



TECHNISCHE
UNIVERSITÄT
DARMSTADT

**SIMULATION BASED OPTIMIZATION
OF MULTIPHASE FLOW IN THE
CONTEXT OF WETTING
PHENOMENA**

Vom Fachbereich Mathematik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte

Dissertation

von

Elisabeth Andrea Gertrud Diehl, M.Sc.

aus Heppenheim

Referent:	Prof. Dr. Stefan Ulbrich
Korreferent:	Prof. Dr. Dieter Bothe
Tag der Einreichung:	06.09.2023
Tag der mündlichen Prüfung:	20.10.2023

Darmstadt 2023
D 17

Simulation based Optimization of Multiphase Flow in the Context of Wetting Phenomena

Accepted doctoral thesis by Elisabeth Andrea Gertrud Diehl, M.Sc.

Darmstadt, Technische Universität Darmstadt

Date of thesis defense: October 20, 2023

Year of publication on TUprints: 2024

Please cite this document as / Bitte zitieren Sie dieses Dokument als:

URN: urn:nbn:de:tuda-tuprints-263478

URL: <https://tuprints.ulb.tu-darmstadt.de/id/eprint/26347>

This document is provided by / Dieses Dokument wird bereitgestellt von:

TUprints, E-Publishing-Service der TU Darmstadt

<http://tuprints.ulb.tu-darmstadt.de>

tuprints@ulb.tu-darmstadt.de

This work is licensed under a Creative Commons License:

CC BY-NC-SA 4.0

Attribution – NonCommercial – ShareAlike 4.0 International

<https://creativecommons.org/licenses/by-nc-sa/4.0/deed.en>

Die Veröffentlichung steht unter folgender Creative Commons Lizenz:

CC BY-NC-SA 4.0

Namensnennung – Nicht-kommerziell – Weitergabe unter gleichen Bedingungen 4.0

International

<https://creativecommons.org/licenses/by-nc-sa/4.0/deed.de>

Acknowledgments

First of all, I would like to thank my supervisor Prof. Stefan Ulbrich for the opportunity to do a doctorate, for the research topic he chose perfectly for me, the possibility to visit exciting conference locations and for being part of his wonderful and inspiring working group. I am grateful for all the scientific input, the joint programming sessions and always having an open door. But I would like to thank him just as much for his personal support during the last seven years, for his patience while the Corona Pandemic and long lasting kindergarten closures, for his encouraging words and for being always understanding.

Moreover, I thank Prof. Dieter Bothe for acting as referee of this thesis and for his constant interest in my work. Furthermore, I thank Prof. Marc Pfetsch and Prof. Tabea Tscherpel for taking the time to act as members of the examination committee.

I kindly acknowledge the financial support by the German Research Foundation (DFG) within the Collaborative Research Centre (CRC) 1194 „Interaction of Transport and Wetting Processes“, where I was part of the subproject B04 throughout my time as doctoral candidate. I greatly benefited from being part of a great team of researchers within the CRC 1194. In particular, I would like to thank the speaker Prof. Peter Stephan and the managing director Benjamin Lambie for their dedicated and trustworthy leadership, especially with regard to efforts improving the compatibility of family and career. Furthermore, I thank Mathis Fricke, Dirk Gründing, Pauline Brumm, Hans-Martin Sauer and Holger Marschall for very interesting collaborations and insights within the CRC 1194. I always enjoyed our scientific discussions and drinking a beer after that.

Many thanks to my colleagues Isabel Jacob, Lea Rehlich, Marcel Steinhardt and Erik Jansen for proofreading various sections of the thesis and all the scientific input I got in discussions, talks or while drinking coffee and tea together. Moreover, I thank Erik Jansen for being my former research assistant and I am very happy

that he is a colleague now. I also thank my longtime office mate Alexander Matei. There was always a very friendly and respectful atmosphere in our office. A special thank goes to Björn Polenz, with whom I went through many ups and downs, with whom I laughed and cried and who motivated me again and again, especially during the intensive time of writing down the thesis. And I gratefully thank my former colleagues and long standing friends Kristina Janzen, Anna Walter, Benjamin Horn and Paloma Aguilar Schäfer for the wonderful time we had in the working group Optimization and all the funny and crazy activities we did together. Anna and Kristina also had the pleasure to proofread parts of this thesis, many thanks for this.

I especially thank my family for their support, love and encouragement during all times of my doctoral studies. My parents Susanne and Andreas, who have always kept my back free, through short home vacations, through food support, through childcare or simply through an open ear and encouraging words. My big but smaller sister Rita, who has always been a great role model for me in terms of toughness and perseverance. My little but larger sister Judith, who I admire for her incredible clarity and wisdom and from whom I have always received honest and appreciative feedback. And my little brother Sebastian, who always gave me a place of retreat where I could be completely free and unconstrained. As well, I want to thank Till Müller sincerely, who accompanied and supported me for so many years and is still one of the most important people in my life.

Last but not least, I am deeply grateful to my little daughter Ronja, who was always brave and gave me love and power during all the good and the hard times. Thank you! I love you!

Abstract

The wetting and dewetting of surfaces by fluids is pervasive in our nature, but also plays a crucial role in many technical processes and applications. Examples include coating and printing, microfluidics and lab-on-a-chip technologies, as well as cooling in certain reactor geometries or other process industry installations. In general, dynamic wetting processes can be represented as multiphase flows, which can be described mathematically with the aid of the Navier-Stokes equations. In addition, jump conditions are needed to connect the flow of the different fluids or phases across their interfaces, on which a surface tension is applied. If a dynamic contact line occurs besides the liquid-gas interfaces, where liquid and gas touch the solid surface, a slightly different modeling approach is followed. For this purpose, the Navier-Stokes equations are additionally complemented by a transport equation for flow advection. This transport equation originates from an algebraic volume of fluid approach, that leads to an one-field formulation of the problem. The model is completed by appropriate initial and boundary conditions, where the dynamic contact angle enters as a boundary condition. The resulting model is a system of partial differential equations, which is used in the simulation based optimization of wetting processes. The considered optimization problems belong to the class of optimal control problems, in which an objective function is optimized with respect to a state and a control. For existence and uniqueness statements, the differentiability of the related control-to-state mapping is required, where L^p -maximal regularity of the underlying linear two-phase problem is acquired. Proving the differentiability is a central part of this work, where the control consists of an initial velocity field and an over the whole domain distributed component on the right side of the momentum equation. This creates a basis to solve optimal control problems of wetting processes with gradient-based optimization methods. However, the common approach of solving the constrained optimization problem by means of the Lagrangian function is difficult in the context of partial differential equations and unsuitable for our wetting model. Hence, we follow a sensitivity approach and formally derive sensitivity equations for the continuous flow problem. State equations and sensitivity equations are now solved numerically with suitable discretization methods, since an analytical solution for this type of problem is not yet known. Therefore,

the two-phase solver `interFoam`, which originates from the OpenFOAM library, is adapted to the effect that the state equations and the respective sensitivity equations are solved simultaneously. The developed method is tested on a benchmark test case, which is motivated by gravure printing. For good printing results, it is essential to remove excess ink from the printing plate, except for a thin film that remains. For this purpose, a steel strap is pulled over the surface, which is also called a doctor blade. Numerical results are presented exemplary for a parameter identification problem and the optimization with respect to geometric aspects for the above-mentioned wetting process.

Zusammenfassung

Das Be- und Entnetzen von Oberflächen durch Fluide ist in unserer Natur allgegenwärtig, spielt aber auch in vielen technischen Prozessen und Anwendungen eine entscheidende Rolle. Beispiele dafür sind das Beschichten und Drucken, Mikrofluidik und Lab-on-a-Chip-Technologien oder auch das Kühlen in bestimmten Reaktorgeometrien oder anderen Anlagen der Prozessindustrie. Im Allgemeinen können dynamische Benetzungsvorgänge als Mehrphasenströmungen dargestellt werden, die sich mathematisch mithilfe der Navier-Stokes-Gleichungen beschreiben lassen. Zusätzlich werden Sprungbedingungen benötigt, die die Strömung der verschiedenen Fluide oder Phasen über ihre Grenzflächen hinweg verbinden, an welchen zusätzlich eine Oberflächenspannung anliegt. Kommt neben den Flüssigkeits-Gas-Grenzflächen eine dynamische Kontaktlinie vor, an der Flüssigkeit und Gas die Festkörperoberfläche berühren, wird ein etwas anderer Modellierungsansatz verfolgt. Hierfür werden die Navier-Stokes-Gleichungen um eine zusätzliche Transportgleichung ergänzt, welche die Advektion der Strömung beschreibt. Die Transportgleichung wird für einen algebraischen Volume-of-Fluid-Ansatz benötigt, der zu einer Ein-Feld-Formulierung des Problems führt. Das Modell wird durch geeignete Anfangs- und Randbedingungen vervollständigt, wobei der dynamische Kontaktwinkel als Randbedingung eingeht. Es entsteht ein System partieller Differentialgleichungen, welches in die simulationsbasierte Optimierung von Benetzungsvorgängen einfließt. Die betrachteten Optimierungsprobleme gehören zu der Klasse der Optimalsteuerungsprobleme, in denen eine Zielfunktion bezüglich eines Zustandes und einer Steuerung bzw. Kontrolle optimiert wird. Für Existenz- und Eindeutigkeitsaussagen wird die Differenzierbarkeit der zugehörigen Steuerungs-Zustands-Abbildung benötigt, wobei L^p -maximale Regularität des zugrundeliegenden linearen Zweiphasenproblems erarbeitet wird. Die Differenzierbarkeit zu zeigen ist ein zentraler Bestandteil dieser Arbeit, wobei sich bei den theoretischen Betrachtungen die Kontrolle aus einem initialen Geschwindigkeitsfeld und einer über dem gesamten Gebiet verteilten Kontrolle auf der rechten Seite der Impulsgleichung zusammensetzt. Damit wird eine Basis geschaffen, um Optimalsteuerungsprobleme von Benetzungsprozessen mit Methoden der ableitungsbasierten Optimierung zu lösen. Jedoch gestaltet sich die Lösung eines restringierten Optimierungspro-

blems mit partiellen Differentialgleichungen als Nebenbedingungen im Kontext von Benetzungsproblemen als schwierig und kann nicht wie üblich mithilfe der Lagrange-Funktion gelöst werden. Daher verfolgen wir einen Sensitivitätsansatz und leiten formal Sensitivitätsgleichungen für das kontinuierliche Strömungsproblem her. Zustandsgleichungen und Sensitivitätsgleichungen werden nun numerisch mit geeigneten Diskretisierungsverfahren gelöst, da eine analytische Lösung für solche Probleme bislang nicht bekannt ist. Dafür wird der aus der OpenFOAM-Bibliothek stammende zwei-Phasen Löser `interFoam` dahingehend angepasst, dass simultan zu den Zustandsgleichungen auch die zugehörigen Sensitivitätsgleichungen gelöst werden. Das entwickelte Verfahren wird an einem Demonstratorbeispiel getestet, welches durch den Tiefdruck motiviert ist. Für ein gutes Druckergebnis ist es essenziell, überschüssige Farbe bis auf einen dünnen Film von der Druckplatte zu entfernen. Zu diesem Zweck wird ein Stahlband über die Oberfläche gezogen, welches auch als Rakel bekannt ist. Numerische Ergebnisse werden beispielhaft für ein Parameteridentifikationsproblem und die Optimierung hinsichtlich geometrischer Aspekte für den genannten Benetzungsprozess dargestellt.

Contents

1	Introduction	1
1.1	What is Wetting?	2
1.2	Related Work	4
1.3	Outline of the thesis	7
2	Description of Multiphase Flow	9
2.1	Basic Notations and Definitions	10
2.1.1	Function Spaces for Fluid Flow	10
2.1.2	Differential Operators and Theorems	15
2.2	Conservation Equations	17
2.2.1	Conservation of Mass	17
2.2.2	Conservation of Momentum	19
2.2.3	Navier-Stokes Equations	20
2.3	Standard Description of Two-phase Flow	21
2.4	Related One-Field Formulation	25
2.4.1	Volume of Fluid Approach	25
2.4.2	Modeling the Surface Tension	27
2.5	Boundary Conditions	29
2.6	Dynamic Contact Line Treatment	32
3	Optimal Control of Two-phase Flow	37
3.1	Optimal Control Problem	39
3.2	Sensitivity Approach	40
3.3	Differentiability of the Control-to-State Mapping	42
3.3.1	Weak Formulation	44
3.3.2	L^p -maximal Regularity	45
3.3.3	Transformation to a Flat Interface	46
3.3.4	Well-posedness and Differentiability of the Transformed State	50
3.3.5	Differentiability of the Original Problem	58
3.3.6	Results for the Volume of Fluid-type Formulation	61
3.4	Sensitivity System with respect to Liquid Viscosity	75

3.5	Sensitivity-based Optimization Algorithm	79
4	Numerical Solution	81
4.1	Discretization	82
4.1.1	Finite Volume Discretization	82
4.1.2	Temporal Discretization	85
4.1.3	Discretization of Equations	86
4.2	Numerical Solution Procedures	92
4.2.1	Numerical Solution of the α -Transport Equation	93
4.2.2	Numerical Solution of the Navier-Stokes Equations	99
4.3	Discretization and Numerical Solution of the Sensitivity Equations .	104
4.4	Software	107
4.4.1	Simulation Characteristics with OpenFOAM	108
4.4.2	Implementation of Sensitivity Solver interSensFoam	112
5	Optimization of Doctor Blading	117
5.1	Physics of the System	119
5.2	The Doctor Blading Test Case	121
5.2.1	Validation with Experimental Data	122
5.2.2	Optimization Objectives	128
5.3	Optimization Framework	130
5.4	Numerical Optimization Results	133
5.4.1	Optimization of the Liquid Viscosity	134
5.4.2	Optimization of the Gap Height	137
5.4.3	Optimization of the Vorticity	140
6	Conclusion	143
A	Derivation of Equations	147
B	Developer Documentation interSensFoam	149
B.1	Variables and Constants	149
B.2	Walkthrough the OpenFOAM code	153
B.3	Compile Options and User Specifications	166
	List of Symbols	169
	Bibliography	173

List of Figures

2.1	Stationary control volume V with surface S [93].	18
2.2	Interface $\Gamma(t)$ separates open sets $\Omega_1(t)$ and $\Omega_2(t)$	22
2.3	A two-phase domain with outer boundary $\partial\Omega$ [69].	29
2.4	Comparison of no-slip and Navier-slip.	31
2.5	Equilibrium contact angle according to Young's law.	33
2.6	A two-phase domain with interface and contact points.	35
3.1	Schematic diagram of a gradient-based optimization algorithm [36].	38
3.2	Interface parameterized with a function $h(t, x)$, inspired by [16].	46
4.1	Two control volumes with common boundary face f [62].	83
4.2	Exemplary discretized phase fraction field α	88
5.1	Full gravure printing process with zoom to the doctor blade.	118
5.2	Example for static contact angle of a water droplet.	120
5.3	Geometry of doctor blading test case and velocity boundary conditions.	121
5.4	Szenario 1.	122
5.5	Szenario 2.	122
5.6	Different views of the 3D droplet simulation.	123
5.7	Block structure of the domain.	123
5.8	3D mesh of the whole domain.	123
5.9	Droplet real experiment.	126
5.10	Droplet 3D simulation.	126
5.11	Bladed droplet with a transparent doctor blade.	126
5.12	Bladed droplet in simulation.	126
5.13	Snapshots of the droplet simulated in 3D.	127
5.14	Coupling of Matlab and OpenFOAM simulations.	132
5.15	Initial state in Ω_{bd}	135
5.16	Optimized state in Ω_{bd}	135
5.17	Field α_{water}	136
5.18	Field $d\text{Alpha}$	136

5.19	Field U.	136
5.20	Field dU.	136
5.21	Comparison of velocity sensitivity components with the difference quotient.	137
5.22	Position of QOI (red bar).	138
5.23	Triangulation of the domain in 2D	139
5.24	Initial state gap height.	140
5.25	Optimized state gap height.	140
5.26	Initial state vorticity.	142
5.27	Optimized state vorticity.	142
B.1	Structure of interSensFoam.	154
B.2	File interSensFoam.C, part1.	155
B.3	File interSensFoam.C, part2.	156
B.4	File interSensFoam.C, part3.	157
B.5	Structure of a test case for interSensFoam.	167

List of Tables

4.1	Primal system and its derivatives in interSensFoam.	113
5.1	Material properties of air, water and silicone oil [11].	120
5.2	Numerical parameter and solver setup for the 3D simulation.	124
5.3	Discretization schemes for the individual terms.	124
5.4	Setup of OpenFOAM boundary conditions for the doctor blading test case.	125
5.5	Setup of boundary conditions for the sensitivity solver.	134
5.6	Result of the Gauss-Newton algorithm.	135
5.7	Result of the Gauss-Newton algorithm.	140
5.8	Numerical results with the Gauss-Newton method.	141
B.5	OpenFOAM implementation for $UEqn.H$ and $dUEqn.H$	164
B.6	OpenFOAM implementation for $pEqn.H$ and $dPEqn.H$	165

Introduction

In this thesis, we investigate the modeling and simulation based optimization of multiphase flow in the context of wetting phenomena. Wetting is something, that surrounds us every day. In nature, but also in many technical applications. One of the most famous examples from nature is the lotus flower, where the so-called lotus effect can be observed. A droplet of water almost completely rolls off a lotus leaf and can form a contact angle of about 170 degrees with the leaf. This impressively shows how the wetting behavior of a droplet can be changed by the structure of the surface. Such examples from nature serve as a pattern for state-of-the-art technologies, such as superhydrophobic, dirt-repellant or icephobic surfaces. Another example, where we can observe different wetting behaviors is a simple water droplet running down a window. Everyone has already observed these small droplets, which find their way down, merge with other droplets to form larger ones, pinning again and again, or are distracted in a certain direction for example by a moving car. Various wetting phenomena underlie this seemingly trivial process and a deep knowledge of the underlying physics is indispensable to use the natural role models for technical applications. Some of these fundamental questions are addressed by the collaborative research center (CRC) 1194 „Interaction of Transport and Wetting Processes“. As the name already suggests, the CRC investigates wetting processes, where heat or mass transport occurs parallel to momentum transport and where complex fluids or structured surfaces are used. The joint work of mathematicians, physicists, engineers and experimentalists gives the possibility to synergize different disciplines and create a reliable and valid model of complex wetting processes. As far as we know, such complex wetting processes are not yet understood in detail and the mathematical description is incomplete, especially for the flow behavior at the three-phase contact line.

To investigate such complex processes, extensive experiments are necessary involving large parameter studies. These are often very costly and impractical, why they are replaced by numerical simulations. Therewith, a large number of virtual experiments can be performed with a minor effort of time and material. However, these only make sense if we have a correspondingly expressive and realistic model, which we can use to verify experimental results and optimize material or operation parameters in a second step. At best, this leads to improved experimental results. In order to achieve such improvements, the number of simulations can quickly become very large if many parameter settings need to be checked or the number of different parameters is large. Without sophisticated optimization algorithms, this can hardly be calculated in a reasonable time, even with today's computing capacities. The analysis and investigations regarding simulation and targeted influencing of wetting processes, done with methods from optimal control theory, are addressed in this thesis to support decision-making processes and improve technical applications.

1.1 What is Wetting?

Wetting is the formation of an interface between a liquid and a solid surface and is often modeled as a multiphase flow problem. Here, the term multiphase relates to the different aggregation states of a fluid, so there can occur gaseous, liquid or solid phases. In our case, we define a wetting process as the interaction between a two-phase flow and a solid, not deformable surface. The two-phase flow typically consists of a liquid and a gas phase, or two different fluids. At least one of these phases is in contact with the solid surface. A distinction is made between static and dynamic wetting. While in the static case an equilibrium between the involved forces is reached and a static interface and contact angle is formed, the dynamic wetting describes the motion of a fluid onto a substrate. There, a dynamic contact line is shifted with the passage of time and different contact angles arise. This leads to an advancing contact angle during wetting and a receding one during dewetting. So we can summarize, that wetting dynamics deal with the time evolution of moving contact lines on solid surfaces, which describes for instance the spreading of fluids onto a substrate [33].

The mathematical formulation of wetting phenomena is dependent on the considered length scale. We roughly distinguish between the macro-, the micro- and the nanoscale, whereby intermediate states are also described in literature. The classical continuum model approach allows the description of a wide range

of macroscopic phenomena, while molecular models mainly cover the microscopic and nanoscopic scale. Although we can also achieve a very high resolution with the use of continuum models, they are unable to reproduce these microscopic and nanoscopic phenomena. These have to be included as physical constants or boundary conditions to the model. There also exist hybrid approaches, so-called Hybrid-Atomistic-Continuum approaches, which try to cover a wider range of length scales. In this work, we will focus on macroscopic formulations. Then, the mathematical model of a wetting process, or more precisely of multiphase flow, is governed by the Navier-Stokes equations together with jump conditions to connect the flow of the different fluids or phases at their interfaces. In its description, the classification of the investigated fluids is crucial. There exist different equations for incompressible or compressible fluids, for viscous or non viscous fluids, for Newtonian or non-Newtonian fluids and of course, also for a creeping, laminar or turbulent flow behavior.

With a valid model, the scope of wetting applications is very widespread. For example, wetting processes play an important role in functional printing, for Lab-on-a-Chip technologies and for the optimization of airplane wings or car bodies. As this thesis is part of the research topics within the CRC 1194, we focus on a wetting process that is essential for functional printing. Printing electrical devices requires an extraordinary precision on a relatively small length scale since conductive elements must not touch each other. Gravure printing is most suitable for such high-precision printing tasks and is for example also used to print banknotes. Here, the doctor blading is a very important sub-process. A doctor blade is a sharp steel band that is scratched over the engraved printing form to remove the excessive ink. In this way, the ink remains almost exclusively inside the engraved cups and only reaches those areas on the print medium that are to be printed on. This is a crucial point when printing the smallest electronic connections, for example, because otherwise undesirable power bridges would occur and would disturb the printed device. A specific wetting behavior is thereby pursued and leads to the following questions. How should material parameters be chosen to reach a specific wetting behavior, for example to reach a prescribed spreading? How can we influence the film formation in our printing process with changing geometrical quantities like the inclination angle of the doctor blade or the shape of the doctor blade itself? Generally, how can we calculate an optimal solution for this type of wetting phenomena mathematically? Among others, these are the questions that will be answered within this work.

1.2 Related Work

Multiphase problems are the subject of active research as they are very important for many technical processes. While the efforts for single fluids and free boundary problems are already advanced, the multiphase case is based on these results and was therefore only able to develop properly in the last few decades. Let us get a little overview of how the topic has developed. The Navier-Stokes equations describing fluid flow have been studied by many mathematicians. Mathematical questions are the existence, uniqueness and stability of solutions, so it is of interest if a problem is well posed. Then, there are also questions concerning the regularity of solutions as well as appropriate solution concepts and the qualitative properties of solutions, e.g., stability, long time behavior or equilibrium conditions. An early comprehensive work was provided by TEMAM in [91], who proved basic ideas for existence and uniqueness results of the single fluid Navier-Stokes equations. Since we can not mention all of the numerous works concerning the Navier-Stokes equations, we will focus on results where surface tension occurs. In the case of a bounded domain, existence results local in time for the corresponding free boundary problem were for example derived by SOLONNIKOV, who mentioned existence of the problem in the L^2 Sobolev-Slobodetskii space in a series of papers, see e.g., [87] and further paper cited therein, by SCHWEIZER for a semigroup setting [80] and by MOGILEVSKII together with SOLONNIKOV, who covered the case of Hölder spaces [60, 87]. More recent results were provided by SHIBATA and SHIMIZU in [83, 86], who showed local existence and uniqueness in the case of a perturbed infinite layer or halfspace as initial domain and for a setting with anisotropic Sobolev spaces $W_{p,q}^{2,1}$, where $2 < p < \infty$ and $n < q < \infty$. Global existence results were for example provided by BEALE in [9], if the initial state and the initial velocity are close to equilibrium and gravity and surface tension effects are included. Here a layer of viscous, incompressible fluid in an ocean of infinite extend was assumed, bounded by a lower solid and an upper free surface. These works paved the way towards more evolved multiphase flow problems. Based on Lagrangian coordinates, DENISOVA established existence and uniqueness results of local strong solutions for a two-phase case where one of the domains is bounded in [22] and together with SOLONNIKOV for a transformed problem with an implicit representation of the free boundary, see [23]. Further relevant results were for example developed by ABELS in [1] and TANAKA in [90]. Furthermore, PRÜSS and SIMONETT showed local well-posedness of the underlying linear problem by means of L_p -maximal regularity [70], which forms the basis of our differentiability investigations of the control-to-state operator. Moreover, they prove that the interface as well as the solution becomes instantaneously real analytic.

Besides the theoretical aspects, the numerical considerations of the introduced problem formulation and their implementations are considered in the present work, which is assigned to the research field Computational Fluid Dynamics (CFD). This is unavoidable, since the Navier-Stokes equations can be solved analytically just in very specific cases. Also in our case, we have to solve them numerically. Here, the consistency between the continuous and the discretized model equations as well as respective solution procedures are of special interest. The origin of CFD can be traced back to the late 1950s and early 1960s years [93], with the aim to move away from experimental studies and empirical correlations to more general applicable and accurate mathematical models of engineering systems. One of the first numerical methods, which was able to treat the full Navier-Stokes equations, was the marker-and-cell (MAC) method, introduced 1965 by HARLOW and WELSCH in [38]. Here, rudimental multiphase flows, at first only with a free boundary, could be solved. From this, the volume of fluid (VOF) method was developed at the beginning of the 1980s and first described by HIRT and NICHOLS, see [42], where the originally used tracking of particles was replaced by a marker function advected with a transport equation. To overcome the lack of interface diffusion over numerous cells, which resulted from the cell-averaged marker function [93], the VOF method was adapted with different interface reconstructions, e.g., by ASHGRIZ and POO in [6] or later by SCARDOVELLI and ZALESKI in [78], to name only a few of them. Besides the MAC and VOF method, other numerical methods were developed over the course of time, such as front-tracking by UNVERDI and TRYGGVASON [97] as well as the level-set method by OSHER and SETHIAN [63] or the phase-field approach by KOBAYASHI [54], but we will not go into details here.

With the great progress in CFD, also the optimization of flow problems came into focus, especially optimal control of fluid flow. The difficulties of multiphase flow problems are, that we have to deal with highly nonlinear partial differential equations (PDEs), small length and short time scales and that the phase boundaries are part of the solution. PDE constrained optimization embodies analysis, discretization, and the development of dedicated optimization methods for minimization problems constrained by partial differential equations [96]. For the optimization theory, especially the differentiability of the functionals and the constraints is required to fulfill necessary and sufficient optimality conditions. The derivative based optimization is more efficient than derivative-free methods and appropriate for our problem formulation. There are many differentiability results but primarily for the single phase case, for the multiphase case without surface tension or for representations with a diffuse interface model. Various optimal control problems for the single phase time-dependent Navier-Stokes equations were

for example treated in [2, 31, 40, 95] and numerous other contributions. An optimal control problem of a binary fluid described by its density distribution without surface tension is investigated in [8]. Based on the thermodynamically consistent diffuse interface model, GARCKE, HINZE and KAHLE derived necessary optimality conditions for the time-discrete and the fully discrete optimal control problem with respect to distributed and boundary controls [32]. Compared to the sharp interface approach, the interface is assigned a transition region with finite thickness, which leads to a thin interfacial layer. However, research about differentiability properties of the two-phase Navier-Stokes equations are quite rare. The main challenges in this case are the moving interface and the surface tension. To the best of our knowledge, this is the first work providing differentiability properties of control-to-state mappings for sharp interface models of two-phase Navier-Stokes flow with surface tension, previously published in [25]. In our case, the solid-gas or solid-liquid interface maintains its shape, because the solid is not flexible. Moreover, there is also a very interesting field of research which deals with flexible solids where the location of the solid can change in response to fluid flow, see [10, 39]. This sub-discipline of multiphase flow includes Fluid-Solid interactions, but it is beyond the scope of this thesis. It is still worth mentioning, because a technique similar to ours was recently used in [39] to show differentiability properties for shape optimization of fluid-structure interaction, but with the difference of using a different fix point argument.

Different numerical approaches for the optimal control of two-phase flows are discussed in [17] by BRAACK et al., especially a level-set technique and an Allen-Cahn phase-field model. The challenge herein is the treatment of the interface, whereby care must be taken to ensure a sufficiently sharp interface between the two phases with appropriate consideration of the surface tension forces. Also for this challenge, the present work aims to give a consistent numerical formulation and to provide an optimization framework for wetting phenomena. But there are still a lot of open questions regarding wetting dynamics. An example is the moving contact line paradox, which was recently investigated by FRICKE, also a member of the CRC 1194. He observed, that the classical no-slip condition at solid boundaries seems to be incompatible with dynamic wetting phenomena [30]. Despite the intensive work done in this field of research in recent years and the knowledge already gained, there is still much to be done. With our work we would like to contribute to a better understanding of multiphase problems and especially make a profitable contribution in the field of derivative calculations of the incompressible Navier-Stokes equations with surface tension. Furthermore, our results provide new insights to the optimization of problems involving moving contact lines.

1.3 Outline of the thesis

We have seen that optimization of wetting phenomena requires the investigation of multiphase flow which is described by the Navier-Stokes equations. Chapter 2 contains a detailed derivation and description of such multiphase flow problems with different approaches. First, a standard description is given where the Navier-Stokes Equations are supplemented by jump conditions to connect the flow of the different fluids or phases at their interface. Afterwards, we present the volume of fluid approach, an equivalent representation without resorting to jump conditions. Here, the different fluids or phases are treated as one single fluid with an inhomogeneous density and viscosity distribution in the domain. The advection of the interface is described with an additional transport equation. This leads to one single set of equations that represent the flow. Additionally, we will have a closer look to the treatment of surface tension, appropriate boundary conditions as well as the contact line between the different fluids and the solid surface. In Chapter 3, the optimal control problem is introduced. Since we pursue a derivative based optimization approach, the sensitivity equations of the problem have to be determined and a rigorous investigation of differentiability results for the control-to-state mapping is necessary. Therewith, we have all ingredients to formulate a sensitivity-based optimization algorithm for wetting problems. Analytical solutions of the Navier-Stokes equations are available just in very specific cases, for which reason they are usually solved numerically. The numerical solution of the introduced problem is subject of Chapter 4, where we present the spatial discretization of the domain with the Finite Volume method, a temporal discretization and some numerical solution procedures to handle the Navier-Stokes equations as well as the additional transport equation from the VOF model. Furthermore, the simulation with OpenFOAM is mentioned and how governing equations, initial and boundary conditions are treated. In Chapter 5, results from the optimization theory and numerical considerations are combined to optimize the doctor blading process, which exemplifies a wetting process. Different optimization problems are applied to a test case, designed for the investigation of various wetting phenomena. Besides solving parameter identification problems, the shape of the doctor blade and other domain variations are optimized to obtain good printing results. After presenting some numerical results, we conclude with a summary of our investigations and an outlook on future work in Chapter 6. Moreover, the developers documentation of the implemented OpenFOAM code should be mentioned, which is discussed in Appendix B. Here, a walk through the OpenFOAM code as well as compile options and user specifications shall help to reproduce numerical results and apply the developed solver to own optimization problems.

Description of Multiphase Flow

In this chapter we derive the governing equations for fluid flow. Therefore, some basic notations and definitions of function spaces are required, which are provided in the first section and will be frequently used throughout the thesis. Afterwards, we describe the representation of a two-phase domain, before we come to the fundamental conservation equations of fluid flow. Starting from a standard description, we will successively build up a comprehensive model for multiphase flow with a sharp interface separating immiscible fluids or phases respectively. Established results of existence and uniqueness are presented for the classical description with jump conditions and possible boundary conditions are described. If the fluid interface touches the outer boundary, further contact line dynamics come into play, which are discussed in detail subsequently. Then, an equivalent formulation is presented with the Volume of Fluid approach, where additional equations and boundary conditions enter the model and the treatment of surface tension is modified. The formulations are based on the continuum hypothesis, the hypothesis of sharp interfaces and neglecting intermolecular forces. The continuum hypothesis declares that the density of a fluid volume can be well approximated with a smooth function ρ , if the dimensions of the volume are above a few tens of nanometers [93]. As the dimensions range from micro- till millimeters in this work, the hypothesis is an acceptable assumption. The sharp interface hypothesis indicates that a sharp interface separates the different phases, for example a liquid and a gas, or two different liquids. This means that the thickness of the interface is vanished and that the fluid properties generally change across the interface. In addition, the neglect of intermolecular forces includes ignoring forces like electrostatic attractions between atoms or molecules. But note that van der Waals forces are implicitly treated by their most important effect, the capillarity, which enters as surface tension in our model [93].

2.1 Basic Notations and Definitions

Although numerical techniques are used to calculate an explicit solution of the flow system for exclusive test cases, we nevertheless need a suitable solution theory for the governing equations of fluid flow. We will need the basic theory for the derivation of differentiability results in Chapter 3.3, which in turn are fundamental for the theory of derivative-based optimization of the considered flow problems. In general, we can not expect that PDEs have solutions in a strong sense, what also applies to the Navier-Stokes equations. Therefore the concept of weak solutions is used, which can be seen as an extension of a classical solution. While we use C -spaces of continuous functions in the classical solution theory for ordinary partial differential equations, we further need Lebesgue, Sobolev and Bochner spaces to deal with time dependent PDEs. In the following, we introduce the conventional definitions for these infinite dimensional function spaces, see e.g. [102, 92], and recapitulate important and useful theorems.

2.1.1 Function Spaces for Fluid Flow

Let $\Omega \subset \mathbb{R}^n$ be an open domain with $n \in \mathbb{N}$. We denote the set of continuous functions on Ω by $C(\Omega)$. For $k \in \mathbb{N}$, the set $C^k(\Omega)$ contains additionally continuous derivatives up to order k . Furthermore, we write $C^\infty(\Omega) := \bigcap_{k \in \mathbb{N}} C^k(\Omega)$ for the set of infinitely many times differentiable functions. Another important representative of these spaces is the so-called set of test functions $C_0^\infty(\Omega)$, the set of infinitely many times differentiable functions with compact support

$$C_0^\infty(\Omega) := \{\varphi \in C^\infty(\Omega) : \text{supp}(\varphi) \text{ is compact in } \Omega\},$$

where the support of a function φ is defined as $\text{supp}(\varphi) := \{x \in \Omega : \varphi(x) \neq 0\} \subset \Omega$. Moreover, we define the set of β -Hölder continuous functions with $\beta \in (0, 1]$ as

$$C^{k,\beta}(\Omega) := \{\varphi \in C^k(\Omega) : \exists C > 0 \text{ s.t. } |D^\alpha \varphi(x) - D^\alpha \varphi(y)| \leq C|x - y|^\beta \\ \forall x, y \in \Omega \text{ and } |\alpha| = k\}.$$

The function space $C^{0,1}(\Omega)$ is called the set of Lipschitz continuous functions on Ω and hence, a $C^{0,1}$ -boundary is called Lipschitz boundary. For the purpose of clarity, we write $BC(\Omega)$ for the space of bounded continuous functions on Ω and $BUC(\Omega)$ for the space of bounded uniformly continuous functions on Ω , both equipped with

the supremum norm

$$\begin{aligned} BC(\Omega) &:= \{\varphi \in C(\Omega) : \varphi \text{ is bounded and continuous on } \Omega\}, \\ BUC(\Omega) &:= \{\varphi \in C(\Omega) : \varphi \text{ is bounded and uniformly continuous on } \Omega\}. \end{aligned}$$

Analogously to the C spaces, $BC^k(\Omega)$ and $BUC^k(\Omega)$ are defined as the space of k -times continuously differentiable functions with bounded continuous or bounded and uniformly continuous derivatives up to order k . Note that boundedness and uniform continuity are automatically satisfied if Ω becomes compact, so that in this case $BC^k(\Omega)$ and $BUC^k(\Omega)$ conform to $C^k(\Omega)$.

The Lebesgue space of measurable functions on Ω , whose members are Lebesgue-integrable to the power p , is denoted by $L^p(\Omega)$ with $1 \leq p \leq \infty$

$$L^p(\Omega) := \left\{ f : \Omega \rightarrow \mathbb{R}^n : \int_{\Omega} |f(x)|^p dx < \infty \right\}.$$

$(L^p(\Omega), \|\cdot\|_{L^p(\Omega)})$ defines a Banach space, if it is equipped with the following norm

$$\|f\|_{L^p(\Omega)} := \left(\int_{\Omega} |f(x)|^p dx \right)^{1/p}.$$

Let further $L^\infty(\Omega)$ denote the space of measurable functions on Ω , which are essentially bounded, i.e., we define

$$L^\infty(\Omega) := \left\{ f : \Omega \rightarrow \mathbb{R}^n : \operatorname{ess\,sup}_{x \in \Omega} |f(x)| dx < \infty \text{ a.e.} \right\}.$$

$L^\infty(\Omega)$ is complete equipped with the norm $\|f\|_{L^\infty(\Omega)} := \operatorname{ess\,sup}_{x \in \Omega} |f(x)| dx$ and therefore again a Banach space. With $L^1_{loc}(\Omega)$ we denote the space of all measurable functions $f \in C^1(\Omega)$, which are integrable over each compact subset $K \subset \Omega$, hence $f \in L^1(K)$ for all K .

By $W^{k,p}(\Omega)$, with $k \geq 1$ and $1 \leq p \leq \infty$, we denote the well known Sobolev space of functions admitting weak derivatives up to order k in $L^p(\Omega)$ by

$$W^{k,p}(\Omega) := \{f \in L^p(\Omega) : D^\alpha y \in L^p(\Omega) \text{ for all } |\alpha| \leq k\},$$

where D^α are the weak partial derivatives of order $|\alpha| \leq k$, for a multiindex $\alpha = (\alpha_1, \dots, \alpha_n) \in \mathbb{N}_0^n$ with order $|\alpha| := \sum_{i=1}^n \alpha_i$. For $1 \leq p < \infty$, we equip $W^{k,p}(\Omega)$

with the norm

$$\|f\|_{W^{k,p}(\Omega)} := \left(\sum_{|\alpha| \leq k} \int_{\Omega} |D^{\alpha} f(x)|^p dx \right)^{1/p}$$

and $W^{k,\infty}(\Omega)$ is equipped with the norm

$$\|f\|_{W^{k,\infty}(\Omega)} := \max_{|\alpha| \leq k} \|D^{\alpha} f(x)\|_{L^{\infty}(\Omega)}.$$

Note, that all Sobolev spaces $W^{k,p}(\Omega)$ are Banach spaces. In the case $p = 2$, we write as usual $H^k(\Omega) := W^{k,2}(\Omega)$. $H^k(\Omega)$ is a Hilbert space, since it is complete with respect to the norm induced by the scalar product. To treat boundary values of functions in Sobolev spaces, we introduce the space $W_0^{k,2}(\Omega)$ as the closure of $C_0^{\infty}(\Omega)$ with respect to the norm $\|\cdot\|_{W^{k,p}(\Omega)}$. Moreover we set

$$H_0^k(\Omega) := W_0^{k,2}(\Omega).$$

Inhomogeneous boundary values are treated with the trace operator concerning functions in $W^{k,p}(\Omega)$, where the following proposition holds [3].

Proposition 2.1. *Assume an open and bounded domain $\Omega \subset \mathbb{R}^n$ with Lipschitz boundary $\partial\Omega$. For $k \in \mathbb{N}$ and $1 \leq p < \infty$, there exists a unique bounded linear operator $\tau : W^{k,p}(\Omega) \rightarrow L^p(\partial\Omega)$, such that $\tau f = f|_{\partial\Omega}$ for all $f \in W^{k,p}(\Omega) \cap C(\bar{\Omega})$.*

Proof. For a proof of this proposition we refer e.g., to [3]. □

Remark 2.2. 1. We call τf the trace of f on $\partial\Omega$,

2. The trace operator is continuous: $\|\tau f\|_{L^p(\partial\Omega)} \leq c_{\tau} \|f\|_{W^{k,p}(\Omega)}$ with $c_{\tau} > 0$,
3. For bounded Lipschitz domains applies: $H_0^1(\Omega) = \{f \in H^1(\Omega) : \tau f = 0\}$,
4. The trace operator cannot be continuously extended to $L^p(\Omega)$.

Beside the standard Sobolev spaces $W^{k,p}(\Omega)$ with $k \in \mathbb{N}$, a generalization to intermediate Sobolev spaces of fractional order will be necessary to treat questions of existence and uniqueness within this work. Therefore, we introduce the Sobolev-Slobodetskii spaces $W^{s,p}(\Omega)$ with $s > 0$, which coincide with $W^{k,p}(\Omega)$ for integer values of s . For $s > 0$ and $s \notin \mathbb{N}$, the fractional Sobolev-Slobodetskii spaces $W^{s,p}(\Omega)$ are equipped with the norm

$$\|f\|_{W^{s,p}(\Omega)} := \|f\|_{W^{[s],p}(\Omega)} + \sum_{|\alpha|=[s]} \left(\int_{\Omega} \int_{\Omega} \frac{|D^{\alpha} f(x) - D^{\alpha} f(y)|^p}{|x-y|^{n+(s-[s])p}} dx dy \right)^{1/p}.$$

2.1. Basic Notations and Definitions

Here, $s - [s]$ denotes the largest integer smaller than s , where $[s] = s - [s]$. The fractional order Sobolev spaces are also described with so-called spaces of Bessel potentials [3]. The Bessel potential spaces of order s are denoted by $H^{s,p}(\Omega)$, where $1 \leq p < \infty$ and $s \in \mathbb{R}$. Hence, they are an extension of the Hilbert spaces $H^k(\Omega)$ to the non integer case. For $k \in \mathbb{N}_0$ and $1 < p < \infty$ it holds that $H^{k,p}(\Omega) = W^{k,p}(\Omega)$. Therewith, we define the homogeneous Sobolev space $\dot{H}^{1,p}(\Omega)$ by

$$\dot{H}^{1,p}(\Omega) := \{f \in L^1_{loc}(\Omega) : \|\nabla f\|_{L^p(\Omega)} < \infty\},$$

equipped with the norm

$$\|f\|_{\dot{H}^{1,p}(\Omega)} := \|\nabla f\|_{L^p(\Omega)} = \left(\sum_{j=1}^n \|D_j f\|_{L^p(\Omega)}^p \right)^{1/p}.$$

For sufficiently smooth domains, the intermediate spaces can also be represented by the so-called Besov spaces $B^{s,p,q}(\Omega)$ and it holds $W^{s,p}(\Omega) = B^{s,p,p}(\Omega)$ if $s > 0$ and $s \notin \mathbb{N}$ [3]. Besov spaces are especially useful in the study of boundary-value problems. For $p, q = 2$ they also coincide with the Bessel potential spaces.

A further class of function spaces are the Bochner spaces, which generalize the L^p spaces to functions over general Banach spaces. They are very useful when dealing with time dependent PDEs, since they allow us in some way to decouple the variables in time and space. Before we can define the Bochner spaces, another concept of measurability has to be introduced.

Definition 2.3. *Let X be a separable Banach space. A function $f : [0, T] \ni t \mapsto f(t) \in X$ is called strongly measurable, if there exists a sequence of step functions $(s_k)_{k \in \mathbb{N}} : [0, T] \rightarrow X$ such that*

$$\|s_k(t) - f(t)\|_X \rightarrow 0 \quad \text{for almost all } t \in [0, T].$$

This is used for the definition of Banach space valued Lebesgue spaces, also denoted as Bocher spaces.

Definition 2.4. *Let X be a separable Banach space and $I := [0, T]$ an interval in \mathbb{R} . For $1 \leq p \leq \infty$, we define the Bochner-Lebesgue space by*

$$L^p(I; X) := \left\{ f : I \rightarrow X \text{ strongly measurable} : \int_I \|f(t)\|_X^p dt < \infty \right\}$$

with the norm

$$\|f\|_{L^p(I;X)} := \left(\int_I \|f(t)\|_X^p dt \right)^{1/p}.$$

To describe relations and dependencies of the introduced function spaces and the functions defined on it, various embedding theorems exist. In the following, we will develop some results useful for our studies, starting from the normed spaces X and Y . By $\mathcal{L}(X, Y)$, we denote the space of bounded linear operators $F : X \rightarrow Y$, equipped with the operator norm

$$\|F\|_{X,Y} := \sup_{\|u\|_X=1} \|Fu\|_Y < \infty.$$

We write $\mathcal{L}(X) := \mathcal{L}(X, X)$ as customary. With $X^* := \mathcal{L}(X, \mathbb{R})$ we define the dual space of X . Note, that $\mathcal{L}(X, Y)$ is a Banach space if Y is one. For such linear operators between Banach spaces we introduce further concepts of differentiability. Let X and Y be Banach spaces and $U \subset X$ open.

Definition 2.5 (Directional Derivative). *The mapping $F : U \rightarrow Y$ is called directionally differentiable in $u \in U$, if there exists the limit*

$$dF(u, h) = \lim_{t \rightarrow 0^+} \frac{1}{t} (F(u + th) - F(u)) \in Y$$

for all $h \in X$. $dF(u, h)$ is called directional derivative of F in the direction h .

Moreover, if a directional derivative is bounded and linear, i.e., $F'(u) \in \mathcal{L}(X, Y)$ with $F'(u) : X \ni h \mapsto dF(u, h)$, F is called Gâteaux differentiable in u . This is also used in the next definition.

Definition 2.6 (Fréchet Derivative). *The mapping $F : U \rightarrow Y$ is called Fréchet differentiable in $u \in U$, if there exists an operator $F' \in \mathcal{L}(X, Y)$ and a mapping $r(u, \cdot) : X \rightarrow Y$ such that*

$$F(u + h) = F(u) + F'h + r(u, h) \quad \forall h \in X : u + h \in U,$$

and for the remainder term r applies

$$\frac{\|r(u, h)\|_Y}{\|h\|_X} \rightarrow 0 \quad \text{for } \|h\|_X \rightarrow 0.$$

Then, F' is called the Fréchet derivative of F in u .

2.1.2 Differential Operators and Theorems

To describe the announced equations, we need to introduce some further fundamental notations and definitions of operators, see e.g., [81]. Let $\Omega \subset \mathbb{R}^n$, $n \in \{2, 3\}$, be a bounded Lipschitz domain with boundary $\partial\Omega$. The gradient of a continuous differentiable function $y : \Omega \rightarrow \mathbb{R}$ with respect to $x \in \Omega$ is denoted by

$$\nabla y(x) = \begin{pmatrix} \partial_{x_1} y(x) \\ \vdots \\ \partial_{x_d} y(x) \end{pmatrix} \in \mathbb{R}^n.$$

As customary, we write $\partial_{x_i} y$ for the partial derivative of y with respect to the variable x_i . The vector gradient of a differentiable vector field $y : \Omega \rightarrow \mathbb{R}^n$ is denoted by

$$\nabla y(x) = \begin{pmatrix} \nabla y_1(x)^\top \\ \vdots \\ \nabla y_d(x)^\top \end{pmatrix} \in \mathbb{R}^{n \times n}.$$

With $\partial_\nu y$ we denote the derivative in the direction of the outer unit normal ν of $\partial\Omega$

$$\partial_\nu y(x) = \nabla y(x) \cdot \nu(x), \quad \text{for } x \in \partial\Omega.$$

In the following, we will also use the time derivative $\frac{\partial y}{\partial t}$ with the short notation $\partial_t y$. Furthermore, we will use the following standard notations for differential operators

$$\Delta y(x) = \sum_{j=1}^n \partial_{x_j x_j}^2 y(x), \quad \nabla \cdot y(x) = \operatorname{div} y(x) = \sum_{j=1}^n \partial_{x_j} y(x)_j.$$

Here, Δy is the Laplace operator and $\nabla \cdot y$ the divergence of $y \in \Omega$. Moreover, it holds

$$(y(x) \cdot \nabla) = \sum_{j=1}^n y(x)_j \partial_{x_j},$$

where the operator is applied component wise to a vector field.

A fundamental theorem in analysis for differential equations is Banach's fixed point theorem. It forms the basis of further important results like the implicit function theorem or the Picard-Lindelöf theorem and provides the existence and uniqueness of fixed points.

Theorem 2.7 (Banach's Fixed Point Theorem). *Let X be a Banach space, $\emptyset \neq K \subset X$ a closed set and $T : K \rightarrow K$ a contraction mapping, which means that for $0 < \Theta < 1$ it applies*

$$\|Tx - Ty\| \leq \Theta \|x - y\| \quad \forall x, y \in K.$$

Then, T admits a unique fixed point in K .

Proof. A proof can for example be found in [81]. □

The theorem is valid in infinite-dimensional spaces, even in general complete metric spaces. Furthermore, no convexity assumption is needed. To prove Fréchet differentiability of control-to-state mappings in the framework of PDE-constrained optimization, the following implicit function theorem is an important tool [41].

Theorem 2.8 (Implicit Function Theorem). *Let X, Y, Z be Banach spaces and let $F : G \rightarrow Z$ be a continuously Fréchet differentiable map from an open set $G \subset X \times Y$ to Z . Let $(\bar{x}, \bar{y}) \in G$ be such that $F(\bar{x}, \bar{y}) = 0$ and that $F_y(\bar{x}, \bar{y}) \in \mathcal{L}(X, Y)$ has a bounded inverse. Then there exists an open neighborhood $U_X(\bar{x}) \times U_Y(\bar{y}) \subset G$ of (\bar{x}, \bar{y}) and a unique continuous function $w : U_X(\bar{x}) \rightarrow Y$ such that*

- (i) $w(\bar{x}) = \bar{y}$.
- (ii) *For all $x \in U_X(\bar{x})$ there exists exactly one $y \in U_Y(\bar{y})$ with $F(x, y) = 0$, namely $y = w(x)$.*

Moreover, the mapping $w : U_X(\bar{x}) \rightarrow Y$ is continuously Fréchet differentiable with derivative

$$w'(x) = F_y(x, w(x))^{-1} F_x(x, w(x)).$$

If $F : G \rightarrow Z$ is m -times continuously Fréchet differentiable then also $w : U_X(\bar{x}) \rightarrow Y$ is m -times continuously Fréchet differentiable.

Proof. The proof can be found in [106]. □

Our investigations are mostly done in \mathbb{R}^2 or \mathbb{R}^3 so we have to deal with surface and volume integrals, frequently in the same equation. To overcome this problem, we apply the divergence theorem to convert surface and volume integrals into the respective other. The divergence theorem, as well known as Gauss's theorem, is the most important theorem in integral calculus of \mathbb{R}^n . It is the n -dimensional analogon to the fundamental theorem of calculus for functions of one variable. The theorem enables to replace a volume integral of a vector field by a surface integral [29].

Theorem 2.9 (Divergence Theorem). *Let $\Omega \subset \mathbb{R}^n$ be a compact subset with a smooth boundary, $\nu : \partial\Omega \rightarrow \mathbb{R}^n$ the outer unit normal field and $U \supset \Omega$ an open subset of \mathbb{R}^n . Then the following equation holds for every continuously differentiable vector field $F : U \rightarrow \mathbb{R}^n$*

$$\int_{\Omega} \operatorname{div} F(x) \, d^n x = \int_{\partial\Omega} F(x) \cdot \nu(x) \, ds. \quad (2.1)$$

Proof. See for example paragraph 15 in [29]. □

From a physical point of view, the divergence theorem states that the outward flux of a vector field through a closed surface is equal to the volume integral of the divergence over the region inside the surface [29]. Even if the boundary of Ω is not smooth but contains low dimensional singularities like edges or vertexes and the vector field F is not continuously differentiable in a full vicinity around Ω , the divergence theorem still holds [56].

2.2 Conservation Equations

Conservation equations are the mathematical formulation of fundamental physical laws. They describe the conservation of a quantity in a closed system. Conservation equations for mass, momentum and energy, together with state equations, lead to a full description of macroscopic fluid flow. Since the focus of this thesis is on impulse transport, temperature changes do not play a role and therefore energy conservation can be neglected. At this point, however, it should be noted that the energy conservation equation is assumed to be fulfilled for all further investigations. But it is not necessary to solve the energy equation explicitly to find the velocity and the pressure, unless the material properties are functions of the temperature, which is not the case in our problem formulation.

2.2.1 Conservation of Mass

The principle of conservation of mass states, that mass cannot be created or destroyed [93]. Let V be a volume that is fix in space. We assume, that the continuum hypothesis holds, which declares that the density of a fluid volume can be well approximated with a continuous function ρ .

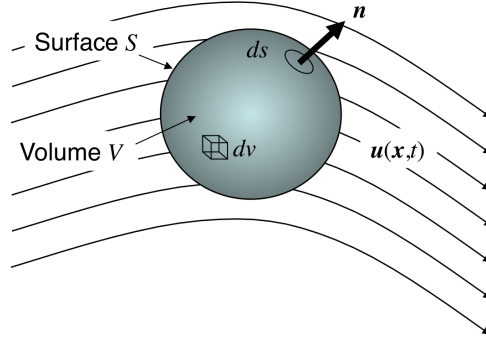


Figure 2.1. Stationary control volume V with surface S [93].

The mass M of the volume is then calculated as integral of the density ρ over V

$$M = \int_V \rho \, dv.$$

Applying the conservation theory, a change of mass in the volume is a result of mass transport across its boundary S . Based on Reynolds transport theorem, the mass flux through a surface element ds is the product of ρ and the velocity u of a small mass volume dv , multiplied with the outward normal n

$$\frac{d}{dt} M = \frac{d}{dt} \int_V \rho \, dv = - \int_S \rho u \cdot n \, ds.$$

See Figure 2.1 for the notations. Now we apply the divergence theorem, equation (2.1), to convert the surface integral to a volume integral and obtain

$$\frac{d}{dt} \int_V \rho \, dv + \int_V \nabla \cdot (\rho u) \, dv = 0.$$

For a volume that is fix in space, we can additional take the derivative inside the integral

$$\int_V [\partial_t \rho + \nabla \cdot (\rho u)] \, dv = 0.$$

For any arbitrary volume V , this equation is true, if the argument of the integral is equal to zero. This leads to the partial differential equation of mass conservation,

which has the following coordinate free vector form

$$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0. \quad (2.2)$$

In the case of incompressible fluids the density does not change over time, so the mass conservation equation simplifies to

$$\nabla \cdot \mathbf{u} = 0, \quad (2.3)$$

which we will denote as continuity equation in the following.

2.2.2 Conservation of Momentum

The principle of momentum conservation indicates, that the rate of change of fluid momentum in a fixed volume V is the difference in momentum flux across the boundary S plus the net forces acting on V [93]. This results in

$$\frac{d}{dt} \int_V \rho \mathbf{u} \, dv = - \int_S (\rho \mathbf{u} \otimes \mathbf{u}) \cdot \mathbf{n} \, ds + \int_V \mathbf{f} \, dv + \int_S \mathbf{T} \cdot \mathbf{n} \, ds. \quad (2.4)$$

The net forces acting on the volume are composed of the total body force on V and the total surface force on S . The body force \mathbf{f} per unit volume can for example include the gravitational force, centrifugal and Coriolis forces, electromagnetic forces, etc. Surface forces are pressures, normal and shear stresses, surface tension, etc. [28]. The term of the total surface force contains a symmetric stress tensor \mathbf{T} [93], where \mathbf{n} multiplied with \mathbf{T} is the force on a surface element ds with a normal \mathbf{n} . For Newtonian fluids, the stress may be assumed to be a linear function of the rate of strain \mathbf{S} [93]

$$\mathbf{T} = (-p + \lambda \nabla \cdot \mathbf{u}) \mathbf{I} + 2\mu \mathbf{S}, \quad (2.5)$$

where p is the pressure, λ the second coefficient of viscosity, \mathbf{I} the unit tensor, μ the viscosity and $\mathbf{S} = \frac{1}{2}(\nabla \mathbf{u} + \nabla \mathbf{u}^\top)$ the rate of strain or deformation tensor [93]. If we assume Stokes' hypothesis to hold, we can choose $\lambda = -(\frac{2}{3})\mu$ [93], which is also referred to be the dilatation viscosity.

By applying the divergence theorem to equation (2.4), similar to the approach for mass conservation in Section 2.2.1, we receive the following expression

$$\partial_t (\rho \mathbf{u}) = -\nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) + \mathbf{f} + \nabla \cdot \mathbf{T}. \quad (2.6)$$

If we use the definition of the substantial derivative, equation (2.6) corresponds exactly to Cauchy's equation of motion and is valid for any continuous medium [93]. Together with the selected stress tensor, equation (2.5), Cauchy's equation of motion results in the momentum equation for fluid flow

$$\partial_t(\rho u) + \nabla \cdot (\rho u \otimes u) = f - \nabla p + \nabla(\lambda \nabla \cdot u) + \nabla \cdot (2\mu \mathbf{S}). \quad (2.7)$$

As already mentioned, we focus on incompressible fluids in this work. This implies a divergence-free velocity field, so the term containing λ vanishes. This term is only of importance for compressible fluids. Another point are the gravitational forces we referred to briefly. Usually, gravity can be neglected if the length scale of the problem is well below the capillary length $l_c = \sqrt{\sigma/(\Delta\rho g)}$, see for example [30], where σ is the surface tension coefficient, $\Delta\rho = \rho_2 - \rho_1$ the density difference between two phases and g the gravitational acceleration. This is not the case in the macroscopic wetting processes we consider, so the gravitational acceleration has to be taken into account in our further investigation with $f = \rho g$. Considering these two aspects we obtain the following momentum equation

$$\rho(\partial_t u + u \cdot \nabla u) + \nabla p = \nabla \cdot (2\mu \mathbf{S}) + \rho g. \quad (2.8)$$

The attentive reader will notice from this representation, that the first two terms look different now. Due to the assumption of mass conservation (2.2), it holds

$$\partial_t(\rho u) + \nabla \cdot (\rho u \otimes u) = \rho(\partial_t u + u \cdot \nabla u) + u(\partial_t \rho + \nabla \cdot (\rho u)) = \rho(\partial_t u + u \cdot \nabla u).$$

This even applies to non-constant densities, as long as equation (2.2) holds. Thus, the convective terms within equation (2.7) and (2.8) are equivalent analytically. But note, they can lead to slightly different numerical approximations [93].

2.2.3 Navier-Stokes Equations

The Navier¹-Stokes² equations are a nonlinear system of second order partial differential equations. They fully describe the macroscopic fluid flow and are based on the conservation equations introduced before.

For a given domain $\Omega \subset \mathbb{R}^n$, $n \in \{2, 3\}$, a constant fluid density $\rho > 0$ and a dynamic viscosity $\mu > 0$, the incompressible Navier-Stokes equations consist of the

¹Claude Louis Marie Henri Navier, 1785 - 1836

²George Gabriel Stokes, 1819 - 1903

2.3. Standard Description of Two-phase Flow

following momentum and continuity equations [81], here for the case with gravity

$$\begin{aligned}\rho(\partial_t u + u \cdot \nabla u) + \nabla p &= \mu \Delta u + \rho g, \\ \nabla \cdot u &= 0.\end{aligned}\tag{2.9}$$

The fluid velocity $u : \Omega \times I \rightarrow \mathbb{R}^n$ and the pressure $p : \Omega \times I \rightarrow \mathbb{R}$ are unknown. So we obtain four partial differential equations with four unknowns for the three dimensional case. The Navier-Stokes equations are nonlinear because of the convective term $(u \cdot \nabla)u$. Additional boundary conditions, e.g. a no-slip condition for the velocity, and an initial velocity field complement problem (2.9) to be local well posed [74]. Up to now, a global analytical solution of the Navier-Stokes equations in the general three dimensional case was not verified. But there are extensive statements of existence, uniqueness and regularity for two dimensions [70].

Note, often a simplified version of the Navier-Stokes equations is used. If we want to describe a very slow flow, the absolute value of the velocity is small everywhere. Then, neglecting the nonlinear convective term is a reasonable simplification which leads to the so-called Stokes equations [81]. Here, effects due to inertia are completely ignored. Another special case are the Euler equations, where the viscosity is assumed to be very small so the diffusion term of equation (2.9) vanishes. These equations are for example used to describe inviscid fluids or problems with large length scales [81]. Furthermore, laminar flows, such as a pipe flow, can be reproduced very well with this model. However, the simplification of the model is also accompanied by the fact that turbulence can not be represented with the Euler equations. In consequence, both simplifications are not valid for possibly highly dynamic wetting processes, so we have to deal with the full Navier-Stokes equations.

2.3 Standard Description of Two-phase Flow

For our considerations, we start with the description of a two-phase problem without the contact angle problem. We consider the case of two viscous incompressible capillary Newtonian fluids, which are separated by a hypersurface Γ . In Figure 2.2, we see a time dependent interface $\Gamma(t)$ which separates our domain $\Omega \subset \mathbb{R}^{n+1}$, $n \leq 1$, into two open sets $\Omega_1(t)$ and $\Omega_2(t)$, where $\Omega := \Omega_1(t) \cup \Omega_2(t)$ and $\Gamma(t) := \overline{\Omega_1(t)} \cap \overline{\Omega_2(t)}$ with $t \in I := [0, T]$. Moreover, we denote the normal field on $\Gamma(t)$ by $\nu(t, \cdot)$, pointing from $\Omega_1(t)$ into $\Omega_2(t)$ [70]. The utilized formulation of such a two-phase problem with sharp interfaces is based on the formulation by Prüss and Simonett in [70] and will be presented in the following.

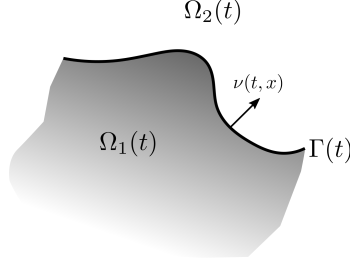


Figure 2.2. Interface $\Gamma(t)$ separates open sets $\Omega_1(t)$ and $\Omega_2(t)$.

With the previous introduced conservation equations we can summarize the mathematical model of a two-phase fluid flow. Therefore, we have to solve the Navier-Stokes equations in every bulk phase $\Omega_i(t)$ with a constant density ρ_i and viscosity μ_i , $i = 1, 2$

$$\begin{aligned} \rho_i(\partial_t u + u \cdot \nabla u) - \mu_i \Delta u + \nabla p &= \rho g & \text{in } \Omega_i(t), \\ \nabla \cdot u &= 0 & \text{in } \Omega_i(t). \end{aligned}$$

Together with the Navier-Stokes equations, we need jump conditions to connect the flow of the different fluids or phases at their interface. The first condition (2.10) describes the continuity of velocities on both sides of the interface, while the second condition (2.11) can be seen as the balance of forces across the interface. It applies

$$[[u]] = 0 \quad \text{on } \Gamma(t), \quad (2.10)$$

$$-[[S(u, p; \mu_i)\nu]] = \sigma \kappa \nu \quad \text{on } \Gamma(t). \quad (2.11)$$

Here, $S(u, p; \mu_i) := -p\mathbf{I} + \mu_i(\nabla u + \nabla u^\top)$ is the viscous stress tensor on $\Omega_i(t)$ respectively with $i = 1, 2$, $\sigma > 0$ is a given surface tension coefficient and κ is the mean curvature of the interface. For a sufficiently smooth $\Gamma(t)$ the mean curvature is given by $\kappa(t, \cdot) = -\text{div}_\Gamma \nu(t, \cdot)$. The brackets $[[\cdot]]$ denote the jump across the interface in direction of ν , which is defined as

$$[[\phi]](t, x) := \lim_{h \rightarrow 0^+} (\phi(t, x + h\nu_\Gamma) - \phi(t, x - h\nu_\Gamma)), \quad \text{for } x \in \Gamma(t), t \in I.$$

The second jump condition, equation (2.11), illustrates the effect of surface tension, which introduces a discontinuity in the normal component of the jump of the viscous stress tensor $[[S(u, p; \mu_i)\nu]]$ proportional to the mean curvature κ [70]. Furthermore, there is an additional condition on the interface, which ensures the transport of the

interface by the fluid velocity

$$V = u^\top \nu \quad \text{on } \Gamma(t),$$

where V denotes the normal velocity of the interface. This condition is also called the kinematic boundary condition, although it should rather be seen as an interface condition. It implies that fluid particles cannot cross the interface [70]. Besides the mentioned interface conditions, appropriate initial conditions belongs to the model. In our case an initial velocity field and the initial position of the interface are required. In summary, we need to solve the following problem with $i = 1, 2$

$$\begin{aligned} \rho_i(\partial_t u + u \cdot \nabla u) - \mu_i \Delta u + \nabla p &= \rho g && \text{in } \Omega_i(t), \\ \nabla \cdot u &= 0 && \text{in } \Omega_i(t), \\ \llbracket u \rrbracket &= 0 && \text{on } \Gamma(t), \\ -\llbracket S(u, p; \mu_i) \nu \rrbracket &= \sigma \kappa \nu && \text{on } \Gamma(t), \\ V &= u^\top \nu && \text{on } \Gamma(t), \\ u(0) &= u_0 && \text{on } \Omega(0), \\ \Gamma(0) &= \Gamma_0. \end{aligned} \tag{2.12}$$

Problem (2.12) is a free boundary or rather a moving boundary problem, due to the fact that the position of Γ is part of the problem and that there are no fixed domains Ω_i . For a clearer representation, we introduce the following notation we will use frequently in the further investigations, where χ_{Ω_i} denotes the indicator function of the set Ω_i

$$\rho = \rho_1 \chi_{\Omega_1} + \rho_2 \chi_{\Omega_2}, \quad \mu = \mu_1 \chi_{\Omega_1} + \mu_2 \chi_{\Omega_2}. \tag{2.13}$$

Note, that this formulation includes no boundary conditions for an outer boundary $\partial\Omega$. This part of the problem description will be considered in Section 2.5, after we presented some existence and uniqueness results for problem (2.12).

Existence and Uniqueness

Results for the existence and uniqueness of the two-phase Navier-Stokes equations with surface tension are for example derived from PRÜSS and SIMONETT in [70, 68] and from KÖHNE et al. in [55] for bounded fluid domains. As we stated before, this is only possible under a certain regularity. In this case, existence and uniqueness is shown by means of L^p -maximal regularity, which we will define in Section 3.3.2

more precisely. The initial interface Γ_0 is assumed to be close to a halfplane and is described with the help of a graph of a function h_0 on \mathbb{R}^n . A so-called smallness condition is placed on this initial interface function h_0 and the initial velocity u_0 , which guarantees sufficiently small data. With the compatibility conditions, the initial data are proven to fulfill necessary conditions of being a solution of the problem. Therewith, PRÜSS and SIMONETT show the following result, see [70], which at first is only valid for a homogeneous right side of the momentum equation, but can also be applied to the case with gravity using a small correction.

Theorem 2.10 (Existence, Uniqueness and Regularity of problem (2.12)).
 (a) Assume that $p > n+3$. Then there exists $\varepsilon_0 = \varepsilon_0(t_0) > 0$ for a given $t_0 > 0$, such that for any initial values $(u_0, h_0) \in W^{2-2/p,p}(\Omega_0, \mathbb{R}^{n+1}) \times W^{3-2/p,p}(\mathbb{R}^n)$, satisfying the following compatibility conditions

$$\llbracket \mu D(u_0)\nu_0 - \mu(\nu_0^\top D(u_0)\nu_0)\nu_0 \rrbracket = 0, \quad \operatorname{div} u_0 = 0 \quad \text{on } \Omega_0, \quad \llbracket u_0 \rrbracket = 0,$$

with $D(u_0) := \nabla u_0 + (\nabla u_0)^T$, and the smallness condition

$$\|u_0\|_{W^{2-2/p,p}(\Omega_0)} + \|h_0\|_{W^{3-2/p,p}(\mathbb{R}^n)} \leq \varepsilon_0,$$

problem (2.12) has a classical solution (u, p, Γ) on $(0, t_0)$.

(b) The solution $(u, p, \llbracket p \rrbracket, h)$ is unique in the following function class

$$\mathbb{E}(t_0) := \{(u, p, r, h) \in \mathbb{E}_1(t_0) \times \mathbb{E}_2(t_0) \times \mathbb{E}_3(t_0) \times \mathbb{E}_4(t_0) : \llbracket p \rrbracket = r\},$$

$$\text{with } \mathbb{E}_1(t_0) := \{u \in H^{1,p}(I; L^p(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})) \cap L^p(I; H^{2,p}(\dot{\mathbb{R}}^{n+1}, \mathbb{R}^{n+1})) : \llbracket u \rrbracket = 0\},$$

$$\mathbb{E}_2(t_0) := L^p(I; \dot{H}^{1,p}(\dot{\mathbb{R}}^{n+1})),$$

$$\mathbb{E}_3(t_0) := W^{1/2-1/(2p),p}(I; L^p(\mathbb{R}^n)) \cap L^p(I; W^{1-1/p,p}(\mathbb{R}^n)),$$

$$\mathbb{E}_4(t_0) := W^{2-1/(2p),p}(I; L^p(\mathbb{R}^n)) \cap H^{1,p}(I; W^{2-1/p,p}(\mathbb{R}^n)) \\ \cap W^{1/2-1/(2p),p}(I; H^{2,p}(\mathbb{R}^n)) \cap L^p(I; W^{3-1/p,p}(\mathbb{R}^n)).$$

(c) $\Gamma(t)$ is the graph of a function $h(t)$ on \mathbb{R}^n , $\mathcal{M} = \bigcup_{t \in (0, t_0)} (\{t\} \times \Gamma(t))$ is a real analytic manifold, and with $\mathcal{O} = \{(t, x, y) : t \in (0, t_0), x \in \mathbb{R}^n, y \neq h(t, x)\}$, the function $(u, p) : \mathcal{O} \rightarrow \mathbb{R}^{n+2}$ is real analytic.

The extensive proof can be found in [70]. Therein, the approach is to transform the free boundary problem to a problem with fixed interface, which is also denoted as direct mapping method [69]. We will make use of this result later for our differentiability results of a two-phase flow problem see Section 3.3.

2.4 Related One-Field Formulation

As we have seen in the previous sections, the macroscopic fluid flow can be modeled with the aid of conservation equations like conservation of mass and momentum. In general, these equations have to be solved for every phase separately and jump conditions are used to couple the solutions at the interfaces. In contrast to this approach, it is possible to write one set of governing equations for the whole flow domain occupied by the various phases, without resorting to jump conditions. The various phases are treated as one single fluid with variable material properties that change abruptly at the phase boundary [93]. These so-called one-field or one-fluid methods differ in their indicator functions and the related advection, where an indicator function specifies how the fluid is distributed in a domain.

Thereby we also distinguish between sharp interface and diffuse interface models. Established sharp interface methods are for example the level-set method [89, 82], or the volume of fluid (VOF) method [42]. While the interface is the zero-level set of an additional state variable ϕ in the level-set method, the VOF method defines the interface as sharp transition between zero and one, where for example the liquid filled region is set to one and the gaseous phase region is set to zero. In contrast to the continuous level-set function, this results in a non-continuous indicator function within the VOF approach. However, this disadvantage is compensated by the useful property, that the VOF approach holds an inherent mass conservation of the different phases. An example for a diffuse interface model is the phase-field approach, where the interface has a finite thickness and is described in a thermodynamically consistent way. This results in a more smooth transition of material and transport quantities. The methods also differ in how the surface tension is modeled [18], what we will discuss in more detail in Section 2.4.2. In this work we will consider the volume of fluid method, where we use a phase fraction function α as indicator function. The method will be described in the next section for the case of a gas and a liquid phase.

2.4.1 Volume of Fluid Approach

The VOF approach belongs to the interface capturing methods, where the discontinuity of the interface is computed as part of the solution. Here, no special treatment is employed to take care of the interface itself. That means, that the interface is not explicitly introduced into the solution using appropriate interface relations. Hence, no a priori assumptions on the nature of the fluid interface are

required, which has the beneficial effect, that also strong topology changes can be handled like the breakup of fluid [72]. In the numerical solution, the interface capturing methods can lead to a not sharp interface, since it may be smeared over several grid elements when time advances. But nevertheless, the one-field formulation of the Navier-Stokes equations has the big advantage, that it allows to use numerical methods developed for single-phase flows [93]. We will discuss these methods in Chapter 4.

For the physical model, we assume $\Omega \subset \mathbb{R}^n, n = \{2, 3\}$, is a physical domain, composed of a liquid phase region Ω_l and a gas phase region Ω_g with $\Omega = \Omega_l(t) \cup \Omega_g(t)$ for $t \in [0, T] = I$. Define the indicator function $\alpha : \mathbb{R}^n \times I \rightarrow [0, 1]$ as

$$\alpha(x, t) := \begin{cases} 1 & \text{if } x \in \Omega_l(t), \\ 0 & \text{if } x \in \Omega_g(t). \end{cases} \quad (2.14)$$

This indicator function is a step function, which is neither continuous nor differentiable or weakly differentiable. But α has a distributional derivative, which will be important for the theoretical investigations in Chapter 3.3.6. For the numerical consideration, α is approximated by integrating over a certain volume, which we will discuss in Chapter 4.1.1. With the integral we avoid the singular nature of the step function, leading to values $\alpha \in (0, 1)$ within the respective volumes. Then, the gradient of α exists at the interface and the normal vector of the interface can be approximated with respect to α [76] by

$$\nu_\Gamma(x, t) = \frac{\nabla \alpha(x, t)}{|\nabla \alpha(x, t)|} \quad \text{on } \Gamma(t). \quad (2.15)$$

With this definition, the curvature can be written in terms of derivatives of α

$$\kappa(x, t) = -\operatorname{div} \nu_\Gamma(x, t) = -\operatorname{div} \frac{\nabla \alpha(x, t)}{|\nabla \alpha(x, t)|} \quad \text{on } \Gamma(t). \quad (2.16)$$

Note, that the curvature is only defined at the interface Γ . For the density and viscosity in Ω we define

$$\rho(\alpha) = \alpha \rho_l + (1 - \alpha) \rho_g, \quad (2.17)$$

$$\mu(\alpha) = \alpha \mu_l + (1 - \alpha) \mu_g, \quad (2.18)$$

where ρ_l and ρ_g as well as μ_l and μ_g are constant values for the respective phase. Note, that density and viscosity are also discontinuous at the interface Γ , due to their dependency of α .

In a dynamic system, the fluid moves through the domain. Hence, the gas-liquid interface is transported with the velocity field. This motion is described with a scalar advection equation, which we also denote as α -transport equation. With the above definition of the density, equation (2.17), and the assumption of a divergence-free velocity field, the following hyperbolic transport equation results from the conservation of mass, see equation (2.2)

$$\partial_t \alpha + \nabla \cdot (\alpha u) = 0 \quad (2.19)$$

In Appendix A.1 we provide a detailed derivation of this equation. Due to the absence of diffusive terms in the scalar transport equation, the phase fraction field α remains discontinuous across the interface $\Gamma(t)$. This leads to challenges in numerical considerations, since oscillations or numerical diffusion can occur. With appropriate methods, this can be counteracted and will be discussed in more detail in Chapter 4.

The VOF method also distinguishes between a geometric and an algebraic approach. In the algebraic approach, the interface is solely implicitly given by the phase fraction function α . Contrary to the algebraic approach, the interface is explicitly reconstructed within the geometric VOF method. But this is associated with a considerably larger computational effort in numerical investigations, especially in the case of unstructured meshes. Extensive studies regarding the geometric approach are carried out in [59]. Summarized, the interface position in the applied VOF method is captured implicitly introducing the phase indicator function α for one of the phases along with its corresponding transport equation. In consequence, our flow system is appended by a transport equation with another state variable α in addition to the Navier-Stokes equations, which are itself only solved for velocity and pressure. We consider this one-field formulation since it plays a fundamental role for the numerical consideration of the problem mentioned in Chapter 4.

2.4.2 Modeling the Surface Tension

In methods based on the one-fluid or one-field formulation where an indicator function is defined for the whole domain, no jump conditions arise anymore, in which the surface tension was located mathematically before. But the surface tension plays an important role for the modeling of multiphase flow. It is the surface energy per unit area of a fluid and keeps the fluid shape. So it has a big impact on the accuracy of the solution. Instead of the use of jump conditions, the surface tension is added as a body force to the discrete version of the Navier-Stokes equations [93]. The standard approach in this case is the continuous surface force

(CSF) method, introduced by BRACKBILL et al. in 1992 [18].

In general, the surface tension is a function of temperature and the equilibrium surface tension depends on the pair of contacting materials [84]. For the given problem we assume, that the temperature does not change in time. According to that, effects like the Marangoni Effect are neglected for the moment. Then the surface tension force is given by f_σ integrated over a surface part S . By using the divergence theorem, see Theorem 2.9, the following surface integral is converted to a volume integral, to involve the force as body force into the momentum equation

$$\int_S f_\sigma ds = \int_V f_\sigma \delta_S dv,$$

with Dirac distribution δ_S . The singular terms δ_S form the counterpart of the jump conditions in the classical description and it can be shown that both formulations are equivalent [93]. For a constant surface tension coefficient σ , the surface tension force can be replaced by the term $\sigma\kappa\nu$

$$f_\sigma \delta_S \approx \sigma\kappa\nu\delta_S,$$

where κ is the curvature and ν the normal of the interface. When we replace the interface normal by (2.15) and approximate the Dirac distribution δ_S by $|\nabla\alpha|$, which corresponds to the CSF approach according to [93], we obtain the following statement

$$f_\sigma \delta_S \approx \sigma\kappa\nabla\alpha.$$

Hence, with this expression we arrive at the following approximation of the momentum equation (2.8) including the surface tension [18]

$$\rho(\partial_t u + u \cdot \nabla u) - \mu\Delta u - \rho g + \nabla p = \sigma\kappa\nabla\alpha. \quad (2.20)$$

In summary, the VOF representation of the introduced multiphase problem consists of the state equations (2.20), (2.3) and (2.19), the additional equations for density and viscosity (2.17) and (2.18) as well as appropriate initial conditions for the three state variables velocity u , pressure p and phase fraction α . Due to the interface capturing character of this formulation and the fact, that all variables are defined for the whole domain, Ω is not dependent on the time anymore.

Summarized we obtain

$$\begin{aligned}
 \rho(\partial_t u + u \cdot \nabla u) - \mu \Delta u - \rho g + \nabla p &= \sigma \kappa \nabla \alpha && \text{in } \Omega \times I, \\
 \nabla \cdot u &= 0 && \text{in } \Omega \times I, \\
 \partial_t \alpha + \nabla \cdot (\alpha u) &= 0 && \text{in } \Omega \times I, \\
 \rho_l \alpha + \rho_g (1 - \alpha) &= \rho && \text{in } \Omega \times I, \\
 \mu_l \alpha + \mu_g (1 - \alpha) &= \mu && \text{in } \Omega \times I, \\
 (u, p, \alpha)(0) &= (u, p, \alpha)_0 && \text{in } \Omega(0).
 \end{aligned} \tag{2.21}$$

Note, that this problem, similar to problem (2.12), does not contain boundary conditions for an outer boundary $\partial\Omega$. This part of the problem formulation will be treated in the following sections and is also not yet taken into account for the theoretical investigations of the optimization problems in chapter 3. Hence, differentiability results are derived in Section 3.3 for problem (2.12) and (2.21), in both cases without boundary conditions and without the contact line problem. This is already a challenging task and has not yet been investigated, so far as we know. The full VOF system with appropriate boundary conditions including a dynamic contact angle treatment will be applied for the numerical considerations and the applications, dealing with in the Chapters 4 and 5.

2.5 Boundary Conditions

So far, we did not considered an outer boundary of our domain, although it is an important part of wetting problems. From now on we suppose a bounded domain Ω with Lipschitz boundary $\partial\Omega$, which is filled with two different fluids or phases.

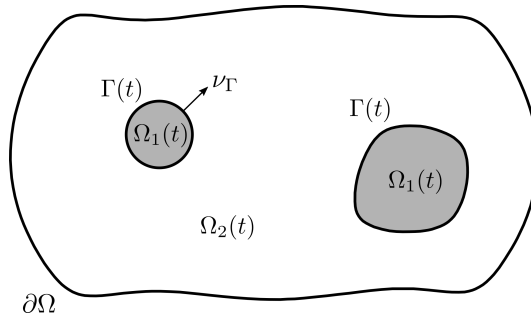


Figure 2.3. A two-phase domain with outer boundary $\partial\Omega$ [69].

As depicted in Figure 2.3, only one fluid touches the outer boundary, so no contact line between the fluids and a solid surface is included at this point. Remember, that we say boundary if we talk about the outer boundary $\partial\Omega$ and indicate the boundary between the different fluids as interface Γ . The most common boundary condition for viscous fluid flow is the no-slip condition, a homogeneous Dirichlet condition for both the normal and the tangential component of the velocity

$$u = 0 \quad \text{on } \partial\Omega. \quad (2.22)$$

If there is an additional movement of the outer boundary, e.g. a tangential wall velocity U_{wall} , the fluid velocity is set equal to the velocity of $\partial\Omega$

$$u = U_{wall} \quad \text{on } \partial\Omega.$$

The two conditions are valid for viscous, incompressible fluids and imply, that the fluid sticks to the boundary [93]. In case that viscous stresses are absent, so when we consider inviscid fluids, the fluid can slip freely at the wall and only the normal velocity is equal to that of the wall [93]. Then, a Navier-slip boundary condition is more suitable, which introduces an artificial slip with slip length $L > 0$ and holds for the tangential velocity component u_t by

$$u_t + L(\mathbf{S}\nu_{\partial\Omega})_t = 0 \quad \text{on } \partial\Omega. \quad (2.23)$$

Here, $\nu_{\partial\Omega}$ denotes the normal of the outer boundary and $\mathbf{S} = \frac{1}{2}(\nabla u + \nabla u^\top)$ again the rate of strain or deformation tensor. Only the tangential part of the product is taken as well. The slip length L is the distance at which the velocity would vanish if it is extrapolated inside the wall [93]. Again, the right hand side of equation (2.23) is set to U_{wall} , if there is a tangential wall velocity. The normal component of the velocity is equal to zero, or, in case of $U_{wall} \neq 0$, equal to the respective value

$$u^\top \nu_{\partial\Omega} = 0 \quad \text{on } \partial\Omega. \quad (2.24)$$

The difference between the no-slip and the Navier-slip condition is clearly shown in Figure 2.4. While the normal and tangential velocity components are equal to zero at the wall if the no-slip condition holds, see the left graph in Figure 2.4, there is a small tangential velocity component at the wall if the Navier-slip condition is applied, see the right graph in Figure 2.4. This allows the flow to slip a bit along the solid surface and a difference between the velocity of the wall and the tangent velocity of the liquid near the wall may occur. In fact, this slip plays an important role for the simulation of real experiments, since the simple no-slip condition is not able to reproduce most of the experimental results in not ideal systems. The

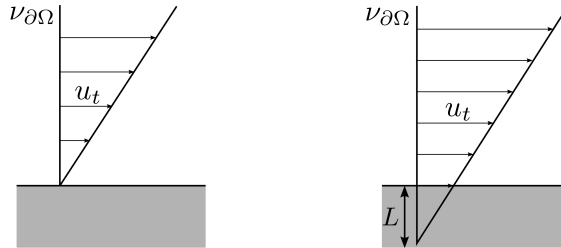


Figure 2.4. No-slip ($L = 0$) on the left and Navier-slip ($0 < L < \infty$) on the right.

slip length depends on the characteristics of the system. A value in the scope of nanometers is suitable for the wetting phenomena considered in this work, since the flow processes close to the contact line take place in the order of a few to tens of micrometers [88]. Note, that from a numerical point of view the real slip length can not necessarily be resolved, then a slip length in order of the mesh size is most efficient [72]. Since these wetting problems mainly deal with inviscid or very low viscosity fluids, applying the Navier-slip condition is appropriate and will be considered in the numerical applications. But there is another reason, why the Navier-slip condition is more suitable in our case. We want to include a three phase contact line in our model, which implies a dynamic contact angle behavior at the boundary. As already known from literature [45] and shown in detail by Mathis Fricke in his dissertation [30], the no-slip boundary condition causes a stress singularity when approaching the contact line and results in a divergence of viscous stresses. This prevents its displacement and is called the moving contact line paradox. However, the Navier-slip condition allows the liquid to move as it slips on the solid surface. A detailed insight to the treatment of dynamic contact lines is given in the next section.

Another important role in fluid dynamics play inflow and outflow boundary conditions, e.g. of special interest in streaming channels or if just a part of the liquid filled domain is considered [93]. Then, artificial boundaries have to be defined for the numerical solution even if the physical problem relates to a larger domain. As inflow condition, a velocity field is prescribed in most of the cases, whereas realistic outflow boundaries are more challenging. Here, a balance has to be found between enlarging the domain to have minimal influence on the upstream flow and keeping the computational costs as low as possible. A standard outflow condition is the so-called do-nothing condition, a homogeneous Neumann condition for both, the

velocity and the pressure at the outflow boundary, defined as

$$(\mu \nabla u - p \mathbf{I}) \cdot \nu_{\partial\Omega} = 0 \quad \text{on } \partial\Omega. \quad (2.25)$$

This condition allows for in- and outflow simultaneously, hence a possible backflow at the outflow boundary is also taken into account. Another inflow boundary with a special inflow condition is not used in the following, so we will not go into it further. The outflow condition (2.25) becomes important in our numerical investigations in Chapter 5 and is reasonable, since the outflow region is far away from the area of interest.

2.6 Dynamic Contact Line Treatment

If besides an interface between different phases and an outer boundary additionally a contact to a solid surface comes into play, we have to consider further hydrodynamic aspects. This is the case because we no longer have just an interface but also a boundary with a three phase contact point between three different phases in two dimensions, or a three phase contact line in the three dimensional case. We write Σ for this part of the outer boundary. Therefore we need to expand our previous models by further equations, in particular by further boundary conditions.

Before introducing suitable equations, we have to examine the different concepts of contact angles. We distinguish between the so-called actual or microscopic contact angle and the macroscopic one. While the microscopic contact angle relates to the surface roughness, the macroscopic contact angle is defined as the angle between the plane surface, that approximates the liquid-gas interface in the vicinity of the contact line, and the solid surface. It is usually assumed to be equal to the static one, even for non-zero contact line speed [84]. Note that the macroscopic contact angle also differs from the apparent contact angle, which is defined as the angle formed by free and solid surfaces far from the contact line. But the main role in the description of the hydrodynamic characteristics of wetting phenomena belongs to the macroscopic contact angle [84], the one we mean when we speak of contact angles in the following.

In a stationary system, a fluid has a specific static contact angle, which depends on the material properties of the fluid and the solid surface. For homogeneous, smooth and flat surfaces the equilibrium contact angle θ_e is described with the Young's

equation [104] as follows

$$\sigma_{lg} \cos(\theta_e) + \sigma_{sl} = \sigma_{sg},$$

where we write σ_{lg} for the surface energy of the liquid-gas interface, σ_{sl} for the surface energy of the solid-liquid interface and σ_{sg} for the surface energy of the solid-gas interface. The surface energy σ_{lg} is what we will refer to in the following as the surface tension σ . The relation of the different surface energies and the equilibrium or static contact angle is depicted in Figure 2.5.

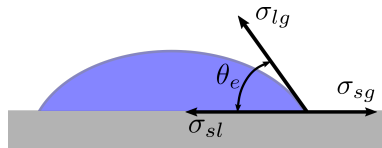


Figure 2.5. Equilibrium contact angle according to Young's law.

For an equilibrium contact angle θ_e equal to 0 we talk about complete wetting, while the case $0 < \theta_e < \pi$ is called partial wetting. If there are further forces acting on the droplet, a more dynamic behavior of the liquid-gas interface can be observed. Imagine a droplet running down the window. The shape of the droplet is no longer the same as for a droplet in equilibrium, it is tilted in some way. In direction of motion, the contact angle is greater than on the other side. We call them advancing and receding contact angles. The advancing contact angle is defined as the largest static contact angle before the three phase contact line starts to move, hence wetting the surface, while the receding angle is defined as the smallest static contact angle before the three phase contact line starts receding, hence dewetting the surface [50]. These two angles are also specific for certain materials and can be measured by experiments. Hence, the more realistic way of considering wetting or dewetting is to take a dynamic contact angle into account. A static contact angle works for static contact lines, but is non-physical when they move. In practice, more or less complicated empirical models have been developed, based on the observations, that the dynamic contact angle increases with increasing the contact line velocity and that the dynamic contact angle increases more rapidly for more viscous liquids [52]. Almost all models are based on the capillary number Ca , defined as

$$Ca = \frac{\mu_l u_{cl}}{\sigma_{lg}},$$

where μ_l is the dynamic viscosity of the liquid phase, u_{cl} the velocity of the contact line and σ_{lg} again the surface tension of the liquid-gas interface. A well recommended model is the Kistler model, which uses the following Hoffman equation to describe

the dynamic contact angle θ_d [52, 43]

$$\theta_d = f_H(Ca) = \arccos \left\{ 1 - 2 \tanh \left[5.16 \left(\frac{Ca}{1 + 1.31Ca^{0.99}} \right)^{0.706} \right] \right\}.$$

Here, f_H is the so-called Hoffman function. This relation applies only to complete wetting. For an equilibrium contact angle $\theta_e > 0$, the Hoffman function has to be shifted secondary as follows

$$\theta_d = f_H [Ca + f_H^{-1}(\theta_e)]. \quad (2.26)$$

To calculate the inverse of Hoffman's empirical function f_H^{-1} , Kistler's model uses an approximation by the Hoffman-Voinov-Tanner law, which is a simpler empirical model for the dynamic contact angle, but applies only to small contact angles. Therein, the dynamic contact angle is described as

$$\theta_d^3 - \theta_e^3 = c_\top Ca, \quad (2.27)$$

where $c_\top > 0$ is a constant depending on the material properties [100]. If we assume complete wetting, i.e., $\theta_e = 0$, and set the equilibrium contact angle θ_e as our dynamic contact angle θ_d in equation (2.27), we obtain for the inverse Hoffman function in (2.26) the following expression

$$f_H^{-1}(\theta_e) = \frac{\theta_e^3}{c_\top}.$$

The Kistler model holds for the whole range of wetting, i.e., for $\theta \in [0^\circ, 180^\circ]$, and for a wide range of Capillary numbers and contact line velocities. So this universal model is appropriate for our problems and will be used in the numerical studies. Other well known models are Shikhmurzaev's model, which is based on Shikhmurzaev's interface formation model, but requires further phenomenological constants coming from experimental data [85]. Or the Cox model, which establishes a connection with the physical slip length [19].

We append the dynamic contact angle from Kistler and Hoffmann to our model formulation as boundary condition. We have to note, that this additional boundary condition leads to a mathematically overdetermined problem which, however, works well in practice and is widely used. Within the numerical considerations of the problem, the dynamic contact angle is a parameter in the correction of the interface normal ν_Γ at the three phase contact point or line, see Section 4.4.1 for a description of the exact formula. The dynamic contact angle model works

together with the Navier-slip boundary condition from Section 2.5, which is needed to allow the fluid to move along the wetted surface [19]. Since there is no generally accepted dynamic contact angle model, there is also a wide range of applications with other approaches. An example is a generalization of the Navier-slip boundary condition, called generalized Navier boundary condition, proposed by QIAN et al. in [71]. This approach considers not only the tangential viscous stress, but also the uncompensated Young stress and therefore combines the microscopic with the macroscopic contact angle. No separate contact angle treatment is necessary when using the generalized Navier boundary condition. The dynamic contact angle is obtained from the model, which is a great advantage of this method. Recently, it is mainly used in several works together with diffuse interface models [103, 14].

If we come back to our example with the droplet skidding on a window, there is still another effect that can be observed. The droplet does not move continuously, it may stop from time to time. A so-called contact line hysteresis can be observed when it holds $\theta_a < \theta < \theta_r$, where θ_a is the advancing contact angle and θ_r the receding one. This phenomena can for example be treated with two separate dynamic contact angle models for receding and advancing contact lines, as described in [33]. We will not consider this effect since it brings a further complexity to the system.

Finally, we have to solve a multiphase flow problem with a sharp interface Γ between the different fluids and a contact point or line Σ between the fluids and a solid surface. A schematic graph of the considered scenario is shown in Figure 2.6.

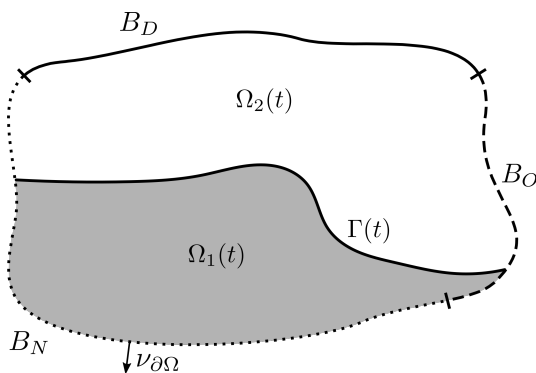


Figure 2.6. A two-phase domain with interface and contact points.

The outer boundary $\partial\Omega$ is split into a Dirichlet part B_D , a Navier-slip part B_N and a natural outflow part B_O , with $\partial\Omega = B_D \cup B_N \cup B_O$. Here, the Navier-slip part B_N also contains the dynamic contact line $\Sigma(t) = \partial\Gamma(t) := \Gamma(t) \cap \partial\Omega$, where the contact angle condition holds. Note, that the outer boundary does not change in time, but the interface Γ does. For a stable solution theory, we assume that $\partial\Omega$ is a Lipschitz boundary. Then the following boundary conditions can be added to the problems (2.12) and (2.21)

$$\begin{aligned}
 u &= 0 && \text{on } B_D, \\
 (\mu\nabla u - p\mathbf{I}) \cdot \nu_{\partial\Omega} &= 0 && \text{on } B_O(t), \\
 u^\top \nu_{\partial\Omega} &= 0 && \text{on } B_N(t), \\
 u_t - L(\mathbf{S}\nu_{\partial\Omega})_t &= 0 && \text{on } B_N(t), \\
 f_{\text{H}}[Ca + f_{\text{H}}^{-1}(\theta_e)] &= \theta_d && \text{on } \Sigma(t).
 \end{aligned}$$

Optimal Control of Two-phase Flow

In this chapter we introduce the optimal control problem with respect to the two-phase Navier-Stokes equations with surface tension. The general goal of optimal flow control problems is of course to optimize flow processes. They aim to achieve the best possible flow behavior regarding for example fluid velocity, vorticity or material parameters, as well as optimize geometrical aspects or the temperature. Optimal flow control can be achieved in several ways. For example, boundary control can be used to separate different fluids or fluid phases in a controlled manner, forcing the interface into a desired position [17]. Another optimal control problem in the context of two-phase flow is shape control to achieve an optimal domain design. Shaping the wing of an aircraft to create a suitable flow behavior such as drag reduction [79, 61] or forming a special tube to minimize power dissipation inside [65, 64] are examples for that.

Due to the complexity of our state equations, a solution of the optimal control problem and the resulting optimality system is difficult with a straight forward one-shot approach. A gradient-based optimization algorithm, schematically shown in Figure 3.1, is therefore used to calculate an optimal solution. The graph points out all iteration steps we will develop in the next sections and chapters. We will start with introducing a general optimal control problem in Section 3.1. The described controls and states refer to step one and two of the algorithm in Figure 3.1. The targeted optimization procedure is a sensitivity approach, which is introduced in Section 3.2 and provides sensitivity equations. In step four, the sensitivity equations have to be solved analogously to the primal state equations, which were solved in step three. The steps three and four are not covered in this chapter directly. Since,

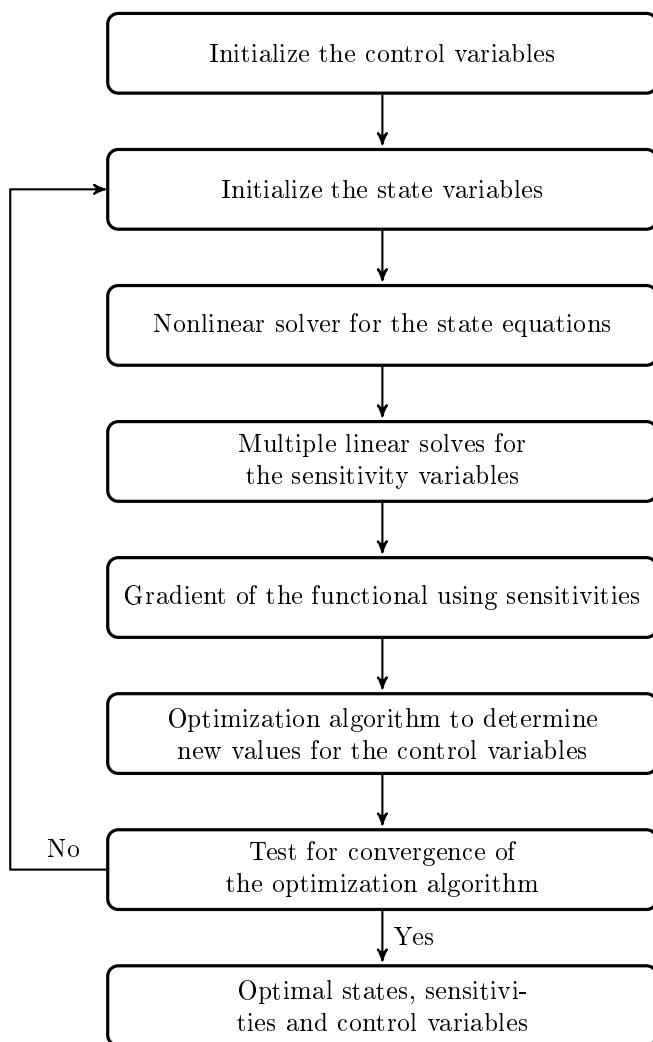


Figure 3.1. Schematic diagram of a gradient-based optimization algorithm [36].

there does not exist an analytical solution for the considered state equations, we have to resort to a numerical solution of the discrete two-phase Navier-Stokes system with surface tension and the sensitivity equations, discussed in detail in Chapter 4. We need the sensitivities as well to calculate the gradient of the objective functional, which belongs to step five. The problem formulation and sensitivity approach are mostly based on the first chapter of [41] and [36]. In Section 3.3, we continue with introducing the weak formulation of the problem and the concept of L^p -maximal

regularity, which we will use later in the section to show the differentiability of the related control-to-state mapping. These parts are based on results we developed in [25]. Thereby, we follow an approach which was introduced by PRÜSS and SIMONETT [70] to show well posedness of the underlying linear problem by means of L^p -maximal regularity. Afterwards, a sensitivity system is derived for the example of optimizing the flow with respect to the liquid viscosity, see Section 3.4. This sensitivity system is the basis for our own numerical sensitivity solver, we present in Section 4.4.2 and Appendix B.2. In the end of the current chapter, in Section 3.5, a summary of the so far developed gradient-based optimization algorithm is shown, including step six, seven and eight of Figure 3.1. Here, we will also discuss the different ways of solving the nonlinear and discrete optimization problem with their advantages and disadvantages.

3.1 Optimal Control Problem

In general, we have to deal with an optimal control problem, which has the form

$$\min_{y \in Y, q \in Q} j(y, q) \quad \text{subject to} \quad C(y, q) = 0, \quad q \in Q_{ad}. \quad (3.1)$$

Here, $j : Y \times Q \rightarrow \mathbb{R}$ is the objective functional, $y \in Y$ the state variable and $q \in Q$ the control variable, where Y and Q are Banach spaces. The state variable in the introduced two-phase flow problem consists of different physical quantities and varies depending on the representation we use. In the classical description based on PRÜSS and SIMONETT, the state is composed of the velocity and the pressure, i.e., $y = (u, p)$ with $y \in Y \subset L^2(I; L^2(\Omega)^n) \times L^2(I; L^2(\Omega))$, $I = [0, t]$. However, in the equivalent VOF representation the phase fraction is added to the state variable, i.e., $y = (u, p, \alpha)$ with $y \in Y \subset L^2(I; L^2(\Omega)^n) \times L^2(I; L^2(\Omega)) \times L^2(I; L^2(\Omega))$. We will use both cases in our theoretical investigations in Section 3.3. The exact spaces will also be defined there.

The control variable q could be a material parameter, for instance, the viscosity of the liquid phase. Besides material parameters, structural properties are also feasible control variables, such as the shape of the domain or parts of it. The nonempty closed set Q_{ad} of admissible controls contains further equality and/or inequality constraints for the control and provides e.g., lower and upper bounds of q . The treatment of state constraints would also be possible, but will not be considered in the following.

$C : Y \times Q \rightarrow Z$ is an operator that describes the behavior of the flow mathematically, where Z is also a Banach space. We denote $C(y, q) = 0$ as the state equation system of our optimization problem. The PDE system comprises all equations used for one of our wetting process representations, i.e., (2.12) or (2.21). We assume that C is continuously Fréchet differentiable, that the state equations $C(y, q) = 0$ have a unique solution $y(q) \in Y$ for each $q \in Q$ and that $C_y(y, q) \in \mathcal{L}(Y, Z)$ has a bounded inverse. Then, the well known implicit function theorem, see Theorem 2.8, can be applied to define a continuously differentiable control-to-state mapping [41] by

$$y : Q \rightarrow Y, \quad q \mapsto y(q) \quad \text{such that} \quad C(y(q), q) = 0.$$

We will prove the differentiability of our problem-specific control-to-state mapping in Section 3.3, since it is not that trivial in our case.

The objective function of our problem depends on the application and can vary. Reaching a desired state for one or more of the state variables, or minimizing the vorticity inside the flow, are just two practical examples. Inserting the above defined control-to-state map into the objective functional $j(y, q)$ results in the reduced objective functional $\bar{j}(y(q), q)$, which does not directly depend on the state anymore. If we additionally suppose, that $q \mapsto \bar{j}(y(q), q)$ is Fréchet differentiable, we obtain the following reduced optimization problem instead of (3.1)

$$\min_{q \in Q} \bar{j}(q) := j(y(q), q) \quad \text{s.t.} \quad q \in Q_{ad}. \quad (\text{P})$$

3.2 Sensitivity Approach

Following a gradient-based optimization approach, the derivative of the objective functional j , or more precisely of the reduced objective functional \bar{j} , is needed. The classical one-shot approach of setting up a Lagrange functional and solving the coupled optimality system, resulting from the first-order necessary optimality condition, at once is not practicable in our case. Instead, an iterative treatment is required, where we have two possibilities for the computation of the desired derivative. On the one hand, we can choose a sensitivity approach, where we compute the derivative of the reduced objective functional with respect to every control variable. Sensitivities are directional derivatives and indicate, how the state variables are effected by changes of the control variables. This is feasible as long as

3.2. Sensitivity Approach

the dimension of q is small, since the effort grows linearly in the dimension of Q . On the other hand we can apply an adjoint approach, where we derive the adjoint problem and solve one adjoint equation together with the primal problem. This approach is to the best advantage for high dimensional controls, since only a single set of equations has to be solved, independent of the dimension of q . However, this is at the expense of high implementation effort, since the adjoint systems must be solved backward in time for unsteady problems. The exemplary optimization problem we investigate in this work has a one dimensional control, so a sensitivity approach is reasonable and we have to solve the sensitivity system only once. The solution of optimization problems with higher dimensional controls will be presented at the appropriate place if it is necessary. In the following, we give a brief survey over the sensitivity approach from [41].

As we already mentioned, we utilize the reduced objective functional \bar{j} for our approach, which only depends on the control q

$$\bar{j}(q) := j(y(q), q).$$

The sensitivity, hence the directional derivative of the reduced objective function \bar{j} is obtained by applying the chain rule. For $q \in Q$ and a direction $s \in Q$ this yields

$$d\bar{j}(q, s) = j_y(y(q), q)y'(q)s + j_q(y(q), q)s. \quad (3.2)$$

j_y and j_q represent the partial derivatives of the objective functional with respect to the state and the control and $y'(q)$ is the derivative of the state y with respect to q . We call $\delta_s y := dy(q, s) = y'(q)s$ the state sensitivity in direction s .

Calculating the terms $j_y(y(q), q)$ and $j_q(y(q), q)$ is easily done in most of the cases, since the objective function is typically simple, for example linear or quadratic. This also applies in our case. The challenge is how to determine the derivative of the state y with respect to q and thus the sensitivities of our system. To obtain this gradient, we can differentiate the equations $C(y(q), q) = 0$ in direction s , also with the chain rule

$$C_y(y(q), q)\delta_s y + C_q(y(q), q)s = 0. \quad (3.3)$$

For a $q = (q_1, \dots, q_n)$, $n \in \mathbb{N}$, we have to solve this system for all $s \in B$, where B is a basis of Q . So the effort of calculating the whole operator $y'(q)$ grows linearly with the dimension of Q and equally also the effort to calculate the whole derivative of the objective function.

Following the sensitivity approach, we have seen that we not only need the derivative of the objective function, but also have to take into account differentiability results for the state equations. In our case, this applies to the wetting system $C(y, q)$, equipped with some control variables q . In order to place our sensitivity calculations on a solid theoretical foundation, we will have a closer look at the existence and uniqueness of differentiability results for the control-to-state mapping in the next section.

But first another comment to this approach. There is the possibility to calculate the derivatives with the help of a difference quotient approximation. That means, we calculate the state at a value $q + ts$ close to q for a small t and obtain

$$y'(q)s \approx \frac{y(q + ts) - y(q)}{t}. \quad (3.4)$$

The derivative of the state equations, which were calculated in this way, can again be used to calculate the derivative of the objective function. With this approach we get a good approximation for the sensitivities in most of the cases. Analogously we can apply this method directly to the objective function, without calculating the sensitivities separately. But then we always require an additional solution of the flow system for each control parameter. Several derivative-free algorithms go this way, but then they usually perform worse than derivative-based methods. In our case, the more precise derivative based approach is chosen and the difference quotient is merely used to test and verify the sensitivity calculations in Chapter 5 as well as for the optimization problems with domain transformations.

3.3 Differentiability of the Control-to-State Mapping

This section contains the important task of proving Fréchet differentiability of the solution operator involving the governing equations of two-phase flow. Since the respective PDEs are not limited to the application of wetting phenomena, we state that we derive differentiability results for a general two-phase problem, which is described with the PDE system given in (2.12), and for an equivalent VOF formulation of the problem, given in (2.21). Here we have to note, that the differentiability is shown only for the unbounded case and for special controls. Nevertheless, the PDEs have to be solved in an a priori unknown domain, where the moving boundary between the different fluids is part of the problem. In

general, those problems are more difficult to solve than in a prescribed domain, for which reason the solution approach is to transform the problem with the moving interface into a fixed domain [69]. More details are presented in the Sections 3.3.3 to 3.3.6, after we introduced the weak formulation of the problems and addressed the regularity assumptions, which we employ to show that the solution operator is Fréchet differentiable.

In the following, we investigate optimization problems with respect to an initial velocity field u_0 and a distributed control c on the right hand side of the momentum equation. Based on the problem formulation (2.12) and the definition (2.13), our so-called control-to-state operator reads

$$\begin{aligned}
 \rho(\partial_t u + u \cdot \nabla u) - \mu \Delta u + \nabla p &= c && \text{in } \Omega(t), \\
 \nabla \cdot u &= 0 && \text{in } \Omega(t), \\
 \llbracket u \rrbracket &= 0 && \text{on } \Gamma(t), \\
 -\llbracket S(u, p; \mu) \nu \rrbracket &= \sigma \kappa \nu && \text{on } \Gamma(t), \\
 V &= u^\top \nu && \text{on } \Gamma(t), \\
 u(0) &= u_0 && \text{on } \Omega(0), \\
 \Gamma(0) &= \Gamma_0.
 \end{aligned} \tag{3.5}$$

Remember, we have to solve this PDE system with respect to the velocity u and the pressure p , and denote the viscous stress tensor as $S(u, p; \mu) = -p\mathbf{I} + \mu(\nabla u + \nabla u^\top)$. System (3.5) is a free boundary problem without the contact angle problem. PRÜSS and SIMONETT state, that methods based on comparison principles, variational inequalities and viscosity solutions do not seem well-adapted in the presence of surface tension [69]. Additionally, the moving interface renders a variational analysis difficult. To show well-posedness of problem (2.12), where the surface tension plays a dominant role, they apply a different approach, the so-called direct mapping method, and follow the idea of maximal regularity. This means that the original problem is first transformed to a problem with fixed interface to establish the regularity of a solution. We follow the same approach to show the differentiability of our control-to-state mapping. This is done first for the transformed problem and then also in the physical coordinates by using the findings and performing similar steps as for showing well-posedness of the not differentiated problem. The approach is reflected in the following sections, where we start with presenting the approach of L^p -maximal regularity. Then we show the transformation to a flat interface, followed by the proof of well-posedness and differentiability of the transformed state. In the third step we use these results to transform back and verify the differentiability of the original problem, in consideration of the corresponding regularities. Furthermore, we derive

differentiability results for the volume of fluid type formulation in Section 3.3.6, based on the former results.

3.3.1 Weak Formulation

Since the pressure is generally discontinuous at the interface in the case of various phases, differentiability results in a strong sense are only valid outside the interface region. So we expect only results in the weak sense at the interface, and therefore the weak formulation of the problems is required. Note, that we will consider the problems in $n + 1$ dimensions, since we want our interface to have n dimensions, which is always one dimension lower than the full domain.

A weak form can be calculated by using test functions from suitable spaces. The governing equations are multiplied with the test functions and the product is integrated over the whole domain, applying partial integration if it is necessary. The weak form of problem (2.12) is then given as follows. For all $\varphi \in C_0^1(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})$ and all $\psi \in C_0^1(\mathbb{R}^{n+1})$ it holds

$$\int_{\mathbb{R}^{n+1}} (\partial_t(\rho u) + \nabla \cdot (\rho u \otimes u) - c)^\top \varphi + (S(u, p; \mu)) : \nabla \varphi \, dx = \int_{\Gamma(t)} \sigma \kappa \nu^\top \varphi \, ds, \quad (3.6)$$

$$\int_{\mathbb{R}^{n+1}} (\nabla \cdot u) \psi \, dx = 0. \quad (3.7)$$

Here, also the jump conditions are incorporated, so this formulation considers the first four equations from (2.12). Since the one-field formulation also brings in discontinuities of the phase-field variable, the weak form of the VOF representation is mandatory to show the differentiability of the respective control-to-state mapping and for the derivation of sensitivity equations in 3.3.6. Hence, if we express the phases by a phase indicator field α , we obtain for all $\varphi \in C_c^1(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})$ and all $\psi \in C_c^1(\mathbb{R}^{n+1})$ the formulation

$$\begin{aligned} & \int_{\mathbb{R}^{n+1}} (\partial_t(\rho(\alpha)u) + \operatorname{div}(\rho(\alpha)u \otimes u))(t, x, y)^\top \varphi(x, y) \\ & \quad + S(u, q; \mu(\alpha))(t, x, y) : \nabla \varphi(x, y) \, d(x, y) \\ & = - \lim_{\varepsilon \searrow 0} \int_{\mathbb{R}^{n+1}} \sigma \frac{\nu_\varepsilon(t, x, y)^\top}{|\nu_\varepsilon(t, x, y)|} (D\varphi - \operatorname{div}(\varphi)I)(x, y) \nabla \alpha(t, x, y) \, d(x, y), \end{aligned} \quad (3.8)$$

$$\int_{\mathbb{R}^{n+1}} \operatorname{div}(u) \psi \, dx = 0, \quad (3.9)$$

where we use the indicator function α from (2.14) as well as density and viscosity defined in (2.17), (2.18). Moreover, α fulfills the transport equation (2.19) and ν_ε is a suitable smoothed normal computed from $\nabla\alpha$, defined later in Section 3.3.6 with equation (3.65).

3.3.2 L^p -maximal Regularity

Maximal regularity is a powerful tool to prove existence and uniqueness of solutions to linear and nonlinear evolution equations [69]. It is the question of how to define function spaces $\mathbb{E}(I)$ and $\mathbb{F}(I)$, such that $\mathcal{L} : \mathbb{E}(I) \rightarrow \mathbb{F}(I) \times \mathbb{E}_\gamma$ is an isomorphism [69]. In this context, \mathbb{E}_γ denotes the time-trace space of $\mathbb{E}(I)$. The spaces are different for different problems, so we have to define them separately for every problem we study.

Let A be a linear operator with domain $D(A) \subset X$, where X is a Banach space. We introduce maximal L^p -regularity for the inhomogeneous initial value problem

$$\dot{u}(t) + Au(t) = f(t), \quad t \in I, \quad u(0) = u_0, \quad (3.10)$$

in $L^p(I; X)$, where $I = \mathbb{R}_+$ or $(0, a)$ for some $a > 0$, $f : I \rightarrow X$ and $1 < p < \infty$. Then the maximal L^p -regularity of problem (3.10) is as follows [69].

Definition 3.1. *Suppose the operator $A : D(A) \subset X \rightarrow X$ is closed and densely defined with $D(A)$ is the domain of A . Then, $A \in \mathcal{MR}^p(I; X)$ – and we say that there is maximal L^p -regularity for (3.10) – if for each $f \in L^p(I; X)$ there exists a unique solution $u \in H^{1,p}(I; X) \cap L^p(I; D(A))$ satisfying (3.10) a.e. in I , with $u_0 = 0$.*

Hence, with $\mathcal{MR}^p(I; X)$ we denote the class of all operators A that admit maximal L^p -regularity to the given problem. The corresponding setting of function spaces for our problem is presented at the appropriate place, see (3.19) and (3.22). Once we know the maximal regularity setting for A , in this case within the framework of L^p spaces, we can apply the contraction mapping principle to obtain local solutions and a generalized version of the implicit function theorem to show the smooth dependency of the local solutions on the data [69]. Therewith, we can prove Fréchet differentiability of the control-to-state mapping as defined in 2.6.

3.3.3 Transformation to a Flat Interface

For the further investigations, the initial interface Γ_0 is considered as graph of a sufficiently smooth function $h_0 : \mathbb{R}^n \rightarrow \mathbb{R}$, so it holds

$$\begin{aligned}\Gamma_0 &= \{(x, y) \in \mathbb{R}^n \times \mathbb{R} : y = h_0(x)\}, \\ \Omega_1(0) &= \{(x, y) \in \mathbb{R}^n \times \mathbb{R} : y < h_0(x)\}, \\ \Omega_2(0) &= \{(x, y) \in \mathbb{R}^n \times \mathbb{R} : y > h_0(x)\}.\end{aligned}$$

Then the interface for $t \in I$ can be expressed as

$$\Gamma(t) = \{(x, h(t, x)) : x \in \mathbb{R}^n\},$$

with $h : [0, t_0] \times \mathbb{R}^n \rightarrow \mathbb{R}$, the final time $t_0 > 0$ and $h(0, \cdot) = h_0$, see Figure 3.2.

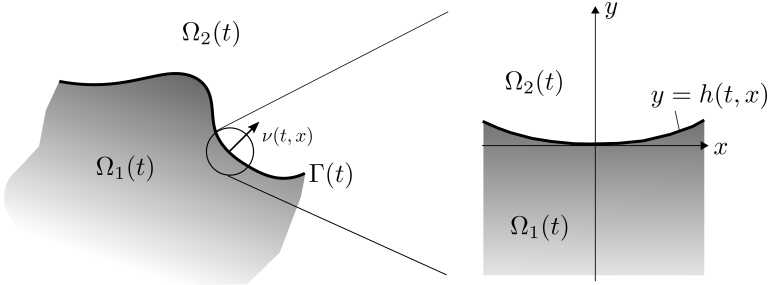


Figure 3.2. Interface parameterized with a function $h(t, x)$, inspired by [16].

Furthermore, we define $\dot{\mathbb{R}}^{n+1}$ and the halfspaces \mathbb{R}_\pm^{n+1} as

$$\begin{aligned}\dot{\mathbb{R}}^{n+1} &= \{(x, y) \in \mathbb{R}^n \times \mathbb{R} : y \neq 0\}, \\ \mathbb{R}_\pm^{n+1} &= \{(x, y) \in \mathbb{R}^n \times \mathbb{R} : \pm y > 0\}.\end{aligned}$$

Our goal is now to transform problem (3.5) to the halfspaces \mathbb{R}_\pm^{n+1} , which are not dependent on the time t anymore. Therefore, it is reasonable to reformulate the unit interface normal and interface curvature as well as the normal velocity of Γ using the height function. For the unit normal of the interface at the point $(x, h(t, x))$ we obtain

$$\hat{\nu}(t, x) = \nu(t, x, h(t, x)) = \frac{1}{\sqrt{1 + |\nabla h(t, x)|^2}} \begin{pmatrix} -\nabla h(t, x) \\ 1 \end{pmatrix}.$$

3.3. Differentiability of the Control-to-State Mapping

Here, $\nabla h(t, x)$ denotes the gradient vector of h with respect to $x \in \mathbb{R}^n$. Analogously, for the normal velocity V we derive

$$\begin{aligned}\hat{V}(t, x) &= \partial_t(x, h(t, x)) \nu(t, x, h(t, x)) \\ &= \frac{1}{\sqrt{1 + |\nabla h(t, x)|^2}} \begin{pmatrix} 0 \\ \partial_t h(t, x) \end{pmatrix}^\top \begin{pmatrix} -\nabla h(t, x) \\ 1 \end{pmatrix} \\ &= \frac{\partial_t h(t, x)}{\sqrt{1 + |\nabla h(t, x)|^2}}.\end{aligned}$$

If we assume that $h(t, \cdot)$ is two times differentiable, Δh denotes the Laplacian of h with respect to $x \in \mathbb{R}^n$ and $\nabla^2 h$ denotes the Hessian matrix of all second order derivatives of h , the curvature of $\Gamma(t)$ can be rewritten as

$$\hat{\kappa}(t, x) = \kappa(t, x, h(t, x)) = -\operatorname{div}_\Gamma \hat{\nu}(t, x) = \operatorname{div}_x \left(\frac{\nabla h(t, x)}{\sqrt{1 + |\nabla h(t, x)|^2}} \right) = \Delta h - G_k(h), \quad (3.11)$$

with

$$G_k(h) = \frac{|\nabla h|^2 \Delta h}{(1 + \sqrt{1 + |\nabla h|^2}) \sqrt{1 + |\nabla h|^2}} + \frac{\nabla h^\top \nabla^2 h \nabla h}{(1 + |\nabla h|^2)^{3/2}}.$$

A derivation of this expression can be found for example in [16, Appendix]. Now we also want to use the height function h to describe our state equations. With the following transformation, $\Gamma(t)$ becomes a flat interface at $y = 0$, where $t \in I$

$$\begin{aligned}\hat{u}(t, x, y) &= \begin{pmatrix} \hat{v}(t, x, y) \\ \hat{w}(t, x, y) \end{pmatrix}, \quad \text{with} \quad \hat{v}(t, x, y) = \begin{bmatrix} u_1(t, x, h(t, x) + y) \\ \vdots \\ u_n(t, x, h(t, x) + y) \end{bmatrix} \\ &\quad \text{and} \quad \hat{w}(t, x, y) = u_{n+1}(t, x, h(t, x) + y), \\ \hat{p}(t, x, y) &= p(t, x, h(t, x) + y).\end{aligned} \quad (3.12)$$

Analogously, we transform the density and viscosity of the domain by

$$\begin{aligned}\hat{\rho}(t, x, y) &= \rho(t, x, h(t, x) + y) = \chi_{\mathbb{R}_-^{n+1}}(x, y) \rho_1 + \chi_{\mathbb{R}_+^{n+1}}(x, y) \rho_2, \\ \hat{\mu}(t, x, y) &= \mu(t, x, h(t, x) + y) = \chi_{\mathbb{R}_-^{n+1}}(x, y) \mu_1 + \chi_{\mathbb{R}_+^{n+1}}(x, y) \mu_2.\end{aligned}$$

Remember, $(x, y) \in \dot{\mathbb{R}}^{n+1}$ with $x \in \mathbb{R}^n$ and $y \in \mathbb{R}$, $y \neq 0$. To derive the transformed state equations, we will first have a look at the required partial derivatives of the

transformed state variables. For $j, k = 1, \dots, n$ it holds [70]

$$\begin{aligned}
 \partial_j u_k &= \partial_j \hat{v}_k - \partial_j h \partial_y \hat{v}_k, & \partial_{n+1} u_k &= \partial_y \hat{v}_k, \\
 \partial_j u_{n+1} &= \partial_j \hat{w} - \partial_j h \partial_y \hat{w}, & \partial_{n+1} u_{n+1} &= \partial_y \hat{w}, \\
 \partial_j p &= \partial_j \hat{p} - \partial_j h \partial_y \hat{p}, & \partial_{n+1} p &= \partial_y \hat{p}, \\
 \partial_t u_k &= \partial_t \hat{v}_k - \partial_t h \partial_y \hat{v}_k, & \partial_t u_{n+1} &= \partial_t \hat{w} - \partial_t h \partial_y \hat{w}, \\
 \Delta u_k &= \Delta_x \hat{v}_k - 2(\nabla h \cdot \nabla_x) \partial_y \hat{v}_k + (1 + |\nabla h|^2) \partial_y^2 \hat{v}_k - \Delta h \partial_y \hat{v}_k, \\
 \Delta u_{n+1} &= \Delta_x \hat{w} - 2(\nabla h \cdot \nabla_x) \partial_y \hat{w} + (1 + |\nabla h|^2) \partial_y^2 \hat{w} - \Delta h \partial_y \hat{w}.
 \end{aligned}$$

Then, the transformed momentum equations, separated for \hat{v} and \hat{w} , result in

$$\begin{aligned}
 &\hat{\rho} \partial_t \hat{v} - \hat{\rho} \partial_t h \partial_y \hat{v} + \hat{\rho} ((\hat{v} \cdot \nabla_x) \hat{v} - (\nabla h^\top \hat{v}) \partial_y \hat{v} + \hat{w} \partial_y \hat{v}) \\
 &- \hat{\mu} (\Delta_x \hat{v} - 2(\nabla h \cdot \nabla_x) \partial_y \hat{v} + \partial_y^2 \hat{v} + |\nabla h|^2 \partial_y^2 \hat{v} - \Delta h \partial_y \hat{v}) + \nabla_x \hat{p} - \partial_y \hat{p} \nabla h = \hat{c}_x, \\
 &\hat{\rho} \partial_t \hat{w} - \hat{\rho} \partial_t h \partial_y \hat{w} + \hat{\rho} ((\hat{v} \cdot \nabla_x) \hat{w} - (\nabla h^\top \hat{v}) \partial_y \hat{w} + \hat{w} \partial_y \hat{w}) \\
 &- \hat{\mu} (\Delta_x \hat{w} - 2(\nabla h \cdot \nabla_x) \partial_y \hat{w} + \partial_y^2 \hat{w} + |\nabla h|^2 \partial_y^2 \hat{w} - \Delta h \partial_y \hat{w}) + \partial_y \hat{p} = \hat{c}_y,
 \end{aligned}$$

where $\hat{c} = (\hat{c}_x, \hat{c}_y)$ denotes the transformed control for the respective components. For the transformed version of the continuity equation we obtain

$$\operatorname{div}_x \hat{v} + \partial_y \hat{w} - \nabla h^\top \partial_y \hat{v} = 0.$$

Now we derive the transformed interface conditions, which are the counterparts to the jump conditions in (3.5). The first jump condition can be easily transformed in

$$[[\hat{u}]] = 0.$$

For the transformation of the second jump condition, we need the transformed version of the deformation tensor $D(u) = \nabla u + \nabla u^\top$, which is given by $\mathcal{D}(\hat{u}, h) = \mathcal{D}(\hat{v}, \hat{w}, h)$. Here it applies

$$\mathcal{D}(\hat{u}, h) = \nabla \hat{u} + \nabla \hat{u}^\top - \begin{pmatrix} \nabla h \partial_y \hat{u}^\top \\ 0 \end{pmatrix} - \begin{pmatrix} \nabla h \partial_y \hat{u}^\top \\ 0 \end{pmatrix}^\top. \quad (3.13)$$

Then we obtain for the individual components \hat{v} and \hat{w} the following transformed jump conditions, see [70]

$$\begin{aligned}
 &-\nabla h^\top [[\hat{p}]] + [[\hat{\mu}(\nabla_x \hat{v} + (\nabla_x \hat{v})^\top)]] \nabla h - |\nabla h|^2 [[\hat{\mu} \partial_y \hat{v}]] \\
 &-(\nabla h^\top [[\hat{\mu} \partial_y \hat{v}]] \nabla h + [[\hat{\mu} \partial_y \hat{w}]] \nabla h - [[\hat{\mu} \partial_y \hat{v}]] - [[\hat{\mu} \nabla_x \hat{w}]] = -\sigma(\Delta h - G_\kappa(h)) \nabla h,
 \end{aligned}$$

3.3. Differentiability of the Control-to-State Mapping

$$\llbracket \hat{p} \rrbracket - 2\llbracket \hat{\mu} \partial_y \hat{w} \rrbracket + \nabla h^\top \llbracket \hat{\mu} \partial_y \hat{v} \rrbracket + \nabla h^\top \llbracket \hat{\mu} \nabla_x \hat{w} \rrbracket - |\nabla h|^2 \llbracket \hat{\mu} \partial_y \hat{w} \rrbracket = \sigma(\Delta h - G_\kappa(h)).$$

Remark 3.2. With the transformed deformation tensor, equation (3.13), the compatibility condition introduced in Theorem 2.10 can equivalently be written as

$$\begin{aligned} \llbracket \hat{\mu} \mathcal{D}(\hat{u}_0, h_0) \hat{v}_0 - \hat{\mu}(\hat{v}_0^\top \mathcal{D}(\hat{u}_0, h_0) \hat{v}_0) \hat{v}_0 \rrbracket &= 0, \\ \operatorname{div} \hat{u}_0 &= \nabla h_0^\top \partial_y \hat{u}_0, \quad \llbracket \hat{u}_0 \rrbracket = 0, \end{aligned} \quad (3.14)$$

$$\text{where } \hat{v}_0 := \hat{v}(0, x) = \frac{1}{\sqrt{1+|\nabla h_0(x)|^2}} \begin{pmatrix} -\nabla h_0(x) \\ 1 \end{pmatrix}.$$

For the transformed version of the kinematic condition $V = (u^\top \nu)_{|\Gamma}$, we insert the transformed normal velocity \hat{V} and the transformed unit interface normal \hat{v} from above

$$\frac{\partial_t h}{\sqrt{1+|\nabla h|^2}} = \frac{1}{\sqrt{1+|\nabla h|^2}} \left((\hat{v}^\top, \hat{w}) \begin{pmatrix} -\nabla h \\ 1 \end{pmatrix} \right)_{|\Gamma}.$$

If we use the trace operator τ at the interface $y = 0$, providing $\tau \hat{v}(x) = \hat{v}(x, 0)$ and analogously $\tau \hat{w}(x) = \hat{w}(x, 0)$, we end up with the following equation

$$\partial_t h - \tau \hat{w} = -(\tau \hat{v})^\top \nabla h.$$

This transport equation describes the evolution of h [16].

All the resulting equations above can be written with the linear terms on the left hand side and the nonlinearities on the right hand side. For a clearer representation we then write the nonlinearities with

$$\begin{aligned} F_{\hat{v}}(\hat{v}, \hat{w}, \hat{p}, h) &= \hat{\mu} (-2(\nabla h \cdot \nabla_x) \partial_y \hat{v} + |\nabla h|^2 \partial_y^2 \hat{v} - \Delta h \partial_y \hat{v}) + \partial_y \hat{p} \nabla h \\ &\quad + \hat{\rho} (-\hat{v} \cdot \nabla_x) \hat{v} + (\nabla h^\top \hat{v}) \partial_y \hat{v} - \hat{w} \partial_y \hat{v}) + \hat{\rho} \partial_t h \partial_y \hat{v}, \\ F_{\hat{w}}(\hat{v}, \hat{w}, h) &= \hat{\mu} (-2(\nabla h \cdot \nabla_x) \partial_y \hat{w} + |\nabla h|^2 \partial_y^2 \hat{w} - \Delta h \partial_y \hat{w}) \\ &\quad + \hat{\rho} (-\hat{v} \cdot \nabla_x) \hat{w} + (\nabla h^\top \hat{v}) \partial_y \hat{w} - \hat{w} \partial_y \hat{w}) + \hat{\rho} \partial_t h \partial_y \hat{w}, \\ F_d(\hat{v}, h) &= \nabla h^\top \partial_y \hat{v}, \\ G_{\hat{v}}(\hat{v}, \hat{w}, \llbracket \hat{p} \rrbracket, h) &= -\llbracket \hat{\mu} (\nabla_x \hat{v} + (\nabla_x \hat{v})^\top) \rrbracket \nabla h + |\nabla h|^2 \llbracket \hat{\mu} \partial_y \hat{v} \rrbracket + (\nabla h^\top \llbracket \hat{\mu} \partial_y \hat{v} \rrbracket) \nabla h \\ &\quad - \llbracket \hat{\mu} \partial_y \hat{w} \rrbracket \nabla h + (\llbracket \hat{p} \rrbracket - \sigma(\Delta h - G_\kappa(h))) \nabla h, \\ G_{\hat{w}}(\hat{v}, \hat{w}, h) &= -\nabla h^\top \llbracket \hat{\mu} \partial_y \hat{v} \rrbracket - \nabla h^\top \llbracket \hat{\mu} \nabla_x \hat{w} \rrbracket + |\nabla h|^2 \llbracket \hat{\mu} \partial_y \hat{w} \rrbracket - \sigma G_\kappa(h), \\ H(\hat{v}, \hat{w}, h) &= -(\tau \hat{v})^\top \nabla h. \end{aligned} \quad (3.15)$$

Note that almost all terms are polynomials in $(\hat{v}, \hat{w}, \hat{p}, \llbracket \hat{p} \rrbracket, h)$ and in the derivatives of $(\hat{v}, \hat{w}, \hat{p}, h)$, with coefficients of first order, except the terms in $G_\kappa(h)$. Moreover, all terms are linear with respect to second derivatives and $G_\kappa(h)$ is the pointwise superposition of a smooth function with ∇h and $\nabla^2 h$ [25]. In summary, we obtain the following transformed version of problem (3.5) by

$$\begin{aligned}
 \hat{p}\partial_t\hat{u} - \hat{\mu}\Delta\hat{u} + \nabla\hat{p} &= \hat{c} + F(\hat{u}, \hat{p}, h) && \text{in } \dot{\mathbb{R}}^{n+1}, \\
 \nabla\hat{u} &= F_d(\hat{v}, h) && \text{in } \dot{\mathbb{R}}^{n+1}, \\
 -\llbracket \hat{\mu}\partial_y\hat{v} \rrbracket - \llbracket \hat{\mu}\nabla_x\hat{w} \rrbracket &= G_{\hat{v}}(\hat{v}, \hat{w}, \llbracket \hat{p} \rrbracket, h) && \text{on } \mathbb{R}^n, \\
 -2\llbracket \hat{\mu}\partial_y\hat{w} \rrbracket + \llbracket \hat{p} \rrbracket - \sigma\Delta h &= G_{\hat{w}}(\hat{v}, \hat{w}, h) && \text{on } \mathbb{R}^n, \\
 \llbracket \hat{u} \rrbracket &= 0 && \text{on } \mathbb{R}^n, \\
 \partial_t h - \tau\hat{w} &= H(\hat{v}, \hat{w}, h) && \text{on } \mathbb{R}^n, \\
 \hat{u}(0) &= \hat{u}_0, \quad h(0) = h_0
 \end{aligned} \tag{3.16}$$

for $t > 0$ and with $F(\hat{u}, \hat{p}, h) = (F_{\hat{v}}(\hat{v}, \hat{w}, \hat{p}, h), F_{\hat{w}}(\hat{v}, \hat{w}, h))$. This is a quasilinear system and can be shortly written as

$$\begin{aligned}
 L(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) &= (\hat{c} + F(\hat{u}, \hat{p}, h), F_d(\hat{u}, h), G_v(\hat{u}, \llbracket \hat{p} \rrbracket, h), G_w(\hat{u}, h), H(\hat{u}, h)), \\
 (\hat{u}(0), h(0)) &= (\hat{u}_0, h_0).
 \end{aligned} \tag{3.17}$$

For the next step we fix the nonlinear right hand sides of problem (3.16) and write the following linearized system instead of (3.17)

$$L(\hat{u}, \hat{p}, r, h) = (f, f_d, g_v, g_w, g_h), \quad (\hat{u}(0), h(0)) = (\hat{u}_0, h_0), \tag{3.18}$$

where we substitute $\llbracket \hat{p} \rrbracket = r$. This fix point formulation is also denoted as a Stokes problem with given inhomogeneities (f, f_d, g_v, g_w, g_h) and free boundary.

3.3.4 Well-posedness and Differentiability of the Transformed State

With the introduced transformation, our control-to-state operator or control-to-state mapping is now given as the mapping $(\hat{u}_0, \hat{c}) \in \mathbb{U}_{\hat{u}}(h_0) \times \mathbb{U}_{\hat{c}}(t_0) \mapsto (\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0)$, which maps the initial transformed controls \hat{u}_0 and \hat{c} to the state variables \hat{u}, \hat{p} and h . How the spaces $\mathbb{U}_{\hat{u}}(h_0)$ and $\mathbb{U}_{\hat{c}}(t_0)$ are defined will be presented soon. First, we recapitulate the function spaces incorporated in $\mathbb{E}(t_0)$, which were already introduced for the existence and uniqueness results of the two-phase flow description

by PRÜSS and SIMONETT in 2.3. It holds

$$\begin{aligned}
 \mathbb{E}_1(t_0) &:= \{\hat{u} \in H^{1,p}(I; L^p(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})) \cap L^p(I; H^{2,p}(\dot{\mathbb{R}}^{n+1}, \mathbb{R}^{n+1})) : \llbracket \hat{u} \rrbracket = 0\}, \\
 \mathbb{E}_2(t_0) &:= L^p(I; \dot{H}^{1,p}(\dot{\mathbb{R}}^{n+1})), \\
 \mathbb{E}_3(t_0) &:= W^{1/2-1/(2p),p}(I; L^p(\mathbb{R}^n)) \cap L^p(I; W^{1-1/p,p}(\mathbb{R}^n)), \\
 \mathbb{E}_4(t_0) &:= W^{2-1/(2p),p}(I; L^p(\mathbb{R}^n)) \cap H^{1,p}(I; W^{2-1/p,p}(\mathbb{R}^n)) \\
 &\quad \cap W^{1/2-1/(2p),p}(I; H^{2,p}(\mathbb{R}^n)) \cap L^p(I; W^{3-1/p,p}(\mathbb{R}^n)), \\
 \mathbb{E}(t_0) &:= \{(\hat{u}, \hat{p}, r, h) \in \mathbb{E}_1(t_0) \times \mathbb{E}_2(t_0) \times \mathbb{E}_3(t_0) \times \mathbb{E}_4(t_0) : \llbracket \hat{p} \rrbracket = r\}.
 \end{aligned} \tag{3.19}$$

The space $\mathbb{E}(t_0)$ is equipped with the natural norm

$$\|(\hat{u}, \hat{p}, r, h)\|_{\mathbb{E}(t_0)} = \|\hat{u}\|_{\mathbb{E}_1(t_0)} + \|\hat{p}\|_{\mathbb{E}_2(t_0)} + \|r\|_{\mathbb{E}_3(t_0)} + \|h\|_{\mathbb{E}_4(t_0)}.$$

Our aim is now to show differentiability of this control-to-state map, first for the case $\hat{c} = 0$, then also for $\hat{c} \neq 0$. Therefore, some underlying results have to be presented and modified if necessary, to apply an appropriate fixed point argument to (3.17). We start with the following theorem, which holds for $(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0)$ resulting from (3.17), with $I = (0, t_0)$ and for $c = 0$, i.e., also for $\hat{c} = 0$.

Theorem 3.3. *Let $p > n + 3$, $\hat{c} = 0$ and let*

$$\mathbb{U}_{\hat{u}} := W^{2-2/p,p}(\dot{\mathbb{R}}^{n+1}, \mathbb{R}^{n+1}), \quad \mathbb{U}_h := W^{3-2/p,p}(\mathbb{R}^n). \tag{3.20}$$

Then for any $t_0 > 0$ there exists $\hat{\varepsilon}_0 = \hat{\varepsilon}_0(t_0) > 0$ such that for all initial values

$$(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h$$

satisfying, with $u_0(x, h_0(x) + y) = \hat{u}_0(x, y)$, the compatibility conditions

$$\llbracket \mu D(u_0)\nu_0 - \mu(\nu_0^\top D(u_0)\nu_0)\nu_0 \rrbracket = 0, \quad \operatorname{div} u_0 = 0, \quad \llbracket u_0 \rrbracket = 0, \tag{3.21}$$

as well as the smallness condition

$$\|\hat{u}_0\|_{\mathbb{U}_{\hat{u}}} + \|h_0\|_{\mathbb{U}_h} \leq \hat{\varepsilon}_0$$

there exists a unique solution of the transformed problem (3.17) with

$$(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0).$$

Moreover, $(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0)$ depends continuously on $(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h$ satisfying the compatibility conditions (3.21).

Proof. The detailed proof can be found in Theorem 6.3 in [70]. \square

Another result, which is important for the fixed point argumentation, is the following L^p -maximal regularity result for the linearized problem (3.18), see [70].

Theorem 3.4. *Let $1 < p < \infty$ be fixed, $p \neq \{3/2, 3\}$ and assume that ρ_i, μ_i are positive constants. For arbitrary $t_0 > 0$, $I = (0, t_0)$, let $\mathbb{E}_1(t_0), \dots, \mathbb{E}_4(t_0)$ be defined by (3.19) and set $\mathbb{U}_{\hat{u}}, \mathbb{U}_h$ as in (3.20). Moreover, set*

$$\begin{aligned} \mathbb{F}_1(t_0) &= L^p(I; L^p(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})), \\ \mathbb{F}_2(t_0) &= H^{1,p}(I; \dot{H}^{-1,p}(\mathbb{R}^{n+1})) \cap L^p(I; H^{1,p}(\dot{\mathbb{R}}^{n+1})), \\ \mathbb{F}_3(t_0) &= W^{1/2-1/(2p),p}(I; L^p(\mathbb{R}^n, \mathbb{R}^{n+1})) \cap L^p(I; W^{1-1/p,p}(\mathbb{R}^n, \mathbb{R}^{n+1})), \\ \mathbb{F}_4(t_0) &= W^{1-1/(2p),p}(I; L^p(\mathbb{R}^n)) \cap L^p(I; W^{2-1/p,p}(\mathbb{R}^n)), \\ \mathbb{F}(t_0) &= \mathbb{F}_1(t_0) \times \mathbb{F}_2(t_0) \times \mathbb{F}_3(t_0) \times \mathbb{F}_4(t_0). \end{aligned} \quad (3.22)$$

Then, for all initial values $(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h$ and $(f, f_d, g, g_h) \in \mathbb{F}(t_0)$ with $g = (g_v, g_w)$ satisfying the following compatibility conditions

$$\operatorname{div} \hat{u}_0 = f_d(0) \quad \text{on } \dot{\mathbb{R}}^{n+1}, \quad [[\hat{u}_0]] = 0 \quad \text{on } \mathbb{R}^n \quad \text{if } p > 3/2, \quad (3.23)$$

$$[-\hat{\mu} \partial_y \hat{v}_0] - [[\hat{\mu} \nabla_x \hat{w}_0]] = g_v(0) \quad \text{on } \mathbb{R}^n \quad \text{if } p > 3, \quad (3.24)$$

there exists a unique solution $(\hat{u}, \hat{p}, [[\hat{p}]], h) \in \mathbb{E}(t_0)$ of (3.18) and the solution map

$$(f, f_d, g, g_h, \hat{u}_0, h_0) \in \mathbb{F}(t_0) \times \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \mapsto (\hat{u}, \hat{p}, [[\hat{p}]], h) \in \mathbb{E}(t_0)$$

is continuous.

Proof. This follows from Theorem 5.1 and Lemma 6.1, (e) in [70]. \square

At this point, we also introduce spaces with a left subscript 0, which indicate that the corresponding variables vanish on the boundary of the relevant domain. We denote these spaces by

$$\begin{aligned} {}_0\mathbb{E}(t_0) &:= \{(\hat{u}, \hat{p}, r, h) \in \mathbb{E}(t_0) : \hat{u}(0) = 0, r(0) = 0, h(0) = 0\}, \\ {}_0\mathbb{F}(t_0) &:= \{(f, f_d, g_v, g_w, g_h) \in \mathbb{F}(t_0) : f_d(0) = 0, g(0) = 0, g_h(0) = 0\}. \end{aligned}$$

Then, the following corollary is obtained immediately for homogeneous initial data:

Corollary 3.5. *Let $p > 3$ and choose $\mathbb{E}(t_0)$, $\mathbb{F}(t_0)$, ${}_0\mathbb{E}(t_0)$ and ${}_0\mathbb{F}(t_0)$ as above with initial value 0 for all components that admit a trace at $t = 0$. Then, problem (3.18)*

3.3. Differentiability of the Control-to-State Mapping

has a unique and continuous solution map

$$(f, f_d, g, g_h, 0, 0) \in {}_0\mathbb{F}(t_0) \times \mathbb{U}_{\hat{a}} \times \mathbb{U}_h \mapsto (\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in {}_0\mathbb{E}(t_0).$$

Moreover, the following properties of the right hand sides of problem (3.16), summarized in (3.15), are used for the fixed point argument:

Lemma 3.6. *Let $p > n + 3$ and set for $(\hat{u}, \hat{p}, r, h) \in \mathbb{E}(t_0)$*

$$N(\hat{u}, \hat{p}, r, h) := (F(\hat{u}, \hat{p}, h), F_d(\hat{u}, h), G(\hat{u}, r, h), H(\hat{u}, h)), \quad (3.25)$$

with $F = (F_{\hat{v}}, F_{\hat{w}})$, $G = (G_{\hat{v}}, G_{\hat{w}})$, F_d and H defined in (3.15). Then the mapping $N : \mathbb{E}(t_0) \rightarrow \mathbb{F}(t_0)$ is well defined and real analytic, more precisely,

$$N \in C^\omega(\mathbb{E}(t_0), \mathbb{F}(t_0)), \quad N(0) = 0, \quad DN(0) = 0.$$

Moreover,

$$DN(\hat{u}, \hat{p}, r, h) \in \mathcal{L}({}_0\mathbb{E}(t_0), {}_0\mathbb{F}(t_0)) \quad \forall (\hat{u}, \hat{p}, r, h) \in \mathbb{E}(t_0),$$

where DN denotes the Fréchet derivative of N and C^ω is as usual the space of real analytic functions.

Proof. See proposition 6.2 in [70]. □

To show the main differentiability result for the transformed problem, we need an analogue result for the spaces of the initial values:

Lemma 3.7. *Let $p > n + 3$, $\mathbb{U}_{\hat{a}}, \mathbb{U}_h$ as defined in (3.20) and set*

$$\mathbb{U}_{\hat{a},c} := \{\hat{u}_0 = (\hat{v}_0, \hat{w}_0) \in \mathbb{U}_{\hat{a}} : \llbracket \hat{u}_0 \rrbracket = 0\}.$$

Then, with $G = (G_{\hat{v}}, G_{\hat{w}})$ and H defined in (3.15), the mappings

$$(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{a}} \times \mathbb{U}_h \mapsto \hat{v}_0^\top \nabla h_0 \in W^{2-2/p,p}(\mathbb{R}^{n+1}), \quad (3.26)$$

$$(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{a},c} \times \mathbb{U}_h \mapsto H(\hat{v}_0, h_0) \in W^{2-3/p,p}(\mathbb{R}^n), \quad (3.27)$$

$$(\hat{u}_0, r_0, h_0) \in \mathbb{U}_{\hat{a}} \times W^{1-2/p,p}(\mathbb{R}^n) \times \mathbb{U}_h \mapsto G(\hat{u}_0, r_0, h_0) \in W^{1-2/p,p}(\mathbb{R}^n) \quad (3.28)$$

are real analytic and the first derivatives vanish in $(\hat{u}_0, r_0, h_0) = 0$.

Proof. We proved these statements in Lemma 6 in [25]. □

Furthermore, the following extension of Banach's fixed point theorem, based on Theorem 2.7, will be applied, which gives sufficient conditions for solvability.

Theorem 3.8. (a) *Let U, W, Z be real Banach spaces, let $A \in \mathcal{L}(Z, W)$ be an isomorphism and set $M := \|A^{-1}\|_{\mathcal{L}(W, Z)}$. Let $B_Z \subset Z$ be a nonempty closed convex set and $B_U \subset U$ be a nonempty set. Moreover, let $K : B_Z \times B_U \rightarrow W$ be Lipschitz continuous with*

$$\|K(z, u) - K(\tilde{z}, \tilde{u})\|_W \leq L_z \|z - \tilde{z}\|_Z + L_u \|u - \tilde{u}\|_U \quad \forall (z, u), (\tilde{z}, \tilde{u}) \in B_Z \times B_U$$

and assume that

$$A^{-1}K(z, u) \in B_Z \quad \forall (z, u) \in B_Z \times B_U \quad \text{and} \quad ML_z < 1. \quad (3.29)$$

Then for all $u \in B_U$ the equation

$$Az = K(z, u)$$

has a unique solution $z = z(u) \in B_Z$ and

$$\|z(u) - z(\tilde{u})\|_Z \leq \frac{L_u M}{1 - ML_z} \|u - \tilde{u}\|_U \quad \forall u, \tilde{u} \in B_U. \quad (3.30)$$

(b) *Assume in addition that B_U is a relatively open convex subset of $u^* + U_L \subset U$, where U_L is a closed linear subspace of U . Note, if $U_L = U$ is admitted, then $B_U \subset U$ is convex and open. Moreover, assume that $K : B_Z \times B_U \rightarrow W$ is Fréchet differentiable. Then $B_U \ni u \mapsto z(u) \in Z$ is Fréchet differentiable, where $\delta z_d := Dz(u)d$ is for any $d \in U_L$ the unique solution of the problem*

$$A\delta z_d = D_z K(z(u), u)\delta z_d + D_u K(z(u), u)d. \quad (3.31)$$

If $DK : B_Z \times B_U \rightarrow \mathcal{L}(Z \times U_L, W)$ is Lipschitz continuous, then also $Dz : B_U \rightarrow \mathcal{L}(U_L, Z)$ is Lipschitz continuous. If $K : B_Z \times B_U \rightarrow W$ is k -times Fréchet differentiable, then $B_U \ni u \mapsto z(u) \in Z$ is k -times Fréchet differentiable and if $D^k K$ is Lipschitz continuous on $B_Z \times B_U$, then $D^k z$ is Lipschitz continuous on B_U .

Proof. Again, the appropriate proof can be found in [25], Theorem 7. □

Applying this theorem to the linearized Stokes problem (3.18) and the quasilinear system (3.17), we end up with the following extension of Theorem 3.3. This is the main differentiability outcome for the transformed problem formulation, which we will prove with the results shown so far.

3.3. Differentiability of the Control-to-State Mapping

Theorem 3.9. *Let $p > n + 3$ and consider any $t_0 > 0$. Let $\mathbb{E}(t_0)$ and $\mathbb{F}(t_0)$ be defined as in (3.19) and (3.22) and set with $I = (0, t_0)$*

$$\begin{aligned} \mathbb{U}_{\hat{u}} &:= W^{2-2/p,p}(\dot{\mathbb{R}}^{n+1}, \mathbb{R}^{n+1}), & \mathbb{U}_h &:= W^{3-2/p,p}(\mathbb{R}^n), \\ \mathbb{U}_{\hat{c}}(t_0) &:= \mathbb{F}_1(t_0) = L^p(I; L^p(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})). \end{aligned} \quad (3.32)$$

Then, for any $t_0 > 0$ there exists $\hat{\varepsilon}_0 = \hat{\varepsilon}_0(t_0) > 0$ such that for all data

$$(\hat{u}_0, h_0, \hat{c}) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \times \mathbb{U}_{\hat{c}}(t_0)$$

satisfying the transformed compatibility condition (3.14) as well as the smallness condition

$$\|\hat{u}_0\|_{\mathbb{U}_{\hat{u}}} + \|h_0\|_{\mathbb{U}_h} + \|\hat{c}\|_{\mathbb{U}_{\hat{c}}(t_0)} < \hat{\varepsilon}_0, \quad (3.33)$$

there exists a unique solution of the transformed problem (3.17) with

$$(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0).$$

Moreover, the mapping

$$\{(\hat{u}_0, h_0, \hat{c}) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \times \mathbb{U}_{\hat{c}}(t_0) : (\hat{u}_0, h_0, \hat{c}) \text{ satisfy (3.14), (3.33)}\} \mapsto (\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0)$$

is continuous and infinitely many times differentiable with respect to (\hat{u}_0, \hat{c}) .

Proof. The idea is to extend the arguments in [70] and apply Theorem 3.8 to the transformed formulation (3.17). Let $z = (\hat{u}, \hat{p}, r, h) \in \mathbb{E}(t_0)$ and write (3.17) as

$$Lz = N(z) + (\hat{c}, 0), \quad (\hat{u}(0), h(0)) = (\hat{u}_0, h_0), \quad (3.34)$$

with N defined in (3.25). Let further (\hat{u}_0, h_0) satisfy (3.14) and (3.33), where $\hat{\varepsilon}_0$ will be adjusted later. Following [70], we first construct $z^* = z^*(\hat{u}_0, h_0) \in \mathbb{E}(t_0)$ that satisfies the equation

$$Lz^* = (0, f_d^*, g^*, g_h^*), \quad (\hat{u}^*(0), h^*(0)) = (\hat{u}_0, h_0), \quad (3.35)$$

where $(0, f_d^*, g^*, g_h^*) \in \mathbb{F}(t_0)$ resolves the compatibility conditions (3.23) and (3.24). Then we can write (3.34) equivalently as

$$L\tilde{z} = N(\tilde{z} + z^*(\hat{u}_0, h_0)) + (\hat{c}, 0) - Lz^*(\hat{u}_0, h_0) =: K(\tilde{z}; \hat{u}_0, h_0, \hat{c}), \quad \tilde{z} \in {}_0\mathbb{E}(t_0). \quad (3.36)$$

The construction of z^* can be accomplished as in [70]. Suppose that the initial

values (\hat{u}_0, h_0) satisfy the first compatibility condition in (3.14) and set

$$r_0(\hat{u}_0, h_0) = \llbracket \hat{p}_0 \rrbracket := \llbracket \hat{\mu}(\hat{v}_0^\top \mathcal{D}(\hat{u}_0, h_0) \hat{v}_0) \rrbracket + \sigma(\Delta h_0 - G_\kappa(h_0)).$$

The right hand side consists of several terms of $G(\hat{u}_0, 0, h_0)$ in (3.28) and thus Lemma 3.7 yields that the above mapping $(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \mapsto \llbracket \hat{p}_0 \rrbracket = r_0(\hat{u}_0, h_0) \in W^{1-2/p,p}(\mathbb{R}^n)$ is real analytic. Moreover, it is easy to check that the following compatibility conditions hold

$$\begin{aligned} -\llbracket \hat{\mu} \partial_y \hat{v}_0 \rrbracket - \llbracket \hat{\mu} \nabla_x \hat{w}_0 \rrbracket &= G_v(\hat{u}_0, \llbracket \hat{p}_0 \rrbracket, h_0) && \text{on } \mathbb{R}^n, \\ -2\llbracket \hat{\mu} \partial_y \hat{w}_0 \rrbracket + \llbracket \hat{p}_0 \rrbracket - \sigma \Delta h_0 &= G_w(\hat{u}_0, h_0) && \text{on } \mathbb{R}^n. \end{aligned} \quad (3.37)$$

Now let $D_n = -\Delta$ be the Laplacian in $L^p(\mathbb{R}^n)$ with domain $H^{2,p}(\mathbb{R}^n)$ and set

$$g^*(t) := e^{-tD_n} G(\hat{u}_0, r_0(\hat{u}_0, h_0), h_0), \quad g_h^*(t) := e^{-tD_n} H(\hat{u}_0, h_0).$$

By the real analyticity of $r_0(\hat{u}_0, h_0)$ and Lemma 3.7 the mappings

$$\begin{aligned} (\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h &\mapsto G(\hat{u}_0, r_0(\hat{u}_0, h_0), h_0) \in W^{1-2/p,p}(\mathbb{R}^n), \\ (\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h &\mapsto H(\hat{u}_0, h_0) \in W^{2-3/p,p}(\mathbb{R}^n) \end{aligned}$$

are real analytic. Now maximal L^p -regularity for D_n yields, see e.g. [27, Lem. 8.2]

$$\begin{aligned} g^* &\in H^{1,p}(I; W^{-1-1/p,p}(\mathbb{R}^n)) \cap L^p(I; W^{1-1/p,p}(\mathbb{R}^n)) \hookrightarrow \mathbb{F}_3(t_0), \\ g_h^* &\in H^{1,p}(I; W^{-1/p,p}(\mathbb{R}^n)) \cap L^p(I; W^{2-1/p,p}(\mathbb{R}^n)) \hookrightarrow \mathbb{F}_4(t_0), \end{aligned}$$

where the imbeddings follow by real interpolation and g^*, g_h^* are real analytic in $(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h$. (3.37) ensures that (3.24) holds for g^* . Next, let

$$c_d^*(t) = \begin{cases} \mathcal{R}_+ e^{-tD_{n+1}} \mathcal{E}_+ \hat{v}_0^\top \nabla h_0 & \text{in } \mathbb{R}_+^{n+1}, \\ \mathcal{R}_- e^{-tD_{n+1}} \mathcal{E}_- \hat{v}_0^\top \nabla h_0 & \text{in } \mathbb{R}_-^{n+1}, \end{cases}$$

where $\mathcal{E}_\pm \in \mathcal{L}(W^{2-2/p,p}(\mathbb{R}_\pm^{n+1}), W^{2-2/p,p}(\mathbb{R}^{n+1}))$ are extension operators and \mathcal{R}_\pm are the restrictions to \mathbb{R}_\pm^{n+1} . Now $(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \mapsto \hat{v}_0^\top \nabla h_0 \in W^{2-2/p,p}(\dot{\mathbb{R}}^{n+1})$ is by Lemma 3.7 real analytic. By L^p -regularity for D_{n+1} $c_d^* \in H^{1,p}(I; L^p(\mathbb{R}^{n+1})) \cap L^p(I; H^{2,p}(\dot{\mathbb{R}}^{n+1}))$ and thus

$$f_d^* := \partial_y c_d^* \in \mathbb{F}_2(t_0) \quad \text{with} \quad f_d^*(0) = F_d(\hat{v}_0, h_0)$$

is real analytic with respect to $(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h$. Therefore, also (3.23) holds for f_d^* and we conclude that $R^* := (0, f_d^*, g^*, g_h^*) \in \mathbb{F}(t_0)$ satisfies the compatibility

3.3. Differentiability of the Control-to-State Mapping

conditions (3.23), (3.24) and by construction $(\hat{u}_0, h_0) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \mapsto R^* \in \mathbb{F}(t_0)$ is real analytic. Hence, by Theorem 3.4 the linear problem (3.35) has a unique solution $z^* = z^*(\hat{u}_0, h_0)$ that is real analytic and by Lemma 3.7 the first derivative vanishes in 0, i.e., $Dz^*(0, 0) = 0$.

Now consider (3.36). By construction of z^* the right hand side of (3.36) is in ${}_0\mathbb{F}(t_0)$. Denote by $L_0 \in \mathcal{L}({}_0\mathbb{E}(t_0), {}_0\mathbb{F}(t_0))$ the restriction of L which is an isomorphism by Corollary 3.5. Hence, (3.36) can be written as

$$L_0 \tilde{z} = N(\tilde{z} + z^*(\hat{u}_0, h_0)) + (\hat{c}, 0) - Lz^*(\hat{u}_0, h_0) =: K(\tilde{z}; \hat{u}_0, h_0, \hat{c}), \quad \tilde{z} \in {}_0\mathbb{E}(t_0). \quad (3.38)$$

To apply Theorem 3.8 we set now with suitable $\hat{\varepsilon}_0 > 0$ and $\delta > 0$

$$\begin{aligned} B_U(\hat{\varepsilon}_0) &:= \{(\hat{u}_0, h_0, \hat{c}) \in \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \times \mathbb{U}_{\hat{c}}(t_0) : (\hat{u}_0, h_0, \hat{c}) \text{ satisfy (3.14), (3.33)}\}, \\ B_Z(\delta) &:= \{\tilde{z} \in {}_0\mathbb{E}(t_0) : \|\tilde{z}\|_{{}_0\mathbb{E}(t_0)} \leq \delta\}, \end{aligned}$$

where $\hat{\varepsilon}_0, \delta > 0$ will be adjusted later. By Lemma 3.6 and the properties of z^* we know, that the right hand side

$$(\tilde{z}, \hat{u}_0, h_0, \hat{c}) \in {}_0\mathbb{E}(t_0) \times \mathbb{U}_{\hat{u}} \times \mathbb{U}_h \times \mathbb{U}_{\hat{c}}(t_0) \mapsto K(\tilde{z}; \hat{u}_0, h_0, \hat{c}) \in \mathbb{F}(t_0) \quad (3.39)$$

is real analytic with $K(0) = 0$ and $D_{(\tilde{z}, \hat{u}_0, h_0)}K(0) = 0$. Hence, the Lipschitz constant L_z of K with respect to \tilde{z} is arbitrary small close to 0 and the Lipschitz constant of K with respect to $(\hat{u}_0, h_0, \hat{c})$ is $L_u = 2$ close enough to 0. Note, that the Lipschitz constant with respect to \hat{c} is 1. This implies, if we set $\delta = 4M\hat{\varepsilon}_0$ for $\hat{\varepsilon}_0$ small enough with $M = \|L_0^{-1}\|_{\mathcal{L}({}_0\mathbb{F}(t_0), {}_0\mathbb{E}(t_0))}$, that K has the Lipschitz constants $L_z = 1/(2M)$ and $L_u = 2$ on $B_Z(\delta) \times B_U(\hat{\varepsilon}_0)$. Hence, for all $(\tilde{z}, \hat{u}_0, h_0, \hat{c}) \in B_Z(\delta) \times B_U(\hat{\varepsilon}_0)$ it holds

$$\begin{aligned} &\|L_0^{-1}K(\tilde{z}; \hat{u}_0, h_0, \hat{c})\|_{{}_0\mathbb{E}(t_0)} \\ &\leq ML_z \|\tilde{z}\|_{{}_0\mathbb{E}(t_0)} + ML_u (\|\hat{u}_0\|_{\mathbb{U}_{\hat{u}}} + \|h_0\|_{\mathbb{U}_h} + \|\hat{c}\|_{\mathbb{U}_{\hat{c}}(t_0)}) < \frac{1}{2}\delta + 2M\hat{\varepsilon}_0 = \delta. \end{aligned} \quad (3.40)$$

This shows, that the mapping $L_0^{-1}K(\tilde{z}; \hat{u}_0, h_0, \hat{c})$ is a contraction for the given initial values. Thus, (3.29) is satisfied and (3.38) has by Theorem 3.8 for all $(\hat{u}_0, h_0, \hat{c}) \in B_U(\hat{\varepsilon}_0)$ a unique solution $\tilde{z} = \tilde{z}(\hat{u}_0, h_0, \hat{c}) \in B_Z(\delta)$ satisfying the Lipschitz stability (3.30). Since also the real analytic operator $z^*(\hat{u}_0, h_0) \in \mathbb{E}(t_0)$ is Lipschitz continuous on $B_U(\hat{\varepsilon}_0)$, the solution $z(\hat{u}_0, h_0, \hat{c}) = \tilde{z} + z^* \in \mathbb{E}(t_0)$ is unique and Lipschitz continuous on $B_U(\hat{\varepsilon}_0)$. Now let $(\hat{u}_0^*, h_0^*, \hat{c}^*) \in B_U(\hat{\varepsilon}_0)$ be arbitrary. Then $\{(\hat{u}_0, h_0^*, \hat{c}) \in B_U(\hat{\varepsilon}_0)\}$ is a relatively open subset of an affine subspace of $\mathbb{U}_{\hat{u}} \times \mathbb{U}_h \times \mathbb{U}_{\hat{c}}(t_0)$. Since (3.39) is real analytic, it follows from Theorem 3.8, b) that

$\tilde{z}(\hat{u}_0, h_0^*, \hat{c}) \in {}_0\mathbb{E}(t_0)$ is infinitely many times differentiable with respect to (\hat{u}_0, \hat{c}) and the same holds for $z(\hat{u}_0, h_0^*, \hat{c}) = \tilde{z} + z^* \in \mathbb{E}(t_0)$. \square

3.3.5 Differentiability of the Original Problem

Once we have shown differentiability for the transformed problem, we want to come back to the original formulation. Therefore, we transfer the results from Theorem 3.9 to problem (3.5). For a $h_0 \in \mathbb{U}_h$, the following function spaces are defined

$$\mathbb{U}_u(h_0) := W^{2-2/p,p}(\mathbb{R}^{n+1} \setminus \Gamma(0), \mathbb{R}^{n+1}), \quad \mathbb{U}_c(t_0) := L^p(I; H^{1,p}(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})). \quad (3.41)$$

Here, the initial interface is no flat surface at $y = 0$ anymore, so we write $\mathbb{R}^{n+1} \setminus \Gamma(0)$ instead of $\dot{\mathbb{R}}^{n+1}$ for h_0 in \mathbb{U}_u . Another point we utilize in our further investigations is, that only the gradient of the pressure occurs, so the transformed pressure $\hat{p}(t, \cdot)$ is only determined up to a constant. Due to this fact, we select from now on without restriction the unique representative satisfying

$$\hat{p} \in \mathbb{E}_1(t_0), \quad \int_{[-1,1]^n} \hat{p}(t, x, 0-) dx = 0 \quad \text{for a.a. } t \in (0, t_0). \quad (3.42)$$

Note, that the jump of \hat{p} , always denoted as $[[\hat{p}]]$, is uniquely determined in (3.17). With the convention from (3.42) and by the trace theorem, we find a Poincaré constant $C_P > 0$ with

$$\|\hat{p}(t, \cdot)\|_{H^{1,p}(\dot{\mathbb{R}}^{n+1})} \leq C_P(\|\hat{p}(t, \cdot)\|_{\dot{H}^{1,p}(\dot{\mathbb{R}}^{n+1})} + \|[[\hat{p}]]\|_{W^{1-1/p,p}(\mathbb{R}^n)}). \quad (3.43)$$

Now we define the back transformation to the original problem, see (3.12), and obtain the following preparatory result.

Theorem 3.10. *Let $(\hat{u}, \hat{p}, [[\hat{p}]], h) \in \mathbb{E}(t_0)$, $h_0 \in \mathbb{U}_h$, $u_0 \in \mathbb{U}_u(h_0)$, and consider*

$$\begin{aligned} u(t, x, y) &= \hat{u}(t, x, y - h(t, x)), \quad p(t, x, y) = \hat{p}(t, x, y - h(t, x)), \\ u_0(x, y) &= \hat{u}_0(x, y - h_0(x)). \end{aligned} \quad (3.44)$$

3.3. Differentiability of the Control-to-State Mapping

Then there exist constants $C(\|h\|_{\mathbb{E}_4(t_0)}) > 0$ and $C(\|h_0\|_{\mathbb{U}_h})$ such that

$$\begin{aligned} & \|u\|_{W^{1,p}(I \times \mathbb{R}^{n+1}, \mathbb{R}^{n+1})} + \left(\int_I \|u(t)\|_{H^{2,p}(\mathbb{R}^{n+1} \setminus \Gamma(t), \mathbb{R}^{n+1})}^p dt \right)^{1/p} \leq C(\|h\|_{\mathbb{E}_4(t_0)}) \|\hat{u}\|_{\mathbb{E}_1(t_0)}, \\ & \left(\int_I \|p(t)\|_{\dot{H}^{1,p}(\mathbb{R}^{n+1} \setminus \Gamma(t), \mathbb{R}^{n+1})}^p dt \right)^{1/p} \leq C(\|h\|_{\mathbb{E}_4(t_0)}) \|\hat{p}\|_{\mathbb{E}_2(t_0)}, \\ & \left(\int_I \|\llbracket p(t) \rrbracket\|_{W^{1-1/p,p}(\Gamma(t))}^p dt \right)^{1/p} \leq C(\|h\|_{\mathbb{E}_4(t_0)}) \|\llbracket \hat{p} \rrbracket\|_{L^p(I; W^{1-1/p,p}(\mathbb{R}^n))}, \\ & \|\hat{u}_0\|_{\mathbb{U}_{\hat{a}}} \leq C(\|h_0\|_{\mathbb{U}_h}) \|u_0\|_{\mathbb{U}_u(h_0)}. \end{aligned}$$

Proof. We showed this relations in [25] by using appropriate imbeddings. \square

With the described back transformation we can further formulate a first differentiability result for the mapping from the transformed variables to the original ones.

Lemma 3.11. *Consider the transformation (3.44), where we choose for $\hat{p} \in \mathbb{E}_2(t_0)$, $\llbracket \hat{p} \rrbracket \in \mathbb{E}_3(t_0)$ the unique representative \hat{p} satisfying (3.42). Then for all $\tilde{p} \in [p, \infty)$ the mapping*

$$(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0) \mapsto u \in C(\bar{I}; L^{\tilde{p}}(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})) \quad (3.45)$$

is continuously differentiable with derivative

$$(\delta \hat{u}, \delta \hat{p}, \llbracket \delta \hat{p} \rrbracket, \delta h) \in \mathbb{E}(t_0) \mapsto \delta u(t, x, y) = \delta \hat{u}(t, x, y - h(t, x)) - \partial_y \hat{u}(t, x, y - h(t, x)) \delta h(t, x).$$

Let $\mathcal{E}_{\pm} \in \mathcal{L}(H^{l,p}(\mathbb{R}_{\pm}^{n+1}), H^{l,p}(\mathbb{R}^{n+1}))$ be extension operators for $l = 1, 2$ and set

$$\begin{aligned} \hat{u}_{\pm}(t, \cdot) &= \mathcal{E}_{\pm} \hat{u}(t, \cdot), & u_{\pm}(t, x, y) &= \hat{u}_{\pm}(t, T_h(t)(x, y)), \\ \hat{p}_{\pm}(t, \cdot) &= \mathcal{E}_{\pm} \hat{p}(t, \cdot), & p_{\pm}(t, x, y) &= \hat{p}_{\pm}(t, T_h(t)(x, y)). \end{aligned} \quad (3.46)$$

Then the mappings

$$(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0) \mapsto u_{\pm} \in L^p(I; H^{1,p}(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})), \quad (3.47)$$

$$(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0) \mapsto p_{\pm} \in L^p(I; L^p(\mathbb{R}^{n+1})) \quad (3.48)$$

are continuously differentiable with derivative

$$(\delta \hat{u}, \delta \hat{p}, \llbracket \delta \hat{p} \rrbracket, \delta h) \in \mathbb{E}(t_0) \mapsto \begin{pmatrix} \delta u_{\pm} \\ \delta p_{\pm} \end{pmatrix}(t, x, y)$$

$$= \begin{pmatrix} \delta \hat{u}_{\pm} \\ \delta \hat{p}_{\pm} \end{pmatrix} (t, x, y - h(t, x)) - \partial_y \begin{pmatrix} \hat{u}_{\pm} \\ \hat{p}_{\pm} \end{pmatrix} (t, x, y - h(t, x)) \delta h(t, x).$$

Proof. The proof of this lemma is somewhat technical and can be found in [25]. Here, we also used appropriate imbeddings, provided by Lemma 9 within the cited paper, and the results from Theorem 3.10. \square

Before we come to the main result for the not transformed problem, there is a further Lemma we need to proof the differentiability of the original problem.

Lemma 3.12. *Let $\mathbb{U}_c(t_0) = L^p(I; H^{1,p}(\mathbb{R}^{n+1}))$. Then the mapping*

$$(c, h) \in \mathbb{U}_c(t_0) \times \mathbb{E}_4(t_0) \mapsto \hat{c}(c, h) \in \mathbb{U}_{\hat{c}}(t_0) \quad (3.49)$$

with $\hat{c}(c, h)(t, x, y) = c(t, x, y + h(t, x))$ is continuously differentiable with derivative

$$(\delta c, \delta h) \in \mathbb{U}_c(t_0) \times \mathbb{E}_4(t_0) \mapsto \delta c(t, x, y + h(t, x)) + \partial_y c(t, x, y + h(t, x)) \delta h(t, x).$$

Proof. Compare to the proof of Lemma 11 in [25]. \square

In conclusion we obtain the following existence and differentiability result for the original data (u_0, h_0, c) .

Theorem 3.13. *Let $p > n + 3$ and $\mathbb{U}_u(h_0), \mathbb{U}_c(t_0)$ be defined by (3.41). Then, for any $t_0 > 0$ there exists $\varepsilon_0 = \varepsilon_0(t_0) > 0$ such that for all data*

$$(h_0, c) \in \mathbb{U}_h \times \mathbb{U}_c(t_0), \quad u_0 \in \mathbb{U}_u(h_0)$$

satisfying the compatibility condition (3.21) as well as the smallness condition

$$\|u_0\|_{\mathbb{U}_u(h_0)} + \|h_0\|_{\mathbb{U}_h} + \|c\|_{\mathbb{U}_c(t_0)} < \varepsilon_0 \quad (3.50)$$

there exists a unique solution of the transformed problem (3.17) with

$$(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0).$$

Moreover, for any h_0 with $\|h_0\|_{\mathbb{U}_h} < \varepsilon_0$ the mapping

$$\{(u_0, c) \in \mathbb{U}_u(h_0) \times \mathbb{U}_c(t_0) : (u_0, h_0, c) \text{ satisfy (3.21), (3.50)}\} \mapsto (\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h) \in \mathbb{E}(t_0)$$

is continuously differentiable. By the chain rule in Lemma 3.11, also the original state (u, q) depends continuously differentiable on (u_0, c) with the spaces given in (3.45), (3.47), (3.48).

3.3. Differentiability of the Control-to-State Mapping

Proof. We adapt the fixed point argument in the proof of Theorem 3.9. Let

$$\hat{c}(c, h)(t, x, y) = c(t, x, y + h(t, x)). \quad (3.51)$$

The only difference compared to the situation in Theorem 3.9 results from the fact, that $\hat{c}(c, h)$ now depends additionally on h . Hence, the fixed point equation (3.38) changes to

$$L_0 \tilde{z} = K(\tilde{z}; \hat{u}_0, h_0, \hat{c}(c, \tilde{z} + z^*(\hat{u}_0, h_0))), \quad \tilde{z} \in {}_0\mathbb{E}(t_0). \quad (3.52)$$

Let $\hat{\varepsilon}_0 > 0$ be as in Theorem 3.9. Then we have

$$\|\hat{c}(c, h)\|_{\mathbb{U}_a} = \|c\|_{\mathbb{U}_a} \quad (3.53)$$

and the last estimate in Theorem 3.10 shows that for $\varepsilon_0 > 0$ small enough the smallness condition (3.50) implies (3.33). Note, (3.53) holds independently of h . Hence, for all (u_0, h_0, c) satisfying (3.50) we have $(\hat{u}_0, h_0, \hat{c}(c, h)) \in B_U(\hat{\varepsilon}_0)$ and thus by (3.40)

$$\|L_0^{-1}K(\tilde{z}; \hat{u}_0, h_0, \hat{c})\|_{{}_0\mathbb{E}(t_0)} < \delta.$$

Finally, the Lipschitz constant of $K(\tilde{z}; \hat{u}_0, h_0, \hat{c})$ with respect to \hat{c} is 1 and the mapping (3.49) with (3.51) is by Lemma 3.12 continuously differentiable. Moreover, the Lipschitz constant with respect to h is bounded by $\|c\|_{\mathbb{U}_c(t_0)} < \varepsilon_0$. Hence, for $\varepsilon_0 > 0$ small enough, (3.52) is a contraction and the existence, uniqueness and continuous differentiability follow as in the proof of Theorem 3.9. Lemma 3.11 and the chain rule yield now the continuous differentiability of the original state (u, q) with respect to (u_0, c) for the spaces given in (3.45), (3.47) and (3.48). \square

3.3.6 Results for the Volume of Fluid-type Formulation

With the differentiability result we obtained for the classical formulation of a two-phase problem, we are now able to show also the differentiability of the volume of fluid-type formulation of the problem, see equation (2.21). Again, we do not consider an outer boundary of our domain, so we do not need to take the boundary conditions into account. Only the extended state equations and initial conditions are considered. Since the phase indicator α is in general discontinuous at the interface, the results are expected in the weak topology of measures and the sensitivity of α is a measure concentrated along the interface. We will close the section with a sensitivity system for the VOF formulation, which invokes measure-valued solutions of the linearized transport equation.

Based on a solution (u, p) , which fulfills (2.12), and the corresponding sensitivities $(\delta u, \delta p)$, we will now show the differentiability of the volume of fluid-type formulation of the control-to-state mapping. Therefore, let the phase indicator $\alpha : \mathbb{R}^{n+1} \rightarrow [0, 1]$ be a distributional solution of the α -transport equation (2.19) and with an appropriate initial condition, so it holds

$$\partial_t \alpha + \nabla \cdot (u \alpha) = 0 \quad \text{in } I \times \mathbb{R}^{n+1}, \quad \alpha(0) = 1_{\Omega_1(0)} \quad \text{on } \mathbb{R}^{n+1}. \quad (3.54)$$

Then we can define uniquely a continuous mapping $(x, y) \mapsto X(t; x, y)$, where $X(t; x, y)$ satisfies the characteristic equation

$$\partial_t X(t; x, y) = u(t, X(t; x, y)), \quad t \in I, \quad X(0; x, y) = (x, y). \quad (3.55)$$

In order to deal with the sensitivity equation of (3.54), it will be beneficial to consider measure-valued solutions of the general equation

$$\partial_t \delta \alpha + \nabla \cdot (u \delta \alpha) = b \quad \text{in } I \times \mathbb{R}^{n+1}, \quad \delta \alpha(0) = \delta \alpha_0 \quad \text{on } \mathbb{R}^{n+1}. \quad (3.56)$$

In the following, we denote by $\mathcal{M}_{loc}(\mathbb{R}^{n+1})$ the space of locally bounded Radon measures. Further we define $\mathcal{M}_{loc}(\mathbb{R}^{n+1}) - \text{weak}^*$ as the space of all locally bounded Radon measures with weak convergence. Then we obtain

Proposition 3.14. *Let $u \in L^1(I; W^{1,\infty}(\mathbb{R}^{n+1}; \mathbb{R}^{n+1}))$. Then, for any $\delta \alpha_0 \in \mathcal{M}_{loc}(\mathbb{R}^{n+1})$, there exists a unique distributional solution of (3.56) in $C(\bar{I}; \mathcal{M}_{loc}(\mathbb{R}^{n+1}) - \text{weak}^*)$, given by*

$$\delta \alpha(t) = X(t)(\delta \alpha_0) + \int_0^t X(t-s)(b(s)). \quad (3.57)$$

Here, X is the forward flow defined by (3.55) and $\delta \alpha_t = X(t)(\delta \alpha_0)$ is the measure satisfying

$$\int_{\mathbb{R}^{n+1}} \phi(x, y) d\delta \alpha_t(x, y) = \int_{\mathbb{R}^{n+1}} \phi(X(t; x, y)) d\delta \alpha_0(x, y) \quad \forall \phi \in C_c(\mathbb{R}^{n+1}).$$

Proof. See [25] and [67] for the detailed proof. □

Therewith, we can show the following important proposition, which states the differentiability of the control-to-state map for the state variable α and gives a formulation for the sensitivity equation of the α -transport equation.

3.3. Differentiability of the Control-to-State Mapping

Proposition 3.15. *If $\hat{u} \in \mathbb{E}_1(t_0)$, $[\hat{u}] = 0$ and u is given by (3.44), then (3.54) has a unique solution given by*

$$\alpha(t, X(t; x, y)) = 1_{\Omega_1(0)}(x, y) \quad (3.58)$$

and thus $\alpha(t, \cdot) = 1_{\Omega_1(t)}$. Moreover, for ε_0 from Theorem 3.13 and any h_0 with $\|h_0\|_{\mathbb{U}_h} < \varepsilon_0$ the mapping

$$\begin{aligned} \{(u_0, c) \in \mathbb{U}_u(h_0) \times \mathbb{U}_c(t_0) : (u_0, h_0, c) \text{ satisfy (3.21), (3.50)}\} \\ \mapsto \alpha \in C(\bar{I}; \mathcal{M}_{loc}(\mathbb{R}^{n+1}) - weak^*) \end{aligned}$$

is continuously differentiable. The derivative

$$(\delta u_0, \delta c) \in \mathbb{U}_u(h_0) \times \mathbb{U}_c(t_0) \mapsto \delta \alpha \in C(\bar{I}; \mathcal{M}_{loc}(\mathbb{R}^{n+1}) - weak^*)$$

is given by the unique measure-valued solution of

$$\partial_t \delta \alpha + \nabla \cdot (u \delta \alpha) = -\nabla \cdot (\delta u \alpha) \quad \text{in } I \times \mathbb{R}^{n+1}, \quad \delta \alpha(0) = 0 \quad \text{on } \mathbb{R}^{n+1}. \quad (3.59)$$

Finally, $\delta \alpha$ satisfies

$$\int_{\mathbb{R}^{n+1}} \phi(x, y) d\delta \alpha(t)(x, y) = \int_{\mathbb{R}^n} \phi(x, h(t, x)) \delta h(t, x) dx. \quad (3.60)$$

Proof. If $\hat{u} \in \mathbb{E}_1(t_0)$, $[\hat{u}] = 0$ and u is given by (3.44), then it holds that $u \in C(\bar{I}; W^{1,\infty}(\mathbb{R}^{n+1}; \mathbb{R}^{n+1}))$ by the following imbeddings, see [25, Lem. 9]

$$\mathbb{E}_1(t_0) \hookrightarrow C(\bar{I}; BUC^1(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})) \cap C(\bar{I}; BUC(\mathbb{R}^{n+1}, \mathbb{R}^{n+1})), \quad (3.61)$$

$$\mathbb{E}_4(t_0) \hookrightarrow C^1(\bar{I}; BC^1(\mathbb{R}^n)) \cap C(\bar{I}; BC^2(\mathbb{R}^n)). \quad (3.62)$$

Following [4, Prop. 2.2] and [26, Cor. II.1], it is well known that (3.58) provides the unique weak solution in $L^1_{loc}(I \times \mathbb{R}^{n+1})$ of the following problem

$$\partial_t \alpha + u \cdot \nabla \alpha = 0 \quad \text{in } I \times \mathbb{R}^{n+1}, \quad \alpha(0) = 1_{\Omega_1(0)} \quad \text{on } \mathbb{R}^{n+1}.$$

Since $\nabla \cdot (u) = 0$ almost everywhere, it is also a distributional solution of (3.54), which is unique by Proposition 3.14.

Let now $(u_0, h_0, c), (\delta u_0, 0, \delta c) \in \mathbb{U}_u \times \mathbb{U}_h \times \mathbb{U}_c(h_0)$ be such that (u_0, h_0, c) and $(u_0, h_0, c) + (\delta u_0, 0, \delta c)$ satisfy the conditions of Theorem 3.13. Denote by $(\hat{u}, \hat{p}, [\hat{p}], h)$ the unique solution of (3.17) for data (u_0, h_0, c) and by $(\hat{u}^s, \hat{p}^s, [\hat{p}^s], h^s)$ the one for

data $(u_0, h_0, c) + s(\delta u_0, 0, \delta c)$. Let (u, p) and (u^s, p^s) be the corresponding states in physical coordinates according to (3.12) and let $\alpha = 1_{\Omega_1(t)}$, $\alpha^s = 1_{\Omega_1^s(t)}$ be the corresponding solutions of (3.54). Finally, let $(\delta u, \delta h, \delta p)$ be the directional derivatives, hence sensitivities, in direction $(\delta u_0, 0, \delta c)$, which exist by Theorem 3.13. We show that

$$\frac{\alpha^s - \alpha}{s} \rightarrow \delta\alpha \quad \text{in } C(\bar{I}; \mathcal{M}_{loc}(\mathbb{R}^{n+1}) - \text{weak}^*) \text{ as } s \rightarrow 0, \quad (3.63)$$

where $\delta\alpha$ solves (3.59). Let $\phi \in C_c(\mathbb{R}^{n+1})$ be arbitrary. Then

$$\begin{aligned} \int_{\mathbb{R}^{n+1}} \frac{\alpha^s - \alpha}{s}(t, x, y) \phi(x, y) d(x, y) &= \int_{\mathbb{R}^n} \int_{h(t, x)}^{h^s(t, x)} \frac{1}{s} \phi(x, y) d(x, y) \\ &\rightarrow \int_{\mathbb{R}^n} \phi(x, h(t, x)) \delta h(t, x) dx \end{aligned}$$

as $s \rightarrow 0$ uniformly in $t \in \bar{I}$, where we have used the differentiability result of Theorem 3.13. Moreover, it is obvious that the middle term is continuous with respect to t . Hence, (3.63) is proven and we have only to show that $\delta\alpha$ solves (3.59). To this end, let $\varphi \in C_c^1(I \times \mathbb{R}^{n+1})$ be arbitrary. Since α, α^s are distributional solutions of (3.54), we have

$$\begin{aligned} 0 &= \int_I \int_{\mathbb{R}^{n+1}} -(\partial_t \varphi + (u \cdot \nabla) \varphi) \frac{\alpha^s - \alpha}{s} + \alpha^s \left(\frac{u^s - u}{s} \cdot \nabla \right) \varphi(t, x, y) d(x, y) dt \\ &\rightarrow \int_I \int_{\mathbb{R}^{n+1}} -(\partial_t \varphi + (u \cdot \nabla) \varphi) \delta\alpha + \alpha(\delta u \cdot \nabla) \varphi(t, x, y) d(x, y) dt \end{aligned}$$

as $s \rightarrow 0$. For the limit transition, we have used $u \in C(\bar{I}; W^{1, \infty}(\mathbb{R}^{n+1}))$, (3.63) and that by Theorem 3.13 $\alpha^s = 1_{\Omega^s(t)} \rightarrow \alpha = 1_{\Omega(t)}$ in $L_{loc}^2(I \times \mathbb{R}^{n+1})$ and $\frac{u^s - u}{s} \rightarrow \delta u$ in $C(\bar{I}; L^p(\mathbb{R}^{n+1}))$. Hence, $\delta\alpha$ is a distributional solution of (3.59), which is unique by Proposition 3.14. \square

What we have shown so far holds only for the α -transport equation and a solution (u, p) of the classical formulation (2.12). Now we also want to take the full VOF-type formulation into account, where the next step is to express the surface tension term by using the phase indicator α such that its sensitivities can be expressed by using the measure $\delta\alpha$. Therefore, we rewrite the surface tension term within the weak formulation (3.6).

3.3. Differentiability of the Control-to-State Mapping

Lemma 3.16. *Let $\varphi \in C_c^1(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})$. Then, one has the following identity with the curvature $\kappa(t)$ of $\Gamma(t)$ according to (3.11)*

$$\begin{aligned}
 & \int_{\Gamma(t)} (\sigma \kappa \nu)(t, x, y)^\top \varphi(x, y) dS(x, y) \\
 &= \int_{\mathbb{R}^n} \sigma \operatorname{div}_x \left(\frac{\nabla h(t, x)}{\sqrt{1 + |\nabla h(t, x)|^2}} \right) \begin{pmatrix} -\nabla h(t, x) \\ 1 \end{pmatrix}^\top \varphi(x, h(t, x)) dx \\
 &= \int_{\mathbb{R}^n} \sigma \frac{(\nabla h(t, x)^\top, -1)}{\sqrt{1 + |\nabla h(t, x)|^2}} (D\varphi(x, h(t, x)) - \operatorname{div}(\varphi)(x, h(t, x))\mathbf{I}) \begin{pmatrix} \nabla h(t, x) \\ -1 \end{pmatrix} dx.
 \end{aligned} \tag{3.64}$$

Proof. The result follows from the definition of κ and integration by parts [25]. \square

To compute the interface normal from $\nabla\alpha$, we use the definition of distributional derivatives and obtain

Lemma 3.17. *Let $\psi \in C_c^1(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})$. Then it holds*

$$\begin{aligned}
 - \int_{\mathbb{R}^{n+1}} \psi(x, y)^\top \nabla \alpha(t, x, y) d(x, y) &= \int_{\Gamma(t)} \psi(x, y)^\top \nu(t, x, y) dS(x, y) \\
 &= \int_{\mathbb{R}^n} \psi(x, h(t, x))^\top \begin{pmatrix} -\nabla h(t, x) \\ 1 \end{pmatrix} dx.
 \end{aligned}$$

Proof. See [25] for the necessary equation transformations. \square

For the further considerations we now assume a $\delta \in (0, 1/2)$ and

$$\psi_\delta \in C_c^1((-1, 1)), \quad \psi_\delta|_{[-1+\delta, 1-\delta]} \equiv 1, \quad \psi_\delta(-s) = \psi_\delta(s) \quad \forall s \in \mathbb{R}, \quad \int_{\mathbb{R}} \psi_\delta(s) ds = 1.$$

and set

$$\phi_\varepsilon(x, y) = \frac{1}{\varepsilon^n} \psi_\delta(y/\varepsilon) \prod_{i=1}^n \psi_\delta(x_i/\varepsilon).$$

We use this definitions to define a smoothed normal ν_ε computed from $\nabla\alpha$, which

is not necessarily of unit length

$$\nu_\varepsilon(t, x, y) := - \int_{\mathbb{R}^{n+1}} \phi_\varepsilon((\tilde{x}, \tilde{y}) - (x, y)) \nabla \alpha(t, \tilde{x}, \tilde{y}) d(\tilde{x}, \tilde{y}). \quad (3.65)$$

Then we obtain by Lemma 3.17

$$\begin{aligned} \nu_\varepsilon(t, x, y) &= \int_{\Gamma(t)} \phi_\varepsilon((\tilde{x}, \tilde{y}) - (x, y)) \nu(t, \tilde{x}, \tilde{y}) dS(\tilde{x}, \tilde{y}) \\ &= \int_{\mathbb{R}^n} \phi_\varepsilon((\tilde{x}, h(t, \tilde{x})) - (x, y)) \begin{pmatrix} -\nabla h(t, \tilde{x}) \\ 1 \end{pmatrix} d\tilde{x}. \end{aligned}$$

Let us further assume that

$$|\nabla h| \leq 1 - \delta \quad \text{on} \quad x + [-\varepsilon, \varepsilon]^n. \quad (3.66)$$

Then we have by the definition of ϕ_ε

$$\nu_\varepsilon(t, x, h(t, x)) = \frac{1}{\varepsilon^n} \int_{\mathbb{R}^n} \prod_{i=1}^n \psi_\delta((\tilde{x}_i - x_i)/\varepsilon) \begin{pmatrix} -\nabla h(t, \tilde{x}) \\ 1 \end{pmatrix} d\tilde{x}. \quad (3.67)$$

The variation of ν_ε is

$$\delta \nu_\varepsilon(t, x, y) := - \int_{\mathbb{R}^{n+1}} \phi_\varepsilon((\tilde{x}, \tilde{y}) - (x, y)) \nabla d\delta \alpha(t)(\tilde{x}, \tilde{y}), \quad (3.68)$$

with the measure-valued solution of (3.59).

Two further lemmas are helpful to show the equivalence between the classical and the VOF-type formulation. The corresponding proofs can be both found in [25].

Lemma 3.18. *Let (3.66) hold. If $h \in C(\bar{I}; BC^2(\mathbb{R}^n))$, then there is a $C > 0$ such that*

$$|\nu_\varepsilon(t, x, h(t, x)) - (-\nabla h(t, x), 1)^\top| \leq C\varepsilon \quad \forall (t, x) \in I \times \mathbb{R}^n.$$

On compact subsets the error is $o(\varepsilon)$.

Lemma 3.19. *Let (3.66) hold. If $\delta h \in C(\bar{I}; BC^2(\mathbb{R}^n))$, then there is a $C > 0$ such that*

$$|\delta \nu_\varepsilon(t, x, h(t, x)) - (-\nabla \delta h(t, x), 1)^\top| \leq C\varepsilon \quad \forall (t, x) \in I \times \mathbb{R}^n.$$

3.3. Differentiability of the Control-to-State Mapping

On compact subsets the error is $o(\varepsilon)$.

Now we can show, that the unique solution (u, p) of (2.12) satisfies the VOF-type formulation, which consists of the α -transport equation (3.54), the weak formulation (3.8) and (3.9) as well as the definitions (2.17) and (2.18).

Theorem 3.20. *If (3.66) holds for the solution (u, p) of (2.12) according to Theorem 3.13, which is satisfied for $\varepsilon_0 > 0$ small enough, then it satisfies the VOF-type formulation (3.8), (3.9) and (3.54).*

Let vice versa (u, p, α) be a solution of the VOF-type formulation (3.8), (3.9) and (3.54), where $\alpha(t)$ is the indicator function of a domain $\Omega_1(t) = \{(x, y) \in \mathbb{R}^n \times \mathbb{R} : y = h(t, x)\}$. If (u, p, h) has the regularity as in Theorem 3.13, then (u, p, h) coincides with the solution of (2.12) according to Theorem 3.13.

Proof. Let (u, p) be the solution of (2.12) according to Theorem 3.13. Then it solves clearly also the weak formulation (3.6) and (3.7). Since the solution of (3.54) is $\alpha = 1_{\Omega_1}(t)$ by Proposition 3.15, the formulations (3.8), (3.9), (3.54) and (3.6), (3.7) are equivalent if the right hand side of (3.8) coincides with the surface tension force term (3.64). To show this, we note that Lemma 3.17 yields for any $\varepsilon > 0$

$$\begin{aligned} & - \int_{\mathbb{R}^{n+1}} \sigma \frac{\nu_\varepsilon(t, x, y)^\top}{|\nu_\varepsilon(t, x, y)|} (D\varphi - \operatorname{div}(\varphi)\mathbf{I})(x, y) \nabla \alpha(t, x, y) d(x, y) \\ & = \int_{\mathbb{R}^n} \sigma \frac{\nu_\varepsilon(t, x, h(t, x))^\top}{|\nu_\varepsilon(t, x, h(t, x))|} (D\varphi - \operatorname{div}(\varphi)\mathbf{I})(x, h(t, x)) \binom{-\nabla h(t, x)}{1} dx. \end{aligned}$$

Now the uniform convergence of $\nu_\varepsilon(t, x, h(t, x))$ to $\binom{-\nabla h(t, x)}{1}$ for $\varepsilon \searrow 0$ by Lemma 3.18 yields the convergence of the above term to (3.64).

Let vice versa (u, p, α) be a solution of the VOF-type formulation (3.8), (3.9) and (3.54) such that (u, p, h) satisfies the regularity assumptions of Theorem 3.13. Then again $\alpha = 1_{\Omega_1}(t)$, where the normal velocity of $\Gamma(t)$ is $u^\top \nu$. Moreover, $\llbracket u \rrbracket = 0$ on $\Gamma(t)$ by the regularity of u and clearly the first two PDEs in (2.12) follow. Finally, the jump condition in the third line of (2.12) follow from (3.8), (3.9) and (3.54), or equivalently from (3.6) and (3.7), by choosing test functions of the form

$$\varphi_\tau(t, x, y) = \phi(t, x) \psi_\tau(y - h(t, x)) \quad (3.69)$$

with $\phi \in C_c^\infty(I \times \mathbb{R}^n; \mathbb{R}^{n+1})$, $\psi_\tau(s) = \psi(s/\tau)$, $\psi \in C_c^1((-1, 1))$, $\psi \geq 0$, $\psi(0) = 1$, $\psi(-s) = \psi(s)$ and letting $\tau \searrow 0$. \square

The last step remaining is to justify the following VOF-type formulation for computing the sensitivities $(\delta u, \delta p)$.

$$\begin{aligned}
 & \int_{I \times \mathbb{R}^{n+1}} (\partial_t(\rho(\alpha)\delta u) + \operatorname{div}(\rho(\alpha)(\delta u \otimes u + u \otimes \delta u)) + \delta c)^\top \varphi d(t, x, y) \\
 & + \int_{I \times \mathbb{R}^{n+1}} S(\delta u, \delta p; \mu(\alpha)) : \nabla \varphi d(t, x, y) \\
 & + \int_I \int_{\mathbb{R}^{n+1}} (\rho_2 - \rho_1) u^\top (\partial_t \varphi + u \cdot \nabla \varphi) d\delta\alpha(t)(x, y) dt \\
 & - \int_I \int_{\mathbb{R}^{n+1}} [S(u, p; \mu(\alpha))] : \nabla \varphi d\delta\alpha(t)(x, y) dt
 \end{aligned} \tag{3.70}$$

$$\begin{aligned}
 & = \lim_{\varepsilon \searrow 0} - \int_{I \times \mathbb{R}^{n+1}} \sigma \left(\frac{\delta \nu_\varepsilon^\top}{|\nu_\varepsilon|} - \frac{\delta \nu_\varepsilon^\top \nu_\varepsilon \nu_\varepsilon^\top}{|\nu_\varepsilon|^3} \right) (D\varphi - \operatorname{div}(\varphi)I) \nabla \alpha d(t, x, y) \\
 & - \int_{I \times \mathbb{R}^{n+1}} \sigma \frac{\nu_\varepsilon^\top}{|\nu_\varepsilon|} (D\varphi - \operatorname{div}(\varphi)I) \nabla d\delta\alpha(t)(x, y) \quad \forall \varphi \in C_c^2(I \times \mathbb{R}^{n+1}; \mathbb{R}^{n+1}), \\
 & \int_{I \times \mathbb{R}^{n+1}} \operatorname{div}(\delta u) \psi d(t, x, y) = 0 \quad \forall \psi \in C_c^1(I \times \mathbb{R}^{n+1}),
 \end{aligned} \tag{3.71}$$

$$\delta\alpha \text{ satisfies (3.59),} \tag{3.72}$$

$$\delta u(0) = \delta u_0, \tag{3.73}$$

where ν_ε and $\delta \nu_\varepsilon$ are given by (3.67) and (3.68). Due to the limited spatial regularity of $\partial_t u$, we have to state time derivatives on the interface in weak form. The following lemma is still necessary for the prove of the hereafter theorem.

Lemma 3.21. *Let $\psi \in C_c^1(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})$. Then*

$$\begin{aligned}
 & - \int_{\mathbb{R}^{n+1}} \psi(x, y)^\top \nabla d\delta\alpha(t)(x, y) \\
 & = \int_{\mathbb{R}^n} \partial_y \psi(x, h(t, x))^\top \begin{pmatrix} -\nabla h(t, x) \\ 1 \end{pmatrix} \delta h(t, x) + \psi(x, h(t, x))^\top \begin{pmatrix} -\nabla \delta h(t, x) \\ 0 \end{pmatrix} dx \\
 & = \int_{\mathbb{R}^n} \operatorname{div}(\psi)(x, h(t, x)) \delta h(t, x) dx.
 \end{aligned}$$

Proof. The lemma can be proofed by the definition of distributional derivatives and

3.3. Differentiability of the Control-to-State Mapping

integration by parts, see [25]. \square

With that we are able to show the main result for the sensitivities of the VOF-type formulation we stated with (3.70)–(3.73).

Theorem 3.22. *Let (u, q) be the solution of (2.12) according to Theorem 3.13 and let (3.66) hold, which is satisfied for $\varepsilon_0 > 0$ small enough. Moreover, let $(\delta u, \delta p)$ be the sensitivities of (u, p) in Theorem 3.13 corresponding to $(\delta u_0, \delta c)$. Then $(\delta u, \delta p)$ solve the linearized VOF-type system (3.70)–(3.73).*

Let vice versa (u, p, α) be a solution of the VoF-type formulation (3.8), (3.9) and (3.54), where $\alpha(t)$ is the indicator function of a domain $\Omega_1(t) = \{(x, y) \in \mathbb{R}^n \times \mathbb{R} : y = h(t, x)\}$. If (u, p, h) has the regularity as in Theorem 3.13 and $(\delta u, \delta p, \delta \alpha)$ is a solution of (3.70)–(3.73) such that $(\delta u, \delta p)$ has the regularity as in Theorem 3.13, then $(\delta u, \delta p)$ coincide with the sensitivities according to Theorem 3.13.

Proof. Let $(u_0, h_0, c), (\delta u_0, 0, \delta c) \in \mathbb{U}_u(h_0) \times \mathbb{U}_h \times \mathbb{U}_c(t_0)$ be such that (u_0, h_0, c) and $(u_0, h_0, c) + (\delta u_0, 0, \delta c)$ satisfy the conditions of Theorem 3.13. Denote now by $(\hat{u}, \hat{p}, \llbracket \hat{p} \rrbracket, h)$ the unique solution of (3.17) for data (u_0, h_0, c) and by $(\hat{u}^s, \hat{p}^s, \llbracket \hat{p}^s \rrbracket, h^s)$ the one for data $(u_0, h_0, c) + s(\delta u_0, 0, \delta c)$. Let (u, p) and (u^s, p^s) be the corresponding states in physical coordinates according to (3.12) and let $\alpha = 1_{\Omega_1(t)}, \alpha^s = 1_{\Omega_1^s(t)}$ be the corresponding solutions of (3.54). Finally, let $(\delta u, \delta h, \delta p)$ be the directional derivatives (sensitivities) in direction $(\delta u_0, 0, \delta c)$ which exist by Theorem 3.13. By the differentiability result of Theorem 3.13 we know that with the extensions u_{\pm}, p_{\pm} in (3.46) the following holds, see (3.45), (3.47) and (3.48)

$$\frac{u^s - u}{s} \rightarrow \delta u \quad \text{in } C(\bar{I}; L^p(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})), \quad (3.74)$$

$$\frac{u_{\pm}^s - u_{\pm}}{s} \rightarrow \delta u_{\pm} \quad \text{in } L^p(I; H^{p,1}(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})), \quad (3.75)$$

$$\frac{p_{\pm}^s - p_{\pm}}{s} \rightarrow \delta p_{\pm} \quad \text{in } L^p(I; L^p(\mathbb{R}^{n+1}; \mathbb{R}^{n+1})). \quad (3.76)$$

We derive now the different terms in (3.70). Therefore we define

$$\Omega_s := \{(t, x, y) : \alpha^s = \alpha\}, \quad \Omega_s^c := \{(t, x, y) : \alpha^s(t) \neq \alpha\}.$$

We have for arbitrary $\varphi \in C_c^2(I \times \mathbb{R}^{n+1}; \mathbb{R}^{n+1})$

$$\int_{I \times \mathbb{R}^{n+1}} \frac{-1}{s} ((\rho(\alpha^s)u^s - \rho(\alpha)u)^\top \partial_t \varphi + \rho(\alpha^s)(u^s)^\top (u^s \cdot \nabla \varphi) - \rho(\alpha)u^\top (u \cdot \nabla \varphi)) d(t, x, y)$$

$$\begin{aligned}
 &= \int_{\Omega_s} \frac{-1}{s} (\rho(\alpha)(u^s - u)^\top \partial_t \varphi + \rho(\alpha)((u^s)^\top (u^s \cdot \nabla \varphi) - u^\top (u \cdot \nabla \varphi))) d(t, x, y) \\
 &+ \int_{\Omega_s^c} \frac{-1}{s} ((\rho(\alpha^s)u^s - \rho(\alpha)u)^\top \partial_t \varphi + \rho(\alpha^s)(u^s)^\top (u^s \cdot \nabla \varphi) - \rho(\alpha)u^\top (u \cdot \nabla \varphi)) d(t, x, y)
 \end{aligned}$$

By (3.74) and (3.75) we obtain for the first summand

$$\begin{aligned}
 &\int_{\Omega_s} \frac{-1}{s} (\rho(\alpha)(u^s - u)^\top \partial_t \varphi + \rho(\alpha)((u^s)^\top (u^s \cdot \nabla \varphi) - u^\top (u \cdot \nabla \varphi))) d(t, x, y) \\
 &\rightarrow \int_{I \times \mathbb{R}^{n+1}} -(\rho(\alpha)\delta u^\top \partial_t \varphi + \rho(\alpha)(\delta u^\top (u \cdot \nabla \varphi) + u^\top (\delta u \cdot \nabla \varphi)) d(t, x, y) \\
 &= \int_{I \times \mathbb{R}^{n+1}} (\partial_t(\rho(\alpha)\delta u) + \operatorname{div}(\rho(\alpha)(\delta u \otimes u + u \otimes \delta u)))^\top \varphi d(t, x, y).
 \end{aligned}$$

For the second summand we have by Theorem 3.13

$$\begin{aligned}
 &\int_{\Omega_s^c} \frac{-1}{s} ((\rho(\alpha^s)u^s - \rho(\alpha)u)^\top \partial_t \varphi + \rho(\alpha^s)(u^s)^\top (u^s \cdot \nabla \varphi) - \rho(\alpha)u^\top (u \cdot \nabla \varphi)) d(t, x, y) \\
 &\int_{I \times \mathbb{R}^n} \frac{-1}{s} \int_{h(t,x)}^{\max(h(t,x), h^s(t,x))} ((\rho_1 u^s - \rho_2 u)^\top \partial_t \varphi + \rho_1 (u^s)^\top (u^s \cdot \nabla \varphi) - \rho_2 u^\top (u \cdot \nabla \varphi)) d(t, x, y) \\
 &+ \int_{I \times \mathbb{R}^n} \frac{-1}{s} \int_{h^s(t,x)}^{\max(h(t,x), h^s(t,x))} ((\rho_2 u^s - \rho_1 u)^\top \partial_t \varphi + \rho_2 (u^s)^\top (u^s \cdot \nabla \varphi) - \rho_1 u^\top (u \cdot \nabla \varphi)) d(t, x, y) \\
 &\rightarrow \int_{I \times \mathbb{R}^n} (\rho_2 - \rho_1) u^\top (\partial_t \varphi + u \cdot \nabla \varphi)(t, x, h(t, x)) \delta h(t, x) d(t, x) \\
 &= \int_I \int_{\mathbb{R}^{n+1}} (\rho_2 - \rho_1) u^\top (\partial_t \varphi + u \cdot \nabla \varphi) d\delta \alpha(t)(x, y) dt,
 \end{aligned}$$

where we have used equation (3.60) in the last step.

3.3. Differentiability of the Control-to-State Mapping

For the next term in (3.70) we note that

$$\begin{aligned}
 & \int_{I \times \mathbb{R}^{n+1}} \frac{1}{s} (S(u^s, p^s; \mu(\alpha^s)) - S(u, p; \mu(\alpha))) : \nabla \varphi d(t, x, y) \\
 &= \int_{\Omega_s} \frac{1}{s} (S(u^s - u, p^s - p; \mu(\alpha)) : \nabla \varphi d(t, x, y) \\
 &+ \int_{\Omega_s^c} \frac{1}{s} (S(u^s, p^s; \mu(\alpha^s)) - S(u, p; \mu(\alpha)) : \nabla \varphi d(t, x, y).
 \end{aligned} \tag{3.77}$$

Now (3.75) and (3.76) yield

$$\int_{\Omega_s} \frac{1}{s} (S(u^s - u, p^s - p; \mu(\alpha)) : \nabla \varphi d(t, x, y) \rightarrow \int_{I \times \mathbb{R}^{n+1}} S(\delta u, \delta p; \mu(\alpha)) : \nabla \varphi d(t, x, y).$$

Moreover, by using (3.61) and Theorem 3.9 we have

$$\begin{aligned}
 & \int_{\Omega_s^c} \frac{1}{s} (S(u^s, p^s; \mu(\alpha^s)) - S(u, p; \mu(\alpha)) : \nabla \varphi d(t, x, y) \\
 &= \int_{I \times \mathbb{R}^n} \frac{1}{s} \int_{h(t,x)}^{\max(h(t,x), h^s(t,x))} (S(u_-^s, p_-^s; \mu_1) - S(u_+, p_+; \mu_2)) : \nabla \varphi d(t, x, y) \\
 &+ \int_{I \times \mathbb{R}^n} \frac{1}{s} \int_{h^s(t,x)}^{\max(h(t,x), h^s(t,x))} (S(u_+^s, p_+^s; \mu_2) - S(u_-, p_-; \mu_1)) : \nabla \varphi d(t, x, y) \\
 &\rightarrow - \int_{I \times \mathbb{R}^n} [S(u, p; \mu(\alpha))](t, x, h(t, x)) \delta h(t, x) : \nabla \varphi(t, x, h(t, x)) d(t, x) \\
 &= - \int_I \int_{\mathbb{R}^{n+1}} [S(u, p; \mu(\alpha))] : \nabla \varphi d\delta \alpha(t)(x, y) dt.
 \end{aligned}$$

Here, we have used (3.60) and the imbedding (3.61) in the last step. Finally, the surface tension term (3.64) has by Theorem 3.13 and with the abbreviations

$$\tilde{\nu}(t, x) = \begin{pmatrix} -\nabla h(t, x) \\ 1 \end{pmatrix}, \quad \delta \tilde{\nu}(t, x) = \begin{pmatrix} -\nabla \delta h(t, x) \\ 0 \end{pmatrix}$$

as well as imbedding (3.62) the following directional derivative

$$\begin{aligned}
 & \int_{I \times \mathbb{R}^n} \sigma \left(\frac{\delta \tilde{\nu}^\top}{|\tilde{\nu}|} - \frac{\delta \tilde{\nu}^\top \tilde{\nu} \tilde{\nu}^\top}{|\tilde{\nu}|^3} \right) (t, x) (D\varphi - \operatorname{div}(\varphi)I)(t, x, h(t, x)) \tilde{\nu}(t, x) d(t, x) \\
 & + \int_{I \times \mathbb{R}^n} \sigma \frac{\tilde{\nu}^\top}{|\tilde{\nu}|} \left(\partial_y (D\varphi - \operatorname{div}(\varphi)I)(t, x, h(t, x)) \delta h(t, x) \tilde{\nu}(t, x) \right. \\
 & \quad \left. + (D\varphi - \operatorname{div}(\varphi)I)(t, x, h(t, x)) \delta \tilde{\nu}(t, x) \right) d(t, x).
 \end{aligned} \tag{3.78}$$

Now the first integral on the right hand side of (3.70) converges to the first integral in (3.78) by first applying Lemma 3.17 and then Lemmas 3.18 and 3.19. By using first Lemma 3.21, where we have to note that $\nu_\varepsilon(t, x, y)$ depends close to $\Gamma(t)$ only on x by (3.66), and then Lemma 3.18 as well as the fact that $\nabla \delta h$ is continuous by (3.62), the second integral on the right hand side of (3.70) converges to the second integral in (3.78). (3.71) and (3.73) are obvious and (3.72) follows by Proposition 3.15.

Let vice versa (u, p, α) be a solution of the VOF-type formulation (3.8), (3.9) and (3.54) and $(\delta u, \delta p, \delta \alpha)$ a solution of (3.70)–(3.73) with the regularities as in Theorem 3.13. By Theorem 3.20, (u, p, h) coincides with the solution of (2.12) in Theorem 3.13 and (3.59) implies by Proposition 3.15 that $\delta \alpha$ and δh correspond to each other via (3.60). Hence, (3.70)–(3.73) ensure that (u, p) satisfy the linearization of (2.12) on $\Omega(t)$ and that $\delta \alpha$ provides for given δu the correct δh . It remains to show that (3.70)–(3.73) implies the correct linearized jump condition. Denote the tested surface tension term from (3.64) for $\phi \in C_c^\infty(I \times \mathbb{R}^n; \mathbb{R}^{n+1})$ by

$$K(h; \phi) := \int_{I \times \mathbb{R}^n} \sigma \operatorname{div}_x \left(\frac{\nabla h(t, x)}{\sqrt{1 + |\nabla h(t, x)|^2}} \right) \binom{-\nabla h(t, x)}{1}^\top \phi(t, x) d(t, x).$$

The jump condition in strong form is equivalent to

$$- \int_{I \times \mathbb{R}^n} \phi(t, x)^\top [S(u, p; \mu)(t, x, h(t, x))] \binom{-\nabla h(t, x)}{1} d(t, x) = K(h; \phi) \tag{3.79}$$

for all $\phi \in C_c^\infty(I \times \mathbb{R}^n; \mathbb{R}^{n+1})$ with $S(u, p; \mu) = -pI + \mu(\nabla u + \nabla u^\top)$. In the transformed variables the jump condition reads

$$- \int_{I \times \mathbb{R}^n} \phi(t, x)^\top [\hat{S}(\hat{u}, \hat{p}, h; \hat{\mu})(t, x, 0)] \binom{-\nabla h(t, x)}{1} d(t, x) = K(h; \phi)$$

3.3. Differentiability of the Control-to-State Mapping

for all $\phi \in C_c^\infty(I \times \mathbb{R}^n; \mathbb{R}^{n+1})$, where with the notation of the transformed deformation tensor, see (3.13), $\hat{S}(\hat{u}, \hat{p}, h; \hat{\mu}) = -\hat{p}I + \hat{\mu}\mathcal{D}(\hat{u}, h)$. Thus, the sensitivities $\delta\hat{u}, \delta\hat{p}, \delta h$ satisfy the linearized jump condition

$$\begin{aligned} - \int_{I \times \mathbb{R}^n} \phi(t, x)^\top \left([\partial_{(\hat{u}, \hat{p}, h)} \hat{S}(\hat{u}, \hat{p}, h; \hat{\mu}) \cdot (\delta\hat{u}, \delta\hat{p}, \delta h)(t, x, 0)] (-\nabla_1^{h(t, x)}) \right. \\ \left. + [\hat{S}(\hat{u}, \hat{p}, h; \hat{\mu})(t, x, 0)] (-\nabla_0^{\delta h(t, x)}) \right) d(t, x) = \partial_h K(h; \phi) \cdot \delta h \end{aligned} \quad (3.80)$$

for all $\phi \in C_c^\infty(I \times \mathbb{R}^n; \mathbb{R}^{n+1})$. We show now, that under the regularity ensured by Theorem 3.13, (3.80) is implied by the weak formulation (3.70) by using test functions of the form

$$\varphi_\tau(t, x, y) = \phi(t, x) \psi_\tau(y - h(t, x)) \quad (3.81)$$

with $\phi \in C_c^\infty(I \times \mathbb{R}^n; \mathbb{R}^{n+1})$, $\psi_\tau(s) = \psi(s/\tau)$, $\psi \in C_c^1((-1, 1))$, $\psi \geq 0$, $\psi(0) = 1$, $\psi(-s) = \psi(s)$ for $\tau \searrow 0$. Then

$$\begin{aligned} \nabla \varphi_\tau(t, x, y) &= \psi'_\tau(y - h(t, x)) (-\nabla_1^{h(t, x)}) \phi(t, x)^\top + \psi_\tau(y - h(t, x)) (\nabla_0^{\phi(t, x)}), \\ \partial_t \varphi_\tau(t, x, y) &= -\phi(t, x) \psi'_\tau(y - h(t, x)) \partial_t h(t, x) + \partial_t \phi(t, x) \psi_\tau(y - h(t, x)). \end{aligned}$$

We test the weak form (3.8) also in time and rewrite it in the transformed variables. This results in

$$\begin{aligned} \int_{I \times \mathbb{R}^{n+1}} ((\partial_t(\rho(\alpha)u) + \operatorname{div}(\rho(\alpha)u \otimes u))(t, x, y)^\top \varphi + S(u, p; \mu) : \nabla \varphi) d(t, x, y) = \\ \int_{I \times \mathbb{R}^{n+1}} \left(-\hat{p}(x, y) \hat{u}^\top (\partial_t \varphi(t, x, y + h(t, x)) + \hat{u} \cdot \nabla \varphi(t, x, y + h(t, x))) \right. \\ \left. + \hat{S}(\hat{u}, \hat{p}, h; \hat{\mu})(t, x, y) : \nabla \varphi(t, x, y + h(t, x)) \right) d(t, x, y) = K(h; \varphi(\cdot, \cdot, h(\cdot, \cdot))). \end{aligned} \quad (3.82)$$

For the right hand side we have used that as in the proof of Theorem 3.20 the right hand side of (3.8) coincides with (3.64). For the test function (3.81) we obtain

$$\begin{aligned} \partial_t \varphi_\tau(t, x, y + h(t, x)) + \hat{u} \cdot \nabla \varphi_\tau(t, x, y + h(t, x)) \\ = -\phi(t, x) \psi'_\tau(y) (\partial_t h(t, x) + v^\top \nabla h(t, x) - w) + \psi_\tau(y) (\partial_t \phi(t, x) + v \cdot \nabla \phi(t, x)). \end{aligned}$$

Moreover, for any $(\bar{x}, \bar{y}) \in \Gamma(\bar{t})$ and $(x(t), y(t))$ with $(x'(t), y'(t)) = u(x(t), y(t))$, $(x(\bar{t}), y(\bar{t})) = (\bar{x}, \bar{y})$ one has $y(t) - h(t, x(t)) = 0$ and thus

$$0 = y'(t) - \partial_t h(t, x(t)) - \partial_x h(t, x(t)) x'(t)$$

$$= w(t, x(t), y(t)) - \partial_t h(t, x(t)) - v(t, x(t), y(t))^\top \nabla h(t, x(t)).$$

Hence,

$$\partial_t \varphi_\tau(t, x, y + h(t, x)) + \hat{u} \cdot \nabla \varphi_\tau(t, x, y + h(t, x)) = \psi_\tau(y)(\partial_t \phi(t, x) + v \cdot \nabla \phi(t, x)).$$

and inserting φ_τ in (3.82) yields

$$\begin{aligned} & \int_{I \times \mathbb{R}^{n+1}} \left(-\hat{\rho}(x, y) \hat{u}^\top \psi_\tau(y) (\partial_t \phi(t, x) + v \cdot \nabla \phi(t, x)) \right. \\ & \left. + \hat{S}(\hat{u}, \hat{p}, h; \hat{\mu}) : \left(\phi(t, x) \psi'_\tau(y) (-\nabla_1^h(t, x)) + \psi_\tau(y) (\nabla_0^{\phi(t, x)}) \right) \right) d(t, x, y) = K(h; \phi). \end{aligned}$$

We have already observed that the right hand side of (3.8) coincides with (3.64) and the right hand side of (3.70) with the derivative (3.78) of (3.64). Since $K(h; \phi)$ corresponds to using φ_τ in (3.64), $\partial_h K(h; \phi) \cdot \delta h$ can be expressed as the sum of the right hand side of (3.70) with $\varphi = \varphi_\tau$ and of the right hand side of (3.8) with $\varphi = -\partial_y \varphi_\tau \delta h$ (note again that φ_τ depends on h). Similarly, taking the derivative of the left hand side of (3.82) corresponds to the sum of the left hand side of (3.70) with $\varphi = \varphi_\tau$ and of the left hand side of (3.82) with $\varphi = -\partial_y \varphi_\tau \delta h$ (note that φ_τ depends on h), which results in

$$\begin{aligned} & \int_{I \times \mathbb{R}^{n+1}} \left(-\hat{\rho}(x, y) \psi_\tau(y) (\delta \hat{u}^\top (\partial_t \phi(t, x) + v \cdot \nabla \phi(t, x)) + \hat{u}^\top (\partial_t \phi(t, x) + \delta v \cdot \nabla \phi(t, x))) \right. \\ & \left. + \partial_{(\hat{u}, \hat{p}, h)} \hat{S}(\hat{u}, \hat{p}, h; \hat{\mu})(x, 0) \cdot (\delta \hat{u}, \delta \hat{p}, \delta h) : \left(\phi(t, x) \psi'_\tau(y) (-\nabla_1^h(t, x)) + \psi_\tau(y) (\nabla_0^{\phi(t, x)}) \right) \right. \\ & \left. + \hat{S}(\hat{u}, \hat{p}, h; \hat{\mu})(x, 0) : \phi(t, x) \psi'_\tau(y) (-\nabla_0^{\delta h(t, x)}) \right) d(t, x, y) = \partial_h K(h; \phi) \cdot \delta h. \end{aligned}$$

Note, we differentiate equivalently the middle term of (3.82) in transformed variables. By the assumed regularity for $\tau \searrow 0$, all terms containing the factor $\psi_\tau(y)$ tend to zero and the remaining terms converge to

$$\begin{aligned} & \int_{I \times \mathbb{R}^n} \left(-[\partial_{(\hat{u}, \hat{p}, h)} \hat{S}(\hat{u}, \hat{p}, h; \hat{\mu})(x, 0) \cdot (\delta \hat{u}, \delta \hat{p}, \delta h)] : (-\nabla_1^h(t, x)) \right. \\ & \left. - [\hat{S}(\hat{u}, \hat{p}, h; \hat{\mu})(x, 0)] : (-\nabla_0^{\delta h(t, x)}) \right) \phi(t, x) d(t, x) = \partial_h K(h; \phi) \cdot \delta h. \end{aligned}$$

This is exactly the linearized jump condition (3.80). \square

3.4 Sensitivity System with respect to Liquid Viscosity

The former section yields fundamental theoretical results for the two-phase system with jump conditions and also treat the VOF approach for special controls. In the following, we will extend the results for a further control, the liquid viscosity μ_l , in a formal way. Extensive proofs of existence and uniqueness for this problem are skipped, as this would go beyond the scope of this work. However, further investigations are planned for the future. The reason why we add this part at this point is, that we will apply the sensitivity equations with respect to the material parameter μ_l to the doctor blading test case in Chapter 5. So we will now formally establish the sensitivity system for the state equations in (2.21), which consists of the momentum equation (2.20), the continuity equation (2.3) and the transport equation (2.19). For a clearer representation we define $C := (C^1, C^2, C^3, C^4, C^5)^\top$ with

$$\begin{aligned} C^{1,2,3} &:= \rho(\partial_t u + u \cdot \nabla u) - \mu \Delta u - \rho g + \nabla p - \sigma \kappa \nabla \alpha, \\ C^4 &:= \nabla \cdot u, \\ C^5 &:= \partial_t \alpha + \nabla \cdot (\alpha u). \end{aligned}$$

Here we have to remember, that the density ρ and the viscosity μ also depend on the phase fraction α , see equations (2.17) and (2.18), so it holds

$$\begin{aligned} \rho(\alpha) &= \alpha \rho_l + (1 - \alpha) \rho_g, \\ \mu(\alpha) &= \alpha \mu_l + (1 - \alpha) \mu_g. \end{aligned}$$

The goal is to find the sensitivities δy of the state variables $y = (u, p, \alpha)$ with respect to the control $q = \mu_l$. Hence, the sensitivities are defined as

$$\begin{aligned} \delta y &:= \frac{d(u, p, \alpha)}{dq} = \left(\frac{du}{dq}, \frac{dp}{dq}, \frac{d\alpha}{dq} \right), \\ \delta u &:= \frac{du}{dq}, \quad \delta p := \frac{dp}{dq}, \quad \delta \alpha := \frac{d\alpha}{dq}. \end{aligned}$$

In order to solve equation (3.3), we have to determine the two terms $C_y(y(q), q)$ and $C_q(y(q), q)$. We start with the second one. The only term in C containing the liquid viscosity is the diffusion term $-\mu \Delta u$, where μ_l is incorporated in μ , see equation (2.18). For the viscosity it holds

$$\partial_{\mu_l} \mu(\alpha, \mu_l) = \alpha.$$

Then, we obtain by direct formal differentiation of the state equations $C^{1,\dots,5}$ with respect to the control parameter $q = \mu_l$ the following partial derivatives

$$\partial_{\mu_l} C^{1,2,3} \delta \mu_l = -\alpha \Delta u, \quad \partial_{\mu_l} C^4 \delta \mu_l = 0, \quad \partial_{\mu_l} C^5 \delta \mu_l = 0.$$

This results in

$$C_q(y(q), q) \delta q = (-\alpha \Delta u, 0, 0). \quad (3.83)$$

For the calculation of $C_y(y(q), q)$, we formally differentiate the equations with the help of product and chain rule again. The challenging part is the momentum equation, where some preparatory considerations must be made. As we already stated above, density and viscosity depend on the phase fraction variable α . Therefore, we obtain the following derivatives with respect to α

$$\begin{aligned} \rho' &:= \partial_\alpha \rho = \rho_l - \rho_g, \\ \mu' &:= \partial_\alpha \mu = \mu_l - \mu_g, \end{aligned}$$

where ρ_l, ρ_g and μ_l, μ_g are constant values. Then the derivative of $C^{1,\dots,5}$ with respect to the individual state variables yield

$$\begin{aligned} \partial_u C^{1,2,3} \delta u &= \rho (\partial_t \delta u + \delta u \cdot \nabla u + u \cdot \nabla \delta u) - \mu \Delta \delta u, \\ \partial_u C^4 \delta u &= \nabla \cdot \delta u, \\ \partial_u C^5 \delta u &= \nabla \cdot (\alpha \delta u), \\ \partial_p C^{1,2,3} \delta p &= \nabla \delta p, \\ \partial_p C^4 \delta p &= 0, \\ \partial_p C^5 \delta p &= 0, \\ \partial_\alpha C^{1,2,3} \delta \alpha &= \rho' \delta \alpha (\partial_t u + u \cdot \nabla u) - \mu' \delta \alpha \Delta u - \rho' \delta \alpha g - \sigma (\kappa' \nabla \alpha + \kappa \nabla \delta \alpha), \\ \partial_\alpha C^4 \delta \alpha &= 0, \\ \partial_\alpha C^5 \delta \alpha &= \partial_t \delta \alpha + \nabla \cdot (\delta \alpha u). \end{aligned}$$

Some of the terms within this derivatives still need to be examined in more detail, e.g. the curvature term in the representation of $\partial_\alpha C^{1,2,3} \delta \alpha$. Just as density and viscosity, the curvature also depends on the phase fraction α , see equation (2.16). As we stated in the referenced equation, the curvature can be expressed in terms of minus the divergence of the interface normal ν_Γ , if the interface Γ is sufficiently smooth. This is the case, if Γ is a Lipschitz boundary, since then the outer normal vector ν_Γ exists almost everywhere. Furthermore, the interface normal ν_Γ itself can

3.4. Sensitivity System with respect to Liquid Viscosity

be calculated with the help of the phase fraction field α and leads to

$$\kappa = -\nabla \cdot \nu_\Gamma = -\nabla \cdot \frac{\nabla \alpha}{|\nabla \alpha|}. \quad (3.84)$$

To find a representation for the derivative of the curvature κ with respect to the phase fraction variable α , we add a small constant $\varepsilon > 0$ to the denominator of the fraction to avoid numerical difficulties in regions where $\nabla \alpha = 0$. Furthermore, this will match our numerical considerations in Chapter 4. This results in

$$\kappa' := \partial_\alpha \kappa = -\partial_\alpha \nabla \cdot \frac{\nabla \alpha}{|\nabla \alpha| + \varepsilon} = -\nabla \cdot \left(\partial_\alpha \frac{\nabla \alpha}{|\nabla \alpha| + \varepsilon} \right).$$

The last equality results from the linearity of the divergence operator. Now let's have a closer look at the argument of divergence. Applying the quotient rule, we obtain

$$\partial_\alpha \frac{\nabla \alpha}{|\nabla \alpha| + \varepsilon} = \frac{\nabla \delta \alpha (|\nabla \alpha| + \varepsilon) - \nabla \alpha \frac{\nabla \alpha^T \nabla \delta \alpha}{|\nabla \alpha|}}{(|\nabla \alpha| + \varepsilon)^2}.$$

Altogether, the derivative of the curvature is

$$\kappa' := \partial_\alpha \kappa = -\nabla \cdot \left(\frac{\nabla \delta \alpha (|\nabla \alpha| + \varepsilon) - \nabla \alpha \frac{\nabla \alpha^T \nabla \delta \alpha}{|\nabla \alpha|}}{(|\nabla \alpha| + \varepsilon)^2} \right). \quad (3.85)$$

Note, that we can calculate the curvature only at the interface, since it is only defined there. Outside the interface region, in $\Omega \setminus \Gamma$, the curvature becomes zero and therefore the surface tension term is also equal to zero.

Altogether, we end up with the following derivative of our state equations with respect to the state $y = (u, p, \alpha)$

$$C_y(y(q), q) \delta y = \begin{pmatrix} \rho' \delta \alpha \partial_t u + \rho (\partial_t \delta u + \delta u \cdot \nabla u + u \cdot \nabla \delta u) + \rho' \delta \alpha u \cdot \nabla u \\ -\mu \Delta \delta u - \mu' \delta \alpha \Delta u + \nabla \delta p - \rho' \delta \alpha g - \sigma (\kappa' \nabla \alpha + \kappa \nabla \delta \alpha) \\ \nabla \cdot \delta u \\ \partial_t \delta \alpha + \nabla \cdot (\alpha \delta u + \delta \alpha u) \end{pmatrix} \quad (3.86)$$

This leads to the following formulation of the sensitivity system with the addi-

tional control term

$$\begin{aligned}
 & \rho' \delta \alpha \partial_t u + \rho (\partial_t \delta u + \delta u \cdot \nabla u + u \cdot \nabla \delta u) + \rho' \delta \alpha u \cdot \nabla u - \mu \Delta \delta u \\
 & - \mu' \delta \alpha \Delta u + \nabla \delta p - \rho' \delta \alpha g - \sigma (\kappa' \nabla \alpha + \kappa \nabla \delta \alpha) - \alpha \Delta u = 0, \\
 & \nabla \cdot \delta u = 0, \\
 & \partial_t \delta \alpha + \nabla \cdot (\alpha \delta u + \delta \alpha u) = 0.
 \end{aligned} \tag{3.87}$$

With system (3.87), we can calculate our sensitivities δu , δp and $\delta \alpha$, where solving this system means to solve equation (3.3). This system also needs appropriate initial and boundary conditions to complete the description. We require

$$(\delta u, \delta p, \delta \alpha)(0) = (\delta u, \delta p, \delta \alpha)_0 \quad \text{in } \Omega(0)$$

and as boundary conditions we obtain

$$\delta u = 0, \quad \nabla \delta p = 0, \quad \nabla \delta \alpha = 0 \quad \text{on } \partial \Omega(t).$$

These sensitivity boundary conditions result, if we assume Dirichlet BCs for the velocity and Neumann BCs for the pressure and phase fraction. Once both, the state variables and the sensitivities are calculated, the objective functional value and the derivative of the objective function can be calculated for the respective control. The objective function we mainly consider in our optimization problems is a tracking type functional. It measures the difference between the actual state y and a desired state y_d through the L^2 -norm in space and time. The objective function then reads

$$j(y, q) := \frac{1}{2} \|y - y_d\|_{L^2(I \times \Omega)}^2.$$

For the derivatives with respect to the state and the control we obtain

$$\begin{aligned}
 j_y(y, q) &:= y - y_d, \\
 j_q(y, q) &:= 0.
 \end{aligned}$$

Using these results for j_y , j_q and the sensitivities δy , obtained from system (3.87), the derivative of the reduced objective function with respect to the control q can be calculated, see (3.2)

$$dj(y(q), q) := (y - y_d, \delta y) = (y - y_d, \delta u, \delta p, \delta \alpha).$$

Therewith, the control q can be updated for the next optimization step, as we will see in the next section.

3.5 Sensitivity-based Optimization Algorithm

In this subsection, we show the general structure of a gradient-based optimization algorithm using sensitivities, inspired by [36, p. 58]. Here, we assume that the control is a vector $q \in \mathbb{R}^N$, $N \in \mathbb{N}$. Hence, the sensitivity calculations of the following algorithm has to be exercised for $n = 1, \dots, N$. The algorithm stops, if a prescribed stopping criteria is fulfilled and a satisfactory convergence is achieved.

Algorithm 3.1 Sensitivity-based Optimization Algorithm

- 1: Initialize control variables with $q^0 \in \mathbb{R}^N$
- 2: For $k = 0, 1, 2, \dots$:
- 3: Determine state $y^k = y(q^k)$ by solving the state equations $C(y^k, q^k) = 0$
- 4: Determine the sensitivities δy_n^k by solving the N linear sensitivity systems

$$C_y(y^k, q^k)\delta y_n^k + C_{q_n^k}(y^k, q^k) = 0, \quad n = 1, \dots, N$$

- 5: Compute the gradient of the objective functional, for $n = 1, \dots, N$, by solving

$$\frac{dj}{dq_n^k} = j_y(y^k, q^k)\delta y_n^k + j_{q_n^k}(y^k, q^k)$$

- 6: Determine the increment in the control parameters, e.g., in a simple gradient method with an appropriate chosen size β_k from

$$\delta q_n^k = -\beta_k \frac{dj}{dq_n^k}, \quad n = 1, \dots, N$$

- 7: Update the control variables with $q^{k+1} = q^k + \delta q^k$
-

Note, that each iteration of this optimization algorithm requires at least one evaluation of the flow system. Once, the function evaluation $j(q^k)$ and the derivative $\frac{dj}{dq^k}$ are known, this first order optimization algorithm can be executed. Nevertheless, more sophisticated methods could be used to determine the increment of the controls δq^k in step 6, for example a quasi Newton approach such as the BFGS method [36]. Although the method approximates the Hessian matrix iteratively, we would prefer to avoid the application of second order derivatives. If the objective function is of a specific form, for example a tracking type function, there is the possibility to utilize more specialized methods such as the Gauss-Newton method. In Chapter 5, more precisely in Section 5.2.2 and 5.3, we will discuss this subject in more detail and we will see the justification for this approach. At the moment, different methods are imaginable in step 6.

As we already stated before, solving the investigated flow control problem exactly is not possible due to the complexity of the state equations and therefore approximative solutions must be considered. Now the question is, when this discretization takes place. In general, two different proceedings are possible, *first-discretize-then-optimize* or *first-optimize-then-discretize*. If we follow an *first-discretize-then-optimize* approach, the continuous flow equations are discretized at first. Then, the discrete flow equations are differentiated to obtain a discrete sensitivity system. A popular way to obtain the discrete sensitivity equations is to use automatic differentiation techniques, see e.g., [44]. The other possibility is to follow the *first-optimize-then-discretize* approach. Here, the sensitivity equations are obtained at the continuous PDE level and then the results are discretized. Solving the discretized sensitivity equations provides a discrete approximation of the sensitivities. In general, the discrete sensitivities from the *first-discretize-then-optimize* and the *first-optimize-then-discretize* approach do not conform, but they should converge to their continuous counterparts respectively, as the grid size goes to zero [36]. The same applies to the gradients of the discretized objective functionals, since they are calculated from the discrete sensitivities. Both approaches can also be applied analogously to adjoint variables, see e.g., [75].

There are good arguments for both of these ways and it depends on the problem formulation itself, the software requirements and not least on the user, which method is selected. An advantage of the *first-discretize-then-optimize* approach is the consistency of functional gradients. Discretizing the differentiated functionals does not necessarily yield the true gradients, but differentiating the discrete functionals still leads to the exact gradients of the discrete state equations and therefore also to the exact gradient of the discrete objective function [36]. In the case of the *first-optimize-then-discretize* approach, this drawback can be overcome with error estimators and adequate mesh refinement. However, the advantage of the second approach is the ease of realization. The more complex the flow system, the more time-consuming automatic differentiation is to implement and to solve. If in our case an interface and a three-phase contact line are added to the standard flow equations, then a lot of equations have to be differentiated automatically. Furthermore, due to its modular structure the targeted software offers ideal conditions for discretizing the individual equations operator-wise. Surely, care must be taken to ensure that the state and sensitivity equations are discretized as accurately as possible, otherwise the discrete approximations of the state and the sensitivities will not be consistent. Therefore, we discretize and implement the system in a similar way to the original system and make use of the modular structure of the existing solver and operator implementations. More details will be discussed in Section 4.3.

Numerical Solution

In times of high computational power the solution of the complex PDE systems describing multiphase flow with direct numerical simulations (DNS) is preferable and well feasible. It was mainly used and advanced for the application of homogeneous fluids and as a standard tool for turbulence problems. In this thesis we follow the DNS approach although we have a predominant laminar flow regime, since the equations, governing the fluid flow in the several phases, are highly nonlinear and the position of the phase boundaries has to be found as a part of the solution. Turbulent effects that might occur are captured by solving the previous equations on a sufficiently fine mesh. There are exact analytical solutions only for the simplest problems, for example the steady-state motion of bubbles and droplets in Stokes flow [93]. For the more sophisticated questions, a numerical treatment with discretization is indispensable.

We start in 4.1 with a description of the finite volume discretization, see Section 4.1.1, and the temporal discretization using an implicit Euler method, see Section 4.1.2. Based on these methods, the discretized equations are derived for the primal equation system in Section 4.1.3, using various operator discretizations and interpolation schemes. The numerical solutions of the α -transport equation with the compression approach and a flux-corrected transport algorithm as well as of the Navier-Stokes equations with a pressure-correction method follow in the Sections 4.2.1 and 4.2.2. The discretization and solution of the sensitivity system is subsequently discussed in Section 4.3. Afterwards, Section 4.4 contains a description of the applied software, where we first point out some special simulation characteristics of OpenFOAM in Section 4.4.1 and then discuss the main implementation facts of the new implemented sensitivity solver `interSensFoam`, based on the OpenFOAM inherent incompressible flow solver `interFoam`, see Section 4.4.2.

4.1 Discretization

When we talk about discretization of our problem, we always mean a finite-dimensional formulation of the problem. This includes the discretization of the solution domain as well as the discretization of the equations. The discretization of the solution domain is divided in a spatial and a temporal discretization. We use the finite volume method (FVM) as spatial discretization and the implicit Euler method for the temporal discretization, both described in detail in the next sections. The aim of discretizing the equations is to obtain a discrete system of algebraic equations, which can be solved by standard linear solvers. Since we have to deal with nonlinear PDEs of second order, the discretization practice should be also second order accurate in space and time, if possible.

In general, we use an Eulerian frame of reference for our quantities of interests, which is widely used in fluid dynamics. That means, that all variables are given on fixed points in space in the discrete setting. As a consequence, the variables are represented as a function of the position x and the time t , e.g. the velocity is represented as $u(x, t)$. In contrast, the Lagrangian approach describes the motion of individual fluid particles following their path lines, without a fixed grid.

4.1.1 Finite Volume Discretization

In the presence of fluid, it is advantageous and common to investigate a defined spatial range filled with fluid, called control volume. We know these control volumes already from the description of the conservation equations and we follow this idea also in our discretization approach. With the finite volume method, the continuous solution domain Ω is transformed into the approximated domain Ω^h with mesh size h , partitioned into finitely many control volumes V_i

$$\Omega^h := \bigcup_i V_i \subset \mathbb{R}^d, \quad \text{for } i \in \{1, \dots, n\}, n \in \mathbb{N}, d \in \{2, 3\}.$$

In the following, we will also denote Ω^h as mesh and V_i as cells. This approximation provides a finite number of non-overlapping cells bounded by a finite number of planar faces f . The discrete values of the variables are calculated at the cell centers

as volume averages over the control volumes. For a quantity ψ then it holds

$$\psi_c = \frac{1}{|V_i|} \int_{V_i} \psi(x) dv.$$

Here, ψ_c is the cell-centered approximation of ψ . It can be shown, that this piecewise constant approximation of the cells is second-order accurate, if the value ψ_c is associated with the cell centroid x_c of the control volume, see e.g. in [59]. Since all dependent variables share the same control volumes, we have a so-called collocated arrangement. This has some advantages over a staggered grid, where the different variables are stored at both the cell centers of the control volumes and the cell faces, especially for complex geometries and discontinuous variables [28].

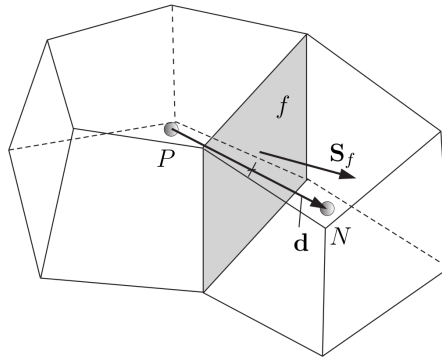


Figure 4.1. Two control volumes with common boundary face f [62].

In Figure 4.1, P and N are the centroids of two adjacent cells and \mathbf{d} is the distance between P and N . We write ψ_P and ψ_N for the cell-centered values of a variable ψ , if we need to distinguish between the owner and the neighbor cell, and ψ_f for the face-centered value. The intersection between \mathbf{d} and f does not necessarily have to coincide with the face center point of f . The face area vector \mathbf{S}_f is normal to the face f with the magnitude equal to the area of the face itself. A distinction is made between internal and boundary cells.

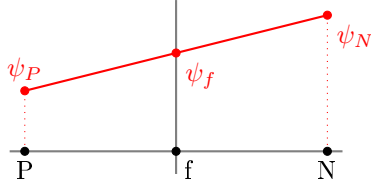
Remember, with the help of the divergence theorem, see Theorem 2.9, we are able to convert the volume integrals into surface integrals. This is used to discretize the governing equations. As a consequence, we not only need the cell-centered values of the variables but also face-centered values. Face-centered variables are obtained from the cell-centered variables by interpolation. Various interpolation schemes are available for this purpose. They differ regarding order of accuracy, boundedness

and the degree of diffusiveness they produce. Two prominent representatives of interpolation schemes are the linear interpolation, also called central differencing and the upwind differencing scheme, we will introduce in the following.

Central Differencing (CD)

The face-centered values are calculated by

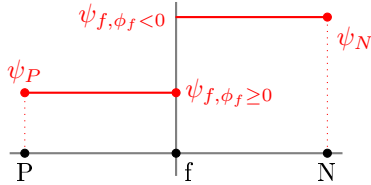
$$\psi_{f,cd} = f_x \psi_P + (1 - f_x) \psi_N,$$

$$f_x = \frac{fN}{PN} = \frac{|x_f - x_N|}{|d|}.$$


This linear interpolation scheme is second order accurate, but solutions are possibly unbounded and oscillatory [20].

Upwind Differencing (UD)

The face-centered value is equal to the upstream cell-centered value

$$\psi_{f,ud} = \begin{cases} \psi_P & \text{for } \phi_f \geq 0, \\ \psi_N & \text{for } \phi_f < 0. \end{cases}$$


Here, ϕ_f is defined as the mass flux which enters the face f , see equation (4.7) for computational details. This interpolation scheme is first order accurate and bounded, but diffusive [20].

Blended Differencing (BD)

This higher order differencing schemes is a mixture of the central and the upwind scheme and aims to preserve both boundedness and accuracy of the solution [20]

$$\psi_{f,bd} = (1 - \lambda_\psi) \psi_{f,ud} + \lambda_\psi \psi_{f,cd}. \tag{4.1}$$

Inserting the upper schemes leads to the following expression, which only depends on the face-centered values ψ_P and ψ_N

$$\psi_{f,bd} = [(1 - \lambda_\psi) \max(\text{sgn}(\phi_f), 0) + \lambda_\psi f_x] \psi_P + [(1 - \lambda_\psi) \max(-\text{sgn}(\phi_f), 0) + \lambda_\psi (1 - f_x)] \psi_N.$$

We naturally denote by $\text{sgn}(\phi_f)$ the sign of the flux ϕ_f . Various possibilities can be selected for the limiter λ_ψ , e.g. Minmod [73], SuperBee [73] or vanLeer [98]. We will discuss these limiters in detail, if we need them, at the appropriate place.

Linear Upwind Differencing (LUD)

Furthermore, there exist schemes using more than the direct neighboring cells. An example for such a second order upwind differencing scheme is the so-called linear upwind differencing scheme. It is a second order extension of the upwind interpolation scheme with a possible explicit correction based on the local cell gradient. The face values of a variable ψ can be calculated with

$$\psi_{f,lud} = \begin{cases} \psi_P + \frac{1}{2}(\psi_P - \psi_{PP}) = \frac{2}{3}\psi_P - \frac{1}{2}\psi_{PP} & \text{for } \phi_f \geq 0, \\ \psi_N + \frac{1}{2}(\psi_N - \psi_{NN}) = \frac{2}{3}\psi_N - \frac{1}{2}\psi_{NN} & \text{for } \phi_f < 0. \end{cases}$$

The second order accuracy results again in an unbounded scheme. With the help of a gradient or slope limiter function, this drawback can be overcome. By multiplying the second term with a well-designed function, this scheme detects strong gradients or changes in slope leading to oscillations, and switches locally to a simple upwind method. This yields a second order accurate and bounded scheme, which belongs to the TVD schemes. Note, since another cell-centered value is needed in upstream direction for this scheme, a careful handling of arbitrarily unstructured meshes is necessary.

4.1.2 Temporal Discretization

The time is discretized into a series of time intervals, also called time steps. The size of a time step is Δt . Let t denote the time in a given time interval $I := [0, T]$. In the following we will use the notation

$$\begin{aligned} \psi^o &:= \psi(t) && \text{value at the previous time } t \text{ (o: old),} \\ \psi^n &:= \psi(t + \Delta t) && \text{value at the actual time (n: new).} \end{aligned}$$

For the temporal discretization, the implicit Euler time differencing scheme is utilized. It is first-order accurate in time, guarantees boundedness and is unconditionally stable [57]. Then the temporal terms are discretized as

$$\partial_t \psi = \frac{\psi^n - \psi^o}{\Delta t}. \tag{4.2}$$

In computational fluid dynamics the Courant-Friedrichs-Lewy (CFL) condition is often used for the discretization of time dependent partial differential equations. It can be seen as a measure of the maximum number of mesh cells a given quantity can move through per time step. The CFL condition is defined as

$$C = \frac{u\Delta t}{\Delta x} \leq C_{\max}, \quad (4.3)$$

with velocity u , discrete time step Δt and mesh size Δx . The dimensionless number C is called the Courant number. For the stability of the explicit Euler method, a CFL number limit $C_{\max} = 1$ is necessary. Otherwise, it results in the propagation and amplification of numerical errors. The implicit Euler method is less sensitive to numerical instabilities, hence a larger value of C_{\max} is possible. But due to the solution procedure we choose for the α -transport equation, a CFL number limit is still necessary. Here, the fluxes are calculated and corrected in an explicit manner, details will follow in Section 4.2.1, which requires a strict time step limit during the numerical calculations. Hence, a maximal Courant number of $C_{\max} = 1$ is postulated for all further investigations, if not stated otherwise.

In our simulations, an adaptive time step control is beneficial to get the largest possible time step without violating the CFL number limit [57]. We set

$$\Delta t = \min \left\{ \min \left[\min \left(\frac{C_{\max}}{C} \Delta t_0, \left(1 + \lambda_1 \frac{C_{\max}}{C} \right) \Delta t_0 \right), \lambda_2 \Delta t_0 \right], \Delta t_{\max} \right\}, \quad (4.4)$$

with an initial time step Δt_0 . Moreover, λ_1 and λ_2 are dumping factors to avoid large changes of the time step [57], and Δt_{\max} is a maximal time step size the user can select.

4.1.3 Discretization of Equations

The considered conservation equations can be written in form of a general transport equation. We will start with a general formulation, since it contains all relevant operators and terms, which we also need for the discretization of our state equations. For a variable ψ and a velocity field u the general transport equation in conservation form is

$$\partial_t \psi + \nabla \cdot (u\psi) - \nabla \cdot (\mu \nabla \psi) = \mathbf{S}_\psi,$$

where \mathbf{S}_ψ is a source term depending on ψ .

4.1. Discretization

To obtain a discrete version of the general transport equation, and therefore also of the governing equations, we have to integrate over a time interval Δt and an arbitrary control volume V , see e.g. [28], which results in the following integral form

$$\int_t^{t+\Delta t} \int_V [\partial_t \psi + \nabla \cdot (u\psi) - \nabla \cdot (\mu \nabla \psi)] dv dt = \int_t^{t+\Delta t} \int_V \mathbf{S}_\psi dv dt. \quad (4.5)$$

For the discretization of the equations we will take a closer look at the individual terms and their operators in the following. Again, we write the subscript c for cell-centered values and the subscript f for face-centered values, while the superscript o stands for the previous time step and n for the actual one. An important tool to reformulate the equations in a discrete way is the Gauss theorem, which we already introduced in Section 2.1.2.

The discretization of the temporal term we already discussed in the section before, see equation (4.2). Of course, the other variables in equation (4.5) are also assigned to the actual or previous time step, which will be presented with the discrete versions of the individual state equations. The second term in equation (4.5) is the convective term, calculated by the divergence of $u\psi$. As discrete version we obtain

$$\int_V \nabla \cdot (u\psi) dv = \int_{\partial V} (u\psi) \cdot \nu ds = \sum_f S_f \cdot (u\psi)_f = \sum_f S_f \cdot u_f \psi_f = \sum_f \phi_f \psi_f, \quad (4.6)$$

where ν is the outer normal of ∂V and the mass flux through a face is defined as

$$\phi_f := S_f \cdot u_f. \quad (4.7)$$

For the diffusion term, the third term in equation (4.5), it holds

$$\int_V \nabla \cdot (\mu \nabla \psi) dv = \int_{\partial V} (\mu \nabla \psi) \cdot \nu ds = \sum_f S_f \cdot (\mu \nabla \psi)_f = \sum_f \mu_f S_f \cdot (\nabla \psi)_f.$$

The term $(\nabla \psi)_f$ is the gradient normal to the face, which is also denoted as the surface normal gradient. At the face f , it is calculated by

$$(\nabla \psi)_f = \frac{\psi_N^n - \psi_P^n}{|\mathbf{d}|}, \quad (4.8)$$

where \mathbf{d} is again the distance between the owner cell P and the neighbor cell N . The face values of the other variables are calculated with the differencing schemes we

introduced in Section 4.1.1. Which difference scheme is used will be specified at the relevant place. Since the source term on the right hand side of equation (4.5) varies, we will also treat them at the appropriate time. In some notations the Laplacian operator arises, but in our case it is replaced by the divergence of the gradient, since the viscosity is not constant. For this reason it is not necessary to consider it separately. For the full Navier-Stokes equations, the cell-centered gradient of a quantity is also required. The gradient is discretized as

$$(\nabla\psi)_P = \frac{1}{|V|} \sum_f S_f \psi_f.$$

Now we will combine the separate operator discretizations to match our state equations. The goal is to represent the discretized equations only with cell-centered values of the owner and neighbor cell of the respective face of the cell, to obtain a system of equations, depending only on cell-centered values again. For the resulting equations, we will subsequently present suitable solution procedures in 4.2.

α -Transport Equation

Remember that we defined the phase fraction function α as a characteristic function in Chapter 2.4, see equation (2.14). For the discretization we integrate this quantity over the volume corresponding to a computational control volume V_i . The liquid fraction field is now given by

$$\alpha_{c,i}(x, t) = \frac{1}{|V_i|} \int_{V_i} \alpha(x, t) dv. \quad (4.9)$$

Is there a cell completely filled with liquid, the liquid fraction $\alpha_{c,i}$ is equal to one and if the cell is completely filled with gas, the liquid fraction $\alpha_{c,i}$ is equal to zero. In any other case, $\alpha_{c,i}$ adopts a value between zero and one.

1	0,8	0,1	0	0
1	1	0,6	0	0
1	1	0,9	0,1	0
1	1	1	0,3	0

Figure 4.2. Exemplary discretized phase fraction field α .

Figure 4.2 shows an excerpt of a discrete phase fraction field, where the red curve displays the real interface, represented by a thin belt of cells with values between zero and one. Since with the algebraic VOF approach no reconstruction of the interface is used and an unstructured mesh discretization is straightforward. Hence, more complex geometries do not require any special treatment.

Due to the formulation as a one-field problem with the VOF approach, the phase fraction is transported with the α -transport equation through the domain, see equation (2.19), which involves some difficulties. The main challenge is, that we have to transport a discontinuous indicator function α that should imply a sharp interface between the different phases. Of course, this function can also adopt values between zero and one on a discrete level, but then it also has to be ensured that this transition zone, normally consisting of not more than one cell, remains limited and does not smear out. A solution in this context is offered by the surface compression approach, which is for example described by RUSCHE in [76]. The idea of this approach is, that an additional term prevents the smearing out of the interface over more and more cells. Therefore, a relative velocity between the phases is defined and integrated into the transport equation by a so-called compression term. This artificial term was introduced by JASAK and WELLER [49] with $\nabla \cdot (\alpha(1 - \alpha)u_r)$ to reduce spurious currents around the interface. u_r is the compression velocity, we also denote as relative velocity, and was originally defined as

$$u_r = u_1 - u_2,$$

where u_1 is for example the velocity of the liquid phase and u_2 the velocity of the gaseous phase. The problem is, that u_r can not be determined in the frame of the classical VOF approach, because there is only a single velocity u for both phases [53]. For the solution of the α -transport equation, we need to approximate this velocity and respectively the cell face flux ϕ_r . In Section 4.4.1 we will see how this is implemented in OpenFOAM. Note, due to the term $\alpha(1 - \alpha)$, the compression term acts only in close proximity to the interface, if α is between zero and one. In this region, the term limits the smearing of the interface because of the compensation of the diffusive fluxes [53]. Together with the surface compression term, we now have to solve the following extended α -transport equation

$$\partial_t \alpha + \nabla \cdot (\alpha u) + \nabla \cdot (\alpha(1 - \alpha)u_r) = 0. \quad (4.10)$$

The formal derivation of equation (4.10) can be found in Appendix A.1. In integral

form, this equation can be written as

$$\int_{V_i} \partial_t \alpha \, dv + \int_{\partial V_i} (\alpha u) \cdot \nu \, dv + \int_{\partial V_i} [\alpha(1-\alpha)u_r] \cdot \nu \, dv = 0.$$

Without loss of generality, we will consider only one single control volume V , so we will write V instead of V_i and α_c instead of $\alpha_{c,i}$ in the following. Again, the subscript f denotes the face-centered values of the variables, with $f \in \partial V$. Then we obtain as discretized α -transport equation with compression term

$$\begin{aligned} 0 &= \frac{\alpha_c^n - \alpha_c^o}{\Delta t} + \frac{1}{|V|} \sum_f S_f \cdot u_f^o \alpha_f^n + \frac{1}{|V|} \sum_f S_f \cdot u_r^o \alpha_f^n (1 - \alpha_f^n) \\ &= \frac{\alpha_c^n - \alpha_c^o}{\Delta t} + \frac{1}{|V|} \sum_f [\phi_f^o \alpha_f^n + \phi_r^o \alpha_f^n (1 - \alpha_f^n)]. \end{aligned} \quad (4.11)$$

Here, we used the implicit Euler method for the time derivative, see equation (4.2), and the convective discretization from equation (4.6). The time discretization is implicit, since we use α_f^n from the actual time step. In contrast, the velocity fields u_f^o and u_r^o are taken from the previous time step, wherefore we can substitute them together with the surface area vector S_f by ϕ_f^o or respectively ϕ_r^o , see equation (4.7). Remember, since we are in a multidimensional setting, we consider the sum of the fluxes over all cell faces f . How the fluxes ϕ_f^o and ϕ_r^o can exactly be interpolated and how to calculate the different α_f^n , we will discuss in more detail later in Section 4.2.1 when the numerical solution of the advection equation is presented.

Momentum Equation

To derive the discrete version of the momentum equation, some small equation transformations have to be made for a more efficient numerical evaluation. Based on equation (2.20), we rewrite the convective and the Laplacian term and obtain

$$\partial_t(\rho u) + \nabla \cdot (\rho u \otimes u) = -\nabla p + \nabla \cdot (\mu \nabla u) + \nabla \cdot (\mu \nabla u^\top) + \rho g + \sigma \kappa \nabla \alpha. \quad (4.12)$$

Then, the integral form of equation (4.12) over a control volume V results in

$$\begin{aligned} \int_V \partial_t(\rho u) \, dv + \int_{\partial V} (\rho u \otimes u) \cdot \nu \, ds &= - \int_V \nabla p \, dv + \int_{\partial V} (\mu \nabla u) \cdot \nu \, ds \\ &\quad + \int_{\partial V} (\mu \nabla u^\top) \cdot \nu \, ds + \int_V \rho g \, dv + \int_V \sigma \kappa \nabla \alpha \, dv. \end{aligned}$$

While forming the integrals, the Gauss theorem was used at three places to substitute the divergence with a surface integral. Most of the terms can be discretized with the operators introduced in the beginning of this section. The pressure term will be also treated as surface force, see equation (4.13), which corresponds to a conservative approach and is standard in the FV discretization context. A non-conservative approach, preserving the volumetric form, would create a global mesh dependent error [28]. Within the last term, the curvature appears. As stated in Chapter 2.3, the curvature κ is calculated as the negative divergence of the interface normal. In a discrete setting, this corresponds to the negative divergence of the face unit normal flux of a cell, we with interface normal ν_f , and it holds

$$\int_V \kappa \, dv = - \int_V \nabla \cdot \nu \, dv = - \sum_f S_f \cdot \nu_f, \quad \text{with} \quad \nu_f = \frac{(\nabla \alpha)_f}{|(\nabla \alpha)_f|}.$$

Here, $(\nabla \alpha)_f$ is the surface normal gradient as in equation (4.8). Using the implicit Euler time scheme of Section 4.1.2 and the discretized operators from above, the discrete momentum equation for the actual time step can be derived as follows

$$\begin{aligned} \int_V \frac{\rho_P^n u_P^n - \rho_P^o u_P^o}{\Delta t} \, dv + \sum_f \rho_f^n S_f \cdot u_f^o u_f^n &= -(\nabla p^n)_P |V| + \sum_f \mu_f^n S_f \cdot (\nabla u^n)_f \\ &+ \sum_f \mu_f^n S_f \cdot (\nabla u^o)_f^\top + (\rho^n g)_P |V| + (\sigma \kappa^n)_P (\nabla \alpha^n)_P |V|. \end{aligned} \quad (4.13)$$

Note, the convective term can be discretized as in equation (4.6), since a kind of linearization is performed by using an existing velocity field u^o from the former time step. For sufficiently small time steps it holds with a fixed point argument

$$(\rho u \otimes u)^n \approx \rho^n u^o \otimes u^n. \quad (4.14)$$

Furthermore, the second part of the diffusion term is considered at the old time step, since this term will be used in another way in Section 4.4.1. We can further transform equation (4.13) in

$$\begin{aligned} \frac{\rho_P^n u_P^n - \rho_P^o u_P^o}{\Delta t} |V| + \sum_f \rho_f^n \phi_f u_f^n &= -(\nabla p^n)_P |V| + \sum_f \mu_f^n S_f \frac{u_N^n - u_P^n}{|\mathbf{d}|} \\ &+ \sum_f \mu_f^n S_f \left(\frac{u_N^o - u_P^o}{|\mathbf{d}|} \right)^\top + (\rho^n g)_P |V| + (\sigma \kappa^n)_P (\nabla \alpha^n)_P |V|, \end{aligned} \quad (4.15)$$

where we again use the flux representation for velocities from the old time step at some point and the definition of the surface normal gradient. If no divergence occurs within a term, we can simply consider the face-centered values, as it is done for the last three terms in the momentum equation.

Continuity Equation

The discretized continuity equation applies

$$\int_V \nabla \cdot u \, dv = \sum_f S_f \cdot u_f^n = \sum_f \phi_f^n = 0.$$

Note, that we do not solve this equation per se, we rather use it to correct a not divergence-free velocity field, obtained in a first step of the solution procedure of the momentum equation. A detailed description follows in 4.2.2.

4.2 Numerical Solution Procedures

In this section, we present solution procedures for the α -transport equation and the Navier-Stokes equations. A summary of the solution algorithm as it is used in the standard OpenFOAM implementation can be found in [53, Algorithm 1]. This algorithm is the basis of the sensitivity solver, that we will present in the end of the chapter.

The goal of the discretization is to obtain a system of linear algebraic equations, divided in values for the owner and the neighbor cell. In general, we obtain the following linear equation formulation for the respective equations, where ψ represents one of the solution variables

$$a_P \psi_P^n + \sum_N a_N \psi_N^n = R_P. \quad (4.16)$$

The above equation refers to a certain control volume with cell values ψ_P and neighboring cell values ψ_N . R_P is known, since it contains all terms with known variable values. Overall, one obtains a system of the form

$$\mathbf{A} \psi^n = \mathbf{R}.$$

This system of algebraic equations can be solved with appropriate methods, we will

present among others in the next subsections. Since the resulting coefficient matrix \mathbf{A} is usually sparse, iterative methods are preferred. If additionally a_P and a_N depend on ψ , a fixed point iteration is necessary. The first step, however, is to set up this system and reach a formulation only based on the cell centered variables.

4.2.1 Numerical Solution of the α -Transport Equation

The numerical solution of the extended α -transport equation (4.11) is based on a flux-corrected transport (FCT) algorithm, introduced for one-dimensional problems in [15] and extended for multiple dimensions in [105]. Besides the challenge of solving such an advection equation for multiple phases while keeping the interface sharp, the boundedness of the α -field is of importance. A pure first order upwind method would lead to a smearing of the interface due to numerical diffusion, while higher order schemes have the disadvantage that they are unstable and produce numerical oscillations [34]. The idea of a FCT method is to use a flux limiter formulation of the problem to maintain a bounded solution. In the case of one-dimensional problems, a global boundedness constraint can be formulated for the discretization weights, which is not available in the case of multidimensional problems. Alternatively, the FCT method introduce local limiting of fluxes in multidimensional problems and for arbitrary cell shapes. This is achieved by splitting the advective fluxes into a lower order bounded flux and hence diffusive scheme, and a higher order anti-diffusive correction [21]. Therewith both the diffusiveness of the upwind scheme and the instability of the higher order scheme can be eliminated [34], while the solution stays bounded. This leads to an iterative procedure, where we calculate an intermediate value of α using the lower order monotonic scheme by upwind interpolation. Then, this value is corrected with an anti-diffusive flux. Since applying this flux completely would result in an unstable higher order flux, a correction factor or limiter λ is introduced. Starting from equation (4.11), a rearrangement according to the cell-centered phase fraction value at the actual time α_c^n results in

$$\begin{aligned} \alpha_c^n &= \Delta t \left[\frac{\alpha_c^o}{\Delta t} - \frac{1}{|V|} \sum_f [\phi_f^o \alpha_f^n + \phi_r^o \alpha_f^n (1 - \alpha_f^n)] \right] \\ &= \alpha_c^o - \frac{\Delta t}{|V|} \sum_f [\phi_f^o \alpha_f^n + \phi_r^o \alpha_f^n (1 - \alpha_f^n)]. \end{aligned} \quad (4.17)$$

Similar to the splitting of interpolation schemes in equation (4.1), the fluxes of the α -values of the first part in the second term of equation (4.17) are split into a lower order bounded part calculated with upwind differencing and a higher order scheme.

The higher order scheme is indicated with the subscript *ho* and will be specified precisely later on. The α -values within the compression term is also provided with a suitable interpolation scheme, denoted with the subscript *ic*, which is also clarified later. This leads to the following expanded formulation

$$\alpha_c^n = \alpha_c^o - \frac{\Delta t}{|V|} \sum_f [\phi_f^o \alpha_{f,ud}^n + \phi_f^o (\alpha_{f,ho}^n - \alpha_{f,ud}^n) + \phi_r^o \alpha_{f,ic}^n (1 - \alpha_{f,ic}^n)]. \quad (4.18)$$

Based on this formulation, lets define

$$\begin{aligned} \Phi_{\alpha,bd}^o &:= \phi_f^o \alpha_{f,ud}^n, \\ \Phi_{\alpha,corr}^o &:= \phi_f^o (\alpha_{f,ho}^n - \alpha_{f,ud}^n) + \phi_r^o \alpha_{f,ic}^n (1 - \alpha_{f,ic}^n). \end{aligned} \quad (4.19)$$

Here, the subscript *bd* notifies the bounded flux, in our case obtained with an upwind differencing scheme, and the subscript *corr* of the second mentioned flux indicates the anti-diffusive correction flux. This one is defined as the difference between the α -values calculated with a higher order scheme and the face value obtained by upwind interpolation, plus the compression flux. Without the surface compression approach, the definition of the bounded and correction flux would be the same, just without the compression term. Note, the correction flux is negative in the case of ingoing flow into a control volume and analogously positive, if the fluid is leaving the cell. With the next step, the FCT method introduces a limiter λ in equation (4.18), which is locally limiting the fluxes to maintain a bounded solution. Then, equation (4.18) can be rewritten as

$$\alpha_c^n = \alpha_c^o - \frac{\Delta t}{|V|} \sum_f (\Phi_{\alpha,bd}^o + \lambda \Phi_{\alpha,corr}^o). \quad (4.20)$$

In the following, we will describe how we calculate the flux limiter λ .

Determination of the Limiter with MULES

The limiter formulation we use is MULES, which is short for multidimensional universal limiter with explicit solution and was introduced by WELLER in [101]. We mention this method here, since it is originally used in the OpenFOAM solver implementations, we will discuss in detail in Chapter 4.4. As already mentioned above, the new α^n -value shall be calculated as convex combination of an upwind flux and a higher order correction flux, but with an explicit treatment of the α -values. To ensure that no new extrema are introduced into the solution after applying the anti-diffusive fluxes, so-called limiters weight these fluxes. Due to the name of the method we use, these limiters are denoted as λ_M in the following section. Therefore,

first the local minimal and maximal α -values are calculated for the respective time

$$\begin{aligned}\alpha_{\min} &= \min(\alpha_N^o, \alpha_c^o), \\ \alpha_{\max} &= \max(\alpha_N^o, \alpha_c^o),\end{aligned}$$

where $\alpha_c^o = \alpha_P^o$ and α_N^o are all the phase fraction values of the neighbors by face for the respective control volume. Since the flux-corrected approach aims to bound the α -values to the interval $[a, b] = [0, 1]$, the local extrema are again corrected by the limits

$$\begin{aligned}\alpha_{\min}^a &= \max(a, \alpha_{\min}), \\ \alpha_{\max}^b &= \min(b, \alpha_{\max}).\end{aligned}$$

For all new centered phase fraction values the following condition must be valid now

$$\alpha_{\min}^a \leq \alpha_c^n \leq \alpha_{\max}^b.$$

Inserting equation (4.20) in this condition results in

$$\alpha_{\min}^a \stackrel{(1)}{\leq} \alpha_c^o - \frac{\Delta t}{|V|} \sum_f (\Phi_{\alpha, bd}^o + \lambda_M \Phi_{\alpha, corr}^o) \stackrel{(2)}{\leq} \alpha_{\max}^b.$$

Lets consider the lower (1) and upper (2) limit separately

$$\begin{aligned}(1) &\Leftrightarrow \frac{|V|}{\Delta t} (\alpha_{\min}^a - \alpha_c^o) + \sum_f \Phi_{\alpha, bd}^o \leq - \sum_f \lambda_M \Phi_{\alpha, corr}^o, \\ (2) &\Leftrightarrow - \sum_f \lambda_M \Phi_{\alpha, corr}^o \leq \frac{|V|}{\Delta t} (\alpha_{\max}^b - \alpha_c^o) + \sum_f \Phi_{\alpha, bd}^o.\end{aligned}$$

Then, the upper and lower bounds for the face-based limited correction fluxes can already be calculated. They are defined as

$$\begin{aligned}Q^+ &:= \frac{|V|}{\Delta t} (\alpha_{\min}^a - \alpha_c^o) + \sum_f \Phi_{\alpha, bd}^o, \\ Q^- &:= \frac{|V|}{\Delta t} (\alpha_{\max}^b - \alpha_c^o) + \sum_f \Phi_{\alpha, bd}^o.\end{aligned}$$

Q^+ and Q^- are cell-centered values. Now, they have to fulfill the condition

$$-Q^+ \leq \sum_f \lambda_M \Phi_{\alpha,corr}^o \leq Q^-. \quad (4.21)$$

This condition guarantees a bounded solution of the transported phase fraction field α^n . The next step is to calculate the inflows and outflows for each CV. Therefore, we consider incoming and outgoing fluxes of a cell separately. We define the sum of all incoming and outgoing fluxes in a cell as

$$\begin{aligned} P^+ &:= - \sum_f \min(0, \Phi_{\alpha,corr}^o) && \text{(incoming fluxes),} \\ P^- &:= \sum_f \max(0, \Phi_{\alpha,corr}^o) && \text{(outgoing fluxes).} \end{aligned}$$

Analogously we define the sum of the limited incoming and outgoing fluxes as

$$\begin{aligned} S^+ &:= - \sum_f \min(0, \lambda_M \Phi_{\alpha,corr}^o) && \text{(incoming limited fluxes),} \\ S^- &:= \sum_f \max(0, \lambda_M \Phi_{\alpha,corr}^o) && \text{(outgoing limited fluxes).} \end{aligned}$$

In order to fulfill equation (4.21), which means not to create a new maximum or minimum, the difference of the sums of all limited outgoing fluxes and limited incoming fluxes have to be between $-Q^+$ and Q^-

$$-Q^+ \leq S^- - S^+ \leq Q^-.$$

If we assume R^+ and R^- are cell-centered limiters and replace S^+ when considering the lower bound $-Q^+$ and analogously replace S^- when considering the upper bound Q^- , we obtain the two relations

$$\begin{aligned} S^- - S^+ &\geq S^- - R^+ P^+ \geq -Q^+, \\ Q^- &\geq S^- - S^+ \geq R^- P^- - S^+. \end{aligned}$$

So these relations are true, if we calculate R^\pm by

$$\begin{aligned} R^+ &:= \max \left[0, \min \left(1, \frac{Q^+ + S^-}{P^+} \right) \right], \\ R^- &:= \max \left[0, \min \left(1, \frac{Q^- + S^+}{P^-} \right) \right]. \end{aligned}$$

Then the MULES limiter is calculated in the way, that this face-based limiter needs to prevent the actual CV from falling under the minimum value and the neighboring cells from rising above the maximum value. This gives us a division of the face-based limiters R^\pm in values from the actual cell and the neighboring cells, resulting in

$$\lambda_M = \begin{cases} \min[\lambda_M, \min(R_c^-, R_N^+)] & \text{for } \Phi_{\alpha,corr}^o \geq 0, \\ \min[\lambda_M, \min(R_c^+, R_N^-)] & \text{for } \Phi_{\alpha,corr}^o < 0. \end{cases}$$

The calculation of the limiters is iterative and starts with $\lambda_M = 1$ for all faces. After a predefined number of iterations ($N_{\alpha-corr}$) the algorithm stops and uses these limiters for the correction, hence for limiting the anti-diffusive fluxes. Therewith, the new cell-centered phase fraction field values α_c^n are calculated for every control volume of our domain. Since fractional α -values only appear in the vicinity of the interface, the limiter λ_M is only here of importance. Away from the transition region the value is equal to zero [24]. This results in a complete upwind scheme for the advected phase fraction field everywhere outside the interface region. Since the MULES is fundamentally explicit, a strict Courant number limit is necessary as already mentioned in Section 4.1.2.

Appropriate Interpolation Schemes

Now we still have to take a closer look at the utilized interpolation schemes, we already mentioned in equation (4.18). Here, we distinguish between three interpolation schemes, whereby they refer in each case to the entire convective term. This means, that we use the mentioned scheme to calculate the respective face value for the whole flux $(\phi\alpha)_f$. The scheme for the first convective term $\phi_f^o \alpha_{f,ud}^n$ is clear, here a classical upwind interpolation is used as described in Section 4.1.1. For the higher order scheme, indicated by $\phi_f^o \alpha_{f,ho}^n$, a van Leer scheme is used and for the compression part $\phi_r^o \alpha_{f,ic}^n (1 - \alpha_{f,ic}^n)$ the so-called interfaceCompression scheme is applied. Now we will specify the last two in more detail.

To calculate the face value of a representative flux field ψ_f , a mixture of the introduced central and upwind interpolation schemes can be chosen, what we denoted as blended differencing, see equation (4.1). Then we obtain

$$\psi_f = (1 - \lambda_\psi) \psi_{f,ud} + \lambda_\psi \psi_{f,cd}. \quad (4.22)$$

The mentioned discretization schemes differ in the choice of the limiter function. We choose $\lambda_\psi = \lambda_{vl}$ for inducing the van Leer scheme introduced in [98]. The symmetric

van Leer limiter function is defined as

$$\lambda_{vl}(r) := \frac{r + |r|}{1 + |r|}, \quad \text{with } \lim_{r \rightarrow \infty} \lambda_{vl}(r) = 2.$$

Here, the argument r of the flux limiter function represents the ratio of gradients on the solution mesh and, according to [48], can be calculated by

$$r = 2 \frac{\mathbf{d} \cdot (\nabla \phi)_P}{\mathbf{d} \cdot (\nabla \phi)_f} - 1 = 2 \frac{\mathbf{d} \cdot (\nabla \phi)_P}{\phi_N - \phi_P} - 1,$$

with corresponding flux ϕ and \mathbf{d} as defined in Section 4.1.1. The BD scheme together with a vanLeer limiter belongs to the total variation diminishing (TVD) schemes, since it is well known that the total variation of the solution decreases monotonically in this case. Furthermore, it is second order accurate and bounded. The representation with cell-centered values then applies [20]

$$\psi_f^n = \psi_P^n + \frac{\psi_N^n - \psi_P^n}{2} [1 - \zeta(\phi_f)(1 - \lambda_{vl})], \quad (4.23)$$

where the step function $\zeta(\phi_f)$ is defined by

$$\zeta(\phi_f) = \begin{cases} 1 & \text{for } \phi_f \geq 0, \\ -1 & \text{for } \phi_f < 0. \end{cases}$$

The second discretization scheme is the so-called interfaceCompression scheme, specially adapted to the compression term, which we indicate with $\lambda_\psi = \lambda_{ic}$. Here, the limiter function is calculated by

$$\lambda_{ic}(\psi_P, \psi_N) = \min \left(\max \left\{ 1 - \max \left[\sqrt{1 - 4\psi_P(1 - \psi_P)}, \sqrt{1 - 4\psi_N(1 - \psi_N)} \right], 0 \right\}, 1 \right),$$

see [53], which is also a bounded limiter scheme. Inserting λ_{ic} into equation (4.23), again leads to the desired interpolation scheme.

In summary, the numerical solution of the α -transport equation includes several components or steps that aim to provide a stable, accurate and bounded solution of the new cell-centered phase fraction value α_c^n . The smearing out of the interface is prevented by the surface compression approach and the boundedness is ensured by the numerical solution with MULES. In our case, the α -field has to be bounded between zero and one, so this approach seems to be recommended and works very well. Furthermore, MULES is constructed to additionally treat source terms on the right hand side of the α -transport equation, if this is required. For the calculation

of the α -sensitivity, this approach is not recommended, since the $\delta\alpha$ -values do not have to be bounded by two predefined values. How this changes the solution procedure will be treated in Section 4.3.

Besides the flux-corrected approach, there are very efficient other methods to solve the advection equation. For example, the piecewise linear interface construction (PLIC) by VAN WACHEM and SCHOUTEN [99] is based on a geometrical reconstruction of the interface, which, however, is associated with a higher computational effort and is impractical for unstructured or arbitrary meshes. Also the compressive interface capturing scheme for arbitrary meshes (CICSAM) by UBBINK [94] is a good option, which is a high resolution differencing scheme based on the idea of approximating the donor-acceptor flux and using a normalized variable diagram (NVD) [34]. The method is completely mass conservative, but it has to be mentioned, that the Courant number limit has to be very strict [34], which can lead to significantly higher computational costs.

4.2.2 Numerical Solution of the Navier-Stokes Equations

The numerical solution of the incompressible Navier-Stokes Equations is complicated by the absence of a separate pressure equation [28]. A widely used approach are the so-called pressure-correction methods, which are utilized to overcome this difficulty. Here, a temporary velocity field is calculated in a first step, ignoring the pressure gradient and further source terms. Then, the solution is projected into a space of divergence-free velocity fields [93]. The idea behind this approach is, that the pressure-velocity coupling is much stronger than the non-linear convective coupling for small time steps [58]. Therefore, only the terms containing the velocity u at the actual time step are considered, the velocity at the old time step, the pressure term, gravitational forces and the surface tension part are neglected in a first step. Then, a pressure equation is solved to iteratively calculate the new pressure and correct the velocity field. Since velocity and pressure equation are solved one after the other, we also speak of a segregated pressure-based approach. Next, we will introduce two representatives of such pressure-velocity coupling methods and their fields of application. The description is mainly based on results in [28] and [47].

Pressure-Velocity Coupling

One possible method to calculate the numerical solution of the momentum equation is the semi-implicit method for pressure linked equations (SIMPLE), introduced

by Patankar and Spalding in 1972 [66]. The SIMPLE algorithm is an iterative method and works fine for steady state simulations [28]. If transient problems are considered, the pressure-implicit with splitting of operators (PISO) algorithm [46], an extension of the SIMPLE algorithm, is more favorable. Here, an additional pressure correction equation is solved compared to SIMPLE. Since we want to solve transient problems, we will focus on PISO in the following, but having in mind, that we can simplify the procedure to SIMPLE with neglecting the pressure correction. Within the implementations, this can be controlled by an additional parameter. The advantage of using an implicit procedure here is, that time step constraints are often not necessary and therefore these procedures are more efficient [28]. However, since we mainly consider time-dependent problems, we have to make sure that the transient solution also satisfies the nonlinear equations at each time step and therefore the time step must become sufficiently small.

We will now introduce the PISO algorithm first in a general manner, and then apply it to our specific case. Remember, n is the indicator of the time step. Within the pressure-velocity loop we now introduce the superscript m , which counts the number of iterations. They are denoted as outer iterations, while an inner iteration means here solving a linear system with fixed coefficients in every outer iteration step. Then, $m = 0$ corresponds to the old time level, for example of the velocity u^o , and after finishing the pressure-velocity loop we obtain the velocity $u^m = u^n$ at the new time. In general, the discretized and linearized momentum equation from equation (4.15) can also be written as follows

$$\mathcal{A}u^m = \mathcal{H} - \nabla p^m, \quad (4.24)$$

where \mathcal{A} includes the coefficients of the current velocity components and \mathcal{H} contains all other terms, also those which can be written explicitly as functions of u^o from the former time step. Then we can rearrange the above equation as follows

$$u^m = \frac{\mathcal{H}}{\mathcal{A}} - \frac{1}{\mathcal{A}} \nabla p^m. \quad (4.25)$$

From this equation we calculate a momentum predictor, which means that we calculate a velocity field without the influence of the pressure by

$$u^* = \frac{\mathcal{H}}{\mathcal{A}}. \quad (4.26)$$

We correct this predicted velocity u^* iteratively with the help of the pressure equation. Therefore, our velocity field has to fulfill also the continuity equation. When

applying the divergence to equation (4.25), we obtain

$$\nabla \cdot u^m = \nabla \cdot u^* - \nabla \cdot \left(\frac{1}{\mathcal{A}} \nabla p^m \right).$$

Since we assume a divergence free velocity field, hence $\nabla \cdot u = 0$ at the end of the iteration loop, we can set $\nabla \cdot u^m = 0$ and the equation simplifies to

$$\nabla \cdot u^* = \nabla \cdot \left(\frac{1}{\mathcal{A}} \nabla p^m \right). \quad (4.27)$$

This is a Poisson equation for the pressure, called pressure equation, and provides a velocity fulfilling the continuity equation. We have to solve this equation for the pressure p^m , while we take the predicted velocity u^* , calculated in a step before. Afterwards, the velocity field also needs to be corrected, which is done using equation (4.25) in an explicit manner. Then, the PISO loop iterates over solving the pressure equation (4.27) and correcting the velocity field with equation (4.25) until a predefined number of correction steps N_{corr} is reached.

Starting from equation (4.15), we will now formulate \mathcal{A} and \mathcal{H} for our problem. For the calculation of the momentum predictor u^* , we will first consider only the temporal, viscous and advective terms; the pressure, surface tension and gravity terms are neglected at the moment. Hence, we start with the following equation

$$\frac{\rho_P^n u_P^n - \rho_P^o u_P^o}{\Delta t} |V| + \sum_f \rho_f^n \phi_f u_f^n = \sum_f \mu_f^n S_f \frac{u_N^n - u_P^n}{|\mathbf{d}|} + \sum_f \mu_f^n S_f \left(\frac{u_N^o - u_P^o}{|\mathbf{d}|} \right)^\top. \quad (4.28)$$

Note, as we have seen in Section 2.4.1, the density and viscosity are dependent on the phase fraction field α within the two-phase flow setting. Since the α -transport equation is solved for the new time step before solving the momentum equation in the numerical procedure, density and viscosity can be already updated with the α -value from the new time step, which is actually done in our numerical investigations. Another point to mention is, that the nonlinear differential equations were linearized before discretization, cf. (4.14). Then we adapt the indices and define

$$A := \frac{\rho_P^n u_P^m}{\Delta t} |V| + \sum_f \rho_f^n \phi_f u_f^m - \sum_f \mu_f^n S_f \frac{u_N^m - u_P^m}{|\mathbf{d}|}. \quad (4.29)$$

In A we find all terms concerning the actual time step. Remember, that A can also be divided in coefficients of the owner and the neighboring cells, as seen in equation

(4.16). We set

$$A = a_P u_P^m + \sum_N a_N u_N^m. \quad (4.30)$$

How a_P and a_N are exactly defined is described in [24]. For a better understanding of the following steps we will note, that a_P and a_N are each multiplied with the reciprocal of $|V|$, so we can omit this factor in the following. We now want to rearrange equation (4.28) to the actual velocity at the current cell center u_P^m . Then we obtain

$$\begin{aligned} a_P u_P^m &= \frac{\rho_P^o u_P^o}{\Delta t} + \frac{1}{|V|} \sum_f \mu_f^n S_f \left(\frac{u_N^o - u_P^o}{|\mathbf{d}|} \right)^\top - \sum_N a_N u_N^m \\ \Leftrightarrow u_P^m &= \frac{H}{a_P}, \end{aligned} \quad (4.31)$$

with

$$H := \frac{\rho_P^o u_P^o}{\Delta t} + \frac{1}{|V|} \sum_f \mu_f^n S_f \left(\frac{u_N^o - u_P^o}{|\mathbf{d}|} \right)^\top - \sum_N a_N u_N^m. \quad (4.32)$$

H contains the terms calculated at the old time as well as the convective and diffusion terms evaluated at all nearest neighboring cells. Therefore, H can already be calculated explicitly. The notation of terms with A and H is auxiliary to match with the OpenFOAM implementation of the PISO algorithm, see Appendix B.2 for more details.

The next step is to include the gravitational and the surface tension term on the right hand side of equation (4.31), to calculate the predicted velocity field u_P^* . This results in

$$u_P^* = \frac{H}{a_P} + \frac{(\rho^n g)_P}{a_P} + \frac{(\sigma \kappa^n)_P (\nabla \alpha^n)_P}{a_P}.$$

For numerical stability reasons, the flux of u_P^* is calculated from this, i.e. the terms are interpolated to the cell faces and multiplied with the face areas. Then we obtain

$$\phi_f^* = S_f \cdot \left(\frac{H}{a_P} \right)_f + \left(\frac{1}{a_P} \right)_f S_f \cdot (\rho^n g)_f + \left(\frac{1}{a_P} \right)_f (\sigma \kappa^n)_f S_f \frac{\alpha_N^n - \alpha_P^n}{|\mathbf{d}|}. \quad (4.33)$$

With this flux, the pressure equation (4.27) is solved to obtain p^m

$$\nabla \cdot \phi_f^* = \nabla \cdot \left[\left(\frac{1}{a_P} \right)_f S_f \cdot (\nabla p^m)_f \right]. \quad (4.34)$$

The face value of the pressure gradient is evaluated as in equation (4.15) by the sum of the pressure on the cell faces. Equation (4.34) can be solved explicitly for p^m , using a numerical method appropriate for elliptic equations. The resulting linear system can be solved efficiently for example with the preconditioned conjugate gradient (PCG) method, see [77, 28] for more details. These calculations are called the inner iterations, while iterating over the coupling terms, we call the PISO loop, is denoted as outer iteration. The pressure serves in the case of incompressible flows as an operator, which projects the not divergence-free velocity onto a divergence-free velocity field, hence it ensures continuity. When the PISO loop is finished, the velocity is corrected with the right pressure gradient and a predicted velocity field as in equation (4.25) by

$$\phi_f^m = \phi_f^* - \left(\frac{1}{a_P} \right)_f S_f \cdot (\nabla p^m)_f. \quad (4.35)$$

Altogether, we solved the discretized problem by constructing a predicted velocity field and then correcting it using PISO to time advance the pressure and velocity fields. We call equation (4.26) the predictor step and equation (4.27) the corrector step.

To match the description we chose to describe the general PISO algorithm, solving the predictor equation (4.26) and the pressure equation (4.27), we end up with the following formulation for \mathcal{A} and \mathcal{H}

$$\begin{aligned} \mathcal{A} &= a_P, \\ \mathcal{H} &= H + (\rho^n g)_P + (\sigma \kappa^n)_P (\nabla \alpha^n)_P. \end{aligned}$$

Remember, \mathcal{A} includes the coefficients of the current velocity components, but in this case only for the considered control volume. Hence \mathcal{A} is a diagonal matrix. All other terms are collected in \mathcal{H} , also those which can be written explicitly as functions of u^o from the former time step and terms calculated with values from neighboring cells. Hence, \mathcal{H} is a matrix also with off-diagonal entries.

For the sake of completeness we note, that we will mention an algorithm called PIMPLE when we present the numerical results with OpenFOAM in Chapter 5.4.

PIMPLE is a combination of SIMPLE and PISO. Depending on the setting of a certain value, the corresponding algorithm is selected. Since we consider mainly transient problems, we will always use the PISO algorithm described above.

4.3 Discretization and Numerical Solution of the Sensitivity Equations

For the implementation of the sensitivity equations we follow the *Optimize-then-Discretize* approach. Here, the continuous sensitivity equations are discretized with suitable numerical tools, instead of deriving the discrete problem. Although this leads to inexact derivatives of the sensitivity equations, we can reach a good consistency with the derivatives of the continuous problem with the appropriate choice of discretization procedures. In our case, a simultaneous solving of state and sensitivity equations is pursued. This means, in every time step, the sensitivity equations are solved right after calculating the state variables. This will ensure that discrete sensitivities are consistent with the continuous ones. Furthermore, we will use similar discretization and interpolation schemes for the various terms. However, at some points there are differences between solving the state equations and the sensitivity equations numerically, which we will discuss in more detail below.

Analogously to the discretized primal equation, the discretized sensitivity equation of the α -transport equation has the following form

$$0 = \frac{\delta\alpha_c^n - \delta\alpha_c^o}{\Delta t} + \frac{1}{|V|} \sum_f S_f \cdot \delta u_f^o \alpha_f^n + \frac{1}{|V|} \sum_f S_f \cdot u_f^o \delta\alpha_f^n. \quad (4.36)$$

Since we use the velocity field of the old time step in the discretized primal advection equation, see (4.11), we here also use the old velocity sensitivity δu^o and the old velocity u . The state variable α is used from the new time step. After rearranging this equation, we get an implicit formulation for $\delta\alpha_c$

$$\delta\alpha_c^n = \Delta t \left[\frac{\delta\alpha_c^o}{\Delta t} - \frac{1}{|V|} \sum_f S_f \cdot \delta u_f^o \alpha_f^n - \frac{1}{|V|} \sum_f S_f \cdot u_f^o \delta\alpha_f^n \right] \quad (4.37)$$

$$= \delta\alpha_c^o - \Delta t \left[\frac{1}{|V|} \sum_f S_f \cdot \delta u_f^o \alpha_f^n - \frac{1}{|V|} \sum_f S_f \cdot u_f^o \delta\alpha_f^n \right]. \quad (4.38)$$

The numerical solution of this equation differs from the solution of the primal α -transport equation. Here, we do not use the flux-corrected transport algorithm MULES, we introduced in Section 4.2.1. The reason is, that the sensitivities $\delta\alpha$ do not need to be corrected to that effect, that they have to be bounded between 0 and 1. Quite the opposite is the case, we expect to get high values for $\delta\alpha$ at the interface. Instead, the discretized $\delta\alpha$ -transport equation is solved straightforward with an upwind scheme and the symmetric Gauss-Seidel algorithm as smoother. Details to the Gauss-Seidel algorithm can be found for example in [77], Chapter 4.

For the solution of the differentiated momentum and continuity equation, we again choose the PISO algorithm, since we also want to calculate two variables, the sensitivity of the velocity δu and the sensitivity of the pressure δp . Here, we have to be particularly careful with the surface tension term. A poor estimate of the curvature, and hence the surface tension term, can often cause unphysical velocities around the interface [7]. This in turn leads to very high δu values at the interface and a convergence to the difference quotient is questionable.

The PISO method is simultaneously used as for the primal equations due to consistency reasons and can be modified with a few steps to match the sensitivity equations. The discretized version of the sensitivity equations of momentum and continuity can be derived from the conservative form, see the sensitivity system (3.87). In this formulation, the control term originating from the sensitivity approach is added as source term on the right hand side. Simultaneously to the Laplacian term within the primal equation, we use for the discretized momentum sensitivity equation the following representation

$$\alpha\Delta u = \nabla \cdot [\alpha(\nabla u + \nabla u^\top)] = \nabla \cdot (\alpha\nabla u) + \nabla \cdot (\alpha\nabla u^\top).$$

Then, the integral form of the momentum and continuity sensitivity equations is

$$\begin{aligned} & \int_V \rho' \partial_t(\delta\alpha u) dv + \int_V \partial_t(\rho\delta u) dv + \int_{\partial V} (\rho\delta u \otimes u) \cdot \nu ds + \int_{\partial V} (\rho u \otimes \delta u) \cdot \nu ds \\ & + \int_{\partial V} \rho' (\delta\alpha u \otimes u) \cdot \nu ds - \int_{\partial V} (\mu\nabla\delta u) \cdot \nu ds - \int_{\partial V} (\mu\nabla\delta u^\top) \cdot \nu ds \\ & - \int_{\partial V} (\mu'\delta\alpha\nabla u) \cdot \nu ds - \int_{\partial V} (\mu'\delta\alpha\nabla u^\top) \cdot \nu ds \\ & = - \int_V \nabla\delta p dv + \int_V \rho'\delta\alpha g dv - \int_{\partial V} (\sigma\nu'_\Gamma\delta\alpha\nabla\alpha) \cdot \nu ds + \int_V \sigma\kappa\nabla\delta\alpha dv \end{aligned}$$

$$\begin{aligned}
 & + \int_{\partial V} (\alpha \nabla u) \cdot \nu \, ds + \int_{\partial V} (\alpha \nabla u^\top) \cdot \nu \, ds, \\
 & \int_V \nabla \cdot \delta u \, dv = 0.
 \end{aligned}$$

Here we used for the first curvature term the fact, that the curvature is the negative divergence of the interface normal, see equation (3.84). Moreover, the Laplacian within the diffusion term was replaced by the deformation tensor within this representation. With the discrete operators presented in Section 4.1.3 we then obtain

$$\begin{aligned}
 & \frac{\rho'_P(\delta\alpha_P^n u_P^n - \delta\alpha_P^o u_P^o)}{\Delta t} |V| + \frac{\rho_P^n \delta u_P^n - \rho_P^o \delta u_P^o}{\Delta t} |V| + \sum_f \rho'_f \delta\alpha_f^n \phi_f u_f^n \\
 & + \sum_f \rho_f^n S_f \cdot \delta u_f^n u_f^n + \sum_f \rho_f^n S_f \cdot u_f^n \delta u_f^n - \sum_f \mu'_f S_f \cdot \delta\alpha_f^n (\nabla u^n)_f \\
 & - \sum_f \mu_f^n S_f \cdot (\nabla \delta u^n)_f - \sum_f \mu'_f S_f \cdot \delta\alpha_f^n (\nabla u^n)_f^\top - \sum_f \mu_f^n S_f \cdot (\nabla \delta u^o)_f^\top \\
 & + (\nabla \delta p^n)_P |V| - (\rho'g)_P \delta\alpha_P^n |V| + \sum_f (\sigma \nu_f^\top \delta\alpha^n)_f S_f \cdot (\nabla \alpha^n)_f \\
 & - (\sigma \kappa^n)_P (\nabla \delta \alpha^n)_P |V| - \sum_f \alpha_f^n S_f (\nabla u^n)_f - \sum_f \alpha_f^n S_f (\nabla u^n)_f^\top = 0, \\
 & \sum_f S_f \cdot \delta u_f^n = 0.
 \end{aligned}$$

Here we can see that the discrete version of the sensitivity equations are similar to the primal ones, with some additional advection terms. Since we formulate all additional terms in an explicit manner, the main procedure of PISO can be transferred and the additional terms can be treated as source terms. So simultaneously, the first step is to define only the terms containing the velocity sensitivity δu in A . We obtain

$$\begin{aligned}
 A := & \frac{\rho_P^n \delta u_P^n}{\Delta t} |V| + \sum_f \rho_f^n S_f \cdot \delta u_f^n u_f^n + \sum_f \rho_f^n S_f \cdot u_f^n \delta u_f^n \\
 & - \sum_f \mu_f^n S_f \cdot (\nabla \delta u^n)_f.
 \end{aligned} \tag{4.39}$$

All other terms concerning the primal temporal, convective and diffusion terms are collected in H . Here, also the control term enters the expression, since this also results from the diffusion term. Furthermore, the neighboring terms stemming from splitting A into coefficients of the owner and neighboring cells as in (4.30), are added

to H . This results in

$$\begin{aligned}
 H := & \frac{\rho_P^o \delta u_P^o}{\Delta t} |V| - \frac{\rho'_P (\delta \alpha_P^n u_P^n - \delta \alpha_P^o u_P^o)}{\Delta t} |V| - \sum_f \rho'_f \delta \alpha_f^n \phi u_f^n \\
 & + \sum_f \mu'_f S_f \cdot \delta \alpha_f^n (\nabla u^n)_f + \sum_f \mu'_f S_f \cdot \delta \alpha_f^n (\nabla u^n)_f^\top - \sum_f \mu_f^n S_f \cdot (\nabla \delta u^o)_f^\top \\
 & + \sum_f \alpha_f^n S_f (\nabla u^n)_f + \sum_f \alpha_f^n S_f (\nabla u^n)_f^\top - \sum_N a_N u_N^m.
 \end{aligned} \tag{4.40}$$

To perform PISO, the source terms are again added to H , which leads to

$$\mathcal{A} := a_P, \tag{4.41}$$

$$\mathcal{H} := H + (\rho' g)_P \delta \alpha_P^n - \frac{1}{|V|} \sum_f (\sigma \nu'_f \delta \alpha^n)_f S_f \cdot (\nabla \alpha^n)_f + (\sigma \kappa^n)_P (\nabla \delta \alpha^n)_P. \tag{4.42}$$

4.4 Software

Solving the governing equations, in particular the Navier-Stokes equations, requires a careful and well elaborated numerical approach for correct and reliable simulation results. For this purpose we use the free, open source C++ program library OpenFOAM, which is implemented for the numerical simulation of continuum mechanical transport problems. Besides the standard use as CFD tool for the solution of flow problems, other physical areas are also covered such as structural mechanics, electromagnetism, combustion and heat conduction. OpenFOAM, where FOAM stands for **F**ield **O**peration **A**nd **M**anipulation, offers more than hundred solvers and numerous custom extensions exist. For the discretization of the PDEs three different approaches are available, which are the Finite Volume Method, the Finite Element Method and the Finite Difference Method. Turbulent flow behavior can be simulated by means of Large Eddy Simulation, Reynolds Averaged Navier Stokes modeling or Direct Numerical Simulation (DNS).

OpenFOAM was originally developed and implemented by Henry Weller and Hrvoje Jasak at the Imperial College in London, see e.g., [47]. The open source character and a pretty active and smart developer community in the scientific world as well as in the industry are very beneficial. Just as helpful are the various inherent features like automatic dimension control and strong tools for pre- and post-processing. Meanwhile, there exist three development branches of Open-

FOAM with slightly different core areas. All investigations and implementations in this work are based on the OpenFOAM-6 version from the OpenFOAM Foundation.

In this thesis we are especially interested in the solver family `interFoam`, based upon the algebraic volume of fluid method. Associated solvers are designed for two-phase flow of incompressible fluids and conduce to the capturing and handling of a moving contact line on a fixed mesh. An extension to the multiphase case is available. First ideas and implementations of this solver family go back to Onno Ubbink in 1997 [94] and were extended by Henrik Rusche in 2002 [76]. The algebraic VOF solver `interFoam` is widely used in industry and for scientific issues. Implementation details about `interFoam` can be found in Appendix B. In addition to the advantageous reasons already mentioned, the software was also chosen because of the affiliation to the collaborative research center 1194. A common software is here mandatory to cooperate and therefore, OpenFOAM was used as cross-project software platform for simulation and analysis of various wetting phenomena. As example, we refer to the recent studies about the well-known capillary rise benchmark [35].

Besides OpenFOAM, Matlab is also an an important software for this thesis. While OpenFOAM provides the numerical simulation of the flow processes, Matlab is mainly used for the optimization procedures. We use Matlab version R2019a. The connection between OpenFOAM and Matlab is mainly done via unix commands, called in `matlab` to run OpenFOAM from the bash terminal. However, Matlab is often also used the other way around for post-processing of OpenFOAM simulations or for visualizing complex correlations. This is advantageous because OpenFOAM does not have its own graphical user interface. Although ParaView is mainly used to display the simulation results, Matlab is applied if further calculations have to be made with the simulation results that go beyond OpenFOAM's own post-processing tools.

4.4.1 Simulation Characteristics with OpenFOAM

Due to numerical limitations, several adjustments of the presented model equations are necessary for the implementation, which will discussed for the OpenFOAM solver `interFoam` in this section.

Regarding the α -transport equation

The VOF approach, the interFoam solver is based on, does not exactly rely on the original VOF method by Hirt and Nichols [42], because the velocity of the liquid is not assumed to be equal to the mixed velocity [53]. With this assumption, the surface compression approach is derived, where an additional compression term supplements the α -transport equation. Within this compression term, the relative velocity u_r appears, which is not given by default, because there is only one velocity u assumed for the whole domain. The face flux velocity within the transition region is then in interFoam approximated by the term

$$(u_r)_f = \min \left(c_\alpha \cdot \left| \frac{\phi_f}{|S_f|} \right|, \left| \frac{\phi_f}{|S_f|} \right|_{\max} \right),$$

where the coefficient c_α weights the compression flux [53]. Within our numerical calculations we always set $c_\alpha = 1$. Then, we are able to calculate ϕ_r of the respective time step with

$$\phi_r = (u_r)_f \cdot \tilde{\nu}_f, \quad (4.43)$$

employing $\tilde{\nu}_f$, which is introduced below in (4.44).

Regarding the momentum equation

To solve the momentum equation we use a coupling of velocity and pressure, which does not guarantee a divergence-free velocity field all along. Therefore, we have to use the full momentum equation (2.7), with $\lambda = -(\frac{2}{3})\mu$ and $\mathbf{S} = \frac{1}{2}(\nabla u + \nabla u^\top)$. Then we obtain

$$\rho \partial_t u + \nabla \cdot (\rho u \otimes u) + \underbrace{\nabla \cdot \left(\frac{2}{3} \mu \nabla \cdot u \mathbf{I} \right) - \nabla \cdot (\mu (\nabla u + \nabla u^\top))}_{(*)} = -\nabla p + f.$$

To match the implementations in interFoam, the diffusion term is further converted in the following form

$$\begin{aligned} (*) &= \nabla \cdot \left(\frac{2}{3} \mu \nabla \cdot u \mathbf{I} \right) - \nabla \cdot (\mu \nabla u) - \nabla \cdot (\mu \nabla u^\top) \\ &= -\nabla \cdot (\mu \nabla u) - \nabla \cdot \left[\mu \left(\nabla u^\top - \frac{2}{3} \nabla \cdot u \mathbf{I} \right) \right] \\ &= -\nabla \cdot (\mu \nabla u) - \nabla \cdot \left[\mu \left(\nabla u^\top - \frac{2}{3} \text{tr}(\nabla u^\top) \mathbf{I} \right) \right]. \end{aligned}$$

In the last step we used, that for an arbitrary vector a applies

$$\nabla \cdot a = \text{tr}(\nabla a) = \text{tr}(\nabla a^\top).$$

Note, the same adaption is also applied to the control term of the momentum sensitivity equation.

Regarding the curvature

The discrete curvature was defined as the negative divergence of the face unit normal flux, which we denote with ν_f . In the implementations, a stabilization factor δ_N is used within the normalization of the phase fraction gradient. We denote the stabilized normal with $\tilde{\nu}_f$ and the stabilized curvature with $\tilde{\kappa}$. Then it holds

$$\tilde{\kappa} = -\nabla \cdot \tilde{\nu}_f, \quad \text{with} \quad \tilde{\nu}_f = \frac{(\nabla \alpha)_f}{|(\nabla \alpha)_f| + \delta_N} \cdot S_f, \quad (4.44)$$

where S_f is the face area vector again. The stabilization factor δ_N accounts for the nonuniformity of the grid, where N is the number of computational cells [20]. It is defined as

$$\delta_N = \frac{\varepsilon}{\left(\frac{\sum_N V_i}{N}\right)^{1/3}}, \quad \text{with} \quad \varepsilon = 10^{-8}.$$

Regarding the Pressure Term

Instead of the pressure p , a modified version p_m is considered, which also includes a hydrostatic term

$$p_m := p - \rho g \cdot x.$$

This formulation goes back to RUSCHE [76] and aims to enable a more efficient numerical treatment due to removing possibly steep gradients arising from hydrostatic effects. To insert the modified pressure into the momentum equation, we rearrange the equation and take the gradient

$$\begin{aligned} p &= p_m + \rho g \cdot x. \\ \nabla p &= \nabla p_m + \rho g + g \cdot x \nabla \rho. \end{aligned}$$

For the momentum equation (4.12) it then holds

$$\begin{aligned} \partial_t \rho u + \nabla \cdot (\rho u \otimes u) &= -(\nabla p_m + \rho g + g \cdot x \nabla \rho) + \nabla \cdot (\mu(\nabla u + \nabla u^\top)) + \rho g + \sigma \kappa \nabla \alpha \\ \Leftrightarrow \partial_t \rho u + \nabla \cdot (\rho u \otimes u) &= -\nabla p_m - g \cdot x \nabla \rho + \nabla \cdot (\mu(\nabla u + \nabla u^\top)) + \sigma \kappa \nabla \alpha. \end{aligned}$$

Note, that this modification does not change the numerical solution procedure we presented in Section 4.2.2.

Regarding the Contact Angle Boundary Condition

Boundary conditions are used in continuum mechanics models also to describe the hydrodynamic properties in the immediate vicinity of the three-phase contact line. In our case, the contact angle comes as boundary condition of the phase fraction field into play, regardless of whether it is a static or a dynamic one. As we have seen in Chapter 2.6, we use for example the Kistler and Hoffmann model to describe the dynamic contact angle θ_d . Within the interFoam solver family, prescribed contact angles are introduced by correcting the interface normal ν_Γ at the corresponding boundary. The corrected interface normal is denoted by $\hat{\nu}_\Gamma$ and the normal of the substrate is denoted as ν_S . Then, the correction is as follows [5]

$$\hat{\nu}_\Gamma|_S := \nu_S \cos(\theta_d) + \frac{\nu_\Gamma - \nu_S(\nu_\Gamma \cdot \nu_S)}{|\nu_\Gamma - \nu_S(\nu_\Gamma \cdot \nu_S)|} \sin(\theta_d).$$

This corrected interface normal is subsequently used to calculate new curvature values in boundary cells adjacent to the interface. And in turn, the altered curvature is taken into account when calculating the momentum equation. Inside the domain, the correction has no impact on the normal of the interface, it holds $\hat{\nu}_\Gamma|_\Omega = \nu_\Gamma$. Note, that this correction is also performed, if we have a prescribed static contact angle or a dynamic contact angle calculated with another model approach.

Another crucial factor when calculating the dynamic contact angle is the contact line velocity u_{cl} . This can be done in different ways. We decided to take the velocity from the center of the cells at the contact line, as recommended in [33]. An advantage here is, that this choice also enables non-axisymmetric simulations and can be easily implemented.

4.4.2 Implementation of Sensitivity Solver `interSensFoam`

The discretization and numerical solution procedures of the sensitivity equations of the problem were already discussed in Section 4.3 as well as some implementation details of `interFoam` right before. Now we give a short insight, how the `interSensFoam` solver is adapted to solve the primal and sensitivity equations with respect to a specific control, the liquid viscosity μ_l . The section is easier to follow if the solver is already known. Further details on the solver can be found in Appendix B.2.

As we stated before, we discretize and implement the sensitivity system in a similar way to the original PDE system. Then, the solution of the sensitivity equations occurs directly after solving the primal equations in every time step. Similar to the original `interFoam` solver, the α -transport equation and hence the $\delta\alpha$ -sensitivity equation are solved separately from the Navier-Stokes equations. The distribution of the volume fraction field is calculated ahead of the PISO algorithm, which calculates the actual velocity and pressure fields [53]. All files which are relevant for calculating $\delta\alpha$ are stored in the folder with the name VOF, concerning the volume of fluid representation of the problem. Here, the flux calculation and main solution steps are implemented within a header file called `dAlphaEqn.H`. The files for solving the sensitivities δu and δp can be found in the main folder `interSensFoam`. While the temporal, the convective and the diffusive terms of the primal momentum equation are written in a file with the name `UEqn.H`, the differentiated terms for solving δU are contained in a modified version, which is denoted as `dUEqn.H`. The calculation of the momentum predictor flux $\delta\phi^*$ and the PISO loop itself are implemented within the file `dPEqn.H`, where also the source terms are added. Depending on the control q , further terms enter the δu - δp -equation, since we also need to take the derivative of the states with respect to the control into account. In our case, we added a modified version of the term from equation (3.83) on the left hand side of the momentum sensitivity equation, since the considered control, the liquid viscosity, originates from the diffusion term of the primal momentum equation. This control term therefore enters in `dUEqn.H` as additional source term. For another kind of control, a different term would have to be used at this place.

We summarized the ingredients of the primal momentum equation and their derivatives in Table 4.1. Here, we divided the equation into the terms contained in `dUEqn.H` and those contained in `dPEqn.H`, always presented with the corresponding terms from the primal equations. Note, that the control term does not have a counterpart in the original momentum equation. The description is on a formal level, code details and OpenFOAM specific notations of the `interSensFoam`

solver can be found in Appendix B.

	Primal term	Derivative	
UEqn.H	$\partial_t(\rho u)$	$\rho' \partial_t(\delta\alpha u)$ $+ \partial_t(\rho \delta u)$	dUEqn.H
	$+ \nabla \cdot (\rho u \otimes u)$	$+ \nabla \cdot (\rho' \delta\alpha u \otimes u)$ $+ \nabla \cdot (\rho \delta u \otimes u)$ $+ \nabla \cdot (\rho u \otimes \delta u)$	
	$- \nabla \cdot (\mu \nabla u)$	$- \nabla \cdot (\mu' \delta\alpha \nabla u)$ $- \nabla \cdot (\mu \nabla \delta u)$	
	$- \nabla \cdot (\mu \nabla u^\top)$	$- \nabla \cdot (\mu' \delta\alpha \nabla u^\top)$ $- \nabla \cdot (\mu \nabla \delta u^\top)$	
	$+ \nabla \cdot (\frac{2}{3} \mu \nabla \cdot u \mathbf{I})$	$+ \nabla \cdot (\frac{2}{3} \mu' \delta\alpha \nabla \cdot u \mathbf{I})$ $+ \nabla \cdot (\frac{2}{3} \mu \nabla \cdot \delta u \mathbf{I})$	
		$- \nabla \cdot (\alpha \nabla u)$ $- \nabla \cdot (\alpha \nabla u)^\top$	
	==	==	
pEqn.H	$- \nabla p_m$	$- \nabla \delta p_m$	dPEqn.H
	$- \mathbf{g} \cdot \mathbf{x} \nabla \rho$	$- \mathbf{g} \cdot \mathbf{x} \rho' \nabla \delta\alpha$	
	$+ \sigma \kappa \nabla \alpha$	$- \sigma \nabla \cdot (\nu'_\Gamma \delta\alpha) \nabla \alpha$ $+ \sigma \kappa \nabla \delta\alpha$	

Table 4.1. Primal system and its derivatives in interSensFoam.

Now we present the main characteristics of interSensFoam and summarize the solution procedure in Algorithm 4.1 at the end of the chapter. The main steps are implemented in a file with the name interSensFoam.C. Here, also the header files mentioned above are included at appropriate places. As basis, we take algorithm 1 from [53], supplemented with sensitivity calculations and ideas from [58].

After introducing the initial fields together with appropriate boundary conditions (step 1) and calculating the time step (step 2) as well as the actual viscosity and density field (step 3) the main solution cycling starts with solving the α -transport equation in step 4. The corresponding files are alphaEqnSubCycle.H and alphaEqn.H. Here, a sub cycling within the time step is possible. That means, that the α -value is calculated for $N_{\alpha,sub}$ sub time steps. The number $N_{\alpha,sub}$ is chosen by the user, but always set equal to one in our case. A compression flux

is created and solved with MULES, we already described in detail in Section 4.2.1. Updating the α -fluxes, the curvature and the density with the new α -value also belongs to solving the α -transport equation. Back in the main script, a current viscosity field is calculated in step 5. Then the solution of the momentum and continuity equation is prepared in step 6 with the construction of \mathcal{A} and \mathcal{H} within the UEqn.H file. See Section 4.2.2 for more details. The PISO loop itself is performed in the file pEqn.H, which corresponds to step 7. The number of correction steps is defined through the user by N_{corr} . Within the PISO loop, the first step is to calculate a flux predictor ϕ^* and use this not divergence-free flux next to solve the pressure equation. The resulting intermediate pressure is taken to correct the velocity field, which can also be understood as a projection of u^* to a divergence-free velocity field. Of course, we also have to update the boundary field to agree with the required boundary conditions. Right after solving the primal state equations, the sensitivity equations are solved. Again we start with the $\delta\alpha$ -transport equation in step 8. For the calculation of $\delta\alpha$, the mentioned flux is calculated while using an upwind differencing scheme for the face-centered values. Then, the Gauss-Seidel algorithm is performed as linear equation solver. No further limiter algorithm is applied, compare to Section 4.3. In contrast, the differentiated momentum and continuity equation are solved analogously to the primal equations with the PISO algorithm. Therefore, the operators \mathcal{A} and \mathcal{H} are again constructed in step 9, before the actual PISO loops starts in step 10. Here, the same sub steps are performed as for the primal equations, the difference is in the definition of \mathcal{A} and \mathcal{H} , see Section 4.3. Then we calculated all required variables for the current time step. The new solver iteration starts again with calculating the new time step at step 2, until the predefined end time is reached, see step 11.

A section before, some solver adaptations were presented to connect the discretized equations and numerical solution procedures with the source code implemented in OpenFOAM. Some of the interFoam specific adaptations were also taken for the implementation of interSensFoam. In addition to the stabilized curvature, a modified pressure sensitivity is considered. Within the implementation, this leads to a slightly different term with the gravitational acceleration. The formal derivative, as it is then also implemented in interSensFoam, can be found in Table 4.1. Exactly as in the original interFoam implementation, the additional terms for diffusion are considered, which are supposed to represent a possibly not divergence-free velocity field within the PISO loop. Further details of implementation, for example the meaning and content of the various other header files in interSensFoam.C, can be found in a walk through the code in Appendix B.2.

Algorithm 4.1 Solution procedure of sensitivity solver interSensFoam

- 1: Set initial and boundary conditions for u, p, α and the sensitivity fields $\delta u, \delta p, \delta \alpha$
 - 2: Set time step. If time step is variable, calculate Δt with (4.4) in accordance to the CFL condition (4.3)
 - 3: Update μ and ρ with actual phase fraction field α according to (2.17) and (2.18)
 - 4: Solve α -transport equation for α_c^n
 - (a) Calculate compression flux ϕ_r with equation (4.43)
 - (b) Perform α -correction $N_{\alpha-corr}$ times
 - (i) Define or update flux $\Phi_\alpha = \Phi_{\alpha,bd} + \Phi_{\alpha,corr}$ from equation (4.19)
 - (ii) Solve equation (4.17) explicitly with MULES
 - (c) Calculate $\rho^n \phi^n$ with new α^n -value
 - (d) Calculate $\tilde{\nu}_f^n$ and the curvature $\tilde{\kappa}^n$, see equation (4.44)
 - (e) Determine density ρ^n with equation (2.17)
 - 5: Correct viscosity with equation (2.18)
 - 6: Construct \mathcal{A} and \mathcal{H} with equations (4.29) and (4.32)
 - 7: Perform PISO loop $N_{corr} + 1$ times to calculate state variables u and p
 - (a) Solve the flux predictor ϕ^* with equation (4.33)
 - (b) Solve the pressure equation with equation (4.34)
 - (c) Correct the velocity with new pressure field, see equation (4.35)
 - (d) Update boundary conditions
 - 8: Solve $\delta\alpha$ -sensitivity equation (4.37) for $\delta\alpha_c^n$ with upwind scheme
 - (a) Define flux $\delta\phi^o\alpha^n + \phi^n\delta\alpha^n$
 - (b) perform Gauss-Seidel algorithm with upwind scheme
 - (c) Calculate $\delta\phi^n$ with new $\delta\alpha^n$
 - 9: Construct \mathcal{A} and \mathcal{H} for sensitivities with equations (4.29) and (4.32)
 - 10: Perform PISO loop $N_{corr} + 1$ times to calculate sensitivities δu and δp
 - (a) Solve the flux predictor $\delta\phi^*$ as in equation (4.33)
 - (b) Solve the pressure equation as in equation (4.34)
 - (c) Correct the velocity sensitivity with new pressure sensitivity as in (4.35)
 - (d) Update boundary conditions for sensitivity fields
 - 11: Go to step (2) or finish calculation with predefined time step
-

Optimization of Doctor Blading

Now we come to an application of the described problem. We consider a wetting process that is motivated by gravure printing. Gravure printing is a comparatively complex and expensive process, but good replicability and high printing velocities are characteristic and a high printing accuracy can be achieved. It is used, for example to print stamps and banknotes, to print on special substrates such as foil or metal or even to print electronically conductive layers. In a gravure printing process, see Figure 5.1, a gravure cylinder is rotating through a reservoir filled with ink, where the engraved surface is completely wetted with the ink. Slightly above, a sharp steel band, called a doctor blade, is clamped into a holder and pressed onto the printing form at a specific angle. Through the rotating movement, the doctor blade is pulled over the surface and removes the excessive ink from the cylinder before it is printed onto a substrate. This sub process is essential for good printing results and part of every printing or coating task.

The material, shape and position of the doctor blade are mainly based on expert knowledge and often, a lot of experiments are necessary to find the right doctor blade for a particular application. Hence, we want to use various optimization techniques for optimizing the doctor blading process to achieve better printing results. This means for example to reducing print failures like air inclusions, fluid accumulations or particles which arise through material abrasion, to achieve faster printing rates and preserving a uniform lubrication film on the printing form. We aim to accomplish these goals through parameter identification, where we search feasible parameter settings in an optimal way, and by optimizing geometrical issues concerning the doctor blade itself.

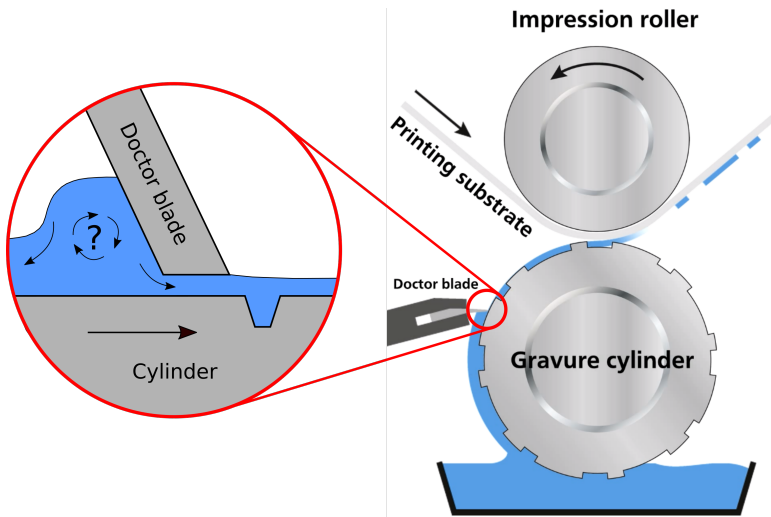


Figure 5.1. Full gravure printing process with zoom to the doctor blade.

In Section 5.1, we describe the physical aspects of such a printing process, especially of the doctor blading sub process. Then, we present the mathematical abstraction as a test case in Section 5.2. On the one hand, this section contains the geometrical setup and different initial scenarios for the fluid. On the other hand, a validation case is presented in Section 5.2, where general parameter settings and solver adjustments are discussed and the simulation results are compared to some experimental results. Moreover, central optimization questions are pointed out in Subsection 5.2.2, which provide the basis for the formulation of the objective functions and control variables of the respective optimization problems. The first class of optimization problems we consider are parameter identification problems. In these kind of problems the control of the problem is a material parameter, that we want to determine for a desired state. The second class of optimization tasks concerns the geometry of the doctor blade, in which also mesh transformations appear. In Section 5.3, the complete optimization framework with Matlab and OpenFOAM is described, where the optimization algorithm itself, a classical Gauss-Newton algorithm, is run with Matlab and a transient simulation is executed in OpenFOAM therein. Finally, in Section 5.4, we present and discuss numerical results for the optimization problems we introduced before. This is divided into the optimization of the liquid viscosity in Subsection 5.4.1, the optimization of the gap height in Subsection 5.4.2 and the optimization of the vorticity in Subsection 5.4.3.

5.1 Physics of the System

In this subsection we take closer look to the physical background of the doctor blading sub process and its influencing factors. Shape and material of the doctor blade and the gravure cylinder are crucial variables of the doctor blading process. We start with describing the gravure cylinder, which has a layered structure. Two layers of copper are electroplated onto a steel core, where the upper layer is machinable copper so that the print design can be engraved. On the engraving, the cylinder is thinly coated with chrome [51]. The doctor blade is typically a thin metal band of thickness d_{DB} and an extension in the length of the cylinder. Depending on the application, other materials are also commonly used like plastic or rubber. In our application two types of materials are considered. For most of the investigations, a simple steel doctor blade is assumed. But for the comparison with some experimental results in Section 5.2, a transparent doctor blade made of plexiglass is considered, providing multiple viewpoints of the intriguing occurrences around it. Of special importance is also the tip of the doctor blade, since the shape influences the behavior of the printing fluid. A new doctor blade has a rounded tip, which deforms in practical use, due to a grinding-in process that sharpens the doctor blade. The resulting shape depends on the contact pressure, which is the pressure the doctor blade is pressed against the printing form surface with, and the inclination angle θ by which the doctor blade is deflected from vertical [11]. In our further investigations, we use the inclination angle θ as well as the gap height d_{GH} between the tip of the doctor blade and the printing form as variable geometry parameter.

In addition, the printing ink is an important influencing factor in gravure printing. The fluid must be applied to engraved cups in the cylinder surface, so the viscosity of the ink should be very low. In literature, viscosities of $\mu = 0.05 \dots 0.2$ Pa·s are recommended for the gravure printing process. For the design of the test case and the optimization framework, the simplest case is considered, a Newtonian fluid that does not contain any particles or surfactants. Different fluids fulfill this demand, primarily simple water. Another approximately Newtonian fluid that has proven useful in experimental studies is silicone oil. Material values for density, viscosity and surface tension of these two fluids can be found in Table 5.1. A surrounding temperature of 25 °C is assumed. The material properties of the gaseous phase are also given, which corresponds to normal air for now.

Material Property	Unit	Water	Silicone Oil	Air
Density ρ	[g m ⁻³]	997	963	1
Kinematic viscosity ϑ	[m ² s ⁻¹]	$1 \cdot 10^{-6}$	$1 \cdot 10^{-4}$	$1.48 \cdot 10^{-5}$
Surface tension σ	[N m ⁻¹]	0.07	0.02	

Table 5.1. Material properties of air, water and silicone oil [11].

Note, the surface tension of water and silicon oil refers to the contact with air in Table 5.1. Furthermore, we only denoted values for the kinematic viscosity although we also use the dynamic viscosity within the theoretical and experimental investigations. The following relation applies as usual

$$\mu = \vartheta \cdot \rho.$$

Another important factor of influence is the particular contact angle, that every fluid has when it comes in contact with a solid surface. For our investigations we use a mean static contact angle θ_e of $62.09^\circ(\pm 1.58^\circ)$ for a water droplet, sitting on a gravure cylinder, see Figure 5.2. This macroscopic contact angle was measured by JULIAN SCHÄFER, a former colleague within the CRC 1194. The mean static

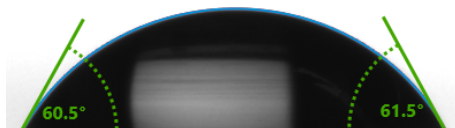


Figure 5.2. Example for static contact angle of a water droplet.

contact of silicone oil on a gravure cylinder is much smaller with around $25^\circ(\pm 5^\circ)$. Unfortunately, this value was not measured during the experimental investigations. Luckily, there are a lot of different silicone oils available, all with slightly different values for the static contact angle. So we assume that we find a silicone oil, which fulfills this declaration. Furthermore, due to the dominant inertial forces of silicone oil and the application of a dynamic contact angle model, where the contact angle has a certain parameter range in which it can be adjusted, the inexact value of this angle can be leveled out. The advancing and receding contact angles of silicone oil are also necessary for the simulation with a dynamic contact angle. These contact angles could not be measured with the existing measurement methods as well, but the choice of plus and minus ten degree is quite realistic, which leads to an advancing contact angle of $\theta_a = 35^\circ$ and a receding contact angle of $\theta_r = 15^\circ$.

5.2 The Doctor Blading Test Case

In a first step, just a small vicinity around the tip of the doctor blade is simulated. For this purpose we consider the following geometric setup, here for the 2D case.

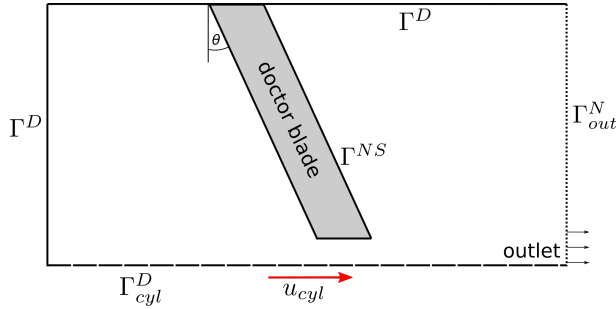


Figure 5.3. Geometry of doctor blading test case and velocity boundary conditions.

The domain has an expansion of 1 mm in x-direction, 0.5 mm in y-direction and also 1 mm in z-direction, when we consider the three dimensional case. The outline of the doctor blade also belongs to the outer boundary of the domain. The doctor blade is inclined with a specific angle θ and the lower wall has the tangential velocity u_{cyl} . When the fluid has passed through the gap between the tip of the doctor blade and the lower wall, it flows out of the domain on the right border Γ_{out}^N . There we have a Neumann boundary condition for the velocity. At the left, the upper and the lower boundary, we have Dirichlet boundary conditions for u . Though, we differ between homogeneous Dirichlet boundary conditions at Γ^D and inhomogeneous at Γ_{cyl}^D . Note that the homogeneous BCs correspond to a no-slip boundary condition, which is valid for the left border and the upper end of the domain. The inhomogeneous Dirichlet condition at the lower wall Γ_{cyl}^D represents the velocity of the rotating gravure cylinder. The slightly curved surface is thereby approximated by a plain interface. This is acceptable since the diameter of the gravure cylinder is with around 20 cm much larger than the considered extension of the domain. At the surface or outline of the doctor blade, a Navier-slip boundary conditions is applied to the velocity, which issuing this boundary is denoted by Γ^{NS} . In the following, we differentiate between two initial scenarios for the fluid.

In a first scenario, Figure 5.4, we assume an ink reservoir in front of the doctor blade, where a film has already been formed on the cylinder surface. The initial film only approximates the actual film, so a short period of time must elapse before a steady state is reached with this scenario and a uniform lubrication film is formed.

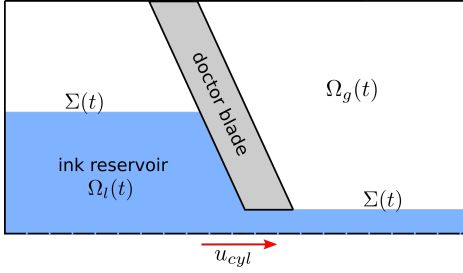


Figure 5.4. Szenario 1.

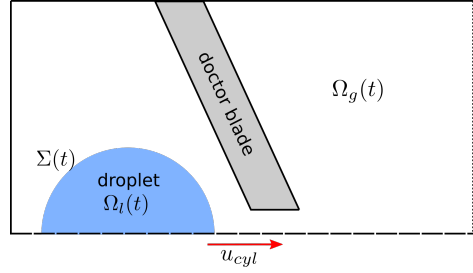


Figure 5.5. Szenario 2.

On the left side of the doctor blade is enough ink, that the steady state can be maintained for a while. In this scenario, the behavior of the fluid within the ink reservoir and the film thickness behind the doctor blade is primarily of interest. In the second scenario, Figure 5.5, a droplet is bladed with no initial film behind the doctor blade. This case is more dynamic because less fluid is available. Here, the focus is on the three phase contact between the droplet, the air and the lower wall as well as on the behavior of the fluid on the doctor blade's surface inside the gap between the tip of the doctor blade and the moving wall. This case is more interesting in three dimensions and the basis for the validation with experimental data, we present in the next section.

5.2.1 Validation with Experimental Data

To compare and validate our simulation results with experimental data, a test stand was built under the scope of the Collaborative Research Centre 1194. A detailed description of the test stand and the resulting experimental outcomes can be found in the dissertation of THORTEN BITSCH [11] and the common paper [12]. In close cooperation, an experimentally and numerically suitable setup was developed and a good agreement could be reached. Some phenomena, observed for the first time in the experiments, could be reproduced within the simulations. We describe the setup, the parameter and solver adjustments and main outcomes in the following.

Scenario 2, see Figure 5.5, forms the basis of this investigation, where a three dimensional simulation is compared to movies obtained from experiments. The inclination angle of the doctor blade is set to $\theta = 25^\circ$. That corresponds to an angle of 65° between the doctor blade and the cylinder surface, which is also used in the experiments. We neglect the deflection of the doctor blade and the resulting

5.2. The Doctor Blading Test Case

reduction of the blade angle, which can occur due to contact pressure [11]. The cylinder has a velocity of $u_{cyl} = 0.115 \text{ m s}^{-1}$. This value occurs, if a gravure cylinder of 22 cm diameter is operated with 10 rotations per minute. For the fluid, a silicone oil with a viscosity of $\vartheta = 100 \text{ cSt}$ is used. Further material parameters can be taken from Table 5.1. The domain has an outer extension of $1.1 \times 0.4 \times 0.8 \text{ cm}$. The start configuration of the simulation is shown in Figure 5.6, where the droplet is sitting right in front of the doctor blade.

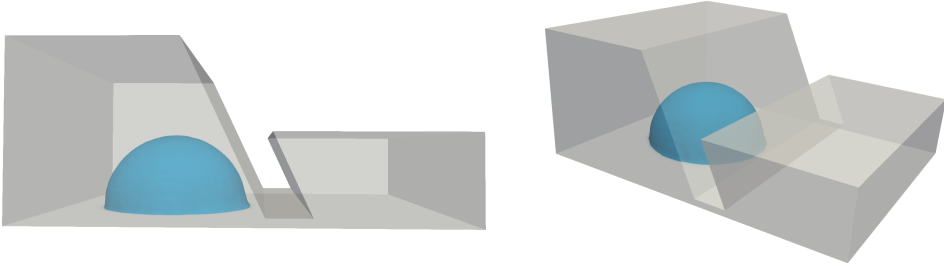


Figure 5.6. Different views of the 3D droplet simulation.

The mesh is generated with a block structure, where the domain is subdivided in five blocks, see the blue numerary in Figure 5.7. In addition, the red numerary indicates the label of the corners. The number of mesh cells is 372000 and the block structure induces hexagonal, more or less orthogonal cells, see Figure 5.8. This corresponds to 130 cells in x -direction, 55 cells in y -direction in front of the doctor blade, 5 cells within the gap and 25 cells behind the doctor blade, the latter two also in y -direction, as well as 80 cells in z -direction. The size of the cells varies slightly, since a grading of the cells was used in x - and y -direction by a factor of 2 to reduce the number of cells in areas, in which the fluid is not expected to be present.

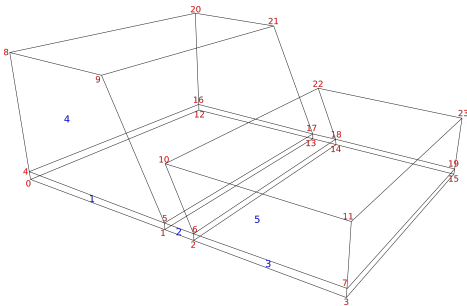


Figure 5.7. Block structure of the domain.

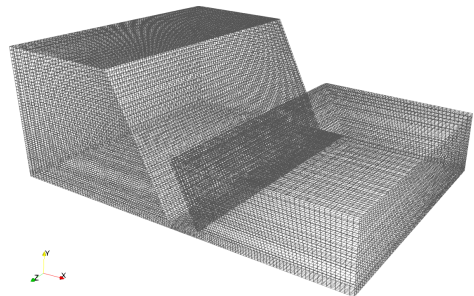


Figure 5.8. 3D mesh of the whole domain.

The simulation runs for 30 milliseconds. Then, the entire droplet volume is passed through the gap between the doctor blade and the lower wall and leaves the domain at the outflow border. The time steps itself vary, since we choose an adjustable time step size. Then the solver calculates the maximum time step depending on the maximal CFL number the user set. In our case, we selected $C_{\max} = 1$, see Table 5.2. The solution of the α -transport equation is calculated with MULES as described in Chapter 4.2.1, where the number of iteration loops calculating the limiter, denoted as `nLimiterIter`, is set as usual to 3. The PIMPLE loop, used to solve the momentum equation with velocity-pressure coupling, is run without a momentum predictor step and no nonorthogonal corrector steps, which are preferably used for heavy nonorthogonal meshes. All further relevant solver adjustments were summarized in Table 5.2. Check Sections 4.2.1 and 4.2.2 as well as Appendix B.1 for further parameter descriptions.

Parameter	Value	Notes
C_{\max}	1	Maximal Courant number for the whole simulation
$C_{\alpha,\max}$	1	Maximal Courant number for the α -transport equation
c_{α}	1	Interface compression weighting parameter
$N_{\alpha,\text{corr}}$	2	Number of α -corrector steps
$N_{\alpha,\text{sub}}$	1	Number of α -sub cycles
N_{corr}	3	Maximal number of PISO loops

Table 5.2. Numerical parameter and solver setup for the 3D simulation.

In addition, the individual discretization schemes are listed below in Table 5.3. Details to the schemes were also presented in Chapter 4.

Term	Discretization Scheme
Temporal derivative	Implicit Euler method
Convection term	Linear upwind scheme
Diffusion term	Linear interpolation with explicit nonorthogonal correction
α -transport term	Van Leer scheme and interface compression scheme
Gradient	Linear interpolation with a central differencing scheme
Surface normal gradient	Linear interpolation with explicit nonorthogonal correction

Table 5.3. Discretization schemes for the individual terms.

Besides the solver settings mentioned above, a choice of the initial fields for the phase fraction field α , denoted in OpenFOAM as `alpha.water`, for the velocity u , denoted as `U`, and for the pressure p , denoted as `p_rgh`, is necessary and of special interest to simulate this kind of wetting process. We follow the description in Section 5.1 and choose for the dynamic contact angle model an equilibrium contact angle of $\theta_e = 25^\circ$, written within the α -field, with advancing angle $\theta_a = 35^\circ$ and receding angle $\theta_r = 15^\circ$. Simultaneously, a Navier-slip boundary condition is set for the velocity at the respective borders, in OpenFOAM denoted as `patches`, to allow the dynamic behavior of the three phase contact line. Note, at the lower wall, where a tangential wall velocity is assumed, this is not necessary since we already allow and fix a velocity greater than zero in this case. The slip length for the Navier-slip condition is set to half of the cell size, which is an appropriate choice in this case. Additionally, BCs for the pressure are claimed by the solver, although this is actually determined by the other constraints. For this reason, OpenFOAM supplies the so-called `fixedFluxPressure` condition, which adjusts the pressure depending on given flux at the respective faces. A summary of the boundary conditions with OpenFOAM conforming designations is given in Table 5.4.

Patch	<code>alpha.water</code>	<code>p_rgh</code>	<code>U</code>
<code>leftWall</code>	<code>zeroGradient</code>	<code>fixedFluxPressure</code>	<code>partialSlip</code>
<code>outletWall</code>	<code>zeroGradient</code>	<code>totalPressure</code>	<code>inletOutlet</code>
<code>movingWall</code>	<code>dynamicContactAngle</code>	<code>fixedFluxPressure</code>	<code>translatingWallVelocity</code>
<code>upperWall</code>	<code>dynamicContactAngle</code>	<code>fixedFluxPressure</code>	<code>partialSlip</code>
<code>atmosphere</code>	<code>inletOutlet</code>	<code>totalPressure</code>	<code>pressureInletOutletVelocity</code>

Table 5.4. Setup of OpenFOAM boundary conditions for the doctor blading test case.

Note, the original name of the dynamic contact angle BC is `dynamicAlphaContactAngle`, which was shortened a bit within the table due to limited space.

Observations

When comparing the initial simulation setup, see Figure 5.10, with the picture of the real droplet, see Figure 5.9, some differences become apparent. On the one hand, the shape of the droplets varies. While the simulated droplet is perfectly round, the real droplet is already shifted. This comes from a longer start-up phase in the experiments. Due to mesh complexity reasons, the domain in the simulation is kept as small as possible and therefore the movement of the droplet starts right in front of the doctor blade. Furthermore, we were unable to reproduce the surface

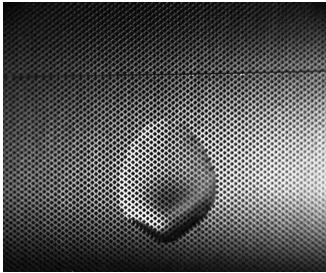


Figure 5.9. Droplet real experiment.



Figure 5.10. Droplet 3D simulation.

structure. While the real printing form surface is uniformly engraved with cups in the range of tens of micrometers, the simulated surface is plain without any kind of structure. This, of course, makes a difference when it comes to the exact behavior at the three phase contact line on the printing plate. But as this validation study focuses more on the overall the general behavior of the droplet, the interaction with the doctor blade and the film formation behind the doctor blade, we can disregard this in good conscience for the moment. Furthermore, it is not possible to resolve the exact behavior of the three phase contact line in interaction with the cups using the existing videos with a view from above. Additional experimental and numerical studies on this are advisable and were recently advanced within the CRC 1194.

Despite the described differences, we can observe a similar behavior of film formation in the simulations such as seen in the videos of the experiments. At first, this regards the general appearance of the fluid film, compare Figure 5.11 and 5.12.

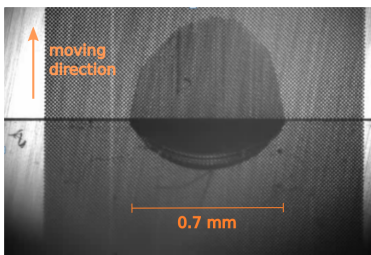


Figure 5.11. Bladed droplet with a transparent doctor blade.

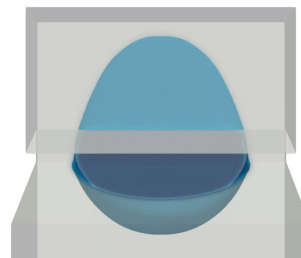


Figure 5.12. Bladed droplet in simulation.

5.2. The Doctor Blading Test Case

It can be observed, that the shape of the top of the advancing films as well as their width agree while the fluid flows out of the domain. In [11], the author describes an approximately logarithmic behavior of the advancing film front, which can also be reproduced within the simulations.

In addition, an instability at the edge of the doctor blade was observed within the experiments, that could be further investigated by the simulations. Some snapshots of the three dimensional simulation are depicted in Figure 5.13, which should contribute to a better understanding of the described observations. The order is from top left to bottom right. The color of the droplet shows the velocity distribution on the surface of the droplet, where the color range from a velocity value of 0 in blue to a value 0.1 m s^{-1} in red. After about 30 milliseconds, the droplet has disappeared out of the domain.

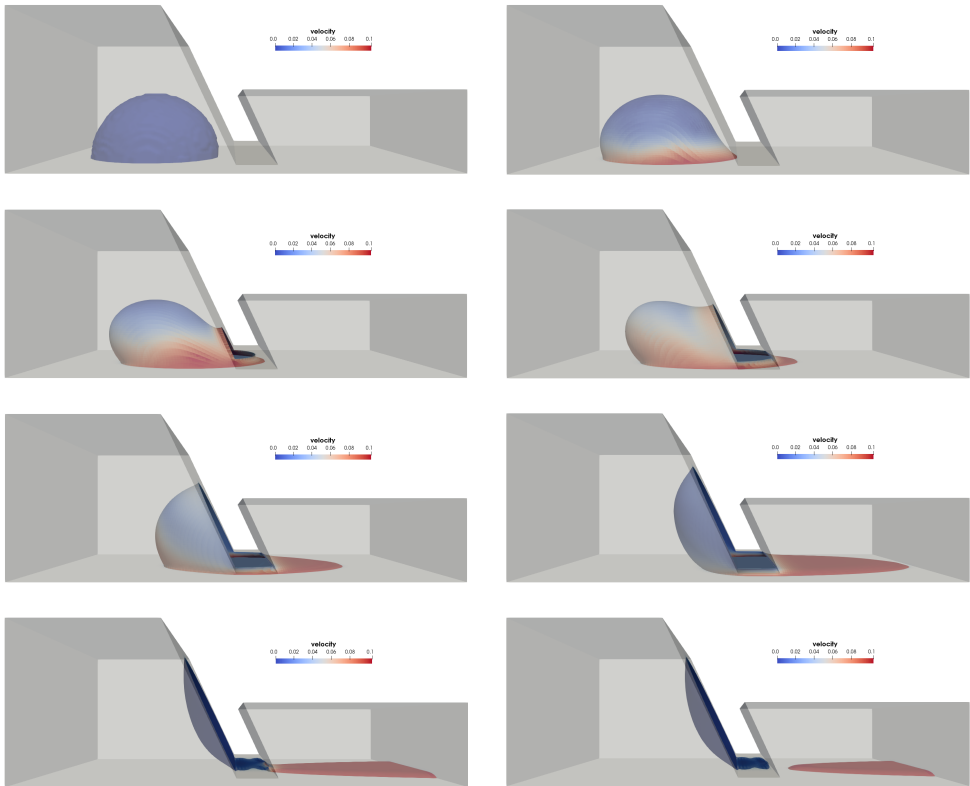


Figure 5.13. Snapshots of the droplet simulated in 3D.

As was also seen in the experiments, a liquid ribbon with a width corresponding to the size of the drop appeared on the moving cylinder surface by the penetration of liquid from the drop into the gliding channel. The width of the ribbon coincided with the endpoints of the contact lines of the drop. Due to the continuous loss of liquid from the drop, the contact lines of the drop on the cylinder and on the doctor blade steadily contracted. Remarkable at this point is, the 3D simulations could show, that a negative pressure occurs at the edges of the droplet. This has not been described in this way before. Once the proceeding contact angle approached a critical value, the droplet loses the contact to the cylinder as expected.

Both in the physical experiment and in the simulation we can further observe, that after the droplet has been bladed, fluid remains behind and under the doctor blade. This is in line with expectations, since not all of the fluid is removed from the surface due to the surface tension and the small contact angle. In practice, this generates serious fluid accumulations over time at the doctor blade, which can also lead to problems, especially when printing for a longer period of time. Further results, especially concerning the pressure distribution at the edges of the doctor blade and within the gap between the top of the doctor blade and the printing form, will be presented soon in a paper within the framework of a cooperation in the CRC 1194.

For now, this should be enough to ensure that we can use the `interFoam` solver to simulate a nontrivial part of the gravure printing procedure, the doctor blading process. The next step is to not only describe and observe the problem, but also its control and optimization.

5.2.2 Optimization Objectives

Optimizing the doctoring process as an exemplary application of a wetting process, is a main goal of this thesis. Optimization means in this case to improve particular aspects of the printing process. In the following we aim to clarify this statement and define some central questions that should be answered in the remainder of this chapter and give us suitable objective functions for our optimization.

Experimental investigations showed, that shape and thickness of the lubrication film behind the doctor blade are, among other things, determined by the viscosity of the printing fluid and the gap height between the tip of the doctor blade and the printing form [37]. We define the shape, and in this regard especially the film thickness, as our quantity of interest (QOI) and use the final time state of the

simulations for the objective function. To optimize the size of this QOI, we consider the phase fraction α . The α value indicates where the domain is filled with the fluid and can therefore be used to calculate for example the film thickness at a predefined position behind the tip of the doctor blade. Then, the goal of the optimization is to reach a desired film thickness at the chosen position, depending on a control variable of the optimization problem. One control is the printing fluid viscosity μ_l and the other one is the gap height d_{GH} . Therewith, we control our solution with respect to the phase fraction field α for a fixed end time T and a desired state α_d as follows

$$j_\alpha(y, q) := \frac{1}{2} \int_{\Omega_{bd}} |\alpha(x, T) - \alpha_d|^2 dx. \quad (5.1)$$

The objective function states that we have to choose the material parameter μ_l such that the phase fraction field $\alpha(x, T)$ is close to a desired field α_d for a particular part of the domain Ω_{bd} in the mean square sense. If we are only interested in the film thickness, i.e., the height of the fluid film at a specific position in x -direction, Ω_{bd} only consists of a single belt of cells orthogonal to the defined position. However, it is also the possibility to consider a larger area, for example the whole fluid film volume behind the doctor blade. We consider both cases in our numerical studies. Overall, two different optimization problems arise, whereas the numerical results of them are presented in Chapter 5.4. The first is optimizing the area behind the doctor blade with respect to the fluid viscosity μ_l to correspond to a predefined form. And the second is finding the optimal gap height for a specific film thickness in a predefined distance behind the doctor blade, this time with a fixed viscosity of the fluid.

Another interesting question is the behavior of the flow itself. Many printing failures can occur if the flow is too brisk. We deliberately do not refer to turbulence in this context, since although the fluid behaves agilely, it does not reach the Reynolds numbers that classify a turbulent regime. Moreover, the mathematical model does not include a special turbulence treatment. But agile behavior of the ink leads to different problems while the printing process, for example air inclusions or fluid accumulations. To avoid these failures we use a measure of turbulence, the vorticity ω . It is calculated with the curl of the flow velocity u

$$\omega := \text{curl } u = \nabla \times u.$$

Hence, we define the vorticity ω as another QOI. A theoretical foundation of this kind of optimization problem is given in [2].

The resulting optimization problem is then

$$\min_{y \in Y, q \in Q} j_\omega(y, q) \quad \text{with} \quad j_\omega(y, q) = \frac{1}{2} \|\text{vort}(y, q)\|^2. \quad (5.2)$$

How we solve these optimization problems in practice, will be examined in the next section.

5.3 Optimization Framework

Our optimization algorithm is based on the general approach for solving optimization problems, we stated in 3.5. As we have just seen in Section 5.2.2, all objective functions have a least squares structure. Due to the structure of the problems, we can apply a modification of Newton's method, the Gauss-Newton algorithm, to solve the optimization problem. It is used to solve nonlinear least squares problems by minimizing the sum of squared function values. The major advantage of this method is, that second derivatives, which can be challenging to compute, are not required [13]. Only the first derivative of the objective function is necessary. We consider a problem with the following structure

$$\min_{x \in \mathbb{R}^n} j(x) \quad \text{with} \quad j(x) = \frac{1}{2} \|J(x)\|^2.$$

Then we determine the search direction $s^k \in \mathbb{R}^n$ by solving the Gauss-Newton equation

$$J'(x^k)^T J'(x^k) s^k = -J'(x^k)^T J(x^k),$$

where x^k is the current solution and the new solution is obtained by $x^{k+1} = x^k + t_k s^k$ with an appropriate step size $t_k > 0$.

If we go back to the optimization problem we stated in Section 3.1, the corresponding discretization of problem (P) has the form

$$\min_{q^h \in Q^h} j^h(y^h(q^h), q^h) \quad \text{s.t.} \quad q^h \in Q_{ad}^h. \quad (P^h)$$

The control is pointwise bounded with the lower bound q_a and the upper bound q_b

$$q_a \leq q(x) \leq q_b \quad \text{in } \Omega.$$

5.3. Optimization Framework

So we assume a convex set

$$Q_{ad} = \{q \in \mathbb{R}^n : q_a \leq q \leq q_b\},$$

where $q_a \leq q_b \in \mathbb{R}^n$ and all inequalities are to be understood component-wise. For this problem, we formulate the optimization framework in Algorithm 5.1. Starting with an initial control q^0 , the following main steps are exercised for $k = 0, 1, 2, \dots$ until a satisfactory convergence is achieved:

Algorithm 5.1 General Optimization Framework for Wetting Problems

- 1: Initialize control variable q^0
- 2: Calculate initial state with OpenFOAM simulation with q^0
- 3: Start optimization loop
- 4: For $k = 0, 1, 2, \dots$:
- 5: Solve state equations by OpenFOAM simulation to obtain the corresponding state $y^k = y(q^k)$
- 6: Compute the gradient of J with respect to the control q^k , $J' := \frac{d}{dq} J(q^k)$
- 7: (a) by determining sensitivities δy^k with interSensFoam and use equation (3.2)
- 8: (b) by calculating the difference quotient, see Chapter 3.2
- 9: Calculate Gauss-Newton step to determine the increment of control:

$$\delta q^k = - (J'(q^k)^T J'(q^k))^{-1} J'(q^k)^T J(q^k)$$

- 10: Set the step size $t_k \in \{1, \frac{1}{2}, \frac{1}{4}, \dots\}$ such that $j(q^k + t_k \delta q^k) < j(q^k)$
 - 11: Update the control with $q^{k+1} = q^k + t_k \delta q^k$
 - 12: Return to step 5 or stop, if a satisfactory convergence criterion is achieved
-

Note, the lower and upper bounds can also be integrated with an adequate choice of a projection. Furthermore, each iteration of the optimization algorithm requires at least one solution of the whole flow process. If we use the sensitivity solver interSensFoam in step 7, the derivatives are calculated together with the state equations, which is more favorable than calculating an additional perturbed state for the difference quotient in step 8. The Gauss-Newton step is calculated in step 9 to find the actual search direction. Furthermore, the step size is controlled in step 10 by halving the step as long as the new residual is greater than the old one. If such a step size t_k is found, the control can be updated and the algorithm repeats the calculation of state and sensitivity equations. Convergence is achieved, if a prescribed stopping criteria is fulfilled. This is for example the case, if the residuum, hence the square value of the objective function, is small enough. Then, the necessary first order optimality condition is fulfilled and a local optimum found.

The above optimization algorithm is composed of an outer initialization and optimization framework, implemented in Matlab, and an inner simulation part, where the state equations and the sensitivities are calculated numerically with the OpenFOAM solver `interSensFoam`. The framework and the inherent coupling mechanisms are schematically depicted in Figure 5.14.

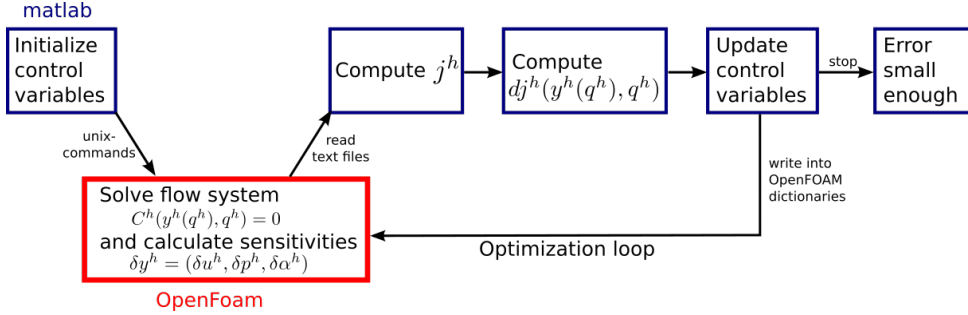


Figure 5.14. Coupling of Matlab and OpenFOAM simulations.

Many extensions of this rough framework are imaginable. To speed up the computational time, a multilevel approach could be favorable. The main idea of a multilevel optimization problem is to solve a major part of the iterations on a comparatively coarse discretization. First if it is necessary, a refinement of the mesh takes place, which can also be a local one. The optimization starts with a coarse mesh h_0 and generates adaptive mesh refinements h_k with the corresponding discretization of P^{h_k} , which leads to the problem

$$\min_{q^{h_k} \in Q^{h_k}} j^{h_k}(y^{h_k}(q^{h_k}), q^{h_k}) \quad \text{s.t.} \quad q^{h_k} \in Q_{ad}^{h_k}. \quad (P^{h_k})$$

Due to the fact, that most of the state evaluations can be executed with a much smaller number of cells, the computational cost are reduced significantly and thus the overall execution time of the optimization as well. Furthermore, with a variation of the optimization method itself, the range of possible objective functions can be enlarged. However, this may entail the necessity of higher derivatives and is outside the scope of this thesis.

The optimization framework presented in this section can solve parameter identification problems, where only the liquid viscosity is controlled at the moment. Further controls, like other material parameter, operational parameter or geometrical aspects are also conceivable for the future. But therefore, also the OpenFOAM solver `interSensFoam` has to be adapted with respect to the sensitivity equations.

To overcome the lack of appropriate sensitivity equations, we calculate the derivatives of the state equations for the optimization problems involving a change of the geometry with the difference quotient, see (3.4). The numerical results of the optimization problems concerning wetting problems, that we discussed in Section 5.2.2, are presented in the next section.

5.4 Numerical Optimization Results

We now revisit the optimization objectives, that we formulated in Section 5.2.2. As we already mentioned, we consider three optimization problems. The first optimization problem is a parameter identification problem, where a desired fluid film has to be achieved by controlling the liquid viscosity, see Section 5.4.1. The second problem concerns the optimization of the fluid film height at a specific point behind the doctor blade, where the gap height between the tip of the doctor blade and the printing form is the control variable, see Section 5.4.2. And finally we present the optimization of the vorticity within the ink chamber with respect to the inclination angle of the doctor blade in Section 5.4.3.

The underlying simulations are similar to the simulation we used in our validation case in Section 5.2.1. One difference is, that we primarily calculate the optimization problems in a two dimensional setting. This only changes the dilatation in z-direction within the simulation setting, all material and operational parameter, the solver adjustments and boundary conditions of the state variables remain the same. The parametrization of the domain with blocks and the numbering of vertices and faces can also be adopted, since a 2D simulation in OpenFOAM means that only a single cell is used in z-direction. For the two dimensional version of the mesh we then obtain 18600 mesh cells, where the size of the cells is the same as in Section 5.2.1. Another aspect to consider is the initial distribution of the fluid, where the first scenario is employed in all three optimization problems. Fluid parameter can vary, since the liquid viscosity is the control variable in the first case. Furthermore, for the numerical simulations within the first optimization problem, the developed OpenFOAM solver `interSensFoam` is used. Similar to the well known application `interFoam`, which we use for the other two optimization problems instead, it is a solver for two incompressible and immiscible phases using finite volume discretization on collocated grids, additionally featured with the calculation of sensitivities for the state variables. See Section 4.4.2 and Appendix B.2 for more details. Therefore, an additional initial setup is required for the sensitivity fields δu , δp and $\delta \alpha$, we denote

as $d\alpha$, dU and dP_rgh within the simulation. The corresponding boundary conditions are partly the same as the BCs of the primary fields from Table 5.4, and are presented in the following Table 5.5.

Patch name	$d\alpha$	dP_rgh	dU
leftWall	zeroGradient	fixedFluxPressure	partialSlip
outletWall	zeroGradient	totalPressure	inletOutlet
movingWall	zeroGradient	fixedFluxPressure	noSlip
upperWall	zeroGradient	fixedFluxPressure	partialSlip
atmosphere	zeroGradient	totalPressure	pressureInletOutletVelocity

Table 5.5. Setup of boundary conditions for the sensitivity solver.

This is a recommended setup for the sensitivity fields with the available boundary conditions implemented in OpenFOAM. Next we present the mentioned optimization problem in more detail and show the numerical results.

5.4.1 Optimization of the Liquid Viscosity

An important role in printing processes is the choice of the printing liquid. Printing liquids, also called inks, are very complex due to non-newtonian behavior and color pigments contained. We do not consider their full complexity here, but we examine how the viscosity of the printing fluid affects the film formation behind the doctor blade. Therefore, we choose as desired viscosity $\mu_l = 1 \cdot 10^{-3} \text{ m}^2\text{s}^{-1}$, resulting in a specific film behind the doctor blade, and optimize the calculated film with respect to the liquid viscosity to perfectly match the desired state. The objective function (5.1) fits to this question, where Ω_{bd} is set as the part of the domain, which lies behind the tip of the doctor blade. For the control μ_l we define the following lower and upper bounds

$$q_a = 1 \cdot 10^{-4} \text{ m}^2\text{s}^{-1}, \quad q_b = 1 \cdot 10^{-2} \text{ m}^2\text{s}^{-1}$$

and set the initial control to

$$q^0 = 5 \cdot 10^{-3} \text{ m}^2\text{s}^{-1}.$$

Note, the unity of the viscosity belongs to the dynamic one. As convergence tolerance we choose a value of 10^{-6} for the residuum of the objective function. The

5.4. Numerical Optimization Results

Gauss-Newton algorithm provides the following result

Iter.	Funct.-Eval.	$q = \mu_l$	Residual
0	1	5.0000e-03	1.2143e+01
1	4	2.1554e-03	3.6828e+00
2	6	1.2013e-03	2.5485e-01
3	7	9.7934e-04	3.3790e-03
4	8	9.9805e-04	2.9651e-05
5	9	9.9985e-04	1.7512e-07

Table 5.6. Result of the Gauss-Newton algorithm.

Table 5.6 shows, that the Gauss-Newton algorithm reaches a sufficiently accurate result after only a few iterations. In five iterations, nine function evaluations are necessary to match the desired viscosity. Then, a residual below the defined tolerance of 10^{-6} is reached, where the residual indicates the value of the objective function. This value should converge to zero, since we considered as objective function a tracking type function as in equation (5.1). Note, that this is a local minimum. For the simulation we obtain the following results, where we focus on the quantity of interest, hence the fluid film behind the doctor blade.



Figure 5.15. Initial state in Ω_{bd} .



Figure 5.16. Optimized state in Ω_{bd} .

The Figures 5.15 and 5.16 show the initial and optimal simulation results, only for the area Ω_{bd} behind the doctor blade. A closer look reveals, that the shape of the fluid film for the optimized state is a little different from the one for the initial state. That is what we expected for liquids with different viscosity. Furthermore, the optimized state coincides perfectly with the desired state, that we preliminary calculated for a specific desired liquid viscosity. Of course, this should be the case since the material parameters of optimized and desired state match very closely, see Table 5.6.

Additionally, the sensitivity calculations provide the following results, first depicted for the phase fraction field α and then for the velocity U .

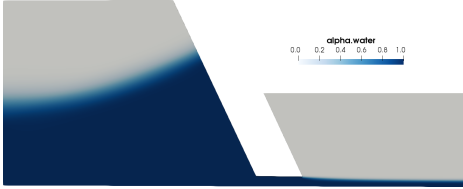


Figure 5.17. Field α .water.

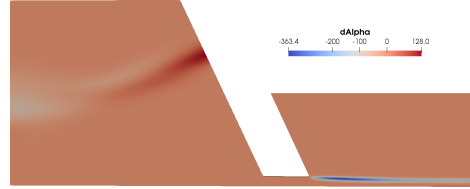


Figure 5.18. Field $d\alpha$.

The sensitivity of α , see Figure 5.18, reflects the variation of the phase fraction field and matches the position of the interface in Figure 5.17. Note, the interface on the left hand side of the doctor blade smears out slightly due to necessary solver adaptations, we discuss in more detail in Appendix B.2. Also the velocity sensitivity in Figure 5.20 shows the variation of the velocity field in Figure 5.19, where the highest values occurring at the moving wall and the tip of the doctor blade.

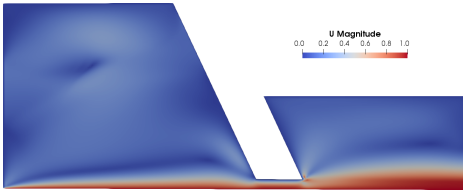


Figure 5.19. Field U .

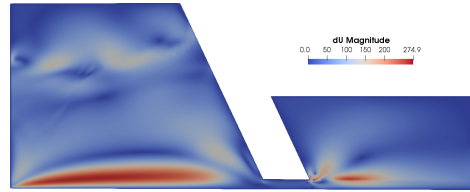


Figure 5.20. Field dU .

The velocity sensitivities agree well with the corresponding difference quotients with a variance of around ten percent. In Figure 5.21 can be observed, that interSensFoam reproduces the main aspects of the difference quotient and in particular also at the interface and at the boundaries. This confirms, that the implemented solver calculates the correct derivatives of the state equations with the sensitivity approach and that the applied numerical methods are suitable in the context of this wetting process.

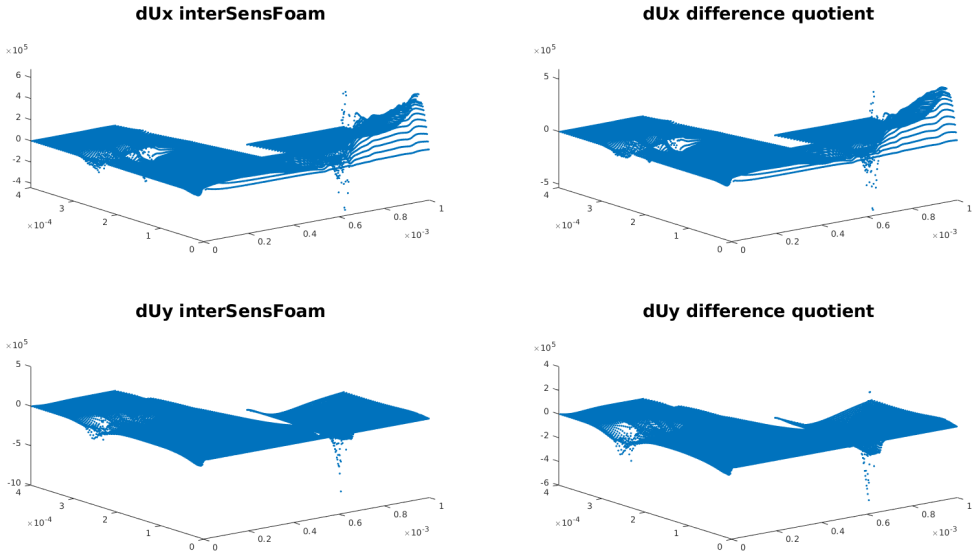


Figure 5.21. Comparison of velocity sensitivity components with the difference quotient.

5.4.2 Optimization of the Gap Height

The task of this optimization problem is to find the optimal gap height for a given film thickness behind the doctor blade. Here, the gap height d_{GH} is the distance between the tip of the doctor blade and the lower wall, the latter representing the printing form. The question of a predefined gap height is more a coating than a printing task, but in this context many applications can be found where an exact thickness of the fluid film after doctor blading is of importance. Hence, the QOI is again concerning the fluid film, resulting in objective function (5.1). In this case, however, only at a certain point behind the doctor blade, in fact at a distance of 0.2 mm to the tip of the doctor blade, confer to Figure 5.22. The red bar represents the film thickness at the specific point. This is an arbitrary value that takes into account a flattening of the film just behind the tip of the doctor blade, to obtain a value as representative as possible for a constant film thickness. As fluid, normal water is assumed, see Table 5.1 for the respective material properties, and a tangential wall velocity of $u_{cyl} = 1 \text{ m s}^{-1}$ is set to the lower wall.

Besides the procedure we described for the former optimization problem, geometrical optimization issues arise in this problem formulation as well. Due to the change of the gap height, the outer domain, and therefore the mesh, has to be adapted in every iteration step of the optimization algorithm. To preserve the

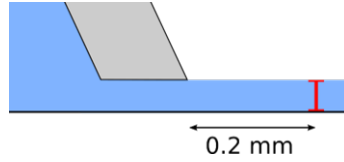


Figure 5.22. Position of QOI (red bar).

number and order of cells in every new simulation, we abstain from remeshing and use a transformation function instead. This economizes computational costs and prevents a change in number and order of mesh elements. One possible meshing routine in OpenFOAM, which is also the most widely used, is called blockMesh. Here, the domain is subdivided in blocks, where different cell sizes can be applied. This is only the case if they match on their common boundaries. The block structure of the three dimensional version of the test case can be seen in Figure 5.7. Now, the transformation function compresses and stretches the outer corners of the respective blocks to create a movement of the doctor blade in vertical direction. In the following we have a closer look to the transformation.

We denote $\Omega(t)$ as the reference domain. The deformation of $\Omega(t)$ is given by the diffeomorphism $\Omega \times I \rightarrow \mathbb{R}^d$, $\tau : (x, t) \mapsto \Omega^\tau(t)$, where $\Omega^\tau(t)$ is the transformed domain and $\tau \in C^1(\Omega, \Omega^\tau)$. Then it holds

$$x^\tau(t) = \tau(x, t) = x + f(x, t),$$

for $x^\tau \in \Omega^\tau$ and $x \in \Omega$. Here, f is the displacement function describing the variation of a vertex within the respective spatial direction.

Now we apply the presented transformation to our specific test case configuration, hence to the block structure of our mesh. In a first step, the whole domain is triangulated as shown in Figure 5.23.

We define the set of domain corners as

$$V = 10^{-3} \text{ m} \cdot$$

$$\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.55 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.65 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0.02 \end{pmatrix}, \begin{pmatrix} 0.55 \\ 0.02 \end{pmatrix}, \begin{pmatrix} 0.65 \\ 0.02 \end{pmatrix}, \begin{pmatrix} 1 \\ 0.02 \end{pmatrix}, \begin{pmatrix} 0.566 \\ 0.2 \end{pmatrix}, \begin{pmatrix} 1 \\ 0.2 \end{pmatrix}, \begin{pmatrix} 0 \\ 0.4 \end{pmatrix}, \begin{pmatrix} 0.37 \\ 0.4 \end{pmatrix} \right\}.$$

Then the reference domain is defined as the convex hull of this set

$$\Omega = \text{conv}(V^T).$$

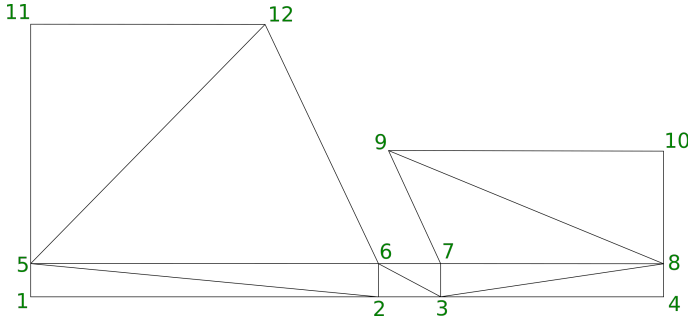


Figure 5.23. Triangulation of the domain in 2D

The set of nodes, spanning the individual triangulation segments in Figure 5.23, is

$$T(\Omega) = \left\{ \begin{pmatrix} 1 \\ 2 \\ 5 \end{pmatrix}, \begin{pmatrix} 2 \\ 6 \\ 5 \end{pmatrix}, \begin{pmatrix} 2 \\ 3 \\ 6 \end{pmatrix}, \begin{pmatrix} 3 \\ 7 \\ 6 \end{pmatrix}, \begin{pmatrix} 3 \\ 4 \\ 8 \end{pmatrix}, \begin{pmatrix} 3 \\ 8 \\ 7 \end{pmatrix}, \begin{pmatrix} 7 \\ 8 \\ 9 \end{pmatrix}, \begin{pmatrix} 8 \\ 10 \\ 9 \end{pmatrix}, \begin{pmatrix} 12 \\ 11 \\ 5 \end{pmatrix}, \begin{pmatrix} 5 \\ 6 \\ 12 \end{pmatrix} \right\}^T.$$

For all cells in the respective segment of $T(\Omega)$, the following transformation function is applied to the mesh points $v_i = (x, y)^T \in V_i$

$$\begin{aligned} \tau(v_i) &= y_i && \text{for } i \in \{1, \dots, 4\}, \\ \tau(v_i) &= y_{i-4} + q^k && \text{for } i \in \{5, \dots, 8\}, \\ \tau(v_i) &= y_i + (q^k - q_0) && \text{for } i \in \{9, \dots, 12\}, \end{aligned}$$

where q^k is the current control. Note, that we differ between the cells within the triangulation segments under and above the doctor blade. The nodes one to four do not change their position, only the y -component of the nodes five to twelve are offset against the new control value q^k . The same transformation can be done in x -direction or both in x - and y -direction. Then the displacement has to be added to the respective spatial component.

To initialize the optimization problem, we choose the following lower and upper bounds for the control

$$q_a = 1 \cdot 10^{-5} \text{ m}, \quad q_b = 1 \cdot 10^{-4} \text{ m}.$$

Starting from the initial gap height $q^0 = 5 \cdot 10^{-5} \text{ m}$, a desired state with the gap height $q_d = 2 \cdot 10^{-5} \text{ m}$ has to be achieved. Again, the desired state was simulated before the optimization to calculate the residual. The numerical results of

the Gauss-Newton algorithm are presented in Table 5.7. After six iteration steps, where altogether eleven function evaluations were executed, the optimization loop converges at the desired state.

Iter.	Funct.-Eval.	$q = d_{GH}$	Residual
0	1	0.0500	3.9133e-10
1	2	0.0209	2.4387e-12
2	5	0.0182	6.4717e-14
3	7	0.0185	4.8778e-14
4	8	0.0189	2.6606e-14
5	10	0.0196	3.0451e-15
6	11	0.0200	7.0849e-19

Table 5.7. Result of the Gauss-Newton algorithm.

The initial and optimized state are depicted in Figures 5.24 and 5.25. The varying fluid level on the left side of the doctor blade, within the fluid reservoir, stems from differences in the time taken to reach a steady state. Due to the reduced gap height, this state is achieved earlier. Then, the fluid level has no further influence on the film thickness until the fluid reservoir is exhausted.



Figure 5.24. Initial state gap height.



Figure 5.25. Optimized state gap height.

5.4.3 Optimization of the Vorticity

Within the context of gravure printing, the improvement of printing failures is a challenging task. One of these printing failures are air bubbles, which are implicated into the ink and can negatively affect the printing result. At high printing speeds, the fluid may even begin to foam, a worst case scenario for printers. Now the question is, if and how we can influence the printing result positively, without slowing down the velocity of the printing process. We select the inclination angle of the doctor blade as the control variable and choose

5.4. Numerical Optimization Results

the vorticity as objective. This is due to the fact that the vorticity is an indicator for the turbulent behavior in front of the doctor blade, within the ink reservoir.

The optimization problem is solved with the objective function for vorticity, see equation (5.2). Again, the outer boundary moves within the optimization loop, hence the transformation introduced for the optimization of the gap height is used here again. The control variable is the inclination angle θ in this case. We initialize the optimization problem with the following lower and upper bounds for θ

$$q_a = 5^\circ, \quad q_b = 40^\circ.$$

As initial inclination angle we set $q^0 = 25^\circ$. For this problem, no desired state has to be calculated. The optimization algorithm reaches a minimum value for the residual after seven Gauss-Newton iterations. Therefore, the state equations had to be evaluated 40 times. A minimal vorticity value was reached for an inclination angle of $\theta_{opt} = 11.63^\circ$. The results are summarized in Table 5.8.

Iter.	Funct.-Eval.	$q = \theta$	Residual
0	1	2.5000e+01	8.0302e+00
1	2	2.2681e+01	5.5489e+00
2	3	1.7784e+01	2.4625e+00
3	4	1.4643e+01	1.3865e+00
4	5	1.2756e+01	1.1536e+00
5	6	1.1582e+01	1.0721e+00
6	13	1.1629e+01	9.8272e-01
7	40	1.1629e+01	9.8229e-01

Table 5.8. Numerical results with the Gauss-Newton method.

It is noticeable, that the residual is considerably larger than in the optimization problems before. The reason for that is the objective function, since we minimize the norm of the vorticity itself and not associated with a desired state. Table 5.8 as well as the graphical representations in Figure 5.26 and Figure 5.27 reinforce, that the vorticity of the velocity field within the ink reservoir can be reduced by a factor of 10 by changing the inclination angle of the doctor blade. This shows, that we can significantly influence and, at best, considerably improve the gravure printing process by using appropriate optimization procedures.

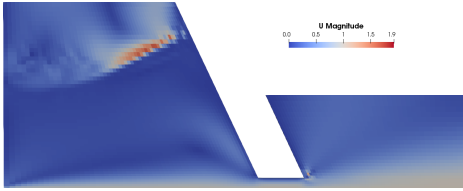


Figure 5.26. Initial state vorticity.

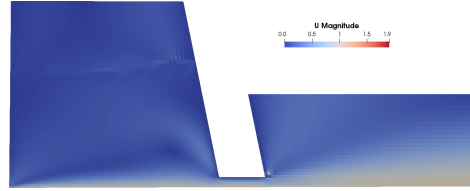


Figure 5.27. Optimized state vorticity.

Conclusion

This thesis deals with the gradient based optimization of two-phase flow problems in the context of wetting phenomena. We developed an optimization framework for the simulation and optimization of such problems, where the application comes from gravure printing. Two different aspects were addressed. On the one hand the theoretical basement, showing existence for the derivative of the corresponding control-to-state mapping, supposing an L^p -maximal regularity setting of the underlying linear problem. Therefore, two different problem formulations, a two-phase formulation with jump conditions and an equivalent one-field approach using the volume of fluid method, were considered. For both formulations we could show that the respective control-to-state mapping of an optimization problem with respect to a distributed control on the left hand side of the momentum equation and to an initial velocity field is continuous and infinitely many times differentiable with respect to the mentioned controls. On the other hand, extensive numerical simulations and optimizations were performed for the doctor blading test case and provided new insights to the underlying wetting phenomena. With a full three dimensional simulation we could reproduce corresponding experimental results and were able to confirm an observed instability, which was not described in the literature before. Furthermore we could show, that the simulation based optimization works for non-trivial wetting processes such as gravure printing and can help to improve and even to optimize the printing results. The developed optimization framework combines an optimization algorithm implemented in Matlab with DNS calculations performed in OpenFOAM. To apply a sensitivity approach for calculating the derivatives of the states, required for the optimization method, we had to modify and extend the incompressible two-phase solver `interFoam` from the OpenFOAM software library. Therefore, the numerical methods had to be adapted to solve the differentiated equations adequately. The efforts were combined in the new solver `interFensFoam`. The optimization framework was tested on several exemplary optimization prob-

lems, which included parameter identification and the optimization of geometrical aspects. Note, that the findings can easily be transferred to the multiphase case, when instead the basic interFoam solver the multiphase extension multiphaseInterFoam is used for the simulations and the optimization. Even in the case of more than two phases, only interfaces between two phases need to be considered, which is covered by the findings we obtained in this thesis.

Outlook

In this work we have solved only one small piece of the puzzle and a lot of work can still be done. Based on the investigations carried out, further research fields concerning theoretical aspects, for example the development of differentiability results for more complex models and in the presence of uncertainties. The complexity of the model can be increased by structured surfaces, by extensive fluids, thinking about non-newtonian fluids and fluids with particles or surfactants, or by considering the influence of temperature. Taking the temperature into account leads to phase changes at the interface due to evaporation or vaporisation and further effects in material parameters, like the Marangoni effect of surface tension or nonconstant densities and viscosities within the phases.

Furthermore, we have concrete ideas about how to proceed with the constructed test case. At the moment, only a small vicinity around the tip of the doctor blade is considered, although controls, for example the viscosity of the printing fluid, also have an impact on the overall printing process. The test case should therefore be extended for a larger setup. First investigations and results were reached including the complete gravure cylinder and ink reservoir. The challenging part is here to combine the different length scales, concerning the three phase contact line effects on a μm -scale and the geometrical setup adjustments on a cm-scale. This can for example be handled with mindful adaptive mesh refinement. Another point to mention is, that the surface of the printing forms are anything other than plain in a realistic scenario. On the one hand, they have a certain surface roughness with grooves, furrows, ridges and channels. And on the other hand, there are engraved cups which receive the ink for the printed design. We do not consider these cases in our thesis due to its complexity, but first investigations were also done to simulate an engraved surface with OpenFOAM.

Also regarding the optimization itself there are still some possible improvements. The dimension of the control is less, so it is reasonable to apply a sensitivity approach. Increasing the dimension of control variables or including domain variations within an OpenFOAM application, for example to consider real shape optimization problems, requires an adjoint approach. Moreover uncertainties in material and operational parameters could be taken into account by means of robust optimization techniques. This is very reasonable, since the material parameters are susceptible to uncertainties for example the liquids due to temperature variations or the steel doctor blade due to manufacturing reasons. Taking uncertainties into account leads to a bilevel structure of the optimization problem, where the minimization is done with worst case values of the objective function and the constraints. These are in turn obtained through a maximization problem, considering all relevant realizations of uncertain parameters in a given uncertainty set. Hence, we have to solve a maximization problem within a minimization problem. These considerations significantly increase the complexity of the problem, what makes the application of reduced models absolutely recommendable. Proper Orthogonal Decomposition is a well known approach for this part of the problem, which should be investigated for the optimization of wetting phenomena in following works. Furthermore, optimal design of experiments is a promising optimization problem, considered for the optimization of wetting phenomena, especially in the context of the CRC. This approach aims to reduce the number of expensive and challenging experimental setups significantly.

Derivation of Equations

A.1 Derivation of α -Transport Equation

In Section 2.2.1 we mentioned, that conservation of mass is described by

$$\partial_t \rho + \nabla \cdot (\rho u) = 0. \tag{A.1}$$

Introducing the density field from equation (2.17) into equation (A.1) and assuming ρ_l, ρ_g are constant, we obtain

$$\begin{aligned} \partial_t \rho(\alpha) + \nabla \cdot (\rho(\alpha) u) &= 0 \\ \Leftrightarrow \partial_t (\rho_l \alpha + \rho_g (1 - \alpha)) + \nabla \cdot ((\rho_l \alpha + \rho_g (1 - \alpha)) u) &= 0 \\ \Leftrightarrow \partial_t (\rho_l \alpha + \rho_g - \rho_g \alpha) + \nabla \cdot (\rho_l \alpha u + \rho_g u - \rho_g \alpha u) &= 0 \\ \Leftrightarrow \partial_t ((\rho_l - \rho_g) \alpha) + \nabla \cdot ((\rho_l - \rho_g) \alpha u + \rho_g u) &= 0 \\ \Leftrightarrow (\rho_l - \rho_g) \partial_t \alpha + \nabla \cdot ((\rho_l - \rho_g) \alpha u) + \rho_g \nabla \cdot u &= 0. \end{aligned}$$

With the assumption of a divergence-free velocity field, i.e., $\nabla \cdot u = 0$, we obtain the following transport equation

$$\partial_t \alpha + \nabla \cdot (\alpha u) = 0. \tag{A.2}$$

Equation (A.2) also comes from the idea that, as the interface moves, the shape of the region occupied by each phase changes, but each fluid particle retains its identity. Thus, the material derivative of α , following the motion of a particle, is equal to zero. This implies

$$\frac{d\alpha}{dt} = \partial_t \alpha + u \cdot \nabla \alpha = 0.$$

A.2 Derivation of compressed α -Transport Equation

In Section 4.1.3 we introduced a compressed α -transport equation. There, an artificial compression term is added to the original α -transport equation (2.19). Without loss of generality, we defined our phase fraction value α as the value of one of the phases, e.g., the liquid one, and the velocity as the velocity of the corresponding phase. So we write the transport equation as

$$\partial_t \alpha_l + \nabla \cdot (\alpha_l u_l) = 0. \quad (\text{A.3})$$

In the original VOF approach by HIRT and NICHOLS [42], the velocity in this equation is assumed to be equal to the mixed velocity, i.e., $u_l = u$, which is only valid without numerical diffusion. To overcome this lack, HENRY WELLER, one of the OpenFOAM developer, defined the mixed velocity u and the relative velocity u_r for a liquid and a gaseous phase by

$$\begin{aligned} u &= \alpha_l u_l + \alpha_g u_g = \alpha_l u_l + (1 - \alpha_l) u_g, \\ u_r &= u_l - u_g. \end{aligned}$$

The addition of these two equations yield

$$\alpha_l u_l = \alpha_l u + (1 - \alpha_l) u_r.$$

Inserting this expression in equation (A.3) results in the desired compressed α -transport equation, we introduced in Section 4.4.1 with equation (4.10)

$$\partial_t \alpha_l + \nabla \cdot (\alpha_l u) + \nabla \cdot [u_r \alpha_l (1 - \alpha_l)] = 0.$$

Developer Documentation

interSensFoam

Appendix B contains further informations about the developed OpenFOAM solver `interSensFoam`. This is an extension of Section 4.4.2 and aims to a deeper understanding of the source code to be able to apply it to its own test cases. Since `interSensFoam` is an extension of the OpenFOAM inherent solver `interFoam`, without omitting any part of the code, it can be also used as documentation of `interFoam`. But note, we do not promise a complete and final description of the solver. All extensions in comparison to the original solver are color-marked at the corresponding location. We start with a collection of the important variables and constants used in OpenFOAM, then we will have a walk through the code of `interSensFoam` and in the end we will describe the setup and compilation options of a test case with `interSensFoam`.

B.1 Variables and Constants

It follows a description of all used variables and constants in a theoretical and practical way. They can be assigned to a scalar or more sophisticated classes as

- `vol...Field`: A field defined at cell centers.
- `surface...Field`: A field defined on cell faces.

Note, that all these fields consist of values for the internal field and also for the boundary patches. As dimension set we use SI units. Furthermore, some basic OpenFOAM operations are listed.

Constants

Symbol	Access Function	Description	Dimension	Class
ρ_1	rho1	Density fluid 1	[1 -3 0 ...]	dimensionedScalar
ρ_2	rho2	Density fluid 2	[1 -3 0 ...]	dimensionedScalar
μ_1	mu1	Dynamic viscosity fluid 1	[1 -1 -1 ...]	dimensionedScalar
μ_2	mu2	Dynamic viscosity fluid 2	[1 -1 -1 ...]	dimensionedScalar
ϑ_1	nu1	Kinematic viscosity fluid 1	[0 2 -1 ...]	dimensionedScalar
ϑ_2	nu2	Kinematic viscosity fluid 2	[0 2 -1 ...]	dimensionedScalar
g	g	Gravitational acceleration	[0 1 -2 ...]	uniformDimensioned VectorField
σ	sigma	Surface tension coefficient	[1 0 -2 ...]	dimensionedScalar
ghRef	ghRef	Reference value of g , if $ g $ is large	[0 2 -2 ...]	dimensionedScalar
ocCoeff	ocCoeff	Off-centering coefficient for the ddt scheme	-	Scalar
cnCoeff	cnCoeff	$= \frac{1}{1+\text{ocCoeff}}$, time blending factor	-	Scalar
c_α	cAlpha	Weight for interface compression	-	Scalar
$N_{\alpha,corr}$	nAlphaCorr	Number of α corrector steps	-	Scalar
$N_{\alpha,sub}$	nAlphaSubCycles	Number of α -sub cycles	-	Scalar
C_{max}	maxCo	Maximal Courant number	-	Scalar
$C_{\alpha,max}$	maxAlphaCo	Maximal Courant number for α -equation	-	Scalar
N_{corr}	nCorrectors	Maximal number of PISO loops	-	Scalar

Variables

Symbol	Access Function	Description	Dimension	Class
u	U	Velocity field	[0 1 -1 ...]	volVectorField
p_{rgh}	p_rgh	Modified pressure Field	[1 -1 -2 ...]	volScalarField
p	p	Pressure field, $p = p_rgh + \rho \cdot gh$	[1 -1 -2 ...]	volScalarField
α	alpha.water	Phase fraction field	[0 0 0 ...]	volScalarField
ρ	rho	Density, $\rho = \alpha_1 \rho_1 + (1 - \alpha_1) \rho_2$	[1 -3 0 ...]	volScalarField
μ	mu	Dynamic viscosity, $\mu = \alpha_1 \rho_1 \nu_1 + (1 - \alpha_1) \rho_2 \nu_2$	[1 -1 -1 ...]	volScalarField
ϕ	phi	$= u_f \cdot S_f = \text{linearInterpolate}(U) \& \text{mesh.Sf}()$	[0 3 -1 ...]	surfaceScalarField
$\rho\phi$	rhoPhi	$= \rho \cdot \phi = \text{interpolate}(\rho) * \text{phi}$	[1 0 -1 ...]	surfaceScalarField
$\alpha\phi$	alphaPhi10	$= \text{phi} * \text{interpolate}(\text{alpha1})$	[0 3 -1 ...]	surfaceScalarField
S_f	Sf()	Face area vectors	[0 2 0 ...]	surfaceVectorField
$ S_f $	magSf()	Face area magnitudes	[0 2 0 ...]	surfaceScalarField
\vec{C}	C()	Cell centers	[0 1 0 ...]	volVectorField
\vec{C}_f	Cf()	Face centers	[0 1 0 ...]	surfaceVectorField
V	V()	Cell volumes	[0 3 0 ...]	volScalarField
\hat{v}_f	nHatf()	normal face vector of the interface onto S_f	[0 2 0 ...]	surfaceScalarField
ϕ_c	phic	Standard face-flux compression coefficient, $= \text{cAlpha} * \text{mag}(\text{phi}/\text{magSf}())$		surfaceScalarField
ϕ_r	phir	$= \text{phic} * \text{nHatf}()$		surfaceScalarField
	gh	$= g * C() - ghRef$	[0 2 -2 ...]	volScalarField
	ghf	$= g * Cf() - ghRef$	[0 2 -2 ...]	surfaceScalarField
	contErr	$= \text{div}(\text{phi}), \text{continuity errors}$	-	volScalarField
	sumLocalContErr	Sum of the local continuity errors	-	Scalar
	globalContErr	Global continuity error	-	Scalar
	cumulativeContErr	Cummulative global continuity error	-	Scalar

New Variables

Symbol	Access Function	Description	Dimension	Class
δu	dU	Velocity field sensitivity	[0 1 -1 ...]	volVectorField
δp_m	dP_rgh	Modified pressure field sensitivity	[1 -1 -2 ...]	volScalarField
δp	dP	Pressure field sensitivity	[1 -1 -2 ...]	volScalarField
$\delta \alpha$	dAlpha	Phase fraction field sensitivity	[0 0 0 ...]	volScalarField
$\delta \phi$	dPhi	$= \delta u_f \cdot S_f = \text{linearInterpolate}(dU) \& \text{mesh.Sf()}$	[0 3 -1 ...]	surfaceScalarField
$\rho \delta \phi$	rhoDPhi	$= \text{interpolate}(\rho) * dPhi$	[1 0 -1 ...]	surfaceScalarField
$\delta \alpha \phi$	dAlphaPhi	$= \text{linearInterpolate}(dAlpha) * phi$	[0 3 -1 ...]	surfaceScalarField
ρ'	gradRho	Derivative of the density	[1 -3 0 ...]	dimensionedScalar
μ'	gradMu	Derivative of the dynamic viscosity	[1 -1 -1 ...]	dimensionedScalar
$\delta \mu$	dMu	Dynamic viscosity sensitivity	[1 -1 -1 ...]	dimensionedScalar

OpenFOAM operations

Operation	Mathematical Description	Description in OpenFOAM	Comment
Scalar multiplication	sa	$s * a$	
Scalar division	a/s	a / s	
Outer product	ab	$a * b$	rank $a, b \geq 1$
Inner product	$a \cdot b$	$a \& b$	rank $a, b \geq 1$
Double inner product	$a : b$	$a \&\& b$	rank $a, b \geq 2$

B.2 Walkthrough the OpenFOAM code

The new solver `interSensFoam` is based on the `interFoam` solver family from OpenFOAM. Most of the following descriptions are directly deduced from the source code. Furthermore, we used the OpenFOAM wiki (<https://openfoamwiki.net/>, accessed August 2023) and the source code guide from the OpenFOAM Foundation (<https://cpp.openfoam.org/dev/>, accessed August 2023). As we already stated in Chapter 4.4, we use version OpenFOAM-6. We find the underlying code when following the path

```
openfoam6/applications/solvers/multiphase/interFoam
```

In the source code, `interFoam` is presented as a solver for two incompressible, isothermal immiscible fluids using a VOF (volume of fluid) phase fraction based interface capturing approach. The momentum and other fluid properties are of the 'mixture' and a single momentum equation is solved. Turbulence modeling is generic, e.g., laminar, RAS or LES may be selected. Several extensions of `interFoam` already exist, e.g., for the multiphase case or with miscible fluids. We will not consider these extensions in our investigations.

In this thesis, we introduced an extension of `interFoam`, called `interSensFoam`, in which, in addition to the primary equations, the associated sensitivity equations are solved. Therefore, the main *interFoam.C*-file was changed as well as several header files. Furthermore, some header files were added to the main `interFoam` folder and the corresponding VOF folder. The latter one contains calculations regarding the phase fraction field and the α -transport equation. All relevant folders and files were renamed to make sure they contain new content, e.g., the folder `interFoam` is now denoted as `interSensFoam`. We summarized all changes in Figure B.1, where files written in green are completely new and files written in red were modified for running with `interSensFoam`. Additionally, we use some conventions in the following description, in order to have a better overview. Header files were introduced in violet color and written in *italics*, when appearing in the text. Also written in *italics* are all further scripts and dictionaries. A green color is used to identify constants, coefficients and variables occurring within the calculations. And keywords mentioned in **bold** will refer to bash commands.

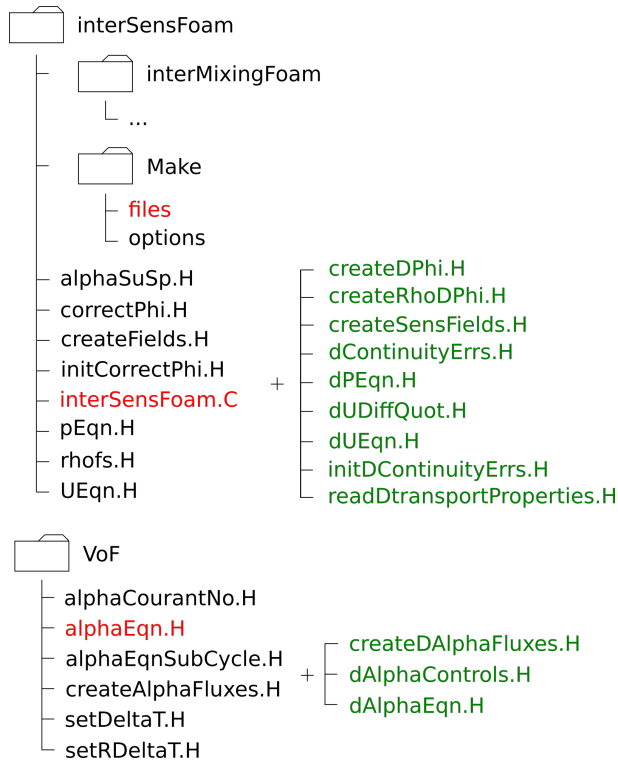


Figure B.1. Structure of interSensFoam.

Access to the full source code of interSensFoam you will find here

<https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/4021>

Following the README file will setup the solver and you can start with your optimization. Appropriate examples are also included in this repository. To understand and reconstruct the doctor blading test case, visit the following repository

<https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/4118>

InterSensFoam.C

We start with explaining the main file *interSensFoam.C*. In the beginning of the file a lot of header files are included. Some of these header files just provide routines for a specific operation, other header files already calculate variables.

```

33   Created by Elisabeth Diehl, 10.04.2018
34
35  \*-----*/
36
37  #include "fvCFD.H"
38  #include "dynamicFvMesh.H"
39  #include "CMULES.H"
40  #include "EulerDdtScheme.H"
41  #include "localEulerDdtScheme.H"
42  #include "CrankNicolsonDdtScheme.H"
43  #include "subCycle.H"
44  #include "immiscibleIncompressibleTwoPhaseMixture.H"
45  #include "turbulentTransportModel.H"
46  #include "pimpleControl.H"
47  #include "fvOptions.H"
48  #include "CorrectPhi.H"
49  #include "fvcSmooth.H"
50
51  // * * * * *
52
53  int main(int argc, char *argv[])
54  {
55      #include "postProcess.H"
56      #include "setRootCaseLists.H"
57      #include "createTime.H"
58      #include "createDynamicFvMesh.H"
59      #include "initContinuityErrs.H"
60      #include "initDContinuityErrs.H" //NEW
61      #include "createDyMControls.H"
62      #include "createFields.H"
63      #include "createSensFields.H" //NEW
64      #include "createAlphaFluxes.H"
65      #include "createDAlphaFluxes.H" //NEW
66      #include "initCorrectPhi.H"
67      #include "createUfIfPresent.H"
68
69      turbulence->validate();
70
71      if (!LTS)
72      {
73          #include "CourantNo.H"
74          #include "setInitialDeltaT.H"
75      }
76
77      // * * * * *
78      Info<< "\nStarting time loop\n" << endl;

```

Figure B.2. File *interSensFoam.C*, part 1.

```
79
80 while (runTime.run())
81 {
82     #include "readDyMControls.H"
83
84     if (LTS)
85     {
86         #include "setRDeltaT.H"
87     }
88     else
89     {
90         #include "CourantNo.H"
91         #include "alphaCourantNo.H"
92         #include "setDeltaT.H"
93     }
94
95     runTime++;
96
97     Info<< "Time = " << runTime.timeName() << nl << endl;
98
99     // --- Pressure-velocity PIMPLE corrector loop
100    while (pimple.loop())
101    {
102        if (pimple.firstIter() || moveMeshOuterCorrectors)
103        {
104            mesh.update();
105
106            if (mesh.changing())
107            {
108                // Do not apply previous time-step mesh compression flux
109                // if the mesh topology changed
110                if (mesh.topoChanging())
111                {
112                    talphaPhi1Corr0.clear();
113                }
114
115                gh = (g & mesh.C()) - ghRef;
116                ghf = (g & mesh.Cf()) - ghRef;
117
118                MRF.update();
119
120                if (correctPhi)
121                {
122                    // Calculate absolute flux
123                    // from the mapped surface velocity
124                    phi = mesh.Sf() & Uf();
125
126                    #include "correctPhi.H"
127
128                    // Make the flux relative to the mesh motion
129                    fvc::makeRelative(phi, U);
130
131                    mixture.correct();
132                }
133            }
134        }
135    }
```

Figure B.3. File interSensFoam.C, part2.

```

133
134         if (checkMeshCourantNo)
135         {
136             #include "meshCourantNo.H"
137         }
138     }
139 }
140
141 #include "alphaControls.H"
142 #include "alphaEqnSubCycle.H"
143
144 // Calculate alpha sensitivity
145 #include "dAlphaEqn.H" //NEW
146
147 mixture.correct();
148
149 #include "UEqn.H"
150
151 // --- Pressure corrector loop
152 while (pimple.correct())
153 {
154     #include "pEqn.H"
155 }
156
157 // Calculate velocity and pressure sensitivity
158 #include "dUEqn.H" //NEW
159
160 while (pimple.correct())
161 {
162     #include "dPEqn.H" //NEW
163 }
164
165 if (pimple.turbCorr())
166 {
167     turbulence->correct();
168 }
169 }
170
171 // Calculation of dU with difference quotient dU
172 // #include "dAlphaControls.H"
173 // #include "dUDiffQuot.H"
174
175 runTime.write();
176
177 Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
178     << " ClockTime = " << runTime.elapsedClockTime() << " s"
179     << nl << endl;
180
181 }
182
183 Info<< "End\n" << endl;
184
185 return 0;
186 }

```

Figure B.4. File `interSensFoam.C`, part3.

We start with a description of the included header-files.

fvCFD.H brings in the most fundamental tools for performing finite volume calculations by including a bunch of other files, each of which represents a building block of the finite volume technique.

dynamicFvMesh.H introduces an abstract base class for geometry and topology changing fvMesh.

CMULES.H CMULES stands for multidimensional universal limiter for explicit corrected implicit solution and is a routine to solve the convective-only transport equation. It uses an explicit universal multi-dimensional limiter to correct an implicit conservative and bounded solution, obtained by using rigorously bounded schemes such as implicit Euler in time and upwind in space. Input parameters are the variable to solve, the normal convective flux and the actual explicit flux of the variable.

EulerDdtScheme.H contains a routine for the temporal derivative with an explicit or implicit Euler scheme, using only the current and previous time step values.

localEulerDdtScheme.H contains the local time step temporal derivative by an first-order explicit or implicit Euler scheme, used for pseudo transient solutions of steady-state problems. Note, the local Euler time scheme is not supported by our solver.

CrankNicolsonDdtScheme.H contains an implicit routine for the second-order Crank-Nicolson temporal derivative using the current and previous time-step fields as well as the previous time-step temporal derivative. Note, the CrankNicolson time scheme is not supported by our solver.

subCycle.H performs a subCycleTime on a field.

immiscibleIncompressibleTwoPhaseMixture.H contains a two-phase incompressible transport model. For this purpose, the two header files *interfaceProperties.H* and *twoPhaseMixture.H* are included.

turbulentTransportModel.H contains type definitions for the turbulence as well as RAS and LES models for incompressible flow, based on the standard laminar

transport package.

pimpleControl.H contains the PIMPLE control class to supply convergence information and checks for the PIMPLE loop. It provides time-loop control methods which exit the simulation once convergence criteria have been reached. It also provides PIMPLE-loop control methods, which exit the iteration once corrector convergence criteria have been met. It may also be used to for PISO-based algorithms, as PISO controls are a sub-set of PIMPLE controls.

fvOptions.H contains options to apply the finite volume method.

CorrectPhi.H provides flux correction functions to ensure continuity. It is required during start-up, restart, mesh-motion etc., when non-conservative fluxes may adversely affect the prediction-part of the solution algorithm, which is the part before the first pressure solution, ensuring continuity. This is particularly important for VOF and other multi-phase solver in which non-conservative fluxes cause unboundedness of the phase fraction.

fvSmooth.H provides the functions smooth, spread and sweep, which use the FaceCellWave algorithm to smooth and redistribute the first field argument.

After the introduction of the described header files, the main function starts with:

```
int main(int argc, char *argv [])
```

Here, argc (**argument count**) indicates the number of arguments sent to the program and argv (**argument vector**) contains these arguments. Inside the function, we need to include some other short routines, which already execute the first calculations:

postProcess.H includes the application functionObjects to post-process existing results. This part of the code is only of interest, if we want to use function objects to calculate output variables.

setRootCase.H checks if argc and argv fit together. Furthermore, the root path and case path are checked for availability and correctness.

createTime.H creates the time.

createDynamicFoMesh.H creates the mesh, depending on the time.

initContinuityErrs.H declares and initializes the cumulative continuity error

$$\text{cumulativeContErr} = 0.$$

initDContinuityErrs.H declares and initializes the cumulative continuity error for the sensitivities

$$\text{cumulativeDContErr} = 0.$$

createDyMControls.H includes *createControl.H* to read solution controls (pisoControl, pimpleControl, simpleControl), and includes *createTimeControls.H* to read the control parameters used by setDeltaT (adjustTimeStep (default: false), maxCo (default: 1), maxDeltaT (default: great), which are defined in the file *controlDict*). Furthermore the boolean variables correctPhi (in dependence of mesh movement), checkMeshCourantNo (default: false) and moveMeshOuterCorrectors (default: false), are set. Note: this file only determines if a variable/routine is used or not. No values or fields are read yet, only a default is set in some cases.

createFields.H includes first the header file *createRDeltaT.H*, which is used to create a reciprocal local face time-step field for the LTS model. Local time stepping is used, if a local Euler technique is used as time scheme, which we will not use in our investigations. Nevertheless, a temporal volScalarField *trDeltaT* is created. Then, the fields *p_rgh* and *U* are read. With this field information, the flux *phi* is calculated in the *createPhi.H* header file. Furthermore, the transport properties are read, which involves the volScalarFields *alpha1*, *alpha2* and the dimensionedScalar fields *rho1*, *rho2*. Then the volScalarField *rho* and the surfaceScalarField *rhoPhi* are calculated

$$\begin{aligned} \text{rho} &= \text{alpha1} * \text{rho1} + \text{alpha2} * \text{rho2}, \\ \text{rhoPhi} &= \text{rho} * \text{phi}. \end{aligned}$$

After constructing the user specific turbulence scheme, the gravitational acceleration coefficient *g* is included with *readGravitationalAcceleration.H*. The dimensioned-Scalar *hRef* is initialized and therewith the following fields are calculated in *gh.H*

$$\begin{aligned} \text{ghRef} &= -\text{mag}(g) * \text{hRef}, \\ \text{gh} &= g \ \& \ C() - \text{ghRef}, \\ \text{ghf} &= g \ \& \ Cf() - \text{ghRef}. \end{aligned}$$

With `gh`, the pressure `p` is calculated without the hydrostatic term by

$$p = p_rgh + rho * gh.$$

`createSensFields.H` is inspired by the former file and initializes the sensitivity fields `dU`, `dP_rgh` and `dAlpha`. Moreover, the flux of the velocity sensitivity is created in the included header file `createDPhi.H` and denoted as `dPhi`. Then, the header files `readDtransportProperties.H` and `createRhoDPhi.H` are included, where the first one calculates the dimensionedScalar fields `dMu`, `gradRho` and `gradMu` and the second header file creates the surfaceScalarField `rhoDPhi` with

$$rhoDPhi = rho * dPhi.$$

Similar to the primal field, also the pressure sensitivity `dP` is calculated without the hydrostatic term.

`createAlphaFluxes.H` creates some initial fields relating to the α -transport equation and used for the solution procedure MULES. Here, the surfaceScalarField `alphaPhi10` and the temporal surfaceScalarField `talphaPhi1Corr0` are initialized. It holds

$$alphaPhi10 = phi * alpha1.$$

`createDAlphaFluxes.H` initializes the surfaceScalarField `dAlphaPhi`, which is calculated as

$$dAlphaPhi = phi * dAlpha + dPhi * alpha1.$$

`initCorrectPhi.H` defines a temporary volScalarField field `rAU`, we need later in `dPEqn.H`, and calls the function `CorrectPhi`. This function guaranties that the total inflow and outflow of mass is conserved with introducing the volScalarField `pcorr`, which only plays a role if the mesh moves or the mesh topology changes. Furthermore, continuity errors are calculated and printed by including `continuityErrs.H`, with

$$\begin{aligned} contErr &= \text{div}(\text{phi}), \\ sumLocalContErr &= \text{deltaT} * \text{mag}(contErr), \\ globalContErr &= \text{deltaT} * contErr, \\ cumulativeContErr &= \sum globalContErr. \end{aligned}$$

`createUfIfPresent.H` Creates and initializes the velocity field `Uf` if required, more precisely if `mesh.dynamic()` is true.

Next, a function is executed to validate the turbulence fields after construction and update derived fields as required

```
turbulence -> validate()
```

If there is no local time stepping (!LTS), the following two header files are further included.

CourantNo.H calculates and outputs the mean and maximum Courant number

$$\begin{aligned} \text{CoNum} &= \frac{1}{2} * \max \left(\frac{\text{sumPhi}}{V()} \right) * \text{deltaT}, \\ \text{meanCoNum} &= \frac{1}{2} * \frac{\sum \text{sumPhi}}{\sum V()} * \text{deltaT}, \end{aligned}$$

where $\text{sumPhi} = \sum_f \text{mag}(\text{phi})$.

setInitialDeltaT.H sets the initial time step corresponding to the time step adjustment algorithm as in *setDeltaT.H* described below.

Now, the time loop starts. Within a while loop the time is incremented until the end time is reached or an other termination or error criterion is matched. Before the time is incremented, further header files are included inside the while loop:

readDyMControls.H additionally includes *readTimeControls.H*: and actually sets the given values for the variables `adjustTimeStep`, `maxCo`, `maxDeltaT`, `correctPhi`, `checkMeshCourantNo` and `moveMeshOuterCorrectors`.

setRDeltaT.H will be included if local time stepping (LTS) is allowed and imposes many variables concerning the local time step setting. If no LTS is allowed, the following three header files are included instead of this one.

CourantNo.H see above.

alphaCourantNo.H calculates and outputs the mean and maximum Courant numbers, used for calculating the phase fraction field α .

setDeltaT.H resets the time step to maintain a constant maximum Courant number in case of the choice of an adjustable time step in the *controlDict* dictionary. The reduction of the time step is immediate, while an increase is damped to avoid

unstable oscillations. Then the time step is set corresponding to equation (4.4) with $\lambda_1 = 0.1$ and $\lambda_2 = 1.2$.

The first solved state equation is the α -transport equation. This calculation is embedded in the file *alphaEqnSubCycle.H*, where a possible α -sub cycling within the actual time step is started, if the user chose a value `nAlphaSubCycles` > 1 . Within the file *alphaEqnSubCycle.H*, the additional field `rhoPhiSum` is created in the case of subcycling. Furthermore, `totalDeltaT` is then set as the whole time step. Inside the subcycling or directly, if no subcycling is chosen, the file *alphaEqn.H* is included, which solves the α -transport equation for α . See Section 4.2 for a detailed description of the numerical solution. As result, we obtain the new values for `alpha1`, `alpha2`, `alphaPhi10` and `rhoPhi`. Subsequent, the density is updated with the new α -value. Directly after solving the α -transport equation, the first sensitivity equation is solved. With *dAlphaEqn.H* we include the $\delta\alpha$ -sensitivity equation. This position is selected to have access to the velocity and flux fields of the old time step, as they also occur in the primal α -transport equation. Note, to match the correct sensitivities, we had to simplify the solution of the α -transport equation to use only an upwind scheme for the convective part. This is similar to the corrector step within the MULES algorithm, although the MULES algorithm is not applied in this case. Furthermore, the compression term is neglected for now, since this causes instabilities when solving the associated sensitivity equation. Here, the linearization of the relative velocity u_r is the critical part. The simplification of the numerical solution of the α -transport equation leads to a slighty smearing out of the interface, but this is needed for the code to run stable and for the sensitivities to match the real derivatives in good consonance. Exactly as in *alphaEqn.H*, the convective terms in *dAlphaEqn.H* are solved with an upwind differencing scheme. As output we obtain the new `dAlpha` field for the actual time. Furthermore, the variables `dAlphaPhi` and `rhoDPhi` are calculated.

After solving the $\delta\alpha$ -sensitivity equation, other fields depending on α , such as the viscosities, are updated with the following expression

```
mixture.correct()
```

Then, the calculation of the momentum and continuity equation starts with including *UEqn.H*, where the coefficient matrices A and H are set up as in (4.29) and (4.32) inside the `fvVectorMatrix UEqn`. The PISO procedure itself will be carried out in *pEqn.H*. Herein, not only the pressure equation is solved, but also the corrector step is executed for the velocity. As result, the fields `p_rgh`, `phi`, `U` and `p` are calculated, see Section 4.2.2 for a detailed description of the PISO algorithm.

The next step is repeating the inner PIMPLE loop also for the momentum and continuity sensitivity equation. Again, this is similar to solving the corresponding primal equations and follows directly after the primal PISO loop terminates. In *dUEqn.H*, the derivatives of the temporal, the convective, the diffusion and the control term are set up. Therefore, we use the same operators as for the primal equations. In Table B.5, the sensitivity terms from the fvVectorMatrix dUEqn are set opposite to the corresponding terms in UEqn.

UEqn	dUEqn
fvm::ddt(rho,U)	gradRho*fvc::ddt(dAlpha, U) + fvm::ddt(rho, dU)
+ fvm::div(rhoPhi, U)	+ gradRho*fvc::div(dAlphaPhi, U) + fvc::div(rhoDPhi, U) + fvm::div(rhoPhi, dU)
+ turbulence-> divDevRhoReff(rho, U)	+ turbulence->divDevRhoReff(rho, dU) - gradMu*fvc::laplacian(dAlpha, U) - gradMu* fvc::div(dAlpha*dev2(T(fvc::grad(U)))) - fvc::laplacian(alpha1, U)*dMu - fvc::div(alpha1*dev2(T(fvc::grad(U))))*dMu

Table B.5. OpenFOAM implementation for *UEqn.H* and *dUEqn.H*

Here, two code terms without counterpart appear in dUEqn in comparison to UEqn. These represent the derivatives of the Navier-Stokes equations with respect to the control, as formally derived in equation (3.83). Since we want to treat the equations as similar as possible to the original implementation, also the compressible part of the diffusion term occurs. Therefore, the control term consists of two terms in the implementation.

When solving the dPEqn in *dPEqn.H*, the operators dUEqn.A() and dUEqn.H() appear, which coincides with the equations (4.39) and (4.40) respectively. They were used to create the volVectorField dHbyA, which is used to calculate the momentum predictor (4.26). The flux of this field is denoted by dPhiHbyA. Then, the right hand side of the momentum equation, hence the gravitational and surface tension term, is created in dPhig and added to dPhiHbyA, which is in turn used within the pressure equation. Altogether, we obtain dP_rgh, dPhi, dU and dP. For the terms in dPEq we obtain the following expression, presented in Table B.6.

pEqn	dPEqn
<code>- fvc::snGrad(p_rgh)</code>	<code>- fvc::snGrad(dP_rgh)</code>
<code>- ghf*fvc::snGrad(rho)</code>	<code>- gradRho*ghf*fvc::snGrad(dAlpha)</code>
<code>+ mixture.surfaceTensionForce()</code>	<code>+ (1/2e-9)*(surfaceTensionForceplus - surfaceTensionForceminus)</code>

Table B.6. OpenFOAM implementation for *pEqn.H* and *dPEqn.H*.

The pressure term in this Table B.6 is shown as the surface normal gradient of `p_rgh` and respectively of `dP_rgh`. Within the source code, the term only appears in this form when using a momentum predictor for the PISO loop, otherwise we use the Laplacian operator to calculate the pressure as well as the pressure sensitivity with the pressure equation, see equation (4.27). In Table B.6, the surface tension force term appears with the third term in `pEqn` as

```
mixture.surfaceTensionForce()
```

which is implemented in *interfaceProperties.H* with the following code line

```
fvc::interpolate(sigmaK())*fvc::snGrad(alpha1)
```

Here, `sigmaK()` is calculated by

```
sigmaK = sigma() * K
K = -fvc::div(nHatf)
nHatf = nHatfv & Sf
nHatfv = gradAlphaf/(mag(gradAlphaf) + deltaN)
gradAlphaf = fvc::interpolate(grad(alpha1))
deltaN = 1e-8/pow(average(alpha1.mesh().V()), 1.0/3.0)
```

where `sigma()` is the surface tension coefficient σ and `K` is the curvature κ of the interface. These calculations agree with the curvature model presented in Section 4.4.1. For the sensitivities, this terms need to be differentiated with respect to the phase fraction α , see equation (3.85). Here, a critical point is the derivative of the curvature, for which we need the derivative of the interface normal, denoted within the implementations as `dNHatf` and `dNHatfv`. The exact calculation has proven to be extremely difficult and is currently solved with the difference quotient.

Another point to mention is, that it is appropriate to relax the calculated `dU`- and `dP_rgh`-values in some cases, which is done with the following code line at the appropriate places.

```
dUEqn.relax()
...
dPEqn.relax()
```

In the end of *dPEqn.H*, the header file *dContinuityErrs.H* is included. This new implemented header file calculates and displays the quantities *dContErr*, *sumLocalDContErr*, *globalDContErr* and *cumulativeDContErr* analogously to the primal continuity errors and will not be explained in more detail here. With this, *dPEqn.H* ends and at the same time the PISO loops terminates. After writing the actual execution time, the calculations for the new time step start. This is executed until the final time is reached or any other termination criterion is achieved.

In Figure B.1, two further header files appear, *dUDiffQuot.H* and *dAlphaControls.H*. We will not explain them in detail, because they are not actually part of the current implementation. Both header files can be used to calculate the difference quotient of the state variables and were used for testing the sensitivities. They may be useful for geometrical optimization issues, so they are still part of the solver. However, they are commented out for the sensitivity calculations done within the mentioned optimization problems. Moreover, the interSensFoam solver is not yet adapted for mesh movement. For the primal equations, the respective solver parts are included, but the sensitivity fields still need to be adjusted at the appropriate places.

B.3 Compile Options and User Specifications

In the following, we will explain the usage of interSensFoam and summarize with which solver specifications the different possible optimization problems are solved. We start with a standard test case for interFoam. In Figure B.5, the general structure of a test case is shown, where we use the same color code as before. The files highlighted in red have been modified, the files highlighted in green are new.

Besides the initial fields for the phase fraction *alpha.water*, the velocity *U* and the modified pressure *p_rgh*, we also need initial fields for the sensitivities. As already mentioned, the sensitivity fields are denoted as *dAlpha*, *dU* and *dP_rgh* and are also stored within the *0* folder, see Figure B.5. Furthermore, we have to do some file modifications in the files *controlDict*, *fvSchemes* and *fvSolution* we will specify in the following.

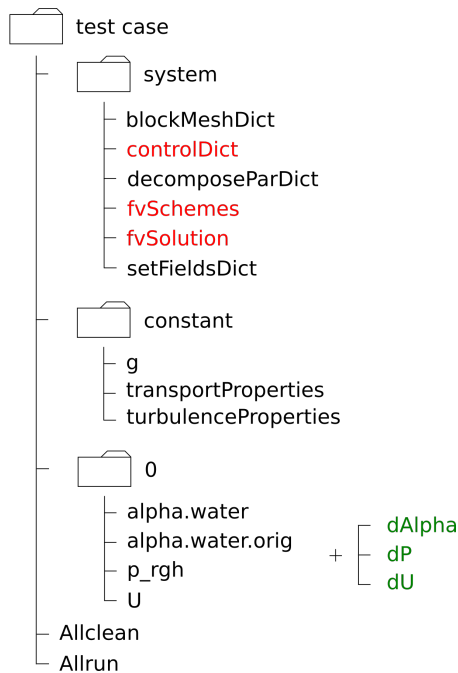


Figure B.5. Structure of a test case for `interSensFoam`.

File Modifications

The additional fields `dAlpha`, `dU` and `dP` are similar to the primal initial fields in content and structure. That means they have the same field class and therefore appropriate boundary conditions have to be defined for the single patches, see for example Table 5.5. However, we need to pay attention to the correct field denotation in the preamble and the correct choice of the dimension set. You can find the corresponding dimension in section B.1.

Within the file `controlDict`, the name of the application has to be adapted to `interSensFoam` and a careful choice of the time step size is recommendable. Furthermore, additional divergence schemes has to be added in `fvSchemes` for the convection terms. These are used to calculate the divergence of the single terms, by which the sensitivity equations were extended in `interSensFoam`. Hence, these modifications can be made once for all test cases calculated with the solver once. Other divergence schemes are also conceivable, but not yet tested in practice. The following additional entries have to be provided in the `divSchemes` sub dictionary.

```

divSchemes
{
    ...
    div(dPhi, alpha) Gauss vanLeer;
    div(phi, dAlpha) Gauss upwind;
    div(dAlphaPhi, U) Gauss linearUpwind grad(U);
    div(rhoPhi, dU) Gauss linearUpwind grad(dU);
    div(rhoDPhi, U) Gauss linearUpwind grad(U);
    div(((rho*nuEff)*dev2(T(grad(dU))))) Gauss linear;
    div((alpha.water*dev2(T(grad(U))))) Gauss linear;
    div((dAlpha*dev2(T(grad(U))))) Gauss linear;
}
    
```

Then we also have to define matrix solvers for the additional equations in *fvSolution*. The parameter settings within *fvSolution* should be chosen in dependence of the considered problem. Here, the matrix solver of the individual sensitivity equations are selected and the parameter and error tolerances are set. It is recommended to choose the same solvers and tolerances for the sensitivity variables as for the primal variables. Also the controls for the PIMPLE algorithm has to be set in this dictionary. It is sufficient to define one set of parameters for both the primal and the sensitivity loop.

In this work, we distinguish between the optimization with respect to a material parameter, more precisely the viscosity of the liquid phase, and with respect to geometrical aspects, which refer to the gap height and the inclination angle of the doctor blade. The resulting optimization problems differ in their objective function, calculated with Matlab, and in their control variables. Only for the optimization of the liquid viscosity, see Section 5.4.1, it is possible to calculate the sensitivities with the new solver interSensFoam. If the geometrical aspects are optimized, we have to calculate the derivatives with the difference quotient, since the corresponding shape derivatives are not implemented yet. This is also done with the Matlab framework with the original interFoam solver used for simulations. Note that two varying post processing routines are required for the different optimization objectives. If we are interested in an area behind the doctor blade, we also have to define a cell *Zone* with a *topoSetDict* in the folder system. Furthermore, we have to determine the values within this zone for the alpha.water field and the dAlpha field with the help of the function object volFieldValue, defined in the *controlDict*. If only the film thickness at a certain position is required, we use the file *singleGraph* in the system folder to calculate the respective quantity for every time step.

List of Symbols

General

Ω	Physical domain, composed of a liquid-phase and a gas-phase region Ω_l and Ω_g , $\Omega = \Omega_l \cup \Omega_g$
$\partial\Omega$	Boundary of the domain
Γ	Liquid-gas interface
I	Time horizon, $I = [0, T]$
ν	Interface normal
κ	Interface curvature
f	Total body force
S	Viscous stress tensor
α	Phase fraction
$\hat{\cdot}$	Transformed variables

Physical Quantities

u	$[\text{m s}^{-1}]$	Velocity
p	$[\text{kg m}^{-1} \text{s}^{-2}]$	Pressure
t	$[\text{s}]$	Time
ρ	$[\text{g m}^{-3}]$	Density
ϑ	$[\text{m}^2 \text{s}^{-1}]$	Kinematic viscosity viscosity
μ	$[\text{kg m}^{-1} \text{s}^{-1}]$	Dynamic viscosity
σ	$[\text{N m}^{-1}]$	Surface tension coefficient
g	$[\text{m s}^{-2}]$	Gravitational acceleration

Function Spaces

$C(\Omega)$	Space of continuous functions on Ω
$C^k(\Omega)$	Space of m times continuously differentiable functions on Ω
$C_0^\infty(\Omega)$	Space of infinitely many times continuously differentiable functions on Ω with compact support
$C^{k,\beta}(\Omega)$	Space of β -Hölder continuous functions on Ω
$BC(\Omega)$	Space of bounded continuous functions on Ω equipped with the supremum norm
$BUC(\Omega)$	Space of bounded uniformly continuous functions on Ω equipped with the supremum norm
$L^p(\Omega)$	Standard Lebesgue space on Ω
$L^\infty(\Omega)$	Lebesgue space of essentially bounded functions on Ω
$W^{k,p}(\Omega)$	Sobolev space on Ω
$W_0^{k,p}(\Omega)$	Closure of $C_0^\infty(\Omega)$ in $W^{k,p}(\Omega)$
$H^k(\Omega)$	Sobolev Hilbert space on Ω , short for $W^{k,2}(\Omega)$
$H_0^k(\Omega)$	Sobolev Hilbert space on Ω with homogeneous boundary conditions
$W^{s,p}(\Omega)$	Sobolev-Slobodeckij space on Ω
$H^s(\Omega)$	Sobolev-Slobodeckij Hilbert space on Ω , short for $W^{s,2}(\Omega)$
$H^{s,p}(\Omega)$	Bessel potential spaces
$\dot{H}^{1,p}(\Omega)$	Homogeneous Sobolev space
$L^p(I; X)$	Bochner-Lebesgue space
${}_0W^{s,p}(I; X)$	Sobolev-Slobodeckij space with homogeneous derivatives
$\mathcal{MR}^p(I; X)$	Class of maximal L^p -regularity operators
$\mathcal{M}_{loc}(X)$	Space of locally bounded Radon measures

Discretization Variables

Ω^h	Discretized domain
V	Arbitrary fixed control volume, $V \subset \Omega$
S_f	Surface area evaluated at the faces
ψ_c	Variable calculated at the cell center
ψ_f	Variable calculated at a cell face
ψ^o	Variable at the old time t
ψ^n	Variable at the actual time $t + \Delta t$
ϕ	Face flux, calculated from the velocity
c_α	Compression flux coefficient
λ_M	Phase fraction limiter calculated with MULES

Optimization Variables

j	Objective functional
y	State variable
q	Control variable
Q_{ad}	Set of admissible controls
u_0	Initial velocity field
c	Distributed control
d_{DB}	Thickness of the doctor blade
d_{GH}	Gap height between doctor blade and printing form
θ	Inclination angle of the doctor blade
μ_l	Viscosity of the liquid phase

Acronyms

BC	Boundary Condition
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrich-Lewy
CRC	Collaborative Research Centre
CSF	Continuum Surface Force
CV	Control Volume
DNS	Direct Numerical Simulation
FCT	Flux-Corrected Transport
FVM	Finite Volume Method
MULES	Multidimensional Universal Limiter for Explicit Solution
PCG	Preconditioned Conjugate Gradient (Method)
PDE	Partial Differential Equation
PISO	Pressure Implicit with Splitting of Operators
QOI	Quantity of Interest
SIMPLE	Semi-Implicit Method for Pressure Linked Equations
TPCL	Three Phase Contact Line
VOF	Volume of Fluid

Bibliography

- [1] H. Abels. On generalized solutions of two-phase flows for viscous incompressible fluids. *Interfaces and Free Boundaries*, 9:31–65, 2007.
- [2] F. Abergel and R. Temam. On some control problems in fluid mechanics. *Theoretical and Computational Fluid Dynamics*, 1(6):303–325, 1990.
- [3] R. A. Adams and J. Fournier. *Sobolev spaces*. Pure and applied mathematics. Academic Press, New York, 2nd edition, 2003.
- [4] L. Ambrosio. Transport equation and Cauchy problem for non-smooth vector fields. In: Calculus of variations and nonlinear partial differential equations. volume 1927 of *Lecture Notes in Math.*, pages 1–41. Springer, Berlin, 2008.
- [5] T. Anritter. *Numerical Simulation of Coupled Wetting and Transport Phenomena in Inkjet Printing*. PhD thesis, Technische Universität Darmstadt, 2022.
- [6] N. Ashgriz and J. Poo. Flair: Flux line-segment model for advection and interface reconstruction. *Journal of Computational Physics*, 93(2):449–468, 1991.
- [7] M. W. Baltussen, J. A. M. Kuipers, and N. G. Deen. A critical comparison of surface tension models for the volume of fluid method. *Chemical Engineering Science*, 109:65–74, 2014.
- [8] L. Bañas, M. Klein, and A. Prohl. Control of interface evolution in multiphase fluid flows. *SIAM Journal on Control and Optimization*, 52(4):2284–2318, 2014.
- [9] J. T. Beale. Large-time regularity of viscous surface waves. *Archive for Rational Mechanics and Analysis*, 84(4):307–352, 1984.
- [10] J. K. Biehl. *Adaptive multilevel optimization of fluid–structure interaction problems*. PhD thesis, Technische Universität Darmstadt, 2020.
- [11] T. Bitsch. *Experimente zum Strömungs- und Benetzungsverhalten von Fluiden im Hochscherratenbereich am Beispiel des Rakelprozesses*. PhD thesis, Technische Universität Darmstadt, 2021.
- [12] T. Bitsch, J. Schäfer, E. Diehl, H. M. Sauer, E. Dörsam, and S. Ulbrich. An

- experimental and numerical cooperative research concept for doctor blading. *Advances in Printing and Media Technology*, 46:68–74, 2019.
- [13] A. Björck. *Numerical methods for least squares problems*. SIAM, 1996.
- [14] H. Bonart. *Simulation and Control of Droplets and Films on Surfaces*. PhD thesis, Technische Universität Berlin, 2021.
- [15] J. P. Boris and D. L. Book. Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11(1):38–69, 1973.
- [16] D. Bothe, J. Prüss, and G. Simonett. Well-posedness of a Two-phase Flow with Soluble Surfactant. In *Nonlinear elliptic and parabolic problems*, pages 37–61. Springer, 2005.
- [17] M. Braack, M. Klein, A. Prohl, and B. Tews. Optimal control for two-phase flows. *Trends in PDE constrained optimization*, pages 347–363, 2014.
- [18] J. U. Brackbill, D. B. Kothe, and C. Zemach. A continuum method for modeling surface tension. *Journal of Computational Physics*, 100(2):335 – 354, 1992.
- [19] R. G. Cox. The dynamics of the spreading of liquids on a solid surface. Part 1. Viscous flow. *Journal of Fluid Mechanics*, 168:169–194, 1986.
- [20] S. M. Damián. *Description and utilization of interFoam multiphase solver*. Final Work-Computational Fluid Dynamics, 2010.
- [21] D. Deising. *Modelling and Numerical Simulation of Species Transfer in Bubbly Flows using OpenFOAM*. PhD thesis, Technische Universität Darmstadt, 2019.
- [22] I. Denisova. Problem of the motion of two viscous incompressible fluids separated by a closed free interface. *Acta Applicandae Mathematica*, 37(1):31–40, 1994.
- [23] I. Denisova and V. Solonnikov. Classical solvability of a problem on the motion of an isolated mass of a compressible liquid. *St Petersburg Mathematical Journal*, 14:71–98, 2003.
- [24] S. S. Deshpande, L. Anumolu, and M. F. Trujillo. Evaluating the performance of the two-phase flow solver interFoam. *Computational science & discovery*, 5:36 pp, 2012.
- [25] E. Diehl, J. Haubner, M. Ulbrich, and S. Ulbrich. Differentiability results and sensitivity calculation for optimal control of incompressible two-phase Navier-Stokes equations with surface tension. *Computational Optimization and Applications*, pages 1–41, 2022.
- [26] R. J. DiPerna and P.-L. Lions. Ordinary differential equations, transport theory and Sobolev spaces. *Invent. Math.*, 98(3):511–547, 1989.

- [27] J. Escher, J. Prüss, and G. Simonett. Analytic solutions for a Stefan problem with Gibbs-Thomson correction. *J. Reine Angew. Math.*, (563):1–52, 2003.
- [28] J. H. Ferziger, M. Perić, and R. L. Street. *Numerische Strömungsmechanik*. Springer, 2. edition, 2020.
- [29] O. Forster. *Analysis 3: Maß- und Integrationstheorie, Integralsätze im \mathbb{R}^n und Anwendungen*. Springer Spektrum, 2017.
- [30] M. Fricke. *Mathematical modeling and Volume-of-Fluid based simulation of dynamic wetting*. PhD thesis, Technische Universität Darmstadt, 2020.
- [31] A. V. Fursikov, M. D. Gunzburger, and L. Hou. Boundary value problems and optimal boundary control for the Navier-Stokes system: The two-dimensional case. *SIAM Journal on Control and Optimization*, 36(3):852–894, 1998.
- [32] H. Garcke, M. Hinze, and C. Kahle. Optimal control of time-discrete two-phase flow driven by a diffuse-interface model. *ESAIM: COCV*, 25:13, 2019.
- [33] J. Göhl, A. Mark, S. Sasic, and F. Edelvik. An immersed boundary based dynamic contact angle framework for handling complex surfaces of mixed wettabilities. *International Journal of Multiphase Flow*, 109:164–177, 2018.
- [34] V. R. Gopala and B. G. van Wachem. Volume of fluid methods for immiscible-fluid and free-surface flows. *Chemical Engineering Journal*, 141:204–221, 2008.
- [35] D. Gründing, M. Smuda, T. Anritter, M. Fricke, D. Rettenmaier, F. Kummer, P. Stephan, H. Marschall, and D. Bothe. Capillary rise – A Computational Benchmark for Wetting Processes. *arXiv preprint:1907.05054*, 2019.
- [36] M. D. Gunzburger. *Perspectives in Flow Control and Optimization*. SIAM, 2002.
- [37] D. S. Hariprasad, G. Grau, P. Randall Schunk, and K. Tjiptowidjojo. A computational model for doctoring fluid films in gravure printing. *Journal of Applied Physics*, 119(13), 2016.
- [38] F. H. Harlow 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.
- [39] J. Haubner. *Shape Optimization for Fluid-Structure Interaction*. PhD thesis, Technische Universität München, 2020.
- [40] M. Hinze and K. Kunisch. Second order methods for boundary control of the instationary navier-stokes system. *Journal of Applied Mathematics and Mechanics*, 84(3):171–187, 2004.
- [41] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE Constraints*. Springer Netherlands, 2009.
- [42] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39:201 – 225, 1981.

-
- [43] R. L. Hoffman. A study of the advancing interface. I. Interface shape in liquid-gas systems. *Journal of Colloid and Interface Science*, 50(2):228–241, 1975.
- [44] P. Hovland, B. Mohammadi, and C. Bischof. *Automatic Differentiation and Navier-Stokes Computations*, pages 265–284. Birkhäuser Boston, 1998.
- [45] C. Huh and L. Scriven. Hydrodynamic model of steady movement of a solid/liquid/fluid contact line. *Journal of Colloid and Interface Science*, 35:85–101, 1971.
- [46] R. I. Issa. Solution of the Implicitly Discretised Fluid Flow Equations by Operator-Splitting. *Journal of Computational Physics*, 62(1):40–65, 1986.
- [47] H. Jasak. *Error analysis and estimation for the finite volume method with applications to fluid flows*. PhD thesis, Imperial College London (University of London), 1996.
- [48] H. Jasak, H. Weller, and A. Gosman. High resolution NVD differencing scheme for arbitrarily unstructured meshes. *International journal for numerical methods in fluids*, 31(2):431–449, 1999.
- [49] H. Jasak and H. G. Weller. Interface-tracking capabilities of the intergamma differencing scheme. Technical report, Imperial College, University of London, 1995.
- [50] R. E. Johnson Jr., R. H. Dettre, and D. A. Brandreth. Dynamic contact angles and contact angle hysteresis. *Journal of Colloid and Interface science*, 62(2):205–212, 1977.
- [51] H. Kipphan. *Handbuch der Printmedien – Technologien und Produktionsverfahren*. Springer, 2000.
- [52] S. F. Kistler. Hydrodynamics of wetting. *Wettability*, 6:311–430, 1993.
- [53] J. Klostermann, K. Schaake, and R. Schwarze. Numerical simulation of a single rising bubble by VOF with surface compression. *Int. J. Numer. Meth. Fluids*, 71:960 – 982, 2013.
- [54] R. Kobayashi. Modeling and numerical simulations of dendritic crystal growth. *Physica D: Nonlinear Phenomena*, 63(3-4):410–423, 1993.
- [55] M. Köhne, J. Prüss, and M. Wilke. Qualitative Behaviour of Solutions for the Two-Phase Navier-Stokes Equations with Surface Tension. *Mathematische Annalen*, 356:737–792, 2013.
- [56] H. König. Ein einfacher Beweis des Gaußschen Integralsatzes. *Jahresbericht der DMV*, 66:119 – 138, 1964.
- [57] N. Linder. *Numerical Simulation of Complex Wetting*. PhD thesis, Technische Universität Darmstadt, 2015.
- [58] Q. Liu, F. Gómez, J. Perez, and V. Theofilis. Instability and sensitivity analy-

- sis of flows using OpenFOAM®. *Chinese Journal of Aeronautics*, 29(2):316–325, 2016.
- [59] T. Maric. *Lagrangian/Eulerian numerical methods for fluid interface advection on unstructured meshes*. PhD thesis, Technische Universität Darmstadt, 2017.
- [60] I. S. Mogilevskii and V. A. Solonnikov. On the solvability of a free boundary problem for the Navier-Stokes equations in the Hölder space of functions. *Nonlinear Analysis, Sc. Norm. Super. di Pisa Quaderni*, pages 257–271, 1991.
- [61] B. Mohammadi and O. Pironneau. *Applied Shape Optimization for Fluids*. Oxford University Press, 2009.
- [62] OpenCFD Limited. Openfoam programmer’s guide - v1606+. Technical report, 2016.
- [63] S. Osher 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.
- [64] C. Othmer. A continuous adjoint formulation for the computation of topological and surface sensitivities of ducted flows. *International Journal for Numerical Methods in Fluids*, 58(8):861–877, 2008.
- [65] C. Othmer, E. de Villiers, and H. Weller. Implementation of a continuous adjoint for topology optimization of ducted flows. In *18th AIAA Computational Fluid Dynamics Conference*, page 3947, 2007.
- [66] S. V. Patankar and D. B. Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *International Journal of Heat and Mass Transfer*, 15(10):1787–1806, 1972.
- [67] F. Poupaud and M. Rasle. Measure solutions to the linear multi-dimensional transport equation with non-smooth coefficients. *Comm. Partial Differential Equations*, 22(1-2):337–358, 1997.
- [68] J. Prüss and G. Simonett. Analytic solutions for the two-phase Navier-Stokes equations with surface tension and gravity. In *Parabolic problems*, pages 507–540. Springer, 2011.
- [69] J. Prüss and G. Simonett. *Moving Interfaces and Quasilinear Parabolic Evolution Equations*, volume 105. Springer, 2016.
- [70] J. Prüss and G. Simonett. On the two-phase Navier-Stokes equations with surface tension. *Interfaces Free Bound.*, 12:311 – 345, 2010.
- [71] T. Qian, X.-P. Wang, and P. Sheng. Molecular scale contact line hydrodynamics of immiscible flows. *Physical Review E*, 68:15 pp, 2003.
- [72] M. Renardy, Y. Renardy, and J. Li. Numerical Simulation of Moving Contact Line Problems Using a Volume-of-Fluid Method. *Journal of Computational Physics*, 171(1):243–263, 2001.

-
- [73] P. L. Roe. Characteristic-based schemes for the Euler equations. *Annual Review of Fluid Mechanics*, 18:337–365, 1986.
- [74] R. Roth. *Multilevel Optimization of Turbulent Flows by Discrete Adjoint Techniques*. PhD thesis, Technische Universität Darmstadt, 2012.
- [75] R. Roth and S. Ulbrich. A Discrete Adjoint Approach for the Optimization of Unsteady Turbulent Flows. *Flow, Turbulence and Combustion*, 90(4):763–783, 2013.
- [76] H. Rusche. *Computational fluid dynamics of dispersed two-phase flows at high phase fractions*. PhD thesis, Imperial College London, 2002.
- [77] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2003.
- [78] R. Scardovelli and S. Zaleski. Analytical relations connecting linear interfaces and volume fractions in rectangular grids. *Journal of Computational Physics*, 164(1):228–237, 2000.
- [79] S. Schmidt, C. Ilic, V. Schulz, and N. R. Gauger. Three-Dimensional Large-Scale Aerodynamic Shape Optimization based on Shape Calculus. *AIAA Journal*, 51(11):2615–2627, 2013.
- [80] B. Schweizer. Free Boundary Fluid Systems in a Semigroup Approach and Oscillatory Behavior. *SIAM Journal on Mathematical Analysis*, 28(5):1135–1157, 1997.
- [81] B. Schweizer. *Partielle Differentialgleichungen*. Springer, 2013.
- [82] J. A. Sethian and P. Smereka. Level Set Methods for Fluid Interfaces. *Annual Review of Fluid Mechanics*, 35(1):341–372, 2003.
- [83] Y. Shibata and S. Shimizu. Report on a local in time solvability of free surface problems for the Navier-Stokes equations with surface tension. *Applicable Analysis*, 90(1):201–214, 2011.
- [84] Y. Shikhmurzaev. The moving contact line on a smooth solid surface. *International Journal of Multiphase Flow*, 19:589–610, 1993.
- [85] Y. D. Shikhmurzaev. *Capillary Flows with Forming Interfaces*. Chapman & Hall/CRC, 2008.
- [86] S. Shimizu. Local solvability of free boundary problems for the two-phase Navier-Stokes equations with surface tension in the whole space. In *Parabolic problems*, pages 647–686. Springer, 2011.
- [87] V. A. Solonnikov. Lectures on evolution free boundary problems: classical solutions. In *Mathematical aspects of evolving interfaces*, pages 123–175. Springer, 2003.
- [88] Y. Sui and P. D. Spelt. An efficient computational model for macroscale simulations of moving contact lines. *Journal of Computational Physics*, 242:37–52, 2013.

- [89] M. Sussman, P. Smereka, and S. Osher. A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow. *Journal of Computational Physics*, 114(1):146–159, 1994.
- [90] N. Tanaka. Two-phase free boundary problem for viscous incompressible thermo-capillary convection. *Japanese Journal of Mathematics. New series*, 21(1):1–42, 1995.
- [91] R. Temam. *Navier-Stokes Equations: Theory and numerical analysis*. North-Holland Publishing, 1977.
- [92] F. Tröltzsch. *Optimale Steuerung partieller Differentialgleichungen*, volume 2. Springer, 2005.
- [93] G. Tryggvason, R. Scardovelli, and S. Zaleski. *Direct Numerical Simulations of Gas-Liquid Multiphase Flows*. Cambridge University Press, 2011.
- [94] O. Ubbink. *Numerical prediction of two fluid systems with sharp interfaces*. PhD thesis, Imperial College, London, 1997.
- [95] M. Ulbrich. Constrained optimal control of navier–stokes flow by semismooth newton methods. *Systems & Control Letters*, 48(3-4):297–311, 2003.
- [96] M. Ulbrich and B. v. Bloemen Waanders. An introduction to partial differential equations constrained optimization. *Optimization and Engineering*, 19:515–520, 2018.
- [97] S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multi-fluid flows. *Journal of Computational Physics*, 100(1):25–37, 1992.
- [98] B. van Leer. Towards the Ultimate Conservative Difference Scheme. II. Monotonicity and Conservation Combined in a Second-Order Scheme. *Journal of Computational Physics*, 14:361–370, 1974.
- [99] B. Van Wachem and J. Schouten. Experimental validation of 3-D lagrangian VOF model: Bubble shape and rise velocity. *AIChE Journal*, 48:2744–2753, 2002.
- [100] X. Wang, X. Peng, Y. Duan, and B. Wang. Dynamics of Spreading of Liquid on Solid Surface. *Chinese Journal of Chemical Engineering*, 15(5):730–737, 2007.
- [101] H. Weller. A new approach to vof-based interface capturing methods for incompressible, compressible and cavitating flow. In *Technical Report*. OpenCFD Limited, 2006.
- [102] D. Werner. *Funktionalanalysis*. Springer, 2018.
- [103] X. Xu, Y. Di, and H. Yu. Sharp-interface limits of a phase-field model with a generalized Navier slip boundary condition for moving contact lines. *Journal of Fluid Mechanics*, 849:805–833, 2018.

- [104] T. Young. An essay on the cohesion of fluids. *Philosophical Transactions of the Royal Society of London*, 95:65–87, 1805.
- [105] S. T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31(3):335–362, 1979.
- [106] E. Zeidler. Nonlinear Functional Analysis and Its Applications. I, Fixed-Point Theorems. *Springer-Verlag Berlin*, 1986.

Wissenschaftlicher Werdegang

Elisabeth Andrea Gertrud Diehl
geboren am 24. September 1990 in Heppenheim.

- 07/2016 - 09/2023 **Wissenschaftliche Mitarbeiterin** am Fachbereich Mathematik der Technischen Universität Darmstadt im Rahmen des DFG Sonderforschungsbereiches 1194, Teilprojekt B04 „Simulationsbasierte Optimierung und Optimales Design von Experimenten für Benetzungsvorgänge“
- 10/2013 - 04/2016 **Master of Science** in Mathematik an der Technischen Universität Darmstadt
- 08/2014 - 09/2014 **Praktikum** am Institute of Rock Structure and Mechanics an der Akademie der Wissenschaften der Tschechischen Republik in Prag
- 04/2010 - 01/2017 **Bachelor of Science** in Angewandten Geowissenschaften an der Technischen Universität Darmstadt
- 10/2009 - 03/2013 **Bachelor of Science** in Mathematik an der Technischen Universität Darmstadt
- 06/2009 **Abitur** am Alten Kurfürstlichen Gymnasium in Bensheim