) $h = 1/128$

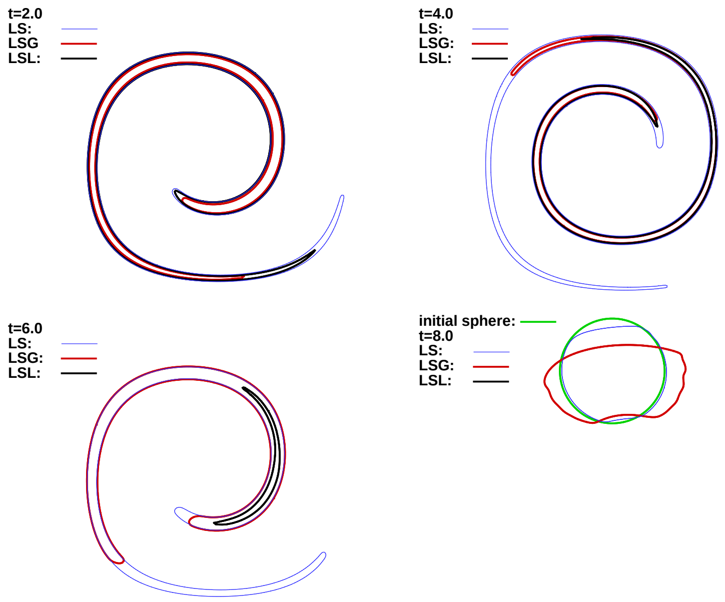


(b) $h = 1/256$

Figure 6.2.: Two-dimensional stretching of a circle in a shear velocity field for two different grid resolutions on the slice $z = 0.25$. The lines mark the zero level-set and are blue for the LS method and red for the CLSVOF method. Quite noticeably, the LS solution always remains in the bounds of the CLSVOF solution. For comparison, we draw the initial circle (green) in the $t = 8$ frame for both grid resolutions.



(a) $h = 1/128$. Note that the black interface tracked by the LSL method has already vanished at $t = 4$.



(b) $h = 1/256$ Note that the black interface tracked by the LSL method has already vanished at $t = 8$.

Figure 6.3.: Two-dimensional stretching of a circle in a shear velocity field for two different grid resolutions on the slice $z = 0.25$. The lines mark the zero level-set and are blue for the LS, red for the LSG and black for the LSL method. For comparison, we draw the initial circle (green) in the $t = 8$ frame for both grid resolutions.

contour extracted with the Tecplot software on the $z = 0.25$-slice at times $t = 2$, 4, 6 and 8. The top four figures always refer to the coarse grid solution ($h = 1/128$), while the four figures on the bottom refer to the fine grid solution ($h = 1/256$). At $t = 8$, we also plot the initial circle in green, which should be recovered by the respective numerical methods.

First, we focus on the comparison of the CLSVOF and the LS method in Figure 6.2. Here, the CLSVOF interface is given by the red line, while the LS solution is marked by the blue line. As expected, the CLSVOF method is much better at mass conservation than the LS method. Note that the interface of the LS method is up to a certain point in time always almost equal to that recovered by the CLSVOF method. However, the thin filaments which develop at the front or back of the stretched circle are simply lost with the LS method.

The mass loss is most notable on the coarse grid with $h = 1/128$ (Fig. 6.2(a)). At maximum stretching ($t = 4$), there is a vast difference between the length of the vortex with the LS and the CLSVOF method. Several satellite droplets evolve at the rear end of the vortex with the CLSVOF method, which tries to recover the thin filaments. After the flow reverses, these droplets become an irregular structure, which can be seen at $t = 6$. Then, the initial circle, which is drawn as a green line in the last picture, cannot be recovered exactly, but the CLSVOF method is still very close. In contrast, due to the vast amount of mass lost with the LS method, the LS solution is nowhere near spherical in the last frame of Figure 6.2(a).

As expected, the differences between the CLSVOF and the LS method are less prominent on the finer grid (Fig. 6.2(b)). Note that on this fine mesh, the filament at the rear end of the vortex is resolved by only 2–3 grid cells (cf. Figure 6.1). The LS solution loses much less mass than on the coarse grid but still performs less well than the CLSVOF method. Additionally, only one satellite drop evolves on the finer grid at $t = 4$, so that the CLSVOF method recovers the circle even more accurately than on the coarse grid. Again, the final shape produced by the LS method is less circular due to the mass loss over time.

Second, we compare the results of the LS with the results of the LSG and the LSL method in Figure 6.3. The LS interface is still marked by the blue line, while the interface is drawn in black for the LSL and in red for the LSG method.

First, we notice that the LS method performs much better for this difficult test case than both volume correction methods. At all times and for both grid resolution, thin filaments are always retained more accurately with the LS method, while both volume correction methods are unable to resolve these very fine structures at the front and, most noticeably, at the rear end of the vortex.

As expected, the LSG method is ill-suited to handle the vast topological changes presented by this test case. This is best seen on the coarse grid in Figure 6.3(a). Due to our implementation and the larger number of interface cells when the circle is stretched compared to its initial shape, the LSG method is unable to retain the initial mass of the droplet. Furthermore, at time $t = 6$, the vortex becomes unnaturally blown up instead of retaining its fine structure, which leads to a very deformed bean shape at the end of the computation.

We already indicated that the Picard iteration, which employs equation (6.3) for the mass computation should be more accurate on the finer grid. This is what is observed in Figure 6.3(b) for a mesh size of $h = 1/256$. Here, much more mass can be retained by the LSG method than on the coarser grid. However, as we observe at $t = 6$, the middle part of

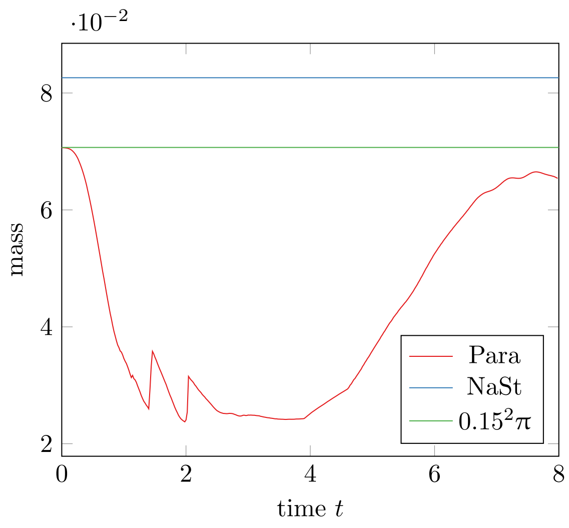


Figure 6.4.: Mass loss over time for the two-dimensional stretching of a circle in a shear velocity field with the LSG method. Computed is the 2D-area of the vortex with the summation rule (6.3) (blue) implemented in NaSt3DGPF and by post-processing with the ParaView software [Par11] (red). The approximate analytical solution is drawn in blue.

the vortex gets blown up a little compared to the LS result, while this structure is lost at the front and rear end of the vortex. Therefore, in the final frame, we again have a very deformed shape instead of a circle.

Let us quantify the difficulty of the LSG method by measuring the mass of the circle over time on the grid with mesh size 1/128. On the one hand, we measure the mass during runtime after each reinitialization and volume correction step by the simple summation rule (6.3). On the other hand, in a post-processing step, we employ the ParaView [Par11] software for the computation of mass for the 200 files that are written by our program. Both results are plotted over time in Figure 6.4. At $t = 0$ we know the initial area/mass of the circle to be πr^2, i.e. $\pi \cdot 0.15^2 \approx 7.07 \cdot 10^{-2}$, which is very well approximated by the ParaView software. However, computed on our rectangular grid with formula (6.3), the mass of the circle is overestimated drastically. Then, the next problem of the LSG method sets in: Seemingly, the mass is conserved perfectly over time. However, this behavior is not reflected by the computation of mass with the ParaView software. Here, we see that a lot of mass is lost during the stretching of the circle until the maximum stretching at about $t = 4$ is achieved. Then, the previously lost mass is regained by the LSG method.

Note that this is the exact behavior that we expected of the LSG method. Due to the larger number of interface cells and our very simple and inaccurate integration scheme, the LSG assumes that the mass loss is not very severe. Thus, the larger the number of interface cells, the less mass is corrected. The largest number of interface cells is achieved at $t = 4$ with the maximum stretching of the circle. After that, the number of interface cells reduces, which leads to more corrected mass until nearly all mass is recovered at $t = 8$.

In addition, two small peaks are computed by the ParaView software at about $t = 2$. We cannot fully explain this sudden onset of the LSG method. Before the first peak, the

	LS	LSL	LSG	CLSVOF
1/128	0 h 33 min	02 h 57 min	0 h 33 min	0 h 19 min
1/256	06 h 10 min	43 h 59 min	06 h 12 min	02 h 56 min

Table 6.2.: Computing time for the two-dimensional stretching of a circle in a shear velocity field on one 16-core node of the *Atacama* cluster.

mass loss is very drastic. This loss of mass might lead to a quite sudden reduction of the number of interface cells, which in turn might tell the LSG method to retain more mass. Such a behavior could also explain the second peak. However, in order to fully explain these peaks, we need computations on finer grids or at least a rough count of the number of interface cells over time.

Even if we fixed the LSG by a more accurate approximation of the volume by using a Marching Cubes algorithm in NaSt3DGPF directly, this method remains problematic. The Picard iteration (4.14) inherently assumes that mass loss occurs uniformly in space, which is not easily remedied.

Actually, we hoped for a much better performance of the LSL method. However, for this special test case, this method performs worst of all. On the coarse grid, no mass of the vortex is left at $t = 4$ and on the finer grid, no mass is left at the final stage of $t = 8$. Note that at $t = 2$ the LSL method retains more of the vortex than the LSG method. However, as the vortex becomes more and more stretched and two interfaces come close to each other, the LSL method fails. Here, we assume that the LSL method cannot correctly evaluate the integrals over the derivative of the Heaviside function given in equation (4.21). As soon as two interfaces are only a few grid cells apart, both wrongly affect the integral in this equation. Most probably this is when the LSL method can no longer retain the mass in each subdomain of Ω. However, further investigations are necessary to confirm this assumption.

In a last step, we measure the computing time of the four methods for both grid resolutions. All simulations were conducted in parallel on one 16-core node of the *Atacama* cluster. The resulting computing time is shown in Table 6.2. Indeed, the LSG and the LS method take about the same computing time on both grids. Furthermore, we expected the LSL to be more expensive than the LS method, which is exactly what we see in Table 6.2. In particular, the LSL method is 6 times slower on the grid with mesh size 1/128 and 7 times slower on the fine grid. Even a five times larger time-step in the reinitialization equation would not be able to compensate for this large difference. In contrast, we are impressed by the performance of the CLSVOF method, which is twice as fast as the LS method on the finest grid. This might be due to the very expensive components of the LS reinitialization scheme such as the fifth order WENO scheme in space and the third-order Runge-Kutta scheme in time, which yield an accurate but computationally expensive solution.

Next, we present some exemplary results from the literature, where the test case of the ‘circle in shear flow’ has already been solved with the CLSVOF method.

Comparison to results from the literature

Our test case has already been studied with the CLSVOF method by Ménard, Tanguy and Berlemont in [MTB07] and by Wang, Yang and Stern in [WYS08] (cf. also [WYKS09]).

In [MTB07] the CLSVOF method is specifically adapted to handle very thin filaments for the simulation of the primary break-up of liquid jets. This is achieved by a modification of the interface reconstruction stencil (cf. Section 5.3) when there is more than one interface front in the stencil domain, i.e. when there are very fine filaments of one fluid phase. Due to this modification, very good results are obtained at $t = \frac{T}{2}$ when the flow is inverted as well as for the final configuration $t = T$ (with $T = 6$ in this work) compared to the exact Lagrangian solution.

The results presented in [WYS08] are obtained by a CLSVOF method which uses a Lagrangian method with a second-order Runge-Kutta scheme for the advection of the piecewise linear interface. The results obtained in that work are comparable to our own results. From a merely qualitative point of view, the circle seems to be recovered slightly better in [WYS08] on the coarse grid with $h = 1/128$. Here, results with a finer mesh resolution are missing.

All in all, these comparisons merely show that although our CLSVOF method already produces good results, there is always room for improvement and the possibility to adapt the CLSVOF method for a specific purpose. Furthermore, all these improvements are straightforward to integrate into our already very accomplished implementation of the method.

Conclusions

Although we have not yet shown any convergence results, we can already draw several remarkable conclusions from this challenging test case.

First, the CLSVOF method performs best of all four presented methods. It is well equipped to deal with severe topological changes and to maintain the mass of the thin filaments developing on the scale of the mesh size. Furthermore, the circle is quite accurately returned to its initial position. Remarkably, the results with the CLSVOF and the LS method are very close to each other: Up to the onset of mass loss, the interface of the LS method is always almost equal to that recovered by the CLSVOF method. This is a very good result, since the re-distance algorithm in the CLSVOF method is completely different from the classical Hamilton-Jacobi reinitialization in the LS method. Therefore, we can strongly infer that our new vertex finding algorithm in the re-distancing of the CLSVOF method works perfectly. Last but not least, the CLSVOF method is least computationally expensive, which is indispensable for the computation of highly resolved real life problems such as droplet impact.

Second, the results with the LS method are not too far from those with the CLSVOF method on the finest grid. Actually, this is what we expect of the LS method, which exhibits several high-performance features like WENO schemes for reinitialization and transport and high-order time discretization methods. These methods, however, have to be paid for in computing time, so that the LS method takes twice as long as the CLSVOF method on the fine grid, where it produces reasonable results. Still, even on the fine grid, the mass loss of the LS method is a severe problem, so that the initial circle can be hardly recovered. This mass loss intensifies on the coarse grid, where the LS method no longer produces results comparable to the CLSVOF method.

Third, our implementation of the LSG method renders an already unphysical method useless for the tracking of topological changes with the development of thin filaments.

Thus, its usefulness is limited to applications where such strong deformations do no occur, such as the rise of bubbles with low density jumps or the impact of droplets in regimes where drastic topological changes or splashing are unlikely to occur.

Fourth, the LSL method performs worst in this special test case. If possible, it seems even worse equipped to deal with thin filaments on the scale of the mesh than the LSG method. The mass loss in this case is so severe, that the whole interface simply vanishes after some time. However, as long as the structures do not become too thin, the LSL might handle strong topological changes better than the LSG method, which will be seen in Subsection 6.3.1. Unfortunately, the LSL method in its current implementation is also prohibitively expensive. To us, it remains unclear how much larger the time-step in the reinitialization equation can be chosen compared to the LS method, so that it becomes a competitive method again.

All in all, the CLSVOF method compares favorably with our well-tested LS method for which convergence results have already been previously established [CGS04, CGS10, Cro10], while maintaining a much larger amount of mass. And, we have to be careful with both volume corrections, whose usefulness is restricted to a certain class of problems only.

6.2.2. Three-dimensional deformation of a sphere

Our second advection test case is the fully three-dimensional deformation of a sphere. We aim to ensure that the CLSVOF method also works superbly in 3D, while outperforming the LS method concerning the conservation of mass. As in the previous section, the deformation velocity field is multiplied by $\cos(\pi t/T)$ which causes the flow field to return to its initial state at $t = T$. In this very demanding configuration, we have a maximum deformation and thinning at $t = 1.5$ which is well suited for an evaluation of our methods. Additionally, when the flow field reverses we can judge the LS and the CLSVOF method by the accuracy with which they recover the initial configuration. In this test, we also measure the amount of mass maintained at $t = T$ quantitatively with the VisIt software[2] and compute convergence rates for both methods.

Here, we do not employ the LSL or the LSG method which already failed in the 2D version of this test and will not perform better in this much harder 3D scenario.

Setup of the numerical experiment

The flow domain Ω is a square with edge length 1 and a sphere of radius 0.15 is centered at $\bar{\boldsymbol{x}}(t = 0) = (0.35, 0.35, 0.35)^\mathsf{T}$. In the following, we denote the area occupied by the sphere by $\Omega_1 = \Omega_1(t) \subset \Omega$. The 3D deformation field is defined by

$$\begin{aligned} u &= 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)\cos\left(\frac{\pi t}{T}\right) \\ v &= -\sin^2(\pi y)\sin(2\pi x)\sin(2\pi z)\cos\left(\frac{\pi t}{T}\right) \\ w &= -\sin^2(\pi z)\sin(2\pi x)\sin(2\pi y)\cos\left(\frac{\pi t}{T}\right) \end{aligned} \tag{6.4}$$

[2]The VisIt software was used instead of ParaView since it can handle larger data set in sequential post-processing operations; see Section 6.1.

u:	$+2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)\cos\left(\frac{\pi t}{T}\right)$
v:	$-\sin^2(\pi y)\sin(2\pi x)\sin(2\pi z)\cos\left(\frac{\pi t}{T}\right)$
w:	$-\sin^2(\pi z)\sin(2\pi x)\sin(2\pi y)\cos\left(\frac{\pi t}{T}\right)$
final time :	$T = 3$
flow domain:	$\Omega = [0,1] \times [0,1] \times [0,1]$
circle radius:	0.15
circle center:	$(0.35, 0.35, 0.35)$
interface thickness:	$\epsilon = 1.75h$
grid resolution:	1/32, 1/64, 1/128, 1/256, 1/512 and 1/150
employed methods:	LS & CLSVOF
boundary conditions:	no-slip

Table 6.3.: Simulation parameters for the three-dimensional deformation of a sphere.

with time t and $T = 3$ the time at which the flow should have returned the sphere back to its initial position. In Figure 6.5, the movement and deformation of the sphere over time is exemplified. Here, we see that the sphere is deformed by two rotating vortices which initially stretch out opposite sides of the sphere and then reverse them back to their initial shape. Note that our test case originates from a two-dimensional setting first proposed by LeVeque [LeV96].

All relevant simulation parameters are summarized in Table 6.3, and on all grids the time step is restricted by

$$\delta t = \text{cfl} \cdot \frac{h}{\max_{(i,j,k)}\left(u_{i,j,k}, v_{i,j,k}, w_{i,j,k}\right)} \tag{6.5}$$

with a CFL number cfl = 0.25. Again for this test, the velocity field is prescribed, so we do not solve the full Navier-Stokes equations. As in the 2D variant above, we are only concerned with the solution of the transport equation and the reinitialization of the LS function. All results are computed in parallel with only two of our methods, namely the LS and the CLSVOF method.

Our numerical experiment is evaluated in four steps.

1. We compare the zero level-set for both the LS and the CLSVOF method on grids with mesh sizes 1/128 and 1/256.
2. We investigate the mass convergence behavior of the LS and the CLSVOF method, i.e. we investigate how much mass can be conserved with both methods at the final time $T = 3$. Therefore, we employ five grids with mesh sizes 1/32, 1/64, 1/128, 1/256 and 1/512, respectively. The analytical mass of the sphere, computed from the parameters in Table 6.3, is

$$m_a = \frac{4}{3}\pi r^3 \approx 0.014137. \tag{6.6}$$

Numerically, we compute the mass retained at $T = 3$ in a post-processing step with the VisIt software [CBW+12]. From the difference between the numerical and the analytical solution, we establish the total mass loss in percent and the convergence rates.

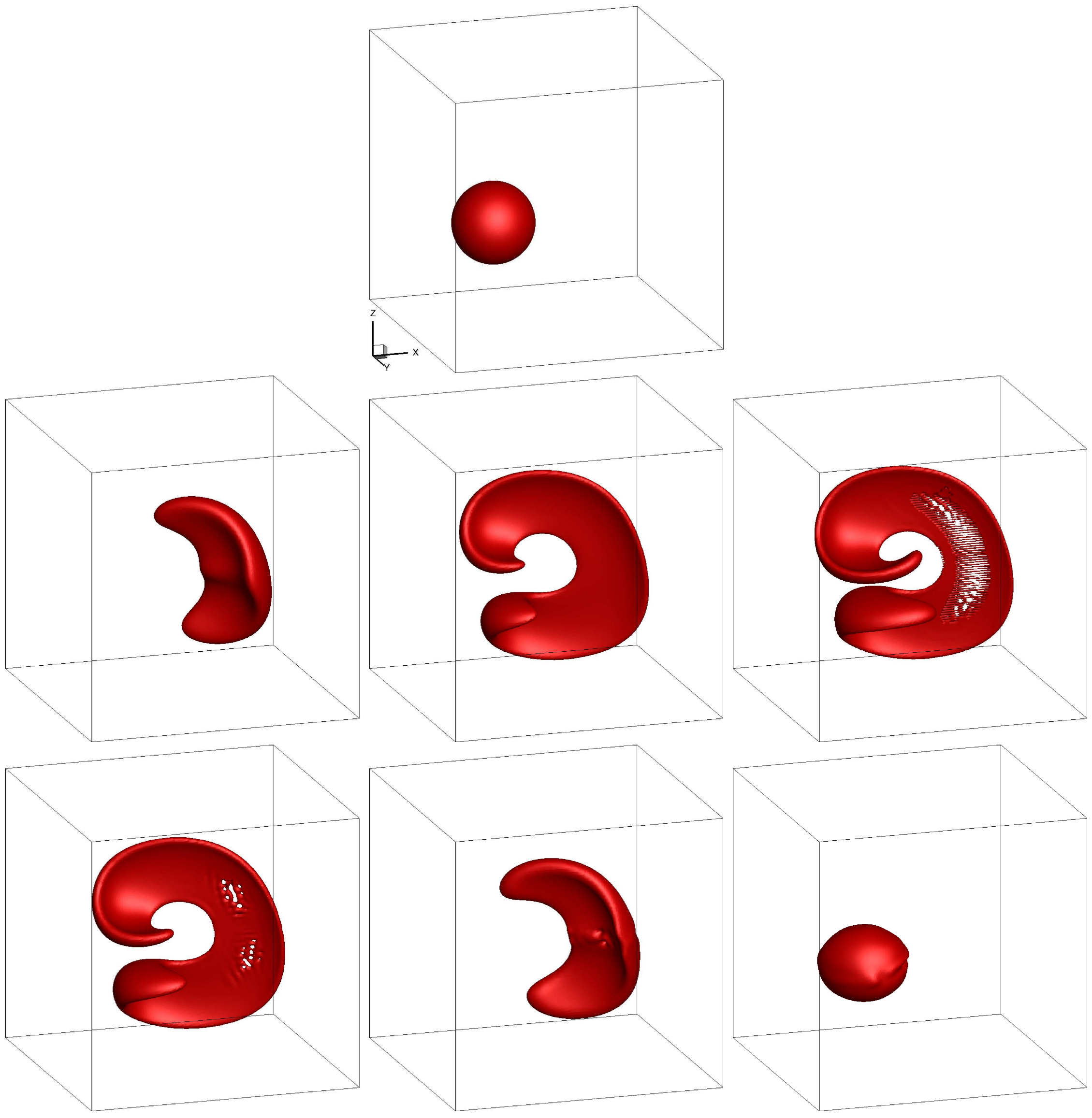


Figure 6.5.: Evolution of the three-dimensional sphere computed with the CLSVOF method on mesh size 1/150 at times 0.0, 0.48, 0.98, 1.49, 1.98, 2.48 and 3.0.

3. We measure how well the shape is returned to its initial position. Therefore, we compute the sphericity of our numerical solution at $T = 3$ which indicates how much the object's shape differs from an actual sphere. This quantity is defined in [Wad35] as
$$\Psi(T) := |\Gamma(T)|^{-1}\, \pi^{1/3}(6|\Omega_1(T)|)^{2/3}, \tag{6.7}$$
which is the surface area of a sphere with mass $|\Omega_1(T)|$ divided by the surface area $|\Gamma(T)| := \text{meas}_2\, \Gamma(T) = \int_{\Gamma(T)} 1\, ds$ of the object. Ψ is equal to 1 if Ω_1 is a sphere and decreases the more the object is flattened. Again, both the surface area and the mass are computed with the VisIt software [CBW$^+$12]. With respect to the sphericity, we are also interested in the barycenter $\bar{\boldsymbol{x}}$ of the object,
$$\bar{\boldsymbol{x}}(t) := |\Omega_1|^{-1} \int_{\Omega_1(t)} \boldsymbol{x}\, d\boldsymbol{x}, \tag{6.8}$$
which should return to $\bar{\boldsymbol{x}}(0) = (0.35, 0.35, 0.35)$ at $T = 3$.
4. We present a qualitative comparison of our results with those from the literature on a grid with mesh size 1/150.

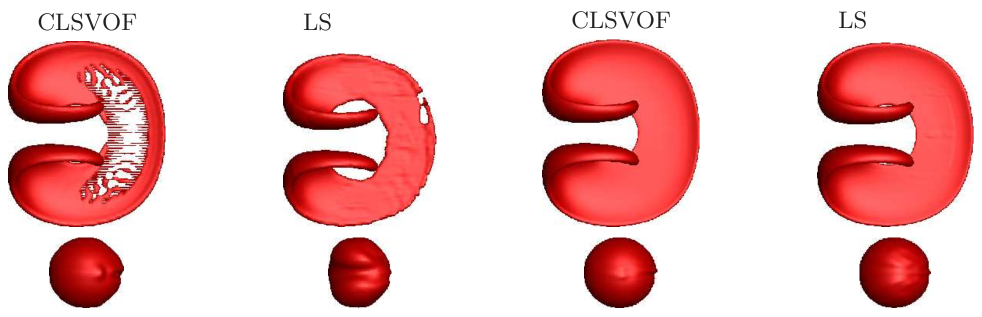
Expected Results

The three-dimensional deformation of a sphere is a very challenging test case concerning the conservation of mass, which should considerably improve with the CLSVOF method. Since a very thin membrane on the scale of the mesh size evolves, both methods have to principally demonstrate their ability to conserve mass despite these severe topological changes. Furthermore, the advected shape has to be returned to its original location.

Certainly, concerning the conservation of mass, we expect that the LS method will perform less well than the CLSVOF method on all grids since the CLSVOF method should be mass conservative. However, small mass losses may still occur with the CLSVOF method due to the truncation of the volume fraction to satisfy $0 \leq F \leq 1$ at all times. On a fine enough grid, both the LS method and the CLSVOF method should be able to conserve close to 100% of the mass.

Concerning convergence, both methods have to show a substantial improvement in their ability to conserve the initial mass of the sphere on successively refined grids. Note that the operator splitting method for the transport of both the LS and the CLSVOF method is second order accurate in space and time. Additionally, the reinitialization of the LS method benefits from a fifth order WENO scheme in space and a third-order Runge-Kutta scheme in time and no interface reconstruction is necessary. Therefore, we expect a quadratic convergence rate with the LS method. In contrast, we perform a piecewise linear interface reconstruction (PLIC) in every time step of the CLSVOF method, which might result in a lower convergence rate.

Both the CLSVOF and the LS method are also evaluated by their ability to recover the initial spherical configuration. On the finest grid, the recovered shape should be almost spherical wherefore the sphericity should tend to 1. On coarser grids, both methods will show their distinctive problems in returning the sphere to its initial shape and location: The LS method tends to lose mass where thin structures and other irregularities occur



(a) $h = 1/128$ **(b)** $h = 1/256$

Figure 6.6.: Three-dimensional sphere deformation with the CLSVOF (left) and the LS method (right) for two grids with mesh widths 1/128 and 1/256. For both resolutions, $t = 1.5$ in the top and $t = 3.0$ in the bottom row. Ideally at time $t = 3.0$, the deformed sphere should have returned to its original shape and location; see Fig. 6.7 for a comparison of the contour lines for mesh width 1/256.

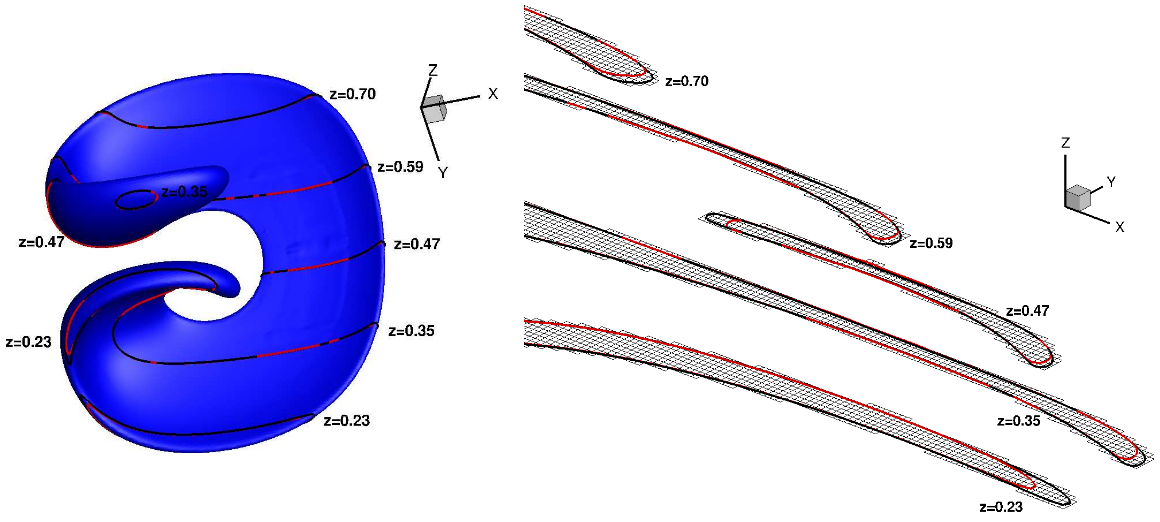
which lessen the accuracy of the LS function. This mass loss will be visible in the final reconstruction of the sphere. The main problem for the CLSVOF method is also the reconstruction of the thin membrane in the middle part of the deformed sphere. Here, mass loss can occur. Additionally, we have to take the accumulated errors of the linear interface reconstruction process into account which can lower the sphericity of the recovered shape at the end of the computation.

Description of Results

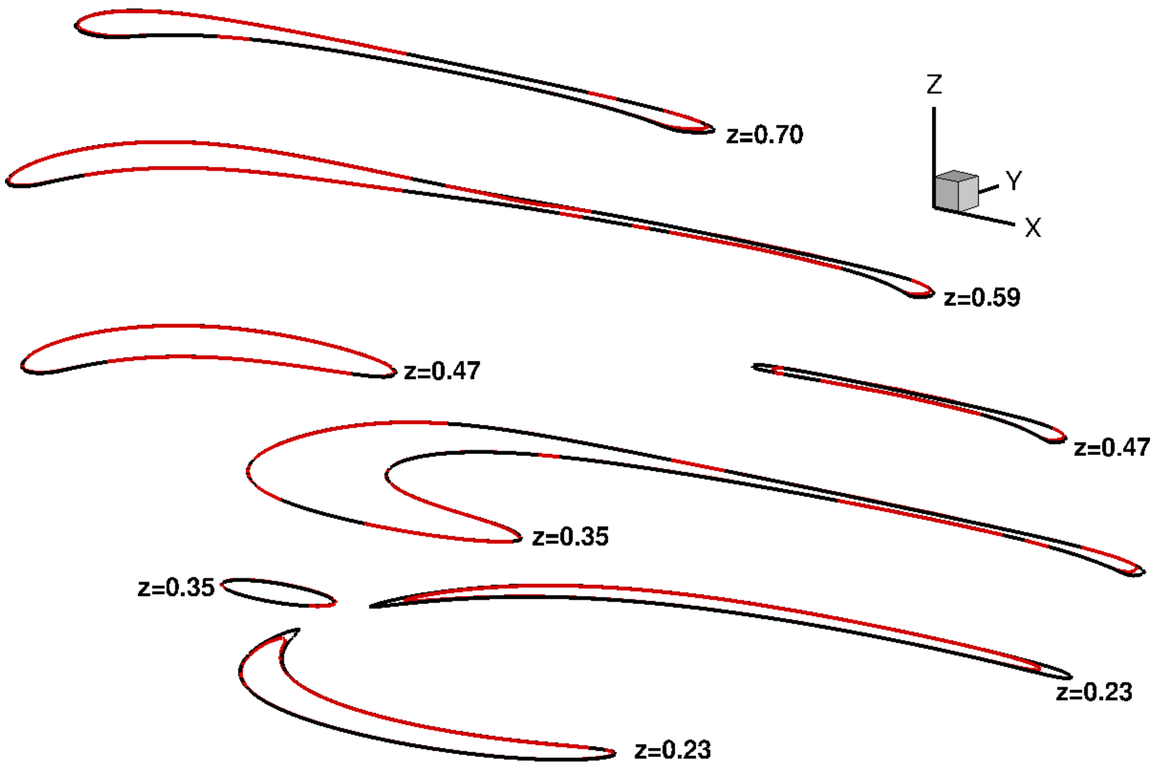
In a first step, we plot the deformed sphere at times $t = 1.5$ and $t = 3.0$ on two grids with mesh sizes 1/128 and 1/256, which is shown in Figure 6.6 for both the CLSVOF and the LS method. The deformed shapes at $t = 1.5$ correspond to maximum stretching, while at $t = 3.0$ the flow has returned the shape back to its initial position.

On the coarser grid (Fig. **??**), the CLSVOF method shows a certain improvement in mass and shape conservation compared to the LS method. At $t = 1.5$, the overall shape of the deformed sphere is very well maintained with slight mass loss in the inner part. Here, the CLSVOF method only partially resolves the thin interface produced at the middle section of the stretched shape. In contrast, the main mass loss with the LS function occurs at the boundary of the shape which becomes completely fragmented and even unsymmetrical. All in all, for both methods, the grid resolution is not sufficient to resolve the thin membrane stretched at the center part of the deformed shape. Consequently, in the final time frame, we see deviations from the initial spherical shape. Due to the larger mass loss, these deviations are more severe for the LS than for the CLSVOF method. For both, a spherical object with a 'scar' in the middle develops.

If we turn to the higher resolution (Fig. 6.6(b)), we see that both the LS and the CLSVOF method behave similarly. In order to see the differences between the CLSVOF and the LS method, we draw the zero level-set on various slices along the z-axis in Figure 6.7(b) at $t = 1.5$. Here, we observe that the LS method still loses a little bit of mass at the edges,



(a) Zero isosurface of the deformed sphere with indication of the extracted slices (left) and the number of grid cells resolving the filaments on the slices (right).



(b) Zero contour of both the LS (red) and the CLSVOF method (black) on the slices normal to the z-axis indicated above.

Figure 6.7.: Three-dimensional sphere deformation with the CLSVOF (black) and the LS method (red) for the grid with mesh width 1/256 (cf. Fig. 6.6(b)). Shown are the zero level-set contour of both the LS and the CLSVOF method on various slices normal to the z-axis at $t = 1.5$.

while the CLSVOF method is now able to resolve the thin structure which only consists of 2–4 grid cells in width (Fig. 6.7(a)). Then, at $t = 3.0$, the scar in the final shape is again visible with the CLSVOF method, which is due to the anticipated accumulated errors of the interface reconstruction process. Similar but more severe deformations occur for the LS method due to its mass loss. However, both methods return the shape to a very near spherical state.

Second, we quantify the mass loss of both methods, i.e. we investigate how much mass can be conserved with the LS and the CLSVOF method at $t = 3.0$. Therefore, we employ grids with mesh sizes 1/32, 1/64, 1/128, 1/256 and 1/512, which correspond to levels $l = 1, \ldots, 5$ in what follows. In Table 6.4, the percentage of the still remaining mass is given. Most notably, this is a very hard test case for the LS method: On the coarsest grid, no mass at all is left. Then, we lose nearly all of the mass on level 2 and about 25% of mass on level 3. Even on the finest grid, the mass cannot be preserved up to 100%. In contrast, the CLSVOF method loses only 15% of mass on the coarsest grid, and from level 3 upwards, the CLSVOF method conserves up to 100% of mass. By comparison, we see that the CLSVOF method on level 1 outperforms the LS method on level 3 by 10% concerning mass conservation. Additionally, on the finer grids, the CLSVOF method conserves almost all of the mass, which the LS method is unable to achieve.

Let us distinguish the effects of the LS and the CLSVOF method on the overall mass convergence behavior in space and time at $t = 3.0$. Therefore, we compute the discrete error norm e and convergence rate ρ by

$$e^l = |m_l - m_a| \text{ and } \rho^{l+1} = \frac{\ln\left(\frac{e^l}{e^{l+1}}\right)}{\ln\left(\frac{h^l}{h^{l+1}}\right)} = \frac{\ln\left(\frac{e^l}{e^{l+1}}\right)}{\ln(2)} \tag{6.9}$$

since $h_l = 2h_{l+1}$ for the discrete mesh width. Here, the analytical mass m_a is given by eq. (6.6) and the discrete mass is computed with the VisIt software. Our results are summarized in Table 6.5 and Figure 6.8 where we see second order convergence for the LS method and a convergence order of about 1.4 for the CLSVOF method. As expected, the convergence rate is slightly worse for the CLSVOF method. However, the absolute error in mass on the coarsest grid for the CLSVOF method is smaller than the error of the LS method on level 3. Furthermore, on the finest grid, the absolute error is still about an order of magnitude smaller with the CLSVOF than with the LS method.

In a third step, we measure how well both methods return the sphere to its initial configuration. Therefore, we compute the numerical solution's sphericity Ψ defined in eq. (6.7) at $t = 3.0$, which indicates how much the object's shape differs from a sphere. These results are shown in Table 6.4 on levels 1 to 5: Both sphericities are close to 1 from level 3 onwards. Despite the large mass loss of the LS solution on levels 2 and 3, the sphericity is close to that obtained by the CLSVOF method. This is exactly what we expected of the LS method where possible irregular structures are simply smoothed away or lost. Then, although much smaller than the original shape, we still have a resemblance to a sphere.

Next, we measure the sphere's barycenter $\bar{\boldsymbol{x}} = \bar{\boldsymbol{x}}(3)$ defined by eq. (6.8) at $t = 3.0$, which should be as close to the initial position $\bar{\boldsymbol{x}}(0) = (0.35, 0.35, 0.35)^{\mathsf{T}}$ as possible. Therefore,

Level l	Mass LS	Mass CLSVOF	Ψ_{LS}	Ψ_{CLSVOF}
1	0	85.8	0	0.775
2	13.2	95.2	0.720	0.831
3	74.5	99.3	0.918	0.966
4	94.0	99.7	0.990	0.996
5	98.5	99.9	0.999	0.998

Table 6.4.: Volume conservation in % and sphericity Ψ at $T = 3.0$ for the three-dimensional deformation of a sphere.

	LS		CLSVOF	
Level	e^l_{LS}	ρ^l_{LS}	e^l_{CLSVOF}	ρ^l_{CLSVOF}
1	1.414_{-2}	–	2.007_{-3}	–
2	1.227_{-2}	0.204	6.777_{-4}	1.566
3	3.605_{-3}	1.767	1.029_{-4}	2.720
4	8.427_{-4}	2.097	3.657_{-5}	1.492
5	2.078_{-4}	2.020	1.377_{-5}	1.409

Table 6.5.: Table of mass convergence for the three dimensional deformation of the sphere with the LS and the CLSVOF method evaluated at $t = 3.0$; see Fig. 6.8 for a convergence plot.

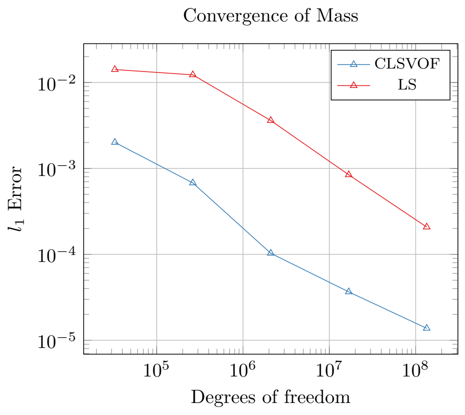


Figure 6.8.: Mass convergence history with the LS and the CLSVOF method for the three-dimensional deformation of a sphere; compare Table 6.5.

	LS			CLSVOF		
Level l	$\bar{\boldsymbol{x}}$	$e^l_{\bar{\boldsymbol{x}}}$	$\rho^l_{\bar{\boldsymbol{x}}}$	$\bar{\boldsymbol{x}}$	$e^l_{\bar{\boldsymbol{x}}}$	$\rho^l_{\bar{\boldsymbol{x}}}$
1	–	–	–	$\begin{pmatrix}0.434002\\0.330987\\0.326961\end{pmatrix}$	8.916_{-02}	–
2	$\begin{pmatrix}0.359359\\0.357897\\0.3557440\end{pmatrix}$	1.353_{-02}	–	$\begin{pmatrix}0.358568\\0.345924\\0.34369\end{pmatrix}$	1.139_{-02}	2.968
3	$\begin{pmatrix}0.347477\\0.36358\\0.362297\end{pmatrix}$	1.849_{-02}	-0.451	$\begin{pmatrix}0.348756\\0.349334\\0.348168\end{pmatrix}$	2.312_{-03}	2.301
4	$\begin{pmatrix}0.350391\\0.35147\\0.351476\end{pmatrix}$	2.120_{-03}	3.125	$\begin{pmatrix}0.35183\\0.349468\\0.348901\end{pmatrix}$	2.200_{-03}	0.072
5	$\begin{pmatrix}0.35056\\0.350067\\0.350067\end{pmatrix}$	5.680_{-04}	1.900	$\begin{pmatrix}0.351129\\0.349603\\0.349314\end{pmatrix}$	1.379_{-03}	0.673

Table 6.6.: Errors e and convergence rates ρ obtained for the barycenter $\bar{\boldsymbol{x}} = \bar{\boldsymbol{x}}(3)$ of the sphere at $t = 3.0$ for the three-dimensional deformation problem; ρ and e are computed by (6.10).

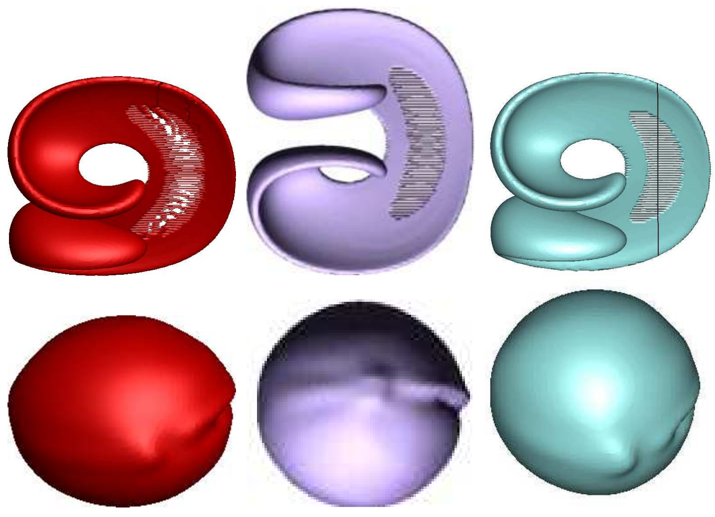


Figure 6.9.: Three-dimensional sphere deformation with our solver NaSt3DGPF (red), results from [WYKS09] (violet) and from [MTB07] (blue) at times $t = 1.5$ (top) and $t = 3.0$ (bottom). Unfortunately, the results in [WYKS09] and [MTB07] employ different perspectives.

we compute the error

$$e^l_{\bar{\boldsymbol{x}}} = \left\| \bar{\boldsymbol{x}}(3)^l - \bar{\boldsymbol{x}}(0) \right\|_2 \text{ and the rate } \rho^l_{\bar{\boldsymbol{x}}} = \frac{\ln\left(\frac{e^l_{\bar{\boldsymbol{x}}}}{e^{l+1}_{\bar{\boldsymbol{x}}}}\right)}{\ln(2)}. \tag{6.10}$$

These results are given in Table 6.6. A final convergence rate cannot be established in this case since the asymptotic regime is not yet reached. Here, computations on even finer grids would be required. The position with the LS method seems to converge from level 3 onwards with a substantial reduction of the error on these levels. However, for the CLSVOF method from level 3 onwards, the errors are of roughly the same order with no substantial improvement. In contrast, on the coarse grids, the final position with the CLSVOF method is already established quite well and better than with the LS method. To summarize, both methods are roughly able to return the shape to its initial position – at least on the finer grids, which is what we expected.

Comparison to results from the literature

Last, we compare our results to those from the literature. In Figure 6.9, for a resolution $h = 1/150$, we display our results (red) with those by Wang et al. (violet) [WYKS09] and with those by Ménard, Tanguy and Berlemont (blue) [MTB07] at $t = 1.5$ and $t = 3.0$. All three results are very close to each other and mainly differ in their ability to resolve the thin membrane in the middle of the droplet. Here, the CLSVOF method by Ménard performs best since it is specifically adapted to handle very thin filaments for the simulation of the primary break-up of liquid jets. The blue results as presented in [WYS08] are obtained by a CLSVOF method which uses a Lagrangian method with a second-order Runge-Kutta scheme for the advection of the piecewise linear interface. These results compare very well with our own work. However, the thin structure is recovered slightly better in [WYS08] and our final shape is a little more stretched than in both examples from the literature. For all CLSVOF methods, a scar is present in the sphere in the final time frame, which is probably due to slight mass loss and accumulated numerical errors in the interface reconstruction process [WYKS09].

Note that the 3D results presented in [WYS08] are obtained by the partly incorrect reinitialization as described in Subsection 5.6.1. Despite this error, the results look good from a qualitative point of view. A convergence study for the applied flow solver could shed light on the question in how far the faulty vertex finding algorithm influences the overall behavior of the CLSVOF method, which cannot yet be deduced from the performed numerical experiments in [WYS08].

Again, from a qualitative point of view, all three results are in very good agreement with each other and the slight differences in mass conservation can be attributed to the applied methods of transport, reinitialization and interface reconstruction.

Conclusions

In this subsection, we established that both the LS and the CLSVOF method fare very well with our three-dimensional test case. From level 3 onwards both methods are able

to return the deformed shape back to its initial position and to its spherical state. From level 4 onwards, both methods conserve a substantial amount of mass. Furthermore, we established the expected second-order convergence rate for the LS and a convergence order 1.4 for the CLSVOF method. Despite the better rate of the LS method, the mass error with the CLSVOF method was always about 1 to 3 orders of magnitude smaller. Therefore, and looking back at Figure 6.8, it seems unlikely that the point where the LS method outperforms the CLSVOF method concerning mass conservation becomes relevant – even with today's computing power.

All in all, our results are very much in favor of the CLSVOF method. Also in three dimensions, this method is well equipped to deal with severe topological changes and to maintain the mass of the thin filaments developing on the scale of the mesh size. Furthermore, this test case strongly supports the 3D suitability of our new vertex finding algorithm in the re-distancing of the CLSVOF method.

Thus, we are now perfectly equipped to move on to test cases which involve the whole flow solver.

6.3. The rising bubble benchmark

The modeling and simulation of two- and three-dimensional two-phase flows is still an area of active research. Various approaches have been developed to improve conservation of mass, capturing of the fluid's interface or computation of interfacial surface tension. For the validation of these new methods, qualitative benchmark data is required.

In this section, we perform 2D and 3D incompressible two-phase flow simulations of rising droplets based on a similar 2D benchmark proposed in [HTK$^+$09]. In 2D, we compare our CLSVOF method to the LS and LSL method. Furthermore, we compare our results with the CLSVOF method to those presented in [HTK$^+$09]. In addition, we propose a 3D benchmark configuration with two test cases in which we compare our flow solver NaSt3DGPF [CGS10] with DROPS [DRO08, RFG$^+$] and OpenFOAM [Ope13]. These results are already partly published in [AEG$^+$14]. In that work, however, we employ the LSL only, not the CLSVOF method. Since the results computed with the different flow solvers are not entirely consistent in [AEG$^+$14], we are very much interested in how the CLSVOF method fares with these problems. Furthermore, we have already established that the use of the LSL method is not without problems. In this section, the CLSVOF method will continue to prove its mass conserving, computationally efficient and accurate characteristics.

6.3.1. Two-dimensional

In this subsection, we consider a two-dimensional bubble with very low density compared to the surrounding fluid, which compares to test case 2 studied extensively in [HTK$^+$09]. The large density and viscosity ratios of $\rho_1/\rho_2 = 1000$ and $\mu_1/\mu_2 = 100$ are roughly the same as for an air bubble in water. Furthermore, a break-up of the bubble is possible due to the low surface tension value. According to the classification by Clift et al. [CGW78], our test case lies somewhere between the skirted and dimpled ellipsoidal-cap regimes.

Here, we are primarily interested in a further comparison of the CLSVOF, the LS and the LSL method. Therefore, we measure and compare three different benchmark quantities

final time	$T = 3$
flow domain:	$\Omega = [0,1] \times [0,2]$
circle radius:	0.25
circle center:	$(0.5, 0.5)$
densities:	$\rho_l = 1000$ and $\rho_g = 1$
viscosities:	$\mu_l = 10$ and $\mu_g = 0.1$
surface tension:	$\sigma = 1.96$
body force:	$\boldsymbol{g} = (0.0, -0.98)$
interface thickness:	$\epsilon = 1.9h$
grid resolution:	31×61, 61×121, 121×241 and 241×481
employed methods:	LS, LSL, CLSVOF
quasi 2D:	periodic boundary conditions in z-direction
z-resolution LS and LSL:	5
z-resolution CLSVOF:	5, 5, 10 and 20
boundary conditions:	slip except for z-direction

Table 6.7.: Material and simulation parameters for the two-dimensional rising bubble.

and present a mass convergence analysis. Additionally, we compare the benchmark quantities computed with the CLSVOF method to those computed by the flow solvers presented in [HTK+09].

In this numerical experiment, we do not employ the LSG due to our previous studies of rising bubbles with this method, where the evolving filaments at the rear end of the bubble were almost always lost. Inherently, the Picard iteration 4.14 does not correct the mass at the rear end of the bubble only but uniformly in space. Therefore, we ended up with results very similar to those with the LS method, which were merely blown up in each spatial direction.

Note that in this experiment, we compute 2D results with our 3D flow solver. Therefore, similar to the previous 2D test case described in Subsection 6.2.1, we employ a small number of cells in the periodic z-direction. For a better readability in this subsection, we omit the third space direction in our description of the experiment and its evaluation. Thus, the evaluation is always carried out on one slice in the z-direction or averaged over all slices. We employ 5 grid cells in the z-direction for the LS and the LSL method and a resolution dependent number for the CLSVOF method; see Table 6.7. Again, this difference is due to our implementation of the CLSVOF method.

Setup of the numerical experiment

Consider a rectangular tank $\Omega = [0,1] \times [0,2]$ and a bubble $\Omega_2 = \Omega_2(t) \subset \Omega$ which is lighter than the surrounding fluid $\Omega_1 = \Omega \setminus \Omega_2$. Initially, this bubble is assumed to be circular with radius $r = 0.25$ and center point $\boldsymbol{x}_c = (0.5, 0.5)^t$, i.e. fluid phase 2 is initially given by

$$\Omega_2(0) = \{\boldsymbol{x} \in \Omega \mid \|\boldsymbol{x} - \boldsymbol{x}_c\| \leq r\}.$$

The final time T is prescribed as $T = 3$, and we assume slip boundary conditions on all walls. The initial condition for the Navier-Stokes equations is given by $\boldsymbol{u}(0) = 0$. Due to buoyancy effects, the bubble will start to rise and change its shape.

Our test case is defined according to the material and simulation properties given in Table 6.7. The material parameters are largely taken from test case 2 of the 2D two-phase flow benchmark proposed in [HTK+09]. Note that in [HTK+09], no-slip boundary conditions are imposed on top and bottom boundaries, while slip boundary conditions are employed on vertical walls. Here, we employ slip boundary conditions everywhere. Only for the final comparison of the CLSVOF method with the flow solvers in [HTK+09], we use the no-slip condition on top and bottom walls.

In this experiment, we aim at a further qualitative and quantitative comparison of the LSL, the CLSVOF and the LS method, since now the whole flow solver is involved in the computational process.

1. We compare the evolving bubble shapes qualitatively on successively refined grids.
2. We measure the mass convergence of all three methods over time against the analytical solution. Therefore, we denote the time steps over which we average by t_i, $i = 1, \dots, N$ with N the number of steps. Then, for $i = 1, \dots, N$ and the bubble's mass m we define the relative error norms

$$l_1\text{-error: } \|e^l\|_1 := \frac{\sum_{t_i=1}^{t_N} \left| m_l^{t_i} - m_a \right|}{N \cdot m_a} \tag{6.11}$$

$$l_2\text{-error: } \|e^l\|_2 := \left(\frac{\sum_{t_i=1}^{t_N} \left| m_l^{t_i} - m_a \right|^2}{N \cdot m_a^2} \right)^{\frac{1}{2}} \tag{6.12}$$

$$l_\infty\text{-error: } \|e^l\|_\infty := \frac{\max_{t_i} \left| m_l^{t_i} - m_a \right|}{m_a}. \tag{6.13}$$

   With these norms, we compute the mass conservation over time, the final mass and the convergence rates for all three methods. Here, m_a is the analytical mass of the bubble given by $m_a = \pi r^2 \approx 0.1963$.
3. We define time-dependent quantities of interest which are used to compare the different simulations with each other (cf. [HTK+09]). The first one is again the barycenter $\bar{\boldsymbol{x}}$ of the bubble,

$$\bar{\boldsymbol{x}}(t) := |\Omega_2|^{-1} \int_{\Omega_2(t)} \boldsymbol{x} \, d\boldsymbol{x}, \tag{6.14}$$

   with $|\Omega_2| := \text{meas}_2\, \Omega_2 = \int_{\Omega_2} 1 \, d\boldsymbol{x}$ the volume/mass of the bubble. The second one is its rise velocity $\bar{\boldsymbol{u}}$ given by

$$\bar{\boldsymbol{u}}(t) := |\Omega_2|^{-1} \int_{\Omega_2(t)} \boldsymbol{u} \, d\boldsymbol{x}. \tag{6.15}$$

   Last, similar to the sphericity defined by eq. (6.7) in 3D, we now define the circularity $\mathbb{c}$ as in [Wad35] by

$$\mathbb{c}(t) := \frac{P}{|\Gamma(t)|} \tag{6.16}$$

   which is the perimeter $P = \pi d$ of a circle divided by the perimeter $\Gamma(t)$ of the bubble.

In particular, the circle with perimeter P is defined by having an area equal to that of the bubble, i.e. we determine its diameter d by

$$\frac{\pi d^2}{4} = |\Omega_2| \Leftrightarrow d = \sqrt{\frac{4}{\pi}|\Omega_2|}. \tag{6.17}$$

Then, we substitute d in (6.16) and obtain

$$\text{¢}(t) = \frac{P}{|\Gamma(t)|} = \frac{2\sqrt{\pi|\Omega_2|}}{|\Gamma(t)|}. \tag{6.18}$$

Thus, ¢ is equal to 1 if Ω_2 is a circle and decreases the more the bubble is flattened.

4. We measure the computing time of the LSL, CLSVOF and LS method, which now also involves the whole flow solver. In particular, we are interested in the computational routines of our flow solver which take up most of the computing time.
5. We compare the benchmark quantities computed with the CLSVOF method to those computed by the flow solvers presented in [HTK+09]. For this final comparison, we do not take the slip boundary condition everywhere. Instead, we impose the no-slip boundary conditions on top and bottom walls, while slip boundary conditions are employed only on vertical walls. Then, we have exactly the same boundary conditions as in [HTK+09].

Note again that we employ the ParaView software [Par11] for the computation of all area and surface integrals involved in the definition of the norms (6.11)–(6.13) and for the computation of the benchmark quantities (6.14), (6.15) and (6.18).

Expected behavior

Due to the low surface tension value, the bubble in our test case will develop thin filaments which can eventually break off. However, we already know from the advection tests that the CLSVOF method is well-equipped to conserve mass and maintain filaments on the scale of the mesh size. Therefore, we anticipate superior performance of this method on almost all successively refined grids in this study. On coarse grids, the LS method loses too much mass for a comparative solution. However, on the finest grid both methods should produce comparable results. As long as the filaments of the bubble do not become too thin (cf. Subsec. 6.2.1), the LSL will also perform well for the rising bubble problem. On the finest grid, the contours of the LSL and the LS method should be very close. However, we are also aware that the LSL method is prone to introduce noise in the curvature [Her05, Kec98] which now might become an issue. All in all, we expect similar contour lines and results of the benchmark quantities for all three methods.

Concerning convergence, all methods are expected to show a substantial improvement in their ability to conserve the initial mass of the bubble on successively refined grids. As in Subsection 6.2.2, we can at best expect close to quadratic convergence with the LS method and a slightly lower convergence rate with the CLSVOF method. Furthermore, the involvement of the whole flow solver might result in a lower overall convergence rate than in Subsection 6.2.2 for both the LS and the CLSVOF method. For the LSL method, we anticipate a similar convergence behavior as with the LS method provided that the solution does not degenerate on finer grids due to noise in the curvature.

Again for the computing time, we expect the CLSVOF method to be more efficient than the LS method due to our results in Subsection 6.2.1. The bottleneck for the LS and the LSL method is the reinitialization of the LS function, which will take up most computing time. The LSL method performs the same reinitialization as the LS method with an added constraint (4.17) for every interface cell as explained in Subsection 6.2.1. Note again, that in the original article [SF99], the time-step τ of the reinitialization equation (4.18) with the constraint (i.e. for the LSL method) is chosen 5 times that of the LS method, since 'the constraint allows us to iterate with a larger time-step and with less error' [SF99, p.1182]. In this experiment, however, we do not adapt the time-step, wherefore the LSL will be considerably slower than the LS method.

Last, we expect that the benchmark quantities computed with the CLSVOF method are close to those presented in [HTK+09]. However in this experiment, we compute 2D results with our 3D flow solver. Naturally, we hope that our quasi-2D setup is very close to an original 2D computation. However, we introduce a source of possible numerical errors by having a third space direction, which in all likelihood cannot be neglected when comparing with real 2D data. Furthermore, since we are primarily interested in a comparison between our three interface capturing methods, the dimension and grid cells in the periodic direction are not optimized to produce the 'best two-dimensional results'. Thus, choosing the right dimension and grid cells in the z-direction would probably allow for an even better comparison with the results in [HTK+09].[3]

Discussion of results

a) Contour comparison

Figures 6.10 shows the contour of the zero level-set on successively refined grids for the LSL, the LS and the CLSVOF method. Let us focus on the coarse grids with mesh sizes $h = 1/31$ and $h = 1/61$ first. On both grids, the LS method is unable to retain the long filaments of the bubble up to the final point in time. In contrast, both the LSL and the CLSVOF method compute comparable contour lines in this case. For both methods, the thickness and elongation of the bubble's filaments is very similar. Furthermore, on the coarsest grid, both predict a break-up of the bubble. Here, even the position and size of the satellite droplet is comparable.

Turning to the mesh sizes $h = 1/121$ and $h = 1/241$, we most notably observe the failure of the LSL method on the finest grid. Here, several unphysical oscillations in the curvature occur. Note that these oscillations can also be seen on the grid with mesh size $h = 1/121$, if we zoom in on the zero level-set early (not presented here). However, on this grid, the curvature dominates the noise produced by the LSL method so that a still smooth result is obtained. If we set aside the oscillations of the LSL method as well as the mass loss of the LS

[3]For example, optimization could be achieved by 'tuning' the periodic direction with the easier test case 1 presented in [HTK+09], so that our quasi-2D results are as close as possible to the (very closely agreeing) 2D ones. This might then also produce close to 2D results for the test case 2 in [HTK+09], where the computed benchmark quantities are in the same range, but the point of break up or the final bubble shape are inconclusive.

method, both contour lines are very close to each other, exhibiting a strong resemblance in the bubble's shape. In contrast, the bubble's base is much more elongated when computed with the CLSVOF method, which leads to much thinner and less smooth filaments of the bubble.

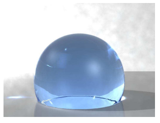
b) Quantification of mass loss

In a second step, we quantify the mass loss of our three methods. In Figure 6.11(a) to 6.11(c) we plot the mass loss over time on all grids, and in Table 6.8 we display the amount of mass retained at the end of the computation. For the example of the $h = 1/121$ grid, we compare the mass loss of all three methods with the analytical solution in Figure 6.11(d). In what follows, the grids with mesh sizes 1/31, 1/61, 1/121 and 1/241 are denoted by levels 1-4.

Let us first comment on the oscillation visible in the graphs of Figure 6.11. For example, in Figure 6.11(a) on both levels 1 and 2 at about $t = 2.3$, there seems to be a little gain of mass, after which the curve declines even steeper than before. Even before this peak, there are 'stair cases' in the mass conservation graph which get more frequent but smaller on the finer levels and reflect our discretization of the LS function. These steps result from the LS function adapting to the underlying grid. Since the ParaView software reconstructs the interface from the level-set values written during our simulation, these steps are then also visible in the computation of mass. In addition, between $t = 1.5$ and 2.5, the bubble deforms the most while forming the filaments. For such thin structures, the level-set function becomes less accurate, which can even result in a slight gain of mass due to the diffusion introduced by the reinitialization equation. Thus, all methods seem to have the most difficulties from about $t = 1.5$ onwards. Here, the oscillation period becomes shorter and most mass is lost in this time frame. Note that the oscillations in the CLSVOF method on level 1 (Fig. 6.11(c)) are more prominent, since overall only a very small amount of mass is lost. In average, the amplitude of the oscillations is less than 0.5%. Furthermore, both other methods lose more mass, which results in a smoother bubble shape and thereby less severe oscillations.

As expected, the mass loss is most severe with the LS method. On level 1, the LS method loses about 30% of mass, the LSL only 4% and the CLSVOF method a mere 2% (Tab. 6.8). Even on the finest grid level 4 there is still 2% mass loss with the LS method. However, from the corresponding Figure 6.11(a), we also observe the substantial improvement in mass conservation from one grid level to the next. This improvement also holds for the CLSVOF method (Fig. 6.11(c)) and, apart from level 4, also for the LSL method (Fig. 6.11(b)). Furthermore, at the beginning of the computation, all methods are equally able to conserve mass, and the point in time for the onset of mass loss depends heavily on the refinement of the grid: The coarser the grid, the sooner the mass loss occurs. In contrast to the LS method, the LSL method loses much less mass on levels 1–3 but deteriorates on level 4, where the oscillations in the interface are most prominent (cf. Fig. 6.10).

Finally, the comparative mass loss over time is shown in Figure 6.11(d) on level 3. Here, the CLSVOF method is at all times extremely close to the analytical solution, while the onset of the mass loss occurs at about $t = 1$ for both the LSL and the LS method. After that, the mass loss is much more intense with the LS than with the LSL method on this specific grid.



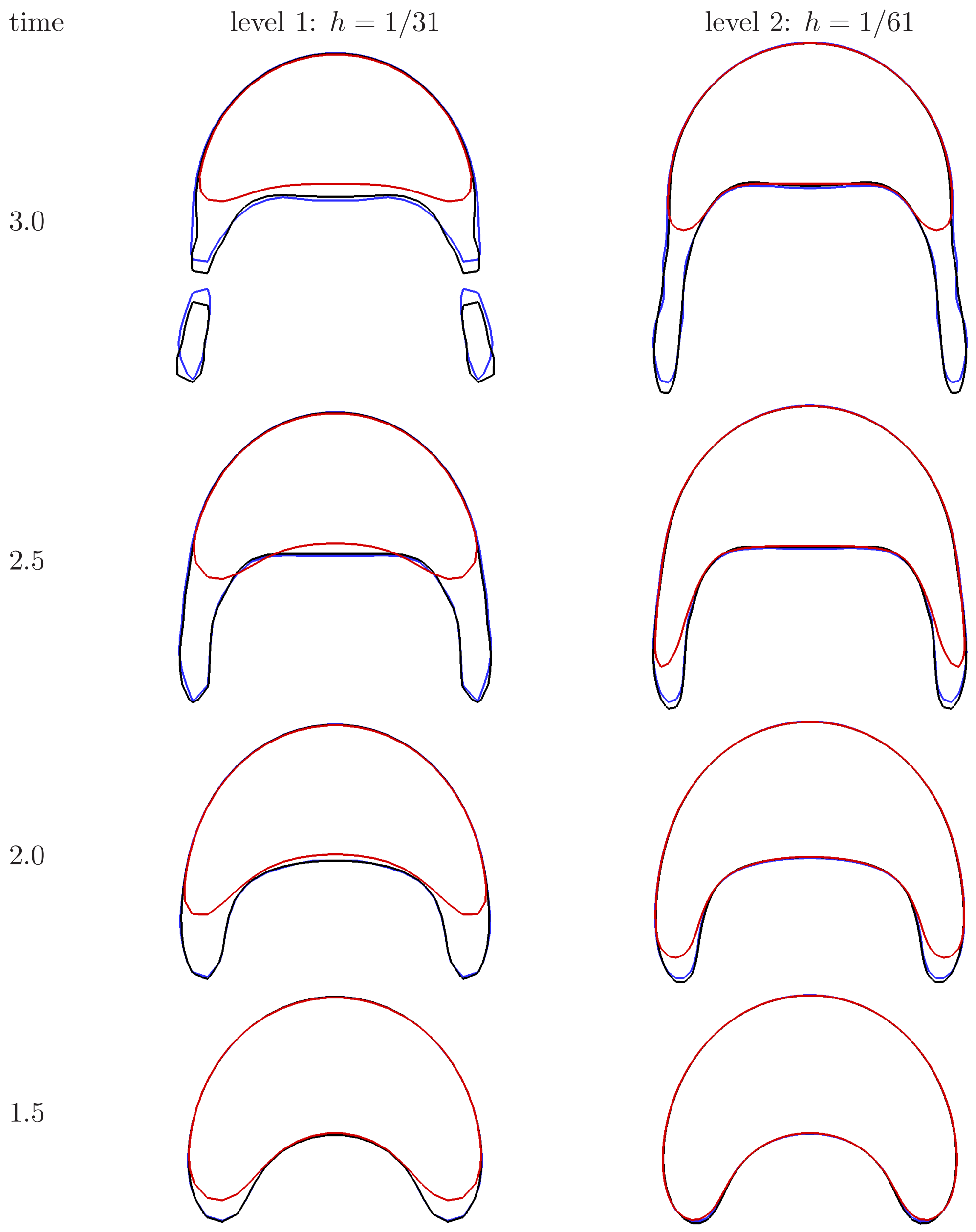


Figure 6.10.: Contours of the 2D rising bubble with the LSL (black), the LS (red), and the CLSVOF method (blue).

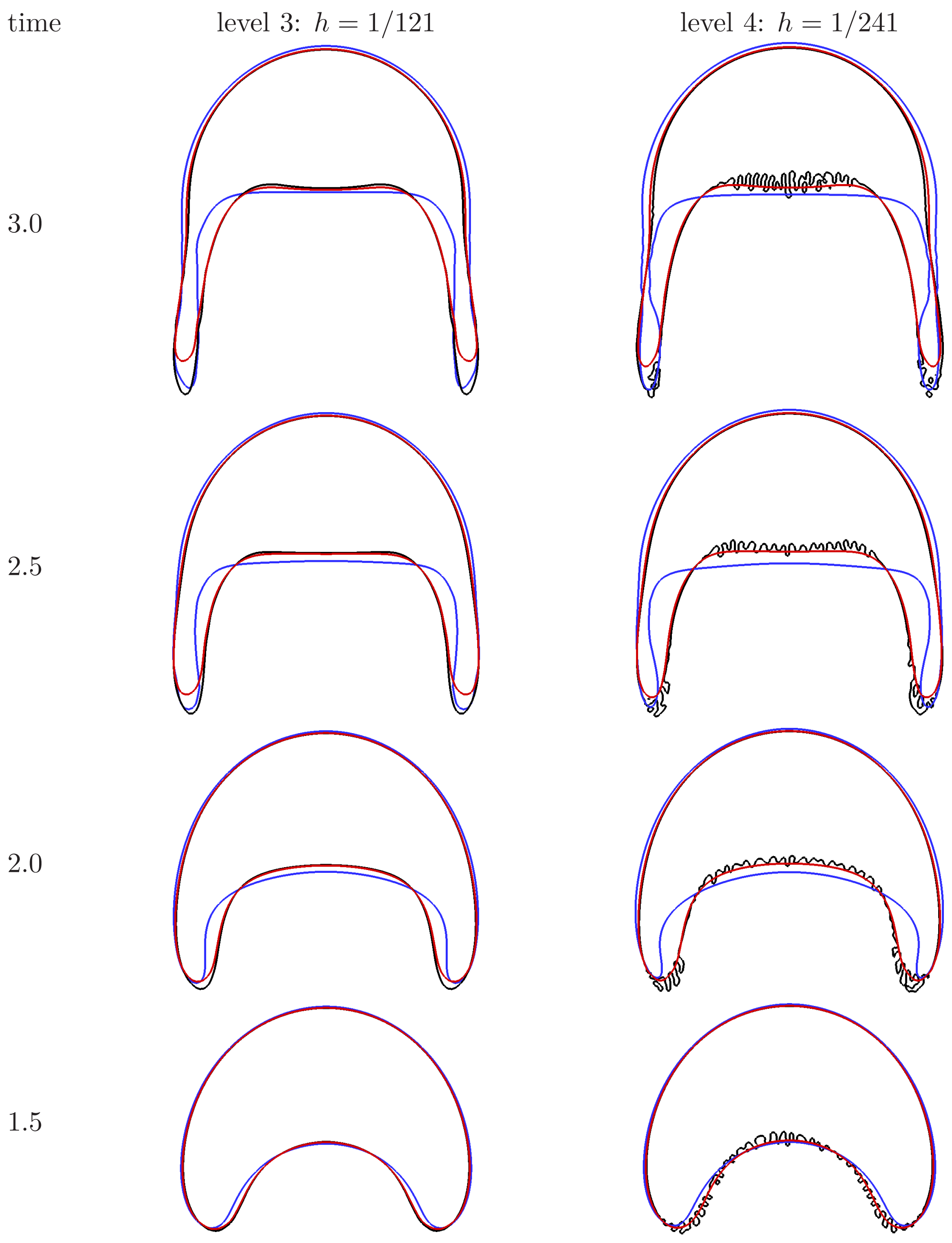
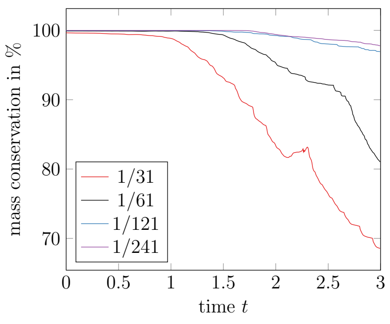
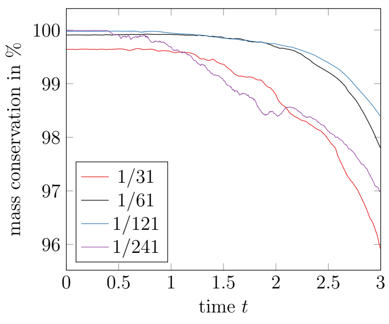


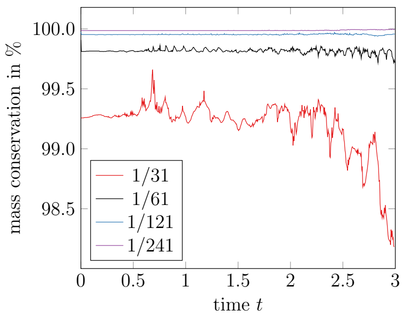
Figure 6.10.: Continued from previous page.



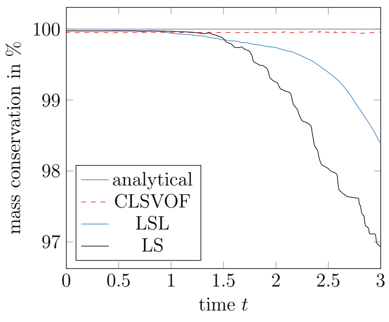
(a) LS method.



(b) LSL method.



(c) CLSVOF method.



(d) All methods on a grid with $h = 1/121$ and compared to the analytical solution (purple).

Figure 6.11.: Mass loss over time in % for the rising bubble benchmark with all three methods; compare Table 6.8 for the amount of mass conserved at the final time $t = 3$.

Level l	LS	LSL	CLSVOF
1	68.56	95.93	98.18
2	81.03	97.81	99.72
3	96.93	98.40	99.96
4	97.79	96.99	100.00

Table 6.8.: Mass conservation in % for the two-dimensional rising bubble benchmark at $t = 3$; compare Figure 6.11 for the amount of mass lost over time.

All in all, the CLSVOF method performs best at the mass conservation over time which is exactly what we anticipated.

Next, we turn to the mass convergence displayed in Table 6.9(a) and Figure 6.12(a) averaged over time $t = 0, \dots, 3$ by equations (6.11)–(6.13). For the LS method, conservation of mass improves drastically from level 1 to 3 in all error norms. However, we then only obtain an order of convergence of about 0.5 on level 4, which is not what we expected. Turning to the LSL method, we obtain no convergence at all due to the oscillative nature of this method which becomes most prominent on level 4. Instead, with the CLSVOF method, we now have optimal quadratic convergence in all norms. For this method, the 2D rising bubble seems ‘easier’ than the 3D deformation of the sphere, where we obtained a convergence order of about 1.5. Probably, the linear reconstruction of the interface has less influence here: From a comparison of the zero level-set for both examples, the 2D bubble can be much easier reconstructed by linear segments than the deformed sphere, which exhibits many more and larger deformations in all three spatial directions.

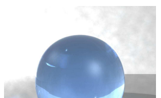
All in all, this is a very hard test case for measuring mass convergence. The bubble’s filaments result in a large number of interface cells, less accurate level-set values and thereby in a less optimal computation of the bubble’s mass with the ParaView software, especially after $t = 1.5$. As a confirmation of this assumption, we now compute the averaged relative error norms only up to $t = 1.5$ by equations (6.11)–(6.13) since until then the interface remains relatively smooth. These results are shown in Table 6.9(b) and Figure 6.12(b). Here, we obtain the anticipated second order convergence rate of the LS method – even slightly better than for the CLSVOF method. In this time frame, the LS method is even partially able to conserve more mass than the CLSVOF method as shown in Figure 6.12(b).

c) Quantities of interest

In a third step, we compare the circularity, center of mass in y-direction and rise velocity in y-direction for all three methods over time on level 3. Here, we do not take the finest grid, since the oscillations of the LSL method are more prominent on that grid which renders the comparison unfruitful.

As expected, all quantities behave similarly for the different interface capturing methods (Fig. 6.13). Thus, the circularity drops from its initial value smoothly to about 0.3 at the end of the computation. Then, both the LSL and the CLSVOF method predict a less circular shape than the LS method due to the longer evolving filaments (cf. Fig. 6.10, $t = 3$). Note that the sudden drop of the LSL method at $t = 0.5$ seems rather unphysical. A comparison of the contour lines of both the LSL and the LS method at this time could give an explanation for this strange behavior.

The center of mass results are even closer to each other than those for the circularity. Roughly, all methods predict a linear rise from about $t = 0.5$ onwards. The terminal rise velocity is identical in case of the CLSVOF and the LS method, and only slightly lower for the LSL method. Thus, the most interesting difference occurs for the computation of the rise velocity. Here, both the LSL and the LS method predict an increase up to $t = 0.7$ after which the velocity declines. Then, at about $t = 2.5$, the bubble starts to rise a little faster again. In contrast, the CLSVOF method predicts a much earlier increase of the velocity at about $t = 1.25$, followed by a decline at $t = 1.8$ and a much later increase shortly before



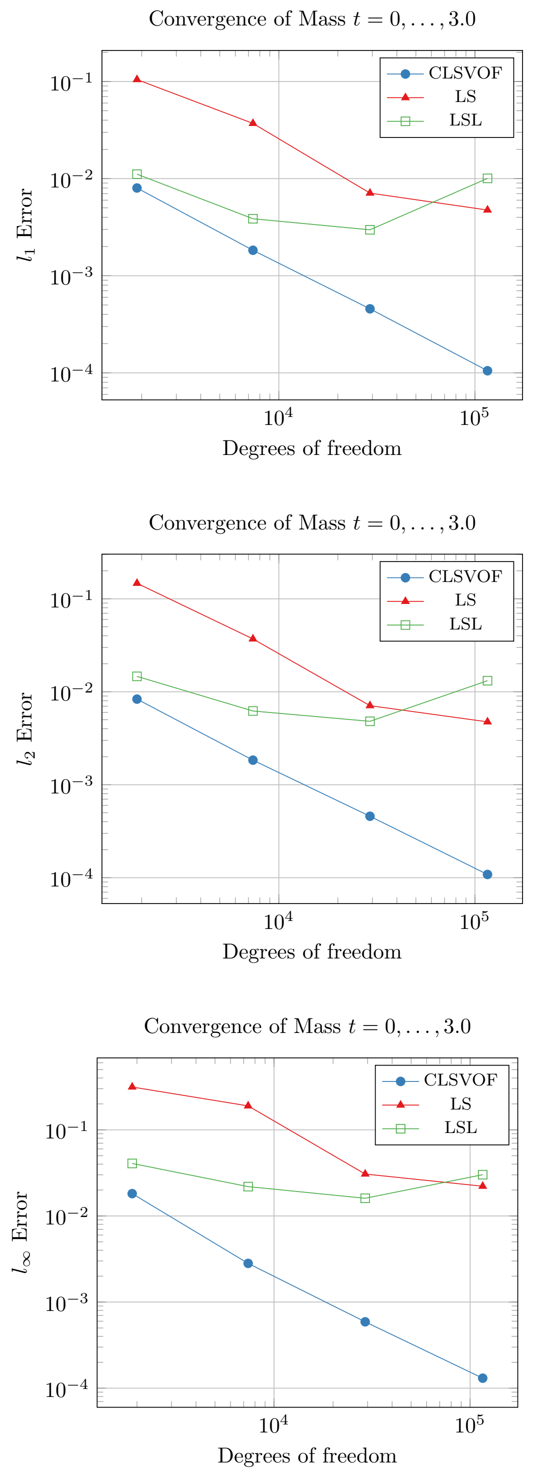
Level	e_1^l	ρ_1^l	e_2^l	ρ_2^l	e_∞^l	ρ_∞^l
			CLSVOF			
1	8.006_{-3}	–	8.339_{-3}	–	1.816_{-2}	–
2	1.829_{-3}	2.130	1.839_{-3}	2.181	2.811_{-3}	2.692
3	4.563_{-4}	2.003	4.582_{-4}	2.005	5.896_{-4}	2.253
4	1.051_{-4}	2.119	1.084_{-4}	2.079	1.306_{-4}	2.175
			LS			
1	1.047_{-1}	–	1.469_{-1}	–	3.144_{-1}	–
2	3.704_{-2}	1.499	6.158_{-2}	1.254	1.897_{-1}	0.729
3	7.093_{-3}	2.385	1.171_{-2}	2.394	3.067_{-2}	2.628
4	4.749_{-3}	0.579	8.122_{-3}	0.528	2.214_{-2}	0.470
			LSL			
1	1.111_{-2}	–	1.465_{-2}	–	4.071_{-2}	–
2	3.862_{-3}	1.524	6.213_{-3}	1.237	2.191_{-2}	0.894
3	2.971_{-3}	0.378	4.812_{-3}	0.369	1.604_{-2}	0.449
4	1.006_{-2}	−1.760	1.316_{-2}	−1.451	3.014_{-2}	−0.910

(a) $t = 0, \dots, 3$; see Fig. 6.12(a) for a convergence plot.

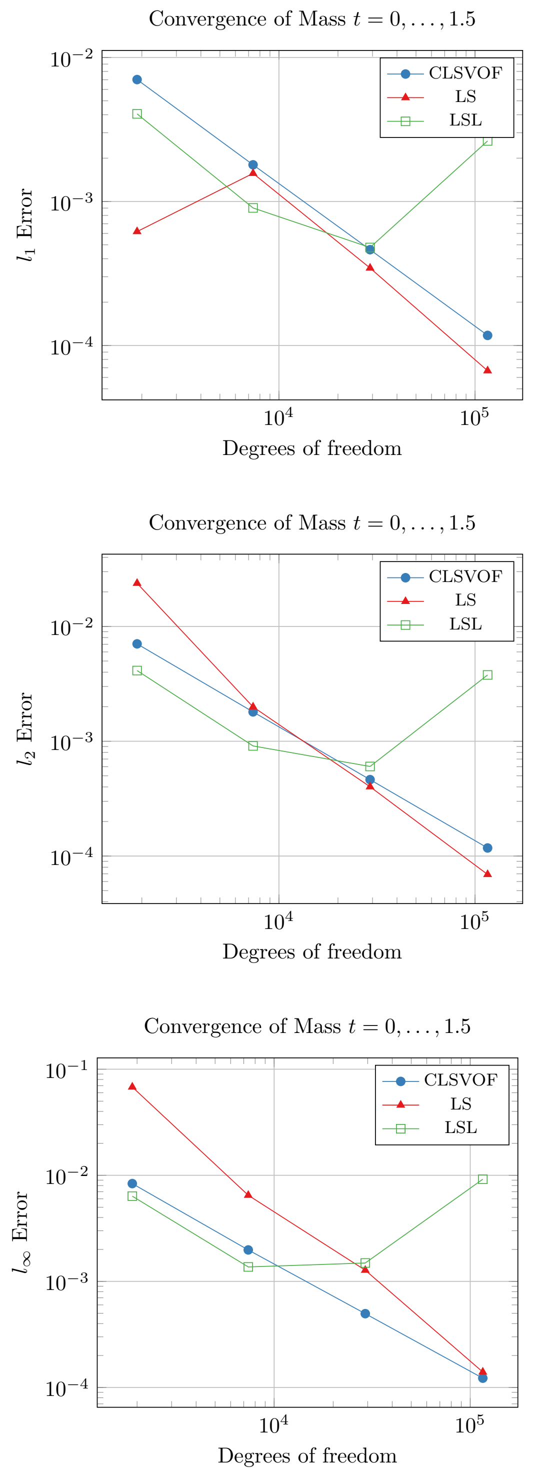
Level	e_1^l	ρ_1^l	e_2^l	ρ_2^l	e_∞^l	ρ_∞^l
			CLSVOF			
1	7.025_{-3}	–	7.053_{-3}	–	8.352_{-3}	–
2	1.798_{-3}	1.966	1.802_{-3}	1.969	1.979_{-3}	2.077
3	4.627_{-4}	1.959	4.632_{-4}	1.960	4.961_{-4}	1.996
4	1.175_{-4}	1.977	1.176_{-4}	1.978	1.223_{-4}	2.021
			LS			
1	6.183_{-4}	–	2.377_{-2}	–	6.799_{-2}	–
2	1.568_{-3}	−1.342	6.158_{-2}	3.576	6.483_{-3}	3.391
3	3.451_{-4}	2.184	4.014_{-4}	2.312	1.274_{-3}	2.348
4	6.678_{-5}	2.370	6.913_{-5}	2.538	1.398_{-4}	3.187
			LSL			
1	4.058_{-3}	–	4.129_{-3}	–	6.360_{-3}	–
2	9.020_{-4}	2.169	9.105_{-4}	2.181	1.368_{-3}	2.217
3	4.791_{-4}	0.913	6.032_{-4}	0.594	1.490_{-3}	−0.124
4	2.633_{-3}	−2.458	3.780_{-3}	−2.648	9.184_{-3}	−2.623

(b) $t = 0, \dots, 1.5$; see Fig. 6.12(b) for a convergence plot.

Table 6.9.: Table of mass convergence for the two-dimensional rising bubble benchmark with the LSL, the LS and the CLSVOF method. Here, ρ and e are defined by equations (6.11)–(6.13).

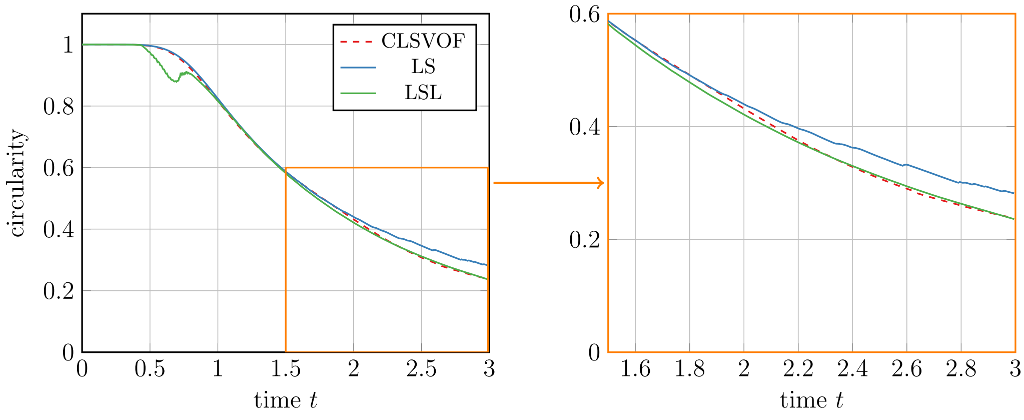


(a) Error norms averaged over $t = 0, \dots, 3.0$.

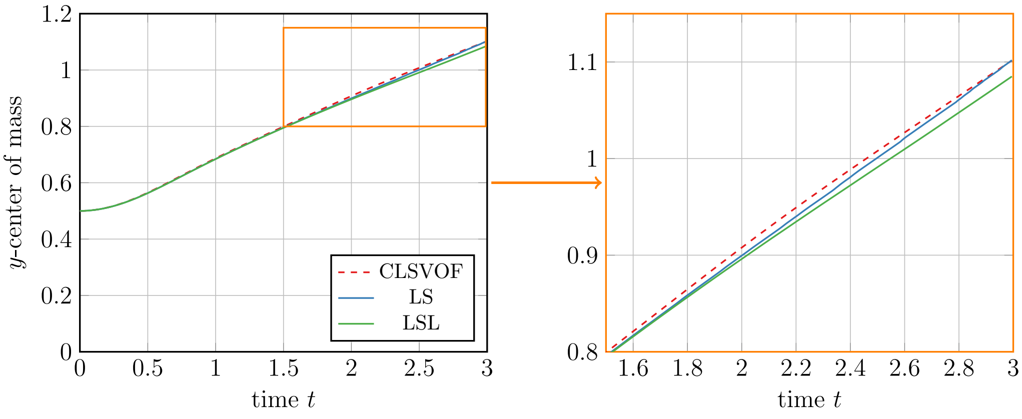


(b) Error norms averaged over $t = 0, \dots, 1.5$.

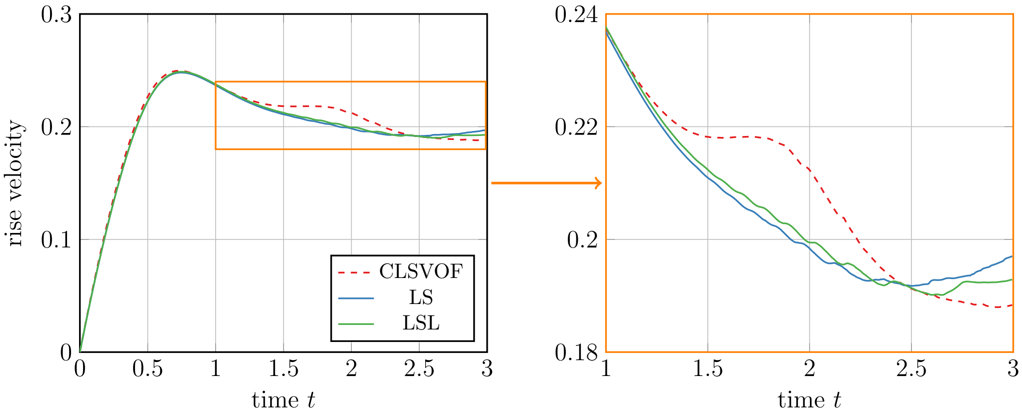
Figure 6.12.: Convergence history for the 2D rising bubble with the the CLSVOF method (blue), the LS method (red) and the LSL method (green); compare Tab. 6.9(a) and 6.9(b).



(a) Circularity over time.



(b) Center of mass in y-direction over time.



(c) Rise velocity in y-direction over time.

Figure 6.13.: Benchmark quantities computed over time for the 2D rising bubble with the CLSVOF (dashed red), the LS (blue) and the LSL method (green).

	LS	LSL	CLSVOF
$61 \times 121 \times 5$	49 min	02 h 19 min	55 min
$121 \times 241 \times 5$	14 h	61 h	9 h

Table 6.10.: Computing time for the two-dimensional rising bubble on one 32-core node of the *Siebengebirge* cluster.

	CLSVOF		LSL		LS	
routine	[%]	[min]	[%]	[min]	[%]	[min]
Poisson solver	29.7	16.44	7.8	12.57	25.3	12.64
reinitialization	20.4	11.30	77.7	125.52	37.5	18.70
parallel communication	44.2	24.47	12.1	19.62	30.0	14.77

Table 6.11.: Computing time of three different routines for the CLSVOF, the LSL and the LS method on one 32-core node of the *Siebengebirge* cluster.

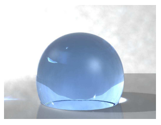
the end of the computation. Note that this behavior is also closer to that predicted by the 2D solvers in [HTK$^+$09].

d) Computing time

To complete our comparison of the three methods, we measure their computing time on a grid with $61 \times 121 \times 5$ and with $121 \times 241 \times 5$ cells (Tab. 6.10). All simulations are conducted in parallel on one 32-core node of the *Siebengebirge* cluster. On the coarse grid, the LS method is the most efficient with 49 min, followed by the CLSVOF method with 55 min and by the LSL method with 2 h19 min. Note that the LS method is faster on this grid, since about 20% of mass is lost. Then, less interface cells have to be treated and the reinitialization becomes faster. As expected, the LSL method is much more expensive than the LS method and about three times slower in this experiment. On the fine grid, the CLSVOF methods outperforms the LS method by a factor of 1.5 which is what we anticipated. Here, the LSL method is four times slower than the LS method.

In a next step, we measure the computing time of the most expensive routines in our flow solver for all three methods on the coarse grid (Tab. 6.11). All methods require between 15 and 25 minutes for parallel communication. Here, the CLSVOF method needs the most resources with 24.47 min. Note that there is certainly room for improvement in the parallelization of this method, which has to be tackled in further code refinements; compare Section 5.8. In addition, the Poisson solver has the most difficulties with the right hand side computed by the CLSVOF method, which takes about 16 min in contrast to 12 min with the LSL and the LS method. Both of these methods certainly tend to produce smoother results than the CLSVOF method (cf. bubble at $t = 3.0$ in Fig. 6.10).

For the reinitialization, the CLSVOF method only needs half the amount of CPU time compared to the LS method due to the very expensive components of the LS reinitialization, such as the fifth order WENO scheme in space and the third-order Runge-Kutta scheme in time, which yield an accurate but computationally expensive solution. As anticipated, this procedure is also the bottleneck for the LSL method. If we chose a five times larger time step for the reinitialization as in [SF99], the computing time would go down from



126 min to 25 min, which is still a little more time than what the LS method needs for the reinitialization. However, we already have stability problems with the LSL method, and a larger time-step in the reinitialization would render the whole computation even less stable.

e) Comparison to [HTK$^+$09]

Finally, we turn to a comparison of the benchmark quantities computed by the CLSVOF method with those presented in [HTK$^+$09]. These results are shown in Figure 6.14. Mind that this is now a computation with the same boundary conditions as in [HTK$^+$09] on the grid with mesh size $h = 1/121$. First of all, we note that the three solvers in [HTK$^+$09], although they show remarkable coincidence for the first test case, predict different results for test case 2 since not all of them are able to handle break-up automatically – which is what happens in the 2D case. In comparison, our quasi-2D CLSVOF method predicts a much steeper increase of the circularity. The center of mass results are very close to those produced by the 2D solvers. Then for the rise velocity, the CLSVOF method underestimates the renewed maximum as well as the terminal velocity. To summarize, the CLSVOF predicts a similar behavior of the curves with a slight downward shift from a certain point in time onwards for all benchmark quantities. Probably, this anticipated discrepancy stems from our quasi-2D solution of a purely 2D problem.

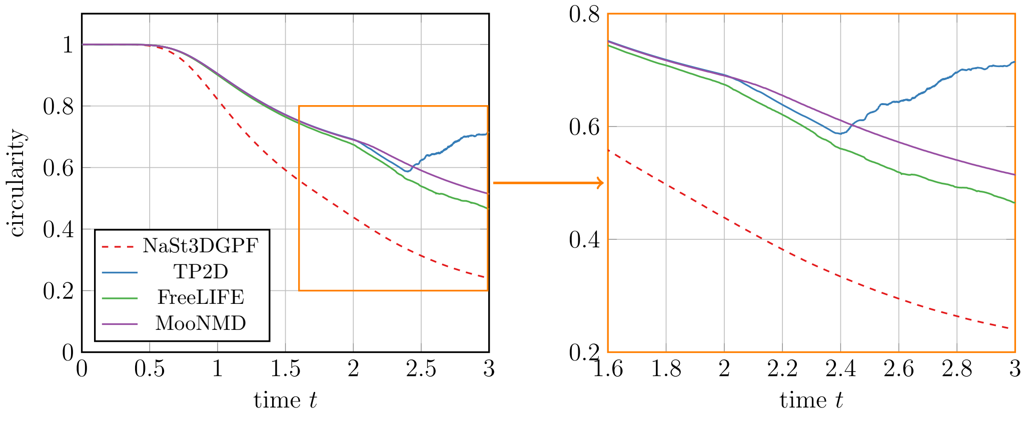
Conclusions

Several interesting conclusions can be drawn from the 2D rising bubble benchmark, which now include the whole flow solver.

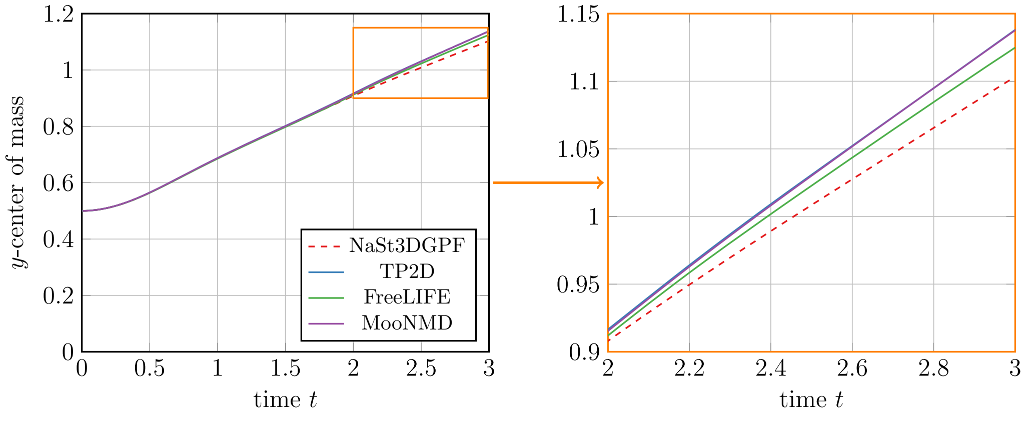
Both the LS and the CLSVOF method are able to handle real world problems with very large density jumps. Both methods predict similar results for several benchmark quantities and the shape of the bubble. However, the LS method requires a much larger amount of grid cells than the CLSVOF method due to the severe mass loss on coarse grids. Again, the final amount of mass conserved with the CLSVOF method on the coarsest grid is close that of the LS method on the finest grid. For the first half of the simulation, both methods show quadratic convergence with similar accuracy in the relative error norms. Averaged over the whole computing time, only the CLSVOF method shows stable second order convergence in all norms.

For this test case, the LSL method displays another unfortunate drawback. As mentioned in [Her05, Kec98], the LSL method is prone to introduce noise in the curvature, which in our case results in very prominent oscillations of the interface on the finest grid. Thus, a reasonable order of mass convergence cannot be established. Additionally, the LSL method is very slow and will probably remain slower than both the LS and the CLSVOF method, even if we increase the time-step as proposed in [SF99]. However, the results on level 1–3 do not reflect these difficulties: Here, the LSL method fulfills its promise of enhanced mass conservation.

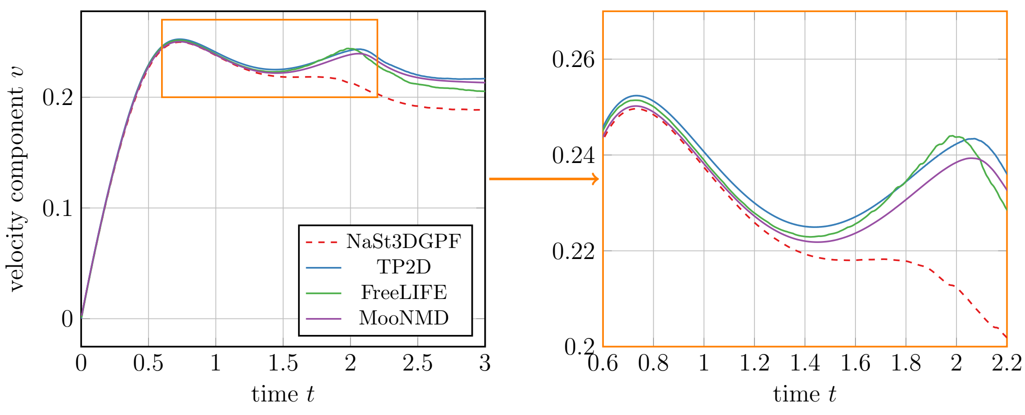
There are several open questions that can only be answered by further investigation. If we average the error norms over the whole computing time, the convergence of the LS method deteriorates. We assume that the formation of filaments combined with the large density jump over the interface lowers the accuracy of the LS function. Since the CLSVOF method is not affected, this problem has to mainly stem from the diffusion and mass loss induced



(a) Circularity over time.

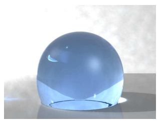


(b) Center of mass in y-direction over time.



(c) Rise velocity in y-direction over time.

Figure 6.14.: Comparison of benchmark quantities computed with the CLSVOF method with those computed by the flow solvers presented in [HTK+09].



	DROPS	NaSt3D	OpenFOAM
developer	IGPM, RWTH Aachen	INS, University of Bonn	open source
space discretization	XFEM	finite differences	finite volumes
interface capturing	LS	LSL and CLSVOF	VOF
time discretization	implicit θ-scheme	Adams-Bashforth 2nd	implicit Euler

Table 6.12.: Comparison of the code features and schemes used for the 3D benchmark.

by the solution of the reinitialization equation. Further investigation of these difficulties might lead to a range of numerical problems for which the LS method is well-equipped, as well as a range of problems which should be avoided with the LS method.

As already mentioned, we could tune our quasi-2D setup with the easier test case 1 presented in [HTK+09], so that our results are as close as possible to the (very closely agreeing) 2D ones. Then our results might also improve for test case 2 as compared to [HTK+09].

All in all, the LS method always requires a sufficiently fine grid and performs best for simple flow problems with smooth distance functions; the LSL method has to be used with very special care when computing curvature dominated problems, and the implemented CLSVOF method is excellently equipped to deal with even the most challenging kinds of two-phase flow problems. Compared to the other methods, the CLSVOF method shows the best convergence behavior, maintains the most mass and is least computationally expensive.

6.3.2. Three-dimensional

In this subsection, we consider a three-dimensional rising bubble for two benchmark suites, which compare to test case 1 and 2 studied in [HTK+09]. The first test deals with a bubble in a fluid with small density and viscosity ratios of $\rho_1/\rho_2 = \mu_1/\mu_2 = 10$. According to the classification by Clift et al. [CGW78], test case 1 can be assigned to the ellipsoidal regime for which we have to expect moderate shape deformation of the bubble. For the second test case, the large density and viscosity ratios of $\rho_1/\rho_2 = 1000$ and $\mu_1/\mu_2 = 100$ are roughly the same as for an air bubble in water and correspond to the 2D setup studied in the previous subsection. Furthermore, a break-up of the bubble is possible due to the low surface tension value. According to the classification by Clift et al. [CGW78], test case 2 lies somewhere between the skirted and dimpled ellipsoidal-cap regimes.

The work presented in this subsection is not solely due to the author of this thesis, and it was in large parts already published in [AEG+14]. Note that the simulation runs, the visualization and the presentation of results were a shared effort of the authors belonging to two different scientific institutes in Germany. Thus, the open source flow solver OpenFOAM [Ope13] is normally employed at the Institute for Numerical Simulation for the simulation of global atmospheric dynamics [Ade14], and the flow solver DROPS [DRO08, RFG+] is developed at the Chair of Numerical Mathematics and the Institute of Scientific Computing at the RWTH Aachen. NaSt3DGPF is a joint code project developed at the Institute for Numerical Simulation for a variety of two-phase flow applications [BG13, Cro10, GR14, ZG11]. All codes adopt different numerical techniques, and we list the main features of each code in Table 6.12. In [AEG+14], simulations are performed with our flow solver NaSt3DGPF employing the LSL method implemented by this author. Here, we also present the results with the CLSVOF method as promised in [AEG+14].

Note that the results presented in this Subsection are still work in progress. Thus, there are many aspects of the numerical experiment which have yet to be investigated:

- In [AEG+14], we use the LSL method because the LS method was prone to mass loss and the CLSVOF method was not yet implemented. In addition, results with the LS method as well as with the LSG always tended to smooth the interesting structures at the rear end of the rising bubble. In this thesis, we have seen that the LSL method must be used with care since oscillations in the curvature can occur. Although it compared very well with the two other flow solvers in [AEG+14], we rather recommend the use of the CLSVOF method only.
- In our simulations with NaSt3DGPF, the velocity in z-direction which should be zero is $\mathcal{O}(10^{-5})$ for test case 2. This might be due to the use of the 2nd order SMART scheme, which showed reduced numerical diffusion compared to the 5th order WENO scheme. Here, the choice of the convective terms' discretization might have a large influence which has to be further investigated.
- The results presented with the CLSVOF method are also not yet final. For example, for testing purposes, we use a smaller interface thickness than in [AEG+14] for both cases; compare Table 6.14. Thus, for a final comparison with the LSL method, a simulation run where both methods use the same thickness would be needed. Furthermore, due to the interface reconstruction and the smaller thickness parameter, the CLSVOF method is acutely sensitive to the non-zero velocity in z-direction which results in a slight asymmetry for test case 2.

However, the 3D rising bubble is a very expensive computational task, so that all of these interesting considerations have to be delayed to our future research.

Note that in the following, if we speak of our flow solver NaSt3DGPF in general, this entails both the LSL and the CLSVOF method.

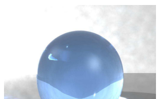
Definition of the numerical experiment

Consider a cuboid tank $\Omega = [0, 1] \times [0, 2] \times [0, 1]$ and a bubble $\Omega_2 = \Omega_2(t) \subset \Omega$ which is lighter than the surrounding fluid $\Omega_1 = \Omega \setminus \Omega_2$ (cf. Figure 6.15). The final time T is prescribed as $T = 3$. We assume no-slip boundary conditions, i. e., $\boldsymbol{u} = 0$ on $\partial\Omega$. The initial condition for the Navier-Stokes equations is given by $\boldsymbol{u}(0) = 0$. The initial bubble is assumed to be spherical with radius $r = 0.25$ and center $\boldsymbol{x}_c = (0.5, 0.5, 0.5)^{\mathsf{T}}$, i. e., fluid phase 2 is initially given by

$$\Omega_2(0) = \{\boldsymbol{x} \in \Omega \mid \|\boldsymbol{x} - \boldsymbol{x}_c\| \leq r\}.$$

Due to buoyancy effects, the bubble will start to rise and change its shape.

The two test cases are defined according to the material properties given in Table 6.13. These parameters are taken from the 2D two-phase flow benchmark proposed in [HTK+09]. Note again, that instead of having a mixture of slip and no-slip boundary conditions as in [HTK+09], we apply the no-slip boundary condition on every wall. For test case 1, the



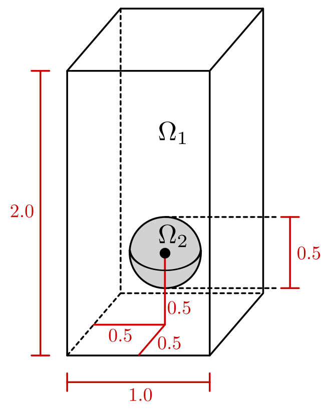


Figure 6.15.: Initial configuration in both test cases.

	test case 1	test case 2
ρ_l	1000	1000
μ_l	10	10
ρ_g	100	1
μ_g	1	0.1
σ	24.5	1.96

Table 6.13.: Material properties for test case 1 and 2.

final time:	$T = 3$
flow domain:	$\Omega = [0,1] \times [0,2] \times [0,1]$
sphere radius:	0.25
sphere center:	$(0.5, 0.5, 0.5)$
material parameters:	see Table 6.13
body force:	$\boldsymbol{g} = (0.0, -0.98, 0.0)$
convective terms:	2nd-order SMART
Poisson solver (LSL):	AMG-preconditioned BiCGStab (residual tolerance 1×10^{-16})
Poisson solver (CLSVOF):	Jacobi-preconditioned CG solver (residual tolerance 1×10^{-15})
interface thickness (CLSVOF):	test case 1: $\epsilon = 1.75h$ and test case 2: $\epsilon = 1.6h$
interface thickness (LSL):	$\epsilon = 1.9h$ for test case 1 and 2
grid resolution:	$121 \times 241 \times 121$
employed methods:	LSL, CLSVOF
boundary conditions:	no-slip

Table 6.14.: Simulation parameters for the three-dimensional rising bubble for NaSt3DGPF.

density and viscosity ratio $\rho_1/\rho_2 = \mu_1/\mu_2 = 10$ is rather moderate, whereas for the second test case, we have larger ratios $\rho_1/\rho_2 = 1000$ and $\mu_1/\mu_2 = 100$ which are roughly the same as for an air bubble in water.

Our simulation parameters for NaSt3DGPF are summarized in Table 6.14. In both test cases, NaSt3DGPF employs an equidistant finite difference grid with $121 \times 241 \times 121$ grid cells. For the capturing of the interface, we use the LSL method on the one hand and the CLSVOF method on the other hand. The pressure Poisson equation is solved by an AMG-preconditioned BiCGStab solver for the LSL method and by a simple preconditioned conjugate gradient method for the CLSVOF method. The convective terms in the momentum equations are treated with a 2nd-order SMART scheme which showed reduced numerical diffusion in previous studies compared to the also implemented 5th-order WENO scheme. For the simulation parameters of DROPS and OpenFOAM, we refer the reader to [AEG+14].

As in the 2D case, we track our three time-dependent quantities of interest which we now define for the 3D rising bubble. The barycenter $\bar{\boldsymbol{x}}$ of the bubble is given by

$$\bar{\boldsymbol{x}}(t) := |\Omega_2|^{-1} \int_{\Omega_2(t)} \boldsymbol{x}\, d\boldsymbol{x},$$

with $|\Omega_2| := \operatorname{meas}_3 \Omega_2 = \int_{\Omega_2} 1\, d\boldsymbol{x}$ the volume of the bubble. Its rise velocity $\bar{\boldsymbol{u}}$ is defined as

$$\bar{\boldsymbol{u}}(t) = |\Omega_2|^{-1} \int_{\Omega_2(t)} \boldsymbol{u}\, d\boldsymbol{x}.$$

And, as in Subsection 6.2.2, we measure the sphericity Ψ by

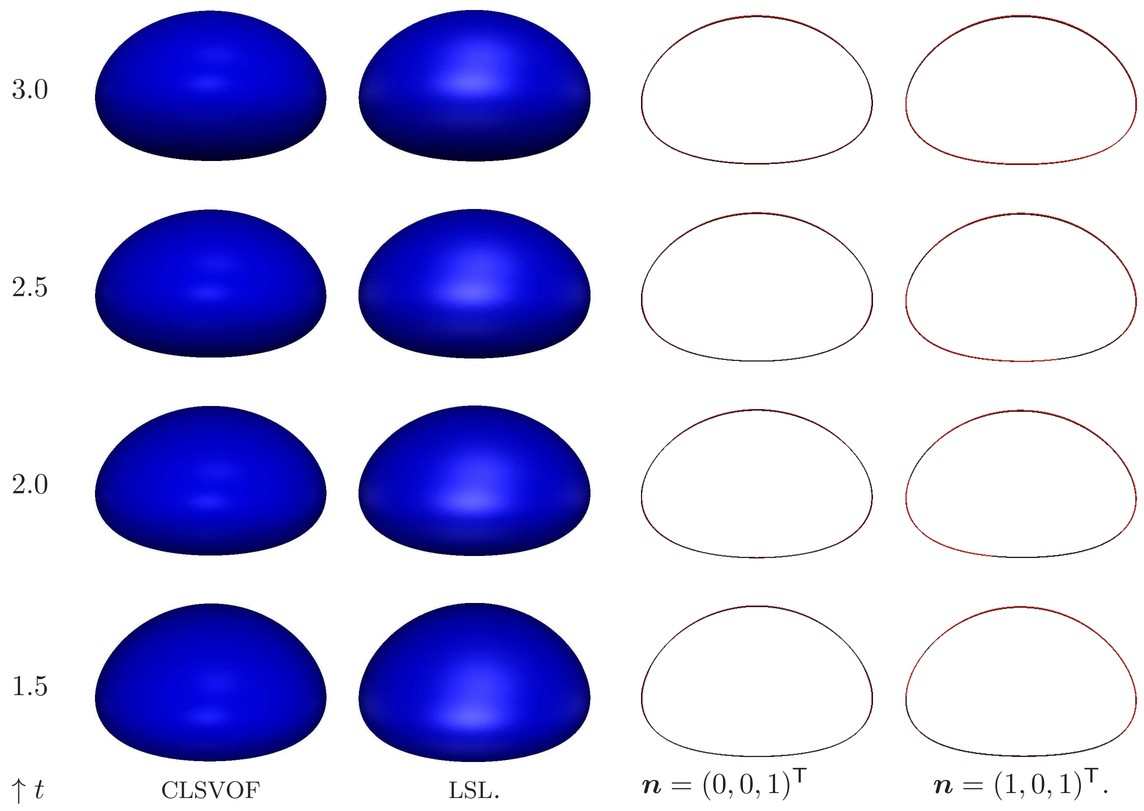
$$\Psi(t) = |\Gamma(t)|^{-1}\, \pi^{1/3} (6|\Omega_2(t)|)^{2/3}.$$

Again, this is the surface area of a sphere with volume $|\Omega_2(t)|$ divided by the surface area $|\Gamma(t)| := \operatorname{meas}_2 \Gamma(t) = \int_{\Gamma(t)} 1\, ds$ of the bubble. Thus, Ψ is equal to 1 if Ω_2 is a sphere and decreases the more the bubble is flattened. In addition we measure the diameter $\mathbf{d}(t) = (d_1(t), d_2(t), d_3(t))^{\mathsf{T}}$, which denotes the bubble's maximum extension in each coordinate direction, i.e.

$$d_i(t) = \max_{\mathbf{x},\mathbf{y} \in \Omega_2(t)} |x_i - y_i|, \qquad i = 1, 2, 3.$$

Expected behavior for both test cases

Since the mesh width is not as fine as in Subsection 6.3.1, we do not anticipate visible oscillations of the LSL method. Therefore, and from the numerical experiments in the previous sections, we do not expect very large differences between the results by the CLSVOF and the LSL method. The largest difference between both methods can be expected for test case 2 due to its stronger deformations. Here, similar to the previous Subsection, we anticipate higher curvature and a less smooth bubble shape at the bottom of the bubble with the CLSVOF method. Furthermore, since our velocity field is not completely symmetric, the CLSVOF method might produce some slightly asymmetric results as well: The interface reconstruction is only first order accurate and prone to accumulate errors over time. In addition, the lower interface thickness also renders the bubble less smooth and susceptible


3.0
2.5
2.0
1.5
↑ t
CLSVOF
LSL.
$\boldsymbol{n} = (0, 0, 1)^\mathsf{T}$
$\boldsymbol{n} = (1, 0, 1)^\mathsf{T}$.

(a) The 3D zero level-set.

(b) The zero level-set on planes through the bubble's center; CLSVOF (black) and LSL (red).

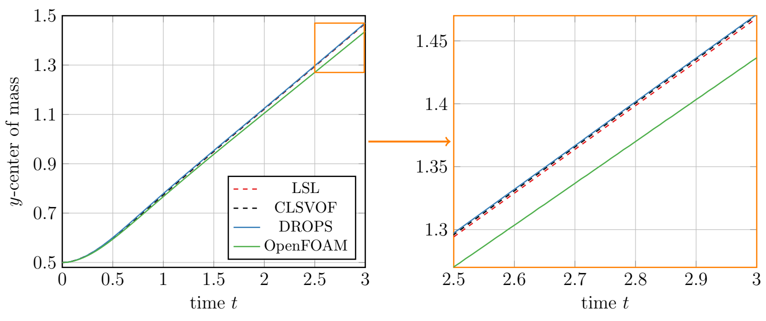
Figure 6.16.: Test case 1: Droplet's evolution over time with the LSL and the CLSVOF method.

to interference.

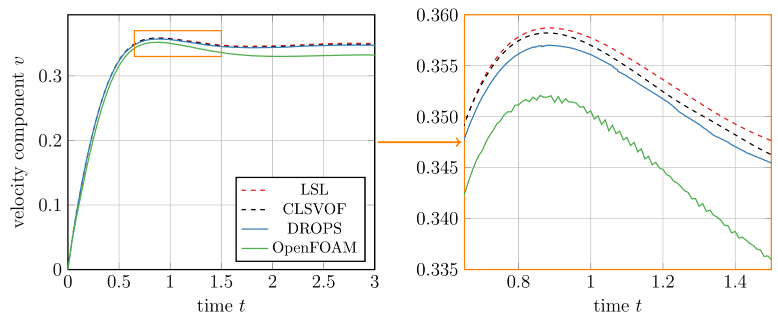
Discussion of results for test case 1

Figure 6.16(a) illustrates the bubble's evolution over time in test case 1 with the LSL and the CLSVOF method. As illustrated, the spherical bubble is extended in directions x and z perpendicular to the flow and compressed in the flow direction y. At $t \approx 2.0$, the bubble reaches a stable ellipsoidal shape up to the final state of the simulation. For a better comparison of the actual bubble shape, we also show the two-dimensional contour on the x/y-plane with normal $\boldsymbol{n} = (0, 0, 1)^\mathsf{T}$ and on a plane with normal $\boldsymbol{n} = (1, 0, 1)^\mathsf{T}$ through the bubble's center (Fig. 6.16(b)). Here, the contour lines of the CLSVOF method (black) and the LSL method (red) are indistinguishable from each other.

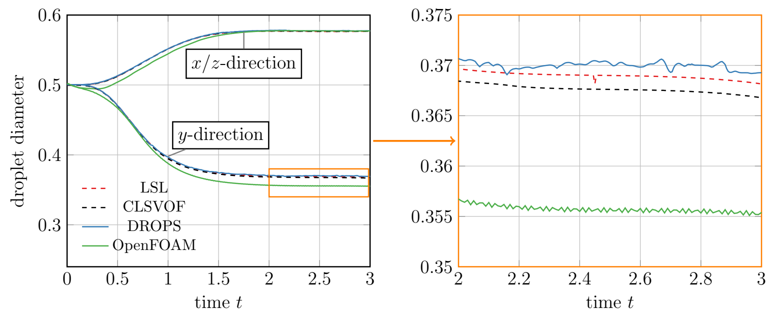
Figure 6.17(a) shows the y-component of the bubble's barycenter position $\bar{\boldsymbol{x}}$ over time. In the early state of the simulation up to $t \approx 1$, the position $\bar{\boldsymbol{x}}$ is roughly identical in all simulations. Then, the vertical position of the bubble in OpenFOAM increases slower than in DROPS and NaSt3DGPF. Here, both the LSL and the CLSVOF method closely predict the same position of the bubble. Of course, this is also what we deduce from the identical



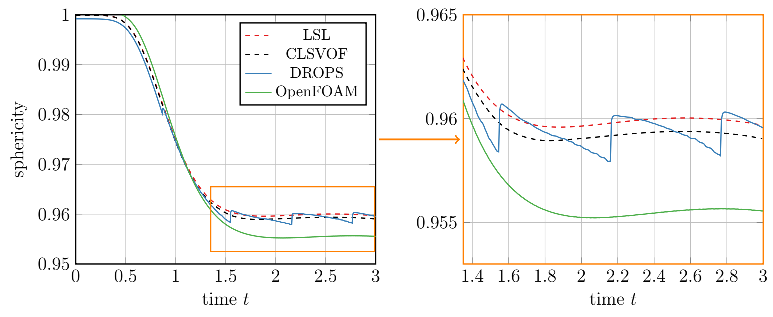
(a) Center of mass $\bar{\boldsymbol{x}}$.



(b) Rise velocity component v.



(c) Droplet diameter **d**.



(d) Sphericity Ψ.

Figure 6.17.: Quantities of interest for test case 1.

contours of our two methods in Figure 6.16.

The bubble's rise velocity is visualized in Figure 6.17(b). In the early state of the simulation, the velocity component v of $\bar{\boldsymbol{u}}$ increases from its initial value zero up to a maximum value in the order of 0.35–0.36 at about $t \approx 0.9$. The maximum velocity is roughly 0.357 in DROPS, 0.358 in NaSt3DGPF and 0.352 in OpenFOAM. The rise velocity in DROPS then decreases to $v \approx 0.347$, to $v \approx 0.35$ in NaSt3DGPF and to $v \approx 0.33$ in OpenFOAM. At all times in NaSt3DGPF, the CLSVOF method predicts a slightly lower rise velocity than the LSL method. However, this subtle difference neither affects the center of mass in Figure 6.17(a) nor the position of the contours in Figure 6.16.

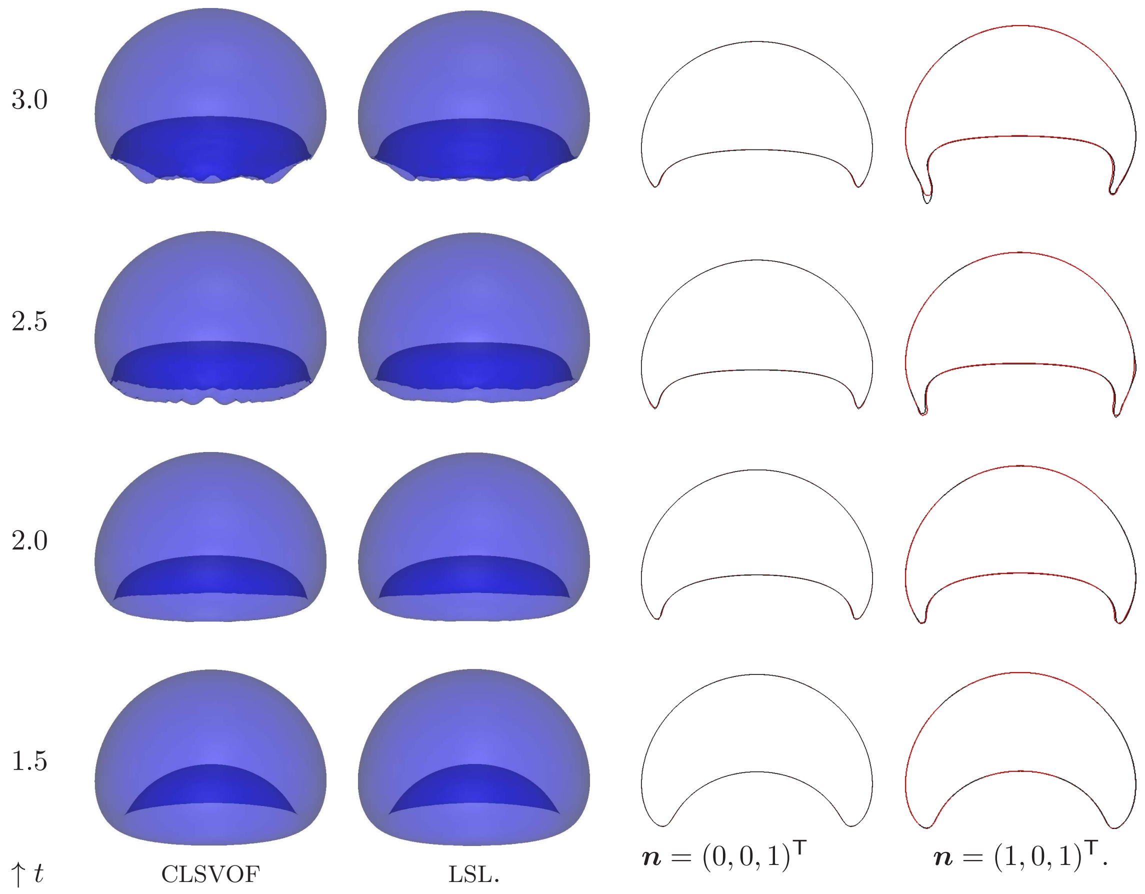
We now concentrate on an analysis of the bubble's shape. For this purpose, Figure 6.17(c) shows the different components of the bubble's diameter $\mathbf{d}$ over time. During the simulation, d_1 and d_3 increase up to a final value of about 0.58 in all four simulations. However, for the component d_2 in y-direction, the OpenFOAM solver predicts a lower diameter. Here, the extension is 0.37 for DROPS and NaSt3DGPF and about 0.355 for OpenFOAM as indicated on the right hand side of Figure 6.17(c). These results suggest that the bubble in the DROPS a nd NaSt3DGPF simulations has a slightly more spherical shape than in the case of OpenFOAM. Note that both the LSL and the CLSVOF method in NaSt3DGPF predict a similar extension $\mathbf{d}$ in all directions. Only in the zoomed in part on the right hand side of Figure 6.17(c), we see that the CLSVOF method is slightly below the results of the LSL method and DROPS for d_2.

A measure of the bubble's similarity with its initial spherical form is given by the sphericity Ψ. Figure 6.17(d) displays Ψ over time. Due to the bubble's compression in vertical direction, the sphericity decreases from 1 to about 0.96 for DROPS and NaSt3DGPF and to $\Psi \approx 0.955$ for OpenFOAM (cf. right hand side of Figure 6.17(d)). The slightly lower sphericity in OpenFOAM primarily results from a stronger compression in vertical direction as indicated in Figure 6.17(c). DROPS and NaSt3DGPF predict similar sphericity values at the end of the simulation but with non-smooth jumps in the case of DROPS. This results from the reinitialization of the LS function in DROPS which is performed only when an error threshold of the distance function at the surface is violated. The less frequent reinitialization slightly affects Ψ which is noticeable in Figure 6.17(d) but not the other quantities of interest such as $\bar{\boldsymbol{x}}$ and the rise velocity $\bar{\boldsymbol{u}}$. Again, the results with the LSL and the CLSVOF method agree very well. Here, the sphericity curve of the CLSVOF method lies only very slightly below that of the LSL method and both are in good agreement with the oscillative DROPS results. To summarize, all sphericity results in test case 1 are in high agreement as the differences at steady state are below 0.5% at most.

Discussion of results for test case 2

Figure 6.18(a) shows snapshots of the bubble's temporal evolution in test case 2 for the LSL and the CLSVOF method. The difference between both methods is most noticeable on the bottom edge of the bubble where the bubble's curvature is largest, while the top edge of all bubbles behaves similarly. At the bottom edge of the 3D plot, we observe a stronger curved bottom for the CLSVOF method from $t = 2.5$ onwards. Furthermore, the distance between the bottom edge and the filaments also seems larger than for the LSL method.

Although the bubble shapes in Figure 6.18(a) differ at the bottom edge, this is not visible


3.0
2.5
2.0
1.5
↑ t
CLSVOF
LSL.
$\boldsymbol{n} = (0, 0, 1)^\mathsf{T}$
$\boldsymbol{n} = (1, 0, 1)^\mathsf{T}$.

(a) The 3D zero level-set.

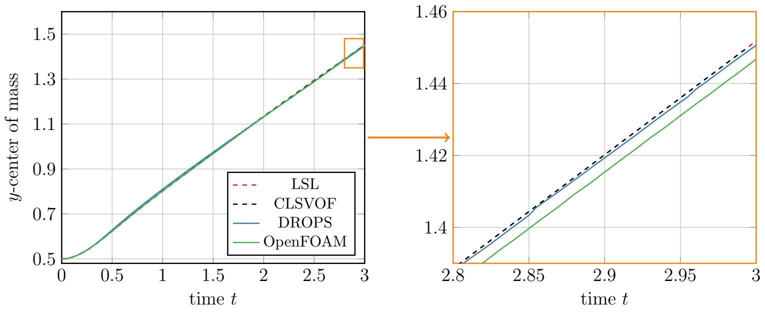
(b) The zero level-set on planes through the bubble's center; CLSVOF (black) and LSL (red).

Figure 6.18.: Test case 2: Droplet's evolution over time with the LSL and the CLSVOF method.

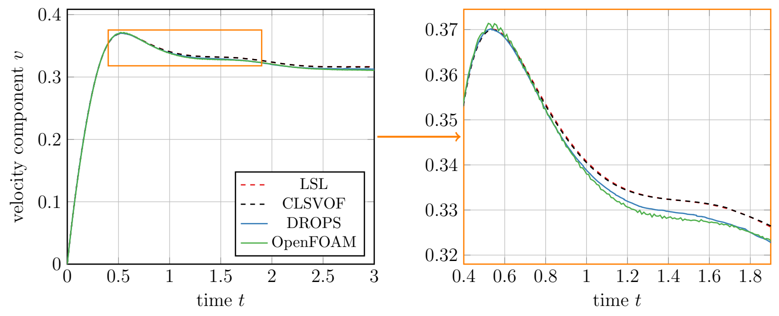
in the two-dimensional contour plot on the plane with normal $\boldsymbol{n} = (0, 0, 1)^\mathsf{T}$ through the bubble's center as shown in Figure 6.18(b). Here, slightly larger differences can be seen on a further plane with normal $\boldsymbol{n} = (1, 0, 1)^\mathsf{T}$. The contour lines on this plane primarily differ in the shape of the bubble's filaments from $t \approx 2.5$ onwards, while both the top edge and bottom edge of the bubble remain similar. Unfortunately, the drop computed with the CLSVOF method exhibits a slight asymmetry, which is what we expected.

Note, that the lower surface tension allows for a stronger bubble deformation as in test case 1, and for a bubble in between the skirted and the dimpled ellipsoidal-cap regimes, satellite droplets might occur. However, no flow solver predicts satellite droplets up to the final time of $t = 3.0$. However, ongoing simulations with OpenFOAM predict 8 satellite droplets that occur pairwise in the direction of each of the lateral four boundary faces at $t \approx 3.9$.

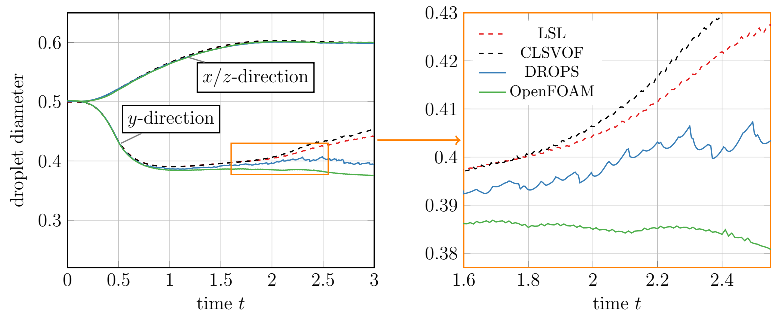
The bubble's position over time is visualized in Figure 6.19(a). All three flow solvers show a high agreement of the bubble's barycenter position for the whole simulation. Despite the



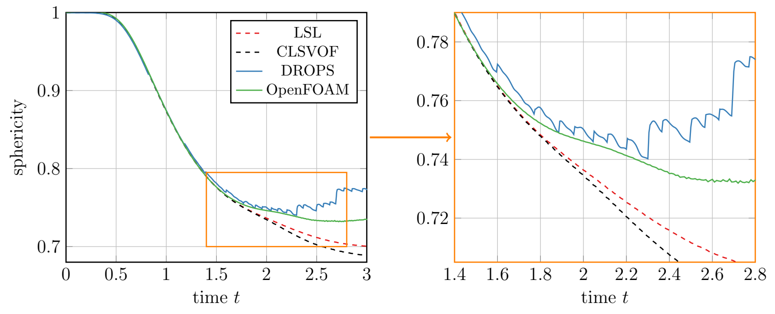
(a) Center of mass $\bar{\boldsymbol{x}}$.



(b) Rise velocity component v.



(c) Droplet diameter $\mathbf{d}$.



(d) Sphericity Ψ.

Figure 6.19.: Quantities of interest for test case 2.

differences of the LSL and the CLSVOF method in the contour plot in Figure 6.18(b), both agree perfectly on the bubble's barycenter position, so that $\bar{\boldsymbol{x}}$ seems relatively insensitive with respect to minor variances in the bubble's shape.

The rise velocity plot of the bubble is also in high agreement for all considered flow solvers and especially for the LSL and the CLSVOF method (Fig. 6.19(b)). All solvers obtain maximum velocities in the order of 0.37 at $t \approx 0.54$. The bubble's velocity decreases with ongoing time so that a final velocity of $v \approx 0.31$ at $t = 3.0$ is obtained. Here, the results of DROPS and OpenFOAM are closer than the final results of NaSt3DGPF with the LSL and the CLSVOF method.

In Figure 6.19(c), we plot the diameter of the bubble for a better analysis of the bubble's shape. The bubble diameters d_1 and d_3 are in high agreement for all three flow solvers similar to test case 1. There are minor differences in y-direction from about $t \approx 1.6$ onwards. The strong bubble deformation at the outer edges in NaSt3DGPF then leads to a new rise in the bubble diameter d_2 from its minimum value of about 0.39 up to 0.44 in the final state of the simulation. Here, the even stronger deformation of the bubble in y-direction with the CLSVOF method is confirmed, since it predicts the highest values for d_2 of all flow solvers. DROPS and OpenFOAM do not predict a similar new rise in d_2. OpenFOAM shows the largest bubble compression in vertical direction with a minimum value for d_2 of about 0.37.

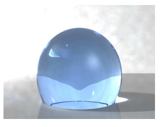
Finally, we compare the sphericity Ψ over time in Figure 6.19(d). The results in the early simulation state are similar up to $t \approx 1.6$. DROPS then predicts a more spherical bubble shape which leads to an increase in Ψ at about $t \approx 2.7$. Due to the strong deformation of the bubble in NaSt3DGPF, the flow solver computes an ongoing decrease of Ψ until it reaches a final value of about 0.7 with the LSL method and 0.69 with the CLSVOF method. The result for OpenFOAM lies in between DROPS and NaSt3DGPF with a final value of about 0.74.

Conclusions for both test cases

We conducted numerical studies for two different test cases of three-dimensional rising bubbles. While the bubble in the first test case experiences only slight deformations, it undergoes strong deformations in test case 2.

In general, DROPS, the LSL and the CLSVOF method show a high agreement in test case 1 for the defined quantities of interest and for the contour plots. The situation differs for the OpenFOAM results. Here, the general bubble shape is similar to the results of the other flow solvers, with differences in the sphericity of 0.5% at most (cf. Fig. 6.17(d)). However, in contrast to DROPS and NaSt3DGPF, OpenFOAM obtains a lower rise velocity in the late state of the simulation so that the bubble's position is slightly shifted in the vertical direction. To summarize test case 1, the general shape of the bubble is in high agreement for all flow solvers while its position and velocity are controversial.

Interestingly, the quantities of interest that were controversial in test case 1 are in relatively high agreement in test case 2 and vice versa. The rise velocity and the bubble's barycenter position resemble each other closely for the complete simulation period. Since the bubble undergoes a strong deformation, it is not surprising that the results for the bubble's diameter and sphericity differ at the end of the simulation. Similar differences oc-



curred in the 2D benchmark experiment by Hysing et al. [HTK$^+$09] for test case 2 described in the previous Subsection. Thus, the actual bubble shape in flow fields with high density and viscosity ratios is a matter of ongoing research, and since the correct bubble shape for the second test case is not even clear in 2D computations, the differences in 3D are to be expected. In general, the results of DROPS and OpenFOAM show a higher level of agreement than the results of NaSt3DGPF. In NaSt3DGPF, both the LSL and the CLSVOF method give very similar results. The bubble exhibits an even higher curvature with the CLSVOF method, so that the difference to DROPS and OpenFOAM becomes even more prominent.

In our article [AEG$^+$14], we assumed that the differences in both test cases could be attributed to the different interface capturing approaches on the one hand and to the different volume correction methods on the other hand. In DROPS, a global volume correction similar to ours is applied and while both NaSt3DGPF and DROPS are based on the LS method, OpenFOAM makes use of the VOF method. Unfortunately, with the CLSVOF method we can neither support nor disprove these assumptions since the differences between the LSL and the CLSVOF method are too small. However, this small difference might indicate that the problems lie somewhere else altogether.

Probably, for a simulation as sensitive as test case 2, there will never be complete agreement. For this test case, it is very difficult to obtain grid independent results. However, we have a chance for test case 1, where we have to try to diminish the differences predicted by the flow solvers, which in turn might render the flow solvers better and more accurate. Thus, from our experience and the numerical results in the previous sections, methods which are stable and very smooth exhibit in many cases simply too much diffusion. For example, all numerical results with the LS and CLSVOF method strongly depend on the interface thickness ϵ of the CSF approach. The higher the thickness, the smoother the result, the larger the error due to numerical regularization with the smoothed Dirac delta function. Thus, the use of a ghost fluid method might simply be more sensible, and one source of possible difference and unphysical smoothing would be taken care of. This method is already partly implemented in NaSt3DGPF and has to be extended to handle the pressure jump. Then, we plan to employ the ghost-fluid method for the 3D rising bubble benchmark.

Furthermore, for our CLSVOF method, we need to the take the same interface thickness as the LSL method and find the source of asymmetry in the velocity field. This asymmetry might also affect the LSL method. Then, we want to perform simulations on finer meshes to obtain results that can be used as reference benchmark data for other flow solvers. This might deliver further insights into both test cases. Furthermore, we aim to encourage other groups to participate in these benchmark simulations so that valid reference data for three-dimensional two-phase flow solvers can be established. For this purpose, we make the data of all quantities of interest available on the website `http://wissrech.ins.uni-bonn.de/research/projects/risingbubblebenchmark`.

After we have thus validated our CLSVOF and LSL method in conjunction with the whole flow solver, we finally turn to the simulation of droplet impact.

6.4. Droplet impact

In this section, we evaluate our mathematical and numerical models by the simulation of droplet impact.

First, we consider the impact of water droplets in a purely qualitative scenario, where we choose the dynamic contact angle to be constant. The so obtained simulation results are compared with physical experiments by Vadillo et al. [VSDR09]. Herewith, we show that our numerical code is able to cope with droplet impact dynamics, that our implementation of the contact angle is valid for a wide range of applied angles and that with these prerequisites, we are already able to predict droplet impact behavior which is very close to what is observed in experiments.

Second, we simulate the impact of water on silicon rubber, which is the specific experiment Yokoi's approach has been designed for [YVHH09]. Therefore, Yokoi's approach computes dynamic contact angles which are very close to those of the practical experiment. With this small modeling error for the contact angle, we compare the LS, the LSL, the LSG and the CLSVOF method for the computation of droplet shapes and diameters. Additionally, we employ the reduced interface formation model by Shikhmurzaev [Shi07] for the dynamic contact angle computation, which we compare with Yokoi's approach and the practical experiment.

Third, we simulate micron-scale drop impact of water. Here, we employ the asymptotic interface formation only, since Yokoi's approach requires knowledge of the dynamic advancing and receding angles as well as adaption of the involved empirical parameters which we cannot provide. The droplet's height and diameter are compared with practical experiments by Dong et al. [DCBM07]. Furthermore, we perform the same simulations with a constant dynamic contact angle to establish the superiority of Shikhmurzaev's model.

Fourth, we compare the impact of a water drop on the micro-scale with the impact of a water/glycerin drop on the millimeter-scale. However, the question in how far the numerical simulation is able to capture the effect of drop size on the drop impaction process as observed in the experiments [DCBM07] cannot be finally answered by this single experiment. Instead, a variety of further research possibilities will be laid open to further study this interesting subject.

Let us first explain the general concepts of droplet impact and the associated dimensionless numbers which principally define the process.

Vadillo et al. [VSDR09] summarize the four phases of wetting as follows: First, in the kinematic phase, the drop remains spherically shaped and does not spread on the surface yet. Second, during the spreading phase, inertial forces are dominant which force the droplet to spread and act against the viscosity and the surface tension, which favor the energetic minimal spherical shape. Here, capillary waves tend to develop on the interface of the droplet. The spreading phase terminates when the maximum spreading diameter is reached. Third, in the relaxation phase, the surface tension minimizes the free surface of the drop. Here, the evolution of the droplet's diameter is often highly dependent on the motion of the rim, which is formed by capillary forces and determines the shape of the free surface at the contact line. The fourth phase is the wetting phase. No equilibrium has yet been reached. In this phase, the droplet's diameter moves very slowly, since the spreading is still opposed to by surface tension forces. At last, the droplet reaches its final diameter

and equilibrium shape.

The impaction process is mostly governed by three independent dimensionless numbers [DCBM07] which are the Ohnesorge number

$$Oh = \frac{\mu_l}{\rho_l \sigma_{lg} d_0}, \tag{6.19}$$

the Weber number

$$We = \frac{\rho_l v_0^2 d_0}{\sigma_{lg}}, \tag{6.20}$$

and the contact angle θ which is determined by Young's equation

$$\sigma_{sg} = \sigma_{sl} + \sigma_{lg} \cos \theta_e, \tag{6.21}$$

where σ_{sg} is the solid-gas, σ_{sl} the liquid-solid, and σ_{lg} the liquid-gas interfacial tension; cf. Figure 2.3. Further, μ_l is the liquid viscosity, ρ_l is the liquid density, v_0 the impact speed and d_0 the initial droplet diameter. An alternative to the definition of the Ohnesorge number is the use of the Reynolds number

$$Re = \frac{\sqrt{We}}{Oh}. \tag{6.22}$$

The Weber number is a measure of the fluid's inertia compared to its surface tension, the Ohnesorge number relates the viscous forces to inertial and surface tension forces, and the Reynolds number quantifies the relative importance of inertial and viscous forces.

In the following, we use the abbreviations

- CCA Constant contact angle approach; $\theta_d = \theta_e$, i.e. the dynamic contact angle is kept equal to its equilibrium value at all times.
- YCA Yokoi's contact angle approach; see eq. (3.27).
- AIFM Asymptotic interface formation model by Shikhmurzaev; see eq. (3.44).

6.4.1. Water on various substrates with constant contact angle

In this subsection, we simulate the impact of water droplets onto smooth surfaces with different degrees of wettability. We do not employ any of our sophisticated models for the dynamic contact angle computation and simply choose $\theta_d = \theta_e$ (the CCA), which is approximately valid as long as inertial forces dominate surface tension forces. The simulation results obtained by this approach are compared with physical experiments by Vadillo et al. [VSDR09]. At this stage, we are interested in a purely qualitative comparison only to show that our numerical code is able to cope with droplet impact dynamics, that our implementation of the contact angle gives reasonable results for a wide range of applied angles and that with these prerequisites, we are already able to predict droplet impact behavior which is very close to what is observed in experiments.

final time:	$T = 9.0,\ 5.5,\ 5.0,\ 12.0\,\mathrm{ms}$ for $\theta_e = 5°,\ 50°,\ 90°,\ 179°$, respectively.
flow domain:	$\Omega = \begin{cases} 9.0 \times 3.0 \times 9.0 \cdot 10^{-3}\,\mathrm{m}^3 & \text{if } \theta_e = 5° \\ 6.0 \times 3.0 \times 6.0 \cdot 10^{-3}\,\mathrm{m}^3 & \text{else} \end{cases}$
droplet radius:	$r = 1.14 \times 10^{-3}\,\mathrm{m}$
droplet center:	$\boldsymbol{x}_c = \begin{cases} (4.5, 1.14, 4.5)^\mathsf{T} \cdot 10^{-3}\,\mathrm{m} & \text{if } \theta_e = 5° \\ (3.0, 1.14, 3.0)^\mathsf{T} \cdot 10^{-3}\,\mathrm{m} & \text{else} \end{cases}$
impact velocity:	$v_0 = 0.35\,\mathrm{m\,s^{-1}}$
material parameters:	see Table 6.16
body force:	$\boldsymbol{g} = (0.0, -9.8, 0.0)^\mathsf{T}\mathrm{m\,s^{-2}}$
interface thickness:	$\epsilon = 1.9h$
grid resolution:	$\#\text{cells} = \begin{cases} 241 \times 91 \times 241 & \text{if } \theta_e = 5° \\ 181 \times 91 \times 181 & \text{else} \end{cases}$
employed method:	CLSVOF
contact angle model:	CCA

Table 6.15.: Simulation parameters for droplet impaction with the CCA.

	$\rho/\mathrm{kg\,m^{-3}}$	$\mu/10^{-3}\,\mathrm{kg\,m^{-1}\,s^{-1}}$	$\sigma_{lg}/10^{-2}\,\mathrm{N\,m^{-1}}$
		$\theta_e \in \{50°, 90°, 179°\}$	
water	969.4	1.005	7.275
air	1.25	0.0182	
		$\theta_e = 5$	
water	1000	1	7.3
air	1.25	0.0182	

Table 6.16.: Material parameters for droplet impaction with the CCA.

Definition of the numerical experiment

We consider a domain Ω filled with air and a droplet of water with impact speed $v_0 = 0.35\,\mathrm{m\,s^{-1}}$. The droplet is assumed to be spherical with radius $r = 1.14 \times 10^{-3}\,\mathrm{m}$ and located directly above the substrate. Thus, the droplet is defined by

$$\Omega_1(0) = \{\boldsymbol{x} \in \Omega \mid \|\boldsymbol{x} - \boldsymbol{x}_c\| \leq r\}.$$

with the center point $\boldsymbol{x}_c$ defined in Table 6.15 along with all relevant simulation parameters. The water droplet impacts on smooth surfaces with different equilibrium contact angles of $\theta_e = 5°,\ 50°,\ 90°$ and $179°$, and our simulation stops at $T = 9.0,\ 5.5,\ 5.0$ and $12.0\,\mathrm{ms}$, respectively. No-slip boundary conditions are assumed on all walls. Here, we only employ the CLSVOF method for the capturing of the free surface. Note that the computational domain is larger for $\theta_e = 5°$ since the droplet spreads further due to the smaller contact angle.

Our material parameters are summarized in Table 6.16. These are chosen very close to the parameters of the practical experiment. However, we opted for a contact angle of 179° instead of 180° so that, in theory, Proposition 3.1 still applies. Since we are interested in a qualitative comparison only, we are not concerned with mirroring the physical parameters

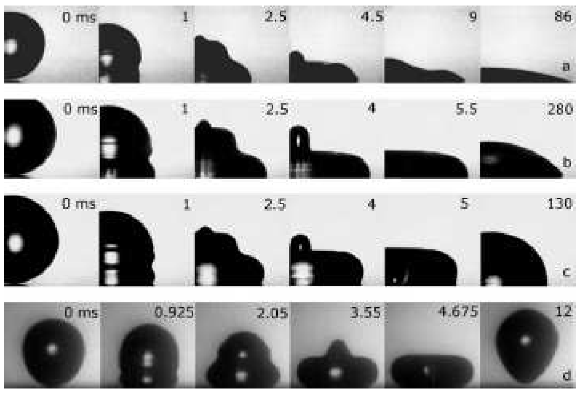


Figure 6.20: Photographs of water drops impacting on smooth surfaces with different equilibrium contact angles: (a) θ_e=5°, (b) θ_e=50°, (c) θ_e=90°, (d) θ_e=180°, $v_0 = 0.35\,\mathrm{m/s}$, $d_0 = 2.28\,\mathrm{mm}$, $\rho_l = 1000\,\mathrm{kg\,m^{-3}}$, $\mu_l = 1.0 \times 10^{-3}\,\mathrm{kg\,m^{-1}\,s^{-1}}$, $\sigma = 7.3 \times 10^{-2}\,\mathrm{N\,m^{-1}}$. Used with permission of Vadillo et al. [VSDR09].

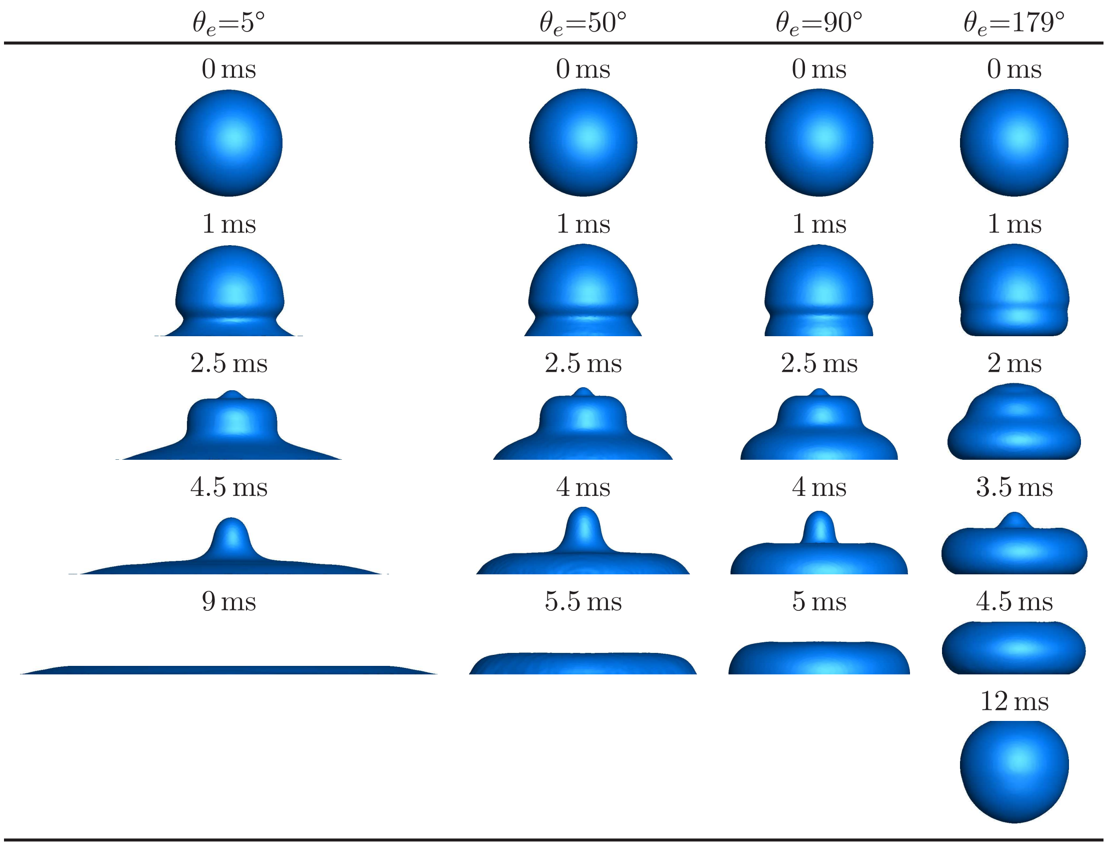


Figure 6.21.: Numerical simulation of water drops impacting on smooth surfaces. From left to right: Different equilibrium contact angles with time increasing from top to bottom. For the experimental results, compare Figure 6.20. For a direct comparison, see Figure 6.22.

exactly. Although we experimented with slightly different density and viscosity ratios for $\theta_e = 5°$, such small differences fall below the numerical error anyway and are possibly only relevant on excessively fine grids which are not studied in this experiment. The droplet shapes photographed in the practical experiment are shown in Figure 6.20 with a summary of the experiment's relevant physical parameters.

In what follows, we compare the numerically simulated droplet shapes qualitatively with the practical experiment.

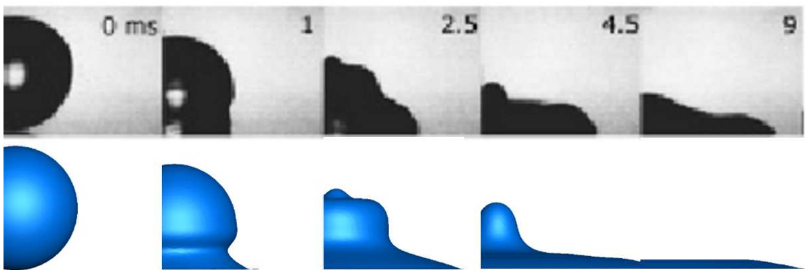
Expected behavior

With this numerical experiment, we want to show that our Navier-Stokes solver combined with the CLSVOF method is able to cope with droplet impact dynamics. We expect that our implementation of the contact angle gives reasonable results for a wide range of applied angles and that we are able to predict droplet impact behavior which is very close to what is observed in experiments. At least in the spreading phase when inertia is dominant, the contact angle model should only play a minor role. Then, all droplet shapes should be very close to those observed in the experiment. Later on, larger differences are bound to occur.

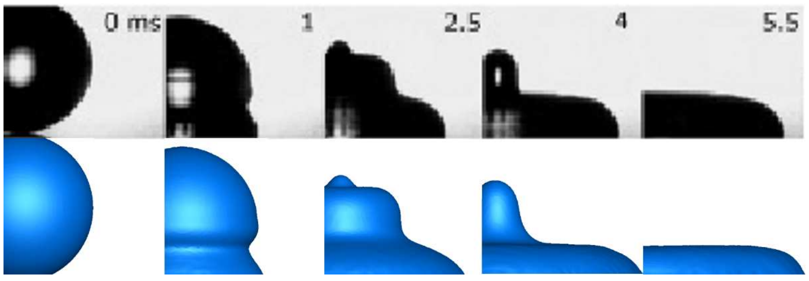
Discussion of results

The simulated droplet shapes are presented in Figure 6.21. As expected, the droplet behavior strongly depends on the applied contact angle. Both extremes are depicted at $t = 9\,\text{ms}$ for $\theta_e = 5°$ where we observe maximal spreading, while at $t = 12\,\text{ms}$ for $\theta_e = 179°$ the droplet rebounds. In between these extreme angles, the droplet behavior is comparable. However, at all times, the droplet with $\theta_e = 50°$ displays a larger droplet diameter than the one with $\theta_e = 90°$. At $t = 1\,\text{ms}$, all droplets still have a spherical cap. They mainly differ in the lower part, where the degree of wetting depends on the contact angle: The lower the angle, the higher the wettability of the surface. Then, at about 2 ms to 2.5 ms, the droplet consists of three layers, whose width again varies with the contact angle. At the next time step, all that remains from the upper two layers of the droplet is a small 'hat', while its lower part nears maximum wetting. Then, even the 'hat' disappears, so that a single layer remains whose thickness increases with increasing contact angle. Finally, at about $t = 12\,\text{ms}$ the droplet rebounds when $\theta_e = 179°$.

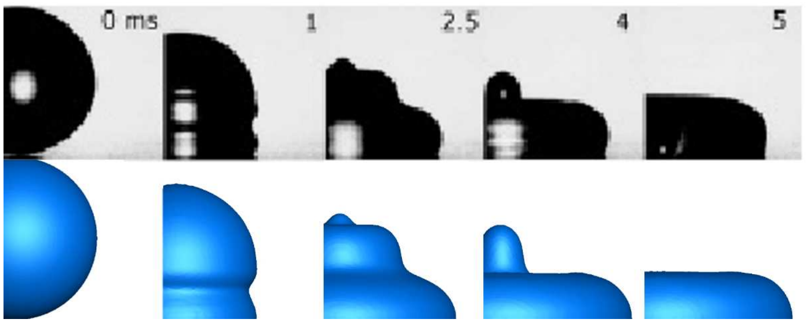
A direct comparison between the numerical results and the practical experiment is offered in Figure 6.22. Overall, we observe amazing agreement of the numerically computed droplet shapes and the experimental ones. Loosely speaking, the amount of layers displayed by the droplets in the practical experiment is always reproduced by the simulation, and the height and width of the droplet is captured quite well. However, there are several small differences between the photographs and the simulation results, which seem to deepen with the progress of time, i.e. when capillary and surface tension forces gain more dominance. With the exception of $\theta_e = 179°$, the dynamic contact angle is always overestimated by the simulation, which results in a larger diameter of the simulated droplets. These imperfections decrease with higher angles and almost vanish for $\theta_e = 179°$; see Figure 6.22(d). In essence, the droplet height and width are then captured perfectly.



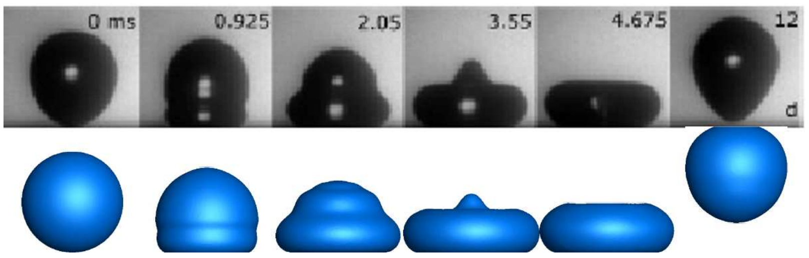
(a) θ_e=5°



(b) θ_e=50°



(c) θ_e=90°



(d) θ_e=179°

Figure 6.22.: Photographs (black) and simulations (blue) of water drops impacting on smooth surfaces with different equilibrium contact angles. Experimental results presented with permission of Vadillo et al. [VSDR09].

Conclusions

With this numerical experiment, we have shown that our Navier-Stokes solver combined with the CLSVOF method is well-suited to cope with droplet impact dynamics. We are able to simulate the rebound of droplets, which is a very important since frequently encountered phenomenon. Furthermore, our implementation of the contact angle showed reasonable results for a wide range of applied angles and the predicted droplet impact behavior with our simple contact angle model was already very close to the photographs. We also saw that our simple model works best in the early stages of impact, i.e. in the spreading phase, when inertia is dominant. As expected, larger differences occur later on and increase with decreasing contact angle.

With this purely qualitative comparison, we see that several questions will have to be addressed in the subsequent numerical experiments. First, we need to measure the droplet height and width and the contact angle evolution to obtain the grounds for a quantitative comparison. Second, we have to employ the sophisticated AIFM and the YCA, compare them with each other and with the CCA used in this subsection. Thereby, we can estimate the ‘real quality’ of our models, since our simple approach already produces very reasonable results.

To summarize, our Navier-Stokes solver is well-equipped to deal with droplet impact scenarios and we are intent on further improving the numerical results with the help of AIFM and YCA for the computation of a truly *dynamic* contact angle.

6.4.2. Water on silicon rubber

In this subsection, we evaluate our mathematical and numerical models for the example of a droplet impact simulation. Specifically, we consider a droplet of distilled water which impacts on a silicon wafer onto which hydrophobic silane has been grafted.

This scenario is valuable due to various reasons: First, Yokoi et al. [YVHH09] provide experimental results for the droplet behavior. Thus, we can compare the droplet shape, droplet diameter and dynamic contact angle from the physical experiments with the numerical results of our dynamic contact angle model by Shikhmurzaev (3.44) (AIFM) and the approach by Yokoi (3.27) (YCA). Additionally, the YCA has been specifically designed for this experiment, so we do not have to re-adapt the parameters k_a and k_r. Second, Yokoi et al. [YVHH09] present two-dimensional numerical results in their work, which we can use for comparison with our three-dimensional results as well. Last, in this specific droplet impact experiment, the numerically computed droplet behavior is very sensitive to the applied contact angles: Using the static contact angle instead of the YCA causes the drop to rebound; the same happens if only static advancing and receding contact angles are applied; see [YVHH09] for further details. Therefore, this is a very sensitive test case for our Navier-Stokes solver, the volume corrections methods, the CLSVOF method, the implemented contact angle boundary condition and the two dynamic contact angle models.

Note that the results of this subsection are already partly published by this author in [GK13]. In that work, we use the LS, LSL and LSG method only, while we now additionally present the results with the CLSVOF method.

In the following, we use the additional abbreviations:

1. AIFM$_{\text{MOF}}$: AIFM with Moffatt's solution (3.47) for the radial velocity.
2. AIFM$_{\text{FAR}}$: AIFM with an arbitrary far field velocity value for the radial velocity.
3. E1: Physical Experiment 1 from which the droplet photographs are taken; presented in black in Figures 6.23, 6.25 and 6.26.
4. E2: Physical Experiment 2 from which the droplet diameter is computed in Figures 6.24 and 6.27. Same experimental setup as E1 with increased resolution around the contact line.

Definition of the numerical experiment

We consider a domain $\Omega = 6.84 \times 3.42 \times 6.84 \cdot 10^{-3}\,\text{m}^3$ filled with air and a droplet of water with impact speed $v_0 = 1.0\,\text{m}\,\text{s}^{-1}$. The droplet is assumed to be spherical with radius $r = 1.14 \times 10^{-3}\,\text{m}$ and center point $\boldsymbol{x}_c = (3.42, 1.14, 3.42)^\mathsf{T} \cdot 10^{-3}\,\text{m}$, i. e.

$$\Omega_1(0) = \{\boldsymbol{x} \in \Omega \mid \|\boldsymbol{x} - \boldsymbol{x}_c\| \leq r\}.$$

The equilibrium contact angle of the substrate with distilled water is 90°, and all material parameters are given in Table 6.17.

We prescribe the final time T by $T = 0.04\,\text{s}$ and assume no-slip boundary conditions on the wall of impact. In this study, we consider 4 levels of successively refined grids (Tab. 6.18). All relevant simulation parameters are listed in Table 6.19.

Our numerical results are compared to those presented by Yokoi et al. in [YVHH09], and we want to compare both the shape of the droplet as well as its diameter to the experimental results. However, this comparison is not without problems since the droplet shapes in [YVHH09] are obtained from a different experiment than the measured contact angle and diameter. For the latter, only the right hand side of the droplet has been photographed to increase the resolution around the contact line. For further comparison, we therefore denote the physical experiment conducted for the photographs by E1 and the physical experiment conducted for the measurement of the droplet diameter by E2. Note that we present examples for the discrepancy between E1 and E2 in the discussion of our numerical results which will necessitate a distinction between the two practical experiments.

Our numerical experiment is thus evaluated as follows.

1. We begin with the results computed by the YCA; see Table 6.19 for the involved parameters. With this approach we employ the LS, LSL, LSG and CLSVOF method. Here, we compare our three-dimensional simulation results to the droplet photographs E1, the droplet diameter E2 and the two-dimensional numerical results from [YVHH09].
2. For all involved numerical methods, we compute the amount of mass retained at $t = 30\,\text{ms}$ on levels 1–4, which corresponds to the last droplet shapes displayed in Figure 6.23. Then, we compute the relative error

$$e_l = \frac{|m_l^{30} - m_l^0|}{|m_l^0|}, \tag{6.23}$$

	$\rho/\mathrm{kg\,m^{-3}}$	$\mu/10^{-3}\,\mathrm{kg\,m^{-1}\,s^{-1}}$	$\sigma_{lg}/\mathrm{N\,m^{-1}}$	θ_e
water	1000	1	7.2×10^{-2}	90°
air	1.25	0.0182		

Table 6.17.: Material parameters for distilled water impacting on a silicon wafer onto which hydrophobic silane has been grafted and for air.

Level l	$\Delta x_l/10^{-4}\,\mathrm{m}$	$\Delta y_l/10^{-4}\,\mathrm{m}$	$\Delta z_l/10^{-4}\,\mathrm{m}$	dof_l
1	2.206	2.280	2.206	$31 \times 15 \times 31$
2	1.121	1.103	1.121	$61 \times 31 \times 61$
3	0.565	0.561	0.565	$121 \times 61 \times 121$
4	0.284	0.283	0.284	$241 \times 121 \times 241$

Table 6.18.: Details of the meshes used for the simulation of distilled water impacting on a silicon wafer onto which hydrophobic silane has been grafted.

final time:	$T = 0.04\,\mathrm{s}$
flow domain:	$\Omega = 6.84 \times 3.42 \times 6.84 \cdot 10^{-3}\,\mathrm{m}^3$
droplet radius:	$r = 1.14 \times 10^{-3}\,\mathrm{m}$
droplet center:	$\boldsymbol{x}_c = (3.42, 1.14, 3.42)^\mathsf{T} \cdot 10^{-3}\,\mathrm{m}$
impact speed:	$v_0 = 1.0\,\mathrm{m\,s^{-1}}$
material parameters:	see Table 6.17
body force:	$\boldsymbol{g} = (0.0, -9.8, 0.0)^\mathsf{T}\mathrm{m\,s^{-2}}$
interface thickness:	$\varepsilon = 1.9h$
grid resolution:	see Table 6.18
employed methods:	LS, LSL, LSG, CLSVOF
Parameters for the YCA	
dynamic advancing angle:	114°
dynamic receding angle:	52°
material-related parameters:	$k_a = 9.0 \times 10^{-9}$ and $k_r = 9.0 \times 10^{-8}$
Parameters for the AIFM	
dimensionless parameter:	$Sc = 11.5$ for the AIFM$_{\mathrm{MOF}}$ and $Sc = 5.5$ for the AIFM$_{\mathrm{FAR}}$
dimensionless surface tension (gas-solid):	$\tilde{\sigma}_{sg} = 0$
phenomenological constant:	$\rho_G^s = 0.9$

Table 6.19.: Simulation parameters for distilled water impacting on a silicon wafer onto which hydrophobic silane has been grafted.

the convergence rate

$$\rho_l = \frac{\ln\left(\frac{e_l}{e_{l+1}}\right)}{\ln(2)}, \tag{6.24}$$

and the percentage of the remaining mass compared to the initial one. Again, $l = 1, \dots, 4$ denotes the level of refinement of our four meshes described in Table 6.18. All volume integrals are approximated with the help of the VisIt software [CBW+12].

3. In a next step, we present results computed with two variants of the AIFM: On the one hand we take Moffatt's solution for the radial velocity (AIFM$_{\text{MOF}}$) and on the other hand we choose a far field velocity value to incorporate the influence of the flow field on the dynamic contact angle (AIFM$_{\text{FAR}}$). The parameter Sc is chosen to be 11.5 for the AIFM$_{\text{MOF}}$ and 5.5 for the AIFM$_{\text{FAR}}$; compare the ending of Subsection 3.6.2. We set the values $\rho_G^s = 0.9$ and $\tilde{\sigma}_{sg} = 0$ according to [Shi07]; see Table 6.19. Here, we simulate the droplet with the LSG and the CLSVOF method only. Again, we compare our three-dimensional simulation results to the droplet photographs E1 and the droplet diameter E2. Furthermore, the relationship between the contact line speed and the contact angle are validated by the experimental results from [YVHH09].

Expected behavior

We already stated in Subsection 2.1.3 that there are fundamental differences between the YCA and the IFM. The YCA lacks a thorough underlying mathematical theory and is developed for this droplet impact experiment only. Thereby, this model is close to a prescription of dynamic contact angle values which fit well with the specific practical experiment. Since the so prescribed dynamic contact angle values are very close to the ones observed in the experiment, the modeling error is very small. Therefore, it gives us the opportunity to test our contact angle implementation as well as our methods for mass conservation and compare them to the results of the experiment.

Using the YCA, we therefore expect that all of our employed methods are able to produce droplet shapes and diameters which are very close to the practical experiment. In particular, we expect that the results from the LSG method are very close to those by the LS method since the fix-point iteration tends to simply inflate the droplet. The LSL method enhances mass conservation in every grid cell near the interface, so that a larger deviation from the LS method is possible. The CLSVOF method uses explicit interface reconstruction and a different implementation of the dynamic contact angle. Since the droplet shape is very sensitive to minor variances of the contact angle, the computed results are anticipated to differ the most from the other methods. Additionally, our three-dimensional results should also be comparable to the 2D ones in [YVHH09]. We cannot hope to exceed these results since the YCA is tuned to the 2D approach by choice of the material-related parameters k_a and k_r, which we do not adapt for the 3D case.

All methods are expected to conserve the mass of the droplet on the finest grid. Note that the mesh width is not as fine as in Subsection 6.3.1, so we do not expect any oscillations with the LSL method. Since the impact speed is moderate, no splashing effects should occur. Therefore, both the LSL and the LSG method will not show the same disadvantageous behavior encountered in Subsection 6.2.1. Mass conservation is expected to be worst with the LS method and best with the CLSVOF and the LSG method. The LSL method only prevents the straying of the level-set function, but does not correct mass errors which occur due to the numerical diffusion introduced when solving the transport equation (3.16). The LSG method, on the other hand, employs an absolute stopping criterion of $\varepsilon = 10^{-7}$ in the fixed point iteration and we anticipate to find a very small mass error only.

Second, we perform simulations with the AIFM, which is able to describe a vast variety of dynamic wetting phenomena. However, the AIFM, is derived for small capillary numbers only. Therefore, we expect to obtain an approximate and smoothed relationship between the contact line velocity and the contact angle as compared to the practical experiments. In contrast, the YCA has been fit to this specific experiment and, therefore, will produce dynamic contact angles which are very close to the ones measured in the experiment. Consequently, the droplet shapes and diameters with the AIFM will also show a larger difference from the practical experiment than those computed with the YCA.

Note that both contact angle models and all numerical methods are expected to be very close to the experimental snapshots in the spreading phase when inertia dominates capillary effects.

Discussion of results with the YCA

Before we start our discussion, we give an example on the discrepancy between the physical experiments E1 and E2: First, we measure the diameter of the photographed droplet in the first row of Figure 6.23 at $t = 15\,\text{ms}$. Compared to the initial droplet diameter at $t = 0\,\text{ms}$, we obtain a droplet diameter of about 1.7 mm. However, if we look at the also experimentally measured droplet diameter in Figure 6.24, we obtain a diameter of about 2.0 mm at $t = 15\,\text{ms}$. This discrepancy between E1 and E2 renders the comparison of the numerical results to the practical experiments complicated. For example, in Figure 6.23 at $t = 4\,\text{ms}$ the LSG method predicts a horizontally wider droplet than the laboratory experiment E1, which is depicted in the first row of the figure. However, if you compare this result with the diameter-time curve in Figure 6.24(b), the numerically computed droplet diameter at $t = 4\,\text{ms}$ reproduces the droplet diameter from the laboratory experiment E2 almost perfectly. However, the discrepancy between E1 and E2 also indicates the involved measuring error of the physical experiment. Hence, our numerical results concerning the droplet diameter are considered fine if they lie in between the values predicted by both experiments.

a) Comparison of droplet shapes

First, we compare the droplet shapes of the LS, LSL, LSG and the CLSVOF method with the experimental snapshots from E1. The numerically obtained droplet shapes (white) during the impact are shown in Figure 6.23 compared with experiments by [YVHH09]. These are always shown in black in the first row of the figures, including a reflection at the bottom of the photograph. Note that these reflections are also added to each numerical result. Additionally, we compare the numerically computed droplet diameter with the experimental results from E2 in Figure 6.24.

Let us start with the LS method depicted in the second row of Figure 6.23. We see that during the first three points in time when inertia is dominant, the droplet shapes

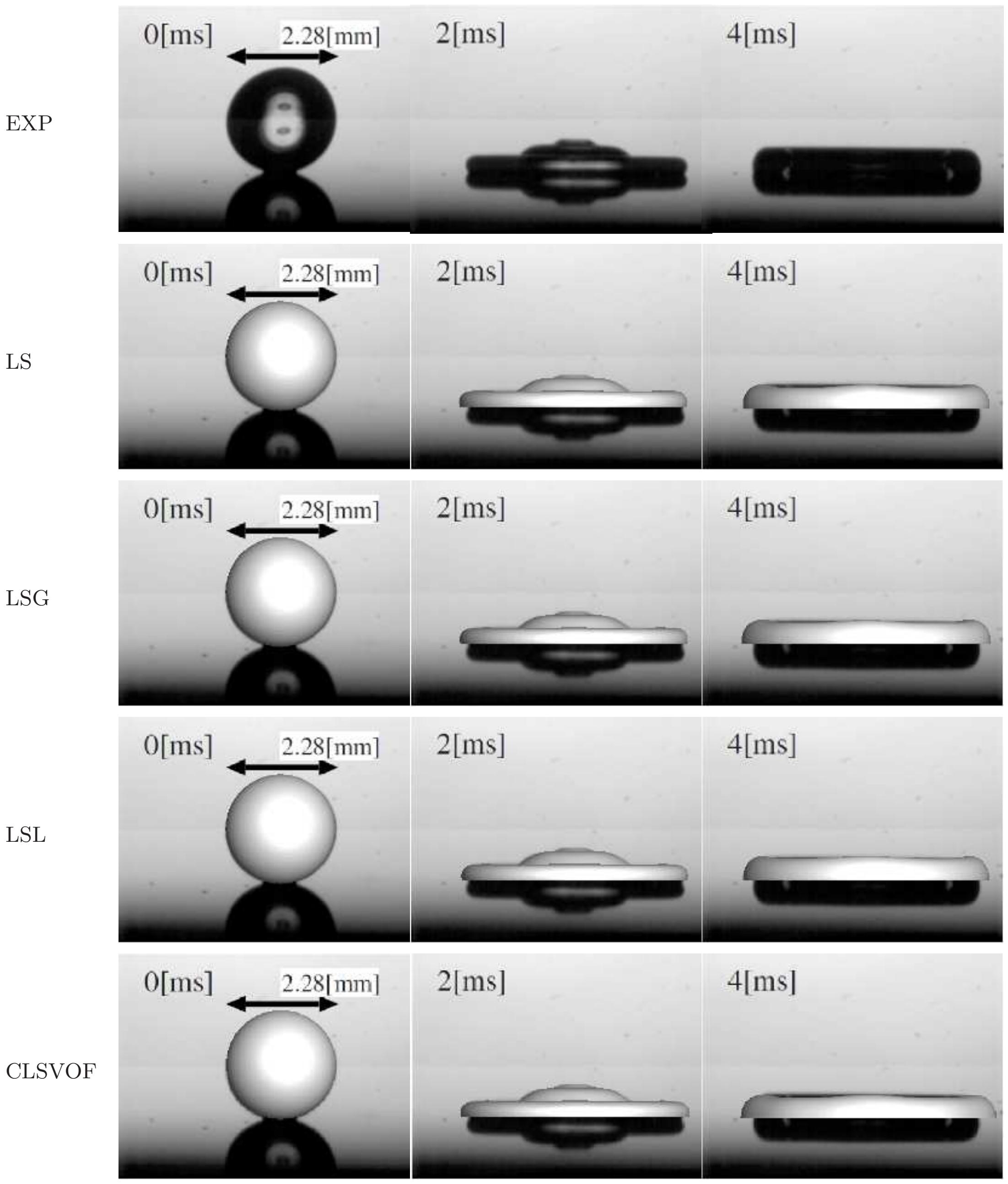


Figure 6.23.: Droplet impact $t = 0$ to 4 ms. In descending order: experimental results from [YVHH09] and results with the LS, the LSG, the LSL and the CLSVOF method. Note that the experimental results are depicted in black with a reflection of the droplet at the bottom. Numerical results are drawn in white and are projected onto the experimental results.

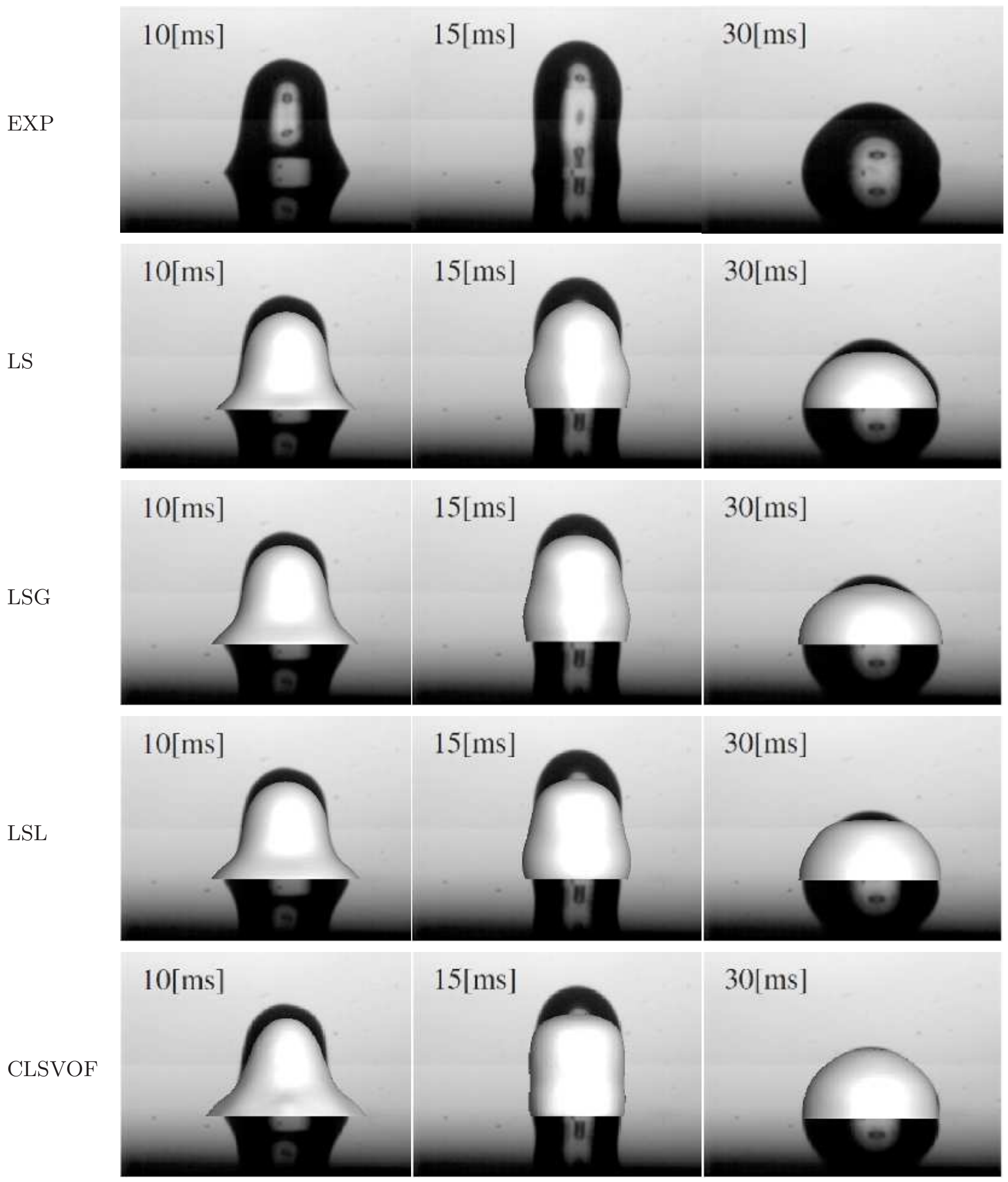
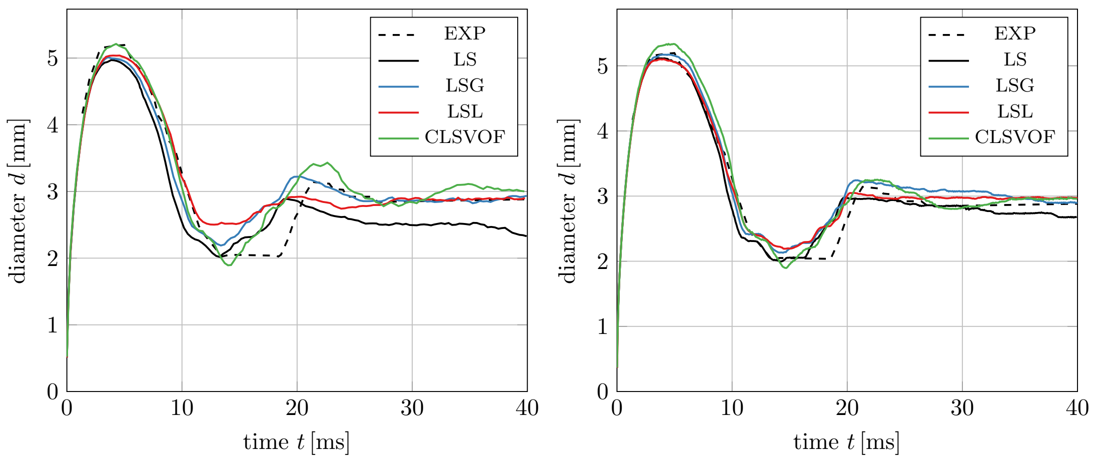


Figure 6.23.: Continued from previous page: Droplet impact $t = 10$ to $30\,\text{ms}$.



(a) 121 × 61 × 121 grid cells. **(b)** 241 × 121 × 241 grid cells.

Figure 6.24.: Comparison of droplet diameter over time with experimental results. Our 3D simulations with the LS (black), LSG (blue), LSL (red) and the CLSVOF method (green) compared to experimental results by Yokoi et al. [YVHH09] (dashed).

compare very well with the experimental snapshots from E1. At $t = 10$ ms the simulated droplet shape is still remarkably close to the experiment, while at the next time steps, the numerical droplet fails to reproduce the correct droplet height and width observed in the experiment. This is partly due to its obvious loss in mass: Despite the very high resolution, the droplet still loses about 20% of its volume during the simulation. This mass loss is also encountered in the computation of the droplet diameter over time: In Figure 6.24, we see that the LS method is unable to reproduce the final droplet diameter for both applied resolutions. Nevertheless, at about $t = 4$ ms the maximum diameter is well recovered and the overall behavior of the diameter-time curve is close to the experiment. We see here that the simulated droplet diameter at times $t = 10$ and 15 ms is closer to E2 than to E1. All in all, despite its obvious loss in volume and within the experimental measuring error, the LS method is able to produce simulation results, which correspond well with the experimentally observed droplet behavior. Nevertheless, at this point, we see the need for volume-conserving simulation methods.

In the next step, we therefore simulate the droplet impact with the LSG method. The resulting droplet shapes are presented in the third row of Figure 6.23. As expected, the results from the LSG method are very close to those by the LS method since the fix-point iteration imply inflates the droplet: At all times, the droplet shapes recovered by the LSG method are similar to those by the LS method, but the volume of the drop is now preserved up to 100%. This similarity is also displayed in the evolution of the diameter over time (Fig. 6.24), which is identical up to a slight shift due to the mass loss of the LS method. On account of the improved mass conservation, the maximum droplet diameter at $t = 4$ ms and the final droplet diameter are captured excellently by the LSG method. Again, from comparison of Figures 6.23 and 6.24, we observe that the droplet diameter at the last three time steps is closer to E2 than to E1.

The simulation results with the LSL method are shown in the fourth row of Figure 6.23.

For the first three points in time, the droplet shapes with the LSL method are in good agreement with the other simulation results and the experiments. Then, the droplet shapes slightly differ from those computed by the LSG and the LS method and: At times $t = 10$ and 15 ms, the droplet width is slightly more overshot than with the other two methods. Again, the droplet diameter results with the LSL method compare better with E2 than with E1: In Figure 6.24(b), we see very good agreement with the experimental results from E2 for the computation of the droplet diameter.

The fifth row of Figure 6.23 displays the simulated droplet shapes with the CLSVOF method. At $t = 2$ ms, the simulation is very close to the practical experiment. Then, at about $t = 4$ ms, we observe a prominent overshoot of the droplet diameter. The width of the droplet remains too large at $t = 10$ ms, while the droplet height is still comparable to the results by the LS, LSL and LSG method. Contrary, at $t = 15$ ms, the droplet width now fits perfectly to the experiment while the droplet height is underestimated similar to the LSL method. At last, the final droplet shape is captured excellently and best by the CLSVOF method. In the diameter over time curve (Fig. 6.24(b)), we observe that the droplet diameter is indeed overshot with the CLSVOF method at $t = 4$ ms. All in all the, CLSVOF method shows the largest local deviations from the experimental results E2, while the overall evolution of the curve is captured best with the CLSVOF method: A maximum is reached at about $t = 4$ ms and a minimum at $t = 15$ ms. Then, the next local maximum is tracked most prominently by the CLSVOF method. Towards the end of the simulation, the CLSVOF method exhibits slight oscillations, which are very well in the range of the final droplet diameter. These anticipated deviations from the other simulation results can be attributed to the differently implemented contact angle and interface capturing approach in the CLSVOF method.

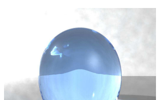
To summarize, all methods are in very high agreement with the experimental results and lie well within the scope of the experimental error.

b) Comparison with 2D simulation results from [YVHH09]

Second, we compare our three-dimensional results to the two-dimensional ones by Yokoi et al. which are also obtained by a CLSVOF method. The 2D numerical droplet diameter is not given in Figure 6.24, but if it were, it would be almost indistinguishable from E2. In consequence, the computed 2D droplet shapes deviate more from the experimental results E1. These shapes can be found in Figure 8 of [YVHH09], where they are visualized as three-dimensional results. Contrary to our simulation, the droplet shape at $t = 2$ ms tends more to a pyramid shape and does not convincingly show the three layers obtained in the experiment. This might well be due to the lacking third dimension. Later, the two-dimensional results are comparable with our 3D results, but the width of the droplet at $t = 10$ ms is even larger than in our case. All in all, we have to remember that the two parameters k_a and k_r in the YCA were used to fit the 2D numerical results to E2, and we did not adapt these parameters for our 3D simulation. Even so, the 2D and 3D simulations show good agreement with each other.

c) Convergence of Mass

Third, we compute the relative mass error (6.23), the convergence rate (6.24) and the droplet's remaining mass at $t = 30$ ms. All of these results are summarized in Table 6.20.



l	LS			LSL			LSG			CLSVOF		
	e_l	ρ_l	%	e_l	ρ_l	%	e_l	ρ_l	%	e_l	ρ_l	%
1	1.000_{+00}	0.72	0.00	3.512_{-01}	2.12	64.9	2.513_{-02}	1.35	102.5	5.878_{-02}	5.13	94.1
2	6.056_{-01}	0.70	39.4	8.064_{-02}	1.27	91.9	9.892_{-03}	2.65	101.0	1.684_{-03}	−0.40	99.8
3	3.716_{-01}	0.82	62.8	3.350_{-02}	2.42	96.7	1.578_{-03}	1.20	100.2	2.223_{-03}	1.40	99.8
4	2.102_{-01}	–	79.0	6.260_{-03}	–	99.4	3.951_{-04}	–	100.0	8.434_{-04}	–	99.9

Table 6.20.: Mass conservation: Relative error e_l computed by eq. (6.23), convergence rate ρ_l computed by eq. (6.24) and mass conservation in % at $t = 30\,\text{ms}$, where l denotes the level of refinement of our meshes; see Table 6.18.

Mass conservation with the LS method is difficult for this particular case: On the coarsest grid, no mass is left after $t = 30\,\text{ms}$ and on the finest grid, we still lose about 20% of mass. However, the LSL, the LSG and the CLSVOF method tend to near 100% mass conservation on the finest grid. As expected, the LSG method is able to conserve about 100% of mass on all meshes. The overcorrection of mass is due to our implementation of the LSG method which has already been discussed in Section 6.2.1. In addition, the CLSVOF method shows the anticipated excellent mass conservation from level 2 onwards, while the LSL method performs worse on coarser grids. For this test case, we obtain a convergence order of about 0.8 for the LS method. The convergence order for the remaining methods is supposedly higher. However, we have only just reached the asymptotic regime on levels 3 and 4, and we would need an even finer mesh for the confirmation of the convergence orders.

Discussion of results with the AIFM

a) Comparison of droplet shapes

Now, we present the results for the droplet impact simulation with the AIFM combined with the LSG and the CLSVOF method. These results are computed with two variants of the AIFM: On the one hand we take Moffatt's solution for the radial velocity (AIFM_{MOF}) and on the other hand we choose a far field velocity value to incorporate the influence of the flow field on the dynamic contact angle (AIFM_{FAR}) (cf. Tab. 6.19). Again, we compare our three-dimensional simulation results to the droplet photographs E1 and the droplet diameter E2 from experiments.

Approximately, both the LSG and the CLSVOF method predict identical results for this test case. The droplet shapes for both the AIFM_{MOF} and the AIFM_{FAR} are very similar in Figures 6.25 and 6.26, and the diameter over time curves in Figure 6.27 hardly deviate. That is why in the following, we focus on the results of the CLSVOF method only.

In Figure 6.26, we compare the results of the laboratory experiments E1 in the first row with the droplet shapes computed by the AIFM_{MOF} in the second and by the AIFM_{FAR} in the third row. In the spreading phase, both simulation results are very close to the experimental snapshots, which is to be expected, since inertia dominates capillary effects. For both methods, also at the later time steps, the computed droplet shapes agree very well with E1. For the AIFM_{MOF} at $t = 10\,\text{ms}$, the height and width of the droplet are reproduced

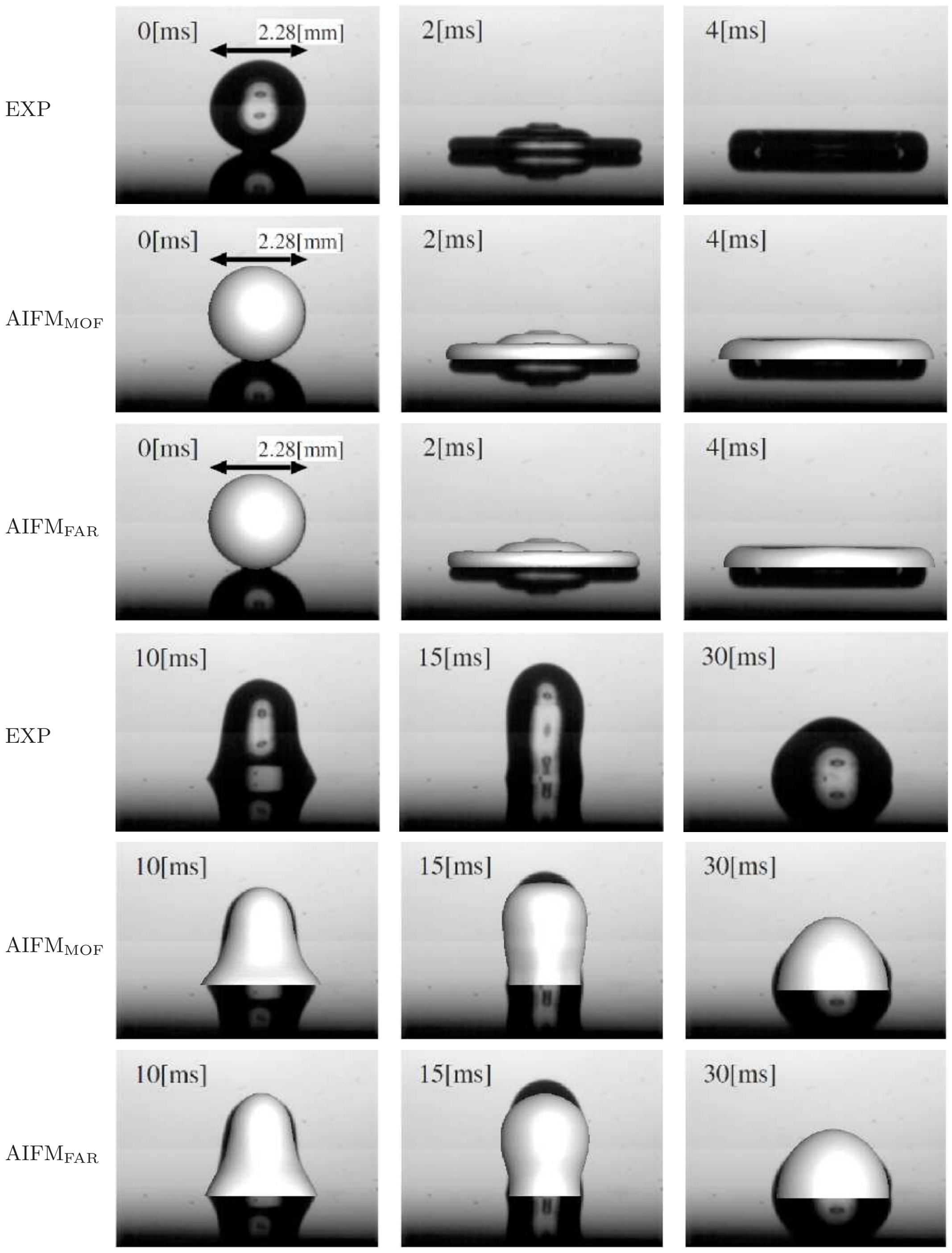


Figure 6.25.: Droplet impact $t = 0$ to $30\,\mathrm{ms}$. In descending order: experimental results from [YVHH09] the AIFM$_{\text{MOF}}$ and the AIFM$_{\text{FAR}}$. Both numerical results are computed by the LSG method. Note that the experimental results are depicted in black with a reflection of the droplet at the bottom. Numerical results are drawn in white and are projected onto the experimental results.

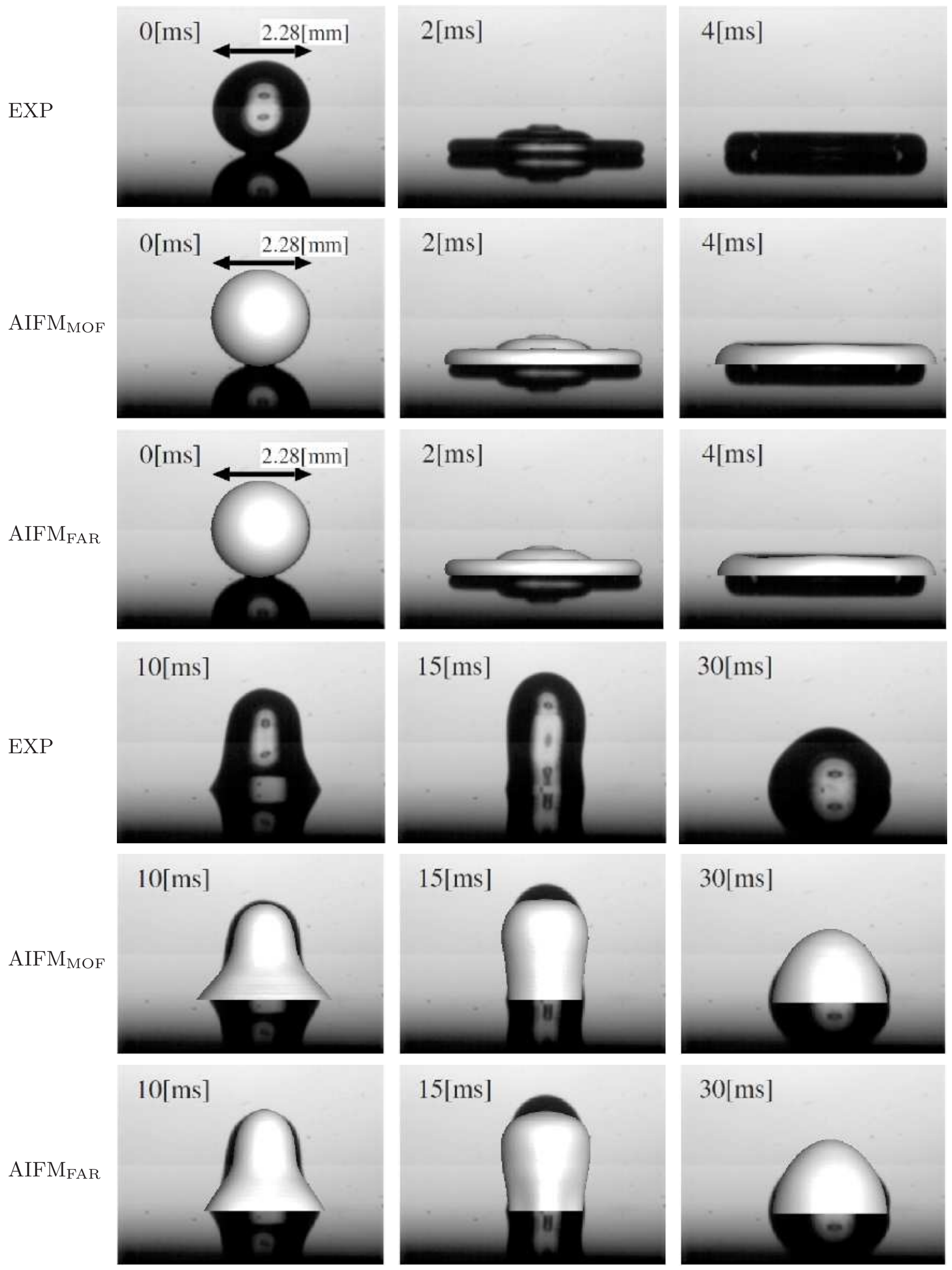
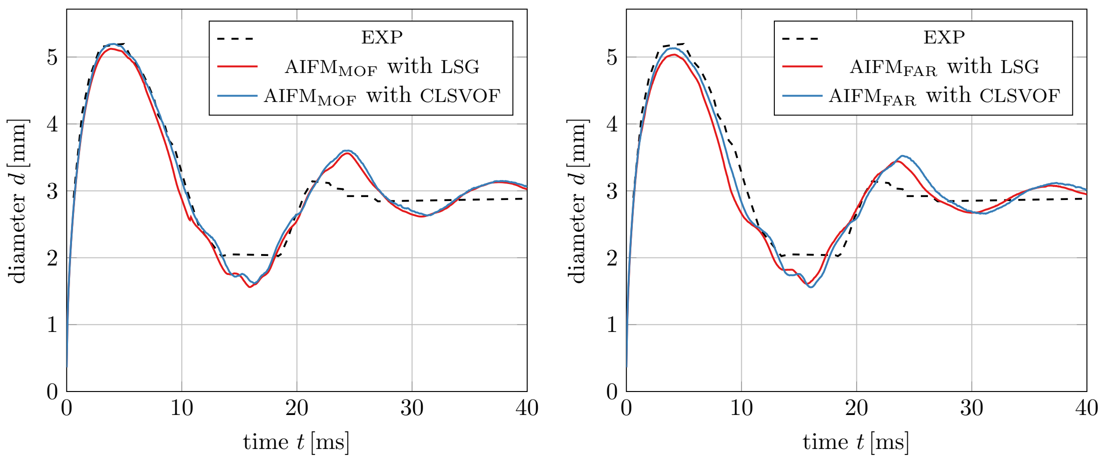


Figure 6.26.: Droplet impact $t = 0$ to $30\,\text{ms}$. In descending order: experimental results from [YVHH09] the AIFM$_{\text{MOF}}$ and the AIFM$_{\text{FAR}}$. Both numerical results are computed by the CLSVOF method. Note that the experimental results are depicted in black with a reflection of the droplet at the bottom. Numerical results are drawn in white and are projected onto the experimental results.



(a) AIFM$_{\mathrm{MOF}}$.   **(b)** AIFM$_{\mathrm{FAR}}$.

Figure 6.27.: Comparison of droplet diameter over time with experimental results. Our 3D simulations are computed with two variants of the AIFM with the CLSVOF method (blue) and the LSG method (red). These results are compared to the experiment by Yokoi et al. [YVHH09] given by the thin dashed line. The grid resolution is $241 \times 121 \times 241$.

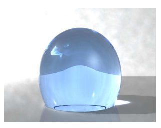
very reasonably. Further, at $t = 15\,\mathrm{ms}$, the droplet even forms a little dent before it meets the substrate and compares best with the experiment of all simulation results. The droplet shapes computed by the AIFM$_{\mathrm{FAR}}$ are close to the results by the AIFM$_{\mathrm{MOF}}$ (Fig. 6.26). In contrast to the AIFM$_{\mathrm{MOF}}$, the most accurately reproduced droplet shape is achieved at $t = 10\,\mathrm{ms}$, while the difference to the experimental snapshot is a little higher at $t = 15\,\mathrm{ms}$.

A look at the droplet diameter evolution (Fig. 6.27) confirms that the interface formation model, although it is not specifically based on this experiment, produces nearly as good results as the YCA: The maximum and final droplet diameter are captured, and the computed curve is very close to that of the experimental results for both the AIFM$_{\mathrm{MOF}}$ and the AIFM$_{\mathrm{FAR}}$.

b) Comparison of contact angles

In a next step, we compare the angle-speed curve of the AIFM with the experiment. The YCA, since it is fit to this experiment, produces dynamic contact angles which are very close to the ones measured in the experiment. From both variants of the AIFM we expect a smooth angle-speed relationship equal to our preliminary computation in Figure 3.5. This is exactly what we see in Figure 6.28: The contact angles computed by the AIFM develop in a smooth and almost linear fashion but show a larger deviation from the experiment. For zero contact line velocity, the model predicts the equilibrium contact angle of 90°. Here, the values measured in the experiment are between 52° and 110°. This is due to the space-time resolution of the practical experiment, where the contact line velocity is considered to be zero, if the liquid interface does not cross any pixel. Interestingly, however, the smooth speed-angle curve predicted by the interface formation model, lies quite well in between the maximum advancing and minimum receding contact angle of the experiment.

Both the AIFM$_{\mathrm{MOF}}$ and the AIFM$_{\mathrm{FAR}}$ compute very similar dynamic contact angles in



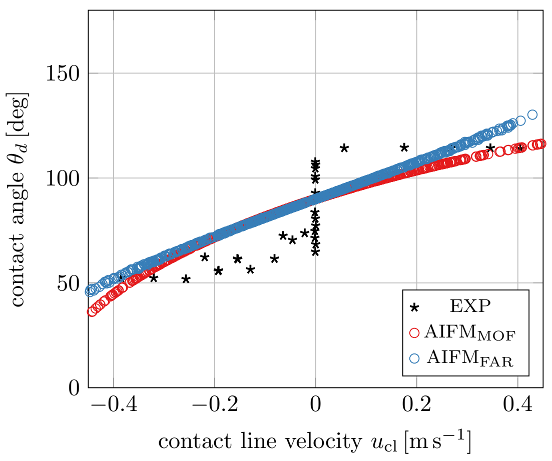


Figure 6.28.: Angle-speed relationship of a 3D simulation with the AIFM$_{\mathrm{MOF}}$ (red) and the AIFM$_{\mathrm{FAR}}$ (blue) compared to experiments (black). Note that for zero contact line velocity, the angles in the practical experiment are between 52° and 110°, which is due to its resolution since the contact line velocity is considered to be zero if the liquid interface does not cross any pixel. The experimental values are from [YVHH09].

specific regimes of the contact line velocity. For large contact line velocities, the AIFM$_{\mathrm{FAR}}$ overshoots the maximum dynamic advancing angle determined from the laboratory experiment even more than the AIFM$_{\mathrm{MOF}}$. However, for small contact line velocities, the AIFM$_{\mathrm{FAR}}$ tends to be closer to the minimum dynamic receding angle than the AIFM$_{\mathrm{MOF}}$.

Conclusions for both models

In this subsection, we simulated the impact of a water droplet on silicone grafted with hydrophobic silane.

We confirmed that the YCA, which is fit to this specific experiment and our 3D Navier-Stokes solver produces droplet shapes and diameters which are very close to the practical experiment. As expected, the results from the LSG method were very close to those by the LS method since the fix-point iteration tends to simply inflate the droplet. Results with the LSL method deviated more from those by the LS method, while the CLSVOF results stood out the most, since it uses explicit interface reconstruction and a different implementation of the dynamic contact angle. However, all methods are in excellent agreement with the experimental results and lie well within the scope of the experimental error. Additionally, our three-dimensional results compared very well to the 2D ones in [YVHH09].

Not all methods were able to conserve the mass of the droplet on the finest grid. Thus, the LS method still loses 20% of mass on the finest grid, while the LSG method and the CLSVOF method conserve almost 100% of mass from level 2 onwards. As expected, the LSL method loses more mass on coarser grids since it only prevents the straying of the level-set function, but does not correct mass errors which occur due to the numerical diffusion introduced when solving the transport equation (3.16). For the LS method, we obtained a little less than linear convergence, while the convergence rate of the CLSVOF and the LSG method tended to about 1.5 and for the LSL method to about 2. Further computations on finer grids are needed to confirm the convergence behavior.

Second, we performed simulations with the AIFM, which is able to describe a vast variety of dynamic wetting phenomena. We confirmed that the asymptotic approach of this model, in combination with our 3D Navier-Stokes solver, produced droplet shapes and diameters which are very close to the practical experiment. We employed two variants of the AIFM: On the one hand we took Moffatt's solution for the radial velocity and on the other hand we chose a far field velocity value. Both results are comparable for the droplet shapes, the diameter and the speed-angle curve. With the AIFM, we employed the CLSVOF and LSG method only and the difference between both outcomes was insignificant compared to the experimental error. Since we used an asymptotic version of the interface formation model which is derived for small capillary numbers only, we obtained the expected approximate and smoothed relationship between the contact line speed and the contact angle in contrast to the practical experiments and the YCA.

To summarize, we conclude that all methods are in very high agreement with the experimental results. Furthermore, the results by the AIFM are most promising. Although it is not based on this specific droplet experiment, the computed droplet shapes are very close to those observed in the experiment, which is supported by the droplet diameter measurements and the speed-angle relationship.

In the following, we focus on the AIFM only since the YCA requires the special knowledge of advancing and receding angles and an adaption of the material-related parameters. In general, these angles are not provided in practical experiments. In contrast, the phenomenological constants in the AIFM can be chosen as described in [BS02, Shi07], so that this model is easily applicable to a wide range of droplet impact scenarios.

6.4.3. Micron-scale drop impact

Most experimental studies of drop impaction focus on drops in a millimeter-scale regime. However, in ink-jet printing and other industrial processes such as spray coating or drop spraying the drop size ranges from sub-microns to a several hundred microns. In order to study the applicability of results on the millimeter-scale to micron-scale drops, practical experiments are conducted in [DCBM07], where the impact of micron-scale water droplets on surfaces with several different equilibrium contact angles is compared to the impact of millimeter droplets consisting of a mixture of water and glycerin.

In this subsection, we use the AIFM$_{\text{FAR}}$ and the CCA to study the micron drop impact of four drops with impact velocities $4.36\,\mathrm{m\,s^{-1}}$ and $12.2\,\mathrm{m\,s^{-1}}$ on two different substrates with equilibrium contact angles 31° and 107°. In all simulations, the interface is tracked by the CLSVOF method. The computed droplet shapes and the heights and widths of the droplets are compared to practical experiments. Furthermore, by comparison with the experiments, we establish the profound superiority of the sophisticated AIFM$_{\text{FAR}}$ over the much simpler CCA.

Note that in this subsection, we focus on micron-scale droplet impactions only, while our next subsection is dedicated to a comparison of micron- and millimeter-scale droplet impact as in [DCBM07].

Definition of the numerical experiment

We consider a domain Ω filled with air and a droplet of water impacting on two substrates with equilibrium contact angles 31° and 107°, respectively. The first substrate is a thermally oxidized silicon wafer, while the second consists of three self-assembled monolayers on a gold-coated silicon wafer. In the practical experiment, the wafer is prepared in such a way that the amounts of hydrophilic (OH) and hydrophobic (CH_3) groups vary in the monolayers [DCBM07]. Here, we consider the case of 100% CH_3 in the three self-assembled monolayers only. In the following, we refer to this substrate as S107 and to the thermally oxidized silicon wafer as S31 due to their equilibrium contact angle with our water droplet.

On these two substrates, we use the AIFM$_{\text{FAR}}$ and the CCA to study the micron drop impact with two impact velocities $4.36\,\mathrm{m\,s^{-1}}$ and $12.2\,\mathrm{m\,s^{-1}}$. We summarize the relevant physical and material parameters in Tables 6.21 and 6.22. For the purpose of comparison, we already include the parameters of the millimeter-scale experiment which is considered in the next subsection. All in all, we perform eight different simulations within our numerical experiment, whose distinctive features are summarized in Table 6.23.

For our simulations, we solve the Navier-Stokes equations combined with the CLSVOF method for the capturing of the interface. As almost always, the numerical interface thickness is $\varepsilon = 1.9h$, and we use a narrow band of four cells for the reinitialization of the level-set function. All relevant simulation parameters are summarized in Table 6.24. In this table, we also find the phenomenological constants and dimensionless parameters of the AIFM$_{\text{FAR}}$. These values are taken from [BS02], where the velocity dependence of the dynamic contact angle is studied experimentally for a series of water-glycerol solutions. The corresponding results are interpreted by the interface formation theory with estimates for the phenomenological parameters of the model.

Liquid	$v_0/\mathrm{m\,s^{-1}}$	$d_0/10^{-6}\,\mathrm{m}$	*Re*	*We*	*Oh*	θ_e
distilled water	4.36	48.8	238	12.8	0.0151	31° and 107°
distilled water	12.2	50.5	689	103	0.0148	31° and 107°
glycerin/water (49.5:50.5)	1.67	2230	700	105	0.0146	99°

Table 6.21.: Impact velocity, diameter and dimensionless numbers for the impact of water and of the glycerin mixture studied subsequently.

	$\rho/\mathrm{kg\,m^{-3}}$	$\mu/10^{-3}\,\mathrm{kg\,m^{-1}\,s^{-1}}$	$\sigma_{lg}/10^{-2}\,\mathrm{N\,m^{-1}}$
	distilled water (micron-scale impact)		
liquid	1000	8.93	7.2
air	1.184	0.01826	
	glycerin/water (49.5:50.5) (millimeter-scale impact)		
liquid	1113	6.0	6.7
air	1.250	0.0180	

Table 6.22.: Material parameters for distilled water and for glycerin/water (49.5:50.5).

Experiment	CA model	Liquid	$v_0/\mathrm{m\,s^{-1}}$	$d_0/10^{-6}\,\mathrm{m}$	substrate	θ_e
1.0	$\mathrm{AIFM_{FAR}}$	distilled water	4.36	48.8	S31	31°
1.1	CCA					
2.0	$\mathrm{AIFM_{FAR}}$	distilled water	4.36	48.8	S107	107°
2.1	CCA					
3.0	$\mathrm{AIFM_{FAR}}$	distilled water	12.2	50.5	S31	31°
3.1	CCA					
4.0[a]	$\mathrm{AIFM_{FAR}}$	distilled water	12.2	50.5	S107	107°
4.1	CCA					
$\mathrm{Exp_{mm}}$	$\mathrm{AIFM_{FAR}}$	glycerin/water (49.5:50.5)	1.67	2230	S107	99°

Table 6.23.: Description of the eight different simulations that we perform in this numerical experiment. The additional $\mathrm{Exp_{mm}}$ is investigated in the next subsection and is listed here for the purpose of comparison only.

[a]Referred to as $\mathrm{Exp_{\mu m}}$ in Subsection 6.4.4 for comparison with $\mathrm{Exp_{mm}}$.

final time:	$T = 50\,\mu\text{s}$
flow domain:	$\Omega = (0.15 \times 0.075 \times 0.15) \cdot 10^{-3}\,\text{m}^3$
material parameters:	see Tables 6.21 and 6.22
body force:	$\boldsymbol{g} = (0.0, -9.8, 0.0)^{\mathsf{T}}\,\text{m}\,\text{s}^{-2}$
interface thickness:	$\epsilon = 1.9h$
grid resolution:	$\text{dof} = 301 \times 151 \times 301$
employed method:	CLSVOF
contact angle model:	AIFM_{FAR} and $\theta_d = \theta_e$
Parameters for the AIFM_{FAR}	
dimensionless parameter:	$Sc = 5.0$
dimensionless surface tension (gas-solid):	$\tilde{\sigma}_{sg} = -0.07$
phenomenological constant:	$\rho_G^s = 0.54$

Table 6.24.: Parameters for distilled water impacting on S31 with equilibrium contact angle 31° and on S107 with equilibrium contact angle 107°.

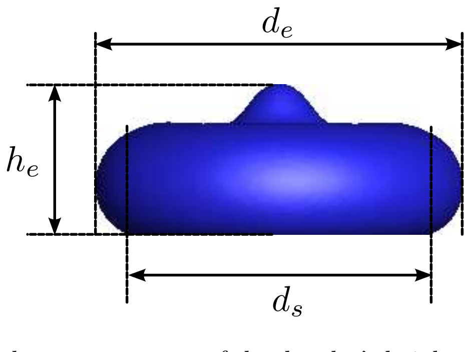


Figure 6.29.: Illustration of the measurement of the droplet's height and diameter. In the practical experiment, d_e is measured near the contact line and h_e is the droplet's maximum height. In the simulation, d_s is measured directly at the contact line and can slightly differ from d_e.

Let us shortly explain how we measure the droplet's diameter and height for the comparison with practical experiments. In the practical experiment, the droplet height h is defined as its maximum height, and the diameter d_e is its maximum width measured macroscopically 'near' the substrate. In our simulation, the droplet diameter d_s is measured directly at the contact line, which can differ slightly from the measurement of d_e as illustrated in Figure 6.29. Furthermore, the simulation droplet height h_s is always measured in the middle of the droplet, which is not always possible in experiments, e.g., when the outer rim is higher.

To account for these differences we conduct a post-processing step with the ParaView software [Par11], in which we measure the outline of the droplet over time[4], whereby we obtain the diameter d_p and the height h_p, which both correspond to the maximum extension in the respective directions. Then, the final simulation height is defined as $h = \max(h_s, h_p)$, so that our run-time results are corrected if the highest point of the droplet is not its center. Furthermore, the diameter d_p is displayed as a dashed line, if $d_p \neq d_s$ and if d_p is measured near the substrate (when the droplet rebounds, its maximum extension is found far from the substrate). Thereby, the final simulation droplet height h and diameter d are measured as similar to the practical experiment as possible.
Our simulations are evaluated as follows.

1. We focus on the evolution of the droplet by plotting its zero contour, which we extract with the Tecplot software.
2. We compare the numerically computed droplet shapes by the AIFM$_{\text{FAR}}$ and the CCA with the experimental ones by Dong et al. [DCBM07].
3. We compute the dimensionless droplet height $H^* = h/d_0$ and the dimensionless diameter $D^* = d/d_0$ over time and compare the results of the AIFM$_{\text{FAR}}$ and the CCA to the data from practical experiments provided by [DCBM07].

Note that we are not using dimensionless quantities in our computations, and for the micron-scale numerical experiments, we convert all input parameters from meters to millimeters. For example, the density $\rho = 1000\,\mathrm{kg\,m^{-3}}$ is converted to $\rho = 1000{\cdot}10^{-9}\,\mathrm{kg\,mm^{-3}}$. This conversion is due to the computational accuracy: When we used meters instead of millimeters, the LS function, which is a distance function, could no longer be accurately defined since the involved distances simply became too small.

Expected behavior

All in all, we expect that our results with the AIFM$_{\text{FAR}}$ are close to the practical experiments for the droplet shapes, the droplet diameter and its height. For all numerical experiments, the AIFM$_{\text{FAR}}$ must show considerable improvement compared to the CCA.

Particularly, at early times after impact, when initial forces are more dominant, we expect good results from both the AIFM$_{\text{FAR}}$ and the CCA. However, we have already seen in Subsection 6.4.1, that the droplet diameter tends to be overestimated with the CCA, which can be expected for these numerical experiments as well. Instead, with the AIFM$_{\text{FAR}}$, we expect a much better approximation of the experimentally measured diameter.

[4]Use ParaView to show the zero contour of the level-set function. Then, in the contour's 'information' menu, the x-, y- and z-ranges of the droplet can be found.

The Weber number for Exp. 1 and 2 is considerably smaller than for Exp. 3 and 4, i.e. surface tension forces are more dominant in the first two experiments than in the last two. If surface tension effects are dominant, the contact angle model is much more influential. Therefore, we expect larger differences between the AIFM$_{\text{FAR}}$ and the CCA for Exp. 1 and 2. These differences will lessen for Exp. 3 and 4. Here, however, we expect the AIFM$_{\text{FAR}}$ to be very close to the practical experiment, since small errors introduced by the contact angle model have less influence.

Discussion of results

a) Experiment 1

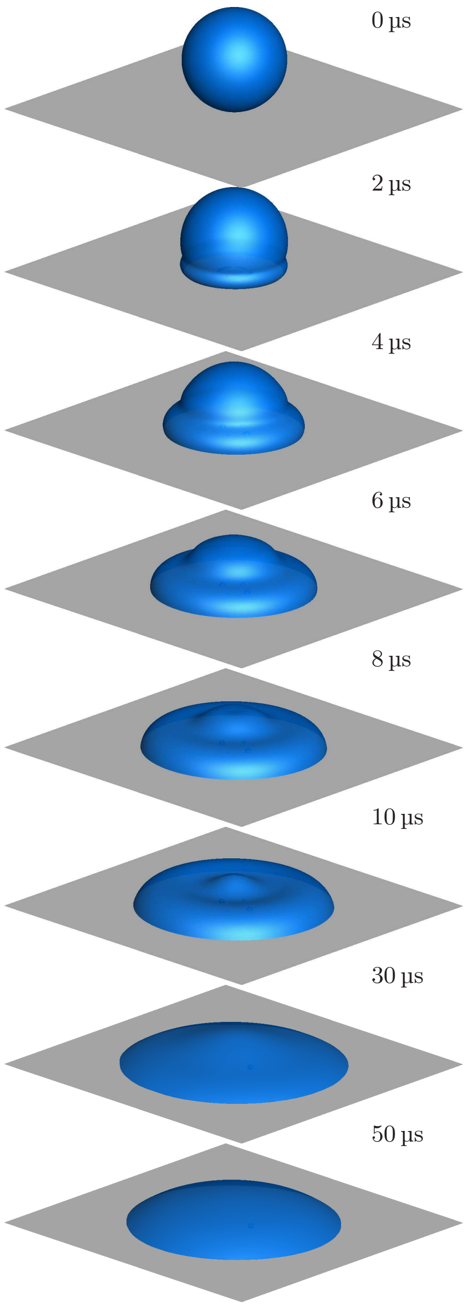
In our first experiment, we use the AIFM$_{\text{FAR}}$ as well as the CCA to simulate the impact of water on S31 with equilibrium contact angle 31°, $v_0 = 4.36\,\mathrm{m\,s^{-1}}$ and $d_0 = 48.8\,\mu\mathrm{m}$ (cf. Tab. 6.23).

First, we are interested in the droplet shapes computed with the AIFM$_{\text{FAR}}$, which are depicted in Figure 6.30(a). In this case, the deformation of the droplet is rather moderate. At $t = 2\,\mu\mathrm{s}$, two layers of the droplet develop, which are slightly stretched throughout the simulation during the spreading phase. In the subsequent relaxation phase, these two layers are no longer visible, and the droplet resembles a spherical dome.

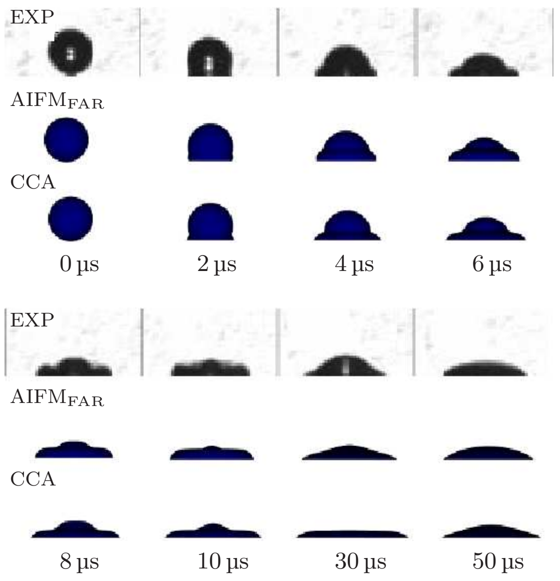
In Figure 6.30(b), we compare the results of the practical experiment (first row) with simulation results by the AIFM$_{\text{FAR}}$ (second row) and by the CCA, where θ_d is chosen equal to θ_e for the whole simulation time (third row). The droplet shapes in both simulations are very close to those observed in the practical experiment, at least up to about $t = 8\,\mu\mathrm{s}$. This is to be expected, since initially, surface tension and capillary forces are less dominant, and the droplet shapes depend less on the applied contact angle. However, even in these early times and as already observed in Subsection 6.4.1, the diameter of the droplet seems to be overestimated with the CCA. This behavior becomes most visible at $t = 10$ and $t = 30\,\mu\mathrm{s}$, where the diameter is considerably too large and the whole droplet is too flat compared to the practical experiment. In contrast, the diameter of the droplet with the AIFM$_{\text{FAR}}$ is smaller at all times, and the computed droplet shapes are very close to the experimental snapshots.

These findings are confirmed by the computation of the dimensionless diameter $D^* = d/d_0$ over time in Figure 6.30(c). The diameter in the practical experiment is slightly below the one computed by the AIFM$_{\text{FAR}}$ from about $t = 7\,\mu\mathrm{s}$ onwards. In contrast, for the CCA, the experimental results are drastically overestimated; much more so than with the AIFM$_{\text{FAR}}$ for the whole simulation. Most noticeably, from $0\,\mu\mathrm{s}$ to $7\,\mu\mathrm{s}$ the diameter computed by the AIFM$_{\text{FAR}}$ and the practical experiment agree almost perfectly, while from about $1\,\mu\mathrm{s}$ onwards, the results with the CCA already diverge.

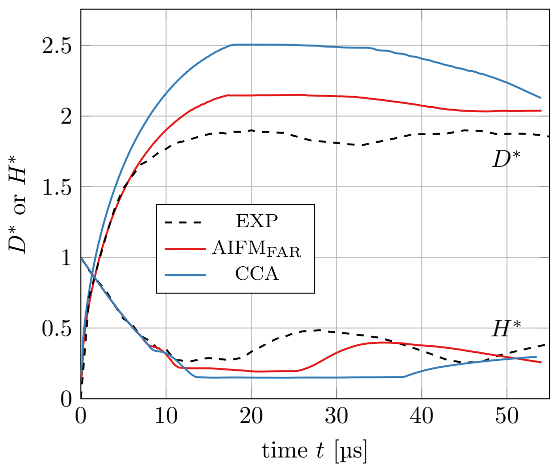
In the same Figure 6.30(c), the dimensionless droplet height $H^* = h/d_0$ is plotted over time. For the droplet height, the distinction between both contact angle models and the practical experiment becomes less prominent. From $0\,\mu\mathrm{s}$ to $10\,\mu\mathrm{s}$ all results agree perfectly. Then, up to about $t = 50\,\mu\mathrm{s}$, the droplet height is underestimated with the CCA, while the AIFM$_{\text{FAR}}$ is a little closer to the experiment. Note that the AIFM$_{\text{FAR}}$ also predicts the new rise of the droplet height as in the practical experiment, but is about $7\,\mu\mathrm{s}$ too late. Thus,



(a) Droplet shapes computed by the AIFM$_{\text{FAR}}$ for Exp. 1.0 from Table 6.23.

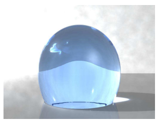


(b) Comparison of the practical experiment (first row), Exp. 1.0 (second row) and Exp. 1.1 (third row); cf. Table 6.23.



(c) Dimensionless droplet diameter D^* and height H^* vs. time.

Figure 6.30.: Distilled water impacting on S31 with equilibrium contact angle 31°, impact speed $4.36\,\mathrm{m\,s^{-1}}$, and droplet size 48.8 µm. Experimental data and figures used with permission by H. Dong [DCBM07].



both curves develop in a similar manner, with a shift for the one computed by the AIFM$_{\text{FAR}}$. This behavior cannot be observed for the CCA, where the curve seems to develop distinctly from the one of the practical experiment.

b) Experiment 2

In our second experiment, we use both models to simulate the impact of the same droplet on another substrate, i.e. on S107 with equilibrium contact angle 107°, which corresponds to Exp. 2 in Table 6.23.

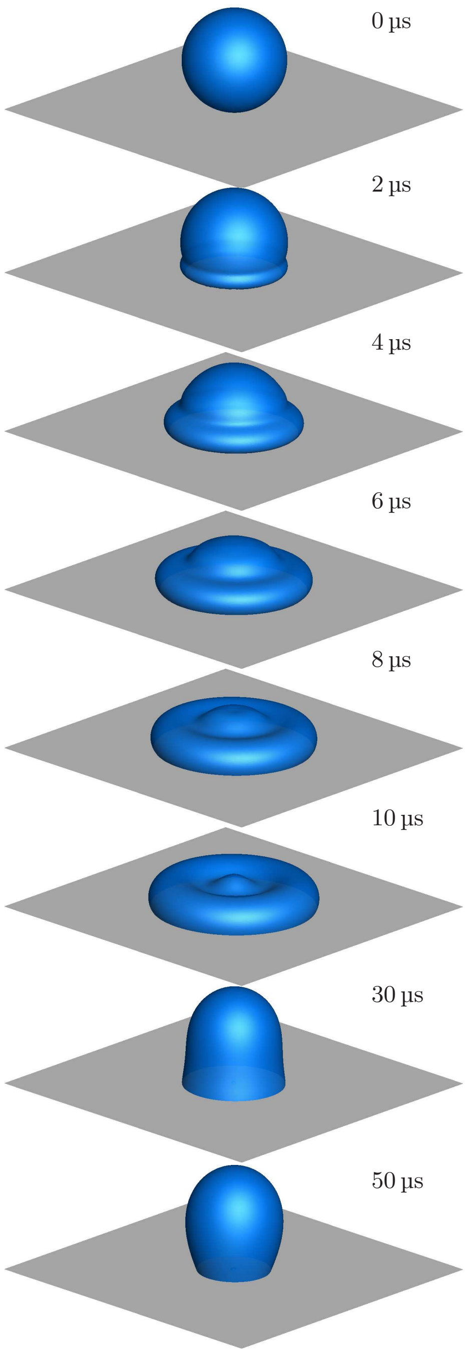
The droplets' shapes are given in Figure 6.31(a). Up to about $t = 10\,\mu\text{s}$, the droplet behaves similarly to Exp. 1. Thus, we see the development of two layers from $t = 2\,\mu\text{s}$ onwards, which are slightly stretched throughout the simulation during the spreading phase. Then, due to the higher contact angle, the droplet recedes in the subsequent relaxation phase, the two layers are no longer visible, and the droplet shape at $t = 50\,\mu\text{s}$ resembles the upper part of an ellipsoid.

In Figure 6.31(b), we again compare the results of the practical experiment (first row) with simulation results by the AIFM$_{\text{FAR}}$ (second row) and those by the CCA (third row). Up to $t = 6\,\mu\text{s}$, the droplet shapes in both simulations are very close to those observed in the practical experiment. However, even in these early times and as already observed in Subsection 6.4.1, the diameter of the droplet seems to be slightly overestimated with the CCA. This overestimation increases with time. Furthermore, at $t = 10\,\mu\text{s}$, a small spherical cap at the droplet's top is still visible for the CCA, which is not the case for the AIFM$_{\text{FAR}}$ and the practical experiment. Then, at $t = 30\,\mu\text{s}$, the height of the droplet is vastly overestimated while the droplet shape computed by the AIFM$_{\text{FAR}}$ remains much closer to the experimental snapshot. Before 50 µs are reached, the droplet even rebounds with the CCA, so that no meaningful numerical results can be obtained after that point in time.

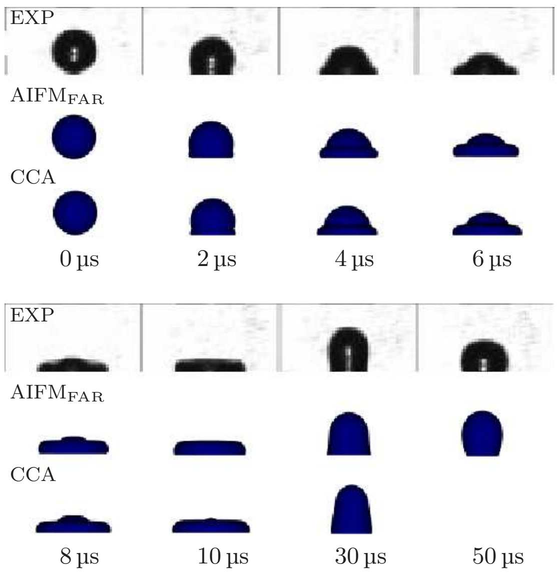
All of these findings are confirmed by the computation of the dimensionless diameter $D^* = d/d_0$ over time in Figure 6.31(c). The diameter in the practical experiment is slightly below the one computed by the AIFM$_{\text{FAR}}$ from about $t = 10\,\mu\text{s}$ to 47 µs, after which the behavior reverses. Again, the droplet diameter curve computed by the AIFM$_{\text{FAR}}$ develops similarly to the one by the practical experiment, but with a slight shift. In particular, up to 10 µs, the diameter is captured perfectly by the AIFM$_{\text{FAR}}$. In contrast, for the CCA, the experimental results diverge more prominently from the beginning, and the maximum droplet diameter is vastly overestimated. From $t = 20\,\mu\text{s}$ to 30 µs, the results are close to the AIFM$_{\text{FAR}}$ after which the diameter decreases until the rebound of the droplet[5].

In the same Figure 6.31(c), the dimensionless droplet height $H^* = h/d_0$ is plotted over time. Again, the droplet height curve computed by the AIFM$_{\text{FAR}}$ develops similarly to the one by the practical experiment, and seems to display the same shift as the diameter curve. As in Exp. 1, the distinction between both contact angle models and the practical experiment is less prominent for the droplet height than for the droplet diameter up to about 23 µs. In particular, from 0 µs to 10 µs all results agree almost perfectly. From $t = 10\,\mu\text{s}$ onwards, the droplet height increases constantly for the CCA, until the droplet rebounds. In contrast, the AIFM$_{\text{FAR}}$ is able to predict the renewed decline of the droplet height, which is earlier on also observed in the practical experiment.

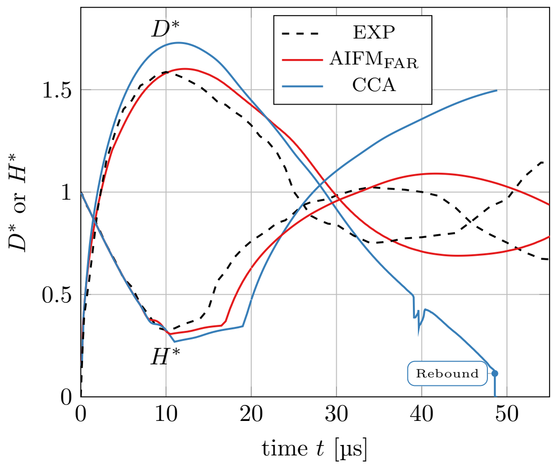
[5]In Figure 6.31(c), the irregularities at about $t = 40\,\mu\text{s}$ are due to irregularities in the interface. The source of these irregularities is discussed subsequently in (e).



(a) Droplet shapes computed by the AIFM$_{\text{FAR}}$ for Exp. 2.0 from Table 6.23.

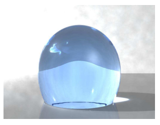


(b) Comparison of the practical experiment (first row), Exp. 2.0 (second row) and Exp. 2.1 (third row); cf. Table 6.23.



(c) Dimensionless droplet diameter D^* and height H^* vs. time.

Figure 6.31.: Distilled water impacting on S107 with equilibrium contact angle 107°, impact speed $4.36\,\mathrm{m\,s^{-1}}$, and droplet size 48.8 µm. Experimental data and figures used with permission by H. Dong [DCBM07].



c) Experiment 3

In our third experiment, we use the AIFM$_{\text{FAR}}$ as well as as the CCA to simulate the impact of water on S31 with equilibrium contact angle 31°, $v_0 = 12.2\,\mathrm{m\,s^{-1}}$ and $d_0 = 50.5\,\mu\mathrm{m}$ (cf. Tab. 6.23). Thus, we have a slightly larger droplet and an about three times higher initial velocity.

The droplet shapes computed with the AIFM$_{\text{FAR}}$ are displayed in Figure 6.32(a). Due to the higher impact speed, the droplet deforms much faster than in the previous examples. At $t = 2\,\mu\mathrm{s}$, two layers of the droplet develop, which are subsequently stretched. From $t = 4\,\mu\mathrm{s}$ onwards, the outer rim of the droplet becomes its highest part, while the droplet's middle remains leveled. Finally, at $t = 50\,\mu\mathrm{s}$ the droplet shape resembles a flat spherical cap.

In Figure 6.32(b), we compare the results of the practical experiment (first row) with simulation results by the AIFM$_{\text{FAR}}$ (second row) and by the CCA (third row). Up to about $t = 4\,\mu\mathrm{s}$ the droplet shapes in both simulations are quite close to those observed in the practical experiment. However, even in these early times and as already observed previously, the diameter of the droplet seems to be overestimated with the CCA, and the difference in diameter between this model and the practical experiment increases with time. Due to this overestimation, at about $t = 7\,\mu\mathrm{s}$, the droplet touches the walls of our simulation box, so that the results after that point in time can no longer be regarded as reliable. In addition to the too large diameter, the whole droplet is too curved in comparison to the AIFM$_{\text{FAR}}$ and the experimental snapshot. At all times, in contrast to the CCA, the droplet diameter with the AIFM$_{\text{FAR}}$ is smaller, and the droplet shapes are exceedingly close to the experimental snapshots.

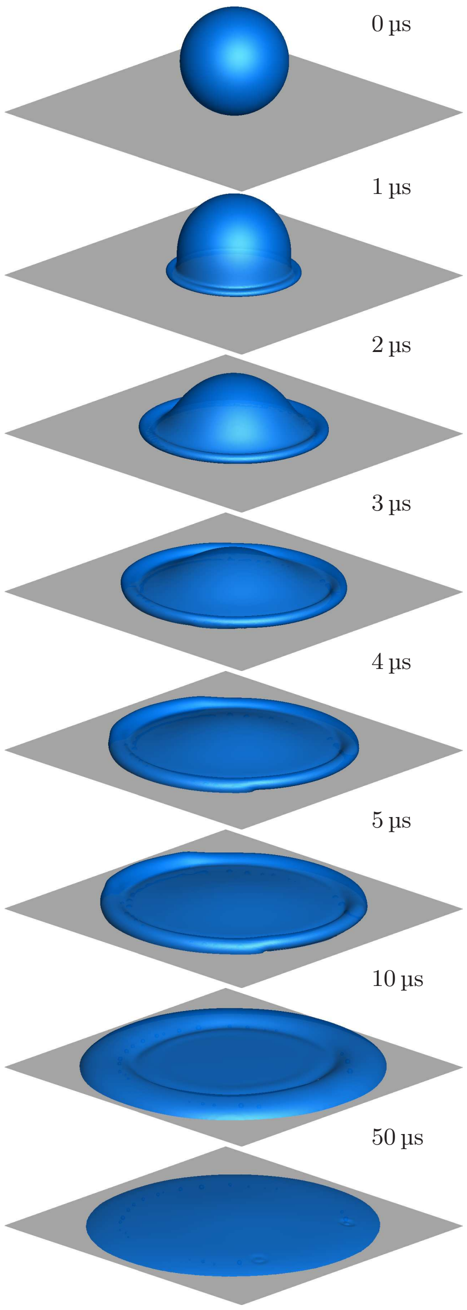
Again, we confirm these qualitative findings by the computation of the dimensionless diameter $D^* = d/d_0$ over time in Figure 6.32(c). For this experiment, the diameter computed by the AIFM$_{\text{FAR}}$ and by the practical experiment agree almost perfectly. Note that the dashed red line in between $t = 0\,\mu\mathrm{s}$ to $5\,\mu\mathrm{s}$ is computed by the ParaView software [Par11] in a post processing step as described at the beginning of this subsection (Fig. 6.29). This 'macroscopic' diameter is even closer to the experiment than the one measured directly at the contact line during run-time. In contrast, the diameter computed by the CCA deviates increasingly from $t = 2\,\mu\mathrm{s}$ onwards until the walls of the simulation domain are reached.

In the same Figure 6.32(c), the dimensionless droplet height $H^* = h/d_0$ is plotted over time. For the droplet height, the distinction between the AIFM$_{\text{FAR}}$ and the CCA is less prominent. Note that the results from about $t = 7\,\mu\mathrm{s}$ onwards are insignificant for the CCA, since the droplets touches the walls of our simulation domain. However, from $0\,\mu\mathrm{s}$ to $6\,\mu\mathrm{s}$ all results agree perfectly. Then, up to about $t = 55\,\mu\mathrm{s}$, the droplet height is only slightly underestimated with the AIFM$_{\text{FAR}}$.

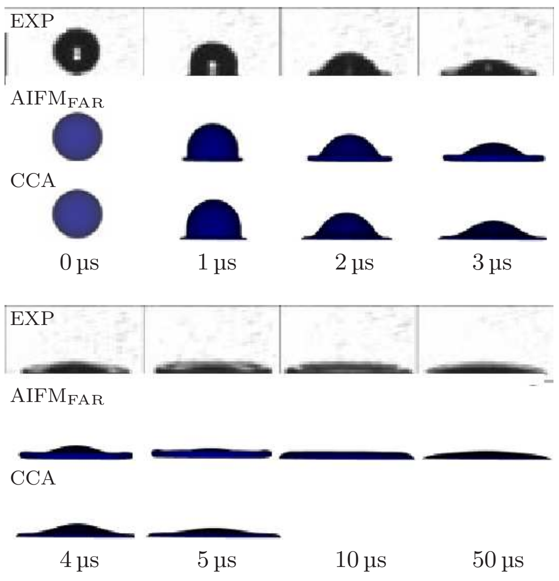
d) Experiment 4

In our fourth experiment, we simulate the impact of the same droplet as in Exp. 3 on another substrate, i.e. on S107 with equilibrium contact angle 107°, which corresponds to Exp. 4 in Table 6.23.

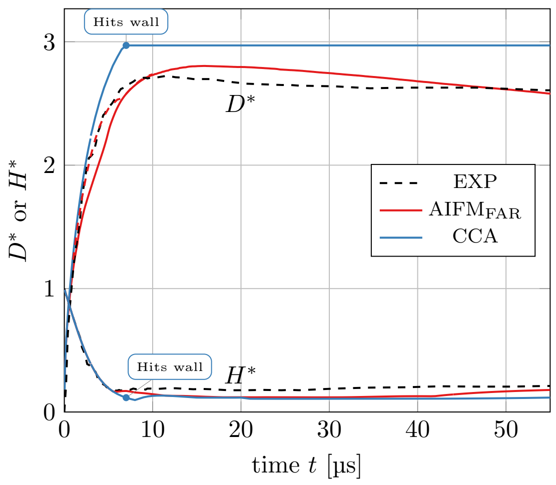
The droplets' shapes are given in Figure 6.33(a). Up to about $t = 5\,\mu\mathrm{s}$, the droplet behaves similarly to Exp. 3. Thus, we see the development of two layers from $t = 2\,\mu\mathrm{s}$ onwards which are slightly stretched throughout the spreading phase, and the rim of the



(a) Droplet shapes computed by the AIFM$_{\mathrm{FAR}}$ for Exp. 3.0 from Table 6.23.

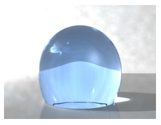


(b) Comparison of the practical experiment (first row), Exp. 3.0 (second row) and Exp. 3.1 (third row); cf. Table 6.23.



(c) Dimensionless droplet diameter D^* and height H^* vs. time. The dashed red line corresponds to $D_p^* = d_p/d_0$ computed by ParaView; see Fig. 6.29

Figure 6.32.: Distilled water impacting on S31 with equilibrium contact angle 31°, impact speed $12.2\,\mathrm{m\,s^{-1}}$, and droplet size 50.5 µm. Experimental data and figures used with permission by H. Dong [DCBM07].



droplet remains its highest part. Then, due to the higher contact angle, the droplet recedes much further at $t = 10\,\mu s$ and is about to rebound at $t = 50\,\mu s$.

In Figure 6.33(b), we compare the results of the practical experiment (first row) with simulation results by the AIFM_{FAR} (second row) and by the CCA (third row). Up to about $t = 4\,\mu s$, the droplet shapes in both simulations are quite close to those observed in the practical experiment. As always, the diameter of the droplet is overestimated with the CCA. In addition, at $t = 5\,\mu s$, the droplet is more curved than the ones computed by the AIFM_{FAR} and the practical experiment. Furthermore, this droplets recoils much faster and has already hit the top of the simulation wall at $t = 50\,\mu s$. In contrast to this approach, the droplet shapes computed by the AIFM_{FAR} are very close to the experimental snapshots at all times.

All of these findings are confirmed by the computation of the dimensionless diameter $D^* = d/d_0$ over time in Figure 6.33(c). Remarkably, the diameter computed by the AIFM_{FAR} and the practical experiment agree almost perfectly at all times. Again, the dashed red line in between $t = 0\,\mu s$ to $5\,\mu s$ is computed by the ParaView software [Par11] in a post processing step, and this 'macroscopic' diameter is even closer to the experiment than the one measured directly at the contact line during run-time. In contrast, the diameter computed by the CCA deviates increasingly from $t = 2\,\mu s$ onwards and the maximum droplet diameter is vastly overestimated. Then, from about $20\,\mu s$ onwards, the diameter is underestimated which indicates the much earlier rebound of this droplet.

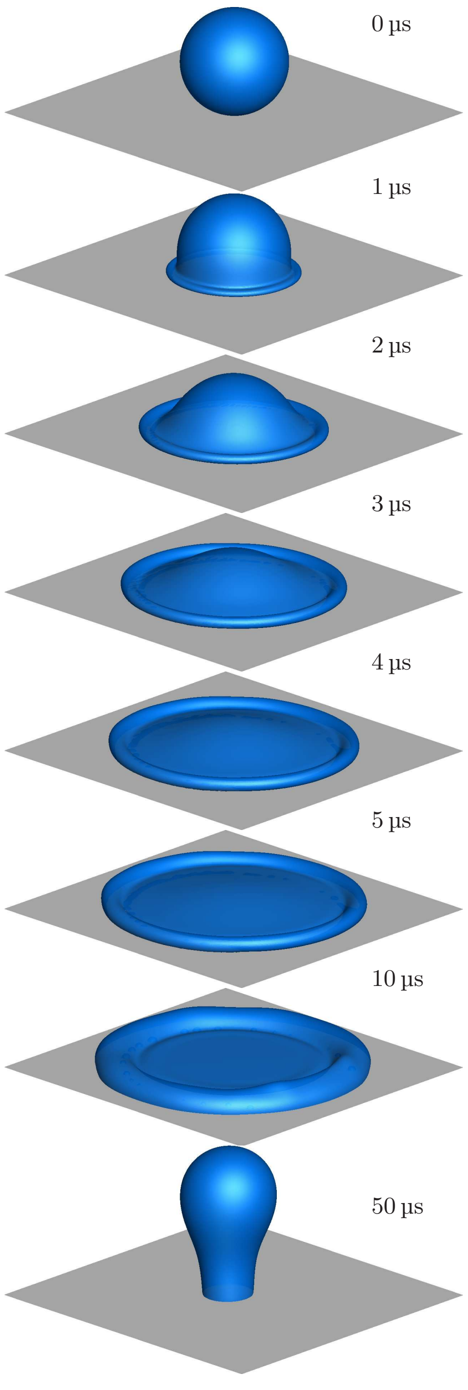
In the same Figure 6.33(c), the dimensionless droplet height $H^* = h/d_0$ is plotted over time. Again, we observe amazing agreement between the droplet height computed by the AIFM_{FAR} and the practical experiment. From $0\,\mu s$ to $20\,\mu s$ the results of the CCA are also in perfect agreement with the experiment. Then, due to the faster recoil of the droplet, the droplet height increases much more dramatically until the droplet hits the upper wall of the simulation domain.

e) Irregularities at the interface

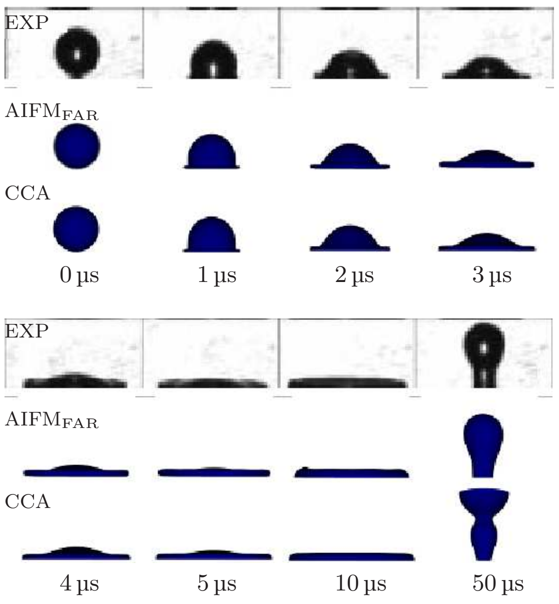
After we have thus finished our discussion on the individual experiments, we would like to comment on some of the irregularities in the interfaces visible in some parts of Figures 6.32–6.33. In these figures, the symmetry of the droplets is partly lost after spreading. Mostly, these irregularities occur in Exp. 3 and Exp. 4 due to the higher impact speed of $12.2\,\text{m}\,\text{s}^{-1}$ as opposed to $4.36\,\text{m}\,\text{s}^{-1}$ in Exp. 1 and 2. In a first investigative step, we draw the droplet of Exp. 4 from different perspectives in Figure 6.34 simulated by the AIFM_{FAR} and by the CCA.

The droplet computed by the AIFM_{FAR} is shown in Figure 6.34(a), where we observe that the interface deforms strongly at the bottom and sticks to the interface while the rest of the droplet spreads further. Therefore, small bubbles and filaments detach which are only a few mesh widths in length. Structures which fall below the mesh width tend to lead to asymmetry, which we already observed in Section 6.2. Combined with the first order interface reconstruction process of the CLSVOF method which tends to accumulate errors over time, this could well be an explanation for the asymmetric rim of the droplet as seen in Figure 6.34(c).

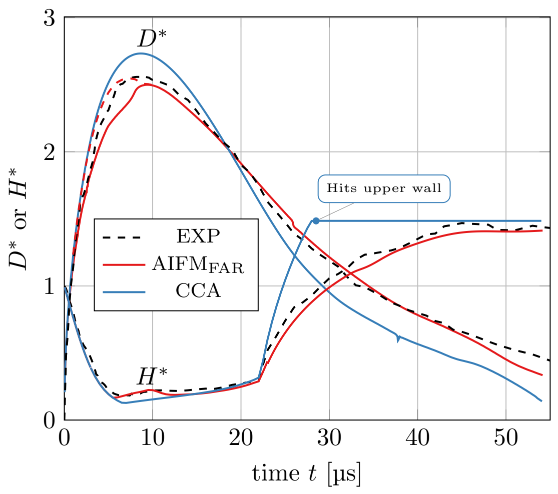
In contrast, if the CCA is chosen as in Figure 6.34(b), we also observe four entrapped air bubbles at the bottom. However, there are no bubbles attached to the outer interface and



(a) Droplet shapes computed by the AIFM$_{\text{FAR}}$ for Exp. 4.0 from Table 6.23.

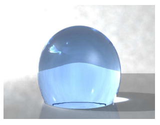


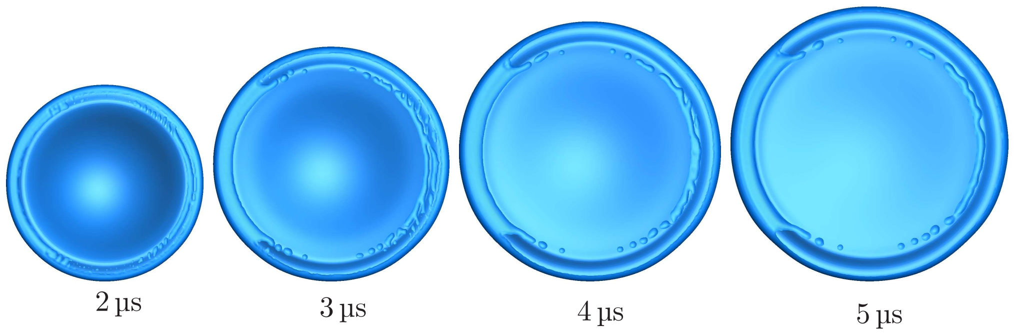
(b) Comparison of the practical experiment (first row), Exp. 4.0 (second row) and Exp. 4.1 (third row); cf. Table 6.23.



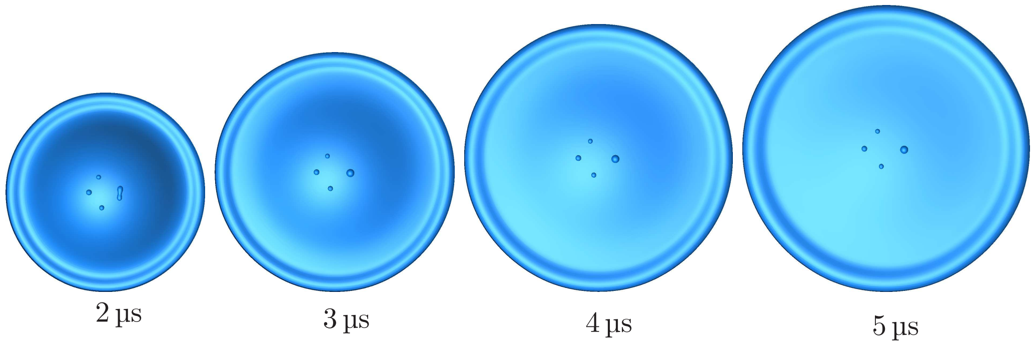
(c) Dimensionless droplet diameter D^* and height H^* vs. time. The dashed red line corresponds to $D_p^* = d_p/d_0$ computed by ParaView; see Fig. 6.29

Figure 6.33.: Distilled water impacting on S107 with equilibrium contact angle 107°, impact speed $12.2\,\mathrm{m\,s^{-1}}$, and droplet size 50.5 µm. Experimental data and figures used with permission by H. Dong [DCBM07].

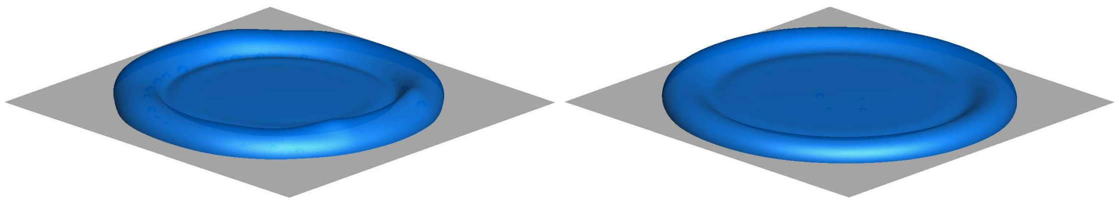




(a) Bottom view with the AIFM$_{\text{FAR}}$.



(b) Bottom view with the CCA.



(c) Side view at 10 µs with the AIFM$_{\text{FAR}}$ (left) and the CCA (right).

Figure 6.34.: Different views of droplet impaction Exp. 4 with the AIFM$_{\text{FAR}}$ and the CCA.

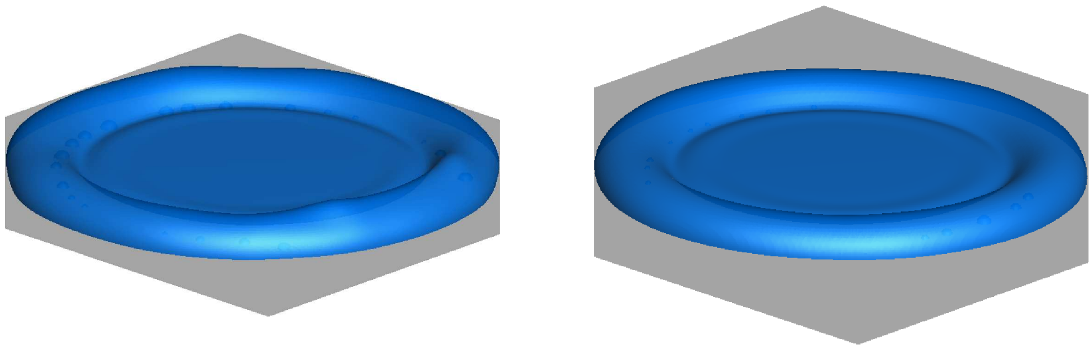


Figure 6.35.: Comparison of the micron-scale droplet (input parameters in mm) on the left and of a millimeter-scale droplet (input parameters in m) on the right. The millimeter-scale droplet shows less irregularities.

the droplet retains its symmetric shape much better as can be seen in Figure 6.34(c).

In general, the entrainment of air bubbles is a common phenomenon during droplet impact but has received scarce attention on the micron scale so far. For millimeter-scale droplets, these effects occur for large Reynolds numbers of about $Re = \mathcal{O}(1e^5)$ [Tho13]. While such droplets can develop during impact, the droplets with the AIFM rather indicate the accumulation of errors in the interface reconstruction process, enhanced by the implementation of the contact angle in the CLSVOF method: Since entrainment is less severe with a constant contact angle, one possible source of inaccuracy might stem from the changes in the contact angle values from one time step to the next. Then, these changes must be accounted for in the time-step size of our flow solver.

An additional source of errors is our rectangular flow domain, which can have a considerable influence on the droplet's symmetry. Due to the higher impact speed, the droplets in Exp. 3 and 4 spread further, so that less grid cells separate the free interface from the boundary. Then, the droplet slightly adapts to the edges of the domain and looks less round than before.

On a last note, our conversion from meters to millimeters for all input parameters can also have an adverse effect on the accuracy of our computations. Although the LS function remains well-defined due to the conversion, the liquid density and gas viscosity become smaller than $\mathcal{O}(10^{-8})$ which might lead to accuracy issues within our flow solver. Qualitative evidence for this effect can be found in the next subsection, where we compare micron- and millimeter-scale droplet impact in very similar settings. Here, the input parameters for the millimeter-scale droplet are not converted to mm but left in metre. In Figure 6.35, we see a comparison of such droplets. In this case, the millimeter-scale droplet displays less asymmetry than the micron-scale droplet, which supports our assumption that the prior conversion from mm to metre might be problematic.

To summarize, the irregularities at the interface cannot be traced back to a single cause at the moment. Instead, the mixture of changing contact angles and the accumulated interface reconstruction errors in the CLSVOF method, as well as the higher impact speed are all part of the problem. Further research will certainly shed light on these difficulties and, in a first step, all of these irregularities should improve with finer mesh sizes and finer time steps. Additionally, the CLSVOF method can certainly benefit from a 'smoother-in-time' implementation of the contact angle: For the level-set method, the contact angle is part of the boundary condition for the reinitialization equation, which takes several artificial time-steps to reach its equilibrium value, similar to the approach in [Sus01]. A further option is the method by Spelt [Spe05] where contact line position and contact angle are determined iteratively.

Conclusions

All in all, our results with the AIFM_{FAR} showed very good agreement with the practical experiments concerning the droplet shapes, droplet diameter and its height. As expected, the AIFM_{FAR} displayed considerable improvement as opposed to the CCA for all numerical experiments.

In particular, we anticipated good results by both contact angle approaches at early times after impact, when initial forces are dominant, which could be confirmed by the experiments. However, as already established in Subsection 6.4.1, the droplet diameter tends to be overestimated with the CCA.

Since the Weber number for Exp. 3 and 4 is considerably larger, we observed exceedingly good agreement of the AIFM$_{\text{FAR}}$ and the practical experiments. Here, the difference between the AIFM$_{\text{FAR}}$ and the CCA was less than for Exp. 1 and 2, where surface tensions forces are dominant. In all four experiments, the diameter/height curve computed by the AIFM$_{\text{FAR}}$ showed a similar development as the one by the practical experiment.

From a purely numerical point of view, the droplets' slight irregularities, air inclusions and developing asymmetries are to be further investigated.

6.4.4. Comparison of micron- and millimeter-scale impact

In this subsection, we simulate the millimeter-scale droplet impact of a water-glycerin mixture compared to the micron droplet Exp. 4.0 in the previous subsection. As Dong et al. write in [DCBM07, p. 2607], comparison between micron and millimeter droplet impact experiments are of particular interest for industrial applications due to the following reason:

> Most of the previous experimental studies of drop impaction on surfaces were conducted in the millimeter-scale regime. In inkjet printing and other applications, however, drop size ranges from several microns to several hundred microns. Therefore, the investigation of drop impactions at the application scale, i.e. the micron-scale, is needed to determine the applicability of rich millimeter drop results to micron drop impaction.

For both impact scenarios considered here, the same Ohnesorge (6.19) and Weber numbers (6.20) and approximate equilibrium contact angles (6.21) are chosen. Then, the evolving droplet shapes, heights and diameters are computed for comparison.

Note that in numerical simulations by the technique of non-dimensionalization, we can fully remove all units from the Navier-Stokes equations by a suitable substitution of variables. Then, a micron-scale drop impact and a millimeter-scale drop impact simulation only differ in the dimensionless numbers describing these processes. In part, these are the dimensionless numbers (6.19)–(6.21), which depend on the initial droplet diameter d_0, the initial velocity v_0 and the equilibrium contact angle θ_e. In the practical experiment, the droplet diameter and velocity, as well as the substrate are adapted in such a way that Oh, We and θ approximately coincide.

However, there are additional dimensionless numbers in this setting, which cannot be simply adapted in the practical experiment. Thus, the Froude number

$$Fr = \frac{v_0^2}{d_0 g}$$

denotes the ratio of inertia to gravitational forces, and the Bond number

$$Bo = \frac{\rho_l g d_0^2}{\tilde{\sigma}_{sg}}$$

final time:	$T = 0.03\,\mathrm{s}$
flow domain:	$\Omega = 0.008 \times 0.003 \times 0.008\,\mathrm{m}^3$
droplet radius:	$r = 1.115 \times 10^{-3}\,\mathrm{m}$
initial velocity:	$v_0 = 1.67\,\mathrm{m\,s^{-1}}$
material parameters:	see Table 6.21
body force:	$\boldsymbol{g} = (0.0, -9.8, 0.0)^\mathsf{T}\mathrm{m\,s^{-2}}$
contact angle:	99°
interface thickness:	$\epsilon = 1.9h$
grid resolution:	dof = $301 \times 113 \times 301$
employed method:	CLSVOF
contact angle model:	$\mathrm{AIFM_{FAR}}$
Parameters for the $\mathrm{AIFM_{FAR}}$	
dimensionless parameter:	$Sc = 5.0$
dimensionless surface tension (gas-solid):	$\tilde{\sigma}_{sg} = -0.07$
phenomenological constant:	$\rho_G^s = 0.54$

Table 6.25.: Parameters for glycerin/water (49.5:50.5) impacting on S107 with equilibrium contact angle 99°.

the ratio of gravitational to surface tension forces. The micron-scale as well as the millimeter-scale droplet impact differ considerably in these numbers. Loosely speaking, it is the influence of *Fr* and *Bo* that is studied in both the practical and the numerical experiment.

Definition of the numerical experiment

For the micron-scale impact, we consider water impacting on S107 with equilibrium contact angle 107°, $v_0 = 12.2\,\mathrm{m\,s^{-1}}$ and $d_0 = 50.5\,\mu\mathrm{m}$, which corresponds to Exp. 4.0 in Table 6.23. The millimeter drop impacts on the same substrate with $\theta_e = 99°$, $v_0 = 1.67\,\mathrm{m\,s^{-1}}$ and $d_0 = 2.23\,\mathrm{mm}$, which is $\mathrm{Exp_{mm}}$ in Table 6.23. Both droplets approximately have the same *We*, *Oh* and θ_e. In the following, we refer to Exp. 4.0 by $\mathrm{Exp_{\mu m}}$ for the purpose of comparison.

Our simulation parameters for the micron-scale drop impact are the same as in the previous section and are summarized in Table 6.24. The new simulation parameters for the millimeter-scale impact are given in Table 6.25. The overall setup of the simulation is the same, i.e. we use the $\mathrm{AIFM_{FAR}}$ and the CLSVOF method for the simulation.

In this subsection, we are particularly interested in how far the numerical simulation is able to capture the effect of drop size on the drop impaction process as observed in the experiments [DCBM07]. Beside *Fr* and *Bo*, there are still minor differences between both impact scenarios: The glycerin-water mixture used for the millimeter drop impaction has a slightly lower surface tension than the distilled water used for the micron drop impaction, and the contact angles are slightly smaller than those of the micron drops on the same substrate; confer Table 6.21.

In order to compare both simulations, we introduce the dimensionless time

$$t^* = \frac{t}{d_0/v_0} \tag{6.25}$$

with $d_0/v_0 = 4.14\,\mu\text{s}$ for $\text{Exp}_{\mu\text{m}}$ and $d_0/v_0 = 1.34\,\text{ms}$ for Exp_{mm} as in [DCBM07].

Note that Dong et al. conduct further experimental studies, on which they base their results. First, they study comparisons with lower Weber and higher Ohnesorge numbers, where droplets impact on three different substrates. For all substrates the contact angles for the millimeter drop are about 5 to 8 degrees higher than for the micron drop. Then, their last experiment corresponds partly to our setup: The parameters for the millimeter drop impact are the same. For the micron drop impact, however, the ratios of OH/CH_3 applied to the silicon wafer are adjusted until the contact angle is roughly the same as for the millimeter-scale impact, i.e. $\theta_e = 98°$.

We do not adapt the contact angle in our numerical experiment due to two reasons: First, the previous comparative practical experiments conducted in [DCBM07] showed very good agreement between the millimeter and micron-scale impact *without* adjustment of the substrate. Second, our numerical results for $\text{Exp}_{\mu\text{m}}$ are already very good, wherefore we expect very good results on the millimeter-scale as well and good grounds for a comparison of both impact scenarios.

Our numerical experiment is evaluated as follows:

1. We compare the evolution of the zero level-set contour (extracted with the VisIt software) of $\text{Exp}_{\mu\text{m}}$ and Exp_{mm} over the dimensionless time t^*.
2. For further quantitative comparison, we compute the dimensionless droplet height $H^* = h/d_0$ and the dimensionless diameter $D^* = d/d_0$ over time t^*, and compare the results of $\text{Exp}_{\mu\text{m}}$ and Exp_{mm} (cf. Subsec. 6.4.3 for a detailed explanation of how we measure h and d in our numerical experiment).
3. We compare H^* and D^* from $\text{Exp}_{\mu\text{m}}$ where $\theta_e = 107°$ to the practical experiment where $\theta_e = 98°$.
4. We compare H^* and D^* from Exp_{mm} to the corresponding practical experiment.

Unfortunately, we do not have the experimental data from [DCBM07] for this experiment as in the previous subsection. Instead, we present a figure from [DCBM07], where the dimensionless droplet height and diameter are plotted and which we use for comparison purposes.

Expected behavior

As already mentioned, gravity has a different effect on the millimeter- and on the micron-scale droplet impact. Let us quantify this effect by a computation of *Fr* and *Bo*. For Exp_{mm}, we have

$$Fr_{\text{mm}} = \frac{v_0^2}{d_0 g} \approx \frac{(1.67\,\text{im/s})^2}{2.23 \cdot 10^{-3}\,\text{m} \cdot 9.81\,\text{m}\,\text{s}^{-2}} \approx 127$$

$$Bo_{\text{mm}} = \frac{\rho_l g d_0^2}{\tilde{\sigma}_{sg}} \approx \frac{1113\,\text{kg}\,\text{m}^{-1}\,\text{s}^{-1} \cdot 9.81\,\text{m}\,\text{s}^{-2} \cdot (2.23 \cdot 10^{-3}\,\text{m})^2}{0.067\,\text{N}\,\text{m}^{-1}} \approx 0.81$$

and for $\mathrm{Exp}_{\mu\mathrm{m}}$

$$\begin{aligned} Fr_{\mu\mathrm{m}} &= \frac{v_0^2}{d_0 g} \approx \frac{(12.2\,\mathrm{m\,s^{-1}})^2}{5.05\cdot 10^{-5}\,\mathrm{m}\cdot 9.81\,\mathrm{m\,s^{-2}}} \approx 300441 \\ Bo_{\mu\mathrm{m}} &= \frac{\rho_l g d_0^2}{\tilde{\sigma}_{sg}} \approx \frac{1000\,\mathrm{kg\,m^{-3}}\cdot 9.81\,\mathrm{m\,s^{-2}}\cdot(5.05\cdot 10^{-5}\,\mathrm{m})^2}{0.072\,\mathrm{N\,m^{-1}}} \approx 0.000347. \end{aligned}$$

All of the involved experimental and material parameters can be found in Table 6.21 and 6.22.

By these computations, we see that the role of gravity in $\mathrm{Exp}_{\mathrm{mm}}$ is clearly of more significance than in $\mathrm{Exp}_{\mu\mathrm{m}}$, since the relative dominance of inertia as opposed to gravity is $\mathcal{O}(10^2)$ for the first and $\mathcal{O}(10^5)$ for the second. Similarly, the Bond number is 0.81 for $\mathrm{Exp}_{\mathrm{mm}}$ and only $\mathcal{O}(10^{-4})$ for $\mathrm{Exp}_{\mu\mathrm{m}}$. Bond numbers much smaller than 1 indicate that gravitational effects have next to no influence on the equilibrium shape of the droplet, which is the case for $\mathrm{Exp}_{\mu\mathrm{m}}$. In contrast, gravity as opposed to surface tension forces has a larger influence on the millimeter drop. From these differences, Dong et al. conclude that

a) The smaller *Fr* leads to a larger $D^*_{\max}$ for millimeter drops.
b) The larger *Bo* causes the equilibrium dimensionless contact line diameter to be larger for millimeter drops.

This is exactly what Dong et al. observe in their practical experiments, which they summarize as follows [DCBM07, p. 2616]:

> [...], with the same *We* and *Oh*, the millimeter and micron drop impactions on the same substrates exhibit very similar spreading, retraction and oscillation behavior, except that the micron-scale drops show a slightly lower D^* and retract more strongly from $D^*_{\max}$. During initial spreading, the behaviors of millimeter and micron drops are almost identical. When spreading approaches $D^*_{\max}$ the millimeter drops have a higher spreading ratio. The difference in $D^*_{\max}$ for millimeter drops, and the corresponding micron drops is less than 20% for most of the cases. The millimeter and micron drops start to recoil at almost the same dimensionless time, but the millimeter drop recoils less and more slowly than the corresponding micron drop.

Furthermore, they state that the subsequent oscillation cycles of the droplets are very close and that the final spreading ratio is slightly larger for the millimeter drops.

All in all, we expect similar results for our numerical simulation. However, due to the complexity of the simulation, we can only simulate the early stages of the oscillation cycles, while the computation of the final diameter would require about 8 times more simulation time than the 168 h already invested in the micron-scale simulation.

Furthermore, we do not adapt the contact angle for the micron-scale impaction in our numerical simulation. This decision might render the direct comparison with the practical experiment more difficult.

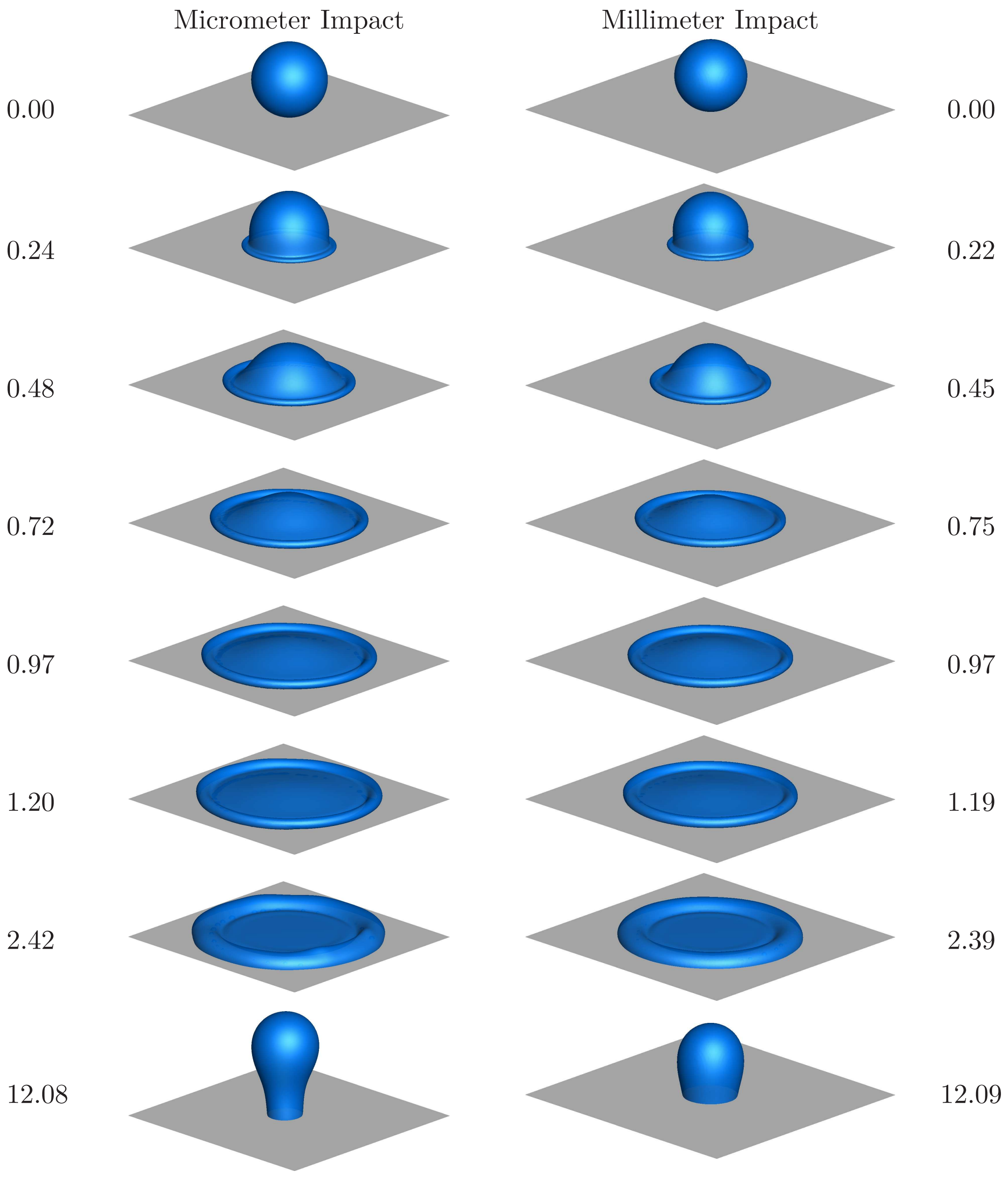


Figure 6.36.: Droplet shapes computed by the AIFM$_{\mathrm{FAR}}$ for Exp$_{\mu\mathrm{m}}$ (micrometer impact) and Exp$_{\mathrm{mm}}$ (millimeter impact) from Table 6.23. The micron-drop is enlarged to the size of the millimeter-drop. Note that the computational domain is slightly larger for the latter.

Discussion of results

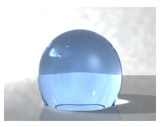
In Figure 6.36, we compare the simulation results of $\mathrm{Exp}_{\mathrm{\mu m}}$ and $\mathrm{Exp}_{\mathrm{mm}}$ on the dimensionless time scale $t^* = \frac{t}{d_0/v_0}$ with $d_0/v_0 = 4.14\,\mathrm{\mu s}$ for $\mathrm{Exp}_{\mathrm{\mu m}}$ and $d_0/v_0 = 1.34\,\mathrm{ms}$ for $\mathrm{Exp}_{\mathrm{mm}}$ as in [DCBM07]. The droplet shapes up to $t^* \approx 2.4$ are almost indistinguishable from each other. Then, the micron-scale droplet recedes much more strongly, takes the shape of a light bulb, and has a smaller diameter than the millimeter-scale droplet which takes an ellipsoidal shape at $t^* \approx 12$.

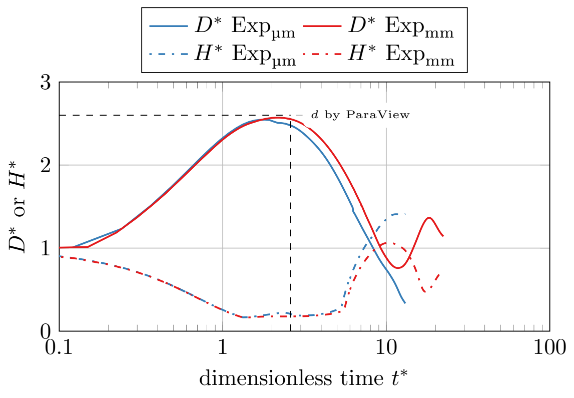
These observations are quantified in Figure 6.37(a), where we compute the dimensionless diameter $D^* = d/d_0$ and the droplet height $H^* = h/d_0$ over time t^*. Before the maximum diameter is reached, both simulation results agree perfectly concerning the droplet diameter. Then, D^*_{max} is slightly smaller in $\mathrm{Exp}_{\mathrm{\mu m}}$ and the subsequent evolution of the diameter remains below the one of $\mathrm{Exp}_{\mathrm{mm}}$ for the whole simulation time. The opposite holds true for the droplet height. Both simulations are in close to perfect agreement until $t^* = 5$. Then, the height measured in $\mathrm{Exp}_{\mathrm{\mu m}}$ is larger than the one in $\mathrm{Exp}_{\mathrm{mm}}$ until $t^* = 13$ where the micron-scale simulation ends.

Qualitatively, in the early stages of impact, our results agree very well with those by Dong et al., since we observe similar spreading and retraction behavior. As in the practical experiment, the micron-scale drop shows a slightly lower D^* and retracts more strongly from D^*_{max}. Our numerical results also support the identical behavior of millimeter and micron drops during initial spreading. However, the subsequent oscillation cycles of the droplets are not very close, at least not in their early stages that we simulate here. Due to the premature end of the simulation, we cannot yet make any assumptions about the final spreading ratio.

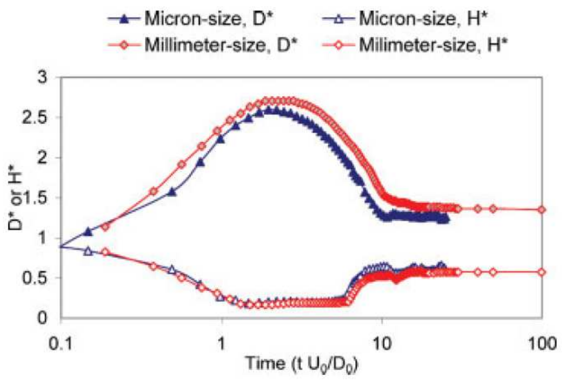
We saw in Subsection 6.4.3, that the micron-scale $\mathrm{Exp}_{\mathrm{\mu m}}$ agrees exceedingly well with the practical experiment. In a next step, we compare these results with the experimental ones in Figure 6.37(b). Note again that the underlying micrometer experiment slightly differs from $\mathrm{Exp}_{\mathrm{\mu m}}$, i.e. $\theta_e = 98°$ instead of $\theta_e = 107°$. Due to this small difference, there is now a much larger difference between the evolution of the droplet height and diameter of $\mathrm{Exp}_{\mathrm{\mu m}}$ (Fig. 6.37(a)) and the micron-scale experiment (Fig. 6.37(b)). Although D^*_{max} is approximately equal, our micron-scale droplet recedes much more strongly and reaches a diameter of about 0.3 at $t^* \approx 13$ as opposed to $D^* \approx 1$ in the practical experiment. The same holds true for the height of the droplet. At $t^* \approx 13$, we have $H^* \approx 1.4$ in $\mathrm{Exp}_{\mathrm{\mu m}}$ while this value is about 0.6 in Figure 6.37(b). We can only conclude that in this particular droplet impact scenario, the contact angle has an extra large effect on the developing droplet shapes. Probably this is also why Dong et al. felt the need to conduct a further micron-scale experiment with an adapted contact angle, which they did not do for all of their previous comparisons, where the angles on the micron- and on the millimeter-scale differed between 5 and 8 degrees.

Let us now compare the evolution of the droplet diameter and height of $\mathrm{Exp}_{\mathrm{mm}}$ (Fig. 6.37(a)) and the corresponding practical experiment drawn in red in Figure 6.37(b). Although we have the same θ_e in both experiments, the results only partly agree which is certainly not what we expected. Approximately, the same D^*_{max} is reached at about $t^* = 2.5$. Then, however, we have oscillations cycles in $\mathrm{Exp}_{\mathrm{mm}}$ with a local minimum of $D^* \approx 0.8$ at $t^* \approx 12$ and a local maximum of 1.4 at $t^* \approx 16$. In contrast, in the practical





(a) Numerical experiments. The dashed rectangle denotes the part where the diameter is computed by ParaView; see Fig. 6.29



(b) Practical experiments. Figure with permission from [DCBM07].

Figure 6.37.: Dimensionless droplet diameter D^* and height H^* vs. dimensionless time t^*. Comparison of millimeter (red) and mircon drop impaction (blue) at $We = 103 - 105$, $Oh = 0.0146 - 0.0148$. Note that the contact angle of the micron-drop is 98° in the practical experiments instead of 107° in the numerical experiments.

experiment, $D^* \in [1.2, 1.4]$ from $t^* \approx 12$ up to 100. Accordingly, for the droplet height, similar differences can be found in Figures 6.37(a) and 6.37(b). Here, we also have stronger oscillations in the numerical experiment as opposed to the practical one.

Note that these differences rather come as a surprise: In the previous two subsections, the numerical simulation of micron- *and* the millimeter-scale drop impactions with the AIFM$_{\text{FAR}}$ yielded results which were in high agreement with practical experiments. At this point, we cannot fully explain the observed discrepancies. Nevertheless, there are a variety of options for research which could shed light on this discrepancy.

Conclusions

To summarize, our results in this very last subsection are highly interesting but cannot be interpreted as straightforwardly as the previous numerical experiments.

All in all, our numerical comparison of micron- and millimeter-scale droplet impact showed many of the qualitative differences observed in the practical experiment, i.e. similar spreading behavior in the early stages, as well as a lower $D^*_{\max}$ and a stronger retraction of the micron droplet from this value.

Due to the differences observed between Exp$_{\mu\text{m}}$ with $\theta_e = 107°$ and the practical experiment with $\theta_e = 98°$, we now know that for this specific scenario, the contact angle has an extra large effect on the developing droplet shapes.

Similar to the micron-scale experiments, the numerical simulation of the millimeter droplet impact and the experiment differed considerably – despite the same contact angle. This is no reason to abandon the AIFM$_{\text{FAR}}$, which has proved valuable in droplet impact simulations on different scales in the previous subsections. One possible source of error for the millimeter-scale experiment might be our neglectfulness to adapt the empirical parameters in the AIFM$_{\text{FAR}}$ to the larger scale.

Although we cannot yet fully explain the behavior of the numerical simulation, we now have several opportunities to localize this discrepancy and to further study the comparison between micron- and millimeter-scale droplet impact.

1. We can study the micron-scale impact Exp$_{\mu\text{m}}$ with $\theta_e = 98°$ instead of 107°, i.e. with the same equilibrium angle as in the practical experiment of Figure 6.37(b). Thereby, we could check if our numerical model is able to capture the difference in the contact angle on the micron-scale.
2. If the change in the contact angle can be captured on the micron-scale, we need to study the influence of the empirical parameters in the AIFM$_{\text{FAR}}$ on the millimeter-scale as proposed in [Shi07].
3. In [DCBM07], further comparisons of droplet impact on the micron- and on the millimeter-scale are conducted for smaller *We* and higher *Oh* numbers. Here, the micron-scale impacts correspond to our already simulated Exp. 1 and Exp. 2 displayed in Table 6.23 and to a further impact of the same droplet on a mixture of $9:1$ OH/CH$_3$ on the gold coated silicon wafer with equilibrium contact angle 67°. In addition, we would have to simulate the according millimeter-scale droplet impacts. These additional experiments would present us with a further opportunity to study the cause of the observed difference between the millimeter-scale simulation and the practical experiment. On the fly, we could further investigate the ability of numerical simulations to capture the effect of droplet size on the impaction process as already partly done in this subsection.

6.5. Concluding remarks

In this chapter, we presented our results for the numerical simulation of dynamic wetting processes for the example of droplets impacting on substrates with different wetting characteristics. Both the implemented AIFM [Shi07] as well as the YCA [YVHH09] were able to produce droplet shapes which are extremely close to results from practical experiments. However, the YCA requires the special knowledge of advancing and receding contact angles, which is not always available beforehand. In contrast, the AIFM proved its general applicability for a vast variety of impact scenarios and its profound superiority over the simple CCA, where the contact angle is kept constant at all times.

Furthermore, this chapter validated our implemented CLSVOF, LS, LSL and LSG metod by a range of numerical benchmarks. We saw that both the LSL and the LSG method should be used with care and that of all considered numerical methods, the CLSVOF method performs best since it is well equipped to deal with severe topological changes and to maintain the mass of thin filaments developing on the scale of the mesh size. For all considered test cases and for droplet impact simulation in particular, the implemented CLSVOF method is a necessity: Since both volume correction methods are unreliable or unphysical and the LS method loses too much mass, the CLSVOF method presents the reliable and computationally highly efficient alternative.

7. Conclusion

Summary

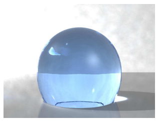
In this thesis, we coupled an asymptotic version of Shikhmurzaev's interface formation model (AIFM) with our three-dimensional incompressible two-phase Navier-Stokes solver NaSt3DGPF. Along with the CLSVOF method for the treatment of the free surface, this combination yielded a highly effective tool for the simulation of droplet impact in three spatial dimensions, which we put to the test for a large range of experiments. The AIFM had never before been employed in 3D.

Additionally, we implemented an approach by Yokoi et al. and simulated the specific droplet experiment that this approach was designed for. The resulting droplet shapes were very close to those of the experiments and the previous two-dimensional results. For Shikhmurzaev's reduced interface formation model, we used Moffatt's solution for the radial velocity on the one hand, and a far field velocity value near the contact line but within the bulk flow on the other hand. The droplet shapes computed with both of these models were very close to each other and in high agreement with practical experiments as also confirmed by the computation of the droplet diameter over time.

For the impact of droplets on the micron-scale, we could not use Yokoi's approach, which cannot be easily adapted for other experiments since it requires further knowledge about experimental parameters which are not always available. Here, the AIFM yielded impressive results for different impact speeds and wetting characteristics compared with practical experiments. Both the droplet diameter and droplet height were very close to experimental measurements and the AIFM clearly outperformed the constant angle approach (CCA), where the dynamic contact angle was fixed to its equilibrium value for the whole simulation.

In this thesis, we also proposed, proved and implemented a contact angle boundary condition for the LS method. This condition links our two dynamic contact angle models and the flow solver. Our prescription of the contact angle works exemplary in the reinitialization of the LS function with pseudo time-stepping, while its application in the reinitialization of the CLSVOF method, where no time-stepping is employed, led to increased numerical diffusion at the contact line for droplets with high impact speed. All in all, our very good numerical results with both the CLSVOF and the LS method depended considerably on the shape of the interface at the contact line. Therefore, we strongly infer the validity of our implemented boundary condition.

Since dynamic wetting simulations require excellent mass conservation properties of our flow solver, we tested two volume correction methods for our LS method. However, both of these methods yielded unphysical results even for the most simple advection tests. Unexpectedly, the LS method with local volume correction (LSL) could not conserve any mass at all for the case of 2D vortex shearing. Furthermore, for the very highly resolved rising bubble, the LSL showed unphysical oscillations in the curvature as already described in the literature [Her05, Kec98]. The global volume correction (LSG), in our implementation,



underestimates the loss of mass – depending on the number of grid cells which contain an interface. Additionally, the LSG inherently adds mass everywhere, while the main loss of mass occurs most prominently in under-resolved regions of our flow domain. Both methods, however, showed results comparable to those of the LS method for a droplet impact simulation with improved mass conservation. All in all, both volume correction methods are useful for particular problems, where topological deformations are not too intense. From a physical and numerical point of view, both methods must be regarded as highly unreliable.

Due to the failure of the volume correction methods, we implemented the CLSVOF method, which proved to be faster than our previous higher-order LS method and conserved mass excellently even on coarse grids. In our description of the CLSVOF method, we included all details of the implementation so that our chapter on this method can be used as a comprehensive guide for an implementation into existing LS Navier-Stokes solvers, which has not been available so far. Additionally, we presented a new technique for the reinitialization of the LS function within the CLSVOF method and showed that an existing approach by Wang et al. [WYKS09] fails in three dimensions. Furthermore, we also addressed the details of parallelization, which is neglected in the standard literature. Here, we focused on the parallelization of our new vertex finding algorithm, which is a key part of the reinitialization whose parallelization is not straightforward.

In addition to the LS method and both volume correction methods, the CLSVOF method was thoroughly validated. Starting with basic advection tests in two and three dimensions, we then included the whole flow solver for the rising bubble benchmark. Finally, all methods were also tested for one droplet impact simulation example. For the convergence of mass, we obtained a convergence order of 1.5–2 for the CLSVOF and an order of 0.5–2 for the LS method, depending on the problem. In all cases, the CLSVOF method showed superb mass conservation properties and performed computations even faster than our high-order LS method whose reinitialization is very costly. For the further micron drop impact simulations, the CLSVOF method proved to be well equipped to track the interface with the contact angles computed by the AIFM.

All in all, the interface formation model in its asymptotic formulation in conjunction with the CLSVOF method for the treatment of the free interface confirmed to be a high quality and easily adaptable tool for the computation of dynamic contact angles for droplet impact simulations.

Outlook

So far, our results are already a significant improvement over other approaches tackling the moving contact line problem. However, there is always room for further research and for the respective mathematical and numerical extensions.

A first natural extension of the implemented AIFM is the computation of the contact angle in every grid cell along the contact line. So far, for our simulations of droplet impact, the assumption of axisymmetric spreading is valid. However, aiming at simulations of further wetting processes such as curtain coating, the AIFM has to be applied at the whole contact line. An example of a curtain coating simulation is given in Figure 7.1 where we employed the CCA and used the CLSVOF method for interface localization.

With respect to the AIFM, the implementation of the whole interface formation model poses an interesting challenge, since the extension of conventional flow solvers to incorporate

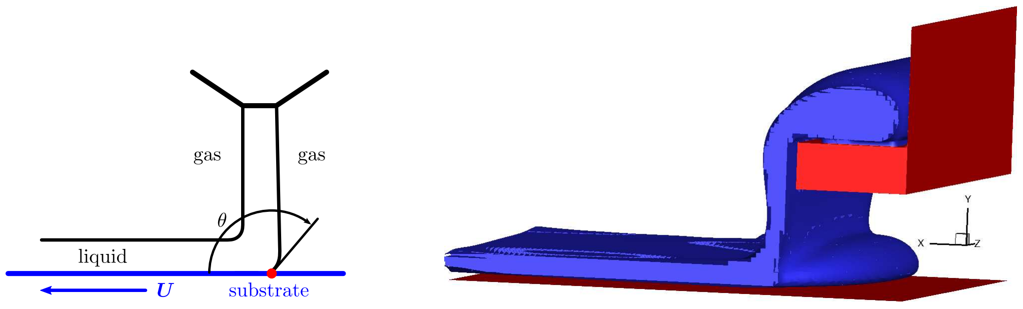


Figure 7.1.: Simulation of curtain coating with the CCA, $\theta_d = 90°$, and the CLSVOF method.

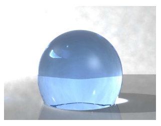
the full interface formation model is far from trivial [SS13]. Thereby, we have to solve the Navier-Stokes equations for the bulk flow along with the interface formation equations at the free surface and at the liquid-solid interface: These equations are themselves partial differential equations and have to be complemented with boundary conditions at the contact line. By the solution of these equations, the dynamic contact angle is determined. In turn, the dynamic contact angle defines the shape of the free surface, which has a large influence on the bulk flow. So far, the full model has also only been used to study axisymmetric drop impact [SS12b] and coalescence [SS12a], but has not yet been implemented in 3D.

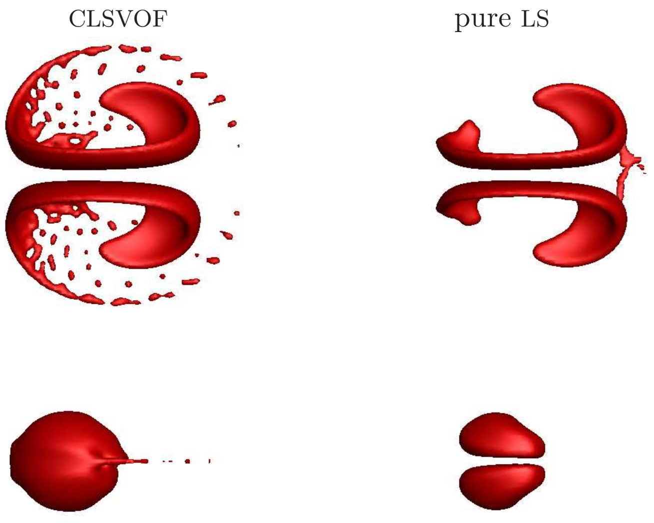
Due to the complexity of the full model, a reasonable continuation of this thesis would be to study further wetting simulations with the AIFM first. These can then be compared with 2D results from [SS12b] and [SS12a]. Thus, we can identify regimes, where the asymptotic approach yields as good results as the full model. These regimes can be found due to the underlying physics (small Capillary and Reynolds number), but also when, despite mesh refinement, the numerical error dominates the modeling one. If vast differences between both models occur, the implementation of the full model can still be tackled. A further, very simple improvement will stem from the use of the Navier-slip condition 2.1 instead of the no-slip one, which we kept so far for the purpose of comparison with Yokoi's approach. At the moment, if our computational mesh is refined, the contact line motion slows down due to the no-slip condition, which also influences the computed contact angle. In the limit, the contact line motion would stop altogether. Note that a Navier-slip condition is already implemented in NaSt3DGPF.

Both volume corrections proved to be unreliable and shall be abandoned in the future. Instead, the CLSVOF method is a most valuable alternative and there are many possibilities for its further improvement.

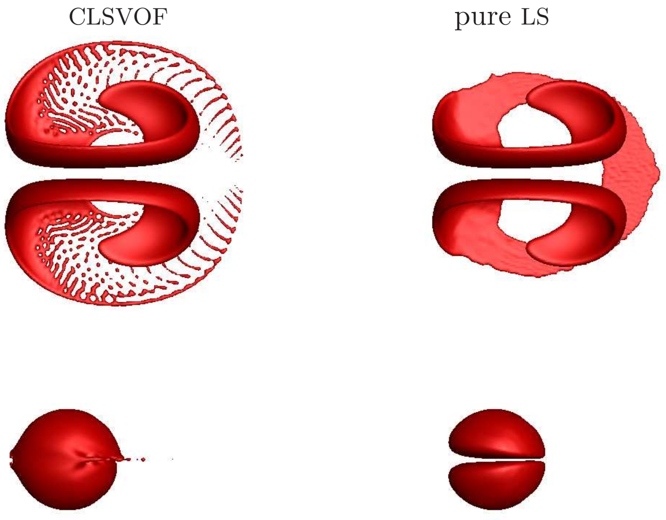
In a first step, our contact angle boundary condition, which seems to introduce numerical diffusion at the contact line within the CLSVOF method has to be reconsidered. Possibly, the effect of drastic changes in the contact angle affects the stability of our numerical scheme. Then, these angles should be taken into account for the computation of the time step.

A further improvement, which affects both the LS and the CLSVOF method, is the ghost-fluid method to accurately capture the sharp discontinuities of pressure, density and viscosity at the interface. This method is already partly implemented in NaSt3DGPF and has to be extended to handle the pressure jump and the special treatment at contact lines.





(a) $h = 1/128$



(b) $h = 1/256$

Figure 7.2.: Three-dimensional sphere deformation with CLSVOF (left) and LS method (right) for two grids with mesh widths 1/128 and 1/256. For both resolutions, $t = 2.0$ in the top and $t = 4.0$ in the bottom row. Ideally at time $t = 4.0$, the deformed sphere should have returned to its original shape and location.

Although the presently used continuum surface force method is stable, a sharp interface approach avoids the introduction of fictitious interface thickness and promises a more accurate discretization of discontinuous terms as well as the reduction of parasitic currents.

When we presented the numerical results of the CLSVOF method, we also compared them to those from the literature. Thus, the CLSVOF method can always be adapted and improved to fit to a specific simulation purpose. Ménard, Tanguy and Berlemont [MTB07] specifically enhance the interface reconstruction process to handle very thin filaments for the simulation of the primary break-up of liquid jets. In [WYS08], a second-order Lagrangian method is chosen for the transport of the volume fractions for the simulation of liquid-liquid drop impact and of plunging breaking waves. Furthermore, unsplit methods for the transport of the LS and the VOF function are considered in [LRL$^+$06]. In an unsplit method, the interface has to be reconstructed only once, which is invaluable if we aim at expensive higher-order interface reconstruction schemes in three-dimensions.

A further development of the CLSVOF method is the coupled level-set and moment-of-fluid (CLSMOF) method, which uses next to the LS and the VOF function also a reference centroid in order to produce a slope and an intercept for the local reconstruction of the interface [JLS$^+$13]. Jemison et al. claim similar accuracy of the CLSMOF and the CLSVOF method for many test problems and a better preserved interface with the CLSMOF method for 3D problems with deforming, stretching or disintegrating interfaces. In parts, the CLSMOF implementation is based on the CLSVOF method, which we described extensively in this thesis. Therefore, our description also offers a natural access to the CLSMOF method, which then, additionally, requires the computation of the centers of mass in each computational cell.

Considering all of these possible improvements, the advection tests used in this thesis can also be made harder. Let us demonstrate this for the three-dimensional deformation of a sphere described in Subsection 6.2.2. Here, at the highest grid resolution, the difference between the LS and the CLSVOF method is hardly visible. To take things to the next level, the final time T could be set from $T = 3$ to $T = 4$. Then, the flow reverses at $T = 2$, which results in more stretching and an even thinner membrane. In Figure 7.2, contours computed by the CLSVOF and the LS method are shown on the two grids with mesh sizes $1/128$ and $1/256$. The deformed shapes at $t = 2.0$ correspond to maximum stretching, while at $t = 4.0$ the flow has returned the shape back to its initial position. Both the CLSVOF and the LS method are unable to resolve the thin membrane at maximum stretching, and the final shape with the LS method is split for both grid resolutions. Hence, this much harder test case is a challenge for even the best interface capturing schemes and thereby offers a good starting point for the testing of further improvements.

To make a last note, the CLSVOF method is not only a valuable tool for the simulation of flows with moving contact lines but for two-phase flows in general. Therefore, all flow simulations with NaSt3DGPF can now be easily and considerably improved due to the implemented CLSVOF method. Our simulations range from interaction of flows with rigid bodies [Cro10], non-Newtonian fluids [GR14] and fluid animation [Zas09, ZG11] to current driven sediment transport processes [Bur10, BG13]. For the latter, we exemplify the contribution of the CLSVOF method in Figure 7.3, where a rain drop impacts on an inclined bed of sediment. Compared to the LS, the CLSVOF method shows improved mass conservation and highly realistic splashing effects during impact.

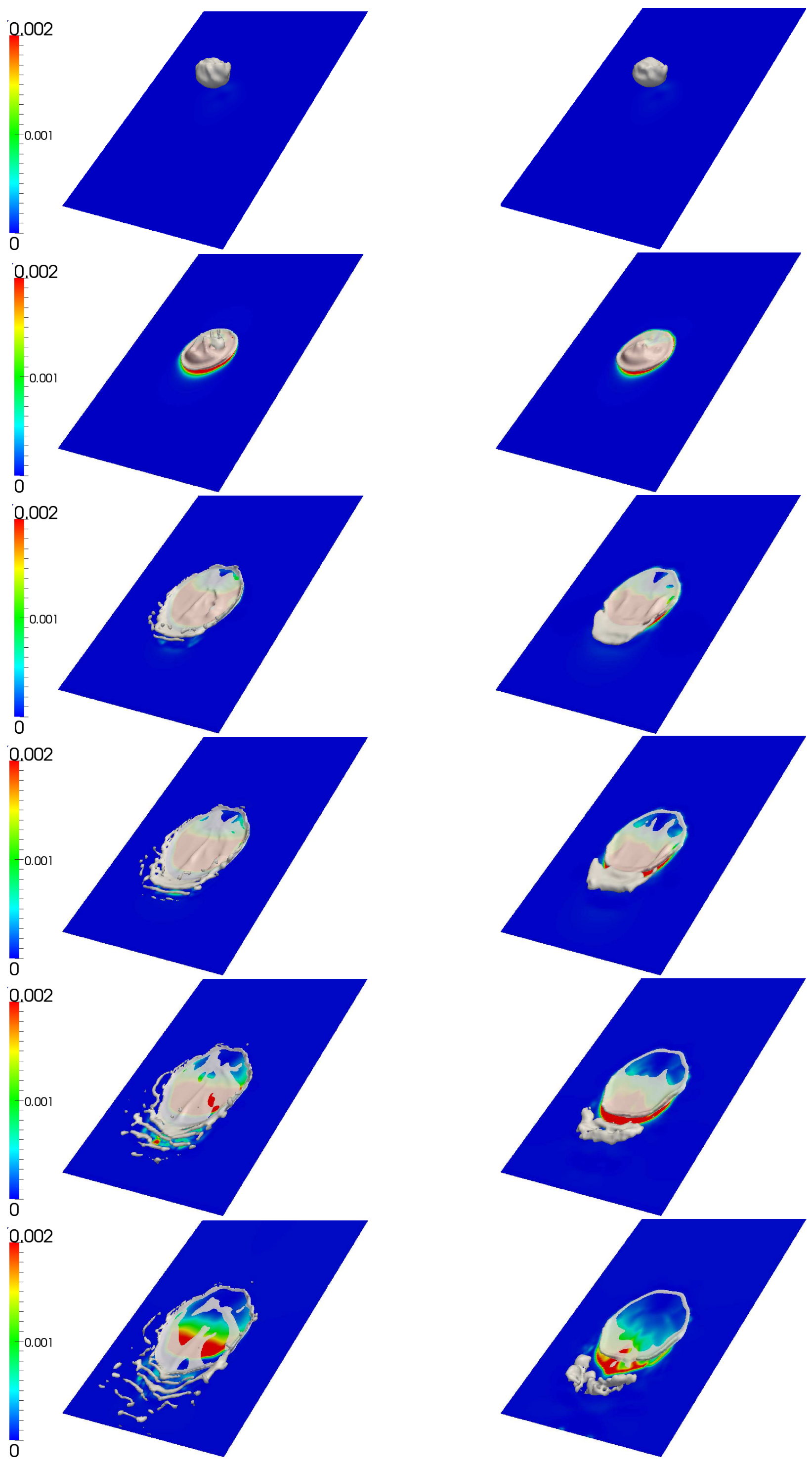


Figure 7.3.: Simulation of rain drop impact on an inclined bed of sand with the CLSVOF method (left) and the LS method (right). The colors denote the shear stress acting from the fluid on the sediment.

In summary, our contributions in this thesis are a very important step towards three-dimensional simulations of wetting problems with accurate dynamic contact angles, and many exciting further developments can yet be done.

Acknowledgements

First of all, I would like to thank my adviser Prof. Dr. Michael Griebel for his helpful ideas and support in the development of this thesis. Especially, the joint composition of articles and research proposals turned out to be an invaluable lesson in the details of scientific writing. The working conditions at the Institute for Numerical Simulation are excellent. Furthermore, I would like to thank the committee member Prof. Dr. Marc Alexander Schweitzer for providing the second opinion on this dissertation and his constructive advice on many mathematical and numerical problems during my time at the INS.

I would also like to thank both Dr. Roberto Croce and Dr. Martin Engel, who introduced me to the field of fluid dynamics and thereby laid the foundation for this thesis.

Thanks to all authors who allowed me to use figures of their work in my thesis. In particular, I want to thank Dr. Kensuke Yokoi and Dr. Hongming Dong for providing experimental results of the droplet impacts and the permission to use the corresponding pictures in this thesis. My sincere gratitude goes to Dr. James Sprittles for innumerable interesting discussions about the interface formation model and the modeling of dynamic wetting problems in general.

Christian Gütter, Dr. Frederik Heber and Alexander Rüttgers all read parts of this thesis and offered practical and constructive criticism for which I thank them most gratefully. Dr. Jutta Neuen read nearly all of it, and I cannot thank her enough. Most of the diagrams in this thesis would look only a quarter as good without you.

Vielen Dank an alle, die sich um meine Kinder gekümmert haben, damit ich diese Arbeit vollenden konnte. Mein Vater und meine Tante Heidi haben mich und meine Familie ganz wunderbar unterstützt. Danke, Jutta und Christian, auf Euch ist immer Verlass. Vielen Dank, liebe Helga, im Besonderen.

Last but not least, I thank my husband Alexander for his unconditional support and encouragement – and my daughters Johanne, Sophia and Elisabeth for their uncanny ability to distract me when I needed it the most.

A. Abbreviations

AIFM	Asymptotic Interface Formation Model
AIFM_{MOF}	Asymptotic Interface Formation Model with Moffatt's solution
AIFM_{FAR}	Asymptotic Interface Formation Model with an arbitrary far field velocity
ALE	Arbitrary Lagrangian-Eulerian
AMG	Algebraic Multigrid
CCA	Constant Contact Angle approach
CLSMOF	Coupled Level-Set Moment-Of-Fluid
CLSVOF	Coupled Level-Set Volume-Of-Fluid
CSF	Continuum Surface Force
DROPS	Navier-Stokes solver developed at the Chair of Numerical Mathematics and the Institute of Scientific Computing at the RWTH Aachen [DRO08]
E1	Experiment 1
E2	Experiment 2
HLPA	Hybrid Linear/Parabolic Approximation
IFM	Interface Formation Model
INS	Institute for Numerical Simulation, Bonn University
LS	Level-Set
LSL	Level-Set with Local volume correction
LSG	Level-Set with Global volume correction
MAC	Marker And Cell
MD	Molecular Dynamics
MPI	Message Passing Interface [MPI14]
NaSt3DGPF	Navier-Stokes solver developed at the Institute for Numerical Simulation, Bonn University [GDN98, NaS]

OpenFOAM	Open source Field Operation And Manipulation flow solver [Ope13]
PLIC	Piecewise Linear Interface Calculation
PLS	Particle Level-Set
QUICK	Quadratic Upstream Interpolation for Convective Kinematics
S31	Substrate consisting of a thermally oxidized silicon wafer
S107	Substrate consisting of 100% CH_3 in three self-assembled monolayers on a gold coated silicon wafer
SLIC	Simple Line Interface Calculation
SMAC	Simplified Marker and Cell
SMART	Sharp and Monotonic Algorithm for Realistic Transport
VOF	Volume-Of-Fluid
VONOS	Variable-Order Non-Oscillatory Scheme
VTK	Visualization Tool Kit
WENO	Weighted Essentially Non-Oscillatory
YCA	Yokoi's Contact Angle approach

Bibliography

[Ade14] Adelsberger, J.: *Cut cell methods in global atmospheric dynamics*. PhD thesis, Institut für Numerische Simulation, Universität Bonn, 2014.

[AEG+14] Adelsberger, J., P. Esser, M. Griebel, S. Groß, M. Klitz, and A. Rüttgers: *3D incompressible two-phase flow benchmark computations for rising droplets*, 2014. Proceedings of the 11th World Congress on Computational Mechanics, Barcelona, Spain, also available as INS Preprint No. 1401.

[AH70] Amsden, A. A. and F. Harlow: *A simplified MAC technique for incompressible fluid flow calculations*. Journal of Computational Physics, 6(2):322–325, 1970.

[BBS99] Blake, T. D., M. Bracke, and Y. D. Shikhmurzaev: *Experimental evidence of nonlocal hydrodynamic influence on the dynamic contact angle*. Physics of Fluids, 11:1995–2007, 1999.

[BEI+09] Bonn, D., J. Eggers, J. Indekeu, J. Meunier, and E. Rolley: *Wetting and spreading*. Reviews of Modern Physics, 81:739–805, May 2009.

[BG13] Burkow, M. and M. Griebel: *Numerical simulation of sediment processes and resulting bedforms - application to scouring at an obstacle*. Technical report, Institute for Numerical Simulation, Bonn University, 2013. Also available as INS Preprint No. 1307.

[BKZ92] Brackbill, J. U., D. B. Kothe, and C. Zemach: *A continuum method for modeling surface tension*. Journal of Computational Physics, 100:335–354, 1992.

[Bla06] Blake, T. D.: *The physics of moving wetting lines*. Journal of Colloid and Interface Science, 299(1):1–13, 2006.

[Bol11] Bolleddula, D. A.: *Droplet impact and spreading of viscous dispersions and volatile solutions*. PhD thesis, University of Washington, 2011.

[BPS13] Bänsch, E., J. Paul, and A. Schmidt: *An ALE finite element method for a coupled Stefan problem and Navier-Stokes equations with free capillary surface*. International Journal for Numerical Methods in Fluids, 71(10):1282–1296, 2013.

[BS02] Blake, T. D. and Y. D. Shikhmurzaev: *Dynamic wetting by liquids of different viscosity*. Journal of Colloid and Interface Science, 253(1):196–202, 2002.

[Bur10] Burkow, M.: *Numerische Simulation strömungsbedingten Sedimenttransports und der entstehenden Gerinnebettformen*. Diplomarbeit, Institut für Numerische Simulation, Universität Bonn, 2010.

[CBW$^+$12] Childs, H., E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. Miller, C. Harrison, G. H. Weber, H. Krishnan, T. Fogal, A. Sanderson, C. Garth, E. W. Bethel, D. Camp, O. Rübel, M. Durant, J. Favre, and P. Navrátil: *VisIt: An end-user tool for visualizing and analyzing very large data.* In *High Performance Visualization–Enabling Extreme-Scale Scientific Insight*, pages 357–372. Oct 2012.

[CGS04] Croce, R., M. Griebel, and M. A. Schweitzer: *A parallel level-set approach for two-phase flow problems with surface tension in three space dimensions.* Technical report, Universität Bonn, 2004. Preprint 157, Sonderforschungsbereich 611.

[CGS10] Croce, R., M. Griebel, and M. A. Schweitzer: *Numerical simulation of bubble and droplet deformation by a level set approach with surface tension in three dimensions.* International Journal for Numerical Methods in Fluids, 62(9):963–993, 2010.

[CGW78] Clift, R., J. R. Grace, and M. E. Weber: *Bubbles, drops and particles.* Academic Press, New York, London, 1978.

[Chi07] Chiquete, C.: *Not even Hercules: The moving contact line problem.* Technical report, University of Arizona, 2007.

[CHMO96] Chang, Y. C., T. Y. Hou, B. Merriman, and S. Osher: *A level set formulation of Eulerian interface capturing methods for incompressible fluid flows.* Journal of Computational Physics, 124(2):449–464, 1996.

[Cla95] Clarke, A.: *The application of particle tracking velocimetry and flow visualisation to curtain coating.* Chemical Engineering Science, 50(15):2397–2407, 1995.

[Cro02] Croce, R.: *Ein paralleler, dreidimensionaler Navier-Stokes-Löser für inkompressible Zweiphasenströmungen mit Oberflächenspannung, Hindernissen und dynamischen Kontaktflächen.* Diplomarbeit, Institut für Angewandte Mathematik, Universität Bonn, 2002.

[Cro10] Croce, R.: *Numerische Simulation der Interaktion von inkompressiblen Zweiphasenströmungen mit Starrkörpern in drei Raumdimensionen.* Dissertation, Institut für Numerische Simulation, Universität Bonn, 2010.

[Dal69] Daly, B. J.: *Numerical study of the effect of surface tension on interface instability.* The Physics of Fluids, 12(7):1340–1354, 1969.

[DCBM07] Dong, H., W. W. Carr, D. G. Bucknall, and J. F. Morris: *Temporally-resolved inkjet drop impaction on surfaces.* AIChE Journal, 53(10):2606–2617, 2007.

[DD74] Dussan, E. B. and S. H. Davis: *On the motion of a fluid-fluid interface along a solid surface.* Journal of Fluid Mechanics, 65(01):71–95, 1974.

[DE07] Dziuk, G. and C. M. Elliott: *Finite elements on evolving surfaces.* IMA Journal of Numerical Analysis, 27(2):262–292, 2007.

[DeB74] DeBar, R.: *Fundamentals of the KRAKEN code.* Technical report, Lawrence Livermore National Laboratory, 1974.

[Dec06] Decent, S. P.: *Hydrodynamic assist and the dynamic contact angle in the coalescence of liquid drops.* IMA Journal of Applied Mathematics, 71(5):740–767, 2006.

[DFG$^+$06] Du, J., B. Fix, J. Glimm, X. Jia, X. Li, Y. Li, and L. Wu: *A simple package for front tracking.* Journal of Computational Physics, 213(2):613–628, 2006.

[DL10] Dupont, J. B. and D. Legendre: *Numerical simulation of static and sliding drop with contact angle hysteresis.* Journal of Computational Physics, 229(7):2453–2478, 2010.

[DRO08] DROPS: *DROPS package for simulation of two-phase flows*, 2008. `http://www.igpm.rwth-aachen.de/DROPS/`.

[EFFM02] Enright, D., R. Fedkiw, J. Ferziger, and I. Mitchell: *A hybrid particle level set method for improved interface capturing.* Journal of Computational Physics, 183(1):83–116, 2002.

[ES04] Eggers, J. and H. A. Stones: *Characteristic lengths at moving contact lines for a perfectly wetting fluid: The influence of speed on the dynamic contact angle.* Journal of Fluid Mechanics, 505:309–321, 2004.

[FAJ$^+$09] Fuster, D., G. Agbaglah, C. Josserand, S. Popinet, and S. Zaleski: *Numerical simulation of droplets, bubbles and waves: State of the art.* Fluid Dynamics Research, 41(6), 2009.

[Fef00] Fefferman, C. L.: *Existence and smoothness of the Navier-Stokes equation.* The millennium prize problems, pages 57–67, 2000.

[FHW$^+$08] Fang, C., C. Hidrovo, F. Wang, J. Eaton, and K. Goodson: *3-D numerical simulation of contact angle hysteresis for microscale two phase flow.* International Journal of Multiphase Flow, 34(7):690–705, 2008.

[Fra02] François, M.: *Computations of drop dynamics with heat transfer.* PhD thesis, University of Florida, 2002.

[GDN98] Griebel, M., T. Dornseifer, and T. Neunhoeffer: *Numerical simulation in fluid dynamics: A practical introduction*, volume 3. SIAM, Philadelphia, 1998.

[GGG$^+$98] Glimm, J., M. J. Graham, J. Grove, X. L. Li, T. M. Smith, D. Tan, F. Tangerman, and Q. Zhang: *Front tracking in two and three dimensions.* Computers & Mathematics with Applications, 35(7):1–11, 1998.

[GH91] Goodwin, R. and G. M. Homsy: *Viscous flow down a slope in the vicinity of a contact line.* Physics of Fluids A, 3(4):515–528, 1991.

[GK13] Griebel, M. and M. Klitz: *Simulation of droplet impact with dynamic contact angle boundary conditions.* In *Singular Phenomena and Scaling in Mathematical Models*, pages 297–325. Springer International Publishing Switzerland, 2013. Also available as INS Preprint No. 1302.

[Gla06] Glasner, K. B.: *Homogenization of contact line dynamics.* Interfaces and Free Boundaries, 8(4):523–542, 2006.

[GLN$^+$99] Gueyffier, D., J. Li, A. Nadim, R. Scardovelli, and S. Zaleski: *Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows.* Journal of Computational Physics, 152(2):423–456, 1999.

[GMOS06] Griebel, M., B. Metsch, D. Oeltz, and M. A. Schweitzer: *Coarse grid classification: A parallel coarsening scheme for algebraic multigrid methods.* Numerical Linear Algebra with Applications, 13(2–3):193–214, 2006. Also available as SFB 611 preprint No. 225, Universität Bonn, 2005.

[GMS08] Griebel, M., B. Metsch, and M. A. Schweitzer: *Coarse grid classification: AMG on parallel computers.* In Münster, G., D. Wolf, and M. Kremer (editors): *NIC Symposium 2008*, volume 39 of *NIC Series*, pages 299–306, February 2008. Also available as SFB 611 preprint No. 368, Universität Bonn, 2008.

[GR11] Groß, S. and A. Reusken: *Numerical methods for two-phase incompressible flows.* Springer, 2011.

[GR14] Griebel, M. and A. Rüttgers: *Multiscale simulations of three-dimensional viscoelastic flows in a square-square contraction.* Journal of non-Newtonian Fluid Mechanics, 205:41–63, 2014. Also available as INS Preprint No. 1313.

[Gro08] Groß, Sven: *Numerical methods for three-dimensional incompressible two-phase flow problems.* PhD thesis, IGPM, RWTH Aachen, 2008.

[GZ10] Griebel, M. and P. Zaspel: *A multi-GPU accelerated solver for the three-dimensional two-phase incompressible Navier-Stokes equations.* Computer Science – Research and Development, 25(1–2):65–73, 2010.

[Had99] Hadjiconstantinou, N. G.: *Hybrid atomistic-continuum formulations and the moving contact-line problem.* Journal of Computational Physics, 154(2):245–265, 1999.

[Her05] Herrmann, M.: *Refined level set grid method for tracking interfaces.* Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford University, pages 3–18, 2005.

[HN81] Hirt, C. W. and B. D. Nichols: *Volume of fluid VOF method for the dynamics of free boundaries.* Journal of Computational Physics, 39(1):201–225, 1981.

[Hoc77] Hocking, L. M.: *A moving fluid interface. Part 2. The removal of the force singularity by a slip flow.* Journal of Fluid Mechanics, 79(2):209–229, 1977.

[HP90] Ho, L. W. and A. T. Patera: *A Legendre spectral element method for simulation of unsteady incompressible viscous free-surface flows.* Computer Methods in Applied Mechanics and Engineering, 80(1):355–366, 1990.

[HS71] Huh, C. and L. E. Scriven: *Hydrodynamic model of steady movement of a solid/liquid/fluid contact line.* Journal of Colloid and Interface Science, 35(1):85–101, 1971.

[HTK$^+$09] Hysing, S., S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, and L. Tobiska: *Quantitative benchmark computations of two-dimensional bubble dynamics.* International Journal for Numerical Methods in Fluids, 60(11):1259–1288, 2009.

[HW65] Harlow, F. H. and J. E. Welch: *Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface.* The Physics of Fluids, 8(12):2182–2189, 1965.

[Hym84] Hyman, J. M.: *Numerical methods for tracking interfaces.* Physica D: Nonlinear phenomena, 12(1–3):396–407, 1984.

[INS13] INSGrid: *High-Performance Cluster Computers at the INS and SFB 1060*, 2013. `http://wissrech.ins.uni-bonn.de/research/atacama/`.

[Jak08] Jakobsen, H. A.: *Chemical reactor modeling.* Springer, 2008.

[JLS$^+$13] Jemison, M., E. Loch, M. Sussman, M. Shashkov, M. Arienti, M. Ohta, and Y. Wang: *A coupled level set-moment of fluid method for incompressible two-phase flows.* Journal of Scientific Computing, 54(2–3):454–491, 2013.

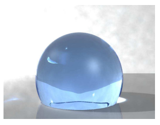
[Kec98] Keck, R.: *Reinitialization for level-set methods.* Diplomarbeit, Fachbereich Mathematik der Universität Kaiserslautern, 1998.

[KL04] Kim, J. and J. Lowengrub: *Interfaces and multicomponent fluids.* Technical report, Department of Mathematics, University of California, Irvine, USA, 2004.

[Kli06] Klitz, M.: *Homogenised fluid flow equations in porous media with application to permeability computations in textiles.* Diplomarbeit, Institut für Numerische Simulation, Universität Bonn, 2006.

[KNP$^+$13] Kalliadasis, S., A. Nold, B. Pereira, A.and Goddard, D. Sibley, M. Pradas, M. Schmuck, N. Savva, P. Yatsyshin, and R. Vellingiri: *Recent progress on the moving contact line problem.* Cambridge, 2013. Isaac Newton Institute for Mathematical Sciences, Presented in the summer school: Mathematical approaches to complex fluids.

[LC87] Lorensen, W. E. and H. E. Cline: *Marching cubes: A high resolution 3d surface construction algorithm.* In *ACM Siggraph Computer Graphics*, volume 21, pages 163–169. ACM, 1987.



[Le05] Le, D. V.: *An immersed interface method for solving viscous incompressible flows involving rigid and flexible boundaries.* PhD thesis, HCMC University of Technology, 2005.

[Ler34] Leray, J.: *Sur le mouvement d'un liquide visqueux emplissant l'espace.* Acta mathematica, 63(1):193–248, 1934.

[LeV96] LeVeque, R. J.: *High-resolution conservative algorithms for advection in incompressible flow.* SIAM Journal on Numerical Analysis, 33(2):627–665, 1996.

[LFO06] Losasso, F., R. Fedkiw, and S. Osher: *Spatially adaptive techniques for level set methods and incompressible flow.* Computers & Fluids, 35(10):995–1010, 2006.

[LNS$^+$94] Lafaurie, B., C. Nardone, R. Scardovelli, S. Zaleski, and G. Zanetti: *Modelling merging and fragmentation in multiphase flows with SURFER.* Journal of Computational Physics, 113(1):134–147, 1994.

[LNY11] Liu, J., N. T. Nguyen, and Y. F. Yap: *Numerical studies of sessile droplet shape with moving contact lines.* Micro and Nanosystems, 3(1):56–64, 2011.

[LRL$^+$06] Liovic, P., M. Rudman, J. L. Liow, D. Lakehal, and D. Kothe: *A 3D unsplit-advection volume tracking algorithm with planarity-preserving interface reconstruction.* Computers & Fluids, 35(10):1011–1032, 2006.

[LS69] Ladyzhenskaya, O. A. and R. A. Silverman: *The mathematical theory of viscous incompressible flow.* Gordon and Breach New York, 2nd edition, 1969.

[Mai06] Maitre, E.: *Review of numerical methods for free interfaces.* Technical report, Ecole thématique, 2006.

[Mén07] Ménard, T.: *Développement d'une méthode Level Set pour le suivi d'interface – Application à la rupture de jet liquide.* PhD thesis, Université de Rouen, 2007.

[MI05] Mittal, R. and G. Iaccarino: *Immersed boundary methods.* In *Annual Review of Fluid Mechanics*, volume 37, pages 239–261. Annual Reviews, 2005.

[MK07] Mukherjee, A. and S. G. Kandlikar: *Numerical study of single bubbles with dynamic contact angle during nucleate pool boiling.* International Journal of Heat and Mass Transfer, 50(1):127–138, 2007.

[Mof64] Moffatt, H. K.: *Viscous and resistive eddies near a sharp corner.* Journal of Fluid Mechanics, 18(1):1–18, 1964.

[Mos14] Moskvitch, K.: *Fiendish million-dollar proof eludes mathematicians.* Nature News, 2014. `http://www.nature.com/news/fiendish-million-dollar-proof-eludes-mathematicians-1.15659`.

[MPI14] MPI: *Message Passing Interface, MPI: Version 1.7.5*, 2014. `http://www.open-mpi.de/software/ompi/v1.7/`.

[MTB07] Ménard, T., S. Tanguy, and A. Berlemont: *Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet.* International Journal of Multiphase Flow, 33(5):510–524, 2007.

[MW04] Monnier, J. and P. Witomski: *Analysis of a local hydrodynamic model with Marangoni effect.* Journal of Scientific Computing, 21(3):369–403, 2004.

[NaS] NaSt3DGPF: *A parallel 3D free surface flow solver.* `http://wissrech.iam.uni-bonn.de/research/projects/NaSt3DGP`.

[Nav27] Navier, C. L. M. H.: *Mémoire sur les lois mouvement des fluides.* Mémoire de l'Academie des Sciences, 6:389–440, 1827.

[NW76] Noh, W. F. and P. Woodward: *SLIC (Simple line interface calculation).* In *Lecture Notes in Physics*, volume 59, pages 330–340, 1976.

[Ope13] OpenFOAM: *The Open Source CFD Toolbox, User Guide Version 2.2.2*, 2013. `http://www.openfoam.org`.

[OS88] Osher, S. and J. A. Sethian: *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations.* Journal of Computational Physics, 79(1):12–49, 1988.

[Par11] ParaView: *ParaView User's Guide (v3.10)*, 2011. `http://www.paraview.org/`.

[Pes72] Peskin, C. S.: *Flow patterns around heart valves: A numerical method.* Journal of Computational Physics, 10(2):252–271, 1972.

[PMO$^+$99] Peng, D., B. Merriman, S. Osher, H. Zhao, and M. Kang: *A PDE-based fast local level set method.* Journal of Computational Physics, 155:410–438, 1999.

[PTVF07] Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery: *Numerical recipes 3rd edition: The art of scientific computing.* Cambridge University Press, 2007.

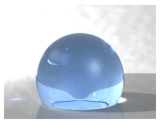
[QWS06] Qian, T., X. P. Wang, and P. Sheng: *Molecular hydrodynamics of the moving contact line in two-phase immiscible flows.* Communications in Computational Physics, 1(1):1–52, 2006.

[RFG$^+$] Reusken, A., O. Fortmeier, J. Grande, S. Groß, M. Larin, and H. Nguyen: *DROPS package, User's guide.* IGPM, RWTH Aachen University.

[RK95] Rider, W. J. and D. B. Kothe: *Stretching and tearing interface tracking methods.* AIAA paper, 95:806–816, 1995.

[RR02] Renardy, Y. and M. Renardy: *PROST: A parabolic reconstruction of surface tension for the volume-of-fluid method.* Journal of Computational Physics, 183(2):400–421, 2002.

[RS00] Russo, G. and P. Smereka: *A remark on computing distance functions.* Journal of Computational Physics, 163(1):51–67, 2000.



[SB00] Somalinga, S. and A. Bose: *Numerical investigation of boundary conditions for moving contact line problems.* Physics of Fluids, 12(3):499–510, 2000.

[Sep96] Seppecher, P.: *Moving contact lines in the Cahn-Hilliard theory.* International Journal of Engineering Science, 34(9):977–992, 1996.

[SF99] Sussman, M. and E. Fatemi: *An efficient, interface preserving level set re-distancing algorithm and its application to interfacial incompressible fluid flow.* SIAM Journal on Scientific Computing, 20(4):1165–1191, 1999.

[SH02] Son, G. and N. Hur: *A coupled level set and volume-of-fluid method for the buoyancy-driven motion of fluid particles.* Numerical Heat Transfer, Part B: Fundamentals, 42(6):523–542, 2002.

[Shi07] Shikhmurzaev, Y. D.: *Capillary flows with forming interfaces.* Chapman & Hall-/CRC, 2007.

[Shi11] Shikhmurzaev, Y. D.: *Some dry facts about dynamic wetting.* The European Physical Journal Special Topics, 197(1):47–60, 2011.

[Son03] Son, G.: *Efficient implementation of a coupled level-let and volume-of-fluid method for three-dimensional incompressible two-phase flows.* Numerical Heat Transfer, Part B: Fundamentals, 43(6):549–565, 2003.

[SP00] Sussman, M. and E. G. Puckett: *A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows.* Journal of Computational Physics, 162(2):301–337, 2000.

[Spe05] Spelt, P. D. M.: *A level-set approach for simulations of flows with multiple moving contact lines with hysteresis.* Journal of Computational Physics, 207(2):389–404, 2005.

[SS12a] Sprittles, J. E. and Y. D. Shikhmurzaev: *Coalescence of liquid drops: Different models versus experiment.* Physics of Fluids, 24(12), 2012.

[SS12b] Sprittles, J. E. and Y. D. Shikhmurzaev: *The dynamics of liquid drops and their interaction with solids of varying wettabilities.* Physics of Fluids, 24(8), 2012.

[SS13] Sprittles, J. E. and Y. D. Shikhmurzaev: *Finite element simulation of dynamic wetting flows as an interface formation process.* Journal of Computational Physics, 233:34–65, 2013.

[SSH$^+$07] Sussman, M., K. M. Smith, M. Y. Hussaini, M. Ohta, and R. Zhi-Wei: *A sharp interface method for incompressible two-phase flows.* Journal of Computational Physics, 221(2):469–505, 2007.

[SSK12] Sibley, D. N., N. Savva, and S. Kalliadasis: *Slip or not slip? A methodical examination of the interface formation model using two-dimensional droplet spreading on a horizontal planar substrate as a prototype system.* Physics of Fluids, 24(8), 2012.

[Sus01] Sussman, M.: *Adaptive Method of Lines*, chapter An adaptive mesh algorithm for free surface flows in general geometries. Chapman & Hall/CRC, 2001.

[Sus03] Sussman, M.: *A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles.* Journal of Computational Physics, 187(1):110–136, 2003.

[SZ99] Scardovelli, R. and S. Zaleski: *Direct numerical simulation of free-surface and interfacial flow.* Annual Review of Fluid Mechanics, 31(1):567–603, 1999.

[Tan79] Tanner, L. H.: *The spreading of silicone oil drops on horizontal surfaces.* Journal of Physics D: Applied Physics, 12(9):1473, 1979.

[TBE$^+$01] Tryggvason, G., B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y. J. Jan: *A front-tracking method for the computations of multiphase flow.* Journal of Computational Physics, 169(2):708–759, 2001.

[Tec13] Tecplot: *Tecplot 360 user's manual*, 2013. `http://www.tecplot.com/`.

[Tho13] Thoraval, M. J.: *Drop impact splashing and air entrapment.* PhD thesis, King Abdullah University of Science and Technology, 2013.

[UT92] Unverdi, S. O. and G. Tryggvason: *A front-tracking method for viscous, incompressible multi-fluids flow.* Journal of Computational Physics, 100:25–37, 1992.

[vM02] Mourik, S. van: *Numerical modelling of the dynamic contact angle.* Master's thesis, University of Groningen, 2002.

[VSDR09] Vadillo, D. C., A. Soucemarianadin, C. Delattre, and D. C. D. Roux: *Dynamic contact angle effects onto the maximum drop impact spreading on solid surfaces.* Physics of Fluids, 21(12):122002, 2009.

[Wad35] Wadell, H.: *Volume, shape and roundness of quartz particles.* The Journal of Geology, 43:250–280, 1935.

[Wik03a] Wikipedia: *Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc.*, 2003. `http://en.wikipedia.org/wiki/Navier%E2%80%93Stokes_existence_and_smoothness`, Oct. 2014.

[Wik03b] Wikipedia: *Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc.*, 2003. `http://en.wikipedia.org/wiki/Level_set_method`, Oct. 2014.

[Wik03c] Wikipedia: *Wikipedia: The Free Encyclopedia. Wikimedia Foundation, Inc.*, 2003. `http://en.wikipedia.org/wiki/Dirac_delta_function`, Oct. 2014.

[WW07] Weiqing, R. and E. Weinan: *Boundary conditions for the moving contact line problem.* Physics of Fluids, 19(2), 2007.

[WYKS09] Wang, Z., J. Yang, B. Koo, and F. Stern: *A coupled level set and volume-of-fluid method for sharp interface simulation of plunging breaking waves.* International Journal of Multiphase Flow, 35(3):227–246, 2009.

[WYS08] Wang, Z., J. Yang, and F. Stern: *Comparison of particle level set and CLSVOF methods for interfacial flows.* In *46th AIAA Aerospace Sciences Meeting and Exhibit*, pages 7–10, 2008.

[WYS09] Wang, Z., J. Yang, and F. Stern: *An improved particle correction procedure for the particle level set method.* Journal of Computational Physics, 228(16):5819–5837, 2009.

[YCZ11] Yun-chao, S., W. Chun-hai, and N. Zhi: *Study on wetting model with combined Level Set-VOF method when drop impact onto a dry surface.* In *Electronic and Mechanical Engineering and Information Technology (EMEIT)*, volume 5, pages 2583–2586, 2011.

[YS07] Yang, J. and F. Stern: *A sharp interface method for two-phase flows interacting with moving bodies.* In *Proc. 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, AIAA-2007-4578*, 2007.

[YVHH09] Yokoi, K., D. Vadillo, J. Hinch, and I. Hutchings: *Numerical studies of the influence of the dynamic contact angle on a droplet impacting on a dry surface.* Physics of Fluids, 21(7):072102, 2009.

[Zas09] Zaspel, P.: *Zweiphasige Navier-Stokes Fluidsimulationen in Maya: Konfiguration, Visualisierung und Animation.* Diplomarbeit, Institut für Numerische Simulation, Universität Bonn, 2009.

[ZG11] Zaspel, P. and M. Griebel: *Photorealistic visualization and fluid animation: Coupling of Maya with a two-phase Navier-Stokes fluid solver.* Computing and Visualization in Science, 14(8):371–383, 2011.

[ZGK09] Zahedi, S., K. Gustavsson, and G. Kreiss: *A conservative level set method for contact line dynamics.* Journal of Computational Physics, 228:6361–6375, 2009.