igpm Institute for Geometry and Applied Mathematics | RWTH AACHEN UNIVERSITY

# Surface Navier–Stokes Equations: Numerical Methods and Analysis

Von der Fakultät für Mathematik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

**Paul Schwering, M.Sc.**
aus Aachen

Berichter: Univ.-Prof. Dr. Arnold Reusken
Univ.-Prof. Dr. Maxim A. Olshanskii

Tag der mündlichen Prüfung: 16.12.2025

**Eidesstattliche Erklärung**

Paul Schwering

erklärt hiermit, dass diese Dissertation und die darin dargelegten Inhalte die eigenen sind und selbstständig, als Ergebnis der eigenen originären Forschung, generiert wurden.

Hiermit erkläre ich an Eides statt

1. Diese Arbeit wurde vollständig oder größtenteils in der Phase als Doktorand dieser Fakultät und Universität angefertigt;
2. Sofern irgendein Bestandteil dieser Dissertation zuvor für einen akademischen Abschluss oder eine andere Qualifikation an dieser oder einer anderen Institution verwendet wurde, wurde dies klar angezeigt;
3. Wenn immer andere eigene- oder Veröffentlichungen Dritter herangezogen wurden, wurden diese klar benannt;
4. Wenn aus anderen eigenen- oder Veröffentlichungen Dritter zitiert wurde, wurde stets die Quelle hierfür angegeben. Diese Dissertation ist vollständig meine eigene Arbeit, mit der Ausnahme solcher Zitate;
5. Alle wesentlichen Quellen von Unterstützung wurden benannt;
6. Wenn immer ein Teil dieser Dissertation auf der Zusammenarbeit mit anderen basiert, wurde von mir klar gekennzeichnet, was von anderen und was von mir selbst erarbeitet wurde;
7. Teile dieser Arbeit wurden zuvor veröffentlicht und zwar in:

   Philip Brandner, Arnold Reusken, and Paul Schwering. On derivations of evolving surface Navier–Stokes equations. Interfaces and Free Boundaries, 24:533-563, 2022, doi:10.4171/IFB/483.

   Maxim A Olshanskii, Arnold Reusken, and Paul Schwering. An Eulerian finite element method for tangential Navier–Stokes equations on evolving surfaces. Mathematics of Computation, 93(349):2031-2065, 2024, doi:10.1090/mcom/3931.

   Maxim A Olshanskii, Arnold Reusken, and Paul Schwering. A Narrow Band Finite Element Method for the Level Set Equation. SIAM Journal on Scientific Computing, 47(2):A762-A789, 2025, doi:10.1137/24m1674182.

Datum

Unterschrift

# Acknowledgments

First, I express my gratitude to the German Research Foundation (DFG) for their financial support through the Research Unit "Vector- and tensor valued surface PDEs" (FOR 3013), project no. RE 1461/11-2, as well as to RWTH Aachen University for providing computing resources under project rwth1817.

I am deeply thankful to my supervisor, Professor Arnold Reusken, for his invaluable guidance, attentive listening, and constructive feedback throughout my doctoral studies. His support has been instrumental in my learning experience during our collaboration. I also extend my thanks to my co-advisor, Professor Maxim Olshanskii, for his contributions related to this thesis and for giving me the opportunity to visit the University of Houston. He made it an enriching experience. My gratitude goes out as well to Professor Christoph Lehrenfeld and his team for their ongoing support with the `Netgen/NGSolve` code through the add-on package `ngsxfem`.

I thank my colleagues at the IGPM ("Institut für Geometrie und Praktische Mathematik"), particularly those from LNM ("Lehrstuhl für Numerische Mathematik"), for fostering a motivating work environment and engaging discussions. The unforgettable Carnival parties, regular tables, and action-packed leisure activities will always be cherished memories. A special thanks to our admin Frank for his tireless support and hours spend to keep everything running smoothly and to David, Fabian, Nam, Nibedita, Sophie, Sven, and Tilman for proofreading my thesis.

Finally, I thank my family and friends, especially my wife Chrissy, for their unwavering support over the years. You have patiently endured my mood swings on days when frustration took over (and I had to hit my desk occasionally). You are truly the best!

# Abstract

This thesis addresses the numerical treatment of Navier–Stokes equations on evolving surfaces, combining advanced concepts from surface fluid modeling, trace finite element approximations, and level set equation discretization schemes. Motivated by applications in biological membranes and thin film flows, the work focuses on the numerical simulation of surface fluid dynamics. Two principal systems are considered: the tangential surface Navier–Stokes system (TSNS), where the evolving surface is prescribed, and the full surface Navier–Stokes system (SNS), where both the surface evolution and the flow are unknowns. A major challenge in the SNS arises from the nonlinear coupling between the velocity field and the surface geometry evolution.

The thesis provides a comprehensive review and comparison of existing derivations of the surface Navier–Stokes equations, highlighting differences in physical principles (such as conservation laws, thin film limits, and energetic variational approaches) and coordinate representations (local curvilinear vs. global Cartesian). It is shown that, while the TSNS systems derived from various approaches are equivalent, the SNS systems may differ depending on the chosen framework.

The main goal of this thesis is the development and analysis of novel numerical methods based on the Trace Finite Element Method (TraceFEM) for both TSNS and SNS on evolving surfaces. TraceFEM uses a background mesh in the embedding space, allowing for robust discretization without the need for surface triangulation or remeshing as the surface evolves. The thesis details the construction of these methods, including stabilization and extension techniques. For the TSNS, an error analysis is provided, and the method's accuracy and stability are demonstrated. The extension to the full SNS system is presented, addressing the additional challenges posed by the nonlinear coupling between surface evolution and fluid flow. Additionally, an efficient narrow band method for the level set equation is introduced to accurately represent surfaces evolved by material flow fields. This method is crucial for the SNS system, where the surface evolution is coupled with the fluid flow.

Numerical experiments validate the theoretical findings, demonstrating the accuracy and efficiency of the proposed methods. This work contributes to computational fluid dynamics by advancing methodologies for discretizing fluid flows on deformable surfaces, paving the way for future research in this vital area of applied mathematics.

# Zusammenfassung

Diese Dissertation befasst sich mit der numerischen Behandlung von Navier–Stokes–Gleichungen auf sich bewegenden Oberflächen, wobei Konzepte aus der Modellierung von Oberflächenströmungen mit Trace–Finite–Elemente–Methoden und Diskretisierungsverfahren für Level–Set–Gleichungen kombiniert werden. Motiviert durch Anwendungen in biologischen Membranen, liegt der Schwerpunkt der Arbeit auf der numerischen Simulation von Strömungen auf deformierbaren Oberflächen. Es werden zwei Systeme betrachtet: das tangentiale Oberflächen–Navier–Stokes–System (TSNS), bei dem die Oberflächenbewegung vorgegeben ist, und das vollständige Oberflächen–Navier–Stokes–System (SNS), bei dem sowohl die Bewegung der Oberfläche als auch das Geschwindigkeitsfeld auf der Oberfläche unbekannt sind. Eine wesentliche Herausforderung im SNS ergibt sich aus der nichtlinearen Kopplung zwischen Geschwindigkeitsfeld und Entwicklung der Oberflächengeometrie.

Die Arbeit bietet einen umfassenden Überblick und Vergleich bestehender Herleitungen des SNS, wobei unterschiedliche physikalische Prinzipien (Erhaltungssätze, Dünnschicht-Grenzfälle und energetische Variation) sowie Koordinatensysteme (lokal vs. global) zum Zuge kommen. Es wird gezeigt, dass die aus verschiedenen Modellierungsansätzen hergeleiteten TSNS äquivalent sind, während die SNS voneinander abweichen können.

Das Hauptziel der vorliegenden Arbeit besteht in der Entwicklung und Analyse numerischer Methoden auf Grundlage der Trace–Finite–Elemente–Methode (TraceFEM). TraceFEM verwendet ein Gitter im $\mathbb{R}^3$, das eine Diskretisierung ermöglicht, ohne dass eine Oberflächentriangulierung erforderlich ist. Die Konstruktion der Methoden, einschließlich Stabilisierungs- und Erweiterungstechniken, wird detailliert beschrieben. Für das TSNS wird eine Fehleranalyse durchgeführt, welche die Genauigkeit der Methode nachweist. Eine Erweiterung auf das SNS wird vorgestellt, wobei die zusätzlichen Herausforderungen (Kopplung zwischen Oberflächenverformung und Strömung) berücksichtigt werden. Darüber hinaus wird ein effizientes Narrow–Band–Verfahren für die Level–Set–Gleichung eingeführt, um die Deformation der Oberfläche darzustellen.

Numerische Experimente validieren die theoretischen Ergebnisse und demonstrieren die Genauigkeit und Effizienz der Methoden. Diese Arbeit leistet einen Beitrag zur numerischen Strömungsmechanik, indem sie Methoden zur Diskretisierung von Strömungen auf verformbaren Oberflächen weiterentwickelt und analysiert und damit den Weg für zukünftige Forschung in diesem Bereich ebnet.

# Contents

# Chapter 1

# Introduction

This doctoral thesis explores the discretization of *Navier-Stokes equations* on evolving surfaces. The work combines advanced concepts from modeling of surface fluids, trace finite element approximations, and discretization schemes for level set equations to develop novel computational techniques for discretizing fluid flows on evolving manifolds embedded in $\mathbb{R}^3$.

## 1.1 Motivation and overview

In recent years, there has been a strongly growing interest in partial differential equations (PDEs) modeling fluid flows on surfaces. This is motivated by a wide range of applications in science and engineering, including biological membranes, surfactant dynamics, and thin film flows. Especially in cell and tissue biology, fluid deformable surfaces play an important role to model membranes and interfaces. For example, lipid membranes and liquid crystal shells are deformable thin structures exhibiting fluidity, cf., e.g., [Dim+06; Keb+14]. Continuum based modeling of such materials leads to systems of partial differential equations posed on two-dimensional surfaces embedded in $\mathbb{R}^3$. A groundbreaking study on the mathematical modeling of surface fluids was done in [Scr60]. Many proposed models for fluidic surfaces available in the literature involve variants of the surface (Navier-)Stokes equations. The surface Navier–Stokes equations are a fundamental model in fluid dynamics, describing the motion of fluids along surfaces or interfaces. Accurately modeling the fluid behavior is crucial for understanding a wide range of physical phenomena, for example in the modeling of biological interfaces, cf. the overview paper [TMA19] and the references therein. Additionally, the surface Navier–Stokes equations are of great interest in the field of computational fluid dynamics, where they can be used to simulate complex fluid-structure interactions and multiphase flows. These equations are also explored as intriguing mathematical problems in their own right (cf. refer to [EM70; Tay92; MT01]). Various modeling strategies and derivations of Navier–Stokes equations on evolving surfaces have been introduced (cf., e.g., [AD09; EHZ07; GKL17; JOR18; Miu17; NRV19; NRV20]). We take a closer look at existing derivation techniques for the surface Navier–Stokes equations and compare the techniques and the resulting equations in Chapter 3.

When considering the surface Navier–Stokes equations, we distinguish between the *tangential surface Navier–Stokes system* (TSNS) and the (full) *surface Navier–Stokes system* (SNS). While the surface is known beforehand in the TSNS (the surface may be stationary as well), the evolving surface becomes an unknown of the system in SNS and only the initial surface is known. In the TSNS, the normal component of the velocity (that defines the surface evolution) is known and the unknowns of the system are the tangential velocity and the pressure. When the surface is stationary (not evolving in time), the normal component of the velocity has to be zero. Thus, only the TSNS can be posed on surfaces, where the surface evolution is known. In the SNS the full flow field (tangential and normal component), the pressure, and the surface evolution are the unknowns of the system. Since the velocity is a so-called material velocity, the movement of the surface depends on the velocity field, defined by the PDE on the surface. This nonlinear dependence between the surface, where the PDE is defined, and the velocity field, that defines the surface evolution, makes the system challenging to analyse and discretize. To the best of our knowledge there is no well-posedness result or numerical analysis of the full surface Navier–Stokes problem.

There are multiple discretization methods for partial differential equations defined on surfaces; examples include the *trace finite element method* (TraceFEM), cf. e.g., [OR17], the surface finite element method (SFEM), cf. e.g., [DE13], and the diffuse interface method, cf. e.g., [RV06]. For scalar surface partial differential equations, like the Laplace-Beltrami problem, established finite element techniques and analyses of discretization errors are available (see [DD07; Reu14; GLR18]). An overview of finite element methods applied to scalar surface partial differential equations is given in [DE13; BDN20]. Also, vector-valued surface PDEs, like the surface vector–Laplace system have been studied, cf. e.g., [JR20]. Numerical simulation techniques for the TSNS and the Stokes system on a stationary manifold have gathered significant attention in recent years, cf. e.g., [Fri18; Ols+18; BR20; BDL20; OY19]. Some of these techniques were adapted to the case of evolving surfaces. In [KKV23; ORZ22] the TSNS system on an evolving (predefined) surface is discretized. However, the analysis of many of these techniques is still an open problem. A first step was done in [ORZ22], where it is shown, that the TSNS system on an evolving surface is well-posed.

Only a few numerical methods for the full surface Navier–Stokes equations (SNS) have been proposed in the literature, cf. e.g., [NRV19; RNV20; KV23]. The main challenges in this context are the coupling between surface geometry and fluid dynamics, the time-dependent domains of definition, and the discretization of geometric quantities.

The goal of this thesis is the development of a novel *numerical method for the surface Navier–Stokes equations*. We present a TraceFEM based discretization schemes for the TSNS and for the SNS and give an overview of an error analysis for the TSNS method.

## 1.2 Navier–Stokes equations on evolving surfaces

In Chapter 3, we give a short introduction into the modeling of surface Navier–Stokes equations posed on evolving surfaces $\Gamma(t)$. There are different physical principles used in the literature to derive surface Navier–Stokes systems. In this thesis, we will compare different derivations of the surface Navier–Stokes equations known from the literature and discuss the differences and similarities between them. We discuss the derivations

given in [JOR18; GKL17; EHZ07; Miu17; NRV19]. In all derivations, the surface is assumed to be closed, orientable, and sufficiently smooth.

The derivations in the literature can be classified by the physical principles used. We distinguish between the following approaches:

- It is possible to obtain surface Navier–Stokes equations as a system of PDEs on a moving material manifold by using *conservation laws of mass and momentum on the surface*. This approach is used in [JOR18; EHZ07].

- Similar to the first approach, one can derive Navier–Stokes equations from *conservation laws of mass and momentum in the volume*. This leads to a system of PDEs in a narrow band around the surface. Combing this system with a *thin film limit* technique leads to a Navier–Stokes system on the surface. This approach is used in [Miu17; NRV19].

- The conservation of surface mass is also used in the last physical principle. But instead of conservation of momentum, an Onsager principle is used to combine the *variation of an action integral with the variation related to a "surface viscosity" energy*, to obtain the surface Navier–Stokes equations. This approach is used in [GKL17].

Besides these differences in physical principles, there is also a difference in the mathematical representation of the resulting flow equations. In some papers, e.g., [EHZ07; AD09; NRV19; NRV20], local coordinate systems (*curvilinear coordinates*) are used, whereas in other literature [JOR18; GKL17; Miu17] the standard *Euclidean basis* of $\mathbb{R}^3$, in which the evolving surface is embedded, is used. Such different coordinate systems lead to different representations of surface differential operators such as a covariant derivative or a surface divergence, and one has to be careful when comparing equations formulated in such different coordinate systems. Both, the local curvilinear and the global Cartesian coordinate system have attractive properties according to different application fields. The local coordinate system can be very useful for modeling more complex fluid properties, e.g., in certain classes of fluid membranes, cf. [EHZ07; TMA19], or in flows of liquid crystals, cf. [NRV19; NRV20]. The representation in global Cartesian coordinates is very convenient for the development of numerical simulation methods for these flow equations. In Chapter 2 we will give a detailed introduction to the different coordinate systems, derive representations of the surface differential operators in both coordinate systems and compare them as a preparation for the comparison of the systems of Navier–Stokes equations. In Chapter 3, we will summarize five derivations of the surface Navier–Stokes equations from [JOR18; GKL17; EHZ07; Miu17; NRV19]. We will discuss the main ingredients and steps of the derivations and compare the resulting systems. Furthermore, we use a splitting of the velocity to derive the simplified *tangential Navier–Stokes* systems from the surface Navier–Stokes equations for each approach. We show that the TSNS systems of all five derivations are the same (besides their representation), but the full surface Navier–Stokes systems may differ.

The (full) surface Navier–Stokes system, that we will study throughout this thesis, is given by the following problem.

**Problem 1.2.1: Surface Navier–Stokes equations**

For given initial velocity $\mathbf{u}^0$, initial surface $\Gamma^0 \subset \mathbb{R}^3$, viscosity coefficient $\mu_0$, time interval $[0, t_{\text{end}}]$, and force term $\mathbf{f}$, find the velocity $\mathbf{u}$, the pressure $p$, and the surface parametrization $\mathbf{X}$ satisfying

$$\begin{cases} \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} = \mathbf{f} & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t), \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{x}, t) = \mathbf{u}(\mathbf{X}(\mathbf{x}, t), t), \quad \mathbf{X}(\mathbf{x}, 0) = \mathbf{x} \in \Gamma^0, \end{cases}$$

with $\Gamma(t) = \mathbf{X}(\Gamma^0, t)$.

When the evolution of the surface $\Gamma(t)$ is known, also the normal component of the velocity is (implicitly) prescribed. For this case, we apply the tangential projection $\mathbf{P}$ to the system, leading to a system defining the tangential component $\mathbf{u}_T$ of the velocity. This leads to the *tangential surface Navier–Stokes equations* (TSNS). The TSNS system is given by the following problem.

**Problem 1.2.2: Tangential surface Navier–Stokes equations**

For given initial value $\mathbf{u}_T^0$, viscosity coefficient $\mu_0$, time interval $[0, t_{\text{end}}]$, surface $\Gamma(t)$, normal velocity $u_N$, and force term $\mathbf{f}_T$, find $\mathbf{u}_T$ and $p$ satisfying

$$\begin{cases} \mathbf{P}\dot{\mathbf{u}}_T - 2\mu_0 \mathbf{P} \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}_T) + \nabla_\Gamma p + u_N \mathbf{H}\mathbf{u}_T = \mathbf{f}_t & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u}_T = f_{\text{div}} & \text{on } \Gamma(t), \\ \mathbf{u}_T \cdot \mathbf{n} = 0 & \text{on } \Gamma(t), \end{cases}$$

with $\mathbf{f}_t = \mathbf{f}_T + 2\mu_0 \mathbf{P} \operatorname{div}_\Gamma (u_N \mathbf{H}) + u_N \nabla_\Gamma u_N$ and $f_{\text{div}} = -u_N \kappa$.

We give an overview of notations that will be used throughout the whole thesis. The main unknowns of the systems are

- $\mathbf{u}(\mathbf{x}, t)$: (vector-valued) velocity with tangential and normal component ($\mathbf{u}_T$ and $u_N$),
- $p(\mathbf{x}, t)$: (scalar) pressure function,
- $\Gamma(t)$: surface at time $t$ (the surface is a known quantity in the TSNS).

Geometric quantities are given by

- $\mathbf{n}(\mathbf{x}, t)$: (vector–valued) outward pointing normal vector of the surface,
- $\mathbf{P}(\mathbf{x}, t)$: (tensor–valued) projection operator onto the tangent space of the surface,
- $\mathbf{H}(\mathbf{x}, t)$: (tensor–valued) shape operator of the surface,
- $\kappa(\mathbf{x}, t)$: (scalar) mean curvature of the surface.

The input quantities are given by

- $\mu_0$: viscosity coefficient,
- $\mathbf{f}$: force term with tangential component $\mathbf{f}_T$,
- for SNS:
  - $\mathbf{u}^0$, $\Gamma^0$: initial values of the velocity and the surface at time $t = 0$,
- for TSNS:
  - $\mathbf{u}_T^0$: initial value of the tangential velocity,
  - $u_N$ and $\Gamma(t)$: normal component of the velocity and surface for all times $t$.

The surface differential operators are given by

- $\nabla_\Gamma$: surface gradient operator,
- $\mathrm{div}_\Gamma$: surface divergence operator,
- $\mathbf{E}_\Gamma$: surface rate of strain rate tensor,
- $\dot{\mathbf{v}}$: material derivative of $\mathbf{v}$.

Definitions of geometric quantities, surface differential operators and their relations are given in Chapter 2.

## 1.3 Discretization methods

Developing robust numerical methods to simulate Navier–Stokes systems on evolving manifolds requires careful consideration of the structure of the equations, especially the coupling between the surface geometry evolution and the PDE on the surface, defining the velocity. This thesis aims at advancing the state-of-the-art in surface Navier-Stokes solvers through an innovative algorithmic approach. The main goal of this thesis is to develop a numerical method for the surface Navier–Stokes equations based on a TraceFE method. In Chapter 4, we summarize the main contributions to the TraceFE method from the literature. In Chapter 5 we start with a discretization scheme for the tangential surface Navier–Stokes equations and extend this approach for the (full) surface Navier–Stokes equations in Chapter 7.

TraceFEM is a finite element method, based on finite element spaces $V_{h,k}$, that are constructed on a bulk mesh surrounding the surface. Thus, no surface triangulation is needed, but a triangulation $\mathcal{T}_h$ of a domain $\Omega \subset \mathbb{R}^3$ that includes the surface $\Gamma \subset \Omega$ is used. In the case of an evolving surface, TraceFEM has the advantage that no remeshing needs to be done since the surface is allowed to cut through the background mesh arbitrarily. Functions from $V_{h,k}$ are defined on the elements that are "cut" by the surface, denoted by

$$\mathcal{T}_\Gamma := \left\{ T \in \mathcal{T}_h \mid T \cap \Gamma \neq \emptyset \right\},$$

and not only on the surface itself. Thus, TraceFEM belongs to the class of *geometrically unfitted* methods. In variational formulations of the surface PDE, we take the traces of

functions from $V_{h,k}$ on the surface $\Gamma$ and do not extend the PDE onto the cut elements. This motivates the name TraceFEM.

Since the mesh is independent of the surface, it is not aligned and arbitrary cuts between mesh and surface may occur. This induces instabilities that can be overcome by adding stabilization terms to the variational formulations. We will use standard stabilization techniques from the literature, e.g., a volume normal derivative stabilization.

To employ a TraceFEM discretization, the surface has to be represented implicitly via a *level set function*. A level set function is a scalar, sufficiently smooth (in space and time) function $\phi : \Omega \times [0, t_{\text{end}}]$. The surface $\Gamma(t)$ is then given by the zero level set of $\phi$. It holds

$$\Gamma(t) = \{\mathbf{x} \in \mathbb{R}^3 \mid \phi(\mathbf{x}, t) = 0\}.$$

Using a sufficiently accurate finite element approximation $\phi_h$ of $\phi$ and an isoparametric mapping to faciliate integration over curved surface patches, cf. [Leh16], allows approximating integrals over $\Gamma(t)$ with high accuracy.

TraceFEM is a well-known numerical method for discretization of PDEs on stationary surfaces. It was for example used and analysed in [ORG09; OR17]. There are some more recent works where TraceFEM is used to discretize scalar PDEs on evolving surfaces, cf. e.g., [ORX14; LOX18]. Also, a variant of the TraceFEM has recently been applied to time-dependent Stokes equations on a moving domain, cf. [WRL21; BFM22]. For this reason, Taylor-Hood finite elements or equal order finite element spaces combined with a continuous interior penalty pressure stabilization are applied for space discretization. We adapt these methods to vector-valued PDEs on evolving surfaces.

When the surface is evolving, the discretization of the time derivative becomes more delicate. Time stepping schemes need the evaluation of the difference $\mathbf{u}(t_2) - \mathbf{u}(t_1)$, where $t_1$ and $t_2$ are two different time points. This is in general not possible in the case of evolving surfaces, since the surfaces $\Gamma(t_1)$ and $\Gamma(t_2)$ differ and the velocity $\mathbf{u}(t)$ is only defined on $\Gamma(t)$ by the PDE. In [LOX18], an implicit extension by a stabilization term is used to construct an approximation to $\mathbf{u}(t_1)$ that is well defined on the subsequent surface $\Gamma(t_2)$. We use this approach to obtain a well defined TraceFEM formulation for the Navier–Stokes system on an evolving surface. This extension is used in the TSNS and the SNS.

As described above, the surface $\Gamma(t)$ is an unknown in the SNS system. In Problem 1.2.1, the surface is defined by the parametrization $\mathbf{X}(\mathbf{x}, t)$, which depends on the velocity $\mathbf{u}$ on the surface. In our TraceFEM setting, we replace the third equation of the SNS system ($\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{x}, t) = \mathbf{u}(\mathbf{X}(\mathbf{x}, t), t)$) by a level set equation. The level set equation describes the evolution of the surface in terms of the corresponding level set function. It is given by

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot \mathbf{u} = 0.$$

The level set equation is a pure transport equation, which depends on the velocity field $\mathbf{u}$. In contrast to the equation defining the parametrization $\mathbf{X}$, the level set equation cannot be solved on the surface only. Since the level set function has to be defined on a neighborhood of $\Gamma(t)$, the transport equation also has to be solved on a narrow band around the surface. We denote such a sufficiently large neighborhood by $\Omega_{\text{Tr}} \subset \Omega$. To construct a well defined level set transport problem, the velocity field $\mathbf{u}$ has to be

defined on $\Omega_{\mathrm{Tr}}$ from velocity data on $\Gamma(t)$. As described in Chapter 7, we use an extension algorithm to obtain a suitable velocity field on $\Omega_{\mathrm{Tr}}$. Additionally, we need initial values for the level set function on $\Omega_{\mathrm{Tr}}$ and Dirichlet boundary values on the inflow boundary $\partial\Omega_{\mathrm{Tr}}^{\mathrm{in}} \subset \partial\Omega_{\mathrm{Tr}}$ for each time step. To obtain these values, we need to extend the level set function from the transport domain (the narrow band) of the previous time step to the transport domain of the current time step. We develop a new *extension method* based on a ghost penalty technique. This extension allows us to solve the level set equation in a narrow band around the surface with high accuracy even if only low order boundary values can be constructed.

There are different methods for solving the transport equation on $\Omega_{\mathrm{Tr}}$ with given initial and boundary values. We will use a *discontinuous Galerkin* scheme from [BDF16] for the spatial discretization of the level set equation. DG-schemes are widely used to approximate hyperbolic PDEs like the level set equation, since they need no stabilization and allow for higher order accuracy, cf. e.g., [BMS04; DF15; BQS10]. The full narrow-band transport method, including the extension of the level set function, an error analysis of this extension method, and a method to construct boundary values, is treated in Chapter 6.

To discretize the time derivative in the Navier–Stokes equations and in the level set equation, one can use different approaches from the literature. For example, it is possible to apply a space-time approach, cf. [SR26], or an explicit method, e.g., a Runge Kutta scheme. We will use a full Eulerian approach, where we discretize each time derivative with a backward differentiation formula (BDF). BDF methods are widely used in the literature for time discretization since they are A($\alpha$)-stable and allow for large time steps. Since the solving of the linear system arising in the space discretization in each time step is expensive, the possibility of relatively large time-steps is computationally efficient.

The main challenge in the numerical simulation of the surface Navier–Stokes equations is the coupling between the surface geometry and the fluid dynamics. In Chapter 7, we present a complete computational framework for the surface Navier–Stokes equations. We combine the TraceFEM discretization of the PDE on the surface with the narrow-band approach for the level set equation to obtain a fully Eulerian discretization for the SNS.

We implemented the methods using the finite element library `Netgen/NGSolve`, cf. [Sch97; Sch14], which provides a flexible framework for implementing finite element methods and generate meshes. The library is extended by the add-on `ngsxfem`, cf. [Leh+], which provides many useful tools suitable for the TraceFEM.

## 1.4 Main new results

We summarize the main contributions of this work:

- Derivation of surface differential operators in local and global coordinate systems, including a comparison of the two representations.
- Comprehensive summary of derivations of surface Navier-Stokes equations from the literature. The summaries include multiple mathematical frameworks and physical principles. We compare the resulting equations.

- Development of a novel trace finite element method for spatial discretization of tangential Navier-Stokes equations on evolving surfaces (TSNS) and an error analysis for the proposed numerical scheme.
- Design of an efficient narrow band level set method, including an extension method, for evolving surface representation.
- Formulation of a complete TraceFE based computational framework for the surface Navier–Stokes equations (SNS).
- Numerical experiments validating the proposed methods, including convergence studies and simulations of specific physical phenomena.

Parts of this thesis are based on the following joint publications:

[BRS22] Philip Brandner, Arnold Reusken, and Paul Schwering. On derivations of evolving surface Navier–Stokes equations. Interfaces and Free Boundaries, 24:533-563, 2022, doi:10.4171/IFB/483.
Parts of Chapter 2 and Chapter 3 are based on this publication, where different derivations of the surface Navier–Stokes equations are summarized and compared. Tangential surface Navier–Stokes equations are derived from the full surface Navier–Stokes equations.

[ORS24] Maxim A Olshanskii, Arnold Reusken, and Paul Schwering. An Eulerian finite element method for tangential Navier–Stokes equations on evolving surfaces. Mathematics of Computation, 93(349):2031-2065, 2024, doi:10.1090/mcom/3931.
Chapter 5 is based on this publication, where we present a TraceFEM based discretization method for the tangential surface Navier–Stokes equations. We summarize the proofs for stability and optimal order convergence of the method, and present numerical experiments to validate the theoretical results.

[ORS25] Maxim A Olshanskii, Arnold Reusken, and Paul Schwering. A Narrow Band Finite Element Method for the Level Set Equation. SIAM Journal on Scientific Computing, 47(2):A762-A789, 2025, doi:10.1137/24m1674182.
Chapter 6 is based on this publication, where we present a narrow band level set method for the level set equation. We introduce and analyse a novel level set extension method and couple it with a discontinuous Galerkin method for the transport equation. We present various numerical experiments to validate the accuracy of the method.

## 1.5 Outline of the thesis

The remainder of this thesis is organized into the following chapters:

- *Chapter 2: Basic Notations and Differential Operators* introduces the mathematical foundations for describing evolving surfaces and geometric quantities like the normal vector, the tangential projection, the shape operator, and the curvature. We construct surface differential operators in two coordinate systems: local (curvilinear) and global (Cartesian), and show relations between the two coordinate

systems and the corresponding surface differential operators. We present differentiation and integration rules that are needed in the following chapters. The chapter concludes with the introduction of the level set representation of the surface and the derivation of the level set transport equation.

- *Chapter 3: Surface Navier–Stokes Equations* presents a comprehensive overview over five derivations of surface Navier-Stokes equations using multiple approaches, including conservation laws, energetic variational principles, and thin film limits. The derivations are performed in curvilinear and Cartesian coordinates. We outline the main concepts and steps for each derivation and compare the resulting equations. The chapter also includes splittings of the resulting surface Navier–Stokes systems into tangential and normal components, leading to tangential surface Navier–Stokes equations. We show that all five derivations lead to the same TSNS system, but the (full) surface Navier–Stokes systems may differ.

- *Chapter 4: Trace Finite Element Method* summarizes the trace finite element framework for PDEs on stationary and evolving surfaces. We recall important key features of the TraceFEM from the literature. Topics covered include parametric finite element spaces to obtain higher order geometry approximations, geometric approximations via the level set function, stabilization techniques to obtain linear systems with acceptable conditioning. Furthermore, we discuss extensions of TraceFEM to time-dependent surfaces, where the main idea is a volume normal derivative stabilization, used as an extension to obtain well-posed systems.

- *Chapter 5: Discretization of Tangential Surface Navier–Stokes Equations* proposes a numerical scheme for the tangential Navier-Stokes equations posed on an evolving surface. The method is based on the trace finite element method and the BDF scheme introduced in Chapter 4. We give a detailed derivation of the numerical scheme, including the variational formulation, the discretization of the surface differential operators, and the stabilization techniques. The chapter includes a short overview of an error analysis and stability estimates. We conclude by presenting numerical experiments validating the theoretical results.

- *Chapter 6: Narrow band method for the level set equation* introduces an efficient narrow band approach for the level set equation to get a representation of the evolving surface depending on a given flow field. The method incorporates specially constructed boundary conditions and a discontinuous Galerkin (DG) method from the literature for the spatial discretization, combined with a BDF scheme in time. One main ingredient in narrow band methods is a suitable extension method for the level set function. We present a new extension approach based on a ghost penalty technique. We analyse the extension and show optimal order convergence for smooth solutions. Numerical experiments showcasing the performance of the full narrow band method on various test cases are conducted.

- *Chapter 7: Discretization of Surface Navier–Stokes Equations* integrates the numerical techniques developed in previous chapters to obtain a complete computational framework for simulating the full surface Navier-Stokes system. We present an extension method for the velocity based on the volume normal derivative stabilization and a new method to construct a suitable normal approximation. A

main result and the goal of this thesis is a TraceFEM based algorithm detailed in 7.7.2. Numerical experiments demonstrate the method's performance on various test cases including convergence tests showing the reliability of the method.

- *Chapter 8: Summary and Outlook* summarizes the key findings and contributions of the thesis and discusses potential directions for future research.

# Chapter 2

# Basic Notations and Differential Operators

In this chapter, we introduce the basic notations and the differential operators used throughout this thesis. We start with the definition of moving material surfaces and corresponding geometric quantities. Then, we introduce local and global coordinate systems and surface differential operators namely the surface gradient and the surface divergence. We derive representations of surface differential operators in local and global coordinates and show relations between them. Next, we derive the material derivative and give representations in local and global coordinate systems. These definitions and results are motivated by the work [BRS22]. In Sections 2.4 and 2.5, we define surface integrals and derive some rules for integration and differentiation that will be used throughout this thesis. Finally, we introduce an implicit representation of surfaces by level set functions and derive the level set equation in Section 2.6.

## 2.1 Material surfaces

We start by outlining how evolving material surfaces are defined. A more precise formal description of the notion "material" is given in [GM75; CM79]. Let $[0, t_{\text{end}}]$ be the time interval, where $t_{\text{end}} > 0$ denotes the end time, and $\Gamma^0 \subset \mathbb{R}^3$ be a *material*, two-dimensional, simply connected, closed, and oriented $C^m$-hypersurface (with $m \geq 2$) embedded in $\mathbb{R}^3$. Let $\rho^0 = \rho^0(\mathbf{x})$, $\mathbf{x} \in \Gamma^0$, denote a strictly positive continuous particle density distribution on $\Gamma^0$.

The initial surface $\Gamma^0$ and its density distribution are advected by the density flow $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$, which is a sufficiently smooth vector field. Below, we often omit the argument $(\mathbf{x}, t)$. A material point $\mathbf{z} \in \Gamma^0$ moves in time along the trajectory with coordinates $\mathbf{X}(\mathbf{z}, t)$ which is defined by the initial value problem

$$\begin{cases} \mathbf{X}(\mathbf{z}, 0) = \mathbf{z}, \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{z}, t) = \mathbf{u}\big(\mathbf{X}(\mathbf{z}, t), t\big) \qquad \text{for } t \in [0, t_{\text{end}}]. \end{cases} \tag{2.1.1}$$

Under mild regularity conditions, which we assume to be satisfied, the existence and uniqueness of a solution of (2.1.1) is guaranteed. We refer to Remark 2.1.1 for a short discussion of the well-posedness. Equation (2.1.1) leads to the following definition of a

smoothly evolving material surface $\Gamma(t)$, given by

$$\Gamma(t) := \left\{ \mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x} = \mathbf{X}(\mathbf{z}, t),\ \mathbf{z} \in \Gamma^0 \right\}. \tag{2.1.2}$$

It is evident that $\Gamma(0) = \Gamma^0$ holds. We further assume that $\Gamma(t)$ remains a simply connected, closed, and oriented $C^m$-hypersurface for all $t \in [0, t_{\text{end}}]$. The density distribution on $\Gamma(t)$ is given by $\rho = \rho(\mathbf{x}, t) = \rho(\mathbf{X}(\mathbf{z}, t), t)$ with $\rho(\mathbf{x}, 0) = \rho^0(\mathbf{x})$.

Since we assume the surface to be orientable, there exists a uniquely defined outward pointing normal of unit length of the surface. Following [DE13, Lemma 2.8.], the surface normal is well defined and continuously differentiable in a neighborhood of the surface. It is denoted by $\mathbf{n}(\cdot, t) \in C^1(U_\Gamma)^3$, with $U_\Gamma \subset \mathbb{R}^3$ a neighborhood of $\Gamma(t)$. The maximal possible neighborhood where the continuously differentiable normal can be defined depends on the curvature of the surface.

The velocity of the surface in the normal direction is given by $\mathbf{u}_N = (\mathbf{u} \cdot \mathbf{n})\mathbf{n}$. Since a tangential movement of $\Gamma(t)$ like a rotation of a sphere does not change the shape of the surface, the geometric evolution of the surface can be fully described by its normal velocity $\mathbf{u}_N$. Thus, we can substitute the second equation of (2.1.1) and get trajectories induced by the normal flow field $\mathbf{u}_N$, defined by

$$\begin{cases} \mathbf{X}_N(\mathbf{z}, 0) = \mathbf{z}, \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}_N(\mathbf{z}, t) = \mathbf{u}_N\left(\mathbf{X}_N(\mathbf{z}, t), t\right) \qquad \text{for } t \in [0, t_{\text{end}}]. \end{cases} \tag{2.1.3}$$

Substituting $\mathbf{X}$ in equation (2.1.2) with the normal trajectories $\mathbf{X}_N$ as defined in equation (2.1.3) results in the same surface $\Gamma(t)$, despite potential differences between the trajectories $\mathbf{X}$ and $\mathbf{X}_N$. This means that the subset $\Gamma(t) \subset \mathbb{R}^3$ generated by advecting $\Gamma^0$ using the velocity field $\mathbf{u}$ is identical to that obtained by advecting $\Gamma^0$ with the normal velocity field $\mathbf{u}_N$. In general, it holds $\mathbf{X}(\mathbf{z}, t) \neq \mathbf{X}_N(\mathbf{z}, t)$ for $t > 0$ and $\mathbf{z} \in \Gamma^0$.

**Remark 2.1.1 (Well-posedness of the surface evolution)**

The material velocity $\mathbf{u}$ is, in general, only defined on $\Gamma(t)$, and in some applications, also on a small time-dependent neighborhood of $\Gamma(t)$. In these cases, we cannot use the Picard-Lindelöf theorem to prove the existence of a solution of (2.1.1) (or (2.1.3)), because it requires a globally defined velocity. However, following [Bot92] and [BPS05, Appendix B], one can prove well-posedness of (2.1.1) and (2.1.3). They only require mild regularity assumptions on the velocities $\mathbf{u}$ and $\mathbf{u}_N$ defined on the surface $\Gamma(t)$. Thus, in the following, we assume the surface $\Gamma(t)$ is uniquely defined by the material velocity $\mathbf{u}$ (or $\mathbf{u}_N$) and the initial surface $\Gamma^0$. ◇

In order to define a local representation of $\Gamma(t)$ for $0 \leq t \leq t_{\text{end}}$, we use the flow map $\Phi_t : \Gamma(0) \to \Gamma(t)$ of a material point defined by $\Phi_t(\mathbf{z}) := \mathbf{X}(\mathbf{z}, t)$. In addition, we denote a local parametrization of the initial surface $\Gamma^0$ by $\Phi_U : \mathbb{R}^2 \supset U \to \Gamma^0$. More precisely, for each point $\mathbf{z}$ in $\Gamma^0$, there exist a neighborhood $V \subset \mathbb{R}^3$ of $\mathbf{z}$, a base point $\boldsymbol{\xi} \in \mathbb{R}^2$, and a neighborhood $U \subset \mathbb{R}^2$ of $\boldsymbol{\xi}$ such that $\mathbf{z} = \Phi_U(\boldsymbol{\xi})$ and $\Phi_U(U) = V \cap \Gamma^0$. We assume that the mapping $\Phi_U : U \to \Phi_U(U) \subset \Gamma^0$ is a diffeomorphism. Thus, the composition of $\Phi_U$ and $\Phi_t$ defines a local parametrization of $\Gamma(t)$, denoted by

$$R(\boldsymbol{\xi}, t) := \Phi_t(\Phi_U(\boldsymbol{\xi})), \tag{2.1.4}$$

where $\boldsymbol{\xi} = (\xi_1, \xi_2)$ is a coordinate in $U$. Let $\mathbf{z} = \Phi_U(\boldsymbol{\xi}) \in \Gamma(0) \subset \mathbb{R}^3$ be a material point represented by coordinates $\boldsymbol{\xi} \in U$. Then, the position of $\mathbf{z}$ at time $t$ is given by $R(\boldsymbol{\xi}, t)$. If the particle velocity $\mathbf{u}$ satisfies $\mathbf{u} \cdot \mathbf{n} = 0$ on $\Gamma(t)$ for $t \in [0, t_{\text{end}}]$, there is no normal velocity of the surface which means that the geometry of $\Gamma(t)$ does not change over time. But even in this case, the parametrization is a *non*-constant function with respect to $t$ when the flow field $\mathbf{u}$ is not identically zero.

We define the space-time manifold by

$$\mathcal{S} := \bigcup_{t \in [0, t_{\text{end}}]} \Gamma(t) \times \{t\} \subset \mathbb{R}^3 \times [0, t_{\text{end}}].$$

### 2.1.1 Geometric quantities

In this section, we define important geometric quantities of $\Gamma(t)$, namely the projection $\mathbf{P}$ on the tangential plane, the Weingarten mapping (or second fundamental form) $\mathbf{H}$, and the mean curvature $\kappa$.

**Definition 2.1.2**

At $\mathbf{x} \in \Gamma(t)$, the projection $\mathbf{P}$ on the tangential plane, the Weingarten mapping $\mathbf{H}$, and the mean curvature $\kappa$ are defined by

$$\mathbf{P}(\mathbf{x}, t) := \mathbf{I} - \mathbf{n}(\mathbf{x}, t)\mathbf{n}(\mathbf{x}, t)^T, \quad \mathbf{H}(\mathbf{x}, t) := \nabla \mathbf{n}(\mathbf{x}, t), \quad \kappa(\mathbf{x}, t) := \operatorname{tr}(\mathbf{H}(\mathbf{x}, t)),$$

respectively, where $\mathbf{I}$ denotes the identity operator and $\nabla$ the Jacobian with respect to the Cartesian coordinates in $\mathbb{R}^3$.

Below, we often omit the argument $(\mathbf{x}, t)$ in the notation. Note, that $\mathbf{P}$, $\mathbf{H}$, and $\kappa$ are defined in the local neighborhood $U_\Gamma$ of $\Gamma$ where the normal vector $\mathbf{n}$ is well defined and twice differentiable.

The tangential projection satisfies

$$\mathbf{P}^2 = \mathbf{P} \quad \text{as well as} \quad \mathbf{P}^T = \mathbf{P},$$

where the transposed linear operator $\mathbf{T}^T$ is defined by the relation $\mathbf{T}\mathbf{u} \cdot \mathbf{w} = \mathbf{u} \cdot \mathbf{T}^T\mathbf{w}$ for all $\mathbf{u}, \mathbf{w} \in \mathbb{R}^3$ according to the Euclidean inner product in $\mathbb{R}^3$. Since $\nabla \mathbf{n}$ is symmetric and the identity $0 = 2\nabla(\mathbf{n} \cdot \mathbf{n}) = \nabla \mathbf{n}\mathbf{n}$ holds, the Weingarten mapping satisfies

$$\mathbf{H}^T = \mathbf{H} \quad \text{and} \quad \mathbf{H} = \nabla \mathbf{n} = \mathbf{P}\nabla \mathbf{n} = \nabla \mathbf{n}\mathbf{P} = \mathbf{P}\nabla \mathbf{n}\mathbf{P}.$$

As $\mathbf{H}$ is symmetric and satisfies $\mathbf{H}\mathbf{n} = 0$, the Weingarten mapping has three real eigenvalues, one of which is zero. Let $\kappa_1$ and $\kappa_2$ denote the other two eigenvalues of $\mathbf{H}$. The mean curvature is then given by $\kappa = \kappa_1 + \kappa_2$.

For a vector field $\mathbf{v}$ on $\Gamma(t)$, we use the notation $\mathbf{v}_T = \mathbf{P}\mathbf{v}$ for the tangential component of $\mathbf{v}$ and $v_N = \mathbf{v} \cdot \mathbf{n}$ for the coordinate in normal direction. Then, it holds

$$\mathbf{v} = \mathbf{v}_T + v_N \mathbf{n}.$$

## 2.2 Coordinate systems and surface differential operators

Since the surface $\Gamma(t)$ (for a fixed $t$) is not an open subset of $\mathbb{R}^3$, the derivatives with respect to the standard Cartesian basis are not well defined for functions that are defined on $\Gamma(t)$ only. Thus, we have to define surface differential operators. We collect results concerning the representations of functions and their derivatives in a local (curvilinear) and a global (Cartesian) coordinate system. In the remainder of this section, let $t \in [0, t_{\text{end}}]$ be fixed and denote the local representation of $\Gamma = \Gamma(t)$ by $R(\boldsymbol{\xi}) = R(\boldsymbol{\xi}, t)$, for $\boldsymbol{\xi} \in U \subset \mathbb{R}^2$. Without loss of generality, we assume that we have a global immersion $R : U \to \mathbb{R}^3$ such that $R(U) = \Gamma$. We are interested in representations of scalar-valued functions $q : \Gamma \to \mathbb{R}$, vector fields $\mathbf{v} : \Gamma \to \mathbb{R}^3$, and operator-valued mappings $\mathbf{T} : \Gamma \to L(\mathbb{R}^3, \mathbb{R}^3)$, where $L(\mathbb{R}^3, \mathbb{R}^3)$ denotes the space of linear mappings $\mathbb{R}^3 \to \mathbb{R}^3$ as well as their surface derivatives.

The main idea of the surface derivatives defined in the local coordinate system is to differentiate functions defined on $\Gamma$ (embedded in $\mathbb{R}^3$) with respect to the local coordinates $\boldsymbol{\xi} \in \mathbb{R}^2$. The representation of surface differential operators in the global Cartesian coordinate system is obtained by an extension of functions off the surface and applying the tangential projection $\mathbf{P}$.

### 2.2.1 Coordinate systems

First, we define a local and a global coordinate systems. Most of the results presented in this section can be found in many textbooks, cf. e.g., [Cia13; PS16]. To simplify the notation, we use the Einstein summation convention: Using Latin indices (i, j, k, ...) we sum over $1, 2, 3$, using Greek indices $(\alpha, \beta, \gamma, \dots)$ we sum over $1, 2$. The Cartesian coordinates of a point $\boldsymbol{\xi} \in \mathbb{R}^2$ in the standard basis are given by $\xi_\alpha$. Partial derivatives with respect to these coordinates are denoted by $\overline{\partial}_\alpha = \frac{\partial}{\partial \xi_\alpha}$. We assume that the local parametrization $R$ is an immersion, i.e., the matrix

$$\begin{pmatrix} \overline{\partial}_1 R(\boldsymbol{\xi}) & \overline{\partial}_2 R(\boldsymbol{\xi}) \end{pmatrix} \in \mathbb{R}^{3\times 2}$$

has rank two for each $\boldsymbol{\xi} \in U$. Hence, the representation of each point $\mathbf{x} \in R(U)$ by the immersion ($\mathbf{x} = R(\boldsymbol{\xi})$) is unambiguous. In general, it is not possible to describe the whole surface $\Gamma$ as an image of a single immersion $R$. In this case, we use a partition of unity of several immersions. Thus, we can assume that for each $\mathbf{x} \in \Gamma$, there exists a neighborhood $V \subset \mathbb{R}^3$ of $\mathbf{x}$, a base point $\boldsymbol{\xi} \in \mathbb{R}^2$ and a neighborhood $U \subset \mathbb{R}^2$ of $\boldsymbol{\xi}$ such that $\mathbf{x} = R(\boldsymbol{\xi})$ and $R(U) = V \cap \Gamma$.

We define a global and a local basis of $\mathbb{R}^3$. The standard Cartesian (global) basis of $\mathbb{R}^3$ is given by $\{\mathbf{e}_1 = \mathbf{e}^1, \cdots, \mathbf{e}_3 = \mathbf{e}^3\}$. The covariant (local) basis of the tangent space at $\mathbf{x} = R(\boldsymbol{\xi}) \in \Gamma$ is given by

$$\mathbf{g}_\alpha = \mathbf{g}_\alpha(\boldsymbol{\xi}) := \overline{\partial}_\alpha R(\boldsymbol{\xi}) \in \mathbb{R}^3.$$

Note that $\mathbf{g}_\alpha$ are linearly independent since $R$ is an immersion and tangential to the surface at $\mathbf{x} = R(\xi)$, cf. [Cia13, Section 8.2] for a proof.

We extend this basis by the normal vector and write $\mathbf{g}_3 := \mathbf{n}$. Thus, $\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3\}$ forms a basis of $\mathbb{R}^3$ which is called *covariant basis*. This is a local basis since it depends on the point $\mathbf{x} \in \Gamma$ (or $\boldsymbol{\xi} \in \mathbb{R}^2$). The two coordinates $\xi_\alpha$ of $\boldsymbol{\xi}$ are called *curvilinear* or *local coordinates* of $\mathbf{x} = R(\boldsymbol{\xi})$. See figure 2.2.1 for an illustration of the local and global bases.

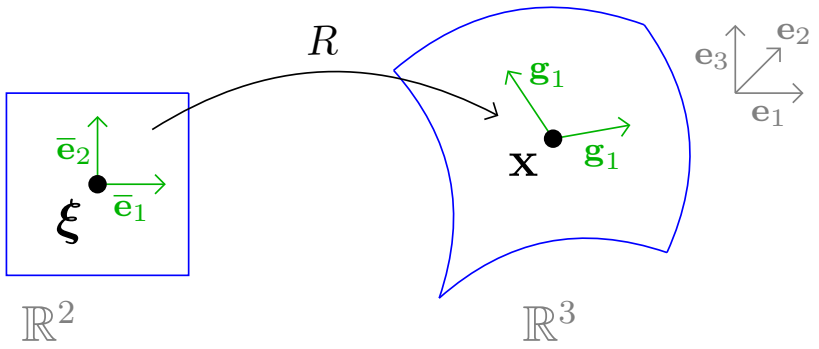



Figure 2.2.1: Local basis based on immersion $R$ holding $\mathbf{x} = R(\boldsymbol{\xi})$.

We define a second (local) basis, named contravariant basis, denoted by $\mathbf{g}^j = \mathbf{g}^j(\boldsymbol{\xi})$. It is uniquely defined by the relations $\mathbf{g}_j \cdot \mathbf{g}^i = \delta^i_j$, with the Kronecker symbol $\delta^i_j$. The normal vector $\mathbf{n}$ is part of both the covariant and the contravariant basis. It can be written as

$$\mathbf{n} = \mathbf{g}_3 = \mathbf{g}^3 = \frac{\mathbf{g}_1 \times \mathbf{g}_2}{\|\mathbf{g}_1 \times \mathbf{g}_2\|} = \frac{\mathbf{g}^1 \times \mathbf{g}^2}{\|\mathbf{g}^1 \times \mathbf{g}^2\|}, \tag{2.2.1}$$

cf. [Cia13].

**Remark 2.2.1**

The sign of the *outward* pointing normal $\mathbf{n}$ is uniquely defined by the orientation of the surface $\Gamma$. Since the cross-product satisfies $a \times b = -b \times a$, equation (2.2.1) defines the order of the co- and contravariant bases. ◇

The vectors $\mathbf{g}_i(\boldsymbol{\xi})$ and $\mathbf{g}^i(\boldsymbol{\xi})$ for $i = 1, 2, 3$ form a local (depending on the base point $\boldsymbol{\xi}$) basis of the (whole) space $\mathbb{R}^3$. We can (locally) interpret the basis functions $\mathbf{g}_i$ and $\mathbf{g}^i$ as functions defined on the surface by $\mathbf{g}_i(\mathbf{x}) := \mathbf{g}_i(R(\boldsymbol{\xi}))$, where $\mathbf{x} = R(\boldsymbol{\xi})$.

Now we can define representations of vector- and tensor-valued functions in different bases. Using the tensor calculus format (cf. [Cia13, Section 8.4]) and the outer product given by $(\mathbf{u} \otimes \mathbf{v})\mathbf{w} = (\mathbf{v} \cdot \mathbf{w})\mathbf{u}$ for $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$, we define the following representations.

**Definition 2.2.2**

For a vector field $\mathbf{v} : \Gamma \to \mathbb{R}^3$ and an operator-valued mapping $\mathbf{T} : \Gamma \to L(\mathbb{R}^3, \mathbb{R}^3)$, we introduce the representations

$$\mathbf{v} = \hat{v}^i \mathbf{g}_i = \hat{v}_i \mathbf{g}^i = v_i \mathbf{e}^i, \quad \mathbf{T} = \hat{T}_{ij}(\mathbf{g}^i \otimes \mathbf{g}^j) = \hat{T}^{ij}(\mathbf{g}_i \otimes \mathbf{g}_j) = T_{ij}(\mathbf{e}^i \otimes \mathbf{e}^j),$$

with $\hat{v}_i := \mathbf{v} \cdot \mathbf{g}_i$, $\hat{v}^i := \mathbf{v} \cdot \mathbf{g}^i$, $v_i := \mathbf{v} \cdot \mathbf{e}_i$, $\hat{T}_{ij} = \mathbf{g}_i \cdot (\mathbf{T}\mathbf{g}_j)$, $\hat{T}^{ij} = \mathbf{g}^i \cdot (\mathbf{T}\mathbf{g}^j)$, and $T_{ij} = \mathbf{e}_i \cdot (\mathbf{T}\mathbf{e}_j)$. The coefficients $\hat{v}_i$ and $\hat{T}_{i,j}$ ($\hat{v}^i$ and $\hat{T}^{i,j}$) are called covariant (contravariant) components. The coefficients $v_i$ and $T_{ij}$ are the Cartesian components.

Note that if the vector- or tensor-valued functions are smooth, all component functions are differentiable because the basis vectors $\mathbf{g}_i$, $\mathbf{g}^i$, and $\mathbf{e}_i$ are smooth.

An important quantity of the surface is the so-called metric tensor (or first fundamental form). It behaves like the identity on the tangential plane (to the surface), see

Remark 2.2.3. The components of the metric tensor are defined by

$$g_{\alpha\beta}(\boldsymbol{\xi}) \coloneqq \mathbf{g}_\alpha(\boldsymbol{\xi}) \cdot \mathbf{g}_\beta(\boldsymbol{\xi}), \quad \text{for } 1 \leq \alpha, \beta \leq 2. \tag{2.2.2}$$

Note that they are symmetric, and they form a positive definite tensor. The contravariant components of the metric tensor $\mathbf{g}$ are defined by $g^{\alpha\beta}(\boldsymbol{\xi}) \coloneqq \mathbf{g}^\alpha(\boldsymbol{\xi}) \cdot \mathbf{g}^\beta(\boldsymbol{\xi})$. The metric tensor and the local bases are related by the relations

$$\mathbf{g}_\alpha = g_{\alpha\beta}\mathbf{g}^\beta, \qquad \mathbf{g}^\alpha = g^{\alpha\beta}\mathbf{g}_\beta, \qquad g^{\alpha\gamma}g_{\gamma\beta} = \delta^\alpha_\beta, \quad \text{and} \quad \|\mathbf{g}_1 \times \mathbf{g}_2\| = \sqrt{\det(\mathbf{g})},$$

cf. [Cia13]. Here, the metric tensor $g_{\alpha\beta}$ and its inverse $g^{\alpha\beta}$ perform basis transformations between the co- and the contravariant basis and vice versa. To change the representation of a given vector field, we use the relations

$$v_j = v_i\mathbf{e}^i \cdot \mathbf{e}_j = \hat{v}_i\mathbf{g}^i \cdot \mathbf{e}_j, \quad \text{and} \quad \hat{v}_j = \hat{v}_i\mathbf{g}^i \cdot \mathbf{g}_j = v_i\mathbf{e}^i \cdot \mathbf{g}_j.$$

Tensors can also be represented using *mixed* components (cf. [Cia13, Section 8.4]). For a *symmetric* linear operator $\mathbf{T}$, i.e., $\mathbf{T} = \mathbf{T}^T$, we introduce the mixed (between covariant and contravariant) matrix representation by

$$\mathbf{T} = \hat{T}^i_j(\mathbf{g}_i \otimes \mathbf{g}^j) = \hat{T}^i_j(\mathbf{g}^j \otimes \mathbf{g}_i).$$

The relations $\hat{T}^i_j = \mathbf{g}^i \cdot (\mathbf{T}\mathbf{g}_j) = \mathbf{g}_j \cdot (\mathbf{T}\mathbf{g}^i)$ hold. One can extend this definition to non-symmetric tensors. We refer the reader to the literature for more details, e.g., [Cia13, Section 8.4].

For a linear operator $\mathbf{T}$ the sum of its eigenvalues is denoted by $\operatorname{tr}(\mathbf{T})$. Since eigenvalues are invariant under basis transformations, the trace can be written as

$$\operatorname{tr}(\mathbf{T}) = \hat{T}_{ii} = \hat{T}^{ii} = T_{ii} = \mathbf{g}_i \cdot (\mathbf{T}\mathbf{g}_i) = \mathbf{g}^i \cdot (\mathbf{T}\mathbf{g}^i) = \mathbf{e}_i \cdot (\mathbf{T}\mathbf{e}_i). \tag{2.2.3}$$

In the case of a symmetric linear operator $\mathbf{T}$, we also have $\operatorname{tr}(\mathbf{T}) = \hat{T}^i_i$. See Appendix A for a proof.

**Remark 2.2.3 (Behavior of metric tensor)**

We investigate the behavior of the metric tensor $g_{\alpha\beta}\left(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta\right)$ applied to a tangential vector $\mathbf{v} = \hat{v}^\gamma\mathbf{g}_\gamma$. A simple calculation yields

$$\begin{aligned} g_{\alpha\beta}\left(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta\right)\mathbf{v} &= \left(g_{\alpha\beta}\hat{v}^\gamma\mathbf{g}^\beta \cdot \mathbf{g}_\gamma\right)\mathbf{g}^\alpha = \left(g_{\alpha\beta}\hat{v}^\gamma\delta^\beta_\gamma\right)\mathbf{g}^\alpha = \hat{v}^\gamma g_{\alpha\gamma}\mathbf{g}^\alpha = \hat{v}^\gamma\mathbf{g}_\gamma \\ &= \mathbf{v}. \end{aligned}$$

This result shows that the components of the metric tensor $g_{\alpha\beta}$ form the identity operator on the tangential plane. ◇

We represent geometric quantities of $\Gamma(t)$ in local coordinate systems. Note that it holds $\mathbf{P}(\hat{v}^i\mathbf{g}_i) = \hat{v}^\alpha\mathbf{g}_\alpha$, $\mathbf{v}_T = \mathbf{P}\mathbf{v} = \hat{v}^\alpha\mathbf{g}_\alpha = \hat{v}_\alpha\mathbf{g}^\alpha$, and $v_N = \mathbf{v} \cdot \mathbf{n} = \hat{v}^3 = \hat{v}_3$. We give some representations of geometric quantities in local coordinates. A proof can be found in Appendix A.

**Lemma 2.2.4**

The mixed components of the tangential projection in local coordinates are given by

$$\hat{P}_\alpha^\beta = \delta_\alpha^\beta \quad \text{and} \quad \hat{P}_3^i = \hat{P}_i^3 = 0.$$

The components in local bases of the Weingarten mapping are given by

$$\hat{H}_{\alpha\beta} = \hat{H}_{\beta\alpha} = -\mathbf{g}_3 \cdot \overline{\partial}_\alpha \mathbf{g}_\beta = \overline{\partial}_\alpha \mathbf{g}_3 \cdot \mathbf{g}_\beta, \quad \hat{H}_3^i = \hat{H}_i^3 = 0 \quad \text{and} \quad \hat{H}_\alpha^\beta = g^{\beta\sigma} \hat{H}_{\sigma\alpha}.$$

The mean curvature can be represented in mixed components by $\kappa = \hat{H}_\alpha^\alpha$.

Note that the mixed components of $\mathbf{H}$ are in general not symmetric. This lemma shows, that our definition of the Weingarten mapping is consistent with the one in [Cia13].

### 2.2.2 Surface differential operators

In this section, we recall several surface differential operators represented in different bases. For fixed $t \in [0, t_{\text{end}}]$, we again denote $\Gamma = \Gamma(t)$. Let $q : \Gamma \to \mathbb{R}$ be a scalar function, $\mathbf{v} : \Gamma \to \mathbb{R}^3$ be a (not necessarily tangential) vector field, and $\mathbf{T} : \Gamma \to L(\mathbb{R}^3, \mathbb{R}^3)$ an operator-valued mapping. All are assumed to be at least $C^1$-smooth. We will study partial surface derivatives and surface gradients of $q$, $\mathbf{v}$, and $\mathbf{T}$ and surface divergence operators for $\mathbf{v}$ and $\mathbf{T}$. We start with the derivation of the local representations of these operators. Note that in this case, the basis used in $\mathbb{R}^3$ depends on the (base) point $\mathbf{x} \in \Gamma$. As a next step, we define analogous gradient and divergence operators, represented in Cartesian coordinates in $\mathbb{R}^3$. Then, we derive relations between local and global representations of the operators.

#### Surface differential operators in curvilinear coordinates

We recall some basic differential geometry concepts, e.g., [Cia13, Chapter 8]. We start with definitions of partial derivatives for functions defined on $\Gamma$. Note that we have representations of vector- and tensor-valued functions in curvilinear coordinates given by $\mathbf{v} = \hat{v}^i \mathbf{g}_i = \hat{v}_i \mathbf{g}^i$ and $\mathbf{T} = \hat{T}^{ij}(\mathbf{g}_i \otimes \mathbf{g}_j) = \hat{T}_{ij}(\mathbf{g}^i \otimes \mathbf{g}^j)$. All component functions are differentiable because the basis vectors $\mathbf{g}_i$ and $\mathbf{g}^i$ are smooth. Since $R$ is an immersion, there are uniquely defined functions $\overline{q} : U \to \mathbb{R}$, $\overline{\mathbf{v}} : U \to \mathbb{R}^3$, and $\overline{\mathbf{T}} : U \to L(\mathbb{R}^3, \mathbb{R}^3)$ such that $\overline{q}(\boldsymbol{\xi}) = q(R(\boldsymbol{\xi}))$, $\overline{\mathbf{v}}(\boldsymbol{\xi}) = \mathbf{v}(R(\boldsymbol{\xi}))$, and $\overline{\mathbf{T}}(\boldsymbol{\xi}) = \mathbf{T}(R(\boldsymbol{\xi}))$ hold for all $\xi \in U$.

Now, we use these to define the local surface derivatives of $q$, $\mathbf{v}$, and $\mathbf{T}$ with respect to the curvilinear coordinates $\boldsymbol{\xi}$.

**Definition 2.2.5: Partial surface derivatives**

The local *partial* derivatives $\partial_\alpha^\Gamma$ of a scalar function $q$, a vector field $\mathbf{v}$, and an operator-valued mapping $\mathbf{T}$ are defined in terms of the corresponding functions $\overline{q}$, $\overline{\mathbf{v}}$, and $\overline{\mathbf{T}}$ by

$$\partial_\alpha^\Gamma q(\mathbf{x}) := \overline{\partial}_\alpha \overline{q}(\boldsymbol{\xi}), \quad \partial_\alpha^\Gamma \mathbf{v}(\mathbf{x}) := \overline{\partial}_\alpha \overline{\mathbf{v}}(\boldsymbol{\xi}), \quad \partial_\alpha^\Gamma \mathbf{T}(\mathbf{x}) := \overline{\partial}_\alpha \overline{\mathbf{T}}(\boldsymbol{\xi}), \quad \text{with } \mathbf{x} = R(\boldsymbol{\xi}),$$

where $\overline{\partial}_\alpha$ denotes the derivative with respect to the Euclidean basis in $\mathbb{R}^2$.

Now, we derive representations of these partial derivatives in terms of a curvilinear coordinate system. We use the Christoffel symbols (cf. [Cia13, Theorem 8.13-1]) given by

$$\Gamma^{\sigma}_{\alpha\beta} := \mathbf{g}^{\sigma} \cdot \partial^{\Gamma}_{\alpha}\mathbf{g}_{\beta} = \Gamma^{\sigma}_{\beta\alpha}.$$

They can also be expressed in terms of the metric tensor (cf. [Cia13, Theorem 8.13-1, Theorem 8.14-1]) as

$$\Gamma^{\sigma}_{\alpha\beta} = \frac{1}{2} g^{\sigma\tau}(\partial^{\Gamma}_{\beta} g_{\alpha\tau} + \partial^{\Gamma}_{\alpha} g_{\beta\tau} - \partial^{\Gamma}_{\tau} g_{\alpha\beta}).$$

The Christoffel symbols are also used, e.g., in [PS16, equation (2.15)], [EHZ07], and [NNV19, equation (4)].

Representations of partial derivatives of vector fields and tensor-valued functions in terms of a curvilinear coordinate system are given in the following theorem.

**Theorem 2.2.6**

Derivatives of the vectors of the covariant and contravariant bases are given by

$$\partial^{\Gamma}_{\alpha}\mathbf{g}_{\beta} = \Gamma^{\sigma}_{\alpha\beta}\mathbf{g}_{\sigma} - \hat{H}_{\alpha\beta}\mathbf{g}_3, \quad \partial^{\Gamma}_{\alpha}\mathbf{g}^{\beta} = -\Gamma^{\beta}_{\alpha\sigma}\mathbf{g}^{\sigma} - \hat{H}^{\beta}_{\alpha}\mathbf{g}^3, \quad \partial^{\Gamma}_{\alpha}\mathbf{g}_3 = \partial^{\Gamma}_{\alpha}\mathbf{g}^3 = \hat{H}_{\alpha\beta}\mathbf{g}^{\beta} = \hat{H}^{\sigma}_{\alpha}\mathbf{g}_{\sigma}.$$

For a vector field $\mathbf{v} = \hat{v}_i\mathbf{g}^i = \hat{v}^i\mathbf{g}_i$, partial derivatives have the representations

$$\begin{aligned} \partial^{\Gamma}_{\alpha}\mathbf{v} &= (\hat{v}_{\beta|\alpha} + \hat{H}_{\alpha\beta}\hat{v}_3)\mathbf{g}^{\beta} + (\hat{v}_{3|\alpha} - \hat{H}^{\beta}_{\alpha}\hat{v}_{\beta})\mathbf{g}^3 \\ &= (\hat{v}^{\beta}{}_{|\alpha} + \hat{H}^{\beta}_{\alpha}\hat{v}^3)\mathbf{g}_{\beta} + (\hat{v}^3{}_{|\alpha} - \hat{H}_{\alpha\beta}\hat{v}^{\beta})\mathbf{g}_3, \end{aligned} \tag{2.2.4}$$

where we use the abbreviations

$$\hat{v}_{\beta|\alpha} := \partial^{\Gamma}_{\alpha}\hat{v}_{\beta} - \Gamma^{\gamma}_{\alpha\beta}\hat{v}_{\gamma}, \qquad \hat{v}^{\beta}{}_{|\alpha} := \partial^{\Gamma}_{\alpha}\hat{v}^{\beta} + \Gamma^{\beta}_{\gamma\alpha}\hat{v}^{\gamma}, \quad \text{and} \quad \hat{v}_{3|\alpha} = \hat{v}^3{}_{|\alpha} := \partial^{\Gamma}_{\alpha}\hat{v}_3 = \partial^{\Gamma}_{\alpha}\hat{v}^3.$$

Let $\mathbf{T} = \hat{T}^{\alpha\beta}(\mathbf{g}_{\alpha} \otimes \mathbf{g}_{\beta}) = \hat{T}_{\alpha\beta}(\mathbf{g}^{\alpha} \otimes \mathbf{g}^{\beta})$ be a function with values in $L(\mathbb{R}^3, T\Gamma)$, where $T\Gamma$ denotes the tangent bundle of $\Gamma$. For the partial derivatives, we have the representations

$$\begin{aligned} \partial^{\Gamma}_{\gamma}\mathbf{T} &= \hat{T}_{\alpha\beta|\gamma}(\mathbf{g}^{\alpha} \otimes \mathbf{g}^{\beta}) - \hat{T}_{\alpha\beta}\hat{H}^{\alpha}_{\gamma}(\mathbf{g}^3 \otimes \mathbf{g}^{\beta}) - \hat{T}_{\alpha\beta}\hat{H}^{\beta}_{\gamma}(\mathbf{g}^{\alpha} \otimes \mathbf{g}^3) \\ &= \hat{T}^{\alpha\beta}{}_{|\gamma}(\mathbf{g}_{\alpha} \otimes \mathbf{g}_{\beta}) - \hat{T}^{\alpha\beta}\hat{H}_{\gamma\alpha}(\mathbf{g}_3 \otimes \mathbf{g}_{\beta}) - \hat{T}^{\alpha\beta}\hat{H}_{\gamma\beta}(\mathbf{g}_{\alpha} \otimes \mathbf{g}_3), \end{aligned} \tag{2.2.5}$$

where we use the abbreviations

$$\hat{T}^{\alpha\beta}{}_{|\gamma} := \partial^{\Gamma}_{\gamma}\hat{T}^{\alpha\beta} + \Gamma^{\alpha}_{\mu\gamma}\hat{T}^{\mu\beta} + \Gamma^{\beta}_{\mu\gamma}\hat{T}^{\alpha\mu} \quad \text{and} \quad \hat{T}_{\alpha\beta|\gamma} := \partial^{\Gamma}_{\gamma}\hat{T}_{\alpha\beta} - \Gamma^{\mu}_{\alpha\gamma}\hat{T}_{\mu\beta} - \Gamma^{\mu}_{\beta\gamma}\hat{T}_{\alpha\mu}.$$

A proof of (2.2.5) is given in the appendix. The rest of the theorem is taken from [Cia13, Theorem 8.13-1].

**Remark 2.2.7**

For tangential vector fields $\mathbf{v}$, it holds $\hat{v}_{\beta|\alpha}\mathbf{g}^{\beta} = \mathbf{P}\partial^{\Gamma}_{\alpha}\mathbf{v}$, which motivates the notation $\hat{v}_{\beta|\alpha}$.

◇

Using these partial derivatives, we can define the surface differential operators in curvilinear coordinates.

**Definition 2.2.8: Surface differential operators (local bases)**

For a scalar function $q \in C^1(\Gamma, \mathbb{R})$ the surface gradient is defined by

$$\widehat{\nabla}_\Gamma q := \partial_\alpha^\Gamma q \, \mathbf{g}^\alpha.$$

For a vector field $\mathbf{v} \in C^1(\Gamma, \mathbb{R}^3)$ we define the $\alpha$-th partial covariant derivative $\widehat{\nabla}_\alpha \mathbf{v}$ and the surface Jacobian $\widehat{\nabla}_\Gamma \mathbf{v}$ by

$$\widehat{\nabla}_\alpha \mathbf{v} := \mathbf{P} \partial_\alpha^\Gamma \mathbf{v}, \qquad \widehat{\nabla}_\Gamma \mathbf{v} := \widehat{\nabla}_\alpha \mathbf{v} \otimes \mathbf{g}^\alpha = \mathbf{P} \partial_\alpha^\Gamma \mathbf{v} \otimes \mathbf{g}^\alpha.$$

The surface divergence of $\mathbf{v} \in C^1(\Gamma, \mathbb{R}^3)$ and $\mathbf{T} \in C^1(\Gamma, L(\mathbb{R}^3, \mathbb{R}^3))$ are defined by

$$\widehat{\operatorname{div}}_\Gamma \mathbf{v} := \partial_\alpha^\Gamma \mathbf{v} \cdot \mathbf{g}^\alpha = \mathbf{P} \partial_\alpha^\Gamma \mathbf{v} \cdot \mathbf{g}^\alpha, \qquad \widehat{\operatorname{div}}_\Gamma \mathbf{T} := (\partial_\alpha^\Gamma \mathbf{T})^T \mathbf{g}^\alpha,$$

respectively.

The definitions of the surface gradient, the surface Jacobian, and surface divergence operators above do not depend on the choice of the parametrization, cf. [PS16; BRS22]. We comment on the definitions in the following remark.

**Remark 2.2.9**

In most literature, cf. [PSW20; PS16; NRV19], the surface Jacobian $\widehat{\nabla}_\Gamma \mathbf{v}$ is defined for *tangential* vector functions only. In this case, $\widehat{\nabla}_\Gamma \mathbf{v}$ equals the so-called covariant derivative. However, we use the more general definition of the surface Jacobian for general (not necessarily tangential) vector fields $\mathbf{v}$.

Another natural surface differential operator for vector fields is a *surface gradient* of $\mathbf{v}$, defined by $\widehat{\nabla}_S \mathbf{v} := \mathbf{g}^\alpha \otimes \partial_\alpha^\Gamma \mathbf{v}$, cf. [PS16]. Note that it maps into the tangent bundle. It is related to the surface Jacobian via $\widehat{\nabla}_\Gamma \mathbf{v} = \mathbf{P} \widehat{\nabla}_S^T \mathbf{v}$ (where we use the notation $\widehat{\nabla}_S^T \mathbf{v} = (\widehat{\nabla}_S \mathbf{v})^T$). We will use this surface gradient in the proof of Theorem 2.2.12. ◇

Using Theorem 2.2.6, we obtain representations of different differential operators in covariant and contravariant bases. For example, a representation of the $\alpha$-th partial covariant derivative in the covariant basis $\mathbf{g}_\alpha$ in terms of (derivatives of) the contravariant components $\hat{v}^i$ in $\mathbf{v} = \hat{v}^i \mathbf{g}_i$ is given by

$$\widehat{\nabla}_\alpha \mathbf{v} = \mathbf{P} \partial_\alpha^\Gamma \mathbf{v} = (\partial_\alpha^\Gamma \hat{v}^\beta + \Gamma_{\alpha\gamma}^\beta \hat{v}^\gamma + \hat{H}_\alpha^\beta \hat{v}^3) \mathbf{g}_\beta = (\hat{v}^\beta_{\ |\alpha} + \hat{H}_\alpha^\beta \hat{v}^3) \mathbf{g}_\beta. \tag{2.2.6}$$

This result shows that the notation $\hat{v}^\beta_{\ |\alpha}$ introduced in Theorem 2.2.6 is natural, in the sense that for a tangential $\mathbf{v}$, we have $\widehat{\nabla}_\alpha \mathbf{v} = \hat{v}^\beta_{\ |\alpha} \mathbf{g}_\beta$. The relation

$$\widehat{\nabla}_\Gamma \mathbf{v} = \hat{v}_{\alpha|\beta} (\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) \tag{2.2.7}$$

holds, i.e., the covariant components of $\widehat{\nabla}_\Gamma \mathbf{v}$ are given by $\hat{v}_{\alpha|\beta}$. This induces an equivalent alternative definition of the covariant derivative, that is sometimes used in the literature.

In the following lemma, we present a representation result for the divergence of a vector- or operator-valued function. A proof is given in the appendix.

**Lemma 2.2.10: Local coordinates of divergence operators**

For $\mathbf{v} \in C^1(\Gamma, \mathbb{R}^3)$ it holds

$$\widehat{\operatorname{div}}_\Gamma \mathbf{v} = \hat{v}^\alpha_{|\alpha} + \hat{H}^\alpha_\alpha \hat{v}_3 = \hat{v}^\alpha_{|\alpha} + \kappa \hat{v}_3.$$

For $\mathbf{T} = \hat{T}^{\alpha\beta}(\mathbf{g}_\alpha \otimes \mathbf{g}_\beta) \in C^1(\Gamma, \mathbb{R}^{3\times 3})$ it holds

$$\widehat{\operatorname{div}}_\Gamma \mathbf{T} = \hat{T}^{\alpha\beta}_{|\alpha}\mathbf{g}_\beta - \hat{T}^{\alpha\beta}\hat{H}_{\alpha\beta}\mathbf{g}_3 = \hat{T}_{\alpha\beta|\gamma} g^{\alpha\gamma}\mathbf{g}^\beta - \hat{T}_{\alpha\beta}\hat{H}^{\alpha\beta}\mathbf{g}^3.$$

From the first equation of the lemma, we see that for a tangential vector-field $\mathbf{v}$, the divergence $\widehat{\operatorname{div}}_\Gamma \mathbf{v}$ is given by

$$\widehat{\operatorname{div}}_\Gamma \mathbf{v} = \hat{v}^\alpha_{|\alpha}. \tag{2.2.8}$$

Lemma 2.2.10 and equation (2.2.6) show the dependence of the surface different operators on the geometry of the surface given by the Weingarten mapping $\mathbf{H}$.

### Surface differential operators in Cartesian coordinates

In this section, we recall definitions of surface differential operators in terms of representations in the Cartesian coordinate system similar to the definitions in [JOR18]. The partial derivatives with respect to the standard basis $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ in $\mathbb{R}^3$, i.e., $\mathbf{x} = x_i \mathbf{e}_i$, are denoted by $\partial_k := \frac{\partial}{\partial x_k}$. The gradient of a scalar function $q$ with respect to the Cartesian coordinates is given by the column vector $\nabla q := (\partial_i q)\mathbf{e}_i$. The Jacobian of a vector-valued function $\mathbf{v} = v_i \mathbf{e}_i$ is given by $\nabla \mathbf{v} := \partial_k \mathbf{v} \otimes \mathbf{e}_k$. The components of the Jacobian in the Euclidean basis are given by $(\nabla \mathbf{v})_{ij} = \partial_j(\mathbf{v} \cdot \mathbf{e}_i)$. Thus, it holds $\nabla \mathbf{v} = \left(\nabla v_1 | \nabla v_2 | \nabla v_3\right)^T$. Note that for the definitions of these differential operators, we need the functions $q$ and $\mathbf{v}$ to be defined on an open subset of $\mathbb{R}^3$ and not only on a surface $\Gamma$. We emphasize the structural analogy between $\nabla \mathbf{v} = \partial_k \mathbf{v} \otimes \mathbf{e}_k$ and the definition of the surface Jacobian in local components $\widehat{\nabla}_\Gamma \mathbf{v} = \mathbf{P}\partial^\Gamma_\alpha \mathbf{v} \otimes \mathbf{g}^\alpha$ (cf. Definition 2.2.8).

In order to define a Cartesian representation of the *surface* differential operators, we extend functions defined on the surface to an open neighborhood of the surface. We define the (signed) distance function $d$ by

$$d(\mathbf{x}, t) = \begin{cases} \inf\limits_{\mathbf{y}\in\Gamma(t)} \|\mathbf{x} - \mathbf{y}\|, & \mathbf{x} \in \mathbb{R}^3 \backslash G(t), \\ -\inf\limits_{\mathbf{y}\in\Gamma(t)} \|\mathbf{x} - \mathbf{y}\|, & \mathbf{x} \in G(t), \end{cases}$$

where $G(t) \subset \mathbb{R}^3$ is the enclosed volume of $\Gamma(t)$, i.e., $\Gamma(t) = \partial G(t)$ holds. Thus, the signed distance function satisfies $|d(\mathbf{x}, t)| = \operatorname{dist}(\mathbf{x}, \Gamma(t))$, where $\operatorname{dist}(\mathbf{x}, \Gamma(t))$ denotes the Euclidean distance from $\mathbf{x}$ to $\Gamma(t)$. Since we assume the surface to be orientable, $d$ is well defined. We define a (tubular) neighborhood of the surface $\Gamma(t)$ by

$$\mathcal{N}^\delta_\Gamma(t) = \mathcal{N}^\delta(\Gamma(t)) = \{\mathbf{x} \in \mathbb{R}^3 \mid \operatorname{dist}(\mathbf{x}, \Gamma(t)) < \delta\} \tag{2.2.9}$$

for some $\delta > 0$. Following [DE13, Lemma 2.8.], there exists a $\delta > 0$, independent of $t$, such that $d \in C^m(\mathcal{N}^\delta_\Gamma(t))$ holds for some $m \geq 1$ depending on the smoothness of $\Gamma$. Again, we fix $t \in [0, t_{\text{end}}]$ and denote $\Gamma = \Gamma(t)$. The closest point projection is implicitly

defined by $\mathbf{p}(\mathbf{x}) := \mathbf{x} - d(\mathbf{x})\mathbf{n}(\mathbf{p}(\mathbf{x}))$. Here, $\mathbf{n} = \nabla d$ denotes the normal that is constant in the normal direction off the surface. Note that for $\delta$ small enough, the decomposition $\mathbf{x} = \mathbf{p}(\mathbf{x}) + d(\mathbf{x})\mathbf{n}(\mathbf{p}(\mathbf{x}))$ is unique on $\mathcal{N}_\Gamma^\delta$.

For a smooth (scalar, vector-valued, or operator-valued) function $f$ defined on $\Gamma$, we denote its smooth extension on $\mathcal{N}_\Gamma^\delta$ by $f^e$. The specific choice of extension is not essential. One may use a constant extension along normals, which is given by $f^e(\mathbf{x}) := f(\mathbf{p}(\mathbf{x}))$. If $\delta$ is small enough, the extension is well defined and smooth.

Now, we introduce surface differential operators based on the extended functions.

**Definition 2.2.11: Surface differential operators (Cartesian basis)**

For a scalar function $q \in C^1(\Gamma, \mathbb{R})$ the Cartesian surface gradient is defined by

$$\nabla_\Gamma q := \mathbf{P}\nabla q^e.$$

For a vector field $\mathbf{v} \in C^1(\Gamma, \mathbb{R}^3)$ we define the Cartesian surface Jacobian $\nabla_\Gamma \mathbf{v}$ by

$$\nabla_\Gamma \mathbf{v} := \mathbf{P}\nabla \mathbf{v}^e \mathbf{P}.$$

The Cartesian surface divergence of $\mathbf{v} \in C^1(\Gamma, \mathbb{R}^3)$ and $\mathbf{T} \in C^1(\Gamma, L(\mathbb{R}^3, \mathbb{R}^3))$ are defined by

$$\operatorname{div}_\Gamma \mathbf{v} := \operatorname{tr}(\mathbf{P}\nabla \mathbf{v}^e \mathbf{P}) = \operatorname{tr}(\nabla_\Gamma \mathbf{v}), \quad \operatorname{div}_\Gamma \mathbf{T} := \operatorname{div}_\Gamma (\mathbf{T}\mathbf{e}_i)\, \mathbf{e}_i,$$

respectively. Additionally, for a scalar function $q \in C^2(\Gamma, \mathbb{R})$, the Laplace-Beltrami operator is defined by

$$\Delta_\Gamma q := \operatorname{div}_\Gamma (\nabla_\Gamma q).$$

Note that in the literature (cf. [JOR18; Fri18; BDL20]), the representation of the surface divergence of the operator-valued function $\mathbf{T}$ in Cartesian coordinates is often based on the surface divergence of the vector field $\mathbf{T}^T\mathbf{e}_i$. Therefore, the surface divergence from the literature often differs from the one in Definition 2.2.11 in case of a non-symmetric operator $\mathbf{T}$. In this work, we use the definition from [PS16], leading to a Cartesian representation of the surface divergence based on the surface divergence of the vector field $\mathbf{T}\mathbf{e}_i$. The surface gradient and the surface Jacobian equal the ones from [JOR18; Miu17; GKL17]. With slight abuse of notation, we denote the extended function $f^e$ by $f$ in the remainder of this thesis.

### Relations between the surface differential operators in different coordinate systems

In this section, we derive relations between surface differential operators from Definitions 2.2.8 and 2.2.11. The results are as expected and have been used (implicitly) at several places in the literature. We show that the surface differential operators in local and global coordinates are identical. This is important, and it shows that one can compare results derived in different coordinate systems. Additionally, it is important for the numerical realization of surface differential operators since in practice, it is not obvious how to construct the local bases and one often uses the global representation of the surface differential operators.

Moreover, the theorem shows, that the definitions of the surface differential operators in Cartesian coordinates are independent of the choice of the extension and only depend on the function values on the surface.

**Theorem 2.2.12**

Let $q \in C^1(\Gamma,\mathbb{R})$, $\mathbf{v} \in C^1(\Gamma,\mathbb{R}^3)$ and $\mathbf{T} \in C^1(\Gamma, L(\mathbb{R}^3,\mathbb{R}^3))$. The surface gradients, Jacobians, and divergence operators defined in Definitions 2.2.8 and 2.2.11 satisfy

$$\nabla_\Gamma q = \widehat{\nabla}_\Gamma q, \quad \nabla_\Gamma \mathbf{v} = \widehat{\nabla}_\Gamma \mathbf{v}, \quad \operatorname{div}_\Gamma \mathbf{v} = \widehat{\operatorname{div}}_\Gamma \mathbf{v}, \quad \operatorname{div}_\Gamma \mathbf{T} = \widehat{\operatorname{div}}_\Gamma \mathbf{T}.$$

**Proof**

A proof for the first equality can be found in [PS16; DE13]. For completeness, we include an elementary proof. Using the chain rule (cf. Theorem A.1), we get

$$\partial_\alpha^\Gamma q(\mathbf{x}) = \partial_\alpha (q \circ R)(\boldsymbol{\xi}) = \partial_\alpha (q^e \circ R)(\boldsymbol{\xi}) = \partial_k q^e(R(\boldsymbol{\xi}))\,(\partial_\alpha R(\boldsymbol{\xi}) \cdot \mathbf{e}_k) = \partial_k q^e(\mathbf{x})\,(\mathbf{g}_\alpha \cdot \mathbf{e}_k),$$

where $\mathbf{x} = R(\boldsymbol{\xi}) \in \Gamma$. Thus, we conclude

$$\begin{aligned}\widehat{\nabla}_\Gamma q(\mathbf{x}) &= \partial_\alpha^\Gamma q(\mathbf{x})\mathbf{g}^\alpha = \big[\partial_k q^e(\mathbf{x})(\mathbf{g}_\alpha \cdot \mathbf{e}_k)\big]\mathbf{g}^\alpha = \partial_k q^e(\mathbf{x}) \overbrace{(\mathbf{g}_\alpha \cdot \mathbf{e}_k)\mathbf{g}^\alpha}^{=\mathbf{P}\mathbf{e}_k} \\ &= \partial_k q^e(\mathbf{x})\mathbf{P}\mathbf{e}_k = \mathbf{P}\big[\partial_k q^e(\mathbf{x})\mathbf{e}_k\big] = \mathbf{P}\nabla q^e(\mathbf{x}) = \nabla_\Gamma q(\mathbf{x}).\end{aligned}$$

This proves the first equality. For vector fields, the transposed Jacobian is given by

$$\nabla^T \mathbf{v}^e = \mathbf{e}_k \otimes \partial_k \mathbf{v}^e. \tag{2.2.10}$$

Using the chain rule, $\mathbf{x} = R(\boldsymbol{\xi}) \in \Gamma$, and the definition $\partial_\alpha^\Gamma \mathbf{v}(\mathbf{x}) := \partial_\alpha (\mathbf{v}^e \circ R)(\boldsymbol{\xi})$ we get

$$\partial_\alpha^\Gamma \mathbf{v}(\mathbf{x}) \cdot \mathbf{e}_i = \big(\partial_k \mathbf{v}^e(R(\boldsymbol{\xi})) \cdot \mathbf{e}_i\big)\big(\partial_\alpha R(\boldsymbol{\xi}) \cdot \mathbf{e}_k\big) = \big(\partial_k \mathbf{v}^e(\mathbf{x}) \cdot \mathbf{e}_i\big)\,(\mathbf{g}_\alpha \cdot \mathbf{e}_k)\,.$$

Combing this with (2.2.10) and using the surface gradient of a vector valued function, which is defined in Remark 2.2.9 by $\widehat{\nabla}_S \mathbf{v} = \mathbf{g}^\alpha \otimes \partial_\alpha^\Gamma \mathbf{v}$, we obtain

$$\begin{aligned}\widehat{\nabla}_S \mathbf{v}\,\mathbf{e}_i &= (\mathbf{g}^\alpha \otimes \partial_\alpha^\Gamma \mathbf{v})\mathbf{e}_i = (\partial_\alpha^\Gamma \mathbf{v} \cdot \mathbf{e}_i)\mathbf{g}^\alpha = \big[(\partial_k \mathbf{v}^e \cdot \mathbf{e}_i)(\mathbf{g}_\alpha \cdot \mathbf{e}_k)\big]\mathbf{g}^\alpha \\ &= (\partial_k \mathbf{v}^e \cdot \mathbf{e}_i)\,\mathbf{P}\mathbf{e}_k = \mathbf{P}\,(\mathbf{e}_k \otimes \partial_k \mathbf{v}^e)\,\mathbf{e}_i = \mathbf{P}\nabla^T \mathbf{v}^e \mathbf{e}_i.\end{aligned} \tag{2.2.11}$$

Since the vectors $\mathbf{e}_i$ form a basis of $\mathbb{R}^3$, it follows $\widehat{\nabla}_S \mathbf{v} = \mathbf{P}\nabla^T \mathbf{v}^e$. Using $\widehat{\nabla}_\Gamma \mathbf{v} = \mathbf{P}\widehat{\nabla}_S^T \mathbf{v}$ completes the proof of the relation for the surface Jacobian of $\mathbf{v}$. For the surface divergence of a vector function, we get

$$\begin{aligned}\widehat{\operatorname{div}}_\Gamma \mathbf{v} &= \partial_\alpha^\Gamma \mathbf{v} \cdot \mathbf{g}^\alpha = \delta_\beta^\alpha (\partial_\alpha^\Gamma \mathbf{v} \cdot \mathbf{g}^\beta) = (\mathbf{g}^\alpha \cdot \mathbf{g}_\beta)(\partial_\alpha^\Gamma \mathbf{v} \cdot \mathbf{g}^\beta) = [(\mathbf{g}^\alpha \otimes \partial_\alpha^\Gamma \mathbf{v})\mathbf{g}^\beta] \cdot \mathbf{g}_\beta \\ &= (\widehat{\nabla}_S \mathbf{v}\,\mathbf{g}^\beta) \cdot \mathbf{g}_\beta \overset{(2.2.11)}{=} (\mathbf{P}\nabla^T \mathbf{v}^e \mathbf{g}^\beta) \cdot \mathbf{g}_\beta = (\mathbf{P}\nabla^T \mathbf{v}^e \mathbf{P}\mathbf{g}^i) \cdot \mathbf{g}_i = \operatorname{tr}(\mathbf{P}\nabla \mathbf{v}^e \mathbf{P}) = \operatorname{div}_\Gamma \mathbf{v}.\end{aligned}$$

For the surface divergence of $\mathbf{T}$ ($\widehat{\operatorname{div}}_\Gamma \mathbf{T} = (\partial_\alpha^\Gamma \mathbf{T})^T \mathbf{g}^\alpha$), we have

$$\begin{aligned}\widehat{\operatorname{div}}_\Gamma \mathbf{T} \cdot \mathbf{e}_i &= (\mathbf{g}^\alpha \cdot \mathbf{g}_\beta)(\partial_\alpha^\Gamma \mathbf{T}\mathbf{e}_i) \cdot \mathbf{g}^\beta = ((\mathbf{g}^\alpha \otimes \partial_\alpha^\Gamma \mathbf{T}\mathbf{e}_i)\mathbf{g}^\beta) \cdot \mathbf{g}_\beta = ((\mathbf{g}^\alpha \otimes \partial_\alpha^\Gamma \mathbf{T}\mathbf{e}_i)\mathbf{P}\mathbf{g}^i) \cdot \mathbf{P}\mathbf{g}_i \\ &= \operatorname{tr}\Big(\mathbf{P}(\mathbf{g}^\alpha \otimes \partial_\alpha^\Gamma \mathbf{T}\mathbf{e}_i)\mathbf{P}\Big) = \operatorname{tr}\Big(\mathbf{P}\widehat{\nabla}_S(\mathbf{T}\mathbf{e}_i)\mathbf{P}\Big) \overset{(2.2.11)}{=} \operatorname{tr}\Big(\mathbf{P}\nabla^T(\mathbf{T}^e\mathbf{e}_i)\mathbf{P}\Big) = \operatorname{div}_\Gamma(\mathbf{T}\mathbf{e}_i).\end{aligned}$$

□

## 2.3 Moving material surfaces

In this section, we take a closer look at the time evolution of the surface $\Gamma(t)$ in $\mathbb{R}^3$. We introduce a time derivative of functions defined on the time-dependent surface $\Gamma(t)$ only. The standard time derivative cannot be used: Taking a function $f$ defined on $\Gamma(t)$ only and a point $\mathbf{x} \in \Gamma(t)$, the difference $f(\mathbf{x},t) - f(\mathbf{x},t+\varepsilon)$ is in general not defined since it holds $\mathbf{x} \notin \Gamma(t+\varepsilon)$ for $\varepsilon > 0$ and $f(\mathbf{x}+\varepsilon)$ is not well defined. Thus, the standard partial time derivative $\frac{\partial}{\partial t} f$ is not well defined for functions solely defined on the evolving surface $\Gamma$. We will define a time derivative that takes the movement of the surface into account. This derivative is called the *material* (or Lagrangian) derivative. We define the material derivative in terms of the parametrization $R(\boldsymbol{\xi},t)$ and derive a representation in terms of Cartesian coordinates as well as another representation that shows the dependence of the derivative of the particle trajectories defined in (2.1.1). After that, we derive the *rate of strain tensor* as the time derivative of the metric tensor.

Let $U \subset \mathbb{R}^2$ be an open set and $\Phi_U : U \to \Gamma(0)$ a local parametrization of the surface $\Gamma(0)$ as described in Section 2.1. Let $I = [0, t_{\text{end}}]$ be a time interval with end time $t_{\text{end}} > 0$ sufficiently small such that for all $\mathbf{z} \in \Phi_U(U) \subset \Gamma(0)$ the trajectories from (2.1.1) are well defined and have a unique solution for $t \in I$.

We define the (local) evolving surface and the space-time manifold by

$$\begin{aligned} \Gamma_U(t) &:= \{\, \mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x} = R(\boldsymbol{\xi},t),\ \boldsymbol{\xi} \in U \,\} \quad \text{for } t \in I, \\ \mathcal{S}_U &:= \bigcup_{t \in I} \Gamma_U(t) \times \{t\} \subset \mathbb{R}^4. \end{aligned}$$

Note that $\mathcal{S}_U$ is parametrized by $\mathcal{R} : U \times I \to \mathcal{S}_U$, $\mathcal{R}(\boldsymbol{\xi},t) = (R(\boldsymbol{\xi},t),t)$.

Given the velocity $\mathbf{u}(\mathbf{x},t)$ for $\mathbf{x} \in \Gamma_U(t)$ that defines the trajectories, we define the corresponding velocity $\overline{\mathbf{u}}(\boldsymbol{\xi},t) := \mathbf{u}(R(\boldsymbol{\xi},t),t)$ for $(\boldsymbol{\xi},t) \in U \times I$. The following relation holds

$$\overline{\mathbf{u}}(\boldsymbol{\xi},t) = \frac{\partial}{\partial t} R(\boldsymbol{\xi},t) \quad \text{on } U \times I, \tag{2.3.1}$$

cf. [EHZ07, page 3]. Here, $\frac{\partial}{\partial t}$ denotes the standard time-derivative of the function $R : \mathbb{R}^2 \supset U \times I \to \mathbb{R}^3$.

### 2.3.1 The material derivative

We define the material derivative of a function defined on the surface $\Gamma(t)$. We start with a definition in terms of the parametrization $R(\boldsymbol{\xi},t)$, i.e., in the local coordinates $\boldsymbol{\xi}$ of the surface $\Gamma(0)$. After that, we derive a representation of the material derivative in terms of Cartesian coordinates.

**Definition 2.3.1: The material derivative**

Let $f \in C^1(\mathcal{S})$ be a scalar or vector function and $\overline{f} \in C^1(U \times I)$ the function defined by $\overline{f}(\boldsymbol{\xi},t) = f(R(\boldsymbol{\xi},t),t)$ for $(\boldsymbol{\xi},t) \in U \times I$. The *material* derivative of $f$ is defined by

$$\dot{f}(\mathbf{x},t) := \frac{\partial}{\partial t} \overline{f}(\boldsymbol{\xi},t), \quad \mathbf{x} = R(\boldsymbol{\xi},t).$$

We obtain a Cartesian representation of the material derivative along the same lines as in the previous section: We extend the functions defined on the space-time manifold $\mathcal{S}$ to an open neighborhood $\mathcal{N}_{\mathcal{S}}^{\delta}$ defined by

$$\mathcal{N}_{\mathcal{S}_U}^{\delta} = \mathcal{N}^{\delta}(\mathcal{S}_U) := \bigcup_{t\in I} \mathcal{N}^{\delta}(\Gamma_U(t)) \times \{t\},$$

where $\mathcal{N}^{\delta}(\Gamma_U(t))$ is the neighborhood defined in (2.2.9).

The following lemma yields a representation of the material derivative defined above in terms of derivatives with respect to Cartesian coordinates in $\mathbb{R}^3 \times \mathbb{R}$. The result is well-known and easy to prove based on the application of the chain rule. For completeness, we include an elementary proof.

**Lemma 2.3.2**

Let $q \in C^1(\mathcal{S}, \mathbb{R})$ be a scalar- and $\mathbf{v} \in C^1(\mathcal{S}, \mathbb{R}^3)$ a vector-valued functions with smooth extensions $q^e \in C^1(\mathcal{N}_{\mathcal{S}}^{\delta}, \mathbb{R})$ and $\mathbf{v}^e \in C^1(\mathcal{N}_{\mathcal{S}}^{\delta}, \mathbb{R}^3)$ satisfying $q^e|_{\Gamma(t)} = q$ and $\mathbf{v}^e|_{\Gamma(t)} = \mathbf{v}$. Then, the material derivatives of $q$ and $\mathbf{v}$ satisfy

$$\dot{q} = \frac{\partial}{\partial t} q^e + \nabla q^e \cdot \mathbf{u}, \quad \text{and} \quad \dot{\mathbf{v}} = \frac{\partial}{\partial t} \mathbf{v}^e + \nabla \mathbf{v}^e \mathbf{u} \quad \text{on } \mathcal{S},$$

where $\mathbf{u}$ is the material velocity defining the evolution of the surface.

**Proof**

For $(\mathbf{x}, t) \in \mathcal{S}$ we write $\mathbf{x} = R(\boldsymbol{\xi}, t)$. We use the chain rule (Theorem A.1 from the appendix) for the function $q^e(R(\boldsymbol{\xi}, t), t) = (q^e \circ F)(t)$ with the auxiliary function $F : I \to \mathbb{R}^4$, defined by $F(t) = (R(\boldsymbol{\xi}, t), t)$ and get

$$\begin{aligned}
\dot{q}(\mathbf{x}, t) &= \frac{\partial}{\partial t} \overline{q}(\boldsymbol{\xi}, t) = \frac{\mathrm{d}}{\mathrm{d}t} q(R(\boldsymbol{\xi}, t), t) \\
&= \sum_{k=1}^{3} \partial_k q^e(R(\boldsymbol{\xi}, t), t) \left( \frac{\partial}{\partial t} R(\boldsymbol{\xi}, t) \cdot \mathbf{e}_k \right) + \frac{\partial}{\partial t} q^e(R(\boldsymbol{\xi}, t), t) \cdot 1 \\
&= \frac{\partial}{\partial t} q^e(R(\boldsymbol{\xi}, t), t) + \nabla q^e(R(\boldsymbol{\xi}, t), t) \cdot \frac{\partial}{\partial t} R(\boldsymbol{\xi}, t).
\end{aligned}$$

Using $\mathbf{x} = R(\boldsymbol{\xi}, t)$ and relation (2.3.1), we obtain

$$\dot{q}(\mathbf{x}, t) = \frac{\partial}{\partial t} q^e(\mathbf{x}, t) + \nabla q^e(\mathbf{x}, t) \cdot \mathbf{u}(\mathbf{x}, t).$$

The same arguments can be used to derive the relation for $\mathbf{v}$. □

As can be seen in Theorem 2.2.12, the Cartesian representations of the material derivatives do not depend on the choice of the extension. A constant extension along normals may be used, but is not essential. Furthermore, the Cartesian representation of the material derivative is independent of the choice of the local representation $R$. Thus, Definition 2.3.1 does not depend on the choice of $R$.

Now, we show the equivalence of our definition of the material derivative from Definition 2.3.1 and the one from [PS16]. The representation from [PS16] emphasizes the

material derivative as a derivative along the trajectories defined by the velocity field $\mathbf{u}$ in (2.1.1). Therefore, we generalize the definition of the trajectory by calling $\mathbf{X}_{\mathbf{z},t}(t+s)$ the trajectory on $\mathcal{S}$ through $\mathbf{z} \in \Gamma(t)$. It is the unique $C^1$-curve $[s \mapsto \mathbf{X}_{\mathbf{z},t}(t+s)] : (-\delta, \delta) \to \mathbb{R}^3$ such that

$$\begin{cases} \mathbf{X}_{\mathbf{z},t}(t) = \mathbf{z}, \\ \dfrac{\mathrm{d}}{\mathrm{d}s}\mathbf{X}_{\mathbf{z},t}(t+s) = \mathbf{u}(\mathbf{X}_{\mathbf{z},t}(t+s), t+s) \quad \text{for } s \in (-\delta, \delta), \end{cases} \tag{2.3.2}$$

holds for some $\delta > 0$. If $\mathbf{u}$ is a $C^1$−velocity and $\delta > 0$ is sufficiently small, there exists a solution of (2.3.2) for each $\mathbf{z} \in \Gamma(t)$, see [PS16, page 79f] for a proof. The trajectory from (2.3.2) equals the one from (2.1.1) for $t = 0$. Now, we can give a representation using this trajectory of the material derivative in the following lemma.

**Lemma 2.3.3**

Let $f \in C^1(\mathcal{S})$ be a scalar- or vector-valued function. For the material derivative of $f$, we have

$$\overset{\bullet}{f}(\mathbf{z}, t) = \frac{\mathrm{d}}{\mathrm{d}s} f\left(\mathbf{X}_{\mathbf{z},t}(t+s), t+s\right)\Big|_{s=0} \qquad \text{for } (\mathbf{z}, t) \in \mathcal{S}.$$

**Proof**

As in Definition 2.3.1 let $\overline{f} \in C^1(U \times I)$ be defined by $\overline{f}(\boldsymbol{\xi}, t) = f\left(R(\boldsymbol{\xi}, t), t\right)$. We start by providing a proof of a connection between the trajectory defined in (2.3.2) and the immersion $R$. By construction, there exists a unique $\boldsymbol{\xi} \in U$ satisfying $R(\boldsymbol{\xi}, t) = \mathbf{z}$. Using equation (2.3.1), we get

$$\frac{\mathrm{d}}{\mathrm{d}s} R(\boldsymbol{\xi}, t+s) = \overline{\mathbf{u}}(\boldsymbol{\xi}, t+s) = \mathbf{u}\left(R(\boldsymbol{\xi}, t+s), t+s\right) \qquad \text{for } s \in (-\delta, \delta)$$

for some $\delta > 0$. Hence, $[s \mapsto R(\boldsymbol{\xi}, t+s)]$ is the trajectory through $\mathbf{z} = R(\boldsymbol{\xi}, t)$ as defined in (2.3.2). It follows for $(\mathbf{z}, t) \in \mathcal{S}$ that

$$\begin{aligned} \overset{\bullet}{f}(\mathbf{z}, t) = \partial_t \overline{f}(\boldsymbol{\xi}, t) &= \frac{\mathrm{d}}{\mathrm{d}s}\overline{f}(\boldsymbol{\xi}, t+s)\Big|_{s=0} = \frac{\mathrm{d}}{\mathrm{d}s} f\left(R(\boldsymbol{\xi}, t+s), t+s\right)\Big|_{s=0} \\ &= \frac{\mathrm{d}}{\mathrm{d}s} f\left(\mathbf{X}_{\mathbf{z},t}(t+s), t+s\right)\Big|_{s=0} \end{aligned}$$

holds. □

**Remark 2.3.4**

Lemmas 2.3.2 and 2.3.3 show that the definition of the material derivative equals the ones from [JOR18; GKL17; Miu17; PS16]. ◇

### 2.3.2 Time derivative of the first fundamental form

In this section, we consider a time derivative of the first fundamental form or metric tensor $g_{\alpha\beta}$, that will be used in the remainder. The local bases $\mathbf{g}_i$ and $\mathbf{g}^i$ introduced in Section 2.2.1 depend on the time variable $t$. In particular, it holds $\mathbf{g}_\alpha = \overline{\partial}_\alpha R$ and thus, $\mathbf{g}_\alpha = \mathbf{g}_\alpha(\boldsymbol{\xi}, t)$, $(\boldsymbol{\xi}, t) \in U \times I$. Hence, the first fundamental form $g_{\alpha\beta} = g_{\alpha\beta}(\boldsymbol{\xi}, t)$, cf. (2.2.2), also depends on the time variable. The change (as a function of time) of the metric tensor is determined by the velocity field $\mathbf{u}$, which determines the time

dependence of the parametrization $R = R(\boldsymbol{\xi}, t) = \Phi_t(\Phi_U(\boldsymbol{\xi}))$ via the flow map $\Phi_t$. This time derivative of the metric tensor, scaled with a factor $\frac{1}{2}$, is called *rate of strain tensor*, and its covariant components are given by

$$\hat{E}_{\Gamma,\alpha\beta} := \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t} g_{\alpha\beta}. \tag{2.3.3}$$

For a given $(\boldsymbol{\xi}, t) \in \mathcal{S}$ a corresponding linear operator $\mathbf{E}_\Gamma = \mathbf{E}_\Gamma(\boldsymbol{\xi}, t) : \mathbb{R}^3 \to \mathbb{R}^3$ is given by $\mathbf{E}_\Gamma := \hat{E}_{\Gamma,\alpha\beta}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) = \hat{E}_\Gamma^{\alpha\beta}(\mathbf{g}_\alpha \otimes \mathbf{g}_\beta)$. Since the metric tensor and thus also the rate of strain tensor depends on the velocity $\mathbf{u}$, we often write $\mathbf{E}_\Gamma = \mathbf{E}_\Gamma(\mathbf{u})$. This operator can also be expressed in terms of the covariant derivatives introduced in Definitions 2.2.8 or 2.2.11, as shown in the following lemma.

**Lemma 2.3.5: Rate of strain**

The rate of strain can be represented as

$$\mathbf{E}_\Gamma = \mathbf{E}_\Gamma(\mathbf{u}) = \hat{E}_{\Gamma,\alpha\beta}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) = \frac{1}{2}(\nabla_\Gamma \mathbf{u} + \nabla_\Gamma^T \mathbf{u}) = \frac{1}{2}(\mathbf{g}^\alpha \otimes \mathbf{P}\partial_\alpha^\Gamma \mathbf{u} + \mathbf{P}\partial_\alpha^\Gamma \mathbf{u} \otimes \mathbf{g}^\alpha).$$

Here, the transposed of the surface gradient is denoted by $\nabla_\Gamma^T \mathbf{u} := (\nabla_\Gamma \mathbf{u})^T$.

**Proof**

Using Theorem 2.2.6 and equation (2.3.1), the following relation for the time derivative of the metric tensor is derived. With $\mathbf{u} = \hat{u}_i \mathbf{g}^i = \hat{u}^i \mathbf{g}_i$, it holds

$$\begin{aligned}\partial_t g_{\alpha\beta} &= \partial_t \mathbf{g}_\alpha \cdot \mathbf{g}_\beta + \mathbf{g}_\alpha \cdot \partial_t \mathbf{g}_\beta = \partial_t \overline{\partial}_\alpha R \cdot \mathbf{g}_\beta + \mathbf{g}_\alpha \cdot \partial_t \overline{\partial}_\beta R = \overline{\partial}_\alpha \overline{\mathbf{u}} \cdot \mathbf{g}_\beta + \overline{\partial}_\beta \overline{\mathbf{u}} \cdot \mathbf{g}_\alpha \\ &= \partial_\alpha^\Gamma \mathbf{u} \cdot \mathbf{g}_\beta + \partial_\beta^\Gamma \mathbf{u} \cdot \mathbf{g}_\alpha = \hat{u}_{\alpha|\beta} + \hat{u}_{\beta|\alpha} + 2\hat{u}_3 \hat{H}_{\alpha\beta}.\end{aligned}$$

Using $\left(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta\right)^T = \left(\mathbf{g}^\beta \otimes \mathbf{g}^\alpha\right)$ and (2.2.7), we get

$$\left(\hat{u}_{\alpha|\beta} + \hat{u}_{\beta|\alpha}\right)\left(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta\right) = \hat{u}_{\alpha|\beta}\left(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta\right) + \hat{u}_{\alpha|\beta}\left(\mathbf{g}^\beta \otimes \mathbf{g}^\alpha\right) = \nabla_\Gamma \mathbf{u}_T + \nabla_\Gamma^T \mathbf{u}_T.$$

A direct calculation, using (2.2.4) and the representation of the surface Jacobian from Definition 2.2.8, yields

$$\begin{aligned}u_N \hat{H}_{\alpha\beta}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) &= \frac{1}{2}\left(u_N \hat{H}_{\alpha\beta}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) + u_N \hat{H}_{\alpha\beta}(\mathbf{g}^\beta \otimes \mathbf{g}^\alpha)\right) \\ = \frac{1}{2}\left(\mathbf{g}^\alpha \otimes \mathbf{P}\partial_\alpha^\Gamma(u_N \mathbf{n}) + \mathbf{P}\partial_\alpha^\Gamma(u_N \mathbf{n}) \otimes \mathbf{g}^\alpha\right) &= \frac{1}{2}(\nabla_\Gamma(u_N \mathbf{n}) + \nabla_\Gamma^T(u_N \mathbf{n})).\end{aligned}$$

Putting these results together yields

$$\begin{aligned}\mathbf{E}_\Gamma(\mathbf{u}) = \hat{E}_{\Gamma,\alpha\beta}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) &= \left(\frac{1}{2}(\hat{u}_{\alpha|\beta} + \hat{u}_{\beta|\alpha}) + u_N \hat{H}_{\alpha\beta}\right)(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) \\ &= \frac{1}{2}\left(\hat{u}_{\alpha|\beta} + \hat{u}_{\beta|\alpha}\right)\left(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta\right) + u_N \hat{H}_{\alpha\beta}\left(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta\right) \\ &= \frac{1}{2}\left(\nabla_\Gamma \mathbf{u}_T + \nabla_\Gamma^T \mathbf{u}_T\right) + \frac{1}{2}\left(\nabla_\Gamma(u_N \mathbf{n}) + \nabla_\Gamma^T(u_N \mathbf{n})\right) = \frac{1}{2}(\nabla_\Gamma \mathbf{u} + \nabla_\Gamma^T \mathbf{u}).\end{aligned}$$

The definition of the surface gradient 2.2.8 yields the final equality. □

**Remark 2.3.6**

Lemma 2.3.5 shows that our definition of the rate of strain tensor (2.3.3) equals the one from [PSW20, page 19], [EHZ07, equation (13)], and [JOR18]. ◇

## 2.4 Integrals on surfaces

In this section, we introduce a surface integral for stationary and evolving surfaces. We recall the well-known partial integration identities (divergence theorem) and the Reynolds transport theorem.

Let $\mathbf{v}$ be a vector valued function defined on the surface $\Gamma$. The (surface) integral of $\mathbf{v}$ is defined in the usual way: We choose a fixed (not necessarily orthogonal) basis of $\mathbb{R}^3$, say $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$. For $\mathbf{v}(\mathbf{x}) = \tilde{v}^i(\mathbf{x})\mathbf{w}_i$, we then define

$$\int_\Gamma \mathbf{v}(\mathbf{x})\,\mathrm{d}s := \mathbf{w}_i \int_\Gamma \tilde{v}^i(\mathbf{x})\,\mathrm{d}s$$

and similarly for integrals on moving surfaces $\Gamma(t)$ and subsets of surfaces. The results derived below are independent of the choice of $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$. We recall well-known integration identities, starting with the integration by parts for closed surfaces (i.e., without boundary).

**Theorem 2.4.1: Integration by parts**

For a tangential vector field $\mathbf{v} \in C^1(\Gamma)^3$ satisfying $\mathbf{P}\mathbf{v} = \mathbf{v}$ and a scalar function $q \in C^1(\Gamma)$, the relation

$$\int_\Gamma (\mathrm{div}_\Gamma\, \mathbf{v}) q\,\mathrm{d}s = -\int_\Gamma \mathbf{v} \cdot \nabla_\Gamma q\,\mathrm{d}s$$

holds. A tensor valued function $\mathbf{T} \in C^1(\Gamma)^{3\times 3}$ satisfying $\mathbf{P}\mathbf{T} = \mathbf{T}\mathbf{P} = \mathbf{T}$ and a vector valued function $\mathbf{w} \in C^1(\Gamma)^3$ satisfy

$$\int_\Gamma (\mathrm{div}_\Gamma\, \mathbf{T}) \cdot \mathbf{w}\,\mathrm{d}s = -\int_\Gamma \mathbf{T}^T : \nabla_\Gamma \mathbf{w}\,\mathrm{d}s,$$

where $\mathbf{A} : \mathbf{B} = \mathrm{tr}(\mathbf{A}^T\mathbf{B})$ denotes the usual scalar product on $\mathbb{R}^{n\times n}$.

**Proof**

The first equality is proven in [GR11, equation (14.16)]. The second result follows from a componentwise application of the first equality and Lemma 2.5.1. It holds

$$\begin{aligned}
\int_\Gamma (\mathrm{div}_\Gamma\, \mathbf{T}) \cdot \mathbf{w}\,\mathrm{d}s &= \int_\Gamma (\mathrm{div}_\Gamma\, \mathbf{T}\mathbf{e}_i)\, w_i\,\mathrm{d}s = -\int_\Gamma \mathbf{T}\mathbf{e}_i \cdot \nabla_\Gamma w_i\,\mathrm{d}s \\
&\overset{(2.5.4)}{=} -\int_\Gamma T_{ji}\,(\nabla_\Gamma w_i)_j\,\mathrm{d}s = -\int_\Gamma \mathrm{tr}\,(\mathbf{T}\nabla\mathbf{w}\mathbf{P})\,\mathrm{d}s \\
&= -\int_\Gamma \mathrm{tr}\,(\mathbf{T}\mathbf{P}\nabla\mathbf{w}\mathbf{P})\,\mathrm{d}s = -\int_\Gamma \mathbf{T}^T : \nabla_\Gamma \mathbf{w}\,\mathrm{d}s,
\end{aligned}$$

which completes the proof. □

Using this theorem and the identity

$$\mathbf{S} : \nabla_\Gamma \mathbf{v} = \frac{1}{2}\left(\mathrm{tr}(\mathbf{S}^T\nabla_\Gamma \mathbf{v}) + \mathrm{tr}(\mathbf{S}^T\nabla_\Gamma \mathbf{v})\right) = \frac{1}{2}\left(\mathrm{tr}(\mathbf{S}^T\nabla_\Gamma \mathbf{v}) + \mathrm{tr}(\mathbf{S}^T\nabla_\Gamma^T \mathbf{v})\right) = \mathbf{S} : \mathbf{E}_\Gamma(\mathbf{v})$$

for every symmetric matrix $\mathbf{S} = \mathbf{S}^T$, we get

$$\int_\Gamma (\operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u})) \cdot \mathbf{v} \, \mathrm{d}s = - \int_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) : \mathbf{E}_\Gamma(\mathbf{v}) \, \mathrm{d}s.$$

An important tool in the derivation of Navier–Stokes equations is the transport theorem, which is also given in [JOR18, equation (3.1)].

**Theorem 2.4.2: Transport theorem**

Let $\gamma(t) \subset \Gamma(t)$ be a subdomain of the surface $\Gamma(t)$ that is advected by the material velocity $\mathbf{u}$ and a scalar- or vector-valued function $f \in C^1(S)$. Then, it holds

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\gamma(t)} f \, \mathrm{d}s = \int_{\gamma(t)} \overset{\bullet}{f} + f \operatorname{div}_\Gamma \mathbf{u} \, \mathrm{d}s.$$

**Proof**

A proof for the scalar case is given in, e.g., [DE13, Theorem 5.1]. For the vector-valued case, the proof can easily be derived using a componentwise application of the scalar version. □

Using equation (2.2.3) and Lemma 2.3.5, we can rewrite $\operatorname{div}_\Gamma \mathbf{u}$ as

$$\operatorname{div}_\Gamma \mathbf{u} = \operatorname{tr}(\nabla_\Gamma \mathbf{u}) = \operatorname{tr}\left(\frac{1}{2}(\nabla_\Gamma \mathbf{u} + \nabla_\Gamma^T \mathbf{u})\right) = \operatorname{tr}\left(\mathbf{E}_\Gamma(\mathbf{u})\right) = \hat{E}^\alpha_{\Gamma,\alpha}, \tag{2.4.1}$$

where $\hat{E}^\alpha_{\Gamma,\alpha} = g^{\alpha\beta} \hat{E}_{\Gamma,\beta\alpha}$. This relation shows the equality of Theorem 2.4.2 and the Reynolds transport theorem from [EHZ07, equation (15)].

An important integral identity for surfaces *with* boundary is the divergence theorem, taken from [GR11, equations (14.18) and (14.19)].

**Theorem 2.4.3: Divergence theorem**

Let $\gamma \subset \Gamma$ be a hypersurface with boundary $\partial\gamma$. The unit outer normal to $\partial\gamma$ that is tangential to $\Gamma$ is denoted by $\boldsymbol{\nu}$ and the surface measure on $\partial\gamma$ by $\mathrm{d}\tilde{s}$. It follows

$$\begin{aligned}
\int_\gamma \nabla_\Gamma q \, \mathrm{d}s &= \int_\gamma q\kappa \mathbf{n} \, \mathrm{d}s + \int_{\partial\gamma} q\boldsymbol{\nu} \, \mathrm{d}\tilde{s} && \text{for all } q \in C^1(\bar{\gamma}), \\
\int_\gamma \operatorname{div}_\Gamma \mathbf{v} \, \mathrm{d}s &= \int_{\partial\gamma} \mathbf{v} \cdot \boldsymbol{\nu} \, \mathrm{d}\tilde{s} && \text{for all } \mathbf{v} \in C^1(\bar{\gamma})^3, \\
\int_\gamma \operatorname{div}_\Gamma (\mathbf{TP}) \, \mathrm{d}s &= \int_{\partial\gamma} \mathbf{T}\boldsymbol{\nu} \, \mathrm{d}\tilde{s} && \text{for all } \mathbf{T} \in C^1(\bar{\gamma})^{3\times 3}.
\end{aligned}$$

## 2.5 Properties of surface differential operators and useful identities

We summarize some important properties and calculation rules of the surface differential operators introduced in Section 2.2.2. The properties are used throughout the remainder of this thesis. Again, we use the Einstein summation convention. Differential operators

for smooth scalar functions $q$, vector-valued functions $\mathbf{v}$ and matrix-valued functions $\mathbf{T}$ are given by

$$\begin{aligned} (\nabla_\Gamma q)_i &= P_{ik}\partial_k q, & \operatorname{div}_\Gamma \mathbf{v} &= (\nabla_\Gamma \mathbf{v})_{ii} = P_{ik}\partial_k \mathbf{v}_l P_{li} = P_{lk}\partial_k \mathbf{v}_l, \\ (\nabla_\Gamma \mathbf{v})_{ij} &= P_{ik}\partial_l \mathbf{v}_k P_{lj}, \quad \text{and} & (\operatorname{div}_\Gamma \mathbf{T})_i &= \operatorname{div}_\Gamma(\mathbf{T}\mathbf{e}_i) = P_{lk}\partial_k \mathbf{T}_{li}, \end{aligned}$$

with the partial differentiation $\partial_k = \dfrac{\partial}{\partial x_k}$ in $\mathbb{R}^3$. Note that we identify functions $f$ defined on $\Gamma$ with their smooth extension $f^e$.

**Lemma 2.5.1: Differentiation rules**

The following identities hold for scalar functions $p, q \in C^1(\Gamma)$, vector-valued functions $\mathbf{v}, \mathbf{w} \in C^1(\Gamma)^3$ and tensor-valued functions $\mathbf{T} \in C^1(\Gamma)^{3\times 3}$ that satisfy $\mathbf{PT} = \mathbf{TP} = \mathbf{T}$.

(i) $\nabla_\Gamma(p \cdot q) = \nabla_\Gamma p \cdot q + \nabla_\Gamma q \cdot p$

(ii) $\nabla_\Gamma(q \cdot \mathbf{v}) = q\nabla_\Gamma \mathbf{v} + \mathbf{v}_T \nabla_\Gamma^T q$

(iii) $\nabla_\Gamma \mathbf{v} = \nabla_\Gamma \mathbf{v}_T + \mathbf{H} v_N$ (2.5.1)

(iv) $$\begin{aligned} \nabla_\Gamma(\mathbf{v} \cdot \mathbf{w}) &= \nabla_\Gamma^T \mathbf{v}_T \mathbf{w} + \nabla_\Gamma^T \mathbf{w}_T \mathbf{v} + \nabla_\Gamma(v_N \cdot w_N) \\ &= \nabla_\Gamma^T \mathbf{v}_T \mathbf{w} + \nabla_\Gamma^T \mathbf{w}_T \mathbf{v} + v_N \nabla_\Gamma w_N + w_N \nabla_\Gamma v_N \end{aligned}$$

(v) $\operatorname{div}_\Gamma(q\mathbf{v}) = \nabla_\Gamma q \cdot \mathbf{v} + q \operatorname{div}_\Gamma \mathbf{v}$ (2.5.2)

(vi) $\operatorname{div}_\Gamma(q\mathbf{T}) = q \operatorname{div}_\Gamma \mathbf{T} + \mathbf{T}^T \nabla_\Gamma q$ (2.5.3)

(vii) $\nabla_\Gamma(v_i) = (\nabla \mathbf{v}\mathbf{P})_{i,-}$ (2.5.4)

Here, $(v_i) = \mathbf{v} \cdot \mathbf{e}_i$ denotes the $i$-th Cartesian component of $\mathbf{v}$.

A proof of this Lemma is given in Appendix A.

**Remark 2.5.2**

Applying (iii) on tangential functions $\mathbf{v} = \mathbf{v}_T$ and $\mathbf{w} = \mathbf{w}_T$, we get a product rule given by

$$\nabla_\Gamma(\mathbf{v} \cdot \mathbf{w}) = \nabla_\Gamma^T \mathbf{v}\mathbf{w} + \nabla_\Gamma^T \mathbf{w}\mathbf{v}.$$

This relation resembles the equation

$$\nabla(\mathbf{v} \cdot \mathbf{w}) = \nabla^T \mathbf{v}\mathbf{w} + \nabla^T \mathbf{w}\mathbf{v},$$

which formulates the product rule in the Euclidean space. ◇

Next, we prove a different representation of the tangential projection using the closest point projection given by $\mathbf{p}(\mathbf{x}) = \mathbf{x} - d(\mathbf{x})\mathbf{n}(\mathbf{x})$, with $d$ the signed distance function and $\mathbf{n} = \nabla d$ the unit normal vector defined in a neighborhood of the surface.

**Lemma 2.5.3**

The tangential projection $\mathbf{P}$ can be represented by the closest point projection $\mathbf{p}$ as

$$\mathbf{P} = \mathbf{I} - \mathbf{n}\mathbf{n}^T = \nabla \mathbf{p} - d\mathbf{H}.$$

Thus, on $\Gamma$, it holds

$$\mathbf{P} = \nabla \mathbf{p}.$$

**Proof**

An elementary calculation yields

$$\begin{aligned}\nabla \mathbf{p}(\mathbf{x}) &= \nabla(\mathbf{x} - d(\mathbf{x})\mathbf{n}(\mathbf{x})) = \nabla \mathbf{x} - \nabla(d(\mathbf{x})\mathbf{n}(\mathbf{x})) = \mathbf{I} - \mathbf{n}(\mathbf{x})\nabla^T d(\mathbf{x}) + d(\mathbf{x})\nabla \mathbf{n}(\mathbf{x}) \\ &= \mathbf{I} - \mathbf{n}(\mathbf{x})\mathbf{n}(\mathbf{x})^T + d(\mathbf{x})\mathbf{H}(\mathbf{x}) = \mathbf{P}(\mathbf{x}) + d(\mathbf{x})\mathbf{H}(\mathbf{x}).\end{aligned}$$

The second equation follows since $d = 0$ holds on $\Gamma$. □

The following lemma gives a representation of the surface divergence of a vector field that highlights the dependence on the mean curvature of the surface. In the proof, we use the representation of the mean curvature by the surface divergence of the normal vector field $\mathbf{n}$ given by $\kappa = \operatorname{div}_\Gamma(\mathbf{n})$.

**Lemma 2.5.4**

Let $\mathbf{v} \in C^1(\Gamma)^3$ be a vector field defined on $\Gamma$. Then, it holds

$$\operatorname{div}_\Gamma \mathbf{v} = \operatorname{div}_\Gamma \mathbf{v}_T + \kappa v_N. \tag{2.5.5}$$

**Proof**

A simple calculation yields

$$\operatorname{div}_\Gamma \mathbf{u} = \operatorname{div}_\Gamma \mathbf{u}_T + \operatorname{div}_\Gamma(u_N \mathbf{n}) \overset{(2.5.2)}{=} \operatorname{div}_\Gamma \mathbf{u}_T + u_N \operatorname{div}_\Gamma \mathbf{n} + \mathbf{n} \cdot \nabla_\Gamma u_N = \operatorname{div}_\Gamma \mathbf{u}_T + \kappa u_N,$$

where we used $\mathbf{n} \cdot \nabla_\Gamma u_N = 0$. □

In the derivation of the Navier–Stokes equations, we need the surface divergence of the product of a scalar function with the tangential projection. It is given in the following lemma.

**Lemma 2.5.5**

Let $q \in C^1(\Gamma)$ be a scalar field and $\mathbf{P}$ the projection onto the tangent bundle of $\Gamma$. Then the surface divergence of the product $q \cdot \mathbf{P}$ is given by

$$\operatorname{div}_\Gamma(q\mathbf{P}) = \nabla_\Gamma q - q\kappa \mathbf{n}.$$

**Proof**

We start with a proof of an auxiliary result. For the outer product of the normal vector $\mathbf{n}$ with itself, it holds

$$\operatorname{div}_\Gamma\left(\mathbf{n}\mathbf{n}^T\right)_i = \operatorname{div}_\Gamma\left(\mathbf{n}\mathbf{n}^T\mathbf{e}_i\right) = \operatorname{div}_\Gamma(\mathbf{n}n_i) \overset{(2.5.2)}{=} \underbrace{\nabla_\Gamma n_i \cdot \mathbf{n}}_{=0} + n_i \underbrace{\operatorname{div}_\Gamma \mathbf{n}}_{=\kappa} = \kappa n_i.$$

Using the product rule for the divergence (2.5.3) from Lemma 2.5.1 and the auxiliary equation, we get

$$\operatorname{div}_\Gamma(q\mathbf{P}) = \mathbf{P}^T\nabla_\Gamma q + q\operatorname{div}_\Gamma(\mathbf{P}) = \nabla_\Gamma q + q\operatorname{div}_\Gamma(\mathbf{I} - \mathbf{n}\mathbf{n}^T) = \nabla_\Gamma q - q\kappa\mathbf{n}.$$

□

The identity function on $\Gamma$ is defined by $\mathrm{id}_\Gamma : \Gamma \to \mathbb{R}^3,\ \mathbf{x} \mapsto \mathbf{x}$. Applying the Laplace-Beltrami operator and utilizing Lemma 2.5.5 yields the following corollary.

**Corollary 2.5.6**

The mean curvature vector $\kappa\mathbf{n}$ can be represented as

$$\kappa\mathbf{n} = -\Delta_\Gamma \mathrm{id}_\Gamma.$$

**Proof**

It holds

$$\Delta_\Gamma \mathrm{id}_\Gamma = \operatorname{div}_\Gamma(\nabla_\Gamma \mathrm{id}_\Gamma) = \operatorname{div}_\Gamma(\mathbf{P}\nabla \mathrm{id}_\Gamma^e \mathbf{P}) = \operatorname{div}_\Gamma(\mathbf{P}\mathbf{I}\mathbf{P}) = \operatorname{div}_\Gamma(\mathbf{P}^2) = \operatorname{div}_\Gamma(\mathbf{P}) = -\kappa\mathbf{n},$$

where we used Lemma 2.5.5 for $q = 1$. □

## 2.6 Representations of the surface: The level set function

There are different ways to represent the surface $\Gamma(t)$. Given the trajectories (2.1.1) or (2.1.3), one can use the *explicit* representation of $\Gamma(t)$ from (2.1.2) when an explicit representation of the initial surface $\Gamma^0$ is known. Since this is hard to realize in a numerical setting, we use an *implicit* representation of the surface in this thesis given by a level set function $\phi = \phi(\mathbf{x}, t)$. This level set method is a widely used numerical method for the representation and approximation of moving surfaces (or interfaces) [OF03; Set96]. The method is based on the embedding of the surface as the zero level of a higher-dimensional scalar smooth level set function, denoted by $\phi$. This level set function satisfies

$$\Gamma(t) = \left\{\mathbf{x} \in \mathbb{R}^3 \mid \phi(\mathbf{x}, t) = 0\right\} \qquad \text{for } t \in [0, t_{\text{end}}].$$

The level set function is not unique, e.g., the zero level of $\tilde{\phi} = c\phi$ also represents the same surface for every $c \neq 0$. A special case is the signed distance function $\phi(\mathbf{x}, t) = d(\mathbf{x}, t)$. We summarize some properties of the level set function in general and the signed distance function in particular. Following [DE13, Lemma 2.8.], there exists a tubular neighbor-

hood $\mathcal{N}_\Gamma^\delta$ of $\Gamma(t)$ with some $\delta > 0$ independent of $t$ such that $d \in C^m(\mathcal{N}_\Gamma^\delta(t))$ for some $m \geq 1$ depending on the smoothness of $\mathcal{S}$. We refer to (2.2.9) for the definition of $\mathcal{N}_\Gamma^\delta$. The signed distance function fulfills the Eikonal equation given in the following lemma. A proof is given in [AD00].

**Lemma 2.6.1**

The signed distance function $d$ satisfies the Eikonal equation

$$\|\nabla d\| = 1 \quad \text{on } \mathcal{N}_\Gamma^\delta.$$

For fixed $t \in [0, t_{\text{end}}]$, we denote $\Gamma = \Gamma(t)$ and assume that $\phi$ is a smooth level set function but not necessarily a signed distance function. However, we assume that it is *close to a signed distance function* in the sense that there exists a constant $c > 0$ with $|\nabla\phi| \geq c$ in some neighborhood of $\mathcal{S}$.

**Lemma 2.6.2**

The normal $\mathbf{n}$ of the surface $\Gamma$ can be represented in terms of a level set function $\phi$ and the signed distance function $d$ as

$$\mathbf{n} = \frac{\nabla\phi}{\|\nabla\phi\|} \quad \text{and} \quad \mathbf{n} = \nabla d \quad \text{on } \Gamma.$$

### 2.6.1 The level set function of an evolving surface

In this section, we derive a transport equation for the level set function $\phi$ of an evolving surface. The surface $\Gamma(t)$ is advected by the material velocity $\mathbf{u}$. We derive a partial differential equation describing the evolution of the level set function which describes the evolution of the surface. The derivation of the equation is based on the trajectories $\mathbf{X}(\mathbf{z}, t)$ defined in (2.1.1). We assume the level set value of each particle $\mathbf{z}$ to remain constant for all $t$, i.e.,

$$\phi(\mathbf{X}(\mathbf{z}, t), t) = \phi(\mathbf{X}(\mathbf{z}, 0), 0) = \phi(\mathbf{z}, 0) = \text{const.} \tag{2.6.1}$$

That means that the values and especially the zero level of $\phi$ are transported with the velocity $\mathbf{u}$ of the density flow. By differentiating equation (2.6.1), we obtain

$$\begin{aligned} 0 &= \frac{\mathrm{d}}{\mathrm{d}t}\phi(\mathbf{X}(\mathbf{z}, t), t) = \frac{\partial}{\partial t}\phi(\mathbf{X}(\mathbf{z}, t), t) + \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{z}, t) \cdot \nabla\phi(\mathbf{X}(\mathbf{z}, t), t) \\ &\overset{(2.1.1)}{=} \frac{\partial}{\partial t}\phi(\mathbf{X}(\mathbf{z}, t), t) + \mathbf{u}(\mathbf{X}(\mathbf{z}, t), t) \cdot \nabla\phi(\mathbf{X}(\mathbf{z}, t), t). \end{aligned}$$

Since this equation holds for all $t$ and each particle $\mathbf{z}$ we obtain the level set equation

$$\frac{\partial}{\partial t}\phi + \nabla\phi \cdot \mathbf{u} = 0 \quad \text{on } \Gamma(t), \tag{2.6.2}$$

which is a time-dependent advection equation and thus, a hyperbolic PDE.

In order to derive a well-posed formulation of the level set equation, we extend the equation to a bounded background domain $\Omega \subset \mathbb{R}^3$ with Lipschitz boundary $\partial\Omega$ such that $\Gamma(t) \subset \Omega$ holds for all $t \in [0, t_{\text{end}}]$. We assume that the material velocity field $\mathbf{u} \in C(\Omega)$ is defined on $\Omega$ (not only on $\Gamma(t)$) and bounded.

Following [Ago98], boundary conditions have to be prescribed on the inflow boundary of $\Omega$. The inflow boundary $\partial\Omega_{\text{in}} \subset \partial\Omega$ is the part of the boundary where the material velocity $\mathbf{u}$ points into $\Omega$. That means that on the inflow boundary, information is transported into the domain. The inflow boundary is defined as

$$\partial\Omega_{\text{in}}(t) := \left\{ \mathbf{x} \in \partial\Omega \ : \ \mathbf{u}(\mathbf{x}, t) \cdot \mathbf{n}_\Omega(\mathbf{x}) < 0 \right\},$$

where $\mathbf{n}_\Omega$ denotes the outward pointing normal to $\partial\Omega$. As $\mathbf{u}$ changes in time, it is possible, that $\partial\Omega_{\text{in}}(t)$ is time dependent even if $\Omega$ is independent of time. Following [Loc13, Theorem 3.4], the following strong formulation has a unique solution if $\mathbf{u}$ is Lipschitz-continuous in $\Omega$ with respect to $\mathbf{x}$ and continuous in $[0, t_{\text{end}}]$ with respect to $t$.

**Problem 2.6.3**

Let $\phi_0$ be a given initial condition defined in $\Omega$, $\phi_D(t)$ a boundary condition defined on $\partial\Omega_{\text{in}}(t)$ for every $t \in [0, t_{\text{end}}]$ and $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$ a given velocity field. The level set transport problem can be formulated as follows: Find $\phi : \Omega \times [0, t_{\text{end}}] \to \mathbb{R}$ differentiable in space and time, such that

$$\begin{cases} \dfrac{\partial}{\partial t}\phi + \mathbf{u} \cdot \nabla\phi = 0 & \text{in } \Omega \times [0, t_{\text{end}}], \\ \phi(\mathbf{x}, 0) = \phi_0(\mathbf{x}) & \text{in } \Omega, \\ \phi(\mathbf{x}, t) = \phi_D(\mathbf{x}, t) & \text{on } \partial\Omega_{\text{in}}(t) \times [0, t_{\text{end}}]. \end{cases}$$

There exists a well-posed variational formulation of Problem 2.6.3, cf. [Loc13]. A modified version of the transport equation was analysed in [**BotheFrickeMaricRoismanVuckovic2023**]. In Chapter 6, we introduce a method to solve the level set equation on a narrow band around $\Gamma(t)$. Thus, we construct a method that approximates the zero level of the level set function with a material velocity that is given in a close neighborhood of the surface. In Chapter 7, we introduce an extension of the material velocity field from the surface to a narrow band around $\Gamma(t)$.

# Chapter 3

# Surface Navier–Stokes Equations

This chapter is based on the article [BRS22].

## 3.1 Introduction

In this chapter, we focus on derivation of Navier–Stokes equations posed on evolving surfaces. In the past decades, several derivations have been presented in the literature, e.g., [AD09; EM70; EHZ07; GKL17; JOR18; MT01; Miu17; NRV19], which differ in the mathematical formulations, the physical principles used in the modeling approach and in the coordinate systems in which the resulting equations are represented. To provide a comprehensive overview of the modeling methodologies, we systematically review five distinct approaches from the literature for deriving the surface Navier–Stokes equations. We can classify the approaches by the physical principles as follows: The conservation of mass and momentum can be used as basic physical principles. One can either directly use the conservation laws on the surface by conserving *surface* mass and momentum, or one can use the conservation laws for *volume* mass and momentum quantities in a thin tubular neighborhood of the surface. In the latter case, a thin film limit procedure is applied to derive the surface Navier–Stokes equations. In [EHZ07; JOR18], the surface mass and momentum conservation laws are used, whereas, in [Miu17; NRV19], the laws of volume mass and momentum for a material volume (a neighborhood of the surface) are used. Another physical principle used in the literature is the variational energy principle, which is based on energy minimization principles, cf. [GKL17].

Furthermore, the approaches differ in the coordinate systems used to represent the resulting flow equations. In some approaches, e.g., [EHZ07; NRV19], local coordinate systems (curvilinear coordinates) are used. These coordinates are also used in [AD09; NRV20]. In [JOR18; GKL17; Miu17], the standard Euclidean basis of $\mathbb{R}^3$, in which the evolving surface is embedded, is used. Both coordinate systems are introduced and analysed in Section 2.2.1.

The different coordinate systems lead to different representations of surface differential operators such as a surface gradient or a surface divergence, and one has to be careful when comparing equations formulated in different coordinate systems. Both the local curvilinear and the global Cartesian coordinate system have attractive properties,

advantages and disadvantages. The local coordinate system can be very useful for modeling more complex fluid properties, e.g., in certain classes of fluid membranes [EHZ07; TMA19] or in flows of liquid crystals [NRV19; NRV20]. The representation in global Cartesian coordinates is very convenient for the development of numerical simulation methods for these flow equations since a local basis is often hard to realize numerically. Furthermore, in [EHZ07] no surface differential operators are used in the derivation. This leads to a different looking system of equations of partial derivatives, and comparison of the resulting equations is not straightforward.

The equations also differ in the known and unknown quantities. In [EHZ07; JOR18], the unknowns are the evolving surface $\Gamma(t)$, the surface velocity $\mathbf{u}$ (tangential and normal), and the surface pressure $p$. In [GKL17; Miu17], the evolution of the surface $\Gamma(t)$ is prescribed by the known surface velocity $V_\Gamma$, and the unknowns of the systems are the velocity $\mathbf{u}$, with the prescribed normal component matching the surface velocity ($\mathbf{u} \cdot \mathbf{n} = V_\Gamma$), and the surface pressure $p$. Finally, in [NRV19] the normal velocity of the surface $V_\Gamma$ is known, and the unknowns are the tangential velocity $\mathbf{u}_T$ and the surface pressure $p$.

The goal of this chapter is to provide a unified framework for comparing these derivations and to show that some, but not all of them lead to the same resulting models. We also present a splitting approach in tangential and normal components of the velocity which shows that all five derivations yield the same tangential surface Navier–Stokes equations. At the end of this chapter, we present the formulation of the surface Navier–Stokes equations, for which we will develop tailor-made discretization methods in the subsequent chapters of the thesis.

The remainder of this chapter is organized in two sections. The five derivations of surface Navier–Stokes equations, known from the literature, are summarized in Section 3.2. In Section 3.3, we discuss and compare these equations.

## 3.2 Summary of derivations of surface Navier–Stokes equations

In this section, we outline five different derivations of surface Navier–Stokes equations known from the literature [EHZ07; JOR18; GKL17; Miu17; NRV19], which use different physical principles and representations in different coordinate systems. In the five subsections below we present, in a unified framework, the following derivations:

(1) In [EHZ07], the conservation laws of *surface* mass and momentum quantities are used. Surface Navier–Stokes equations in *curvilinear coordinates* are derived.

(2) In [JOR18], also conservation laws of *surface* mass and momentum quantities are used. Surface Navier–Stokes equations in *Cartesian coordinates* in $\mathbb{R}^3$ are derived.

(3) In [GKL17], the surface mass conservation law is used and combined with a *variational energy principle*. The geometric evolution is prescribed, and the resulting equations are derived in *Cartesian coordinates* in $\mathbb{R}^3$.

(4) In [Miu17], the conservation laws of *volume* mass and momentum quantities are used in a thin tubular neighborhood of the (evolving) surface with known normal velocity. Combined with a thin film limit procedure, surface Navier–Stokes equations are derived in *Cartesian coordinates*.

(5) In [NRV19], the same physical principles of *volume* mass and momentum conservation in a thin tubular neighborhood as in [Miu17] are used. The resulting volume Navier–Stokes equations are represented in a thin film *curvilinear* local coordinate system. A thin film limit procedure is applied to derive *tangential* surface Navier–Stokes equations in curvilinear coordinates.

Using conservation of momentum to model Navier–Stokes equations (on the surface or in the volume) as in the approaches (1), (2), (4), and (5) is a common approach in fluid mechanics. It is a statement of Newton's second law of motion, which specifies that the rate of change of momentum of a body is equal to the net force acting on the body. Therefore, one needs an ansatz for the net force and thus the stress tensor. In the approaches (4) and (5), one uses the standard Newtonian ansatz, and in (1) and (2), the Boussinesq-Scriven ansatz is used. In (3), an ansatz for the viscous surface dissipation energy is used.

Furthermore, the approaches differ in the unknowns of the derived systems. In ansatz (1) and (2), the unknowns are the evolving surface $\Gamma(t)$, the surface material velocity $\mathbf{u}$ (tangential and normal component), and the surface pressure $p$. In ansatz (3), (4) and (5), the geometric evolution of the surface $\Gamma(t)$ is prescribed. The unknowns are the pressure $p$ and the velocity $\mathbf{u}$ in ansatz (3) and (4), whereas the tangential velocity $\mathbf{u}_T$ and the surface pressure $p$ are unknown in ansatz (5). In ansatz (4), an additional unknown is the auxiliary variable $p^1$.

We summarize the key differences of the derivations in the following table.

| | **Literature** | **Physical Principle** | **Coordinates** | **Unknowns** |
|---|---|---|---|---|
| (1) | [EHZ07] | Surface conservation laws | Curvilinear | $\Gamma(t)$, $\mathbf{u}$, $p$ |
| (2) | [JOR18] | Surface conservation laws | Cartesian | $\Gamma(t)$, $\mathbf{u}$, $p$ |
| (3) | [GKL17] | Variational energy principle | Cartesian | $\mathbf{u}$, $p$, $p^1$ |
| (4) | [Miu17] | Thin film limit | Cartesian | $\mathbf{u}$, $p$ |
| (5) | [NNV19] | Thin film limit | Curvilinear | $\mathbf{u}_T$, $p$ |

Table 3.1: Summary of derivation approaches for surface Navier–Stokes equations.

Below we outline only the key ideas of the derivations and refer to the corresponding papers for more details.

### 3.2.1 Surface mass and momentum conservation in curvilinear coordinates

In this section, we summarize the derivation of surface Navier–Stokes equations presented in [EHZ07]. In that paper, the resulting surface Navier–Stokes equations are formulated in tensor calculus *without* using surface differential operators like $\nabla_\Gamma$ and $\mathrm{div}_\Gamma$. To compare the resulting equations with those obtained in the other approaches, we rewrite these using the differential operators represented in local bases introduced in Section 2.2.2. The derivation is based on the conservation laws of surface mass and momentum.

In [EHZ07], the surface is modeled as a layer of incompressible fluid of lipid molecules which results in the presence of a director field. This gives rise to an additional term in the momentum balance equation. In this summary, we do not take the director field into account, and thus this additional term vanishes.

Firstly, we assume the material surface $\Gamma(t)$ to be inextensible. For a material subdomain $\gamma(t) \subset \Gamma(t)$ the inextensibility reads

$$\frac{\mathrm{d}}{\mathrm{dt}} \int_{\gamma(t)} 1 \,\mathrm{d}s = 0.$$

Applying the Leibniz integral rule and the arbitrariness of $\gamma(t)$ yield

$$\hat{E}^{\alpha}_{\Gamma,\alpha} = 0, \tag{3.2.1}$$

with $\hat{E}_{\Gamma,\alpha\beta} := \frac{1}{2}\frac{\mathrm{d}}{\mathrm{dt}} g_{\alpha\beta}$ the rate of strain tensor, cf. [EHZ07, equation (17)].

Next, we apply the conservation of surface mass. Let $\rho$ denote the surface mass density. Conservation of mass, the transport theorem 2.4.2 and the inextensibility ($\hat{E}^{\alpha}_{\Gamma,\alpha} = 0$) lead to

$$0 = \frac{\mathrm{d}}{\mathrm{dt}} \int_{\gamma(t)} \rho \,\mathrm{d}s = \int_{\gamma(t)} \dot{\rho} + \rho \hat{E}^{\alpha}_{\Gamma,\alpha} \,\mathrm{d}s = \int_{\gamma(t)} \dot{\rho} \,\mathrm{d}s.$$

Arbitrariness of $\gamma(t)$ and a smoothness assumption on $\rho$ imply $\dot{\rho} = 0$. Hence, assuming $\rho$ is constant on $\Gamma(0)$, it follows that the surface mass density $\rho$ is constant on the evolving surface $\Gamma(t)$.

The second conservation law used in [EHZ07] is the conservation of surface momentum. It reads

$$\frac{\mathrm{d}}{\mathrm{dt}} \int_{\gamma(t)} \rho \mathbf{u} \,\mathrm{d}s = \mathbf{F}(\gamma(t)) \tag{3.2.2}$$

with $\gamma(t)$ an arbitrary material subsurface of $\Gamma$ and a force $\mathbf{F}$ that can be decomposed into external area forces acting on $\gamma(t)$ and internal forces acting on the boundary $\partial\gamma(t)$.

The external forces, consisting of normal and shear stresses, are collected in the force term $\mathbf{f} = \hat{f}^{\alpha}\mathbf{g}_{\alpha} + \hat{f}^{3}\mathbf{n}$. For the internal forces, the Cauchy ansatz is made, i.e., these forces are of the form $\hat{T}^{\alpha\beta}\hat{\nu}_{\alpha}\mathbf{g}_{\beta}$ with $\hat{T}^{\alpha\beta}$ and $\hat{\nu}_{\alpha}$ the curvilinear coordinates of a stress tensor and the in-plane unit normal on $\partial\gamma(t)$, that is normal to $\partial\gamma(t)$ but tangential to $\Gamma(t)$. The total net force on $\gamma(t)$ can be written as

$$\mathbf{F}(\gamma(t)) = \int_{\gamma(t)} \mathbf{f} \,\mathrm{d}s + \int_{\partial\gamma(t)} \hat{T}^{\alpha\beta}\hat{\nu}_{\alpha}\mathbf{g}_{\beta} \,\mathrm{d}\tilde{s}. \tag{3.2.3}$$

As in [EHZ07], we apply the Leibniz rule on the left-hand side of (3.2.2) and Greens formula on the boundary integral of the right-hand side of (3.2.3). Using $\hat{E}^{\alpha}_{\Gamma,\alpha} = 0$, this yields

$$\int_{\gamma(t)} \rho \dot{\mathbf{u}} \,\mathrm{d}s = \int_{\gamma(t)} \mathbf{f} + \hat{T}^{\beta\alpha}_{|\beta}\mathbf{g}_{\alpha} - \hat{T}^{\alpha\beta}\hat{H}_{\alpha\beta}\mathbf{n} \,\mathrm{d}s.$$

Due to the arbitrariness of $\gamma(t)$, we obtain the following system of surface partial differential equations

$$\begin{cases} \rho\left(\dot{\mathbf{u}} \cdot \mathbf{g}^{\alpha}\right) = \hat{f}^{\alpha} + \hat{T}^{\beta\alpha}_{|\beta}, \\ \rho\left(\dot{\mathbf{u}} \cdot \mathbf{n}\right) = \hat{f}^{3} - \hat{T}^{\alpha\beta}\hat{H}_{\alpha\beta}, \end{cases} \quad \text{on } \Gamma(t), \tag{3.2.4}$$

which consists of two equations for the time derivative of the tangential velocity $\dot{\mathbf{u}} \cdot \mathbf{g}^{\alpha}$,

for $\alpha = 1, 2$, and one equation for velocity change in normal direction $\dot{\mathbf{u}} \cdot \mathbf{n}$, cf. [EHZ07, equation (31)]. The stress tensor $\hat{T}^{\alpha\beta}$ is modeled using the Boussinesq-Scriven ansatz can be represented by

$$\hat{T}^{\alpha\beta} = -pg^{\alpha\beta} + 2\mu_0 \hat{E}_\Gamma^{\alpha\beta}. \tag{3.2.5}$$

Here, $p$ denotes the surface pressure, $\mu_0$ the viscosity coefficient, and again $\hat{E}_\Gamma^{\alpha\beta}$ denote the local coordinates of the rate of strain tensor, cf. (2.3.3).

The equations (3.2.1), (3.2.4), and (3.2.5) form the surface Navier–Stokes system derived in [EHZ07] which we summarize here:

$$\begin{cases} \hat{E}^\alpha_{\Gamma,\alpha} = 0, & \\ \rho\left(\dot{\mathbf{u}} \cdot \mathbf{g}^\alpha\right) = \hat{f}^\alpha + \hat{T}^{\beta\alpha}_{|\beta}, & \text{on } \Gamma(t), \\ \rho\left(\dot{\mathbf{u}} \cdot \mathbf{n}\right) = \hat{f}^3 - \hat{T}^{\alpha\beta}\hat{H}_{\alpha\beta}. & \end{cases} \tag{3.2.6}$$

Note that the evolution of the surface $\Gamma(t)$ depends on the unknown velocity $\mathbf{u}$ as described in (2.1.1) and (2.1.2).

To compare this surface Navier–Stokes system (3.2.6), which is formulated in terms of curvilinear (local) coordinates, to equations derived in the sections below, we rewrite these equations using surface differential operators from Definition 2.2.8 and representations of tensor- and vector fields that are independent of the choice of the basis. The first equation in (3.2.6) can be rewritten using the surface divergence operator $\operatorname{div}_\Gamma$ and equation (2.4.1) as

$$\hat{E}^\alpha_{\Gamma,\alpha} = \operatorname{div}_\Gamma \mathbf{u} = 0.$$

From Lemma 2.3.5, we obtain for the rate of strain tensor $\mathbf{E}_\Gamma = \hat{E}_\Gamma^{\alpha\beta}(\mathbf{g}_\alpha \otimes \mathbf{g}_\beta)$ the representation

$$\mathbf{E}_\Gamma = \mathbf{E}_\Gamma(\mathbf{u}) = \frac{1}{2}(\nabla_\Gamma \mathbf{u} + \nabla_\Gamma^T \mathbf{u}),$$

and an easy computation yields for an arbitrary vector $\mathbf{c}$ the equation

$$g^{\alpha\beta}\left(\mathbf{g}_\alpha \otimes \mathbf{g}_\beta\right)\mathbf{c} = g^{\alpha\beta}\left(\mathbf{g}_\beta \cdot \mathbf{c}\right)\mathbf{g}_\alpha = \hat{c}_\beta\left(\mathbf{g}^\alpha \cdot \mathbf{g}^\beta\right)\mathbf{g}_\alpha = \left(\mathbf{g}^\alpha \cdot \mathbf{P}\mathbf{c}\right)\mathbf{g}_\alpha = \hat{c}^\alpha \mathbf{g}_\alpha = \mathbf{P}\mathbf{c},$$

from which we get the representation $g^{\alpha\beta}\left(\mathbf{g}_\alpha \otimes \mathbf{g}_\beta\right) = \mathbf{P}$. Thus, the stress tensor $\hat{T}^{\alpha\beta}\left(\mathbf{g}_\alpha \otimes \mathbf{g}_\beta\right) = \mathbf{T}$ can be represented as an operator by

$$\mathbf{T} = -p\mathbf{P} + 2\mu_0 \mathbf{E}_\Gamma.$$

From Lemma 2.2.10, we get that equation (3.2.4) can be rewritten as

$$\begin{aligned} \rho\dot{\mathbf{u}} &= \rho\left[\left(\dot{\mathbf{u}} \cdot \mathbf{g}^\alpha\right)\mathbf{g}_\alpha + \left(\dot{\mathbf{u}} \cdot \mathbf{n}\right)\mathbf{n}\right] = \left(\hat{f}^\alpha + \hat{T}^{\beta\alpha}_{|\beta}\right)\mathbf{g}_\alpha + \left(\hat{f}^3 - \hat{T}^{\alpha\beta}\hat{H}_{\alpha\beta}\right)\mathbf{g}_3 \\ &= \mathbf{f} + \operatorname{div}_\Gamma \mathbf{T}. \end{aligned}$$

Using this and the identity $\operatorname{div}_\Gamma(p\mathbf{P}) = \nabla_\Gamma p - p\kappa\mathbf{n}$ from Lemma 2.5.5, we obtain the following representation of the surface Navier–Stokes system (3.2.6): For a given initial surface $\Gamma(0)$, viscosity $\mu_0$, force term $\mathbf{f}$ and a constant surface mass density $\rho$, find $\mathbf{u}$, $p$

and $\Gamma(t)$, parameterized by $\mathbf{X}(\mathbf{x},t)$, cf. (2.1.2), such that

$$\begin{cases} \rho\dot{\mathbf{u}} + \nabla_\Gamma p - p\kappa\mathbf{n} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma\mathbf{u} = \mathbf{f} & \text{on } \Gamma(t), \\ \qquad\qquad \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t), \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{x},t) = \mathbf{u}(\mathbf{X}(\mathbf{x},t),t), \quad \mathbf{X}(\mathbf{x},0) = \mathbf{x} \in \Gamma(0). \end{cases} \tag{3.2.7}$$

Note that the sign of the curvature $\kappa$ depends on the definition of the Weingarten mapping $\mathbf{H}$. In some literature, e.g., [BRS22; NRV19], the Weingarten mapping is defined with a minus sign, $\mathbf{H} = -\nabla\mathbf{n}$, which leads to a different sign of the curvature.

In Section 3.3, we will compare this system to the equations derived in the following sections.

### 3.2.2 Surface mass and momentum conservation in Cartesian coordinates

We recall the model derived in [JOR18]. The equations are represented using Cartesian coordinates in $\mathbb{R}^3$ and the used surface differential operators ($\nabla_\Gamma$ and $\operatorname{div}_\Gamma$) are defined in Definition 2.2.11. The material derivative $\dot{\mathbf{u}}$ is defined in Cartesian coordinates as formulated in Lemma 2.3.2.

As in the previous section, the modeling is based on the fundamental laws of surface continuum mechanics (conservation of surface mass and momentum). The unknowns are the evolving surface $\Gamma(t)$, the surface (tangential and normal) velocity $\mathbf{u}$ and the surface pressure $p$. We again emphasize that the normal component of the surface velocity governs the evolution of the surface, cf. (2.1.1), (2.1.3), and (2.1.2). Since the physical principles are the same as in the previous section, the derivation of the equations is similar, and we only give a short overview.

The inextensibility condition reads

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\gamma(t)} 1 \,\mathrm{d}s = 0,$$

where $\gamma(t)$ is an arbitrary material subdomain of $\Gamma(t)$. Using the transport theorem 2.4.2, this implies

$$\operatorname{div}_\Gamma \mathbf{u} = 0. \tag{3.2.8}$$

Again, we assume that the surface mass density $\rho$ is constant on $\Gamma(0)$. The conservation of surface mass

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\gamma(t)} \rho \,\mathrm{d}s = 0$$

yields with the same arguments as in the previous section that $\rho$ remains constant on $\Gamma(t)$.

The conservation of surface momentum is expressed by the equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\gamma(t)} \rho\mathbf{u} \,\mathrm{d}s = \int_{\gamma(t)} \mathbf{f} \,\mathrm{d}s + \int_{\partial\gamma(t)} \mathbf{f}_{\boldsymbol{\nu}} \,\mathrm{d}\tilde{s} \tag{3.2.9}$$

with a contact force term $\mathbf{f}_{\boldsymbol{\nu}}$ on $\partial\gamma(t)$ and an area force term $\mathbf{f}$. The integrals are defined as in Section 2.4. As in [EHZ07] and in the previous section, the authors of [JOR18] use

a Cauchy ansatz and Boussinesq-Scriven ansatz for the contact force term:

$$\mathbf{f}_{\boldsymbol{\nu}} = \mathbf{T}\boldsymbol{\nu}, \quad \mathbf{T} = -p\mathbf{P} + 2\mu_0\mathbf{E}_\Gamma(\mathbf{u}), \quad \mathbf{E}_\Gamma(\mathbf{u}) = \frac{1}{2}(\nabla_\Gamma \mathbf{u} + \nabla_\Gamma^T \mathbf{u}). \tag{3.2.10}$$

From the Divergence Theorem 2.4.3 and the identity $\operatorname{div}_\Gamma(p\mathbf{P}) = \nabla_\Gamma p - p\kappa\mathbf{n}$ from Lemma 2.5.5, we obtain the momentum balance for $\gamma(t)$

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{\gamma(t)} \rho\mathbf{u}\,\mathrm{d}s &= \int_{\gamma(t)} \mathbf{f}\,\mathrm{d}s + \int_{\partial\gamma(t)} \mathbf{T}\boldsymbol{\nu}\,\mathrm{d}\tilde{s} = \int_{\gamma(t)} \mathbf{f} + \operatorname{div}_\Gamma(\mathbf{T})\,\mathrm{d}s \\ &= \int_{\gamma(t)} \mathbf{f} - \nabla_\Gamma p + p\kappa\mathbf{n} + 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u})\,\mathrm{d}s.\end{aligned}$$

Using the Leibniz rule and combining the result with (3.2.8), we obtain the following surface Navier–Stokes system. For a given initial surface $\Gamma(0)$, viscosity $\mu_0$, force term $\mathbf{f}$ and a constant surface mass density $\rho$, find $\mathbf{u}$, $p$ and $\Gamma(t)$, parameterized by $\mathbf{X}(\mathbf{x},t)$, cf. (2.1.2), such that

$$\begin{cases} \rho\dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} = \mathbf{f} & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t), \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{x},t) = \mathbf{u}(\mathbf{X}(\mathbf{x},t),t), \quad \mathbf{X}(\mathbf{x},0) = \mathbf{x} \in \Gamma(0). \end{cases} \tag{3.2.11}$$

Note that the evolution of the surface depends on the unknown velocity $\mathbf{u}$ as described in (2.1.1)-(2.1.2).

### 3.2.3 Energetic variational principle in Cartesian coordinates

The third derivation of surface Navier–Stokes equations is based on a variational energy principle and is presented in [GKL17]. The equations are formulated in Cartesian coordinates in $\mathbb{R}^3$ and the surface differential operators are the same as in Definition 2.2.11. We will briefly summarize the key aspects of the derivation.

Assuming that $\Gamma(t)$ is a closed surface and that *the geometric evolution in terms of the surface normal velocity $V_\Gamma$ is given*, we note that the material velocity $\mathbf{u}$ satisfies

$$\mathbf{u}\cdot\mathbf{n} = V_\Gamma. \tag{3.2.12}$$

Hence, in this framework the evolution of the surface and the surface velocity are known. The unknowns are the material velocity $\mathbf{u}$, with its normal component described by (3.2.12) and the surface pressure $p$.

Following a similar approach to previous sections, we use the inextensibility of the surface and the mass conservation to obtain the following equation

$$\operatorname{div}_\Gamma \mathbf{u} = 0, \tag{3.2.13}$$

where it is established that $\rho$ remains constant on $\mathcal{S}$. Instead of employing a momentum conservation ansatz as presented in prior works [EHZ07; JOR18] (cf. equations (3.2.2) and (3.2.9)), we adopt an energetic variational approach grounded in the Least Action and Minimum Dissipation Principles. The essential steps are outlined below.

The action integral, representing kinetic energy, is defined by

$$A(\mathbf{X}) := \int_0^{t_{\text{end}}} \int_{\Gamma(t)} \frac{1}{2}\rho|\mathbf{u}|^2 \,\mathrm{d}s\,\mathrm{d}t.$$

Here, we observe that $\mathbf{X} = \mathbf{X}(\mathbf{x}, t) \in \Gamma(t)$, with particles following trajectories starting in $\mathbf{x} \in \Gamma(0)$ and the corresponding velocity field $\mathbf{u}(\mathbf{x}, t)$, cf. equation (2.1.1). It is noteworthy that both $\Gamma(t)$ and $\mathbf{u}$ are uniquely determined by these trajectories.

The variation of the action integral with respect to the particle trajectories can be formally expressed as

$$D_{\mathbf{X}} A(\mathbf{X})(\mathbf{w}) = \int_0^{t_{\text{end}}} \int_{\Gamma(t)} F_{\text{cons}} \cdot \mathbf{w} \,\mathrm{d}s\,\mathrm{d}t =: \langle F_{\text{cons}}, \mathbf{w} \rangle$$

for an appropriately chosen class of admissible velocities $\mathbf{w}$. This relation defines a conservative force $F_{\text{cons}}$, as demonstrated in [XSL14]. Under reasonable assumptions, it has been established in [GKL17, Theorem 1.5] that

$$F_{\text{cons}} = -\rho \dot{\mathbf{u}} \tag{3.2.14}$$

holds. Furthermore, another force, namely the dissipation force, arises from variations related to "surface viscosity" energy, modeled by the functional

$$E_{\text{diss}}(\mathbf{u}) = -\int_0^{t_{\text{end}}} \int_{\Gamma(t)} \mu_0 |\mathbf{E}_\Gamma(\mathbf{u})|^2 \,\mathrm{d}s\,\mathrm{d}t,$$

where $\mu_0$ denotes the viscosity coefficient and $\mathbf{E}_\Gamma(\mathbf{u})$ represents the rate of strain tensor as in equation (3.2.10). Variation with respect to the velocity field $\mathbf{u}$ leads to the dissipation force

$$D_{\mathbf{u}} E_{\text{diss}}(\mathbf{u})(\mathbf{w}) = \langle F_{\text{diss}}, \mathbf{w} \rangle,$$

for a "suitable" class of admissible velocities $\mathbf{w}$ (cf. [XSL14]). In [GKL17, Theorem 1.6], the relation

$$F_{\text{diss}} = 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) \tag{3.2.15}$$

is derived.

The Onsager principle (cf. [XSL14; Ons31a; Ons31b]) asserts that the system dynamics arise from a competition between internal energy (here, kinetic energy) and dissipation. In our setting the corresponding equation is formally given by

$$D_{\mathbf{X}} A = -D_{\mathbf{u}} E_{\text{diss}},$$

cf. [GKL17, p. 385]. This relationship implies that for all admissible velocity fields $\mathbf{w}$, we have

$$\langle F_{\text{cons}} + F_{\text{diss}}, \mathbf{w} \rangle = 0.$$

Due to the fact that we consider incompressible surface flows, we restrict to velocity fields $\mathbf{w}$ satisfying $\operatorname{div}_\Gamma \mathbf{w} = 0$. Based on [GKL17, Lemma 2.7], we present the following corollary.

**Corollary 3.2.1**

Let $\mathbf{g} \in C(\mathcal{S})^3$ be such that $\langle \mathbf{g}, \mathbf{w} \rangle = 0$ for all $\mathbf{w} \in C^\infty(\mathcal{S})$ with $\operatorname{div}_\Gamma \mathbf{w} = 0$. Then there exists $p \in C^1(\mathcal{S})$ such that

$$\mathbf{g} = \nabla_\Gamma p - p\kappa\mathbf{n}.$$

By applying this corollary, we derive $F_{\text{cons}} + F_{\text{diss}} = \nabla_\Gamma p - p\kappa\mathbf{n}$ for an appropriate (pressure) function $p$. Combining this with the equations (3.2.12), (3.2.13), (3.2.14), and (3.2.15), we obtain the following surface Navier–Stokes equations (with *given* normal velocity $V_\Gamma$, viscosity $\mu_0$ and a constant surface mass density $\rho$):

$$\begin{cases} \mathbf{u} \cdot \mathbf{n} = V_\Gamma & \text{on } \Gamma(t), \\ \rho \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} = 0 & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t). \end{cases} \tag{3.2.16}$$

There is a subtle issue discussed in detail in [GKL17]. Since the surface normal velocity is not an unknown in this system, (3.2.16) is generally overdetermined. Specifically, there are four unknowns: the velocity components (essentially three unknowns) and the pressure. However, there are five equations including the incompressibility equation and the first equation defining the normal component $u_N$ of the velocity. To form a closed system, these equations can be "projected" to yield a system for the tangential velocity and pressure. This process is further elaborated in Section 3.3.

### 3.2.4 Thin film approach in Cartesian coordinates

An alternative method for deriving surface Navier–Stokes equations, based on a thin film limit procedure, is presented in [Miu17].

Similar to the previous section, it is assumed that $\Gamma(t)$ is a closed surface and that the *geometric evolution in terms of the normal velocity* $V_\Gamma$ *of* $\Gamma(t)$ *is given.*

A thin film domain $\Omega_\varepsilon(t) := \{\mathbf{x} \in \mathbb{R}^3 \mid \operatorname{dist}(\mathbf{x}, \Gamma(t)) < \varepsilon\}$ with a sufficiently small $\varepsilon > 0$ is defined around this surface. This domain evolves with constant thickness such that the surface remains located in its middle. In this evolving thin film, the incompressible three-dimensional Navier–Stokes equations with appropriate boundary conditions on $\partial\Omega_\varepsilon(t)$ are applied. These equations describe mass and momentum conservation within the volume domain $\Omega_\varepsilon(t)$. The limit, as the thickness approaches zero, and the resulting surface equations are then studied. In [Miu17], these limit equations are derived using formal asymptotic expansions (in parameter $\varepsilon$). We outline a few key steps in this derivation and refer to [Miu17] for further details.

The signed distance function to $\Gamma(t)$ is denoted by $d(\cdot, t)$. For sufficiently small $\varepsilon > 0$, the closest point projection of $\mathbf{x} \in \Omega_\varepsilon(t)$ given by $\mathbf{p}(\mathbf{x}, t) = \mathbf{x} - d(\mathbf{x}, t)\mathbf{n}(\mathbf{p}(\mathbf{x}, t), t)$ is well defined. We define the space-time domain $Q_{\varepsilon,I}$ and its boundary $\partial Q_{\varepsilon,I}$ by

$$Q_{\varepsilon,I} := \bigcup_{t \in I} \Omega_\varepsilon(t) \times \{t\}, \qquad \partial Q_{\varepsilon,I} := \bigcup_{t \in I} \partial\Omega_\varepsilon(t) \times \{t\}. \tag{3.2.17}$$

The unit outward normal vector $\mathbf{n}_\varepsilon(\mathbf{x},t)$ and outward normal velocity $V_\varepsilon(\mathbf{x},t)$ on $\partial\Omega_\varepsilon$ are given by

$$\mathbf{n}_\varepsilon(\mathbf{x},t) = \begin{cases} \mathbf{n}(\mathbf{p},t), & \text{if } d(\mathbf{x},t) = \ \ \varepsilon, \\ -\mathbf{n}(\mathbf{p},t), & \text{if } d(\mathbf{x},t) = -\varepsilon, \end{cases} \qquad V_\varepsilon(\mathbf{x},t) = \begin{cases} V_\Gamma(\mathbf{p},t), & \text{if } d(\mathbf{x},t) = \ \ \varepsilon, \\ -V_\Gamma(\mathbf{p},t), & \text{if } d(\mathbf{x},t) = -\varepsilon, \end{cases}$$

with $\mathbf{p} = \mathbf{p}(\mathbf{x},t)$ and $V_\Gamma$ the normal velocity of the surface $\Gamma(t)$.

We consider an incompressible Navier–Stokes system in $Q_{\varepsilon,I}$ with perfect slip Navier boundary conditions:

$$\begin{aligned} \partial_t \mathbf{u}_\varepsilon + (\mathbf{u}_\varepsilon \cdot \nabla)\mathbf{u}_\varepsilon + \nabla p_\varepsilon = \mu_0 \Delta \mathbf{u}_\varepsilon \quad & \text{in } Q_{\varepsilon,I}, \\ \operatorname{div} \mathbf{u}_\varepsilon = 0 \quad & \text{in } Q_{\varepsilon,I}, \\ \mathbf{u}_\varepsilon \cdot \mathbf{n}_\varepsilon = V_\varepsilon \quad & \text{on } \partial Q_{\varepsilon,I}, \\ [\mathbf{E}_3(\mathbf{u}_\varepsilon)\mathbf{n}_\varepsilon]_{\tan} = 0 \quad & \text{on } \partial Q_{\varepsilon,I}, \end{aligned} \tag{3.2.18}$$

where $[\mathbf{a}]_{\tan}$ represents the tangential component to $\partial\Omega_\varepsilon(t)$ of a vector $\mathbf{a} \in \mathbb{R}^3$, and the rate of strain tensor is given by $\mathbf{E}_3(\mathbf{u}) := \frac{1}{2}(\nabla\mathbf{u} + \nabla^T\mathbf{u})$. We use the notation $\mathbf{E}_3$ to distinguish this three-dimensional rate of strain tensor from the surface rate of strain tensor $\mathbf{E}_\Gamma$. The differential operators div and $\nabla$, as referenced in Section 2.2.2, are the standard ones in $\mathbb{R}^3$, and $\partial_t$ is the usual time derivative. The Laplace operator is given by $\Delta\mathbf{u}_\varepsilon := \operatorname{div}\nabla\mathbf{u}_\varepsilon + \nabla\operatorname{div}\mathbf{u}_\varepsilon$. Note that, for a solenoidal velocity field $\mathbf{u}_\varepsilon$, the Laplacian can be represented using the rate of strain tensor by $\Delta\mathbf{u}_\varepsilon = 2\operatorname{div}(\mathbf{E}_3\mathbf{u}_\varepsilon)$. Here, we use the representation $\Delta\mathbf{u}_\varepsilon$ without the rate of strain tensor since this is the one used in [Miu17]. Note that, following the presentation in [Miu17], the density is scaled to $\rho = 1$; however, this scaling is not essential.

The system determines the velocity $\mathbf{u}_\varepsilon$ and pressure $p_\varepsilon$ of a fluid within $Q_{\varepsilon,I}$. To derive equations that define the velocity of the fluid solely on $\Gamma(t)$, consistent with (3.2.18) and dependent only on values of functions on $\Gamma(t)$, formal asymptotic expansions are *assumed*. In particular, it is assumed that, for the solution pair $(\mathbf{u}_\varepsilon, p_\varepsilon)$, there exist vector fields $\mathbf{u}$, $\mathbf{u}^1$, $\mathbf{u}^2$ and scalar functions $p$, $p^1$ such that

$$\mathbf{u}_\varepsilon(\mathbf{x},t) = \mathbf{u}(\mathbf{p},t) + d(\mathbf{x},t)\mathbf{u}^1(\mathbf{p},t) + d(\mathbf{x},t)^2\mathbf{u}^2(\mathbf{p},t) + r(d^3), \tag{3.2.19a}$$

$$p_\varepsilon(\mathbf{x},t) = p(\mathbf{p},t) + d(\mathbf{x},t)p^1(\mathbf{p},t) + r(d^2) \tag{3.2.19b}$$

hold. Here, $r(d^k) = r(d(\mathbf{x},t)^k)$ denotes a higher order term, cf. [Miu17].

The analysis in [Miu17] is not rigorous in the sense that it is not clear whether or under which assumptions such expansions exist. These expansions are substituted in (3.2.18) to derive equations for the zero order terms $\mathbf{u}$ and $p$.

In the following lemma, which is taken from [Miu17, Lemma 2.7], we give a crucial ingredient in deriving surface Navier–Stokes equations from the Navier–Stokes system in the thin film domain $Q_{\varepsilon,I}$. Namely, for scalar and vector-valued functions $q$ and $\mathbf{v}$ on $\mathcal{S}$, we derive the derivatives of the composite functions $q(\mathbf{p}(\mathbf{x},t),t)$ and $\mathbf{v}(\mathbf{p}(\mathbf{x},t),t)$ in terms of the remainder $r(d(\mathbf{x},t)^2)$.

**Lemma 3.2.2**

Let $q$ be a scalar and $\mathbf{v}$ a vector-valued function on $\mathcal{S}$. The derivatives of the composite functions $q(\mathbf{p}(\mathbf{x},t),t)$ and $\mathbf{v}(\mathbf{p}(\mathbf{x},t),t)$ with respect to $\mathbf{x}$ and $t$ are of the form

$$\begin{aligned}\nabla q(\mathbf{p},t) &= \nabla_\Gamma q(\mathbf{p},t) - d\mathbf{H}(\mathbf{p},t)\nabla_\Gamma q(\mathbf{p},t) + r(d^2),\\ \partial_t q(\mathbf{p},t) &= \frac{\mathrm{d}}{\mathrm{d}t} q(\mathbf{p},t) + d\nabla_\Gamma V_\Gamma(\mathbf{p},t)\cdot\nabla_\Gamma q(\mathbf{p},t) + r(d^2),\end{aligned}$$

and

$$\begin{aligned}\nabla \mathbf{v}(\mathbf{p},t) &= (\nabla \mathbf{v}\mathbf{P})(\mathbf{p},t) - d\nabla\mathbf{v}(\mathbf{p},t)\mathbf{H}(\mathbf{p},t) + r(d^2),\\ \partial_t \mathbf{v}(\mathbf{p},t) &= \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{v}(\mathbf{p},t) + d\nabla\mathbf{v}(\mathbf{p},t)\nabla_\Gamma V_\Gamma(\mathbf{p},t) + r(d^2),\end{aligned}$$

with $\mathbf{p} = \mathbf{p}(\mathbf{x},t)$ and $d = d(\mathbf{x},t)$ for $(\mathbf{x},t) \in Q_{\varepsilon,I}$.

Substituting the expansions (3.2.19) in the Navier–Stokes equations, using Lemma 3.2.2, and collecting zero and first order (in $\varepsilon$) terms, the following result is derived in [Miu17, Section 4].

**Theorem 3.2.3**

Let $\mathbf{u}_\varepsilon$ and $p_\varepsilon$ satisfy the Navier–Stokes equations (3.2.18) in the moving domain $\Omega_\varepsilon(t)$ with given normal velocity $V_\Gamma$. Then, the zero order velocity field $\mathbf{u}$ and the zero and first order terms $p$ and $p^1$ satisfy the following equations.

$$\begin{cases} \mathbf{u}\cdot\mathbf{n} = V_\Gamma, \\ \dot{\mathbf{u}} + \nabla_\Gamma p + p^1\mathbf{n} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) = 0, \qquad \text{on } \Gamma(t). \\ \operatorname{div}_\Gamma \mathbf{u} = 0, \end{cases} \tag{3.2.20}$$

Here, the surface rate of strain tensor $\mathbf{E}_\Gamma$ is defined as in (2.3.3).

We briefly examine the result of Theorem 3.2.3. When comparing (3.2.20) with (3.2.16), we observe that instead of $-p\kappa\mathbf{n}$ in (3.2.16), we now have $p^1\mathbf{n}$ and an additional equation, $\mathbf{u}\cdot\mathbf{n} = V_\Gamma$, with a given $V_\Gamma$. Thus, in (3.2.20), we obtain a closed system for the unknowns $\mathbf{u}, p, p^1$. The redundant equation $\mathbf{u}\cdot\mathbf{n} = V_\Gamma$ can be eliminated, allowing us to derive a "projected" system for the tangential velocity and surface pressure $p$. This is further elaborated in Section 3.3.

We indicate why in (3.2.20) the first order term $p^1$, but no first order term $\mathbf{u}^1$ arise. From differentiation of the expansion (3.2.19b), we get for a fixed $t \in I$:

$$\begin{aligned}\nabla p_\varepsilon(\mathbf{x},t) &= \nabla[p(\mathbf{p},t)] + \nabla[d(\mathbf{x},t)p^1(\mathbf{p},t)] + r(d)\\ &= \nabla_\Gamma p^e + \nabla d(\mathbf{x},t)\; p^1(\mathbf{p},t) + d(\mathbf{x},t)\nabla[p^1(\mathbf{p},t)] + r(d)\\ &= \nabla_\Gamma p^e + \mathbf{n}(\mathbf{x},t)p^1(\mathbf{p},t) + r(d),\end{aligned} \tag{3.2.21}$$

with $\mathbf{p} = \mathbf{p}(\mathbf{x},t)$ and $d = d(\mathbf{x},t)$. For $\varepsilon \to 0$, we obtain the relation $\nabla p_\varepsilon = \nabla_\Gamma p + p^1\mathbf{n}$ on

$\Gamma$. Hence, we expect $\nabla_\Gamma p + p^1\mathbf{n}$, and not only $\nabla_\Gamma p$, to occur in (3.2.20). Analogously to (3.2.21) we get, for $\varepsilon \to 0$, expressions for $\nabla \mathbf{u}_\varepsilon$ and $\operatorname{div} \nabla \mathbf{u}_\varepsilon$ that contain $\mathbf{u}^1$ and $\mathbf{u}^2$. The functions $\mathbf{u}^1$ and $\mathbf{u}^2$, however, do *not* occur in the surface Navier–Stokes system (3.2.20). This is based on the relations (cf. [Miu17, Remark 4.5])

$$\mathbf{u}^1 = -\mathbf{P}(\nabla^T \mathbf{u})\mathbf{n}, \quad \mathbf{u}^2 = \frac{1}{2}\left(\mathbf{H}\nabla_\Gamma \mathbf{u} + \nabla_\Gamma \mathbf{u}^1\right)\mathbf{n} = 0.$$

Therefore, in contrast to the pressure (where $p^1$ occurs), the unknowns $\mathbf{u}^1$ and $\mathbf{u}^2$ corresponding to the higher order terms in the $\mathbf{u}_\varepsilon$-expansion can be eliminated and the resulting system of equations (3.2.20) has unknowns $\mathbf{u}$, $p$ and $p^1$.

### 3.2.5 Thin film approach in curvilinear coordinates

Following a similar modeling approach to that described in the previous section, the authors of [NRV19] derive *tangential* surface Navier–Stokes equations through a thin-film limit process. Instead of Cartesian coordinates, they utilize a three-dimensional curvilinear coordinate system specifically adapted for thin films. In the subsequent subsections, we provide an overview of this methodology and present the derived surface Navier–Stokes equations. Furthermore, the authors of [RNV20] claim that, along the same lines, the full (tangential and normal) surface Navier–Stokes equations can be derived in a curvilinear coordinate system. We will discuss this system in Section 3.3.

Consistent with the previous section, it is assumed that the evolution of $\Gamma(t)$ is governed by a *given* normal velocity $V_\Gamma$. Furthermore, an evolving thin film domain $\Omega_\varepsilon(t)$ is considered which retains a constant thickness with the surface situated at its midpoint.

#### Thin film curvilinear coordinate system

We introduce a surface parametrization that is based on the normal velocity field. Specifically, we consider the initial value problems (2.1.1) with the velocity field $\mathbf{u}$ replaced by $V_\Gamma \mathbf{n}$ as in (2.1.3). The corresponding flow map is denoted by $\Phi_t^{\mathbf{n}}$ (see Section 2.1). Instead of using the parametrization in (2.1.4), we adopt $R^{\mathbf{n}}(\boldsymbol{\xi}, t) := \Phi_t^{\mathbf{n}}(\Phi_U(\boldsymbol{\xi}))$. A natural parametrization of the thin film domain $\Omega_\varepsilon(t)$ is then given by

$$R_\varepsilon^{\mathbf{n}}(\boldsymbol{\xi}, \zeta, t) := R^{\mathbf{n}}(\boldsymbol{\xi}, t) + \zeta \mathbf{n}\left(R^{\mathbf{n}}(\boldsymbol{\xi}, t), t\right),$$

with $\boldsymbol{\xi} \in U \subset \mathbb{R}^2$, $\zeta \in (-\varepsilon, \varepsilon)$, and $t \in [0, t_{\text{end}}]$.

Building on this thin film parametrization, we introduce curvilinear coordinates and representations of differential operators in these coordinates, analogous to Section 2.2. Note that, in Section 2.2, we employed a two-dimensional surface parametrization with the first fundamental form denoted by $g_{\alpha\beta}$. In contrast, this section deals with a three-dimensional parametrization of the tubular domain $\Omega_\varepsilon(t)$. Similar to previous sections, Greek letters are used to sum over indices 1 and 2, while Latin letters sum over indices 1, 2, and 3. Partial derivatives with respect to the local coordinates $\boldsymbol{\xi}$ and $\zeta$ are denoted by $\overline{\partial}_i$, i.e., $\overline{\partial}_i = \frac{\partial}{\partial \xi_i}$ for $i = 1, 2$, and $\overline{\partial}_3 = \frac{\partial}{\partial \zeta}$. We define the covariant basis $\mathbf{G}_i = \overline{\partial}_i R_\varepsilon^{\mathbf{n}}$, the corresponding contravariant basis $\mathbf{G}^i$, the metric tensor $G_{ij} := \mathbf{G}_i \cdot \mathbf{G}_j$, and the Christoffel symbols as $\boldsymbol{\Gamma}_{ij}^k := \frac{1}{2} G^{kl}(\overline{\partial}_i G_{jl} + \overline{\partial}_j G_{il} - \overline{\partial}_l G_{ij})$. Differential symbols in curvilinear coordinates $(\boldsymbol{\xi}, \zeta)$ can be defined analogously to Section 2.2.2. Let $q$ be a scalar function, $\mathbf{v}$ a vector field, and $\mathbf{T}$ an operator-valued function defined on $\Omega_\varepsilon(t)$

for a fix $t$. All functions are sufficiently smooth. We define the gradient in terms of the local parametrization $R_\varepsilon^{\mathbf{n}}$. Let $\mathbf{x} \in \Omega_\varepsilon(t)$ be given by $\mathbf{x} = R_\varepsilon^{\mathbf{n}}(\boldsymbol{\xi}, \zeta)$. In the scalar case, we define the corresponding function $\overline{q}$ by $\overline{q}(\boldsymbol{\xi}, \zeta) = q(\mathbf{x})$. Along the same lines $\overline{\mathbf{v}}$ and $\overline{\mathbf{T}}$ are defined. Then, the gradient in curvilinear coordinates is given by $\widehat{\nabla} q := \overline{\partial}_i \overline{q} \mathbf{G}^i$ and $\widehat{\nabla} \mathbf{v} = \overline{\partial}_i \overline{\mathbf{v}} \otimes \mathbf{G}^i$ and the divergence $\widehat{\operatorname{div}}\ \mathbf{v} := \overline{\partial}_i \overline{\mathbf{v}} \cdot \mathbf{G}^i$ and $\widehat{\operatorname{div}}\ \mathbf{T} = (\overline{\partial}_i \overline{\mathbf{T}})^T \mathbf{G}^i$.

Using the fact that these operators do not depend on the choice of the parametrization, cf. [Cia13], one obtains the following relations with differential operators in Euclidean three-dimensional space, for which we used the notation without the $\widehat{\ }$ symbol, cf. Section 2.2.2

$$\nabla q = \widehat{\nabla} q, \quad \nabla \mathbf{v} = \widehat{\nabla} \mathbf{v}, \quad \operatorname{div} \mathbf{v} = \widehat{\operatorname{div}}\ \mathbf{v}, \quad \operatorname{div} \mathbf{T} = \widehat{\operatorname{div}}\big(\mathbf{T}^T\big), \tag{3.2.22}$$

with $\operatorname{div} \mathbf{T} := \operatorname{div}\big(\mathbf{T}\mathbf{e}_i\big)\mathbf{e}_i$ the usual column-wise divergence of a tensor. Below we use the notation without $\widehat{\ }$.

Similar to Definition 2.2.2, we define the curvilinear components of a vector field $\mathbf{v}$ in the narrow band by $\hat{v}_i := \mathbf{v} \cdot \mathbf{G}_i$. The components $\hat{v}^i$, $\hat{T}_{ij}$, $\hat{T}^{ij}$, and $\hat{T}_i{}^j$ for a tensor-valued function $\mathbf{T}$ are defined analogously. Comparable to Theorem 2.2.6, cf. also equations (2.2.6), (2.2.7), and (2.2.8), one can use the curvilinear coordinates to represent the differential operators in terms of local components, e.g.,

$$\nabla \mathbf{v}_{ij} = \hat{v}_{i|j} \left( \mathbf{G}^i \otimes \mathbf{G}^j \right), \quad \operatorname{div} \mathbf{v} = \hat{v}^i{}_{|i}, \quad \operatorname{div} \mathbf{T} = \hat{T}_i{}^j{}_{|j} \mathbf{G}^i,$$

with

$$\begin{aligned} \hat{v}_{i|j} &:= \overline{\partial}_j \hat{v}_i - \boldsymbol{\Gamma}_{ij}^k \hat{v}_k, \quad \hat{v}^j{}_{|i} := \overline{\partial}_i \hat{v}^j + \boldsymbol{\Gamma}_{ki}^j \hat{v}^k, \quad T_i{}^j{}_{|k} := G^{jl} T_{il|k}, \\ T^{ij}{}_{|k} &:= \overline{\partial}_k T^{ij} + T^{lj} \boldsymbol{\Gamma}_{lk}^i + T^{il} \boldsymbol{\Gamma}_{lk}^j, \quad T_{ij|k} := \overline{\partial}_k T_{ij} - T_{lj} \boldsymbol{\Gamma}_{ik}^l - T_{il} \boldsymbol{\Gamma}_{jk}^l. \end{aligned}$$

To derive the limit equation, it is beneficial to relate the three-dimensional metric tensor $G_{ij}$ to an appropriate surface metric on $\Gamma(t)$. For the latter, we use the metric induced by the parametrization $R^{\mathbf{n}}$. With a slight abuse of notation, we employ the same symbols as in Section 2.2, such as $g_{\alpha\beta}$ for the metric tensor induced by $R^{\mathbf{n}}$. The following useful results can be derived for these metric tensors

$$\begin{aligned} G_{\alpha\beta} &= g_{\alpha\beta} + 2\zeta \hat{H}_{\alpha\beta} + \zeta^2 \hat{H}_{\alpha\gamma} \hat{H}_\beta^\gamma, & G_{33} &= 1, & G_{3\alpha} &= G_{\alpha 3} = 0, \\ G^{\alpha\beta} &= g^{\alpha\beta} + \mathcal{O}(\zeta), & G^{33} &= 1, & G^{3\alpha} &= G^{\alpha 3} = 0, \end{aligned} \tag{3.2.23}$$

and for the Christoffel symbols

$$\begin{aligned} \boldsymbol{\Gamma}_{\alpha\beta}^\gamma &= \Gamma_{\alpha\beta}^\gamma + \mathcal{O}(\zeta), & \boldsymbol{\Gamma}_{\alpha 3}^\beta &= \boldsymbol{\Gamma}_{3\alpha}^\beta = \hat{H}_\alpha^\beta + \mathcal{O}(\zeta), \\ \boldsymbol{\Gamma}_{\alpha\beta}^3 &= -\hat{H}_{\alpha\beta} + \mathcal{O}(\zeta), & \boldsymbol{\Gamma}_{i3}^3 &= \boldsymbol{\Gamma}_{3i}^3 = \boldsymbol{\Gamma}_{33}^j = 0, \end{aligned} \tag{3.2.24}$$

cf. [NRV19]. Analogous to in Section 2.3, we define the material derivative, but now with respect to the parametrization $R_\varepsilon^{\mathbf{n}}(\boldsymbol{\xi}, \zeta, t)$:

$$\mathring{\mathbf{v}}(\mathbf{x}, t) := \partial_t \bar{\mathbf{v}}(\boldsymbol{\xi}, \zeta, t) = \partial_t \mathbf{v}(R_\varepsilon^{\mathbf{n}}(\boldsymbol{\xi}, \zeta, t), t), \qquad \mathbf{x} = R_\varepsilon^{\mathbf{n}}(\boldsymbol{\xi}, \zeta, t) \in \Omega_\varepsilon(t). \tag{3.2.25}$$

For the velocity field corresponding to the parametrization $R_\varepsilon^{\mathbf{n}}$, we use the notation

$$\mathbf{w}_R(\mathbf{x}, t) := \frac{\partial}{\partial t} R_\varepsilon^{\mathbf{n}}(\boldsymbol{\xi}, \zeta, t), \quad \mathbf{x} = R_\varepsilon^{\mathbf{n}}(\boldsymbol{\xi}, \zeta, t) \in \Omega_\varepsilon(t). \tag{3.2.26}$$

Using $\frac{\partial}{\partial t} R^{\mathbf{n}}(\boldsymbol{\xi}, t) = (V_\Gamma \mathbf{n})(R^{\mathbf{n}}(\boldsymbol{\xi}, t), t)$, it follows that $\mathbf{w}_R = V_\Gamma \mathbf{n} + \mathcal{O}(\zeta)$ holds. The material derivative can be reformulated in Cartesian coordinates as

$$\mathring{\mathbf{v}}(\mathbf{x}, t) = \partial_t \mathbf{v}(\mathbf{x}, t) + \nabla \mathbf{v}(\mathbf{x}, t) \mathbf{w}_R(\mathbf{x}, t), \quad \mathbf{x} \in \Omega_\varepsilon(t). \tag{3.2.27}$$

Note that, for the partial time derivative, the function $\mathbf{v}$ has to be extended off the domain $\Omega_\varepsilon(t)$, cf. Section 2.3. In the remainder of this section, we identify functions defined on $\Omega_\varepsilon(t)$ (or $\Gamma(t)$) with their extension, such that all partial time derivatives are well defined.

### Navier–Stokes equations in thin film

In [NRV19], the authors develop a surface Ericksen-Leslie model by starting from a simplified local three-dimensional Ericksen-Leslie model (see [NRV19, equations (B1)-(B3)]) within the evolving thin film domain $\Omega_\varepsilon(t)$. We further simplify these equations by setting $\lambda = 0$ in equation (B1) and deleting (B3). The resulting Navier–Stokes equations bear resemblance to those presented in Section 3.2.4. It is important to note, however, that, while Cartesian coordinates are employed in Section 3.2.4, curvilinear thin film coordinates are utilized in this section. The space-time domain is defined as in (3.2.17).

The thin film Navier–Stokes system used in [NRV19] is given by

$$\begin{aligned} \partial_t \bar{\mathbf{u}}_\varepsilon + \nabla^{\mathbf{v}_R} \mathbf{u}_\varepsilon &= -\nabla p_\varepsilon + \mu_0 \Delta \mathbf{u}_\varepsilon && \text{in } Q_{\varepsilon,I}, \\ \operatorname{div} \mathbf{u}_\varepsilon &= 0 && \text{in } Q_{\varepsilon,I}, \\ \mathbf{u}_\varepsilon \cdot \mathbf{n}_\varepsilon &= \pm V_\Gamma && \text{on } \partial Q_{\varepsilon,I}, \\ [\mathbf{E}_3(\mathbf{u}_\varepsilon)\mathbf{n}_\varepsilon]_{\tan} &= 0 && \text{on } \partial Q_{\varepsilon,I}, \end{aligned} \tag{3.2.28}$$

with $V_\Gamma$ the *given* normal velocity of $\Gamma(t)$, $[\cdot]_{\tan}$ the tangential component to $\partial\Omega_\varepsilon(t)$ as in (3.2.18) and $\mathbf{E}_3(\mathbf{u}_\varepsilon) := \frac{1}{2}(\nabla \mathbf{u}_\varepsilon + \nabla^T \mathbf{u}_\varepsilon)$ the rate of strain tensor. In [NRV19], this rate of strain tensor is denoted by the Lie derivative of the metric tensor, $\mathcal{L}_{\mathbf{u}_\varepsilon} \mathbf{G} = \nabla \mathbf{u}_\varepsilon + \nabla^T \mathbf{u}_\varepsilon$. The time derivative $\partial_t \bar{\mathbf{u}}_\varepsilon$ is defined in curvilinear coordinates as in (3.2.25). As in the previous section, we use the Laplacian $\Delta \mathbf{u}_\varepsilon$ instead of a representation in terms of the rate of strain tensor $\mathbf{E}_3$, similar to [NRV19]. The direction $\mathbf{v}$ in the directional derivative $\nabla^{\mathbf{v}_R} \mathbf{u}_\varepsilon := \nabla \mathbf{u}_\varepsilon \mathbf{v}_R$, is the relative fluid velocity defined by $\mathbf{v}_R := \mathbf{u}_\varepsilon - \mathbf{w}_R$, with $\mathbf{w}_R$ as in (3.2.26). Using (3.2.27) and (3.2.22) we obtain

$$\partial_t \bar{\mathbf{u}}_\varepsilon + \nabla^{\mathbf{v}_R} \mathbf{u}_\varepsilon = \partial_t \mathbf{u}_\varepsilon + \nabla \mathbf{u}_\varepsilon \mathbf{w}_R + \nabla \mathbf{u}_\varepsilon (\mathbf{u}_\varepsilon - \mathbf{w}_R) = \partial_t \mathbf{u}_\varepsilon + \nabla \mathbf{u}_\varepsilon \mathbf{u}_\varepsilon,$$

and thus this is the usual material derivative in Cartesian coordinates, in particular the same as in (3.2.18). We conclude that the two volume Navier–Stokes systems (3.2.28) and (3.2.18) are equal.

### Tangential surface Navier–Stokes system

In [NRV19], the limit $\varepsilon \downarrow 0$ is performed. Thus, the narrow band $\Omega_\varepsilon(t)$ shrinks to the surface $\Gamma(t)$. Using the curvilinear coordinate system, a tangential limit system of (3.2.28) is derived. We give an overview over the main ingredients of the derivation.

Below, with some abuse of notation, we write $\mathbf{u}$ instead of $\mathbf{u}_\varepsilon$ for the velocity in the thin film domain $\Omega_\varepsilon(t)$. Thus, different from the notation used in the previous sections, $\mathbf{u}$ now denotes a velocity defined in the volume instead of on the surface. In this section,

we use $\mathbf{u}_T$ to denote a tangential velocity defined only on the surface. Since we will only derive a system for the tangential component of the velocity on the surface, this notation is unambiguous.

The homogeneous Navier boundary condition can be rewritten as

$$\hat{u}_{\alpha|3} + \hat{u}_{3|\alpha} = 0 \text{ on } \partial\Omega_\varepsilon(t).$$

Using this, the covariant components of the rate of strain tensor $\frac{1}{2}\left(\hat{u}_{j|i} + \hat{u}_{i|j}\right)$, Taylor expansions, and the results in (3.2.23) and (3.2.24), the following relationships can be obtained

$$\begin{aligned} \hat{u}_{3|3}\big|_\Gamma &= \mathcal{O}(\varepsilon^2), & \hat{E}_{3_{\alpha 3}}\big|_\Gamma &= \mathcal{O}(\varepsilon^2), \\ \widehat{\bar{\partial}_3 \mathbf{E}_{3\alpha 3}}\big|_\Gamma &= \mathcal{O}(\varepsilon^2), & \hat{E}_{3_{\alpha 3|3}}\big|_\Gamma &= \mathcal{O}(\varepsilon^2), \end{aligned} \tag{3.2.29}$$

with $\mathbf{E}_3 = \mathbf{E}_3(\mathbf{u})$, cf. [NRV19, equations (B9)-(B11), (B13)]. On the surface $\Gamma$ we denote the tangential component of the velocity by $\mathbf{u}_T$, i.e.,

$$\mathbf{u}_T = (\widehat{u_T})_\alpha \mathbf{g}^\alpha = (\hat{u}_\alpha \mathbf{g}^\alpha)|_\Gamma = \mathbf{P}\mathbf{u}|_\Gamma.$$

Following [NRV19, equation (B18)], it holds

$$\hat{u}_{\alpha|\beta}\big|_\Gamma = (\widehat{u_T})_{\alpha|\beta} + u_N \hat{H}_{\alpha\beta}, \tag{3.2.30}$$

which is an equivalent representation of (2.5.1) in covariant components. We aim to derive equations for $\mathbf{u}_T$ and $p = p_\varepsilon|_\Gamma$ on the surface. We first consider the second equation of (3.2.28).

**Lemma 3.2.4**

On $\Gamma$, the tangential velocity $\mathbf{u}_T$ satisfies

$$\operatorname{div}_\Gamma \mathbf{u}_T = -u_N \kappa + \mathcal{O}(\varepsilon^2). \tag{3.2.31}$$

**Proof**

Using (3.2.29) and (3.2.30), the following relation can be derived

$$0 = (\operatorname{div} \mathbf{u})\big|_\Gamma = \operatorname{div}_\Gamma \mathbf{u}_T + u_N \kappa + \mathcal{O}(\varepsilon^2),$$

cf. [NRV19, equation (B22)]. □

We now treat the projection of the material derivative in the first equation of (3.2.28).

**Lemma 3.2.5**

On the surface $\Gamma$ it holds

$$\mathbf{P}\left(\partial_t \bar{\mathbf{u}} + \nabla^{\mathbf{v}_R}\mathbf{u}\right) = \partial_t\big(\widehat{\mathbf{u}}_T^\beta\big)\mathbf{g}_\beta + \nabla_\Gamma^{\mathbf{u}_T}\mathbf{u}_T - u_N \nabla_\Gamma u_N + 2u_N \mathbf{H}\mathbf{u}_T.$$

**Proof**

We start with the representation of the tangential component of the partial time derivative of the velocity field $\mathbf{u}$ on the surface $\Gamma$. The splitting $\mathbf{u} = \hat{u}^\alpha \overline{\partial}_\alpha R_\varepsilon^{\mathbf{n}} + \hat{u}_3 \mathbf{n}_\varepsilon$ and the relation $\partial_t R_\varepsilon^{\mathbf{n}}\big|_\Gamma = u_N \mathbf{n}$ are used to derive

$$\begin{aligned} \partial_t \bar{\mathbf{u}} \cdot \mathbf{g}_\alpha\big|_\Gamma &= g_{\alpha\beta} \partial_t \left( \bar{\mathbf{u}}_T \cdot \mathbf{g}^\beta \right) - u_N \left( \overline{\partial}_\alpha u_N - \hat{H}_{\alpha\beta} (\widehat{u_T})^\beta \right) \\ &= g_{\alpha\beta} \partial_t \left( \bar{\mathbf{u}}_T \cdot \mathbf{g}^\beta \right) - \left( u_N (\nabla_\Gamma u_N - \mathbf{H} \mathbf{u}_T) \right) \cdot \mathbf{g}_\alpha, \end{aligned} \tag{3.2.32}$$

cf. [NRV19, equation (B24)].

For the tangential component of the directional derivative $\nabla^{\mathbf{v}_R} \mathbf{u}$, we use the identity $\mathbf{v}_R\big|_\Gamma = \mathbf{P}\mathbf{u}\big|_\Gamma$, equation (3.2.30), and the relation

$$\nabla_\Gamma^{\mathbf{u}_T} \mathbf{u}_T = (\nabla_\Gamma \mathbf{u}_T) \mathbf{u}_T = (\widehat{u_T})^\beta (\widehat{u_T})_{\alpha|\beta} \mathbf{g}^\alpha$$

to obtain

$$\begin{aligned} \left[ \nabla^{\mathbf{v}_R} \mathbf{u} \right] \cdot \mathbf{g}_\alpha\big|_\Gamma &= \hat{v}_R^i \hat{u}_{\alpha|i}\big|_\Gamma = \hat{u}^\beta \hat{u}_{\alpha|\beta}\big|_\Gamma \\ &= (\widehat{u_T})^\beta \left( (\widehat{u_T})_{\alpha|\beta} + u_N \hat{H}_{\alpha\beta} \right) \\ &= \left[ \nabla_\Gamma^{\mathbf{u}_T} \mathbf{u}_T + u_N \mathbf{H} \mathbf{u}_T \right] \cdot \mathbf{g}_\alpha. \end{aligned}$$

Combining this with (3.2.32), we receive

$$\begin{aligned} \mathbf{P} \left( \partial_t \bar{\mathbf{u}} + \nabla^{\mathbf{v}_R} \mathbf{u} \right) &= \left[ (\partial_t \bar{\mathbf{u}} + \nabla^{\mathbf{v}_R} \mathbf{u}) \cdot \mathbf{g}_\alpha \right] \mathbf{g}^\alpha \\ &= \partial_t \left( \widehat{\mathbf{u}}_T^\beta \right) \mathbf{g}_\beta - u_N (\nabla_\Gamma u_N - \mathbf{H} \mathbf{u}_T) + \left[ (\nabla^{\mathbf{v}_R} \mathbf{u}) \cdot \mathbf{g}_\alpha \right] \mathbf{g}^\alpha \\ &= \partial_t \left( \widehat{\mathbf{u}}_T^\beta \right) \mathbf{g}_\beta - u_N (\nabla_\Gamma u_N - \mathbf{H} \mathbf{u}_T) + \nabla_\Gamma^{\mathbf{u}_T} \mathbf{u}_T + u_N \mathbf{H} \mathbf{u}_T \\ &= \partial_t \left( \widehat{\mathbf{u}}_T^\beta \right) \mathbf{g}_\beta + \nabla_\Gamma^{\mathbf{u}_T} \mathbf{u}_T - u_N \nabla_\Gamma u_N + 2 u_N \mathbf{H} \mathbf{u}_T, \end{aligned}$$

on $\Gamma$, cf. [NRV19, equation (B26)]. This completes the proof. □

For the pressure term $p = p_\varepsilon\big|_\Gamma$ in (3.2.28), we immediately obtain the following representation.

**Lemma 3.2.6**

It holds

$$\mathbf{P} \nabla p_\varepsilon = \nabla_\Gamma p \quad \text{on } \Gamma.$$

Finally, we consider the projection of the Laplacian in the first equation in (3.2.28).

**Lemma 3.2.7**

The tangential component of the Laplacian of the velocity field $\mathbf{u}$ can be represented on the surface $\Gamma$ by

$$\mathbf{P} \Delta \mathbf{u} = 2 \mathbf{P} \operatorname{div}_\Gamma (\mathbf{E}_\Gamma \mathbf{u}_T + u_N \mathbf{H}) + \mathcal{O}(\varepsilon^2). \tag{3.2.33}$$

**Proof**

Throughout this proof, we use the notations $\mathbf{E}_3 = \mathbf{E}_3(\mathbf{u})$ and $\mathbf{E}_\Gamma = \mathbf{E}_\Gamma(\mathbf{u}_T)$. For a solenoidal vector field, we have

$$\mathbf{P}\Delta\mathbf{u}\big|_\Gamma = (\Delta\mathbf{u}\cdot\mathbf{g}_\alpha)\mathbf{g}^\alpha\big|_\Gamma = 2\,(\operatorname{div}\mathbf{E}_3\mathbf{u}\cdot\mathbf{g}_\alpha)\mathbf{g}^\alpha\big|_\Gamma .$$

Using equation (3.2.29), the following relation can be derived

$$(\operatorname{div}\mathbf{E}_3\mathbf{u})\cdot\mathbf{g}_\alpha\big|_\Gamma = g^{\beta\gamma}\left(\widehat{E_3}_{\alpha\gamma|\beta} + \hat{H}_{\alpha\beta}\widehat{E_3}_{\gamma 3}\right)\Big|_\Gamma + \mathcal{O}(\varepsilon^2) = g^{\beta\gamma}\,\widehat{E_3}_{\alpha\gamma|\beta}\big|_\Gamma + \mathcal{O}(\varepsilon^2),$$

cf. [NRV19, equation (B17)]. Using $\widehat{E_\Gamma}_{\alpha\gamma} = \frac{1}{2}\left((\widehat{u_T})_{\alpha|\gamma} + (\widehat{u_T})_{\gamma|\alpha}\right)$, we get

$$\widehat{E_3}_{\alpha\gamma}\big|_\Gamma = \frac{1}{2}(\hat{u}_{\alpha|\gamma} + \hat{u}_{\gamma|\alpha})\Big|_\Gamma = \frac{1}{2}\left(\widehat{u_T}_{\alpha|\gamma} + \widehat{u_T}_{\gamma|\alpha}\right) + u_N\hat{H}_{\alpha\gamma} = \widehat{E_\Gamma}_{\alpha\gamma} + u_N\hat{H}_{\alpha\gamma}.$$

Combining these results and using Lemma 2.2.10, we obtain

$$\begin{aligned}
\mathbf{P}\Delta\mathbf{u}\big|_\Gamma &= 2\,(\operatorname{div}\mathbf{E}_3\cdot\mathbf{g}_\alpha)\,\mathbf{g}^\alpha\big|_\Gamma \\
&= 2g^{\beta\gamma}\widehat{E_3}_{\alpha\gamma|\beta}\mathbf{g}^\alpha\Big|_\Gamma + \mathcal{O}(\varepsilon^2) \\
&= 2g^{\beta\gamma}\left(\widehat{E_\Gamma}_{\alpha\gamma|\beta} + u_N\hat{H}_{\alpha\gamma|\beta}\right)\mathbf{g}^\alpha + \mathcal{O}(\varepsilon^2) \\
&= 2\left(\widehat{E_\Gamma}^{\beta\mu}_{\ \ |\beta} + u_N\hat{H}^{\beta\mu}_{\ \ |\beta}\right)g_{\alpha\mu}\mathbf{g}^\alpha + \mathcal{O}(\varepsilon^2) \\
&= 2\left(\widehat{E_\Gamma}^{\beta\mu}_{\ \ |\beta} + u_N\hat{H}^{\beta\mu}_{\ \ |\beta}\right)\mathbf{g}_\mu + \mathcal{O}(\varepsilon^2) \\
&= 2\mathbf{P}\operatorname{div}_\Gamma(\mathbf{E}_\Gamma + u_N\mathbf{H}) + \mathcal{O}(\varepsilon^2),
\end{aligned}$$

which completes the proof. □

**Remark 3.2.8**

We observe that (3.2.33) appears to differ from the first equation of (B21) in [NRV19]. This discrepancy arises due to the use of different definitions for the surface divergence operators when applied to operator-valued functions. Let $\widetilde{\operatorname{div}}_\Gamma$ denote the surface divergence operator employed in [NRV19]. For an operator-valued function $\mathbf{T}$, the relations $\widetilde{\operatorname{div}}_\Gamma\mathbf{T} = \mathbf{P}\operatorname{div}_\Gamma\mathbf{T}$ holds. Using this and $2\mathbf{E}_\Gamma(\mathbf{u}_T) = \nabla_\Gamma\mathbf{u}_T + \nabla_\Gamma^T\mathbf{u}_T$, it follows that the first identity in [NRV19, equation (B21)] and equation (3.2.33) coincide. ◇

Combining the results from Lemmas 3.2.5, 3.2.6, and 3.2.31, equation (3.2.33), and performing the thin film limit $\varepsilon \to 0$, we obtain the tangential Navier–Stokes equations on the surface $\Gamma(t)$ in local coordinates

$$\begin{cases} \partial_t\big(\widehat{\mathbf{u}}_T^\beta\big)\mathbf{g}_\beta + \nabla_\Gamma^{\mathbf{u}_T}\mathbf{u}_T - u_N\nabla_\Gamma u_N + 2u_N\mathbf{H}\mathbf{u}_T = -\nabla_\Gamma p + 2\mu_0\mathbf{P}\operatorname{div}_\Gamma\left(\mathbf{E}_\Gamma(\mathbf{u}_T) + u_N\mathbf{H}\right), \\ \operatorname{div}_\Gamma\mathbf{u}_T = -u_N\kappa, \end{cases} \tag{3.2.34}$$

cf. [NRV19, equation (B27)-(B28)].

We rewrite the time derivative to get a closed formulation independent of the choice of the parametrization. Therefore, we use the following relation.

**Lemma 3.2.9**

On the surface $\Gamma$, the tangential part of the time derivative of the velocity field $\mathbf{u}_T$ can be represented as

$$\mathbf{P}\partial_t \mathbf{u}_T = \partial_t(\widehat{u_T}^\beta)\mathbf{g}_\beta + u_N \mathbf{H}\mathbf{u}_T.$$

**Proof**

On the surface $\Gamma(t)$, we have the following relations

$$\begin{aligned}\partial_t \mathbf{u}_T &= \partial_t\left((\mathbf{u}_T\cdot\mathbf{g}^\beta)\mathbf{g}_\beta\right) = \partial_t(\widehat{u_T}^\beta)\mathbf{g}_\beta + \widehat{u_T}^\beta \partial_t \mathbf{g}_\beta = \partial_t(\widehat{u_T}^\beta)\mathbf{g}_\beta + \widehat{u_T}^\beta \partial_t \overline{\partial}_\beta R^{\mathbf{n}} \\ &= \partial_t(\widehat{u_T}^\beta)\mathbf{g}_\beta + \widehat{u_T}^\beta\ \overline{\partial}_\beta \partial_t R^{\mathbf{n}} = \partial_t(\widehat{u_T}^\beta)\mathbf{g}_\beta + \widehat{u_T}^\beta\ \overline{\partial}_\beta\left(u_N \mathbf{n}\right)\end{aligned} \tag{3.2.35}$$

and

$$\hat{H}_{\alpha\beta}\widehat{u_T}^\beta \mathbf{g}^\alpha = \hat{H}_{\alpha\beta}\left(\mathbf{u}_T\cdot\mathbf{g}^\beta\right)\mathbf{g}^\alpha = \hat{H}_{\alpha\beta}\left(\mathbf{g}^\alpha\otimes\mathbf{g}^\beta\right)\mathbf{u}_T = \mathbf{H}\mathbf{u}_T. \tag{3.2.36}$$

Combining these results, we obtain

$$\begin{aligned}\mathbf{P}\partial_t\mathbf{u}_T = (\partial_t\mathbf{u}_T\cdot\mathbf{g}_\alpha)\mathbf{g}^\alpha &\overset{(3.2.35)}{=} \left[\left(\partial_t(\widehat{u_T}^\beta)\mathbf{g}_\beta + \widehat{u_T}^\beta\ \overline{\partial}_\beta\left(u_N\mathbf{n}\right)\right)\cdot\mathbf{g}_\alpha\right]\mathbf{g}^\alpha \\ &= \partial_t(\widehat{u_T}^\beta)g_{\alpha\beta}\mathbf{g}^\alpha + \widehat{u_T}^\beta\Big[\left(\overline{\partial}_\beta u_N\right)\underbrace{(\mathbf{n}\cdot\mathbf{g}_\alpha)}_{=0} + u_N\underbrace{\left(\overline{\partial}_\beta\mathbf{n}\cdot\mathbf{g}_\alpha\right)}_{=\hat{H}_{\alpha\beta}}\Big]\mathbf{g}^\alpha \\ &= \partial_t(\widehat{u_T}^\beta)g_{\alpha\beta}\mathbf{g}^\alpha + \widehat{u_T}^\beta u_N\hat{H}_{\alpha\beta}\mathbf{g}^\alpha \overset{(3.2.36)}{=} \partial_t(\widehat{u_T}^\beta)\mathbf{g}_\beta + u_N\mathbf{H}\mathbf{u}_T,\end{aligned}$$

which completes the proof. □

Hence, the tangential surface Navier–Stokes equations (3.2.34) posed on the surface $\Gamma(t)$ can be rewritten as

$$\begin{cases}\mathbf{P}\partial_t\bar{\mathbf{u}}_T + \nabla_\Gamma^{\mathbf{u}_T}\mathbf{u}_T - u_N\nabla_\Gamma u_N + u_N\mathbf{H}\mathbf{u}_T = -\nabla_\Gamma p + 2\mu_0\mathbf{P}\,\mathrm{div}_\Gamma(\mathbf{E}_\Gamma(\mathbf{u}_T) + u_N\mathbf{H}), \\ \mathrm{div}_\Gamma\,\mathbf{u}_T = -u_N\kappa,\end{cases} \tag{3.2.37}$$

cf. [NRV19, equation (B30)-(B31)].

**Remark 3.2.10 (Full surface Navier–Stokes system)**

In a further publication (cf. [RNV20]), the authors of [NRV19] claim that, similar to the presentation above, one can also derive a full surface Navier–Stokes system which includes the normal velocity $u_N$ as an additional unknown. This system is given in [RNV20, equations (2.3) - (2.5) and (2.7) - (2.8)]. Furthermore, the authors of [NRV19] claim that the full surface Navier–Stokes system from [RNV20] is equivalent to the one derived in [JOR18] and [EHZ07] that we presented in (3.2.11) and (3.2.16). ◇

## 3.3 Discussion of surface Navier–Stokes equations

In this section we compare the different equations and discuss a directional splitting into an equation for the tangential and one for the normal component of the velocity. For the surface differential operators, we use the notation without $\widehat{\ }$. We recall the resulting

equations from approach (1) and (2), cf. (3.2.7), (3.2.11), with $\rho = 1$:

$$\begin{cases} \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} = \mathbf{f} & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t), \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{x},t) = \mathbf{u}(\mathbf{X}(\mathbf{x},t),t), \quad \mathbf{X}(\mathbf{x},0) = \mathbf{x} \in \Gamma(0). \end{cases} \tag{3.3.1}$$

In the approaches (3), (4), and (5), the evolution of the evolving surface is assumed to be given, and thus the surface parametrization, which is an unknown in (3.3.1), is a known quantity.

The approach (3) yields the system (3.2.16), which reads (with normalized density $\rho = 1$)

$$\begin{cases} \mathbf{u} \cdot \mathbf{n} = V_\Gamma & \text{on } \Gamma(t), \\ \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} = 0 & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t). \end{cases} \tag{3.3.2}$$

Obviously, the second and third equations of (3.3.2) coincide with the first two equations in (3.3.1), with $\mathbf{f} = 0$. Due to the fact that the normal velocity is known, (3.3.2) is an overdetermined system. Below we derive a (closed) projected system for the tangential velocity, cf. (3.3.8).

We recall the system of equations posed on the surface $\Gamma(t)$ resulting from ansatz (4), cf. (3.2.20)

$$\begin{cases} \mathbf{u} \cdot \mathbf{n} = V_\Gamma & \text{on } \Gamma(t), \\ \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p + p^1\mathbf{n} = 0 & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0. & \text{on } \Gamma(t). \end{cases} \tag{3.3.3}$$

Here, an additional unknown scalar function $p^1$ appears that is not present in the other systems.

In approach (5) (only) a *tangential* surface Navier–Stokes system on the surface $\Gamma(t)$ is derived, given in (3.2.37), which we repeat here:

$$\begin{cases} \mathbf{P}\partial_t \bar{\mathbf{u}}_T + \nabla_\Gamma \mathbf{u}_T \mathbf{u}_T = -\nabla_\Gamma p + 2\mu_0 \mathbf{P} \operatorname{div}_\Gamma(\mathbf{E}_\Gamma(\mathbf{u}_T) + V_\Gamma \mathbf{H}) + V_\Gamma(\nabla_\Gamma V_\Gamma - \mathbf{H}\mathbf{u}_T), \\ \operatorname{div}_\Gamma \mathbf{u}_T = -V_\Gamma \kappa. \end{cases} \tag{3.3.4}$$

**Remark 3.3.1**

The systems differ in the known and unknown variables. We focus on the unknowns of each system. In equation (3.3.1), the surface is not predetermined, resulting in both the tangential and normal components of velocity being an unknown quantity of the system. Conversely, in (3.3.2), (3.3.3), and (3.3.4), the evolution of the surface is given. Since the surface velocity has to match the normal component of the material velocity, this leads to the equation $\mathbf{u} \cdot \mathbf{n} = V_\Gamma$ defining the normal component of the material velocity. This equation is present in (3.3.2) and (3.3.3). In these systems, the material velocity $\mathbf{u}$ is the unknown quantity and its normal component is defined by the first equation. This indicates that these systems are overdetermined. In contrast, equation (3.3.4) lacks an equation defining the normal component of velocity; consequently, this

system is classified as a tangential surface Navier–Stokes system where $\mathbf{u}_T$ is the quantity of interest. ◇

In the following, we derive a splitting of the equations based on the splitting of the velocity $\mathbf{u} = \mathbf{u}_T + u_N\mathbf{n}$ for (3.3.1), (3.3.2), and (3.3.3). We gain *coupled* equations for $\mathbf{u}_T$, $p$ ("tangential surface Navier–Stokes") depending on the normal velocity $u_N$ and for $u_N$ (normal velocity) depending on $\mathbf{u}_T$. Firstly, we split the incompressibility equation using (2.5.5) leading to

$$\operatorname{div}_\Gamma \mathbf{u} = 0 \quad \Leftrightarrow \quad \operatorname{div}_\Gamma \mathbf{u}_T = -u_N\kappa.$$

Furthermore, we use the relations

$$\mathbf{P}\dot{\mathbf{u}} = \dot{\mathbf{u}}_T + (\dot{\mathbf{n}}\cdot\mathbf{u}_T)\mathbf{n} + u_N\dot{\mathbf{n}}, \tag{3.3.5}$$

$$\dot{\mathbf{u}}\cdot\mathbf{n} = \dot{u}_N - \mathbf{u}_T\cdot\dot{\mathbf{n}}, \tag{3.3.6}$$

$$\mathbf{n}\cdot\operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) = -\operatorname{tr}(\mathbf{H}\nabla_\Gamma\mathbf{u}_T) - u_N\operatorname{tr}(\mathbf{H}^2), \tag{3.3.7}$$

that are derived in [JOR18, Lemma 2.1, equation (3.9)]. Applying the tangential projection $\mathbf{P}$ to the first equation in (3.3.1) leads to

$$\begin{aligned}
0 &= \mathbf{P}\left(\dot{\mathbf{u}} - 2\mu_0\operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} - \mathbf{f}\right)\\
&\overset{(3.3.5)}{=} \dot{\mathbf{u}}_T + (\dot{\mathbf{n}}\cdot\mathbf{u}_T)\mathbf{n} + u_N\dot{\mathbf{n}} - 2\mu_0\mathbf{P}\operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - \mathbf{P}\mathbf{f}\\
&\overset{(2.5.1)}{=} \dot{\mathbf{u}}_T + (\dot{\mathbf{n}}\cdot\mathbf{u}_T)\mathbf{n} + u_N\dot{\mathbf{n}} - 2\mu_0\mathbf{P}\operatorname{div}_\Gamma\left(\mathbf{E}_\Gamma(\mathbf{u}_T) + u_N\mathbf{H}\right) + \nabla_\Gamma p - \mathbf{f}_T.
\end{aligned}$$

This results in the following equation for the tangential velocity $\mathbf{u}_T$ and the pressure $p$ from system (3.3.1)

$$\dot{\mathbf{u}}_T = \mathbf{f}_T - \nabla_\Gamma p + 2\mu_0\mathbf{P}\operatorname{div}_\Gamma\left(\mathbf{E}_\Gamma(\mathbf{u}_T) + u_N\mathbf{H}\right) - (\dot{\mathbf{n}}\cdot\mathbf{u}_T)\mathbf{n} - u_N\dot{\mathbf{n}}. \tag{3.3.8}$$

To get an equation for the normal velocity $u_N$, we multiply the first equation from (3.3.1) with $\mathbf{n}$ to obtain

$$\begin{aligned}
0 &= \mathbf{n}\cdot\left(\dot{\mathbf{u}} - 2\mu_0\operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} - \mathbf{f}\right)\\
&\overset{(3.3.6)}{=} \dot{u}_N - \mathbf{u}_T\cdot\dot{\mathbf{n}} - 2\mu_0\operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u})\cdot\mathbf{n} - p\kappa - \mathbf{f}\cdot\mathbf{n}\\
&\overset{(3.3.7)}{=} \dot{u}_N - \mathbf{u}_T\cdot\dot{\mathbf{n}} + 2\mu_0\left(\operatorname{tr}(\mathbf{H}\nabla_\Gamma\mathbf{u}_T) + u_N\operatorname{tr}(\mathbf{H}^2)\right) - p\kappa - f_N.
\end{aligned}$$

Thus, for the normal velocity $u_N$, we get the equation

$$\dot{u}_N = f_N - 2\mu_0\big(\operatorname{tr}(\mathbf{H}\nabla_\Gamma\mathbf{u}_T) + u_N\operatorname{tr}(\mathbf{H}^2)\big) + p\kappa + \mathbf{u}_T\cdot\dot{\mathbf{n}}. \tag{3.3.9}$$

We used the splitting $\mathbf{f} = \mathbf{f}_T + f_N\mathbf{n}$. Note that $\dot{\mathbf{u}}_T$ denotes the material derivative (along $\mathbf{u}$) of $\mathbf{u}_T$ and not $(\dot{\mathbf{u}})_T = \mathbf{P}\dot{\mathbf{u}}$; similarly for $\dot{u}_N$. We call the system (3.3.8) *tangential surface Navier–Stokes equations*. Note that in these equations the normal velocity $u_N$ occurs. Obviously, the second equation in system (3.3.2) from ansatz (3) leads to the same splitting with $\mathbf{f} = \mathbf{f}_T = f_N = 0$.

**Remark 3.3.2 (Derivation of tangential Navier–Stokes equations in [GKL17])**
The variational principle used in [GKL17] to derive (3.2.16) also facilitates a formulation of a tangential surface Navier–Stokes system when the set of "admissible" velocities $\mathbf{w}$ in the definitions of the terms $F_{\text{cons}}$ and $F_{\text{diss}}$ is restricted to tangential functions ($\mathbf{P}\mathbf{w} = \mathbf{w}$). This restriction results in tangential force expressions given by $F_{\text{cons}} = -\rho\mathbf{P}\dot{\mathbf{u}}$ and $F_{\text{diss}} = 2\mu_0\mathbf{P}\operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u})$. Consequently, the momentum equation for tangential motion aligns with the first equation presented in (3.3.8), where $\mathbf{f}_T = 0$ holds. ◇

**Remark 3.3.3**
A proof of the relation

$$\dot{\mathbf{n}} = \mathbf{H}\mathbf{u}_T - \nabla_\Gamma u_N \tag{3.3.10}$$

can be found in [JOR18, Lemma 2.2]. Inserting this relation into the normal equation (3.3.9), one obtains that no time derivative ($\frac{\partial}{\partial t}$) is involved in $\dot{\mathbf{n}}$ and thus also no time derivative is involved in the right-hand side of (3.3.9). This indicates that the equation (3.3.9) determines the time dynamics of the normal velocity $u_N(\cdot, t)$, and thus of the geometric evolution of the surface $\Gamma(t)$, whereas the tangential surface Navier–Stokes equations (3.3.8) determine the time dynamics of the tangential velocity $\mathbf{u}_T(\cdot, t)$. ◇

Using the same procedure as above, we split the Navier–Stokes system (3.3.3) from ansatz (4) into tangential and normal components. Firstly, we apply the tangential projection $\mathbf{P}$ to the second equation in (3.3.3). Here, the term $p^1\mathbf{P}\mathbf{n}$ vanishes, and the remaining terms are the same as in the projected version of the first equation in (3.3.1), cf. (3.3.8). It follows that system (3.3.3) results in *the same tangential surface Navier–Stokes equations* as in (3.3.8) (with $\mathbf{f}_T = 0$). Next, we take the scalar product of the second equation in (3.3.3) with the normal $\mathbf{n}$ and get

$$\begin{aligned} 0 &= \mathbf{n}\cdot\left(\dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p + p^1\mathbf{n}\right) \\ &\overset{(3.3.6)}{=} \dot{u}_N - \dot{\mathbf{n}}\cdot\mathbf{u}_T + \nabla_\Gamma p\cdot\mathbf{n} + p^1(\mathbf{n}\cdot\mathbf{n}) - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u})\cdot\mathbf{n} \\ &\overset{(3.3.7)}{=} \dot{u}_N - \dot{\mathbf{n}}\cdot\mathbf{u}_T + p^1 + 2\mu_0\left(\operatorname{tr}(\mathbf{H}\nabla_\Gamma\mathbf{u}_T) + u_N\operatorname{tr}(\mathbf{H}^2)\right). \end{aligned}$$

This leads to the following equation for the normal component $u_N$ of the velocity

$$\dot{u}_N = -2\mu_0\left(\operatorname{tr}(\mathbf{H}\nabla_\Gamma\mathbf{u}_T) + u_N\operatorname{tr}(\mathbf{H}^2)\right) - p^1 + \dot{\mathbf{n}}\cdot\mathbf{u}_T.$$

This equals the normal velocity equation in (3.3.9) (with $f_N = 0$) if and only if for the first order term $p^1$ the relation $p^1 = -p\kappa$ holds. Since the first order pressure term is defined by the assumed Taylor expansion (3.2.19b), it holds $p^1 \neq -p\kappa$ in general.

Finally, we compare the tangential Navier–Stokes equations (3.3.8) and the tangential equations (3.3.4) derived from ansatz (5). Using $\mathbf{n}\cdot\dot{\mathbf{u}}_T = -\dot{\mathbf{n}}\cdot\mathbf{u}_T$, which follows from $\mathbf{n}\cdot\mathbf{u}_T = 0$, we rewrite the left-hand side of (3.3.8) by

$$\dot{\mathbf{u}}_T = \mathbf{P}\dot{\mathbf{u}}_T + (\mathbf{n}\cdot\dot{\mathbf{u}}_T)\mathbf{n} = \mathbf{P}\dot{\mathbf{u}}_T - (\dot{\mathbf{n}}\cdot\mathbf{u}_T)\mathbf{n}.$$

Using this relation, we rewrite the tangential momentum equation (3.3.8), with $\mathbf{f}_T = 0$, as

$$\begin{aligned} \mathbf{P}\dot{\mathbf{u}}_T &= -\nabla_\Gamma p + 2\mu_0\mathbf{P}\operatorname{div}_\Gamma\left(\mathbf{E}_\Gamma(\mathbf{u}_T) + u_N\mathbf{H}\right) - u_N\dot{\mathbf{n}} \\ &\overset{(3.3.10)}{=} -\nabla_\Gamma p + 2\mu_0\mathbf{P}\operatorname{div}_\Gamma\left(\mathbf{E}_\Gamma(\mathbf{u}_T) + u_N\mathbf{H}\right) - u_N\left(\mathbf{H}\mathbf{u}_T - \nabla_\Gamma u_N\right). \end{aligned}$$

The right-hand side of this equation is the same as the right-hand side of (3.3.4) if the normal component of the fluid velocity equals the velocity of the surface ($u_N = V_\Gamma$). For the left-hand side of system (3.3.4), we obtain

$$\begin{aligned}
\mathbf{P}\partial_t \bar{\mathbf{u}}_T + \nabla_\Gamma \mathbf{u}_T \mathbf{u}_T &= \mathbf{P}(\partial_t \bar{\mathbf{u}}_T + \nabla \mathbf{u}_T^e \mathbf{u}_T) \\
&\overset{(3.2.25)}{=} \mathbf{P}(\mathring{\mathbf{u}}_T + \nabla \mathbf{u}_T^e \mathbf{u}_T) \\
&\overset{(3.2.27)}{=} \mathbf{P}(\partial_t \mathbf{u}_T^e + \nabla \mathbf{u}_T^e \mathbf{w}_R + \nabla \mathbf{u}_T^e \mathbf{u}_T) \\
&= \mathbf{P}(\partial_t \mathbf{u}_T^e + u_N \nabla \mathbf{u}_T^e \mathbf{n} + \nabla \mathbf{u}_T^e \mathbf{u}_T) \\
&= \mathbf{P}(\partial_t \mathbf{u}_T^e + \nabla \mathbf{u}_T^e \mathbf{u}) \\
&= \mathbf{P}\mathring{\mathbf{u}}_T^\bullet,
\end{aligned}$$

where in the last equality we used Lemma 2.3.2. Hence, we conclude that the two tangential momentum equations in (3.3.8) and (3.3.4) are the same (for $\mathbf{f}_T = 0$).

In summary, we have demonstrated that all five derivations (1) – (5) yield *identical tangential surface Navier–Stokes equations* as represented in (3.3.8). Furthermore, the derivations from (1) – (3) result in the same expression for the normal velocity, in particular (3.3.9).

**Remark 3.3.4**

In the study of fluid dynamics, particularly concerning the Navier–Stokes equations on evolving surfaces, there exists further literature modeling similar problems besides the presented ones. For a comprehensive discussion on these topics, one may refer to [CCD17] and [JOR18, Section 3.2]. For stationary surfaces, foundational works include, e.g., [EM70; Tay92; MT01]. Furthermore, the dynamics of evolving surfaces have been explored in detail, e.g., in [AD09]. Some, but not all, of the surface Navier–Stokes equations from the literature, that are not mentioned in this chapter, are similar to the ones we presented. ◇

## 3.4 Surface Navier–Stokes equations

At the end of this chapter, we present the formulation of the surface Navier–Stokes equations, that we will use in the subsequent chapters. We will derive a discretization method for a surface Navier–Stokes system based on (3.3.1), since it provides equations for both the tangential and normal components of the velocity and the surface $\Gamma(t)$. In this system, all terms describe some physical quantity, in contrast to $p^1$ in equation 3.3.3 from ansatz (4). Additionally, several papers, cf. e.g., [JOR18; EHZ07; RNV20; PSW20] have already modeled or used this system underlying its significance in recent literature. For simplicity, we consider the density $\rho = 1$.

We revisit a simplified formulation of the surface Navier–Stokes equations, wherein we assume that both the normal velocity $u_N$ and the surface $\Gamma(t)$ are predetermined. This system is referred as the *tangential surface Navier–Stokes equations* and is articulated as follows.

**Problem 3.4.1: Tangential surface Navier–Stokes equations**

For given initial value $\mathbf{u}_{T,0}$, viscosity coefficient $\mu_0$, surface $\Gamma(t)$, normal velocity $u_N$, and force term $\mathbf{f}_T$, find $\mathbf{u}_T$ and $p$ satisfying

$$\begin{cases} \mathbf{P}\dot{\mathbf{u}}_T - 2\mu_0 \mathbf{P} \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}_T) + \nabla_\Gamma p + u_N \mathbf{H}\mathbf{u}_T = \mathbf{f}_t & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u}_T = f_{\text{div}} & \text{on } \Gamma(t), \\ \mathbf{u}_T \cdot \mathbf{n} = 0 & \text{on } \Gamma(t), \end{cases} \tag{3.4.1}$$

with $\mathbf{f}_t = \mathbf{f}_T + 2\mu_0 \mathbf{P} \operatorname{div}_\Gamma (u_N \mathbf{H}) + u_N \nabla_\Gamma u_N$ and $f_{\text{div}} = -u_N \kappa$.

In the following chapters, we will derive a numerical method for the full surface Navier–Stokes system, which includes the normal velocity $u_N$ as an additional unknown. It is given by the following problem.

**Problem 3.4.2: Surface Navier–Stokes equations**

For given initial values $\mathbf{u}^0$, $\Gamma^0$, viscosity coefficient $\mu_0$, and force term $\mathbf{f}$, find $\mathbf{u}$, $p$, and $\mathbf{X}$ satisfying

$$\begin{cases} \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa \mathbf{n} = \mathbf{f} & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t), \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{x},t) = \mathbf{u}(\mathbf{X}(\mathbf{x},t),t), \quad \mathbf{X}(\mathbf{x},0) = \mathbf{x} \in \Gamma(0), \end{cases} \tag{3.4.2}$$

with $\Gamma(t) = \mathbf{X}(\Gamma^0, t)$.

When discretizing the equations, we will describe the surface $\Gamma(t)$ implicitly by a level set function $\phi(\mathbf{x}, t)$, cf. Section 2.6 for further details. Thus, instead of the third equation of Problem 3.4.2 defining the trajectories $\mathbf{X}$ of the surface particles, we will add the level set equation (2.6.2) to the system. We will solve the level set equation in a sufficiently large narrow band around $\Gamma(t)$ with a reasonable extension of the velocity $\mathbf{u}$ off the surface.

The system under consideration (3.4.2) exhibits a strong nonlinearity. The velocity field $\mathbf{u}$ is governed by the first two equations, which depend on the surface $\Gamma(t)$ and its associated geometric properties, namely the normal vector $\mathbf{n}$ and curvature $\kappa$. Conversely, the surface itself is described by the third equation, which incorporates the velocity $\mathbf{u}$. This interdependence establishes a strong nonlinear relationship between the surface $\Gamma(t)$ and the velocity field $\mathbf{u}$, rendering the solution of system (3.4.2) particularly challenging. To date, there exists no well-posedness results for this problem in the literature. Recently, in [ORZ22] a well-posed weak formulation of the tangential surface Navier–Stokes problem 3.4.1 was derived.

In Chapter 5, we will present and analyse a numerical method to solve system (3.4.1), including a stability and error analysis. In Chapter 7, we will introduce a numerical method aimed at discretizing system (3.4.2).

# Chapter 4
# Trace Finite Element Method

In this chapter, we introduce the *Trace finite element method* (TraceFEM) for discretizing partial differential equations (PDEs) posed on surfaces. We give basic definitions and properties of the method and discuss the construction of the finite element spaces for a stationary surface and an Eulerian method based on the TraceFEM for evolving surfaces. The method is based on the idea of using finite element spaces defined on a background domain to solve PDEs on a surface by taking the traces of finite element functions. In the following chapters, we will use a Trace finite element method to solve the tangential surface Navier–Stokes Problem 3.4.1 and the full surface Navier–Stokes Problem 3.4.2 on an evolving surface.

We first take a look at different approaches to solve surface PDEs from the literature. A general overview of such finite element techniques is given in [DE13]. A well-established finite element method, named *SurfaceFEM*, for discretizing partial differential equations on surfaces, as initiated in [Dzi88], involves approximating the surface $\Gamma$ with a piecewise polygonal surface mesh $\Gamma_h$ and employing a finite element space defined on this discrete surface. Thus, a regular surface mesh with its nodes located on the exact surface is used. The SurfaceFEM method and higher order versions of it were further used and analysed, e.g., in [Dem09; Fri18] for scalar- and vector-valued problems. In the recent paper [Reu24], a comprehensive discretization error analysis of the Taylor-Hood-SFEM for a Stokes problem on a time-independent surface is derived. The method has been adapted to solve equations on *evolving surfaces*, cf. [EV15; KKV23; NNV19; RNV20; RV18]. In case of a time-dependent surface, SurfaceFEM is mostly based on a Lagrangian tracking of the surface evolution. This methodology necessitates the use of time-dependent triangulations and finite element spaces, which entails significant data management and programming efforts. In scenarios where a surface experiences strong deformations, topological changes, or is implicitly defined – such as being represented as the zero level of a level set function – numerical methods that utilize a Lagrangian framework face disadvantages and limitations. To circumvent the need for remeshing and to advantage from the implicit definition of a surface as the zero level of a level set function, it was initially proposed in [Ber+01] to extend the partial differential equation from the surface to a region of positive Lebesgue measure in $\mathbb{R}^3$. This leads to a PDE solved in one higher dimension using a mesh that does not need to be aligned with the

surface. Such extension methodologies have been examined, e.g., in [Gre05] for finite difference approximations and in [DE08; OS15] using finite element methods, including applications to PDEs on moving surfaces. Another technique is the so-called *closest point method*, which embeds surface problems within a Cartesian bulk framework; see [PR16; RM08]. This method utilizes the closest point projection to extend the problem from the surface into a small neighborhood surrounding it, where standard Cartesian finite differences are employed for discretizing differential operators. The surface PDE is subsequently embedded and discretized within this neighborhood. Successful implementation necessitates either knowledge of or calculation of the closest point on the surface corresponding to any given point in its vicinity. A similar approach based on extension of the PDE was introduced in [Dec+09]. Methods that embed a surface PDE within a bulk PDE are known to encounter specific challenges, such as requirements for artificial boundary conditions and difficulties associated with geometrical singularities; we refer to [Gre05] for further discussion.

In this thesis, we use a different technique for solving partial differential equations numerically defined on a surface, namely the Trace finite element method (TraceFEM). The key idea of the TraceFEM is to utilize time- and surface independent finite element spaces induced by triangulations of an "outer" domain containing the surface to discretize the PDE on the surface by taking the traces of finite element functions. This motivates the name TraceFEM. The technique is considered to be part of the CutFEM class that has originally been developed as unfitted finite element methods for interface problems, cf. the overview paper [Bur+14]. In TraceFEM one "cuts" of the parts on both sides of the surface and only keeps the part on the surface. In contrast to the approach detailed, e.g., in [Gre05], we do not extend the surface partial differential equation but instead restrict the functions defined on the background mesh to functions defined on the discrete surface approximation. TraceFEM does not utilize a surface mesh (as it is done in SurfaceFEM), but a background mesh that is independent of the surface $\Gamma$ and does not have to be aligned with it. This is an advantage of the TraceFEM for time-dependent surfaces, since no expensive re-meshing is needed. The method is particularly advantageous for problems where the surface is implicitly defined by a level set function. The asymptotic behavior of the number of unknowns in the algebraic systems arising from TraceFEM is analogous to that in methods based on SurfaceFEM. For a detailed comparison between SurfaceFEM and TraceFEM, we refer to [Bra+22], where a Stokes problem on a stationary surface is solved using both finite element methods.

We give basic definitions and properties of TraceFEM mainly taken from the literature. TraceFEM was introduced in [ORG09] for scalar elliptic problems. Optimal order of convergence for the TraceFEM applied on the Laplace-Beltrami problem was shown, e.g., in [Reu14] where the geometric errors induced by the calculation of surface integrals are neglected. In [Leh16], the authors tackle the problem of accurately approximating surface integrals when the surface is described via a higher order level set function. They use a parametric mapping to calculate integrals on curved surfaces by integrating over linear approximations leading to a computationally efficient and flexible method. We will use this technique in the remainder of the thesis. Other integrating techniques on surfaces are discussed in, e.g., [OS16; Say15; SW13]. The parametric TraceFEM was used to discretize the surface Stokes problem in [JOR18] and [BR20]. TraceFEM was also used and analysed for non-stationary surfaces. A space-time TraceFEM method for differential equations on evolving surfaces has been introduced in [SR23] and analysed

in [SR26]. In [HLP23] a space-time TraceFEM for a PDE on an evolving domain has been derived. In [LOX18], an Eulerian TraceFE method for solving a scalar partial differential equation posed on an *evolving* surface is introduced. It does not utilize a space-time approach and is further used and analysed in [WRL21; LL22; ORZ22]. The key ingredient of this method is a stabilization technique applied in a narrow band around the surface. We will use this Eulerian method in the following chapters.

This chapter is organized as follows. In Section 4.1 we introduce the basic definitions and properties of the TraceFEM for solving partial differential equations on a stationary surface. We define the finite element spaces on the background domain and the surface approximation, and introduce a parametric mapping to calculate surface integrals accurately and efficiently. In Section 4.2, we discuss stabilization techniques used in the TraceFEM to ensure stability and solvability of the resulting linear systems. At the end of this chapter, we extend the method to solve partial differential equations on an evolving surface. We introduce a fully Eulerian formulation of the TraceFEM and discuss an extension technique used in the method.

## 4.1 Basics of the TraceFEM for stationary surfaces

We start with the basic definitions and properties of the TraceFEM for solving partial differential equations on a stationary surface $\Gamma$. We assume there exists a polygonal domain $\Omega \subset \mathbb{R}^3$ and an underlying sufficiently smooth level set function $\phi \in C^2(\Omega)$ such that the surface is implicitly described by the zero level

$$\Gamma = \{\mathbf{x} \in \Omega \mid \phi(\mathbf{x}) = 0\} \subset \Omega.$$

This implicit description of the surface is a key assumption of the method. Note that $\phi$ is not necessarily a signed distance function, but it is *close to a signed distance function* in the sense that there exists a constant $c > 0$ and a tubular neighborhood $\mathcal{N}_\Gamma^\delta$ of $\Gamma$ for some $\delta > 0$ such that

$$\|\nabla\phi\| \approx 1 \quad \text{and} \quad \left\|\nabla^2\phi\right\| \leq c \quad \text{in } \mathcal{N}_\Gamma^\delta \tag{4.1.1}$$

hold. This assumption on the level set function to be close to a signed distance function can always be satisfied when the surface is sufficiently smooth, cf. [DE13, Lemma 2.8.] and the neighborhood $\mathcal{N}_\Gamma^\delta$ is sufficiently small. We assume that the level set function $\phi$ is sufficiently smooth such that the geometric quantities on the surface can be represented in terms of $\phi$ and its derivatives. For further details on the level set function and its properties, we refer to Section 2.6.

We denote a family of consistent shape regular triangulations of $\Omega$ into simplices $T$ with mesh size $h_T = \operatorname{diam}(T)$ and $h = \max_{T\in\mathcal{T}_h} h_T$ by $\{\mathcal{T}_h\}_{h>0}$. The grid size $h$ is assumed to be sufficiently small such that $h < \delta$ holds with $\delta$ denoting the size of the neighborhood $\mathcal{N}_\Gamma^\delta$ where the signed distance function $d$ is smooth, cf. (2.2.9).

We define the base finite element space on the background domain $\Omega$ for some $k \in \mathbb{N}$ as

$$V_{h,k} := \left\{ q \in C(\Omega) \mid q_{|T} \in P_k(T) \text{ for all } T \in \mathcal{T}_h \right\}, \tag{4.1.2}$$

where $P_k(T)$ denotes the space of polynomials of degree $k$ on $T$. For vector-valued functions, we define the space $\mathbf{V}_{h,k} = (V_{h,k})^3$. These are standard continuous finite element spaces independent of the surface $\Gamma$ and how it cuts through the mesh. We define

two interpolation operators. The first one is the standard nodal interpolation operator $I_h^k$ into $V_{h,k}$ or $\mathbf{V}_{h,k}$, respectively. The second is an Oswald-type quasi interpolation operator, that uses $L^2$–projections on each element $T \in \mathcal{T}_h$ and averaging over element interfaces. It is denoted by $\Pi_h^k : (L^2(\Omega))^l \to (V_{h,k})^l$ with $l = 1$ or $l = 3$, respectively. See [Leh16, Section 2.5] for a formal definition of the Oswald quasi interpolation operator.

Since the surface $\Gamma$ is an unknown in the surface Navier–Stokes equations, the exact level set function $\phi$ is not known explicitly and only a discrete approximation $\phi_h \in V_{h,k_g}$ with some $k_g \in \mathbb{N}$ of the level set function is available. We assume $\phi_h$ satisfies the following error estimate

$$\|\phi - \phi_h\|_{L^\infty(\mathcal{N}_\Gamma^\delta)} + h\|\nabla\phi - \nabla\phi_h\|_{L^\infty(\mathcal{N}_\Gamma^\delta)} \lesssim h^{k_g+1}. \tag{4.1.3}$$

Here and in the remainder, we use the notation $a \lesssim b$ for the inequality $a \leq cb$ with a generic constant $c > 0$ that is independent of the mesh size $h$ and of the position of the surface $\Gamma$ in the background mesh. The notations $a \sim b$ and $a \gtrsim b$ are defined along the same lines. Note that the error estimate (4.1.3) is satisfied both for the nodal interpolation $\phi_h = I_h^{k_g}\phi$ and the Oswald quasi interpolation $\phi_h = \Pi_h^{k_g}\phi$ when the exact level set function $\phi$ is given, cf. [Leh16]. The zero level of $\phi_h$ is given by

$$\tilde{\Gamma}_h := \left\{\mathbf{x} \in \Omega \mid \phi_h(\mathbf{x}) = 0\right\}.$$

We use the notation with the tilde here, because the symbol $\Gamma_h$ (without a tilde) will be used for another discrete surface approximation that we will introduce in the following section. The accuracy of the approximation $\phi_h$ is crucial for the higher order parametric method that we will use in the remainder. If the level set function becomes too steep $\|\nabla\phi_h\| >> 1$ or too flat $\|\nabla\phi_h\| << 1$, it is difficult to compute the zero level of $\phi_h$ (and thus the surface) accurately.

### 4.1.1 Parametric Finite Element Spaces

From (4.1.1) and (4.1.3) it follows

$$\operatorname{dist}(\tilde{\Gamma}_h, \Gamma) \lesssim \max_{x\in\Gamma_h} |\phi(\mathbf{x})| = \max_{x\in\Gamma_h} |\phi(\mathbf{x}) - \phi_h(\mathbf{x})| \leq \|\phi - \phi_h\|_{L^\infty(\mathcal{N}_\Gamma^\delta)} \lesssim h^{k_g+1}.$$

Thus, the discrete surface approximation $\tilde{\Gamma}_h$ represents a higher order approximation of the exact surface $\Gamma$. However, the integration over this surface is hard to realize numerically for $k_g > 1$ and $\tilde{\Gamma}_h$ is not feasible for the TraceFEM. Instead, we aim to construct a method where all surface integrals can be calculated using standard quadrature rules for piecewise planar surfaces. The linear approximation of the level set function is given by $\hat{\phi}_h := I_h^1\phi_h$ and the planar surface approximation by

$$\Gamma_h^{\text{lin}} := \left\{\mathbf{x} \in \Omega \mid \hat{\phi}_h(\mathbf{x}) = 0\right\}.$$

This discrete surface satisfies

$$\operatorname{dist}\left(\Gamma, \Gamma_h^{\text{lin}}\right) \lesssim h^2$$

and since it is piecewise planar, integrals on $\Gamma_h^{\text{lin}}$ can be calculated accurately using standard quadrature rules.

The cut elements and the domain formed by these elements are denoted by

$$\mathcal{T}_\Gamma = \mathcal{T}_\Gamma^{\text{lin}} := \left\{ T \in \mathcal{T}_h \mid T \cap \Gamma_h^{\text{lin}} \neq \emptyset \right\} \quad \text{and} \quad \Omega_\Gamma = \Omega_\Gamma^{\text{lin}} := \bigcup_{T \in \mathcal{T}_\Gamma} T. \tag{4.1.4}$$

Since $h$ is assumed to be sufficiently small, it holds $\Omega_\Gamma \subset \mathcal{N}_\Gamma^\delta$.

In the following, we construct a transformation $\Theta_h$ that is a finite element function, such that $\Theta_h(\Gamma_h^{\text{lin}}) \approx \Gamma$ holds. This transformation provides a higher order approximation of the surface $\Gamma$ where integrals over the approximation can be computed efficiently using the piecewise planar surface $\Gamma_h^{\text{lin}}$. For a detailed derivation and implementation aspects concerning the transformation, we refer to [Leh16] were the method is introduced.

We start with the construction of an ideal (and in general not available) transformation $\Psi$ that maps the piecewise planar surface $\Gamma_h^{\text{lin}}$ onto the exact surface $\Gamma$. It should satisfy

$$\hat{\phi}_h = \phi \circ \Psi \quad \text{on } \Omega_\Gamma.$$

Since we are discretizing a PDE on the surface $\Gamma$, we only need to apply the transformation to the cut elements $\mathcal{T}_\Gamma$ and not on the whole mesh. The transformation satisfies

$$\mathbf{x} \in \Gamma_h^{\text{lin}} \quad \Leftrightarrow \quad 0 = \hat{\phi}_h(\mathbf{x}) = (\phi \circ \Psi)(\mathbf{x}) = \phi\big(\Psi(\mathbf{x})\big) \quad \Leftrightarrow \quad \Psi(\mathbf{x}) \in \Gamma.$$

Thus, we obtain $\Psi\big(\Gamma_h^{\text{lin}}\big) = \Gamma$. If the exact level set function $\phi$ were known, the transformation $\Psi$ would be constructed by defining $\tilde{d}(\mathbf{x})$ as the smallest number in absolute value such that

$$\phi\big(\mathbf{x} + \tilde{d}(\mathbf{x}) \nabla\phi(\mathbf{x})\big) = \hat{\phi}_h(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Omega_\Gamma$$

holds. Taking $h$ sufficiently small, leads to a well defined $\tilde{d}(\mathbf{x})$. The transformation $\Psi$ is then defined by

$$\Psi(\mathbf{x}) := \mathbf{x} + \tilde{d}(\mathbf{x}) \nabla\phi(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Omega_\Gamma.$$

Note that this mapping is not available in general since it needs the unknown exact level set function $\phi$. But only replacing $\phi$ with the approximation $\phi_h$ will not lead to a well-posed and efficiently computable transformation. The low order level set function $\hat{\phi}_h$ is in general not smooth across mesh facets. This leads to a non smooth transformation within the elements when replacing $\phi$ by $\phi_h$, see [Leh16] for further details. For this reason, we construct the transformation more locally on each element separately. We use the polynomial extension $\mathcal{E}_T : P_{k_g}(T) \to P_{k_g}(\mathbb{R}^3)$ defined by $(\mathcal{E}_T q_h)_{|T} = q_{h|T}$ for $q_h \in V_{h,k_g}$ and $T \in \mathcal{T}_\Gamma$. Due to this local construction, the computation of the transformation can be done in parallel, which leads to a computationally efficient method.

Another issue is that the gradient $\nabla\phi_h$ is in general discontinuous across mesh facets. This leads to a discontinuity which can either be handled by a projection into a finite element space for the whole mapping $\Psi_h$ or by projecting the gradient $\nabla\phi_h$ into a vector valued finite element space. The latter approach is used in the following and we replace $\nabla\phi_h$ with $\Pi_h^{k_g}(\nabla\phi_h)(\mathbf{x})$ in the construction of $\Psi_h$.

To construct the discrete transformation $\Theta_h$, we define $\tilde{d}_h(\mathbf{x})$ as the smallest number in absolute value such that it holds

$$\mathcal{E}_T\phi_h\big(\mathbf{x} + \tilde{d}_h(\mathbf{x})\ \Pi_h^{k_g}(\nabla\phi_h)(\mathbf{x})\big) = \hat{\phi}_h(\mathbf{x}) \quad \text{for } \mathbf{x} \in T \in \mathcal{T}_\Gamma.$$

Then, the mapping $\Psi_h$ is defined by

$$\Psi_h(\mathbf{x}) := \mathbf{x} + \tilde{d}_h(\mathbf{x})\ \Pi_h^{k_g}(\nabla\phi_h)(\mathbf{x}) \quad \text{for } \mathbf{x} \in T \in \mathcal{T}_\Gamma.$$

Note that the projection of the gradient $\nabla\phi_h$ into the vector valued finite element space is not necessary for the construction and sometimes omitted in the literature. Since $\Psi_h$ is discontinuous across mesh facets, we project it into the finite element space using the Oswald-type projection operator $\Pi_h^{k_g}$ and extend it as the identity (which is a finite element function) outside of $\Omega_\Gamma$. On the cut elements, the parametric mapping is given by

$$\Theta_h := \Pi_h^{k_g}\Psi_h \in \mathbf{V}_{h,k_g}.$$

We define the transformed cut domain by

$$\Omega_\Gamma^\Theta := \Theta_h(\Omega_\Gamma)$$

and the transformed surface by

$$\Gamma_h := \Theta_h\big(\Gamma_h^{\text{lin}}\big) = \left\{\mathbf{x} \in \Omega_\Gamma^\Theta \mid \hat{\phi}_h\left(\Theta_h^{-1}(\mathbf{x})\right) = 0\right\}.$$

See Figure 4.1.1 for an illustration of the transformation $\Theta_h$ applied on the piecewise planar surface and the cut elements.

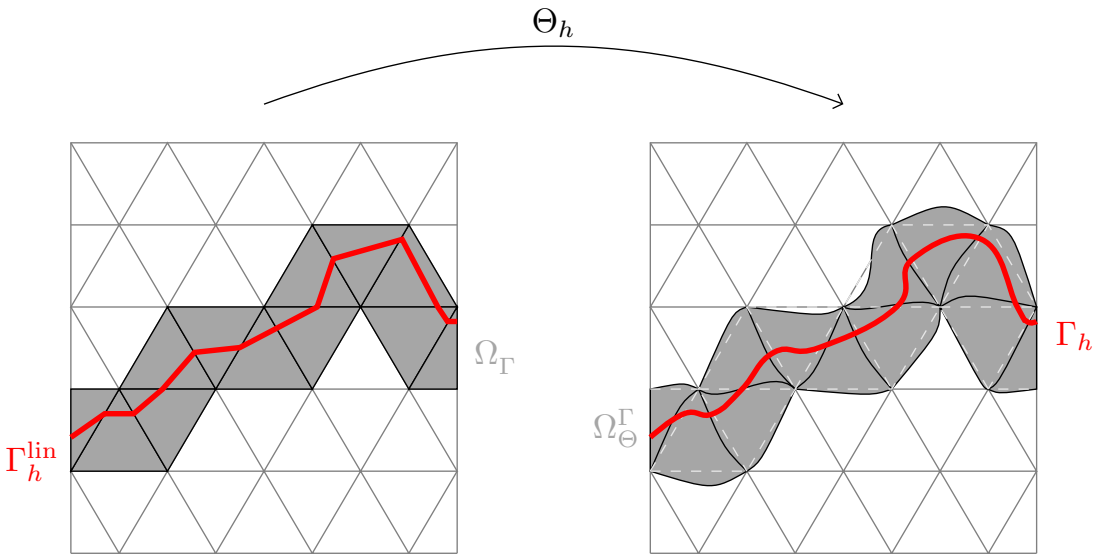



Figure 4.1.1: Transformation of $\Gamma_h^{\text{lin}}$ and $\mathcal{T}_\Gamma$ under parametric mapping $\Theta_h$.

The discrete surface satisfies

$$\operatorname{dist}(\Gamma, \Gamma_h) \lesssim h^{k_g+1},$$

cf. [LR17, Lemma 3.8] for a proof. Thus, it defines a higher order approximation of the exact surface $\Gamma$. Using the equation

$$\int_{\Gamma_h} f \,\mathrm{d}s = \int_{\Theta_h(\Gamma_h^{\text{lin}})} f \,\mathrm{d}s = \int_{\Gamma_h^{\text{lin}}} (f \circ \Theta_h) \det(\mathrm{D}\Theta_h) \,\mathrm{d}s,$$

it is possible to accurately approximate integrals over the surface $\Gamma_h$ only using surface integration with respect to the low order geometry $\Gamma_h^{\text{lin}}$ and the explicitly available mesh transformation $\Theta_h \in \mathbf{V}_{h,k_g}$. Since the surface $\Gamma_h^{\text{lin}}$ is planar, these integrals can be computed efficiently using standard quadrature rules.

The (scalar) parametric finite element space on $\Omega_\Gamma^\Theta$ is defined by

$$\mathcal{V}_{h,k,k_g} := \left\{ q_h \circ \Theta_h^{-1} \;\middle|\; q_h \in \left(V_{h,k}\right)|_{\Omega_\Gamma} \right\}.$$

We denote $\boldsymbol{\mathcal{V}}_{h,k,k_g} := \left(\mathcal{V}_{h,k,k_g}\right)^3$. Note that the subscript $k_g$ indicates the dependence on the precision of the geometric approximation. We introduce the notations $\mathcal{V}_{h,k} := \mathcal{V}_{h,k,k_g}$ and $\boldsymbol{\mathcal{V}}_{h,k} := \boldsymbol{\mathcal{V}}_{h,k,k_g}$ for the parametric finite element spaces, when the geometric order $k_g$ is clear from the context. We call the finite element space $\mathcal{V}_{h,k,k_g}$ *isoparametric finite element space* when $k = k_g$ holds. The parametric interpolation operator is denoted by $\mathcal{I}^k : (C(\Omega_\Gamma^\Theta))^l \to (\mathcal{V}_{h,k})^l$ for $l \in \{1,3\}$ and defined by the relation $(\mathcal{I}^k q) \circ \Theta_h = I^k(q \circ \Theta_h)$.

The parametric mapping $\Theta_h$ is implemented in the finite element code add-on `ngsxfem` [Leh+], that we will use in the following chapters.

### 4.1.2 Geometric quantities

We define several geometric quantities for the discrete surface approximations and cite error estimates from the literature. A natural extension of the normal vector $\mathbf{n}$ is given by the constant normal extension $\mathbf{n}^e = \mathbf{n} \circ \mathbf{p}$ with $\mathbf{p}$ the closest point projection defined in Section 2.2.2. This extension can be formulated in terms of the signed distance function $d$ by $\mathbf{n} = \nabla d$. Since we do not assume the level set function $\phi$ to be a signed distance function, the closest point projection is not available. For this reason we use another (smooth) extension of the normal vector which is given in terms of the exact level set function by

$$\mathbf{n}(\mathbf{x}) = \frac{\nabla\phi(\mathbf{x})}{\|\nabla\phi(\mathbf{x})\|}$$

for $\mathbf{x}$ in a localized vicinity around $\Gamma$. Note that in general this normal vector differs from the natural extended normal $\mathbf{n}^e$ outside of $\Gamma$ but equals it on $\Gamma$. If the level set function equals the signed distance function ($\phi = d$), this normal fulfills $\mathbf{n} = \mathbf{n}^e$ since the signed distance function satisfies $\|\nabla d\| = 1$. Based on the level set approximation, we define the following normal approximations.

**Definition 4.1.1**

An approximate low-order normal is given by

$$\mathbf{n}_h^{\text{lin}} = \mathbf{n}_h^{\text{lin}}(T) := \frac{\nabla\hat{\phi}_h(\mathbf{x})}{\left\|\nabla\hat{\phi}_h(\mathbf{x})\right\|} \quad \text{for } \mathbf{x} \in \Omega_\Gamma.$$

We further define a normal approximation within the transformed active domain by

$$\mathbf{n}_h\left(\Theta_h(\mathbf{x})\right) := \frac{\nabla\Theta_h(\mathbf{x})^{-T}\mathbf{n}_h^{\text{lin}}}{\left\|\nabla\Theta_h(\mathbf{x})^{-T}\mathbf{n}_h^{\text{lin}}\right\|} \quad \text{for } \mathbf{x} \in \Omega_\Gamma.$$

Note that besides the normal $\mathbf{n}_h$ that is computed using the level set function $\phi_h$ with polynomial degree $k_g$, we introduce the so called *improved* normal $\widetilde{\mathbf{n}}_h$. As will be discussed in subsequent sections of this thesis, the normal vector $\mathbf{n}_h$ does not provide sufficient accuracy to achieve optimal bounds for discretization errors when discretizing the tangential Navier–Stokes system from Problem 3.4.1. The improved normal is given in the following definition.

**Definition 4.1.2**

A higher order normal approximation is given by

$$\widetilde{\mathbf{n}}_h := \frac{\nabla \tilde{\phi}_h(\mathbf{x})}{\|\nabla \tilde{\phi}_h(\mathbf{x})\|} \quad \text{for } \mathbf{x} \in \Omega_\Gamma^\Theta,$$

with $\tilde{\phi}_h \in \boldsymbol{\mathcal{V}}_{h,k_g+1}$ a higher order level set approximation satisfying $\|\phi - \tilde{\phi}_h\|_{L^\infty(\mathcal{N}_\Gamma^\delta)} \lesssim h^{k_g+2}$.

The improved normal will be used in a penalty term in Chapter 5 to obtain a tangential velocity field, but not in the construction of the method for solving the "full" surface Navier–Stokes problem in Chapter 7 since the solution of this problem is in general not tangential and we do not need penalty term for the normal component.

On the cut elements $\mathcal{T}_\Gamma$, we introduce discrete versions of the tangential projection $\mathbf{P}$, and the Weingarten mapping $\mathbf{H}$ in the following definition.

**Definition 4.1.3**

We define discrete tangential projections given by

$$\mathbf{P}_h = \mathbf{P}_h(\mathbf{x}) := \mathbf{I} - \mathbf{n}_h(\mathbf{x})\mathbf{n}_h(\mathbf{x})^T \quad \text{and} \quad \widetilde{\mathbf{P}}_h = \widetilde{\mathbf{P}}_h(\mathbf{x}) := \mathbf{I} - \widetilde{\mathbf{n}}_h(\mathbf{x})\widetilde{\mathbf{n}}_h(\mathbf{x})^T \quad \text{for } \mathbf{x} \in \Omega_\Gamma^\Theta.$$

Utilizing both normals, we introduce two approximations of the Weingarten mapping denoted by

$$\mathbf{H}_h := \mathbf{P}_h \nabla \left( \Pi_h^{k_g-1} \mathbf{n}_h \right) \mathbf{P}_h \quad \text{and} \quad \widetilde{\mathbf{H}}_h := \mathbf{P}_h \nabla \left( \Pi_h^{k_g} \widetilde{\mathbf{n}}_h \right) \mathbf{P}_h.$$

Using the discrete Weingarten mapping, a discrete mean curvature is defined by

$$\kappa_h := \operatorname{tr}(\mathbf{H}_h).$$

We give some comments on the discrete geometric quantities.

**Remark 4.1.4**

Note that $\mathbf{n}_h^{\text{lin}}(\mathbf{x})$ with $\mathbf{x} \in \Gamma_h^{\text{lin}}$ constitutes a normal to $\Gamma_h^{\text{lin}}$ and $\mathbf{n}_h$ defines a normal vector on $\Gamma_h$ for points $\mathbf{x} \in \Gamma_h^{\text{lin}}$. Since the surface $\Gamma_h^{\text{lin}}$ is only continuous (and not smooth) over facets of the triangulation, its normal direction is ambiguous for $\mathbf{x} \in T_1 \cap T_2$ with $T_1, T_2 \in \mathcal{T}_\Gamma, T_1 \neq T_2$. Thus, the normal $\mathbf{n}_h^{\text{lin}}$ is discontinuous across facets of the triangulation.

When the exact level set function $\phi$ is known, a possible choice for $\tilde{\phi}_h$ is given by $\tilde{\phi}_h = \mathcal{I}^{k_g+1}\phi$.

In the definition of the discrete Weingarten mappings $\mathbf{H}_h$ and $\widetilde{\mathbf{H}}_h$, we multiply the discrete projection $\mathbf{P}_h$ from the left and the right to obtain discrete analogous of the properties $\mathbf{Hn} = 0$ and $\mathbf{PHP} = \mathbf{H}$, reading $\mathbf{H}_h\mathbf{n}_h = 0$ and $\mathbf{P}_h\mathbf{H}_h\mathbf{P}_h = \mathbf{H}_h$ (same for $\tilde{\mathbf{H}}_h$). ◇

In [GLR18], the following estimates regarding approximation are proven.

**Lemma 4.1.5**

The normal vectors $\mathbf{n}_h$ and $\widetilde{\mathbf{n}}_h$ satisfy the following error estimates

$$\|\mathbf{n} - \mathbf{n}_h\|_{L^\infty(\Omega_\Gamma^\Theta)} \lesssim h^{k_g} \quad \text{and} \quad \|\mathbf{n} - \widetilde{\mathbf{n}}_h\|_{L^\infty(\Omega_\Gamma^\Theta)} \lesssim h^{k_g+1}. \tag{4.1.5}$$

From this error estimates, the following estimates are directly obtained. It holds

$$\|\mathbf{P} - \mathbf{P}_h\|_{L^\infty(\Omega_\Gamma^\Theta)} \lesssim h^{k_g} \quad \text{and} \quad \|\mathbf{P} - \widetilde{\mathbf{P}}_h\|_{L^\infty(\Omega_\Gamma^\Theta)} \lesssim h^{k_g+1}.$$

The Weingarten mappings $\mathbf{H}_h$ and $\tilde{\mathbf{H}}_h$ satisfy

$$\|\mathbf{H} - \mathbf{H}_h\|_{L^\infty(\Omega_\Gamma^\Theta)} \lesssim h^{k_g-1} \quad \text{and} \quad \left\|\mathbf{H} - \widetilde{\mathbf{H}}_h\right\|_{L^\infty(\Omega_\Gamma^\Theta)} \lesssim h^{k_g}.$$

We define discrete versions of the surface differential operators.

**Definition 4.1.6**

The discrete approximations of the surface gradient for a scalar function $q \in V_{h,k}$ and a vector valued functions $\mathbf{v} \in \mathbf{V}_{h,k}$ and the surface divergence for a vector valued functions are given by

$$\nabla_{\Gamma_h} q := \mathbf{P}_h \nabla q, \quad \nabla_{\Gamma_h} \mathbf{v} := \mathbf{P}_h \nabla \mathbf{v} \mathbf{P}_h, \quad \text{and} \quad \operatorname{div}_{\Gamma_h} \mathbf{v} := \operatorname{tr}\left(\nabla_{\Gamma_h} \mathbf{v}\right).$$

## 4.2 Stabilization techniques

For each $T \in \mathcal{T}_\Gamma$, the surface element $\Gamma_T := \Gamma_h \cap T$ is either a (curved) triangle or a quadrilateral. This surface triangulation of $\Gamma_h$ does not necessarily fullfil regularity conditions; that is, the elements $\Gamma_T$ may exhibit very small internal angles, and there can be huge variation in the sizes of neighboring elements. This can result in occurrences of so-called "small cuts". A small cut is characterized by a scenario where the surface intersects an element $T \in \mathcal{T}_\Gamma$ such that $\Gamma_T = \Gamma \cap T$ is much smaller than the intersection of the surface with other elements. In such cases, the condition number of the resulting mass and stiffness matrices may blow up, leading to linear systems that are severely ill-conditioned and difficult to solve. An illustrative example of this phenomenon can be found in [GR11, Remark 13.2.7]. Furthermore, a formal definition of a small cut along with a proof regarding the increase in condition number when a domain intersects with a not aligned background mesh is provided in [Bur10, Lemma 3.1].

We introduce two stabilization techniques from the literature that will be employed to

stabilize our method. Stabilization will be achieved through the application of a bilinear form, which is selected to ensure optimal discretization error bounds while yielding a bound on the condition number of the linear systems of order $\mathcal{O}(h^{-2})$. The formal criteria governing this optimal behavior are given in [GLR18, equation (6.8)]. Also, the stabilization should be easy to implement and should not increase to computational effort too much. The first stabilization technique discussed is known as "ghost penalty" stabilization, while the second technique is referred to as "volume normal derivative" stabilization. In each stabilization, a weight $\rho > 0$ that may depend on the mesh size $h$ will appear. We discuss the choice of this weight in the subsequent sections. We will present the stabilizations for scalar functions, but they can be easily extended to vector-valued functions.

### 4.2.1 Ghost penalty stabilization

The first stabilization technique is the so-called "ghost penalty" stabilization which is introduced in [Bur10] as a stabilization mechanism for an unfitted finite element method to discretize a PDE posed on a domain cutting through a background mesh. The authors of [BHL15] were the first who used this stabilization for a TraceFEM with linear polynomials of a surface PDE, namely the Laplace-Beltrami equation. We introduce relevant notations. The set of active facets is given by

$$\mathcal{F}_h^\Gamma := \{T_1 \cap T_2 \mid T_1, T_2 \in \mathcal{T}_\Gamma,\ T_1 \neq T_2,\ \operatorname{meas}_2(T_1 \cap T_2) > 0\}.$$

For a face $F = T_1 \cap T_2 \in \mathcal{F}_h^\Gamma$, we define the facet patch $\omega_F$ as the union of the two elements $T_1$ and $T_2$ that share the face $F$. It is given by $\omega_F := T_1 \cup T_2$.

There are different versions of the ghost-penalty stabilization, cf. [LO19, section 4.3] for an overview. The first version of the ghost penalty stabilization has been proposed in [Bur10]. It is called an lps-type (local projection stabilization) version of the ghost penalty method. Here, the deviation from a polynomial on the facet patches $\omega_F$ is penalized. The bilinear form is given by

$$s_h^{\text{gp, lps}}(u,v) := \rho \sum_{F \in \mathcal{F}_h^\Gamma} \int_{\omega_F} \left(u - \Pi_{\omega_F} u\right)\left(v - \Pi_{\omega_F} v\right) \mathrm{d}\mathbf{x},$$

where $\Pi_{\omega_F} : L^2(\omega_F) \to P_k(\omega_F)$ is an $L^2$-projection into the space of polynomials on $\omega_F$ up to degree $k$. This stabilization only leads to optimal results for linear finite element methods.

The most well-known version is the derivative jump version where jumps in the (higher order) derivatives across facets are penalized. It is used in, e.g., [Bur+14; Mas+14; SW14]. The corresponding bilinear form when polynomials of degree $k$ are used is given by

$$s_h^{\text{gp, jump}}(u,v) := \rho \sum_{F \in \mathcal{F}_h^\Gamma} \sum_{m=0}^{k} \frac{h^{2m-1}}{m!^2} \int_F [\![\partial_{\mathbf{n}_F}^m u]\!][\![\partial_{\mathbf{n}_F}^m v]\!] \mathrm{d}\mathbf{x},$$

where $\partial_{\mathbf{n}_F}^m$ is the $m$−th directional derivative in the direction of the facet normal $\mathbf{n}_F$ and $[\![\cdot]\!]$ denotes the jump across the facet. The summand corresponding to $m = 0$ is not required in an implementation when the bilinear form is applied on finite element functions from $V_{h,k}$ due to the continuity of these functions. The used higher order derivatives in this stabilization increase the computational cost of the stabilization. For

that reason, we will stick to another variant of the ghost penalty methodology.

We will use the third version of the ghost penalty stabilization, the so-called "direct" one. It has been introduced in [Pre18]. The bilinear form of the stabilization with scaling parameter $\alpha \in \mathbb{Z}$ and weight constant $c_\rho > 0$ is given by

$$s_h^{\mathrm{gp},\alpha}(u,v) := \frac{c_\rho}{h^\alpha} \sum_{F \in \mathcal{F}_h^\Gamma} \int_{\omega_F} (u_1 - u_2)(v_1 - v_2)\,\mathrm{d}\mathbf{x}, \tag{4.2.1}$$

where we use the notation $u_i = \mathcal{E}_{T_i} u|_{T_i}$ for $u \in V_{h,k}$, $i = 1,2$, and $\mathcal{E}_T : P_k(T) \to P_k(\mathbb{R}^3)$ is the canonical extension of a polynomial to $\mathbb{R}^3$. The distinct advantage of this method is that an implementation of this bilinear form is only implicitly (through the extension $\mathcal{E}_T$) depending on the polynomial degree $k$.

For the error analysis in Chapter 6, we need the bilinear form $s_h^{\mathrm{gp},\alpha}$ to be defined for arbitrary functions $u, v \in L^2(\Omega)$. Therefore, we define $u_i = \mathcal{E}_T \Pi_{T_i} u_{|T_i}$, where $\Pi_{T_i}$ is the $L^2(T_i)$-projection into $P_k(T_i)$ for $i = 1,2$. For $u \in V_{h,k}$ it holds $\Pi_{T_i} v_{|T_i} = v_{|T_i}$. With this definition, the bilinear form $s_h^{\mathrm{gp},\alpha}$ from (4.2.1) is well defined for arbitrary functions $u, v \in L^2(\Omega_\Gamma)$.

We will use the direct ghost penalty stabilization in Chapter 6 in the construction of an extension operator and in Chapter 7 as a stabilization method.

### 4.2.2 Volume normal derivative stabilization

The second stabilization technique we will use is the so-called "volume normal derivative" or "volume normal" stabilization. It was introduced in [Bur+18] for linear finite elements on embedded manifolds of arbitrary codimensions and in [GLR18] for a higher order isoparametric TraceFEM. The bilinear form of the stabilization is given by

$$s_h^{\mathrm{vn}}(u,v) := \rho \int_{\Omega_\Gamma^\Theta} (\nabla u \cdot \mathbf{n}_h)\,(\nabla v \cdot \mathbf{n}_h)\,\mathrm{d}\mathbf{x}$$

with $\mathbf{n}_h$ as in Definition 4.1.1. Note that the implementation of this stabilization fits well to the structure of many finite element codes. The bilinear form yields optimal stabilization results not only for linear polynomial but also for higher order methods.

We briefly comment on the idea behind the normal derivative stabilization. Let $u \in C(\Gamma)$ denote a differentiable scalar function defined on $\Gamma$ which is extended constantly in normal direction. The extension is given by $u^e(\mathbf{x}) = u(\mathbf{p}(\mathbf{x}))$ with the closest point projection $\mathbf{p}$ on $\Omega_\Gamma$. It satisfies

$$\nabla u^e \cdot \mathbf{n} = \nabla(u \circ \mathbf{p}) \cdot \mathbf{n} = \big[\nabla \mathbf{p}\,(\nabla u \circ \mathbf{p})\big] \cdot \mathbf{n} = \big[\mathbf{P}\,(\nabla u \circ \mathbf{p})\big] \cdot \mathbf{n} = 0,$$

where we used the identity $\nabla \mathbf{p} = \mathbf{P}$ from Lemma 2.5.3. Thus, functions that are constant in normal direction are in the kernel of the bilinear form $s_h^{\mathrm{vn}}$ when the exact normal is used. The stabilization can be seen as a penalization leading to functions that are quasi constant in normal direction off the surface.

**Remark 4.2.1 (Full gradient stabilization)**

There are more stabilization techniques available in the literature. Similar to the volume normal derivative stabilization, there is a so-called "full gradient" stabilization. Its surface variant was introduced in [DER14; Reu14]. The stabilization does not result in

a uniform bound on the condition number for $k > 1$; cf. [Reu14, Remark 6.5]. Another full gradient stabilization was introduced in [Bur+16]. It uses the full gradient in a volume integral. It is easy to implement this stabilization, but the stabilization yields suboptimal convergence rates for $k > 1$; cf. [Bur+16, Lemma 6.2, Term III] and [GLR18].

◇

### 4.2.3 Example: The Laplace-Beltrami equation

We briefly show how the TraceFEM can be applied to the Laplace-Beltrami equation on a surface $\Gamma$. The Laplace-Beltrami operator is defined as the problem: Find $u \in H^1(\Gamma)$ with $\int_\Gamma u \,\mathrm{d}s = 0$ such that

$$\operatorname{div}_\Gamma \nabla_\Gamma u = f \quad \text{on } \Gamma$$

holds for a right-hand side $f \in L^2(\Gamma)$ with $\int_\Gamma f \,\mathrm{d}s = 0$. A weak formulation of this problem is given by: Find $u \in H^1(\Gamma)$ with $\int_\Gamma u \,\mathrm{d}s = 0$ such that

$$\int_\Gamma \nabla_\Gamma u \cdot \nabla_\Gamma v \,\mathrm{d}s = \int_\Gamma f v \,\mathrm{d}s \quad \text{for all } v \in H^1(\Gamma).$$

This formulation is well-posed, cf. [Dzi88]. A discrete TraceFEM formulation of this problem is given by: Find $u_h \in \mathcal{V}_{h,k}$ with $\int_{\Gamma_h} u_h \,\mathrm{d}s_h = 0$ such that

$$\int_{\Gamma_h} \nabla_{\Gamma_h} u_h \cdot \nabla_{\Gamma_h} v_h \,\mathrm{d}s_h + s_h(u_h, v_h) = \int_{\Gamma_h} f_h v_h \,\mathrm{d}s_h \quad \text{for all } v_h \in \mathcal{V}_{h,k},$$

with some extension $f_h$ of $f$ defined on $\Gamma_h$ and $s_h = s_h^{\mathrm{vn}}$ with $h \lesssim \rho \lesssim h^{-1}$ or $s_h = s_h^{\mathrm{gp},2}$ with $c_\rho \sim 1$. In [GLR18] optimal error bounds for $k = k_g \geq 1$ in the $H^1$-norm on $\Gamma$ have been shown.

## 4.3 TraceFEM for evolving surfaces

In this section, we introduce an Eulerian method based on a TraceFEM for evolving surfaces. We combine a finite difference scheme in time, namely the Backwards Differentiation Formula (BDF), with an extension of functions defined on $\Gamma(t)$ onto the successive surface $\Gamma(t + \Delta t)$ for a time step size $\Delta t > 0$. We will realize the extension using a volume normal derivative stabilization similar to the one described in the previous section. The method was introduced for a scalar problem on an evolving surface in [LOX18] and adapted for a vector-valued PDE in [ORZ22]. It was further used for PDEs on moving domains in [LL22; WRL21].

Firstly, we fix some notations. Let $t_{\mathrm{end}} > 0$ be a fixed end time and $[0, t_{\mathrm{end}}]$ the time interval that is partitioned into $N$ intervals $I_n = [t_{n-1}, t_n]$ with time steps $0 = t_0 < t_1 < \ldots < t_N = t_{\mathrm{end}}$.

The time step sizes $\Delta t_n := t_n - t_{n-1}$ for $n = 1, \ldots, N$ are assumed to be fixed for all $n = 1, \ldots, N$ ($\Delta t_n = \Delta t$) in this section. We note, however, that the methods introduced are also applicable for varying time step sizes. We denote the time-dependent surface by $\Gamma^n = \Gamma(t_n)$ with its discrete approximation $\Gamma_h^n = \Gamma_h(t_n)$. The cut elements, associated domains, and the active facets are denoted by $\mathcal{T}_\Gamma^n$, $\Omega_\Gamma^n$, and $\mathcal{F}_h^{\Gamma^n}$, respectively. We use the notation $f^n = f(\cdot, t_n)$ for time-dependent functions.

As in the previous sections, let $\Omega \subset \mathbb{R}^3$ be a polygonal domain such that $\Gamma(t) \subset \Omega$ holds for all $t \in [0, t_{\text{end}}]$ and $\{\mathcal{T}_h\}_{h>0}$ denotes a consistent family of shape regular triangulations of $\Omega$ that is independent of the surface $\Gamma(t)$.

The evolving surface $\Gamma(t)$ is advected by a smooth velocity field $\mathbf{w} = \mathbf{w}(\mathbf{x}, t)$. For each time step $n$, let $\phi_h^n$ be a discrete level set approximation of $\phi^n$ which is a level set function of the surface $\Gamma^n$ satisfying (4.1.1) and (4.1.3) and $\Theta_h^n$ be the parametric mapping introduced above. We assume that the surface is sufficiently regular in time such that the level set approximation satisfies

$$\|\phi_h^{n+1} - \phi_h^n\|_{L^\infty(\Omega)} \lesssim \Delta t \|w_N\|_{L^\infty(\Omega \times I_{n+1})}, \tag{4.3.1a}$$

$$\|\nabla\phi_h^{n+1} - \nabla\phi_h^n\|_{L^\infty(\Omega)} \lesssim \Delta t \left( \|w_N\|_{L^\infty(\Omega \times I_{n+1})} + \|\nabla w_N\|_{L^\infty(\Omega \times I_{n+1})} \right) \tag{4.3.1b}$$

for $n = 0, \dots, N-1$ with $w_N = \mathbf{w} \cdot \mathbf{n}$.

In the following chapters, we will see that solving a surface Navier–Stokes system needs significant computational effort for each time step. Consequently, we aim to avoid explicit time-stepping schemes, as they are known to require very small time steps to maintain stability. Instead, we will employ $A$-stable or at least $A(\alpha)$-stable Backward Differentiation Formulas (BDF(R)), which permit the use of much larger time steps. The literature on BDF schemes is extensive; for a comprehensive overview, see [HW96]. Our primary focus will be on schemes of order $R < 4$ for discretizing the partial derivative $\partial_t \mathbf{u}(t) = \mathbf{f}(t, \mathbf{u})$. These schemes are defined by

$$\begin{aligned}
\text{BDF(1):} \quad & \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \mathbf{f}(t_{n+1}, \mathbf{u}^{n+1}), \\
\text{BDF(2):} \quad & \frac{3\mathbf{u}^{n+1} - 4\mathbf{u}^n + \mathbf{u}^{n-1}}{2\Delta t} = \mathbf{f}(t_{n+1}, \mathbf{u}^{n+1}), \\
\text{BDF(3):} \quad & \frac{11\mathbf{u}^{n+1} - 18\mathbf{u}^n + 9\mathbf{u}^{n-1} - 2\mathbf{u}^{n-2}}{6\Delta t} = \mathbf{f}(t_{n+1}, \mathbf{u}^{n+1}).
\end{aligned}$$

In this section, we focus on the BDF(1) scheme (or implicit Euler method), but the ideas presented below are also applicable to higher order time stepping schemes, see Remark 4.3.2. Note that for the time stepping, the difference $\mathbf{u}^{n+1} - \mathbf{u}^n$ needs to be evaluated on $\Gamma^{n+1}$. Also for other time stepping schemes (not only for the BDF), this difference is needed.

Given the dynamic nature of the surface, we have $\Gamma^{n+1} \neq \Gamma^n$, and consequently, the velocity field $\mathbf{u}^n$, which is defined solely on $\Gamma^n$, is not defined on $\Gamma^{n+1}$. As a result, the difference $\mathbf{u}^{n+1} - \mathbf{u}^n$ is not well defined. To address this issue, we extend the function $\mathbf{u}^n$ from $\Gamma^n$ to a neighborhood that includes $\Gamma^{n+1}$. Ensuring well-posedness of this extension needs assumptions regarding the time-step size and the regularity of the surface $\Gamma(t)$ that we address now. As detailed in Section 2.2, there exists a constant $\delta > 0$, independent of time, such that the signed distance function is smooth within the tubular domain $\mathcal{N}_\Gamma^\delta(t)$. We aim to construct a method that is well defined without using a smooth signed distance function. We only need a normal defined in a vicinity around $\Gamma(t)$. We introduce a neighborhood $\mathcal{O}(\Gamma(t))$ of the surface $\Gamma(t)$ as the domain where the level set function satisfies $\|\nabla\phi\| \geq c > 0$. In this neighborhood the normal vector $\mathbf{n} = \nabla\phi/\|\nabla\phi\|$ is well defined. In general, it holds $\mathcal{N}_\Gamma^\delta(t) \subset \mathcal{O}(\Gamma(t))$.

Within this domain, we define an extensions operator $\mathcal{E}^e : C^1(\Gamma^n) \to C(\mathcal{O}(\Gamma^n))$ by

$$\begin{cases} \mathcal{E}^e f = f & \text{on } \Gamma^n, \\ \nabla(\mathcal{E}^e f) \cdot \mathbf{n}^n = 0 & \text{in } \mathcal{O}(\Gamma^n). \end{cases}$$

Thus, $\mathcal{E}^e f$ is the constant extension in normal direction of the surface. We assume $\Delta t$ to be sufficiently small (compared to $\delta$), such that

$$\Gamma^{n+1} \subset \mathcal{O}(\Gamma^n) \tag{4.3.2}$$

holds. This relation ensures the well-posedness of the difference $\mathbf{u}^{n+1} - \mathcal{E}^e \mathbf{u}^n$ on $\Gamma^{n+1}$ and thus it also guarantees the well-posedness of the time-stepping scheme. See Figure 4.3.1 for a visualization of equation (4.3.2).

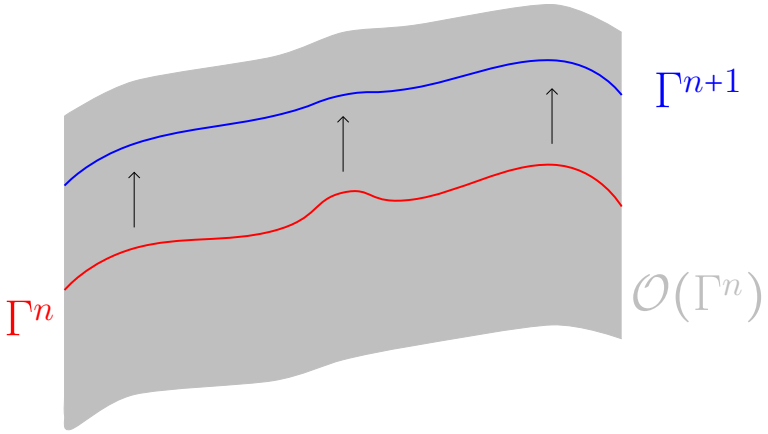



Figure 4.3.1: The neighborhood $\mathcal{O}(\Gamma^n)$ of $\Gamma^n$ containing the surface $\Gamma^{n+1}$.

We construct a numerical analogue of the extension operator $\mathcal{E}^e$ using the volume normal derivative stabilization. To define the extension domain, we specify the thickness

$$\delta_n := c_\delta \Delta t \|w_N\|_{L^\infty(\Omega \times I_n)}, \tag{4.3.3}$$

with $w_N = \mathbf{w} \cdot \mathbf{n}$ and a constant $c_\delta > 0$. The extension elements are given by

$$\mathcal{T}_\delta = \mathcal{T}_\delta^n := \left\{ T \in \mathcal{T}_h : |\phi_h^n(\mathbf{x})| \le \delta_n \text{ for some } \mathbf{x} \in T \right\} \tag{4.3.4}$$

with the corresponding extension domain

$$\Omega_\delta = \Omega_\delta^n := \bigcup_{T \in \mathcal{T}_\delta} T. \tag{4.3.5}$$

We assume the extension domain $\Omega_\delta^n$, the cut domain $\Omega_\Gamma^{n+1}$, and the domain $\mathcal{O}(\Gamma^n)$, where the exact extension operator $\mathcal{E}^e$ is defined, satisfy

$$\Omega_\delta^n \subset \mathcal{O}(\Gamma^n) \quad \text{and} \quad \Omega_\Gamma^{n+1} \subset \mathcal{O}(\Gamma^n).$$

This assumption can be ensured by choosing $\Delta t$ and $h$ sufficiently small.

For $\mathbf{x} \in \Omega_\Gamma^{n+1}$ it holds $|\phi_h^{n+1}(\mathbf{x})| \lesssim h$ and the level set function $\phi_h^n$ satisfies

$$\begin{aligned} |\phi_h^n(\mathbf{x})| &\leq |\phi_h^n(\mathbf{x}) - \phi_h^{n+1}(\mathbf{x})| + |\phi_h^{n+1}(\mathbf{x})| \overset{(4.3.1a)}{\lesssim} \Delta t \|w_N\|_{L^\infty(\Omega \times I_{n+1})} + h \\ &\lesssim c_\delta \Delta t \|w_N\|_{L^\infty(\Omega \times I_n)} \overset{(4.3.3)}{=} \delta_n, \end{aligned}$$

for $\Delta t$ sufficiently small and $c_\delta$ sufficiently large. This implies the crucial relation

$$\Gamma_h^{n+1} \subset \Omega_\Gamma^{n+1} \subset \Omega_\delta^n. \tag{4.3.6}$$

Thus, when an extension $\mathcal{E}_h^e$ from the surface $\Gamma_h^n$ to the neighborhood $\Omega_\delta^n$ is performed, the needed difference $\mathbf{u}_h^{n+1} - \mathcal{E}_h^e \mathbf{u}_h^n$ is well defined on $\Gamma_h^{n+1}$. Here, $\mathbf{u}_h^n$ denotes a finite element approximation of $\mathbf{u}^n$. See Figure 4.3.2 for a visualization of (4.3.6).

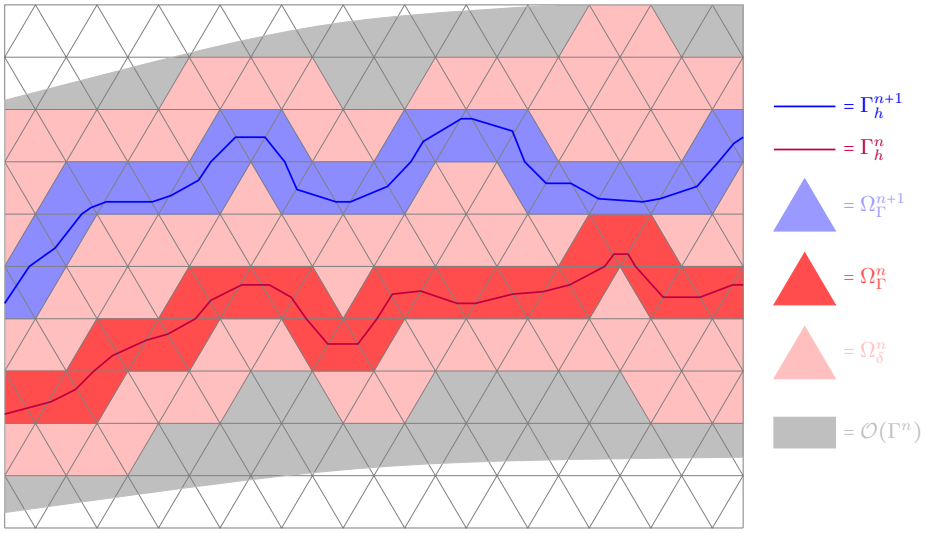



Figure 4.3.2: The discrete extension domain $\Omega_\delta^n$ of $\Gamma_h^n$ containing the surface $\Gamma_h^{n+1}$.

We will define the discrete extension operator $\mathcal{E}_h^e$ implicitly through a volume normal derivative stabilization. We apply the volume normal derivative stabilization introduced in the previous section within the extension domain $\Omega_\delta^n$. The bilinear form associated with this stabilization is defined by

$$s_h^{\text{ext}}(u, v) := \rho \int_{\Omega_\delta^n} (\nabla u \cdot \mathbf{n}_h)(\nabla v \cdot \mathbf{n}_h) \, \mathrm{d}\mathbf{x},$$

with some $\rho > 0$ that may depend on $h$. This stabilization is added to a finite element weak formulation to perform the discrete extension $\mathcal{E}_h^e$ in a weak form.

To illustrate the extension, we consider a scalar model problem characterized by a well defined weak formulation. During the discretization process of the first equation of either the Navier–Stokes system (3.4.2) or the tangential system (3.4.1), weak formulations similar to this model problem appear. Thus, the extension technique applied in the model problem can be directly transferred to the surface Navier–Stokes discretizations.

The model problem reads: Find $u \in V$ such that it holds

$$\int_{\Gamma(t)} \partial_t u(t) \cdot v \, \mathrm{d}s + a(u(t), v) = l(v) \quad \text{for all } v \in V, \tag{4.3.7}$$

where $V$ is a "suitable" function space, $a(\cdot,\cdot)$ represents a bilinear form, and $l(\cdot)$ denotes a linear form. We assume both $a(\cdot,\cdot)$ and $l(\cdot)$ do not contain any time derivatives of $u$ or $v$ and $a(u(t),v)$ is well defined for functions $u(t)$ and $v$ defined on $\Gamma(t)$.

We discretize the model problem (4.3.7) in time using the BDF(1) scheme and in space using a TraceFEM. The steps to perform one time step of the discretization are summarized in the following algorithm.

**Algorithm 4.3.1**

**Input:**

- $u_h^n$ defined on $\Omega_\delta^n$ ▷ FE approximation of $u^n$
- $\phi_h^{n+1} \in V_{h,k_g}$ ▷ FE approximation of $\phi^{n+1}$ defining $\Gamma_h^{n+1}$

**Procedure** Perform one time step

Step 1: Check if $\Omega_\Gamma^{n+1} \subset \Omega_\delta^n$ holds. If not → break

Step 2: $\delta_{n+1} \leftarrow c_\delta \Delta t \|w_N\|_{L^\infty(\Omega\times I_{n+1})}$ ▷ thickness of extension domain

Step 3: Define $\Omega_{\delta_{n+1}}^{n+1}$ along (4.3.4) and (4.3.5) ▷ extension domain

Step 4: $V_h^{n+1} \leftarrow V_{h,k}|_{\Omega_{\delta_{n+1}}^{n+1}}$ ▷ restricted FE space

Step 5: Find $u_h^{n+1} \in V_h^{n+1}$ such that

$$\int_{\Gamma_h^{n+1}} \frac{u_h^{n+1}-u_h^n}{\Delta t} \cdot v_h \,\mathrm{d}s + a(u_h^{n+1}, v_h) + s_h^{\mathrm{ext}}(u_h^{n+1}, v_h) = l(v_h) \tag{4.3.8}$$

holds for all $v_h \in V_h^{n+1}$.

**end Procedure**

**Output:**

- $u_h^{n+1}$ defined on $\Omega_\delta^{n+1}$

We comment on the steps in the algorithm. Note that it is crucial that a level set approximation $\phi_h^{n+1}$ is available for this method. When solving the tangential Navier–Stokes system (3.4.1) in Chapter 5 the (approximative) level set function is known for each time step and can be used in the algorithm. In Chapter 7, when a full Navier–Stokes system is solved, the level set function is an unknown of the system and we will provide a method to construct an approximative level set function depending on the fluid velocity $\mathbf{u}$ in Chapter 6.

In the first step of the algorithm, we check if the extension domain of the previous step is large enough. If this is not the case, the algorithm breaks since the input function $u_h^n$ is not defined on the surface $\Gamma_h^{n+1}$ and the BDF scheme is not applicable. In this case, one can restart the algorithm with a larger $c_\delta$. In Steps 2 – 4, the extension domain and the restricted finite element space on this domain are defined. We use a restricted finite element space to improve the computational efficiency. In Step 5, the solution $u_h^{n+1}$ of the semi-discrete model problem (4.3.8) at time step $t_{n+1}$ is computed. The solution $u_h^{n+1}$ holds $\nabla u_h^{n+1} \cdot \mathbf{n}_h \approx 0$ in $\Omega_\delta^{n+1}$ when $\rho$ is sufficiently large and is thus well defined

on $\Omega_\delta^{n+1}$ and (due to (4.3.6)) on $\Gamma_h^{n+2}$. Hence, $u^{n+1}$ serves as an appropriate input for following time steps.

In [LOX18] the presented method is analysed and stability results and optimal error bounds are derived for a scalar surface PDE.

**Remark 4.3.2 (Higher order time derivative)**

If a higher order time derivative, e.g., a BDF(R)-scheme with $R > 1$ is used, we need the functions $u^{n-i}$, $i = 0, \dots, R-1$ to be defined on $\Gamma^{n+1}$. Hence, we need the domains $\mathcal{O}(\Gamma^{n-i})$, where the exact extension operator is well defined, and the domains $\Omega_{\delta_{n-i}}^{n-i}$, where the discrete extension is performed, to satisfy

$$\Gamma_h^{n+1} \subset \bigcap_{i=0,\dots,R-1} \mathcal{O}(\Gamma^{n-i}) \quad \text{and} \quad \Omega_\Gamma^{n+1} \subset \bigcap_{i=0,\dots,R-1} \Omega_{\delta_{n-i}}^{n-i}$$

instead of (4.3.2) and (4.3.6). ◇

**Remark 4.3.3 (Time dependence of parametric mapping $\Theta_h$)**

As the surface evolves over time, it generally holds $\Gamma^n \neq \Gamma^{n+1}$. Thus, the parametric mapping $\Theta_h$ is time-dependent, and we denote the mapping at time step $n$ by $\Theta_h^n$. Consequently, the finite element spaces $\mathcal{V}_{h,k}$ which depend on the mapping $\Theta_h^n$, are also time-dependent. We denote the space at time step $n$ by $\mathcal{V}_{h,n}^k$ and let $u_h^n \in \mathcal{V}_{h,n}^k$ be a finite element function at time step $n$. It follows that $u_h^n \notin \mathcal{V}_{h,n+1}^k$ holds. In [LL22], a projection operator that maps functions from $\mathcal{V}_{h,n}^k$ to $\mathcal{V}_{h,n+1}^k$ is introduced and analysed. The analysis demonstrates that the application of this projection at each time step does not lead to suboptimal accumulation of errors. This projection will be used to project the solutions of our system from one time step to the subsequent ones in the implementations in Chapters 5 and 7. ◇

In the following Chapters 5 and 7 we will apply the introduced TraceFEM (including the higher order parametric mapping, the BDF scheme and the extension done by the volume normal derivative stabilization) to the (tangential and full) Navier–Stokes systems derived in Chapter 3.

# Chapter 5

# Discretization of Tangential Surface Navier–Stokes Equations

This chapter is based on the work [ORS24].

## 5.1 Introduction

In this chapter, we develop a numerical approach to solve the tangential surface Navier–Stokes system formulated in Problem 3.4.1. The system is a simplified version of the complete surface Navier–Stokes equations, where the normal component of the velocity is given by the known surface velocity $V_\Gamma$. It can be interpreted as a PDE posed on a *passively* evolving surface embedded in $\mathbb{R}^3$. The tangential component $\mathbf{u}_T$ of the velocity and the surface pressure $p$ are the unknown quantities.

A weak variational formulation of TSNS was shown to be well-posed for any finite final time and without smallness conditions on the data in [ORZ22]. For that variational formulation, we introduce a discretization method based on a trace finite element method (TraceFEM) to discretize the system in space, and a backward differentiation formula (BDF) for the time discretization, as described in the previous chapter. The combination of both methods leads to a fully Eulerian finite element method that does not need a surface parametrization and uses tangential calculus in the embedding space $\mathbb{R}^3$.

A variant of the method under consideration has been applied to time-dependent Stokes equations in a moving volume domain $\Omega(t) \subset \mathbb{R}^3$ in [WRL21; BFM22]. In the work by Lehrenfeld et al. [WRL21], Taylor–Hood finite elements are employed, whereas Burman et al. [BFM22] utilize equal order finite element spaces combined with continuous interior penalty pressure stabilization for spatial discretization. Both studies present comprehensive discretization error analyses, resulting in suboptimal error bounds for velocity and pressure in [WRL21], and suboptimal pressure error bounds in [BFM22]. The authors of both papers attribute this suboptimality primarily to the absence of a uniform discrete pressure stability bound, cf. [WRL21, Remark 5.11], [BFM22, Section 4.1].

In [ORS24], a novel argument is introduced, leading to a discrete pressure stability bound that is uniform within the parameter range $h^2 \lesssim \Delta t$. This advancement facilitated the derivation of error estimates for velocity and pressure that are optimal within the parameter range $h^2 \lesssim \Delta t \lesssim h$. These parameter restrictions are reasonable when considering

BDF(1) or BDF(2) time discretization methods combined with low-order finite element pairs such as Taylor–Hood $P_2$-$P_1$ elements. Stability results for the system under consideration are proven, and optimal order error estimates are derived in an energy norm for velocity and a special $H^1$-type norm for pressure. This study presents the first numerical analysis of fluid PDE systems posed on evolving surfaces, with results confirmed by numerical experiments.

Recently, there has been a growing interest in the numerical analysis and computational method development for fluid equations on surfaces; see, e.g., [RV18; Fri18; BDL20; Bra+22; Ols+18; OY19; ORZ21; Jan+21; ORZ22; KKV23]. While many of these studies focus on method development, those addressing numerical error analysis primarily consider the simplified case of homogeneous viscous surface fluid flow on a *steady* smooth surface. Particularly relevant to the present work are the stability and error analyses of finite element discretizations for the Stokes problem on a steady surface, presented in [BDL20; ORZ21; Jan+21]. The authors analysed $H(\mathrm{div})$–conforming finite elements (cf. [BDL20]), unfitted $P_2$-$P_1$ elements (cf. [ORZ21]), and higher order Taylor–Hood bulk elements (cf. [Jan+21]).

The latter two papers provide a comprehensive convergence analysis under the assumption of exact numerical integration over the curvilinear surface. This assumption was subsequently relaxed in [Jan+21], which also extended the analysis to higher order trace Taylor–Hood-type elements. Additionally, in the papers [Gro+18; JR20; HLL19] the surface FEM of Dziuk and TraceFEM for the surface vector-Laplace problem, closely related to the surface Stokes problem, are analysed. Results from these studies are used in our error analysis.

In [KKV23] a problem similar to the one considered in this chapter is discretized using a SurfaceFEM. The authors also use a penalization term to obtain a tangential discrete velocity, but they employ the penalization in the continuous system, leading to a slightly different method. They present various numerical experiments and show numerically the convergence of their penalization technique.

The remainder of this chapter is structured as follows. In Section 5.2 the tangential Navier-Stokes system is repeated. In Section 5.3 the discretization method is introduced, starting with the numerical time integration approach before presenting the fully discrete method. Section 5.4 contains the main results of the error analysis, outlining coercivity and continuity estimates for the discrete problem and demonstrating key ideas for deriving a stability result. Substituting the continuous solution in the discrete variational formulation results in consistency terms for which bounds are derived by using an established approach, based on discrete stability, consistency error bounds and interpolation error bounds. The main result of the analysis is Theorem 5.4.8, which yields optimal order error estimates for the velocity in an energy norm and for pressure in a special $H^1$-type norm. We discuss the main ideas of the proof without giving all the details. Results of numerical experiments presented in Section 5.5 illustrate the optimal order convergence of the method.

## 5.2 Problem formulation

In this section, we recall the tangential surface Navier–Stokes equations that are treated in this chapter. Consider, for $t \in [0, t_{\mathrm{end}}]$, a closed surface $\Gamma(t)$ embedded in $\mathbb{R}^3$ as defined

in Chapter 2, with a density distribution $\rho(\mathbf{x},t)$ and the density flow field $\mathbf{u}(\mathbf{x},t)$, for $\mathbf{x} \in \Gamma(t)$. Here, we assume that the geometric evolution of $\Gamma(t)$ is known beforehand and is determined by a given smooth velocity field $\mathbf{w} = \mathbf{w}(\mathbf{x},t)$, which passively advects the initial surface $\Gamma_0 := \Gamma(0)$. This velocity field is normal to the surface, i.e., $\mathbf{w} = w_N\mathbf{n} = V_\Gamma\mathbf{n}$ holds. The surface $\Gamma(t)$ is given by the image of the initial surface under the flow map $\mathbf{X}_N(\mathbf{z},t)$, i.e.,

$$\Gamma(t) = \left\{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x} = \mathbf{X}_N(\mathbf{z},t),\ \mathbf{z} \in \Gamma_0\right\}, \tag{2.1.2}$$

cf. Chapter 2. Here, $\mathbf{X}_N(\mathbf{z},t)$ represent the trajectories uniquely defined by the problem

$$\begin{cases} \mathbf{X}_N(\mathbf{z},0) = \mathbf{z}, \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}_N(\mathbf{z},t) = \mathbf{w}\left(\mathbf{X}_N(\mathbf{z},t),t\right) \qquad \text{for } t \in [0,t_{\mathrm{end}}], \end{cases}$$

for all $\mathbf{z} \in \Gamma_0$, see (2.1.3). This induces the smooth space-time manifold

$$\mathcal{S} = \bigcup_{t\in[0,t_{\mathrm{end}}]} \Gamma(t) \times \{t\} \subset \mathbb{R}^4.$$

The definitions of geometric quantities on $\Gamma(t)$ (e.g., the normal $\mathbf{n}$, the tangential projection $\mathbf{P}$, and the Weingarten mapping $\mathbf{H}$), the extension of functions off the surface $g^e$, the differential operators (surface gradient, surface divergence, and material derivative), and the rate-of-strain tensor $\mathbf{E}_\Gamma$ are given in Chapter 2.

Besides the material derivative along trajectories evolved by the fluid flow $\mathbf{u}$ (as introduced in Definition (2.3.1)), denoted by $\dot{f}$, we also consider a second material derivative, denoted by $\mathring{f}$, which describes the variation of $f$ along trajectories of points on $\Gamma(t)$ induced by the surface flow field $\mathbf{w}$. It is defined along the same lines as the representation of the material derivative from Lemma 2.3.3. It holds

$$\mathring{f}(\mathbf{y},t) := \frac{\mathrm{d}}{\mathrm{d}s} f\left(\mathbf{X}_{\mathbf{w},\mathbf{y},t}(t+s),t+s\right)\Big|_{s=0} \qquad \text{for } (\mathbf{y},t) \in \mathcal{S},$$

for a scalar- or vector-valued function $f$ defined on $\mathcal{S}$, with $\mathbf{X}_{\mathbf{w},\mathbf{y},t}$ defined by equation (2.3.2) with $\mathbf{u}$ replaced by $\mathbf{w}$. This material derivative is also used in Section 3.2. Similarly to the Cartesian representation of the material derivative $\dot{\mathbf{v}}$ from Lemma 2.3.2, the material derivative $\mathring{\mathbf{v}}$ can be represented in Cartesian coordinates as

$$\mathring{\mathbf{v}} := \frac{\partial}{\partial t}\mathbf{v}^e + \nabla\mathbf{v}^e\mathbf{w} \qquad \text{on } \mathcal{S}, \tag{5.2.1}$$

for vector-valued functions $\mathbf{v}$ defined on $\mathcal{S}$ and extended off the surface. Again, we split velocity fields defined on $\Gamma(t)$ into tangential and normal components

$$\mathbf{v} = \mathbf{v}_T + \mathbf{v}_N = \mathbf{v}_T + v_N\mathbf{n}, \qquad \text{with } \mathbf{v}_T = \mathbf{P}\mathbf{v} \text{ and } v_N = \mathbf{v}\cdot\mathbf{n}.$$

Particles on the surface must have the same normal velocity as the surface itself, since they cannot leave the surface. Thus, the normal component of the fluid velocity $\mathbf{u}$ is completely determined by the ambient flow $\mathbf{w}$, i.e., $u_N = w_N$ holds on $\Gamma(t)$. We derive

a relation between the material derivatives $\dot{\mathbf{v}}$ and $\mathring{\mathbf{v}}$ given by

$$\mathring{\mathbf{v}} = \dot{\mathbf{v}} - \nabla \mathbf{v}^e \mathbf{u}_T \quad \text{on } \mathcal{S}.$$

Using this relation, we derive the following representation of the tangential component of the material derivative of $\mathbf{u}_T$. It holds

$$\mathbf{P}\dot{\mathbf{u}}_T = \mathbf{P}\left(\mathring{\mathbf{u}}_T + \nabla \mathbf{u}_T^e \mathbf{u}_T\right) = \mathbf{P}\mathring{\mathbf{u}}_T + \mathbf{P}\nabla \mathbf{u}_T^e \mathbf{P}\mathbf{u}_T = \mathbf{P}\mathring{\mathbf{u}}_T + \nabla_\Gamma \mathbf{u}_T \mathbf{u}_T. \tag{5.2.2}$$

One could also consider an ambient surface velocity $\mathbf{w}$ that is not normal, i.e., $\mathbf{P}\mathbf{w} \neq 0$, leading to an ambient normal flow $w_N$ that moves the surface itself and a tangential flow $\mathbf{w}_T$ that moves the particles on the surface. The tangential flow induced by the tangential surface Navier–Stokes equations $\mathbf{u}_T$ also moves particles on the surface but does not change the surface evolution. The whole tangential movement of particles on the surface would be given by $\mathbf{w}_T + \mathbf{u}_T$, leading to a more complex system of equations, which is not considered in this work. We stick to a normal ambient flow $\mathbf{w} = w_N \mathbf{n}$.

Based on system (3.4.1) derived in Chapter 3 and equation (5.2.2), we obtain the following problem.

**Problem 5.2.1: Tangential surface Navier–Stokes equations (TSNS)**

For a given closed, smooth surface $\Gamma(t)$ that evolves smoothly in time with surface velocity $\mathbf{w}$, viscosity $\mu_0$, and a (tangential) force term $\mathbf{f}_T$, find the velocity $\mathbf{u}_T$ satisfying the initial condition $\mathbf{u}_T(0) = \mathbf{u}_T^0$ and the pressure $p$, that satisfy

$$\begin{cases} \mathbf{P}\mathring{\mathbf{u}}_T + \nabla_\Gamma \mathbf{u}_T \mathbf{u}_T - 2\mu_0 \mathbf{P} \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}_T) + \nabla_\Gamma p + w_N \mathbf{H}\mathbf{u}_T = \mathbf{f}_t & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u}_T = f_{\text{div}} & \text{on } \Gamma(t), \\ \mathbf{u}_T \cdot \mathbf{n} = 0 & \text{on } \Gamma(t), \end{cases} \tag{5.2.3}$$

with the right-hand sides $\mathbf{f}_t = \mathbf{f}_T + 2\mu_0 \mathbf{P} \operatorname{div}_\Gamma (w_N \mathbf{H}) + w_N \nabla_\Gamma w_N$ and $f_{\text{div}} = -w_N \kappa$. We call this system the tangential surface Navier–Stokes equations (TSNS).

Note that $\mathring{\mathbf{u}}_T$ is the material derivative of $\mathbf{u}_T$ and not the tangential component of the material derivative of the full velocity field $\mathbf{u}$, that is $\mathbf{P}\mathring{\mathbf{u}}$. The system can be regarded as an idealized representation of the dynamics of a thin fluid layer situated within a bulk fluid, where frictional forces between the surface and the bulk fluids are disregarded, along with any influence of the layer on the overall flow of the bulk fluid. System (5.2.3) can also be viewed as an auxiliary problem when directional splitting is applied to the complete set of equations that govern the evolution of a material inextensible fluidic surface; cf. Chapter 3 for further details.

## 5.3 Discretization method

In this section, we treat a discretization method for the tangential surface Navier–Stokes system (TSNS) (5.2.3). We use a trace finite element method (TraceFEM) for the spatial discretization, combined with a BDF scheme for the time discretization. The combination yields a fully Eulerian finite element method. The used TraceFEM is introduced in

Chapter 4. To get a well-posed finite element formulation on the *moving* surface $\Gamma(t)$ we use an extension via volume normal derivative stabilization in a narrow band around the surface, cf. Section 4.3. To simplify the representation, we do not consider the parametric TraceFEM that is used to obtain higher order discretizations and assume that all occurring surface integrals on discrete surfaces can be calculated sufficiently accurate. In the implementation, we use the parametric TraceFEM introduced in Chapter 4.

**Remark 5.3.1 (Generic constants)**

In this chapter, some generic positive constants such as $C, c, c_0, c_1$, etc. appear. These constants are independent of the discretization parameters (like $h, \Delta t$), the time $t$, and the position of $\Gamma$ within the background mesh. However, they may depend on variables such as $w_N$, $\mathcal{S}$, $\mathbf{u}$, and the shape regularity of the mesh. To minimize repetitive usage of these constants, we utilize the notation $x \lesssim y$ that states that there exists a generic constant $c > 0$ such that the inequality $x \leq cy$ holds for quantities $x$ and $y$. Similarly, we use $x \gtrsim y$ to indicate that there exists a constant satisfying the reverse inequality, while the notation $x \simeq y$ means that both relations $x \lesssim y$ and $x \gtrsim y$ are valid. ◇

We recall some notations and definitions. We assume a given sufficiently smooth level set function $\phi = \phi(\mathbf{x}, t)$ such that the surface is implicitly described by the equation

$$\Gamma(t) := \{\mathbf{x} \in \mathbb{R}^3 \mid \phi(\mathbf{x}, t) = 0\} \qquad \text{for } t \in [0, t_{\text{end}}].$$

Note that $\phi$ is not necessarily a signed distance function, but it is *close to a signed distance function* in the sense that $\|\nabla\phi\| \approx 1$ and $\|\nabla\phi\| \geq c$ hold with a constant $c > 0$ in $\mathcal{O}(\mathcal{S})$. Here, $\mathcal{O}(\mathcal{S})$ denotes a neighborhood of $\mathcal{S}$. We assume that the level set function $\phi$ is sufficiently smooth such that the geometric quantities can be represented in terms of $\phi$ and its derivatives in $\mathcal{O}(\mathcal{S})$.

In the remainder, we start with the discretization of the time derivative $\partial_t \mathbf{u}_T$ in the material derivative $\mathring{\mathbf{u}}_T$ and introduce the time-stepping scheme. Then, we present the spatial discretization of the tangential surface Navier–Stokes equations (5.2.3) using TraceFEM.

### 5.3.1 Time-stepping scheme

We start the discretization of equation (5.2.3) by focusing on the discretization of the time derivative $\partial_t \mathbf{u}_T^e$ that appears in the material derivative $\mathring{\mathbf{u}}_T$, cf. equation (5.2.1). For a conventional time discretization approach such as a Backward Differentiation Formula (BDF) scheme or a Runge-Kutta method, the computation of the difference $\mathbf{u}_T(t_1) - \mathbf{u}_T(t_2)$ for two distinct time points $t_1, t_2 \in [0, t_{\text{end}}]$ is required. However, due to the surface's movement, this difference is in general not well defined in our context since $\mathbf{u}_T(t_2)$ is not defined on $\Gamma(t_1)$ and vice versa. To address this issue, we extend functions defined on a fixed surface $\Gamma(t^\star)$ so that they are also well defined on the surface at the following time point, namely $\Gamma(t^\star + \Delta t)$ with some $\Delta t > 0$. This extension allows a consistent comparison between values at different time steps despite the evolving geometry of the surface. The extension is performed using a volume normal derivative stabilization, introduced in Chapter 4.

For simplicity, we use a uniform time step $\Delta t = \frac{t_{\text{end}}}{N}$ with some $N > 0$. We define time nodes $t_n = n\Delta t$, $I_n = [t_{n-1}, t_n]$ and $\Gamma^n = \Gamma(t_n)$ for $1 \leq n \leq N$. If not stated otherwise, the geometric quantities $\mathbf{n}$, $\mathbf{P}$, $\mathbf{w}$, $w_N$, $\mathbf{H}$ and $\kappa$ refer to the actual surface $\Gamma^n$, i.e.,

$\mathbf{n} = \mathbf{n}^n = \mathbf{n}(\cdot, t_n)$. Let $\mathbf{u}_T^{n-1} : \Gamma^{n-1} \to \mathbb{R}^3$ denote the function $\mathbf{u}_T(\cdot, t_{n-1})$ defined on the surface $\Gamma(t_{n-1})$. Again, let $\mathcal{O}(\Gamma^{n-1})$ be a sufficiently large neighborhood of $\Gamma^{n-1}$ such that

$$\Gamma^n \subset \mathcal{O}(\Gamma^{n-1}) \quad \text{for all } n = 1, \dots, N \tag{5.3.1}$$

holds. This assumption is crucial for the extension method. In each time step, $\mathbf{u}_T^{n-1}$ is extended constantly in normal direction in $\mathcal{O}(\Gamma^{n-1})$ off the surface. We denote the extension by $\mathcal{E}\mathbf{u}_T^{n-1}$. Hence, $\mathcal{E}\mathbf{u}_T^{n-1}$ is also defined on $\Gamma^n$, and the implicit Euler (or BDF(1)) scheme

$$\partial_t \mathbf{u}_T(t_n) \approx \frac{\mathbf{u}_T^n - \mathcal{E}\mathbf{u}_T^{n-1}}{\Delta t}$$

is well defined on $\Gamma^n$. We will impose the extension condition in a weak sense in the full discrete system later. As described in Remark 4.3.2, higher order BDF schemes can be applied. Since the surface velocity $\mathbf{w}$ is assumed to be normal, i.e., $\mathbf{w} = w_N \mathbf{n}$, it holds $\nabla(\mathcal{E}\mathbf{v})\,\mathbf{w} = w_N \nabla(\mathcal{E}\mathbf{v})\,\mathbf{n} = 0$ and thus the last term in (5.2.1) vanishes, and we have

$$\overset{\circ}{(\mathcal{E}\mathbf{v})} = \frac{\partial}{\partial t}(\mathcal{E}\mathbf{v}) \quad \text{on } \mathcal{S}. \tag{5.3.2}$$

Based on this simplification of the material derivative, we introduce the normal time derivative approximation

$$\mathbf{P}\overset{\circ}{(\mathcal{E}\mathbf{u}_T)} = \mathbf{P}\frac{\partial}{\partial t}\mathcal{E}\mathbf{u}_T \approx \frac{\mathbf{u}_T(t_n) - \mathbf{P}(t_n)\mathcal{E}\mathbf{u}_T(t_{n-1})}{\Delta t} \quad \text{on } \Gamma(t_n). \tag{5.3.3}$$

We simplify the notations and use $\mathbf{u}_T^n = \mathcal{E}\mathbf{u}_T(\cdot, t_n)$ and $p^n = p(\cdot, t_n)$, respectively. Based on (5.3.3) we introduce the following time discretization method for (5.2.3). Given $\mathbf{u}_T^0$ in $\mathcal{O}(\Gamma_0)$, for $n = 1, \dots, N$, find $\mathbf{u}_T^n$, defined in $\mathcal{O}(\Gamma^n)$ and tangential to $\Gamma^n$, i.e., $(\mathbf{u}_T^n \cdot \mathbf{n})|_{\Gamma^n} = 0$, and $p^n$ defined on $\Gamma^n$ such that

$$\begin{cases} \dfrac{\mathbf{u}_T^n - \mathbf{P}\mathbf{u}_T^{n-1}}{\Delta t} + (\nabla_\Gamma \mathbf{u}_T^n)\mathbf{u}_T^{n-1} - 2\mu_0 \mathbf{P}\,\mathrm{div}_\Gamma\, \mathbf{E}_\Gamma(\mathbf{u}_T^n) + \nabla_\Gamma p^n + w_N^n \mathbf{H}\mathbf{u}_T^n = \mathbf{f}_t^n \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathrm{div}_\Gamma\, \mathbf{u}_T^n = f_{\mathrm{div}}^n \end{cases} \tag{5.3.4a}$$

holds on $\Gamma^n$ and in $\mathcal{O}(\Gamma^n)$ it holds

$$\nabla \mathbf{u}_T^n \mathbf{n} = 0. \tag{5.3.4b}$$

Here, we use the notations $w_N^n := w_N(t_n)$, $\mathbf{f}_t^n := \mathbf{f}_t(t_n)$, $f_{\mathrm{div}}^n := f_{\mathrm{div}}(t_n)$.

The time discretization employed in equation (5.3.4a) is done via a first-order Backward Differentiation Formula (BDF(1)), which has an error of $\mathcal{O}(\Delta t)$. We employ a linear approximation of the quadratic nonlinearity that maintains first-order accuracy with respect to $\Delta t$; specifically, the term $(\nabla_\Gamma \mathbf{u}_T^n)\mathbf{u}_T^n$ is replaced by $(\nabla_\Gamma \mathbf{u}_T^n)\mathbf{u}_T^{n-1}$. This approach allows a straightforward extension to higher order schemes in terms of $\Delta t$, as further discussed in Remark 5.3.3. The spatial discretization associated with equation (5.3.4) will be explained in the next section.

### 5.3.2 Space discretization method

We now explain the spatial discretization of (5.3.4) using a TraceFE method as described in Chapter 4. Consider a fixed polygonal domain $\Omega \subset \mathbb{R}^3$ that strictly contains $\Gamma(t)$ for all $t \in [0, t_{\text{end}}]$. Let $\{\mathcal{T}_h\}_{h>0}$ be a family of shape-regular consistent triangulations of $\Omega$, with $h = \max_{T\in\mathcal{T}_h} \operatorname{diam}(T)$. We repeat the definition of the standard bulk finite element space of piecewise polynomial continuous functions of a fixed degree $k \geq 1$. As in (4.1.2), it is given by

$$V_{h,k} = \{q \in C(\Omega) \mid q_{|T} \in P_k(T) \text{ for all } T \in \mathcal{T}_h\}. \tag{4.1.2}$$

The bulk velocity and pressure finite element spaces are standard Taylor–Hood spaces, denoted by

$$\mathbf{U}_h = \mathbf{V}_{h,m+1}, \quad Q_h = V_{h,m}, \quad \text{with } m \geq 1. \tag{5.3.5}$$

As described in Chapter 4, we do not need an explicit parametric representation of $\Gamma(t_n)$ or the exact level set function $\phi(\cdot, t_n)$. Instead, we use a finite element approximation of $\phi$. This is a more realistic assumption, as the exact level set function is not known in systems in which the normal component of the velocity is unknown, and finding $\Gamma(t_n)$ is part of the problem. In turn, this results in approximation of all geometric quantities involved in (5.2.3). This issue of geometry approximation is explained in the section below.

#### Geometry approximation

We revisit the definitions and properties of the discrete level set function, along with other geometric quantities as outlined in Section 4.1. To simplify the notation in this section and throughout the subsequent error analysis, we assume that *integrals over the discrete surface approximations can be computed accurately.* As discussed in Chapter 4, this assumption is easy to satisfy for piecewise linear level set approximations. However, it becomes more challenging in higher order cases. In our implementation, we employ the parametric finite element technique introduced in the previous chapter to ensure sufficiently accurate approximations of these integrals.

For any fixed $t \in [0, t_{\text{end}}]$, let $\phi_h = \phi_h(\cdot, t) \in V_{h,k_g}$ be a continuous piecewise polynomial approximation of $\phi(\cdot, t)$ with some $k_g \geq 1$. We use the notation $\phi_h^n(\mathbf{x}) = \phi_h(\mathbf{x}, t_n)$ for $n = 0, \dots, N$. The discrete level set function $\phi_h$ is assumed to satisfy the approximation properties (4.1.3) and (4.3.1). It holds

$$\|\phi - \phi_h\|_{L^\infty(\Omega)} + h\|\nabla(\phi - \phi_h)\|_{L^\infty(\Omega)} \lesssim h^{k_g+1}, \tag{4.1.3}$$

and

$$\|\phi_h^{n-1} - \phi_h^n\|_{L^\infty(\Omega)} \lesssim \Delta t \|w_N\|_{L^\infty(\Omega\times I_n)} \tag{4.3.1a}$$

$$\|\nabla\phi_h^{n-1} - \nabla\phi_h^n\|_{L^\infty(\Omega)} \lesssim \Delta t \left(\|w_N\|_{L^\infty(\Omega\times I_n)} + \|\nabla w_N\|_{L^\infty(\Omega\times I_n)}\right) \tag{4.3.1b}$$

for $n = 1, \dots, N$. Therefore, the exact level set function $\phi$ has to be sufficiently smooth, i.e., $\phi(\cdot, t) \in C^{k_g+1}(\Omega)$ is assumed to hold. Additionally, we assume that $|\nabla\phi_h| \geq C > 0$ in $\mathcal{O}(\Gamma(t))$ for $t \in [0, t_{\text{end}}]$.

The discrete Lipschitz surface $\Gamma_h \approx \Gamma$ is defined as the zero level of $\phi_h$. It is given by

$$\Gamma_h(t) := \{\mathbf{x} \in \mathbb{R}^3 \mid \phi_h(\mathbf{x}, t) = 0\}.$$

The geometric quantities on $\Gamma_h$ are given by the discrete normal $\mathbf{n}_h$, its finite element approximation $\overline{\mathbf{n}}_h$, the tangential projection $\mathbf{P}_h$, and the discrete Weingarten mapping $\mathbf{H}_h$. They are defined in terms of the level set function $\phi_h$ and its derivatives by

$$\mathbf{n}_h = \frac{\nabla\phi_h}{\|\nabla\phi_h\|}, \quad \overline{\mathbf{n}}_h = \Pi_h^{k_g-1}\mathbf{n}_h \quad \mathbf{P}_h = \mathbf{I} - \mathbf{n}_h\mathbf{n}_h^T, \quad \text{and} \quad \mathbf{H}_h = \nabla_{\Gamma_h}\overline{\mathbf{n}}_h, \tag{5.3.6}$$

with $\Pi_h^{k_g-1}\mathbf{n}_h$ the (componentwise) Oswald-type interpolation operator that maps into the finite element space $\mathbf{V}_{h,k_g-1}$. Moreover, we need the "improved" normal approximation $\widetilde{\mathbf{n}}_h$ from Definition 4.1.2. The discrete geometric quantities satisfy the approximation properties given in Lemma 4.1.5. We emphasize that all bounds in this lemma are uniform in $t$.

The data $w_N$, $f_{\mathrm{div}}$, $\mathbf{f}_t$ are extended (sufficiently accurate) such that they are also defined on $\Gamma_h$. We also denote these extensions by $w_N$, $f_{\mathrm{div}}$, and $\mathbf{f}_t$.

**Fully discrete method**

As described in Section 5.3.1, we need to extend the functions from the surface to a small neighborhood. For this, we use the volume normal derivative extension introduced in Section 4.3. We recall the definitions of the cut elements from (4.1.4) and the used *narrow band* around $\Gamma_h^n = \Gamma_h(t_n)$ (4.3.4) – (4.3.5). They are given by

$$\mathcal{T}_\Gamma^n = \left\{T \in \mathcal{T}_h \mid T \cap \Gamma_h^n \neq \emptyset\right\} \quad \text{and} \quad \Omega_\Gamma^n = \bigcup_{T\in\mathcal{T}_\Gamma^n} T \tag{4.1.4}$$

and

$$\mathcal{T}_\delta^n = \mathcal{T}_{\delta_n}^n = \left\{T \in \mathcal{T}_h : |\phi_h^n(\mathbf{x})| \leq \delta_n \text{ for some } \mathbf{x} \in T\right\} \quad \text{and} \quad \Omega_\delta^n = \bigcup_{T\in\mathcal{T}_\delta^n} T,$$

with the thickness

$$\delta_n = c_\delta \Delta t \|w_N\|_{L^\infty(\Omega\times I_n)}, \tag{4.3.3}$$

with a constant $c_\delta > 0$.

We restrict the *time-independent bulk spaces* $\mathbf{U}_h$ and $Q_h$ to the narrow bands $\Omega_\delta^n$ and $\Omega_\Gamma^n$ to get the finite element spaces

$$\mathbf{U}_h^n := \{\mathbf{v}_{|\Omega_\delta^n} \mid \mathbf{v} \in \mathbf{U}_h\} \quad \text{and} \quad Q_h^n := \{q_{|\Omega_\Gamma^n} \mid q \in Q_h\}.$$

Note that the velocity space is defined on the narrow band $\Omega_\delta^n$, while the pressure space is solely defined on the cut elements. Since no pressure time derivative is involved in the system, we do not need to extend the pressure onto the next surface, and it is sufficient to define it on the cut elements $\Omega_\Gamma^n$. We also use the notation $V_{h,m}^n := \{v_{|\Omega_\delta^n} \mid v \in V_{h,m}\}$. Now, we have all ingredients to give a full discrete approximation of the semi discrete tangential surface Navier–Stokes system (5.3.4).

We have to consider three critical aspects: the tangentiality of $\mathbf{u}_T^n$, the constant extension of the velocity in normal directions as specified in equation (5.3.4b), and the handling of the nonlinear inertia term.

It is impossible to construct $H^1$-conforming tangential finite element spaces on the surface approximation. We have to weaken the tangentiality condition or the continuity. We relax the requirement for the solution to be tangential to $\Gamma^n$ by allowing $\mathbf{U}_h^n$ to serve

as trial and test velocity space. The tangentiality condition is weakly enforced through a penalty approach, which is commonly used in finite element methods for vector-valued surface partial differential equations, cf. e.g., [HLL19; Ols+18; OY19]. The penalty term reads

$$r_\tau^n(\mathbf{u}_h, \mathbf{v}_h) \coloneqq \tau \int_{\Gamma_h^n} (\mathbf{u}_h \cdot \widetilde{\mathbf{n}}_h)(\mathbf{v}_h \cdot \widetilde{\mathbf{n}}_h) \,\mathrm{d}s$$

with penalty parameter $\tau > 0$ and the improved normal $\widetilde{\mathbf{n}}_h$ from Definition 4.1.2. From the literature, it is known that such a more accurate normal in this penalty term is needed for optimal order discretization errors, cf. [HLL19; JR20].

Furthermore, we also relax the constraint presented in equation (5.3.4b). We employ it weakly through a penalty (or stabilization) method that is standard in trace finite element frameworks. This involves augmenting the discrete bilinear form with a volume normal derivative term, as detailed in Section 4.3. The needed additional volumetric term reads

$$s_{h,\mathbf{u}}^n(\mathbf{u}_h, \mathbf{v}_h) \coloneqq \rho_{\mathbf{u}} \int_{\Omega_\delta^n} (\nabla \mathbf{u}_h \mathbf{n}_h) \cdot (\nabla \mathbf{v}_h \mathbf{n}_h) \,\mathrm{d}\mathbf{x},$$

scaled by the parameter $\rho_{\mathbf{u}} > 0$. Moreover, we also need a similar stabilization term for the pressure, which is given by

$$s_{h,p}^n(p_h, q_h) \coloneqq \rho_p \int_{\Omega_\Gamma^n} (\nabla p_h \cdot \mathbf{n}_h)\, (\nabla q_h \cdot \mathbf{n}_h) \,\mathrm{d}\mathbf{x},$$

with the stabilization parameter $\rho_p > 0$. Note the difference in the stabilization terms for the velocity and pressure: while the integral in $s_{h,\mathbf{u}}^n$ is taken over the narrow band $\Omega_\delta^n$, the integral in $s_{h,p}^n$ is taken over the cut elements $\Omega_\Gamma^n$.

The bilinear form $s_{h,\mathbf{u}}^n$ plays a twofold role. Firstly, due to this term, instabilities caused by "small cuts" are damped, resulting in satisfactory conditioning of the stiffness matrix for the velocity. Secondly, this term weakly enforces the extension condition (5.3.4b) with $\mathcal{O}(\Gamma^n)$ replaced by $\Omega_\delta^n$. The second stabilization term $s_{h,p}^n$ is added for the purpose of numerical stabilization of pressure, both with respect to finite element (LBB) stability and the conditioning of the resulting matrix, cf. [ORZ21]. Due to these stabilizations, the algebraic system (in each time step) is well-posed and has conditioning properties comparable to those of a discretized linearized Navier–Stokes system in Euclidean domains. For further details on the stabilization terms, the difficulties caused by small cuts, and possible alternative stabilization techniques, we refer to Section 4.2.

The treatment of inertia also follows an established approach. We rewrite the corresponding trilinear form by

$$\int_\Gamma (\nabla_\Gamma \mathbf{u}_T \mathbf{u}_T) \cdot \mathbf{v}_T \,\mathrm{d}s = \frac{1}{2} \int_\Gamma (\nabla_\Gamma \mathbf{u}_T \mathbf{u}_T) \cdot \mathbf{v}_T - (\nabla_\Gamma \mathbf{v}_T \mathbf{u}_T) \cdot \mathbf{u}_T \,\mathrm{d}s - \frac{1}{2} \int_\Gamma \operatorname{div}_\Gamma \mathbf{u}_T (\mathbf{u}_T \cdot \mathbf{v}_T) \,\mathrm{d}s.$$

Using $\operatorname{div}_\Gamma \mathbf{u}_T = f_{\mathrm{div}}$ on $\Gamma$, the second term on the right-hand side becomes linear. The first term is linearized using an extrapolation (in time) of the $\mathbf{u}_T$ approximations from the previous time steps. When using a BDF(1) scheme, the extrapolation simply consists of the last velocity approximation $\mathbf{u}_T^{n-1}$.

We define the bilinear form $b_h^n(\cdot,\cdot)$ that covers the pressure and divergence term from (5.3.4a) by

$$b_h^n(q_h, \mathbf{v}_h) \coloneqq \int_{\Gamma_h^n} \nabla_{\Gamma_h} q_h \cdot \mathbf{v}_h \,\mathrm{d}s_h$$

and the antisymmetric bilinear form, which is convenient for the stability analysis of the discretization by

$$\begin{aligned} a_h^n(\mathbf{u}_h, \mathbf{v}_h) := &\int_{\Gamma_h^n} \left( \frac{\mathbf{u}_h - \mathbf{u}_h^{n-1}}{\Delta t} + w_N^n \mathbf{H}_h \mathbf{u}_h \right) \cdot \mathbf{P}_h \mathbf{v}_h \, \mathrm{d}s_h \\ &+ 2\mu_0 \int_{\Gamma_h^n} E_{\Gamma_h}(\mathbf{P}_h \mathbf{u}_h) : E_{\Gamma_h}(\mathbf{P}_h \mathbf{v}_h) \, \mathrm{d}s_h \\ &+ \frac{1}{2} \int_{\Gamma_h^n} \left( \nabla_{\Gamma_h}(\mathbf{P}_h \mathbf{u}_h) \mathbf{u}_h^{n-1} \right) \cdot \mathbf{v}_h - \left( \nabla_{\Gamma_h}(\mathbf{P}_h \mathbf{v}_h) \mathbf{u}_h^{n-1} \right) \cdot \mathbf{u}_h \, \mathrm{d}s_h \\ &- \frac{1}{2} \int_{\Gamma_h^n} f_{\mathrm{div}}^n \mathbf{u}_h \cdot \mathbf{P}_h \mathbf{v}_h \, \mathrm{d}s_h. \end{aligned}$$

Putting these components together leads to the following *fully discrete problem* for one time step: For given $\mathbf{u}_h^{n-1} \in \mathbf{U}_h^{n-1}$ find $\mathbf{u}_h^n \in \mathbf{U}_h^n$, $p_h^n \in Q_h^n$, satisfying

$$\begin{cases} a_h^n(\mathbf{u}_h^n, \mathbf{v}_h) + b_h^n(p_h^n, \mathbf{v}_h) + s_{h,\mathbf{u}}^n(\mathbf{u}_h^n, \mathbf{v}_h) + r_\tau^n(\mathbf{u}_h^n, \mathbf{v}_h) = \left(\mathbf{f}_t^n, \mathbf{v}_h\right)_{\Gamma_h^n} & \text{for all } \mathbf{v}_h \in \mathbf{U}_h^n \\ -b_h^n(q_h, \mathbf{u}_h^n) + s_{h,p}^n(p_h^n, q_h) = \left(f_{\mathrm{div}}^n, q_h\right)_{\Gamma_h^n} & \text{for all } q_h \in Q_h^n, \end{cases} \tag{5.3.8}$$

where we use the notation $(\cdot,\cdot)_{\Gamma_h^n}$ for the $L^2$-scalar product over $\Gamma_h^n$.

Here, we use the discrete surface differential operators from Definition 4.1.6. For example, $\nabla_{\Gamma_h} \mathbf{v}_h = \mathbf{P}_h \nabla \mathbf{v}_h \mathbf{P}_h$ holds. The fully discrete system (5.3.8) is *consistent* up to geometric errors: If the discrete surface $\Gamma_h^n$ and its geometric quantities are replaced by the exact surface $\Gamma^n$ and geometric quantities (e.g., $\mathbf{H}_h$ by $\mathbf{H}$), then the equations in (5.3.8) are satisfied with $\mathbf{u}_h^n$, $p_h^n$, $\mathbf{u}_h^{n-1}$ replaced by the exact solution $\mathbf{u}^n$, $p^n$ and $\mathbf{u}^{n-1}$ of (5.3.4).

We will comment on the choice of the parameters, $\rho_{\mathbf{u}}$, $\rho_p$, $\tau$, and $c_\delta$ in the next section.

### 5.3.3 Discussion of the method

We summarize the parameters of the discrete formulation (5.3.8) and discuss their choice. Besides the mesh and time step size $h$ and $\Delta t$, the finite element method involves:

- $\rho_{\mathbf{u}}$ : volume normal derivative stabilization and extension parameter for the velocity,
- $\rho_p$ : volume normal derivative stabilization parameter for the pressure,
- $\tau$ : penalty parameter for the tangentiality constraint,
- $c_\delta$ : narrow band parameter to define the width $\delta_n$ at time $t_n$.

We assume that the mesh size and the time step size are sufficiently small, i.e., $h \le c_1$, and $\Delta t \le c_2$ with $c_1$, $c_2$ sufficiently small $\mathcal{O}(1)$ constants. The stabilization and penalization parameters are chosen as

$$\rho_{\mathbf{u}} = c_{\rho_{\mathbf{u}}} h^{-1}, \quad \rho_p = c_{\rho_p} h, \quad \text{and} \quad \tau = c_\tau h^{-2}, \tag{5.3.9}$$

where $c_{\rho_{\mathbf{u}}}, c_{\rho_p}$ and $c_\tau$ are sufficiently large $\mathcal{O}(1)$ constants independent of the parameters,

time, and position of $\Gamma_h$ in the mesh. The width of the narrow band is chosen as (4.3.3) with $c_\delta$ sufficiently large.

The parameter conditions outlined in (5.3.9) are known from the literature concerning trace finite element methods. The requirement for a sufficiently large value of $c_{\rho_\mathbf{u}}$ is essential for achieving adequate control over the velocity extension within the narrow band; this condition is instrumental in establishing a crucial estimate, as detailed in Lemma 5.4.2. The parameter $\tau$, as specified in (5.3.9), ensures that the tangentiality condition for the discrete velocity is satisfied with sufficient accuracy. Strongly increasing $\tau$ may result in a phenomenon known as 'locking'.

Regarding the lower bound on $\delta_n$, we note that during a time step from $t_{n-1}$ to $t_n$, the surface $\Gamma(t)$ can move a maximum distance of $\Delta t \sup_{t\in I_n} \|w_N\|_{L^\infty(\Gamma(t))}$ in the normal direction. Under the assumption stated in equation (4.3.1a), it follows that the maximum distance between $\Gamma_h^n$ and $\Gamma_h^{n-1}$ is also proportional to $\Delta t \sup_{t\in I_n} \|w_N\|_{L^\infty(\Gamma(t))}$. Consequently, we can select $c_\delta$ in equation (4.3.3) such that

$$\Omega_\Gamma^n \subset \Omega_\delta^{n-1} \tag{5.3.10}$$

holds. This condition serves as a discrete analog of equation (5.3.1) and is critical for ensuring the well-posedness of the finite element formulation.

The parameters $\tau$ and $\delta_n$ have to satisfy the conditions

$$\Delta t \geq 2\tau^{-1}, \tag{5.3.11a}$$

$$\Delta t \lesssim \delta_n \lesssim h. \tag{5.3.11b}$$

Equation (5.3.11a) represents a technical condition necessary for establishing the stability estimate presented in Theorem 5.4.5. The upper bound specified in (5.3.11b) for the narrow band parameter $\delta_n$ ensures that the complexity of the method remains optimal, specifically that the number of active degrees of freedom is $\mathcal{O}(h^{-2})$ at each time step. The assumptions expressed in (5.3.11) can be fulfilled by choosing $\Delta t$ and $h$ such that they satisfy the scaling conditions

$$h^2 \lesssim \Delta t \lesssim h.$$

These scalings are reasonable. Consider Taylor–Hood elements with $m = 1$. For optimal convergence of order $\mathcal{O}(h^2)$ in the energy norm associated with equation (5.3.8), it is necessary to set $\Delta t \simeq h^2$. In contrast, for enhanced temporal accuracy using the BDF(2) scheme (our practical choice, as discussed in Remark 5.3.3), selecting $\Delta t \simeq h$ yields convergence at an order of $\mathcal{O}(h^2)$ in the energy norm, while setting $\Delta t \simeq h^{3/2}$ aligns with achieving optimal convergence at an order of $\mathcal{O}(h^3)$ in the velocity $L^2$-norm.

We briefly address further aspects of the method, its adaption to the case of a level set function that is not a signed distance function, the extension to higher order BDF schemes, and details on the implementation of surface differential operators, in the remarks below.

**Remark 5.3.2**

In practice, we typically do not have a level set function possessing the signed distance property. Thus, the extension does not satisfy (5.3.4b), but is approximately constant

along the normals to the level lines, and, instead of $\mathbf{P}\mathring{\mathbf{u}}_T = \mathbf{P}\frac{\partial \mathbf{u}_T}{\partial t}$, cf. (5.3.2), one uses the general relation

$$\mathbf{P}\mathring{\mathbf{u}}_T = \mathbf{P}\Big(\frac{\partial \mathbf{u}_T}{\partial t} + \nabla \mathbf{u}_T \mathbf{w}_N\Big),$$

as the basis for the discretization. Hence, for the case of a general level set function, we include the term

$$\int_{\Gamma_h^n} \big[\nabla(\mathbf{P}_h \mathbf{u}_h^n)\mathbf{w}_N\big] \cdot \mathbf{P}_h \mathbf{v}_h \,\mathrm{d}s_h$$

in (5.3.8). We will use this additional term in our numerical experiments. ◇

**Remark 5.3.3**

The finite element formulation (5.3.8) can be extended to higher order time discretizations using, e.g., a BDF(2) or BDF(3) method (other time discretization schemes are also possible, but in this thesis we stick to BDF methods). The BDF(2) and BDF(3) variants are very similar to the method introduced above. The time difference $\mathbf{u}_h^n - \mathbf{u}_h^{n-1}$ in the first line of (5.3.8) is then replaced by $\frac{3}{2}\mathbf{u}_h^n - 2\mathbf{u}_h^{n-1} + \frac{1}{2}\mathbf{u}_h^{n-2}$ for the BDF(2) method and by $\frac{11}{6}\mathbf{u}_h^n - 3\mathbf{u}_h^{n-1} + \frac{3}{2}\mathbf{u}_h^{n-2} - \frac{1}{3}\mathbf{u}_h^{n-3}$ for the BDF(3) method. For the linearization of the inertia term, we use $2\mathbf{u}_h^{n-1} - \mathbf{u}_h^{n-2}$ or $3\mathbf{u}_h^{n-1} - 3\mathbf{u}_h^{n-2} + \mathbf{u}_h^{n-3}$ instead of $\mathbf{u}_h^{n-1}$. All other terms in (5.3.8) remain the same.

For BDF(2), in addition to (5.3.10), one has to guarantee that additionally $\Omega_\Gamma^n \subset \Omega_\delta^{n-2}$ and, for BDF(3), $\Omega_\Gamma^n \subset \Omega_\delta^{n-3}$ is fulfilled. For that reason, the narrow bands $\Omega_\delta^n$ have to be taken broader than in the BDF(1) method. This is done by taking $\delta_n = c_\delta \Delta t \|w_N\|_{L^\infty(\Omega \times I_n)} R$ with $R = 2$ for BDF(2) and $R = 3$ for BDF(3). The choice of the parameters $\rho_{\mathbf{u}}$, $\rho_p$, $\tau$, and $c_\delta$ remains the same.

To use BDF(R) methods with $R > 1$, we also need initial values. One possibility is to start with BDF(1) steps with a smaller time step size and then switch to BDF(R). For simplicity, we will use a different approach. We start with initial values backwards in time, namely $\mathbf{u}^{-1} = \mathbf{u}(-\Delta t), \ldots, \mathbf{u}^{-R+1} = \mathbf{u}(-(R-1)\Delta t)$. Obviously, this is only possible if we have an approximation of the velocity at these time points. We will use this approach in our numerical experiments, where a manufactured solution is available. ◇

**Remark 5.3.4**

In the surface gradient operators $\nabla_{\Gamma_h}$ associated with the inertia term, and the surface rate-of-strain tensor $E_{\Gamma_h}(\cdot)$, we utilize projected vector fields $\mathbf{P}_h \mathbf{v}$. However, differentiating the projector $\mathbf{P}_h$, which is generally discontinuous across element faces, results in a loss of $H^1$ conformity, as discussed in [JR20]. To circumvent the differentiation of $\mathbf{P}_h$, we use the relation $\nabla_\Gamma \mathbf{v}_T = \nabla_\Gamma \mathbf{v} - v_N \mathbf{H}$, and consequently we substitute $\nabla_{\Gamma_h} \mathbf{P}_h \mathbf{v}_h$ with $\nabla_{\Gamma_h} \mathbf{v}_h - (\mathbf{n}_h \cdot \mathbf{v}_h)\mathbf{H}_h$ in our method's implementation. This adjustment avoids issues related to the discontinuity of the projector while maintaining the accuracy of the method. ◇

For our method, it is essential to have a sufficiently accurate finite element approximation $\phi_h^n$ of $\phi^n$. It is possible to obtain the level set approximation by taking the Oswald-type interpolation of the exact level set function $\Pi_h^{k_g} \phi^n$, when the exact level set function is known. However, in some scenarios, the exact level set function may not be available. Importantly, the method is not limited to the specific choice of $\phi_h = \Pi_h^{k_g} \phi^n$. The discrete level set function may be computed, e.g., using a transport method.

We summarize our discretization method in the following algorithm, where a BDF(1) scheme is used.

**Algorithm 5.3.5: TraceFEM for TSNS**

**Input:** $\mathbf{u}^0$, $\phi_h$, $\mathbf{f}_t$, $f_{\text{div}}$
$u_h^0 \leftarrow \Pi_h^{m+1}\mathbf{u}^0 \in \mathbf{U}_h$
$t \leftarrow 0$
**Procedure**
  **for** $n$ from 1 to $N$ **do**
    $t \leftarrow t + \Delta t$
    $\phi_h^n \leftarrow$ approximation of $\phi^n$
    $\Omega_\Gamma \leftarrow \left\{\mathbf{x} \in T \in \mathcal{T}_h \mid T \cap \Gamma_h^n \neq \emptyset\right\}$
    **if** $n > 1$ and $\Omega_\Gamma \not\subset \Omega_\delta$ **then**
      break
    **end if**
    $\delta_n \leftarrow c_\delta \Delta t \|w_N\|_{L^\infty(\Omega \times I_n)}$
    $\Omega_\delta \leftarrow \left\{\mathbf{x} \in T \in \mathcal{T}_h : |\phi_h^n(\mathbf{y})| \leq \delta_n \text{ for some } \mathbf{y} \in T\right\}$
    $\mathbf{U}_h^n \leftarrow \mathbf{V}_{h,m+1}|_{\Omega_\delta}$, $Q_h^n \leftarrow V_{h,m}|_{\Omega_\Gamma}$
    $\mathbf{u}_h^n \in \mathbf{U}_h^n$, $p_h^n \in Q_h^n \leftarrow$ solution of (5.3.8):

$$\begin{cases} a_h^n(\mathbf{u}_h^n, \mathbf{v}_h) + b_h^n(p_h^n, \mathbf{v}_h) + s_{h,\mathbf{u}}^n(\mathbf{u}_h^n, \mathbf{v}_h) + r_\tau^n(\mathbf{u}_h^n, \mathbf{v}_h) = \left(\mathbf{f}_t^n,\ \mathbf{v}_h\right)_{\Gamma_h^n} & \text{for all } \mathbf{v}_h \in \mathbf{U}_h^n, \\ -b_h^n(q_h, \mathbf{u}_h^n) + s_{h,p}^n(p_h^n, q_h) = \left(f_{\text{div}}^n,\ q_h\right)_{\Gamma_h^n} & \text{for all } q_h \in Q_h^n, \end{cases}$$

  **end for**
**end Procedure**

The algorithm is stopped if the narrow band condition (5.3.10) is not fulfilled. This is a crucial condition for the well-posedness of the method.

## 5.4 Error analysis

In this section, we give an overview of the error analysis of the finite element scheme stated in (5.3.8). We show that the method is stable and convergent with optimal error bounds. Since the analysis is technical, we do not include all details in this thesis. Especially, the proofs concerning the stability and error estimates for the pressure are omitted. We only give the main results and ideas of the proofs. The detailed proofs of the stability and convergence results can be found in the corresponding publication [ORS24].

To simplify the presentation and analysis, we assume the following properties hold throughout this section, even if these assumptions do not hold in practice usually.

**Assumption 5.4.1**

We assume that the data extensions of $w_N$, $f_{\mathrm{div}}$, $\mathbf{f}_t$ are constant extensions along the normal $\mathbf{n}$. The level set function $\phi$ is sufficiently smooth, and it has the signed distance property. The solution of (5.2.3) is sufficiently smooth, at least $\mathbf{u} \in W^{m+2,\infty}(\mathcal{S})^3$, $p \in W^{m+1,\infty}(\mathcal{S})$. We assume that the parameters are taken as in (5.3.9) and fulfill the conditions (5.3.11) with $h, \Delta t$ sufficiently small. As common in the analysis of incompressible fluid problems, we assume a homogeneous divergence condition. The right-hand side $f_{\mathrm{div}}$ satisfies

$$f_{\mathrm{div}} = 0.$$

In practice, one typically uses other (sufficiently smooth) data extensions and one does not have a global signed distance function. In addition, the homogeneous divergence condition is typically not fulfilled, since $f_{\mathrm{div}} = -w_N \kappa$ holds on the surface $\Gamma(t)$ and in general, $w_N$ and $\kappa$ do not vanish.

To represent the discrete problem in a compact form, we introduce the trilinear and bilinear forms

$$\begin{aligned}
a^n(\mathbf{z};\mathbf{u},\mathbf{v}) &:= \widehat{a}^n(\mathbf{u},\mathbf{v}) + c_n(\mathbf{z};\mathbf{u},\mathbf{v}),\\
\text{with}\qquad \widehat{a}^n(\mathbf{u},\mathbf{v}) &:= \int_{\Gamma_h^n} w_N^n \mathbf{v}\cdot\mathbf{H}_h\mathbf{u}\,\mathrm{d}s_h + 2\mu_0\int_{\Gamma_h^n} (E_{\Gamma_h}(\mathbf{P}_h\mathbf{u}) : (E_{\Gamma_h}(\mathbf{P}_h\mathbf{v}))\,\mathrm{d}s_h\\
&\quad + r_\tau^n(\mathbf{u},\mathbf{v}) + s_{h,\mathbf{u}}^n(\mathbf{u},\mathbf{v}), \qquad (5.4.1)\\
c_n(\mathbf{z};\mathbf{u},\mathbf{v}) &:= \frac{1}{2}\int_{\Gamma_h^n} \big[\nabla_{\Gamma_h}(\mathbf{P}_h\mathbf{u})\mathbf{z}\big]\cdot\mathbf{v} - \big[\nabla_{\Gamma_h}(\mathbf{P}_h\mathbf{v})\mathbf{z}\big]\cdot\mathbf{u}\,\mathrm{d}s_h,\\
b^n(p,\mathbf{v}) &:= \int_{\Gamma_h^n} \nabla_{\Gamma_h} p\cdot\,\mathbf{v}\,\mathrm{d}s_h.
\end{aligned}$$

Using this, the discrete problem can be rewritten as follows (recall that $f_{\mathrm{div}} = 0$): given $\mathbf{u}_h^{n-1} \in \mathbf{U}_h^{n-1}$ find $\mathbf{u}_h^n \in \mathbf{U}_h^n$, $p_h^n \in Q_h^n$ such that for all $(\mathbf{v}_h, q_h) \in \mathbf{U}_h^n \times Q_h^n$

$$\begin{cases} \left(\dfrac{\mathbf{u}_h^n - \mathbf{u}_h^{n-1}}{\Delta t}, \mathbf{P}_h\mathbf{v}_h\right)_{\Gamma_h^n} + a^n(\mathbf{u}_h^{n-1};\mathbf{u}_h^n,\mathbf{v}_h) + b^n(p_h^n,\mathbf{v}_h) = (\mathbf{f}_t^n, \mathbf{P}_h\mathbf{v}_h)_{\Gamma_h^n} \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad b^n(q_h,\mathbf{u}_h^n) = s_{h,p}^n(p_h^n, q_h) \end{cases} \qquad (5.4.2)$$

holds.

We use the notation $(\cdot,\cdot)_U$ and $\|\cdot\|_U$ to denote the $L^2$-scalar product and $L^2$-norm over a surface or volumetric domain $U$. We introduce the following natural energy norm for the velocity

$$|\!|\!|\mathbf{v}|\!|\!|_n^2 := \frac{1}{2}\|\mathbf{v}\|_{\Gamma_h^n}^2 + 2\mu_0\|E_{\Gamma_h}(\mathbf{P}_h\mathbf{v})\|_{\Gamma_h^n}^2 + \frac{\tau}{2}\|\widetilde{\mathbf{n}}_h\cdot\mathbf{v}\|_{\Gamma_h^n}^2 + \rho_{\mathbf{u}}\|\nabla\mathbf{v}\mathbf{n}_h\|_{\Omega_\delta^n}^2.$$

With some slight abuse of notation, we reuse the notation of the velocity energy norm for the scaled $H^1(\Omega_\Gamma^n)$-norm for the pressure. It is defined by

$$|\!|\!|q|\!|\!|_n := h^{\frac{1}{2}}\|\nabla q\|_{\Omega_\Gamma^n}. \qquad (5.4.3)$$

Note that this norm is equivalent to the *scaled* $H^1(\Gamma(t_n))$ norm in the following sense. Assume $\Gamma(t_n) \subset \Omega_\Gamma^n$ and $q^e$ is the normal extension of $q \in H^1(\Gamma(t))$. Then, $h\|\nabla_\Gamma q\|_{\Gamma(t)} \sim |||q^e|||_n$ holds.

For vector-valued functions, we define componentwise $H^1$-norms by

$$\|\mathbf{v}\|^2_{H^1(\Gamma(t_n))} := \sum_{i=1}^{3} \|v_i\|^2_{H^1(\Gamma(t_n))} \quad \text{for } \mathbf{v} \in H^1(\Gamma(t_n))^3,$$

and similarly on $\Gamma_h^n$.

We outline the structure of the error analysis. We first collect some preliminary results and a few useful estimates, with a particular emphasis on uniformness in discretization parameters, time, and the surface position in the bulk mesh. Using these results, it is easy to derive suitable coercivity and continuity estimates for the trilinear form $a^n(\cdot;\cdot,\cdot)$. In Theorem 5.4.5, stability results for the discrete solution $(\mathbf{u}_h^n, p_h^n)$ of (5.4.2) are given. The stability for the velocity is given in a natural energy norm, and a proof based on rather straightforward arguments is given. We do not give the proof for the pressure stability, which is given in a specific (non-standard) norm, because it is more involved and technical. In Section 5.4.4, we study the consistency error that quantifies geometric errors resulting from the approximation of surface and geometric quantities (normal vectors, tangential projectors, curvatures) in the finite element formulation. We analyse the consistency error, using some results already available in the literature on surface vector-Laplace and Stokes problems. Finally, in Section 5.4.5, we provide discretization error bounds. For the proof, the discretization error is split into an interpolation error and an error component in the finite element space. Bounds for the latter can be derived using the discrete stability, consistency error bounds, and bounds for the linearization error. With respect to the interpolation error, a key point is that in the Euclidean setting of this trace method, we can use a (nodal) interpolation operator on a fixed bulk mesh. Due to this, the time differentiation and interpolation operator commute, which then leads to optimal bounds for the difference of interpolation errors at $t = t_n$ and $t = t_{n-1}$.

### 5.4.1 Preliminaries

We start with some auxiliary results from the literature. We use a lifting $v^\ell$ of functions $v$ defined on $\Gamma_h^n$ to a neighborhood $\mathcal{O}(\Gamma_h^n)$, defined for $\mathbf{y} \in \mathcal{O}(\Gamma_h^n)$ by

$$v^\ell(\mathbf{y}) := v(\mathbf{x}) \quad \text{for} \quad \mathbf{x} \in \Gamma_h^n \quad \text{such that} \quad \mathbf{p}^n(\mathbf{y}) = \mathbf{p}^n(\mathbf{x}),$$

where $\mathbf{p}^n$ is the closest point projection on $\Gamma^n$. Note that the closest point projection and thus the lifting is well defined in a neighborhood of $\Gamma^n$. We need uniform interpolation and Korn-type inequalities given by

$$\|\mathbf{v}\|_{L^4(\Gamma_h^n)} \lesssim \|\mathbf{v}\|^{\frac{1}{2}}_{\Gamma_h^n} \|\mathbf{v}\|^{\frac{1}{2}}_{H^1(\Gamma_h^n)} \quad \text{for all } \mathbf{v} \in H^1(\Gamma_h^n)^3, \tag{5.4.4}$$

$$\|\mathbf{v}\|_{H^1(\Gamma_h^n)} \lesssim |||\mathbf{v}|||_n \quad \text{for all } \mathbf{v} \in \mathbf{U}_h^n. \tag{5.4.5}$$

The first equation follows from [ORZ22, Lemma 3.4] and [JR20, Lemma 5.14]. The Korn-type inequality follows from [ORZ22, Lemma 3.2], [JR20, Lemma 5.14] and [JR20, Lemma 5.16], cf. [ORS24] for details.

The following lemma provides a bound for the trace of a function on $\Gamma_h^n$ through its trace

on $\Gamma_h^{n-1}$ and the volume normal derivative. A proof can be found in [LOX18].

**Lemma 5.4.2**

The inequality

$$\|v_h\|^2_{\Gamma_h^n} \leq (1 + c\Delta t)\|v_h\|^2_{\Gamma_h^{n-1}} + \frac{1}{2}\rho_{\mathbf{u}}\Delta t\|\nabla v_h \cdot \mathbf{n}_h^{n-1}\|^2_{\Omega_\delta^{n-1}}$$

holds for all $v_h \in V_{h,m}^{n-1}$.

### 5.4.2 Coercivity and continuity estimates

We derive coercivity and continuity estimates for the trilinear form $a^n(\cdot\,;\cdot,\cdot)$. The coercivity result is given in the following lemma.

**Lemma 5.4.3**

For arbitrary $\mathbf{z} \in L^2(\Gamma_h^n)$ and $\mathbf{v}_h \in \mathbf{U}_h^n$ the inequality

$$a^n(\mathbf{z}; \mathbf{v}_h, \mathbf{v}_h) \geq |||\mathbf{v}_h|||_n^2 - \xi_h\|\mathbf{P}_h\mathbf{v}_h\|^2_{\Gamma_h^n}$$

holds with $\xi_h := 1 + \max_{n=0,\dots,N}\|w_N^n\mathbf{H}_h(t_n)\|_{L^\infty}$.

**Proof**

From the error estimate for the discrete normal approximation (4.1.5), it follows that, for $h$ sufficiently small, $\mathbf{v} \in \mathbb{R}^3$ satisfies

$$|\mathbf{v}|^2 \leq |\mathbf{n}_h \cdot \mathbf{v}|^2 + |\mathbf{P}_h\mathbf{v}|^2 \leq (1 + ch^{k_g})\big(|\widetilde{\mathbf{n}}_h \cdot \mathbf{v}|^2 + |\mathbf{P}_h\mathbf{v}|^2\big).$$

Using this inequality, $c_n(\mathbf{z}; \mathbf{v}_h, \mathbf{v}_h) = 0$, the tangentiality of $\mathbf{H}$ ($\mathbf{H}_h = \mathbf{P}_h\mathbf{H}_h\mathbf{P}_h$), and $\tau \geq 1 + ch^{k_g}$ (for $h$ sufficiently small and $\tau$ defined as in (5.3.9)), we obtain for all $\mathbf{v}_h \in \mathbf{U}_h^n$ the lower estimate

$$\begin{aligned} a^n(\mathbf{z}; \mathbf{v}_h, \mathbf{v}_h) = \widehat{a}^n(\mathbf{v}_h, \mathbf{v}_h) &= (w_N^n\mathbf{v}_h, \mathbf{H}_h\mathbf{v}_h)_{\Gamma_h^n} + |||\mathbf{v}_h|||_n^2 + \frac{\tau}{2}\|\widetilde{\mathbf{n}}_h \cdot \mathbf{v}\|^2_{\Gamma_h^n} - \frac{1}{2}\|\mathbf{v}_h\|^2_{\Gamma_h^n} \\ &\geq |||\mathbf{v}_h|||_n^2 - \Big(\|w_N^n\mathbf{H}_h(t_n)\|_{L^\infty} + \frac{1}{2}(1 + ch^{k_g})\Big)\|\mathbf{P}_h\mathbf{v}_h\|^2_{\Gamma_h^n}, \end{aligned}$$

from which the result follows. □

A continuity result is given in the next lemma.

**Lemma 5.4.4**

For $\mathbf{z} \in L^4(\Gamma_h^n)$ and $\mathbf{u}, \mathbf{v} \in \mathbf{U}_h^n$ the uniform estimate

$$a^n(\mathbf{z}; \mathbf{u}, \mathbf{v}) \lesssim \big(\|\mathbf{P}_h\mathbf{z}\|_{L^4(\Gamma_h^n)} + 1\big)|||\mathbf{u}|||_n|||\mathbf{v}|||_n$$

holds.

**Proof**

For the terms in the trilinear form $a(\cdot\,;\cdot,\cdot)$ that do not depend on the first argument, cf. (5.4.1), we apply the Cauchy-Schwarz inequality which results in

$$\left|\widehat{a}^n(\mathbf{u},\mathbf{v})\right| \lesssim |\!|\!|\mathbf{u}|\!|\!|_n |\!|\!|\mathbf{v}|\!|\!|_n. \tag{5.4.6}$$

We now consider the term $\int_{\Gamma_h^n}(\mathbf{z}\cdot\nabla_{\Gamma_h}\mathbf{P}_h\mathbf{u})\cdot\mathbf{v}\,\mathrm{d}s_h$. The term $\int_{\Gamma_h^n}(\mathbf{z}\cdot\nabla_{\Gamma_h}\mathbf{P}_h\mathbf{v})\cdot\mathbf{u}\,\mathrm{d}s_h$ can be treated similarly. Using (5.4.4) and (5.4.5) we obtain

$$\left|\int_{\Gamma_h^n}\left[\nabla_{\Gamma_h}(\mathbf{P}_h\mathbf{u})\mathbf{z}\right]\cdot\mathbf{v}\,\mathrm{d}s_h\right| \lesssim \|\mathbf{P}_h\mathbf{z}\|_{L^4(\Gamma_h^n)} |\!|\!|\mathbf{u}|\!|\!|_n |\!|\!|\mathbf{v}|\!|\!|_n.$$

Combining this with the same estimate for the other trilinear term and with (5.4.6) we obtain the desired result. □

### 5.4.3 Stability estimate

In this section, we present stability results for the discrete velocity and pressure functions. We only include the proof of the stability result for the velocity. For the pressure stability result we need the technical assumptions from [ORS24, Assumption 4.1] to hold. Using these assumptions, the velocity stability result, a suitable discrete inf-sup property, and results taken from [ORZ21], the stability result for the pressure can be shown, cf. [ORS24]. We define the upper bound $\mathbf{F}_n = \mathbf{F}_n(\mathbf{u}_h^0, \mathbf{f}_t)$ used in the stability theorem below by

$$\mathbf{F}_n(\mathbf{u}_h^0,\mathbf{f}_t) := \exp(c\,t_n)\left(\|\mathbf{u}_h^0\|_{\Gamma_h^0}^2 + \Delta t |\!|\!|\mathbf{u}_h^0|\!|\!|_0^2 + \Delta t\sum_{k=0}^{n}\|\mathbf{f}_t^k\|_{\Gamma_h^k}^2\right),$$

with $c$ independent of $h$, $\Delta t$, and $n$.

**Theorem 5.4.5**

The discrete problem (5.4.2) has a unique solution $(\mathbf{u}_h^n, p_h^n)$ that satisfies

$$\|\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \sum_{k=1}^{n}\|\mathbf{P}_h\mathbf{u}_h^k - \mathbf{u}_h^{k-1}\|_{\Gamma_h^k}^2 + \Delta t\sum_{k=1}^{n}\left(|\!|\!|\mathbf{u}_h^k|\!|\!|_k^2 + 2\rho_p\|\nabla p_h^k\cdot\mathbf{n}_h\|_{\Omega_\Gamma^k}^2\right) \lesssim \mathbf{F}_n(\mathbf{u}_h^0,\mathbf{f}_t)$$

and

$$\Delta t\sum_{k=1}^{n}|\!|\!|p_h^k|\!|\!|_k \lesssim \mathbf{F}_n + \left(\frac{h}{\sqrt{\Delta t}}+1\right)\mathbf{F}_n^{\frac{1}{2}} \lesssim \mathbf{F}_n + \mathbf{F}_n^{\frac{1}{2}}.$$

**Proof**

We test (5.4.2) with $\mathbf{v}_h = \mathbf{u}_h^n$ and $q_h = p_h^n$. Adding the two identities leads to

$$\begin{aligned}&\frac{1}{2}\left(\|\mathbf{P}_h\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \|\mathbf{P}_h\mathbf{u}_h^n - \mathbf{u}_h^{n-1}\|_{\Gamma_h^n}^2\right) + \Delta t a^n(\mathbf{u}_h^{n-1};\mathbf{u}_h^n,\mathbf{u}_h^n) + \Delta t\rho_p\|\nabla p_h^n\cdot\mathbf{n}_h\|_{\Omega_\Gamma^n}^2\\ &= \frac{1}{2}\|\mathbf{u}_h^{n-1}\|_{\Gamma_h^n}^2 + \Delta t(\mathbf{f}_t^n, \mathbf{P}_h\mathbf{u}_h^n)_{\Gamma_h^n}.\end{aligned}$$

We use the coercivity bound from Lemma 5.4.3 and apply Lemma 5.4.2 to obtain

$$\begin{aligned}&\|\mathbf{P}_h\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \|\mathbf{P}_h\mathbf{u}_h^n - \mathbf{u}_h^{n-1}\|_{\Gamma_h^n}^2 + 2\Delta t\left(|||\mathbf{u}_h^n|||_n^2 + \rho_p\|\nabla p_h^n\cdot\mathbf{n}_h\|_{\Omega_\Gamma^n}^2\right)\\ &\leq (1+c\,\Delta t)\|\mathbf{u}_h^{n-1}\|_{\Gamma_h^{n-1}}^2 + \frac{1}{2}\Delta t\rho_{\mathbf{u}}\|\nabla\mathbf{u}_h^{n-1}\cdot\mathbf{n}_h^{n-1}\|_{\Omega_\delta^{n-1}}^2 + C\Delta t\|\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \Delta t\|\mathbf{f}_t^n\|_{\Gamma_h^n}^2,\end{aligned} \tag{5.4.7}$$

with some $c$ and $C$ independent of $h$, $\Delta t$ and $n$. Equation (5.3.11a) yields

$$\frac{1}{2}\Delta t|||\mathbf{u}_h^n|||_n^2 \geq \|\mathbf{u}_h^n\cdot\mathbf{n}_h\|_{\Gamma_h^n}^2 - c\,\Delta t\,h^{2k_g}\|\mathbf{u}_h^n\|_{\Gamma_h^n}^2. \tag{5.4.8}$$

Using this and

$$\|\mathbf{u}_h^n\|_{\Gamma_h^n}^2 = \|\mathbf{P}_h\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \|\mathbf{n}_h\cdot\mathbf{u}_h^n\|_{\Gamma_h^n}^2$$

in (5.4.7), we obtain

$$\begin{aligned}&\|\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \|\mathbf{P}_h\mathbf{u}_h^n - \mathbf{u}_h^{n-1}\|_{\Gamma_h^n}^2 + \frac{3}{2}\Delta t|||\mathbf{u}_h^n|||_n^2 + 2\Delta t\,\rho_p\|\nabla p_h^n\cdot\mathbf{n}_h\|_{\Omega_\Gamma^n}^2\\ &\qquad\leq (1+c\,\Delta t)\|\mathbf{u}_h^{n-1}\|_{\Gamma_h^{n-1}}^2 + \frac{1}{2}\Delta t|||\mathbf{u}_h^{n-1}|||_{n-1}^2 + C\Delta t\|\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \Delta t\|\mathbf{f}_t^n\|_{\Gamma_h^n}^2.\end{aligned}$$

We sum up the inequalities for $n = 1,\dots,k$, and with $c^* = c + C$ we get

$$\begin{aligned}&\|\mathbf{u}_h^k\|_{\Gamma_h^k}^2 + \sum_{n=1}^k\|\mathbf{P}_h\mathbf{u}_h^n - \mathbf{u}_h^{n-1}\|_{\Gamma_h^n}^2 + \Delta t\sum_{n=1}^k\left(|||\mathbf{u}_h^n|||_n^2 + 2\rho_p\|\nabla p_h^n\cdot\mathbf{n}_h\|_{\Omega_\Gamma^n}^2\right)\\ &\qquad\leq (1+c\Delta t)\|\mathbf{u}_h^0\|_{\Gamma_h^0}^2 + \frac{1}{2}\Delta t|||\mathbf{u}_h^0|||_0^2 + \Delta t\sum_{n=0}^k c^*\|\mathbf{u}_h^n\|_{\Gamma_h^n}^2 + \Delta t\sum_{n=0}^k\|\mathbf{f}_t^n\|_{\Gamma_h^n}^2.\end{aligned}$$

Finally, we shift the term $\Delta t\,c^*\|\mathbf{u}_h^k\|_{\Gamma_h^k}^2$ from the right-hand side to the left-hand side and apply, for $\Delta t \leq (2c^*)^{-1}$, a discrete Gronwall inequality, which yields the desired result for the velocity stability. □

**Remark 5.4.6 (Discussion of the stability bound for the pressure)**

The stability bound for the discrete pressure $p_h$ ensures stability in an $L^1(H^1)$-type norm with a uniform constant when $h^2 \lesssim \Delta t$ holds, which is typical for practical parameter choices for $m = 1$ (the lowest order Taylor–Hood pair). Specifically, $\Delta t \sim h^2$ for BDF(1) and $\Delta t \sim h$ for BDF(R) with $R \geq 2$ are common scenarios. The stability bound can be compared to those derived in the literature, cf. [BFM22] and [WRL21], where a similar discretization method for the Stokes problem on moving domains is examined. These analyses provide a uniform pressure bound in terms of $\Delta t$ and $h$, but only for the quantity $\Delta t^2\sum_{k=1}^n\|p_h^k\|_{\Omega_k}$. Notice the square in the scaling factor $\Delta t^2$ in this term. Consequently, these analyses do not yield uniform bounds for $\Delta t\sum_{k=1}^n\|p_h^k\|_{\Omega_k}$ in the cases $\Delta t \sim h$ or $\Delta t \sim h^2$. ◇

### 5.4.4 Consistency error analysis

In this section, we summarize estimates for the consistency error. The analysis employs a standard methodology and relies on existing estimates in the literature of TraceFEM discretizations of surface vector-Laplace and surface Stokes equations on stationary surfaces. To establish the consistency and error bounds, we use estimates on the derivatives of the extended solution $\mathbf{u}^e(\mathbf{x},t) = \mathbf{u}(\mathbf{p}(\mathbf{x}),t)$ within the domain $\mathcal{O}(\mathcal{S})$, where again $\mathbf{p}$

denotes the closest point projection. For notational simplicity, we denote the extension of (scalar or vector-valued) functions $v$ defined on $\mathcal{S}$ by the same symbol $v$, i.e., with some abuse of notation, it holds $v = v^e$.

We define the consistency error $\mathcal{E}_C^n(\mathbf{v}_h)$ collecting consistency terms due to geometric errors, time derivative approximation, and linearization. It is given by

$$\begin{aligned}
\mathcal{E}_C^n(\mathbf{v}_h) := & \underbrace{\int_{\Gamma_h^n} \left(\frac{\mathbf{u}^n - \mathbf{u}^{n-1}}{\Delta t}\right) \cdot \mathbf{P}_h \mathbf{v}_h \,\mathrm{d}s_h - \int_{\Gamma^n} \mathbf{u}_t(t_n) \cdot \mathbf{v}_h^\ell \,\mathrm{d}s}_{=: I_1} \\
& + \underbrace{\rho_{\mathbf{u}} \int_{\Omega_\delta^n} ((\mathbf{n}_h - \mathbf{n}) \cdot \nabla \mathbf{u}^n)(\mathbf{n}_h \cdot \nabla \mathbf{v}_h) \,\mathrm{d}\mathbf{x}}_{=: I_2} \\
& + \underbrace{\frac{1}{2} \int_{\Gamma_h^n} \left[\nabla_{\Gamma_h}(\mathbf{P}_h \mathbf{u}^n)\mathbf{u}^{n-1}\right] \cdot \mathbf{v}_h - \left[\nabla_{\Gamma_h}(\mathbf{P}_h \mathbf{v}_h)\mathbf{u}^{n-1}\right] \cdot \mathbf{u}^n \,\mathrm{d}s_h}_{=: I_{3(1)}} \\
& \underbrace{-\frac{1}{2} \int_{\Gamma^n} \left[\nabla_\Gamma \mathbf{u}^n \mathbf{u}^n\right] \cdot \mathbf{v}_h^\ell - \left[\nabla_\Gamma(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n\right] \cdot \mathbf{u}^n \,\mathrm{d}s}_{=: I_{3(2)}} \\
& + \underbrace{2\mu_0 \int_{\Gamma_h^n} E_{\Gamma_h}(\mathbf{P}_h \mathbf{u}^n) : E_{\Gamma_h}(\mathbf{P}_h \mathbf{v}_h) \,\mathrm{d}s_h - 2\mu_0 \int_{\Gamma^n} E_\Gamma(\mathbf{u}^n) : E_\Gamma(\mathbf{P}\mathbf{v}_h^\ell) \,\mathrm{d}s}_{=: I_4} \\
& + \underbrace{\int_{\Gamma_h^n} w_N^n \mathbf{v}_h \cdot \mathbf{H}_h \mathbf{u}^n \,\mathrm{d}s_h - \int_{\Gamma^n} w_N^n \mathbf{v}_h^\ell \cdot \mathbf{H}\mathbf{u}^n \,\mathrm{d}s}_{=: I_5} \\
& + \underbrace{\tau \int_{\Gamma_h^n} (\widetilde{\mathbf{n}}_h \cdot \mathbf{u}^n)(\widetilde{\mathbf{n}}_h \cdot \mathbf{v}_h) \,\mathrm{d}s_h - \tau \int_{\Gamma^n} (\mathbf{n} \cdot \mathbf{u}^n)(\mathbf{n} \cdot \mathbf{v}_h^\ell) \,\mathrm{d}s}_{=: I_6} \\
& + \underbrace{\int_{\Gamma_h^n} \nabla_{\Gamma_h} p^n \cdot \mathbf{v}_h \,\mathrm{d}s_h - \int_{\Gamma^n} \nabla_\Gamma p^n \cdot \mathbf{v}_h^\ell \,\mathrm{d}s}_{=: I_7}.
\end{aligned} \tag{5.4.9}$$

We use the notation $I_3 := I_{3(1)} + I_{3(2)}$.

Using this definition, we observe that the smooth solution $p^n = p(t_n)$, $\mathbf{u}^n = \mathbf{u}(t_n)$, $\mathbf{u}^{n-1} = \mathbf{u}(t_{n-1}) = \mathbf{u}(t_{n-1})^e$ of (5.2.3) satisfies the identity

$$\begin{aligned}
& \int_{\Gamma_h^n} \left(\frac{\mathbf{u}^n - \mathbf{u}^{n-1}}{\Delta t}\right) \cdot \mathbf{P}_h \mathbf{v}_h \, ds + a^n(\mathbf{u}^{n-1}; \mathbf{u}^n, \mathbf{v}_h) + b^n(p^n, \mathbf{v}_h) \\
& = \mathcal{E}_C^n(\mathbf{v}_h) + \int_{\Gamma^n} \mathbf{f}_t^n \cdot \mathbf{v}_h^\ell \,\mathrm{d}s,
\end{aligned} \tag{5.4.10}$$

for any $\mathbf{v}_h \in \mathbf{U}_h^n$.

In Lemma B.1 in the Appendix, we collect error bounds for the terms $I_1$, $I_2$, $I_4$, $I_6$, and $I_7$ known from the literature. Error bounds for the terms $I_3$ and $I_5$ are derived in Lemma B.2. Combining these results and equation (5.4.5) we obtain the following corollary.

**Corollary 5.4.7**

For the consistency error, the uniform estimate

$$|\mathcal{E}_C^n(\mathbf{v}_h)| \lesssim (\Delta t + h^{k_g})\|\!|\mathbf{v}_h|\!\|_n \quad \text{for all } \mathbf{v}_h \in \mathbf{U}_h^n$$

holds.

### 5.4.5 Error estimates

In this section, we derive a bound for the velocity error and the pressure error defined by

$$\mathbb{E}_{\mathbf{u}}^n := \mathbf{u}^n - \mathbf{u}_h^n \in H^1(\Omega_\delta^n) \quad \text{and} \quad \mathbb{E}_p^n := p^n - p_h^n \in H^1(\Omega_\Gamma^n).$$

Let $\mathbf{u}_h^0 = \mathbf{u}_I^0 \in \mathbf{U}_h^0$ be a suitable interpolant to $\mathbf{u}^0 \in \mathcal{O}(\Gamma_h^0)$. From (5.4.2) and (5.4.10), we get the error equation for arbitrary $\mathbf{v}_h \in \mathbf{U}_h^n$. It is given by

$$\begin{aligned} &\frac{1}{\Delta t}(\mathbb{E}_{\mathbf{u}}^n - \mathbb{E}_{\mathbf{u}}^{n-1}, \mathbf{P}_h\mathbf{v}_h)_{\Gamma_h^n} + c_n(\mathbb{E}_{\mathbf{u}}^{n-1}; \mathbf{u}^n, \mathbf{v}_h) + a^n(\mathbf{u}_h^{n-1}; \mathbb{E}_{\mathbf{u}}^n, \mathbf{v}_h) + b^n(\mathbb{E}_p^n, \mathbf{v}_h) \\ &= \mathcal{E}_C^n(\mathbf{v}_h) + \mathcal{E}_f^n(\mathbf{v}_h), \end{aligned} \tag{5.4.11}$$

with the data error term

$$\mathcal{E}_f^n(\mathbf{v}_h) := \int_{\Gamma^n} \mathbf{f}_t^n \cdot \mathbf{v}_h^\ell \,\mathrm{d}s - \int_{\Gamma_h^n} (\mathbf{f}_t^n \mathbf{P}_h) \cdot \mathbf{v}_h^\ell \,\mathrm{d}s_h.$$

We proceed by splitting the errors $\mathbb{E}_{\mathbf{u}}^n$ and $\mathbb{E}_p^n$ into approximation and finite element parts. Let $\mathbf{u}_I^n \in \mathbf{U}_h^n$ and $p_I^n \in Q_h^n$ be the nodal interpolants for $\mathbf{u}^n$ in $\Omega_\delta^n$ and $p^n$ in $\Omega_\Gamma^n$, respectively. The splittings of the errors are given by

$$\mathbb{E}_{\mathbf{u}}^n = \underbrace{(\mathbf{u}^n - \mathbf{u}_I^n)}_{=:\,\mathbf{e}_{\mathbf{u},I}^n} + \underbrace{(\mathbf{u}_I^n - \mathbf{u}_h^n)}_{=:\,\mathbf{e}_{\mathbf{u},h}^n} \quad \text{and} \quad \mathbb{E}_p^n = \underbrace{(p^n - p_I^n)}_{=:\,\mathbf{e}_{p,I}^n} + \underbrace{(p_I^n - p_h^n)}_{=:\,\mathbf{e}_{p,h}^n}.$$

Using $a^n(\mathbf{u}_h^{n-1}; \mathbb{E}_{\mathbf{u}}^n, \mathbf{v}_h) = \widehat{a}^n(\mathbf{e}_{\mathbf{u},I}^n, \mathbf{v}_h) + \widehat{a}^n(\mathbf{e}_{\mathbf{u},h}^n, \mathbf{v}_h) + c_n(\mathbf{u}_h^{n-1}; \mathbb{E}_{\mathbf{u}}^n, \mathbf{v}_h)$ equation (5.4.11) can be reformulated as

$$\begin{aligned} &\frac{1}{\Delta t}(\mathbf{e}_{\mathbf{u},h}^n - \mathbf{e}_{\mathbf{u},h}^{n-1}, \mathbf{P}_h\mathbf{v}_h)_{\Gamma_h^n} + \widehat{a}^n(\mathbf{e}_{\mathbf{u},h}^n, \mathbf{v}_h) + b^n(p_I^n - p_h^n, \mathbf{v}_h) \\ &= \mathcal{E}_C^n(\mathbf{v}_h) + \mathcal{E}_I^n(\mathbf{v}_h) + \mathcal{C}^n(\mathbf{v}_h) + \mathcal{E}_f^n(\mathbf{v}_h), \end{aligned} \tag{5.4.12}$$

with the interpolation and nonlinear terms given by

$$\begin{aligned} \mathcal{E}_I^n(\mathbf{v}_h) &:= -\frac{1}{\Delta t}(\mathbf{e}_{\mathbf{u},I}^n - \mathbf{e}_{\mathbf{u},I}^{n-1}, \mathbf{P}_h\mathbf{v}_h)_{\Gamma_h^n} - \widehat{a}^n(\mathbf{e}_{\mathbf{u},I}^n, \mathbf{v}_h) - b^n(\mathbf{e}_{p,I}^n, \mathbf{v}_h), \\ \mathcal{C}^n(\mathbf{v}_h) &:= -c_n(\mathbb{E}_{\mathbf{u}}^{n-1}; \mathbf{u}^n, \mathbf{v}_h) - c_n(\mathbf{u}_h^{n-1}; \mathbb{E}_{\mathbf{u}}^n, \mathbf{v}_h). \end{aligned}$$

Estimates for these terms are given in [ORS24, Lemma 4.12 and 4.13].

Now, we show the main result of the error analysis. Again, we only show the proof for the velocity error. The proof of the pressure error estimate is based on the proof of the stability result for the pressure. It can be found in [ORS24]. We assume the inequalities

in [ORS24, Assumption 4.1] to hold.

**Theorem 5.4.8**

Let $(\mathbf{u}, p)$ be the solution of (5.2.3) and $\mathbf{u}_h^n$, $p_h^n$, $n = 1, \dots, N$, be the finite element solution of (5.4.2). It holds

$$\|\mathbb{E}_{\mathbf{u}}^n\|_{\Gamma_h^n}^2 + \frac{1}{8}\Delta t \sum_{k=1}^{n} \left( |\!|\!| \mathbb{E}_{\mathbf{u}}^k |\!|\!|_k^2 + \rho_p \|\mathbf{n}_h \cdot \nabla \mathbb{E}_p^k\|_{\Omega_\Gamma^n}^2 \right) \lesssim \exp(c\,t_n)\left(\Delta t^2 + h^{2(m+1)} + h^{2k_g}\right), \tag{5.4.13}$$

$$\Delta t \sum_{k=1}^{n} |\!|\!| \mathbb{E}_p^k |\!|\!|_n \lesssim \exp(c\,t_n)\left(\Delta t + h^{m+1} + h^{k_g}\right), \tag{5.4.14}$$

with $c$ depending on the problem data and independent of $h$, $\Delta t$, $n$, and the positions of the surface in the background mesh.

**Proof**

We only outline the proof of the error bound for the discrete velocity. For details, see [ORS24]. The arguments we use largely repeat those used to show the stability result in Theorem 5.4.5 and involve estimates from Corollary 5.4.7 to bound the arising right-hand side terms. Setting $\mathbf{v}_h = \Delta t\, \mathbf{e}_{\mathbf{u},h}^n$ in (5.4.12) and using Lemma 5.4.3 yields

$$\begin{aligned}
&\|\mathbf{P}_h \mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 + \|\mathbf{P}_h \mathbf{e}_{\mathbf{u},h}^n - \mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^n}^2 + 2\Delta t |\!|\!| \mathbf{e}_{\mathbf{u},h}^n |\!|\!|_n^2 + 2\Delta t\, b^n(\mathbf{e}_{p,h}^n, \mathbf{e}_{\mathbf{u},h}^n) \\
&\leq \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^n}^2 + c\Delta t \|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 + 2\Delta t \left( |\mathcal{E}_C^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{E}_I^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{C}^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{E}_f^n(\mathbf{e}_{\mathbf{u},h}^n)| \right).
\end{aligned}$$

As in (5.4.8) we get

$$\|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 \leq \|\mathbf{P}_h \mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 + \frac{1}{2}\Delta t |\!|\!| \mathbf{e}_{\mathbf{u},h}^n |\!|\!|_n^2 + c\Delta t\, h^{2k_g} \|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2.$$

Using this inequality, Lemma 5.4.2, and some calculations, one gets

$$\begin{aligned}
&\|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 + \|\mathbf{P}_h \mathbf{e}_{\mathbf{u},h}^n - \mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^n}^2 + \frac{3}{2}\Delta t |\!|\!| \mathbf{e}_{\mathbf{u},h}^n |\!|\!|_n^2 + 2\Delta t \rho_p \|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2 \\
&\leq \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^{n-1}}^2 + c\Delta t \left( \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^{n-1}}^2 + \|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 \right) + \frac{1}{2}\Delta t |\!|\!| \mathbf{e}_{\mathbf{u},h}^{n-1} |\!|\!|_{n-1}^2 \\
&\quad + 2\Delta t \left( |\mathcal{E}_I^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{E}_C^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{C}^n(\mathbf{e}_{\mathbf{u},h}^n)| \right) \\
&\quad + 2\Delta t \left( |\mathcal{E}_f^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{E}_s(\mathbf{e}_{p,h}^n)| + |b^n(\mathbf{e}_{p,h}^n, \mathbf{u}_I^n)| \right),
\end{aligned} \tag{5.4.15}$$

with $\mathcal{E}_s(\mathbf{e}_{p,h}^n) := \rho_p \left( \nabla p_I^n \cdot \mathbf{n}_h, -\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h \right)_{\Omega_\Gamma^n}$. We estimate the terms in the last line of (5.4.15). One can show the following bounds, cf. [ORS24].

The pressure error terms satisfy

$$\begin{aligned}
|\mathcal{E}_s(\mathbf{e}_{p,h}^n)| &\lesssim (h^{2m+2} + h^{2k_g+1}) + \frac{1}{2}\rho_p \|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2, \\
|b^n(\mathbf{e}_{p,h}^n, \mathbf{u}_I^n)| &\lesssim h^{m+1} |\!|\!| \mathbf{e}_{p,h}^n |\!|\!|_n + h^{k_g} |\!|\!| \mathbf{e}_{p,h}^n |\!|\!|_n.
\end{aligned}$$

and the velocity error terms satisfy

$$
\begin{aligned}
|\mathcal{E}_I^n(\mathbf{e}_{\mathbf{u},h}^n)| &\lesssim (h^{2m+2} + h^{2k_g}) + \frac{1}{16}|\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n^2,\\
|\mathcal{E}_C^n(\mathbf{e}_{\mathbf{u},h}^n)| &\lesssim (\Delta t^2 + h^{2k_g}) + \frac{1}{16}|\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n^2,\\
|\mathcal{C}^n(\mathbf{e}_{\mathbf{u},h}^n)| &\lesssim \left(h^{2m+2}(1 + |\!|\!|\mathbf{u}_h^{n-1}|\!|\!|_{n-1}) + \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^{n-1}}^2 + h^2|\!|\!|\mathbf{e}_{\mathbf{u},h}^{n-1}|\!|\!|_{n-1}^2\right) + \frac{1}{16}|\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n^2,\\
|\mathcal{E}_f^n(\mathbf{e}_{\mathbf{u},h}^n)| &\lesssim h^{2k_g} + \frac{1}{2}\|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2,
\end{aligned}
$$

Collecting these estimates, we obtain an upper bound (up to a constant $\tilde{C}$) of the last line in (5.4.15) by

$$
\begin{aligned}
&|\mathcal{E}_I^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{E}_C^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{C}^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{E}_f^n(\mathbf{e}_{\mathbf{u},h}^n)| + |\mathcal{E}_s(\mathbf{e}_{p,h}^n)| + |b^n(\mathbf{e}_{p,h}^n, \mathbf{u}_I^n)|\\
\leq\ & \frac{3}{16}|\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n^2 + \frac{1}{2}\rho_p\|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2 + \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^{n-1}}^2 + Ch^2|\!|\!|\mathbf{e}_{\mathbf{u},h}^{n-1}|\!|\!|_{n-1}^2 + \Delta t^2 + h^{2k_g}\\
&+ h^{2m+2}(1 + |\!|\!|\mathbf{u}_h^{n-1}|\!|\!|_{n-1}) + (h^{m+1} + h^{k_g})|\!|\!|\mathbf{e}_{p,h}^n|\!|\!|_n.
\end{aligned}
$$

The first two terms in this expression can be shifted to the left in (5.4.15). Assuming $h$ is sufficiently small, we get

$$
\begin{aligned}
&\|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 + \|\mathbf{P}_h\mathbf{e}_{\mathbf{u},h}^n - \mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^n}^2 + \Delta t|\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n^2 + \Delta t\rho_p\|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2\\
\leq\ & \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^{n-1}}^2 + c\Delta t\big(\|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^{n-1}}^2 + \|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2\big) + \frac{5}{8}\Delta t|\!|\!|\mathbf{e}_{\mathbf{u},h}^{n-1}|\!|\!|_{n-1}^2 \qquad (5.4.16)\\
&+ C\Delta t\Big(\Delta t^2 + h^{2k_g} + h^{2m+2}(1 + |\!|\!|\mathbf{u}_h^{n-1}|\!|\!|_{n-1}) + (h^{m+1} + h^{k_g})|\!|\!|\mathbf{e}_{p,h}^n|\!|\!|_n\Big).
\end{aligned}
$$

We need an upper bound for $|\!|\!|\mathbf{e}_{p,h}^n|\!|\!|_n$. For this, we use the same approach as in the proof of Theorem 5.4.5, which is not included in this thesis. See [ORS24] for details. This then yields

$$
\begin{aligned}
|\!|\!|\mathbf{e}_{p,h}^n|\!|\!|_n \lesssim\ & \frac{h}{\Delta t}\|\mathbf{P}_h\mathbf{e}_{\mathbf{u},h}^n - \mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^n} + \rho_p^{\frac{1}{2}}\|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n} + |\!|\!|\mathbf{e}_{\mathbf{u},h}^{n-1}|\!|\!|_{n-1}^2 + |\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n^2\\
&+ |\!|\!|\mathbf{e}_{\mathbf{u},h}^{n-1}|\!|\!|_n + |\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n + \Delta t + h^{m+1} + h^{k_g} + |\!|\!|\mathbf{u}_h^{n-1}|\!|\!|_{n-1}^{\frac{1}{2}}(h^{m+1} + |\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n).
\end{aligned}
$$

Using this, we obtain that the last term in (5.4.16) satisfies

$$
\begin{aligned}
C\Delta t(h^{m+1} + h^{k_g})|\!|\!|\mathbf{e}_{p,h}^n|\!|\!|_n \leq\ & \frac{1}{2}\|\mathbf{P}_h\mathbf{e}_{\mathbf{u},h}^n - \mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^n}^2 + \frac{1}{2}\Delta t\rho_p\|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2\\
&+ \frac{1}{8}\Delta t\left(|\!|\!|\mathbf{e}_{\mathbf{u},h}^{n-1}|\!|\!|_{n-1}^2 + |\!|\!|\mathbf{e}_{\mathbf{u},h}^n|\!|\!|_n^2\right) + C\Delta t\left(\Delta t^2 + \left(h^{2k_g} + h^{2m+2}\right)\left(1 + |\!|\!|\mathbf{u}_h^{n-1}|\!|\!|_{n-1}\right)\right)
\end{aligned}
$$

for $h$ sufficiently small. Substituting this in (5.4.16) and shifting terms from the right to

the left-hand side, we get

$$\begin{aligned}
&\|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 + \frac{7}{8}\Delta t |\!|\!| \mathbf{e}_{\mathbf{u},h}^n |\!|\!|_n^2 + \frac{1}{2}\Delta t \rho_p \|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2 \\
&\leq \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^{n-1}}^2 + c_1 \Delta t \big( \|\mathbf{e}_{\mathbf{u},h}^{n-1}\|_{\Gamma_h^n}^2 + \|\mathbf{e}_{\mathbf{u},h}^n\|_{\Gamma_h^n}^2 \big) + \frac{6}{8}\Delta t |\!|\!| \mathbf{e}_{\mathbf{u},h}^{n-1} |\!|\!|_{n-1}^2 \\
&\qquad + c_2 \Delta t \left( \Delta t^2 + \left( h^{2k_g} + h^{2m+2} \right) \left( 1 + |\!|\!| \mathbf{u}_h^{n-1} |\!|\!|_{n-1} \right) \right).
\end{aligned}$$

We sum over $n = 1, \dots, k$, apply a discrete Gronwall inequality, and use the discrete stability estimate $\Delta t \sum_{n=1}^k |\!|\!| \mathbf{u}_h^{n-1} |\!|\!|_{n-1} \leq C$ to get

$$Q_{e,k} := \|\mathbf{e}_{\mathbf{u},h}^k\|_{\Gamma_h^k}^2 + \frac{1}{8}\Delta t \sum_{n=1}^k \left( |\!|\!| \mathbf{e}_{\mathbf{u},h}^n |\!|\!|_n^2 + \rho_p \|\nabla \mathbf{e}_{p,h}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2 \right) \lesssim \exp(c\, t_k)(\Delta t^2 + h^{2m+2} + h^{2k_g}).$$

The triangle inequality and standard FE interpolation properties give

$$\begin{aligned}
&\|\mathbb{E}_{\mathbf{u}}^k\|_{\Gamma_h^k}^2 + \frac{1}{8}\Delta t \sum_{n=1}^k \left( |\!|\!| \mathbb{E}_{\mathbf{u}}^n |\!|\!|_n^2 + \rho_p \|\nabla \mathbb{E}_p^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2 \right) \\
&\qquad \leq 2Q_{e,k} + 2\|\mathbf{e}_{\mathbf{u},h}^k\|_{\Gamma_h^k}^2 + \frac{1}{4}\Delta t \sum_{n=1}^k \left( |\!|\!| \mathbf{e}_{\mathbf{u},h}^n |\!|\!|_n^2 + \rho_p \|\nabla \mathbf{e}_{p,I}^n \cdot \mathbf{n}_h\|_{\Omega_\Gamma^n}^2 \right) \lesssim Q_{e,k} + C h^{2m+2}.
\end{aligned}$$

From this, we obtain the velocity error estimate. □

Both bounds in Theorem 5.4.8 are optimal in terms of convergence order in $\Delta t$ and $h$.

## 5.5 Numerical experiments

In this section, we present results of numerical experiments designed to validate our method. We consider three geometry evolutions. The first one is a simple evolution, namely that of a slowly moving sphere which does not change its shape. In the second one, we start with an ellipsoid that deforms into a sphere. The last example is a so-called Kite geometry with high curvature that moves slowly. In all the experiments, we take the time interval $I = [0, 1]$ and the evolving surfaces are embedded in the domain $\Omega = [-\frac{7}{3}, \frac{7}{3}]^3$. Using Maple [Map], we calculate corresponding exact force terms $\mathbf{f}_t$ and $f_{\text{div}}$ to our manufactured solutions. The viscosity is set to $\mu_0 = 0.05$.

The method introduced in this chapter is implemented in `Netgen/NGSolve` [Sch97; Sch14] with the add-on `ngsxfem` [Leh+]. For each example, an exact level set function $\phi$ is known and a discrete level set function $\phi_h$ is constructed based on an Oswald type quasi-interpolation $\Pi_h^{k_g}$ in the finite element space $V_{h,k_g}$ (cf. (4.1.2)). The used discrete geometric quantities are defined by

$$\phi_h^n = \Pi_h^{k_g}\left(\phi(\cdot, t_n)\right), \quad \tilde{\phi}_h^n = \Pi_h^{k_g+1}\left(\phi(\cdot, t_n)\right), \quad \mathbf{n}_h^n := \frac{\nabla \phi_h^n}{\|\nabla \phi_h^n\|}, \quad \widetilde{\mathbf{n}}_h^n := \frac{\nabla \tilde{\phi}_h^n}{\|\nabla \phi_h^n\|},$$
$$\text{and} \quad \mathbf{H}_h^n = \nabla_{\Gamma_h} \Pi_h^{k_g - 1} \mathbf{n}_h^n.$$

The parameter choices we use are consistent with (5.3.9). For the stabilization and the penalization, we take $\rho_{\mathbf{u}} = h^{-1}$, $\rho_p = h$ and $\tau = h^{-2}$ with $h$ the corresponding mesh size. As described in Chapter 4, the thickness of the narrow band $\Omega_\delta$ has to be chosen large

enough such that $\Omega_\Gamma^n \subset \Omega_\delta^{n-i}$ holds for $i = 1, \dots, R$, where $R$ is the order of the used BDF–scheme, cf. (5.3.10). A possible choice is $\delta_n = c_\delta R \Delta t \|w_N\|_{L^\infty(\Omega \times I_n)}$, cf. (4.3.3). We approximate $\|w_N\|_{L^\infty(\Omega\times I_n)}$ by the maximum of the absolute values of the linear interpolation of $w_N^{n+1}$ at the nodes of the cut elements $\Omega_\Gamma^n$. In our implementation, we check in each step whether $\delta_n$ is sufficiently large such that $\Omega_\Gamma^n \subset \Omega_\delta^{n-i}$, $i = 1, \dots, R$, holds. If not, we stop the calculation.

We use Taylor–Hood pairs (cf. (5.3.5)), of different orders, i.e., $m = 1, 2$, and consider $k_g = 1, 2, 3$. For the case $k_g > 1$, the zero level of $\phi_h^n$ is not easy to determine, and we use the parametric finite element technique outlined in Section 4.1. We start with a uniform tetrahedral triangulation of $\Omega$ with maximum mesh size $h_0 = 0.5$. In each refinement step, the mesh size is halved. For the time discretization, we use the BDF(R) schemes with $R = 1, \dots 4$. As initial time step size, we chose $\Delta t_0 = 0.2$. In each spatial refinement step we halve the time step size in the higher order BDF-schemes, and we divide it by four when the BDF(1) scheme is used.

For the non linear term $\nabla_{\Gamma_h} \mathbf{u}_h^n \mathbf{u}_h^n$ we use a linearization of the form $\nabla_{\Gamma_h} \mathbf{u}_h^n \mathbf{u}_h^n \approx \nabla \mathbf{u}_h^n \widetilde{\mathbf{u}_h}^n$, where $\widetilde{\mathbf{u}_h}^n$ is a linear combination of the previous velocity approximations. In case of the BDF(1) scheme, we use the first order accurate linearization $(\nabla_{\Gamma_h} \mathbf{u}^n)\mathbf{u}_h^{n-1}$, cf. Section 5.3. For higher order BDF schemes, we consider higher order accurate linearizations given by

$$\begin{aligned} &\text{BDF(2):} && \widetilde{\mathbf{u}_h}^n := 2\mathbf{u}_h^{n-1} - \mathbf{u}_h^{n-2}, \\ &\text{BDF(3):} && \widetilde{\mathbf{u}_h}^n := 3\mathbf{u}_h^{n-1} - 3\mathbf{u}_h^{n-2} + \mathbf{u}_h^{n-3}, \\ &\text{BDF(4):} && \widetilde{\mathbf{u}_h}^n := 4\mathbf{u}_h^{n-1} - 6\mathbf{u}_h^{n-2} + 4\mathbf{u}_h^{n-3} - \mathbf{u}_h^{n-4}. \end{aligned} \tag{5.5.1}$$

The error quantities we use are defined as follows. The velocity errors and pressure errors in each time step are denoted by $\mathbb{E}_{\mathbf{u}}^n = \mathbf{u}^n - \mathbf{u}_h^n$ and $\mathbb{E}_p^n = p^n - p_h^n$ for $n = 1, \dots, N$. The energy norms are given by

$$\begin{aligned} |\!|\!| \mathbf{v} |\!|\!|_n^2 &= \frac{1}{2}\|\mathbf{v}\|_{\Gamma_h^n}^2 + 2\mu_0 \|E_{\Gamma_h}(\mathbf{P}_h \mathbf{v})\|_{\Gamma_h^n}^2 + \frac{\tau}{2}\|\widetilde{\mathbf{n}}_h \cdot \mathbf{v}\|_{\Gamma_h^n}^2 + \rho_{\mathbf{u}} \|\nabla \mathbf{v} \mathbf{n}_h\|_{\Omega_\delta^n}^2, \\ |\!|\!| q |\!|\!|_n &= h^{\frac{1}{2}} \|\nabla q\|_{\Omega_\Gamma^n}. \end{aligned}$$

Note that the norm $|\!|\!| \cdot |\!|\!|_n$ for the pressure is closely related (cf. discussion below (5.4.3)) to the *scaled* $H^1(\Gamma_h^n)$ norm $h\|\nabla \cdot\|_{\Gamma_h^n}$, consistent with the error term $h^{m+1}$ in (5.4.14). This error is only of interest, when the exact pressure function is extended constantly in normal direction off the surface. It is hard to realize manufactured solutions that have this property. We will only measure the error in the $|\!|\!| \cdot |\!|\!|_n$-norm in those experiments, where an exact signed distance function is available and the manufactured solution can be extended using the signed distance function.

In the velocity energy error, the term $\|\nabla(\mathbf{u}_h - \mathbf{u})\mathbf{n}_h\|_{\Omega_\delta^n}$ occurs. This term is also only reasonable, when the exact solution is extended constant along normals, i.e., it holds $\nabla \mathbf{u}\mathbf{n} = 0$ on $\Omega_\delta^n$. Since the construction of an exact solution with this property is difficult when no signed distance function is available, we approximate the error term by $\|\nabla(\mathbf{u}_h - \mathbf{u})\mathbf{n}_h\|_{\Omega_\delta^n} \approx \|\nabla \mathbf{u}_h \mathbf{n}_h\|_{\Omega_\delta^n}$. This is a reasonable approximation, since a (sufficient smooth) exact solution that is constant in normal direction, satisfies

$$\begin{aligned} \|\nabla(\mathbf{u}_h - \mathbf{u})\mathbf{n}_h\| &\le \|\nabla \mathbf{u}_h \mathbf{n}_h\| + \|\nabla \mathbf{u}\mathbf{n}_h\| \le \|\nabla \mathbf{u}_h \mathbf{n}_h\| + \underbrace{\|\nabla \mathbf{u}\mathbf{n}\|}_{=0} + \|\nabla \mathbf{u}(\mathbf{n} - \mathbf{n}_h)\| \\ &\le \|\nabla \mathbf{u}_h \mathbf{n}_h\| + \mathcal{O}(h^{k_g}). \end{aligned}$$

In our experiments, we measure the following error quantities

$$\left(e_{\mathbf{u},2}\right)^2 := \Delta t \sum_{n=1}^{N} \|\mathbb{E}_{\mathbf{u}}^n\|_{\Gamma_h^n}^2, \qquad \left(e_{\mathbf{u},e}\right)^2 := \Delta t \sum_{n=1}^{N} |||\mathbb{E}_{\mathbf{u}}^n|||_n^2,$$
$$\left(e_{p,2}\right)^2 := \Delta t \sum_{n=1}^{N} \|\mathbb{E}_{p}^n\|_{\Gamma_h^n}^2, \qquad e_{p,e} := \Delta t \sum_{n=1}^{N} |||\mathbb{E}_{p}^n|||_n,$$

with $\|\cdot\|_{\Gamma_h^n}^2 = \|\cdot\|_{L^2(\Gamma_h^n)}^2$. The pressure $p$ is only defined up to a constant by the TSNS equations. For that reason, we scale the pressure such that $\int_{\Gamma_h} p_h \,\mathrm{d}s \approx \int_{\Gamma_h} p \,\mathrm{d}s \approx 0$ holds. This scaling is done by subtracting the mean value of the pressure function from itself. The error $e_{p,e}$ is then calculated by using the scaled pressure functions $p_h$ and $p$.

### 5.5.1 Surface with constant curvature

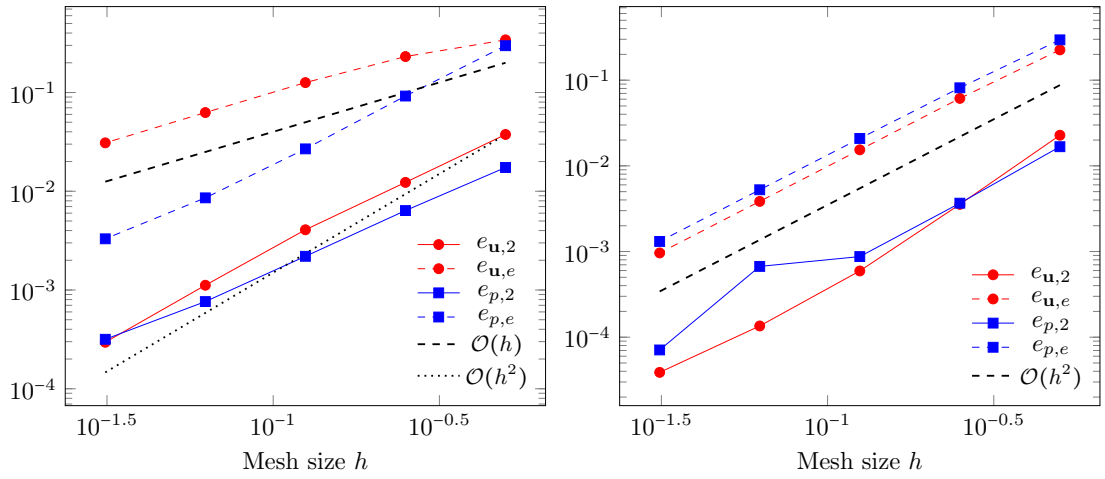
In the first example, we consider a sphere slowly moving to the right. The exact level set function is given by

$$\phi(x,y,z,t) := (x + 0.1 - 0.2t)^2 + y^2 + z^2 - 1$$

We choose a tangential divergence free velocity $\widetilde{\mathbf{u}}_T = \mathbf{P}(\mathbf{n}\times\nabla_\Gamma\psi)$ with the stream function $\psi = xy - 2t$. This function is added to the constant velocity $\mathbf{u}_c = (0.2, 0, 0)^T$ that fits to the desired evolution of the surface $\Gamma(t)$. The full velocity is given by $\mathbf{u} = \widetilde{\mathbf{u}}_T + \mathbf{u}_c$, holding $\mathrm{div}_\Gamma\, \mathbf{u} = 0$. The pressure is taken as $p = (x - 0.2t)yz$, which satisfies $\int_{\Gamma(t)} p \,\mathrm{d}s = 0$ for $t \in [0, 1]$. For the width of the extension domain, we take $c_\delta = 2.5$.

#### Low order Taylor–Hood, BDF(1)

We start with results for the BDF(1) method combined with Taylor–Hood with $m = 1$, thus quadratic polynomials for the velocity and linear ones for the pressure, and $k_g = 1$ and $k_g = 2$. The results are presented in Figure 5.5.1.




(a) $k_g = 1$ (b) $k_g = 2$

Figure 5.5.1: Errors of TSNS approximations on a sphere with $m = 1$, BDF(1), and $\Delta t \sim h^2$.

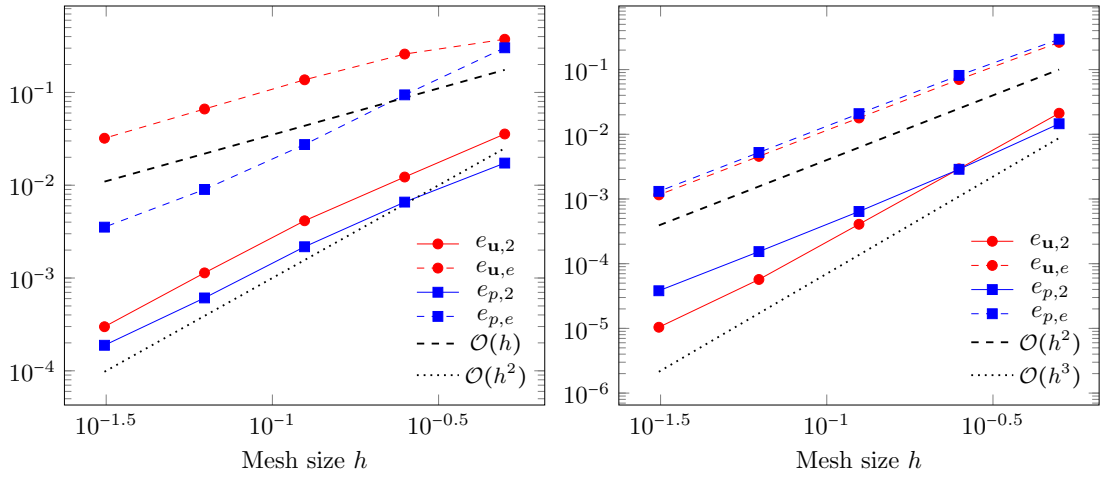
The errors in Figure 5.5.1(a) are derived by using linear polynomials for the level set function ($k_g = 1$). The first order convergence for the error $e_{\mathbf{u},e}$ shows that the geometric

error bound $h^{k_g}$ for this term in (5.4.13) is sharp. The slightly better convergence rate for the pressure error $e_{p,e}$ indicates that the geometric error term $h^{k_g}$ in (5.4.14) might be not sharp. But it is also possible that the error behavior is pre-asymptotic. The results in Figure 5.5.1(b) for $e_{\mathbf{u},e}$ and $e_{p,e}$ where $k_g = 2$ is used, confirm the second order convergence predicted by Theorem 5.4.8.

Although we did not derive a bound for the $L^2$–errors $e_{\mathbf{u},2}$ and $e_{p,2}$ in the analysis, these errors have the expected (approximately) second order convergence in Figure 5.5.1 both for $k_g = 1$ and $k_g = 2$. Due to use of a BDF(1) method and the scaling $\Delta t \sim h^2$, this error is optimal in the time discretization and suboptimal for the space discretization for the velocity. The $L^2$-error of the pressure is optimal in time and in space, since linear polynomials are used to discretize the pressure function.

**Low order Taylor–Hood, BDF(2) and BDF(3)**

We claim that our theoretical analysis from Theorem 5.4.8 can be extended to higher order time discretizations by using BDF(R) schemes with $R > 1$. Then, the time discretization error terms $\Delta t$ in (5.4.13)-(5.4.14) would be replaced by $\Delta t^R$. Results for the BDF(2) method are shown in Figure 5.5.2 for a piecewise linear level set function ($k_g = 1$) and piecewise quadratic level set function ($k_g = 2$) and in Figure 5.5.3 for the BDF(3) method combined with $k_g = 2$.




(a) $k_g = 1$ (b) $k_g = 2$

Figure 5.5.2: Errors of TSNS approximations on a sphere with $m = 1$ and BDF(2).

The asymptotic behavior for the errors $e_{\mathbf{u},e}$ and $e_{p,e}$ equals the behavior of the errors when the BDF(1) method is used with $\Delta t \sim h^2$, cf. Figure 5.5.1. These errors are optimal and in accordance with our expectation, that the theory (Theorem 5.4.8) also holds true for higher order time discretizations. The $L^2$-errors for the pressure and the velocity are slightly better in the BDF(2) case than in the BDF(1) case. For example, the error $e_{\mathbf{u},2}$ shows almost third order convergence, which is optimal considering the polynomials used in space and better than expected concerning the used time discretization.

By employing the BDF(3) time integration with $m = 1$ and $k_g = 2$, we achieve optimal convergence rates of $e_{\mathbf{u},e} \sim e_{p,e} \sim \mathcal{O}(h^2)$ and $e_{\mathbf{u},2} \sim \mathcal{O}(h^3)$. Additionally, the $L^2$–pressure error $e_{p,2}$ is slightly better than the optimal bound of $\mathcal{O}(h^2)$.

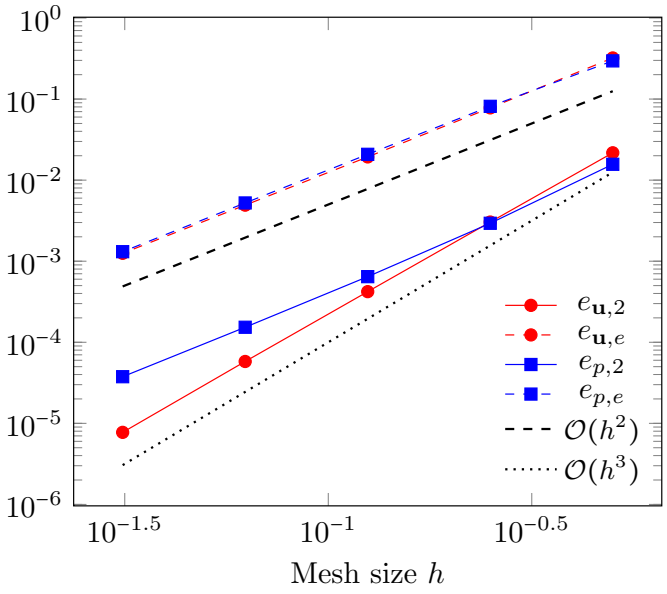
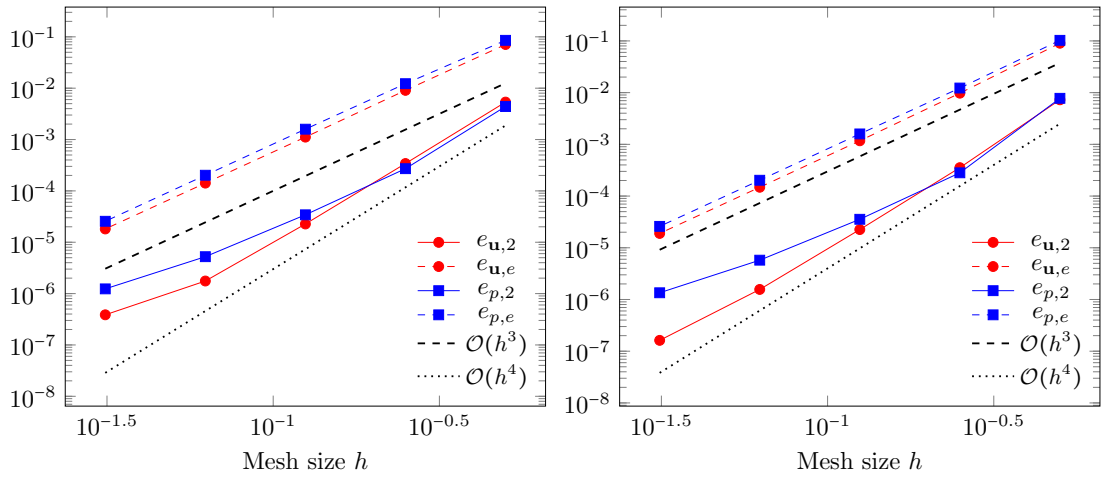



Figure 5.5.3: Errors of TSNS approximations on a sphere with $m = 1$, BDF(3), and $k_g = 2$.

### Higher order Taylor–Hood, BDF(3) and BDF(4)

We also present results obtained by using higher order Taylor–Hood spaces, specifically with $m = 2$ in combination with $k_g = 3$. According to Theorem 5.4.8, our method converges at the optimal rate of $\mathcal{O}(h^3)$ for the errors $e_{\mathbf{u},e}$ and $e_{p,e}$. The optimal convergence rates related to the spatial discretization in the $L^2$–norm are given by $e_{p,2} \sim \mathcal{O}(h^3)$ and $e_{\mathbf{u},2} \sim \mathcal{O}(h^4)$.

These optimal error convergence rates can be observed in Figure 5.5.4(b), where the BDF(4) method is employed. For the BDF(3) method, we get optimal error rates for the errors $e_{\mathbf{u},e}, e_{p,e}$, and $e_{p,2}$; however, we find slightly suboptimal convergence rates for $e_{\mathbf{u},2}$ due to the third-order time integration, cf. Figure 5.5.4(a).




(a) BDF(3) (b) BDF4

Figure 5.5.4: Errors of TSNS approximations on a sphere with $m = 2$ and $k_g = 2$.

### 5.5.2 Surfaces with varying curvature

In the next examples, we consider surfaces with varying curvature, namely an ellipsoid deforming into a sphere and a Kite geometry moving to the right with a constant velocity. In both cases, the exact tangential velocity is taken as $\mathbf{u}_T = \mathbf{P}(1-2t,0,0)^T$. The normal component is given by the relation $w_N = -\frac{\partial_t \phi}{\mathbf{n}\cdot\nabla\phi}$, and the pressure is chosen as $p = xyz$. In these examples, the velocity is not (surface) divergence free, so we change the second equation of the continuous problem (5.2.3) to $\operatorname{div}_\Gamma \mathbf{u}_T = f_{\text{div}}$, with $f_{\text{div}} = \operatorname{div}_\Gamma \mathbf{u} - w_N\kappa$ instead of $f_{\text{div}} = -w_N\kappa$. The pressure is not constant in normal direction. For that reason, we do not measure errors in the pressure norms and stick to the energy norm in the velocity. We use $c_\delta = 3$ for the choice of the narrow band thickness. We use Taylor–Hood pairs with $m = 1$ and $m = 2$ and combine them with $k_g = m+1$ and BDF(R) schemes with $R = m+1$. Following the results of the previous example and the theory, we expect optimal convergence rates for the velocity in the energy norm.

#### Ellipsoid to sphere

A level set function for an ellipsoid deforming into a sphere is given by

$$\phi = \left(x^2 + y^2 + \frac{z^2}{0.75^2} - 1\right)(1-t) + \left(x^2+y^2+z^2-1\right)t \quad \text{for} \quad t \in [0,1].$$

The results for $m = 1$ and $m = 2$ are shown in Figure 5.5.5. The optimal convergence rates for the velocity energy error can be observed for both $m = 1$ and $m = 2$ in the BDF(R) schemes with $R = m+1$ and $k_g = m+1$. The results are in accordance with the theoretical predictions and show that our method is also applicable to surfaces with curvature that varies over time.

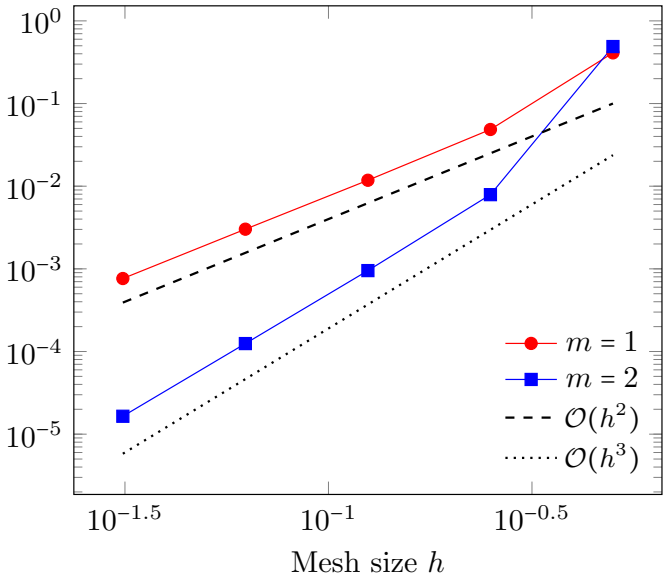



Figure 5.5.5: $e_{\mathbf{u},e}$ errors of TSNS approximations on a deforming ellipsoid for $m \in \{1,2\}$ with BDF$m$ and $k_g = m$.

### Kite to right

The Kite geometry that moves to the right is described by the level set function

$$\phi = ((x + 0.1 - 0.2t) + z^2)^2 + y^2 + z^2 - 1.$$

The surface is taken from [Dzi88]. Figure 5.5.6 shows the shape of the "kite"–like geometry. The surface $\Gamma(t)$ is asymmetric, and its curvature varies. On the left side, the surface exhibits a higher curvature, due to two peaks. On the right-hand side, the surface is smoother, leading to a lower curvature $\kappa$. The geometry is evolving with a constant velocity to the right.

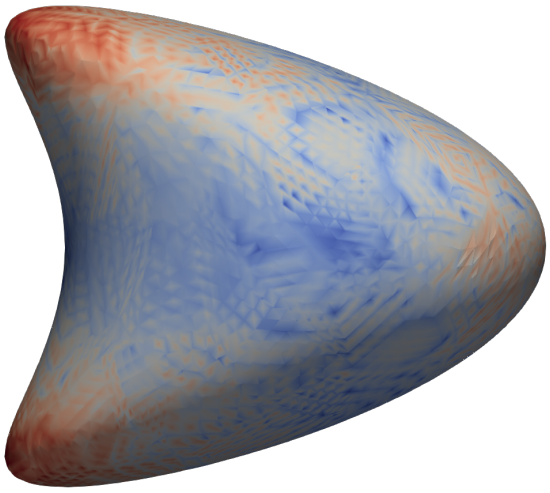


Figure 5.5.6: Errors in $\mathbf{u}_T$ from blue (low) to red (high) at end time point $t = 1$ on the kite geometry for $h = 2^{-4}$, $m = 1$, BDF(2), and $k_g = 2$.

In Figure 5.5.7, the velocity error in the energy norm is presented for $m = 1$ and $m = 2$.

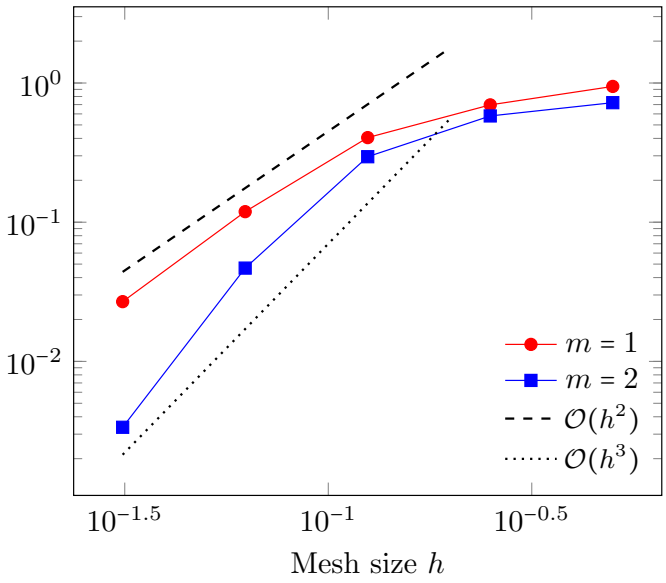



Figure 5.5.7: $e_{\mathbf{u},e}$ errors of TSNS approximations on a deforming kite for $m \in \{1, 2\}$ with BDF$m$ and $k_g = m$.

The convergence observed at coarser mesh sizes is relatively slow, while optimal convergence rates are approximately attained for finer meshes for $m = 1$ and $m = 2$. This phenomenon suggests that the results obtained from the coarse meshes are pre-asymptotic. Note that the errors are much larger than in the previous experiments.

We take a closer look at the errors and the influence of the curvature on the errors. We fix $t = t_{\text{end}} = 1$. In Figure 5.5.6, the error $\|\mathbf{u}_T - \mathbf{u}_h\|$ is shown for $h = 0.0625$ and $m = 1$. We observe higher errors in the tips of the surface on the left side than at the tip on the right side, which has a smaller curvature. In the flat regions, e.g., in the middle of the picture, we observe the smallest errors. This behavior shows the influence of the surface curvature on the quality of the discretization. We also validate the observed error behavior by measuring $\|\mathbf{u}_T - \mathbf{u}_h\|$ on the left and the right side of the surface. For that we sample points on the left and right half of the surface respectively and evaluate the error pointwise. We approximate the $L^\infty$-norm of the error by the maximum of the error at the sampled points for the left and the right half. These errors are displayed in Table 5.5.8 for the different refinement levels.

| | $m = 1$ | | | | $m = 2$ | | | |
|---|---|---|---|---|---|---|---|---|
| $h$ | left | rate | right | rate | left | rate | right | rate |
| $5 \cdot 10^{-1}$ | $6.9 \cdot 10^{-1}$ | – | $1.9 \cdot 10^{-1}$ | – | $6.4 \cdot 10^{-1}$ | – | $1.6 \cdot 10^{-1}$ | – |
| $2.5 \cdot 10^{-1}$ | $4.6 \cdot 10^{-1}$ | 0.56 | $4.7 \cdot 10^{-2}$ | 2 | $3.4 \cdot 10^{-1}$ | 0.92 | $1.8 \cdot 10^{-2}$ | 3.16 |
| $1.3 \cdot 10^{-1}$ | $2.2 \cdot 10^{-1}$ | 1.08 | $9.4 \cdot 10^{-3}$ | 2.31 | $1.8 \cdot 10^{-1}$ | 0.95 | $2.1 \cdot 10^{-3}$ | 3.16 |
| $6.3 \cdot 10^{-2}$ | $5.8 \cdot 10^{-2}$ | 1.93 | $2 \cdot 10^{-3}$ | 2.24 | $2.4 \cdot 10^{-2}$ | 2.88 | $2.3 \cdot 10^{-4}$ | 3.16 |
| $3.1 \cdot 10^{-2}$ | $6.1 \cdot 10^{-3}$ | 3.23 | $3.7 \cdot 10^{-4}$ | 2.43 | $1.4 \cdot 10^{-3}$ | 4.07 | $1.7 \cdot 10^{-5}$ | 3.75 |

Figure 5.5.8: $L^\infty$–errors at the end time point ($\|\mathbf{u}_T - \mathbf{u}_h\|_{L^\infty(\Gamma^N)}$) on different areas of the sphere with $m \in \{1, 2\}$.

We observe that the errors on the left side are higher than on the right side, and the convergence rates are optimal on the right side for the coarse and the fine meshes. The convergence rates for the error on the left side are suboptimal for the coarse meshes and optimal for the fine meshes. This indicates, that the mesh has to be sufficiently fine to resolve the high curvature regions. The results show that the method is applicable to surfaces with varying curvature and that the error behavior is in accordance with the theoretical predictions.

# Chapter 6

# Narrow band method for the level set equation

This chapter is based on the paper [ORS25].

## 6.1 Introduction

In the process of solving the surface Navier–Stokes equations on a *moving* surface $\Gamma(t)$ the geometric evolution of $\Gamma(t)$ depends on the material velocity $\mathbf{u}$ and is itself an unknown in the system. To apply a trace finite element method to solve a PDE posed on the surface, $\Gamma(t)$ has to be represented as the zero level of a level set function. Thus, we need a method that describes the change of the level set function over time depending on the material flow field $\mathbf{u}$.

In this chapter, we introduce a narrow-band transport algorithm to describe the evolution of the surface in terms of the level set function. The surface velocity $\mathbf{u}$ is extended to a velocity field that transports the level set function and is defined in a narrow band around the surface. For a closed hypersurface $\Gamma : [0, t_{\text{end}}] \to \mathbb{R}^d$ moving with speed $V_\Gamma$ in its normal direction, we consider the following *Eulerian* formulation of its evolution in a domain $\Omega \subset \mathbb{R}^d$ such that the surface is contained in $\Omega$ for all times $t \in [0, t_{\text{end}}]$:

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0, \tag{2.6.2}$$

where $\mathbf{u} : \Omega \to \mathbb{R}^d$ and the surface normal velocity are related by $\mathbf{u} \cdot \mathbf{n} = V_\Gamma$ on the zero level set of $\phi$. Here $\mathbf{n}$ denotes the outer unit normal on $\Gamma(t)$. In this chapter, we assume a given smooth velocity $\mathbf{u}$ defined in a sufficiently large neighborhood of the surface.

In this thesis, for the surface Navier–Stokes system, we have $d = 3$, where $d$ is the dimension of the background space in which the surface $\Gamma \subset \mathbb{R}^d$ is embedded. There exists no equivalent system with a one-dimensional surface $\Gamma \subset \mathbb{R}^2$. However, the transport equation (2.6.2) can be applied in 2D and 3D. The method we propose is applicable in both cases. Thus, in this section it holds $d = 2$ or $d = 3$.

There are many numerical approaches in the literature to solve transport equations. In [KMS99], a finite volume scheme for the Hamilton-Jacobi problem which includes a transport equation is proposed. Several papers published by Sethian, e.g., [Set96; A96]

and his monograph [Set99a] describe finite difference-based methods and their analysis. Other authors also use finite difference methods to solve the transport equation, cf. [OF03]. Another numerical scheme is the streamline-upwind Petrov-Galerkin (SUPG) method, where numerical viscosity in the direction of the flow $\mathbf{u}$ is added to the classical Galerkin approach to improve the stability of the method. It was developed in [BH82] and further used and analysed for example in [JNP84].

Also, discontinuous (DG) finite element methods are known to be very suitable for convection-dominated or pure transport problems, cf. e.g., [BMS04; DF15; FŠ04; BQS10]. We use a DG scheme in space that was introduced and analysed in [BDF16]. The given analysis provides an $L^2$–error bound $\mathcal{O}(h^{k_g+\frac{1}{2}})$ for the space semi–discrete scheme, where piecewise polynomials of degree $k_g$ are used. The analysis of that paper applies also if the velocity field is not divergence-free.

For the time discretization, the authors of [BDF16] also use a discontinuous Galerkin method. Alternatively, Runge-Kutta (RK) schemes are used for the time discretization, cf. [MRC06; CS89; CHS90]. In our work, we use a backward differentiation formula (BDF) scheme. The BDF scheme was used for solving transport equations, e.g., in [DF04]. The combination of a discontinuous Galerkin method with a BDF scheme was analysed, e.g., in [SVD06; vS08].

In certain applications, the underlying physics yields a velocity field $\mathbf{u}$ that is defined in the whole domain $\Omega$. A prototypical example from this category is the two-phase immiscible flow problem, where the interface evolution is determined by the continuous flow field associated with the two fluid phases [AS14; GR11]. In these scenarios, it can be computationally advantageous to solve the level set equation (2.6.2) within a narrow band surrounding the interface rather than in the entire domain $\Omega$. In other problem classes, such as the surface Navier–Stokes system (3.4.2) treated in Chapter 3, the surface (normal) velocity is known *only* at the surface, necessitating the construction of an extension velocity $\mathbf{u}$. The governing partial differential equations in this model are formulated on the evolving surface $\Gamma(t)$, with the evolution of the surface geometry being part of the unknown solution. An Eulerian numerical framework for this and similar interface problems, such as geometric flows, typically employs a level set method for implicit surface representation. The required velocity in the level set equation is derived from the PDE posed on the surface. In our context, the velocity is provided by a TraceFEM discretization of the surface PDE, and is not only defined on the surface but also within a narrow band $\Omega_\delta$. Consequently, we focus on developing a finite element solver for (2.6.2) that operates within a thin tube that contains the surface, termed as "narrow band."

These locality requirements and efficiency benefits have driven advancements in narrow band level set methods, e.g., [LDM14; NC17; Xue+21].

In this chapter, we treat a novel finite element narrow band level set method based on a specific extension technique introduced in [ORS25]. For simplicity, we assume that within a narrow band there is a bulk velocity $\mathbf{u}$ available. Narrow band methods for (2.6.2) necessitate an extension procedure. To illustrate why, we assume that at time $t_n$, an approximation $\phi_h^n$ of $\phi(\cdot, t_n)$ within a neighborhood $\Omega^n$ surrounding $\Gamma_h^n \approx \Gamma(t_n)$ is given, and that we know beforehand that $\Gamma_h^{n+1} \subset \Omega^n$; refer to Fig. 6.2.1. Within an Eulerian framework, numerical time integration over $[t_n, t_{n+1}]$ yields an approximation $\phi_h^{n+1}$ of $\phi(\cdot, t_{n+1})$ in $\Omega^n$, which defines $\Gamma_h^{n+1}$. To proceed to the subsequent time step

from $t_{n+1}$ to $t_{n+2}$, one has to extend computed approximations $\phi_h^{n+1}$ on $\Omega^n$ to suitable initial values on $\Omega^{n+1}$, ensuring that $\Gamma_h^{n+2} \subset \Omega^{n+1}$ holds.

In narrow-band methods from the existing literature, such extensions mostly rely upon reinitialization techniques. A prominent example is the fast marching method (FMM), initially described by Sethian [A96], which approximates the distances from points to the given surface. The original FMM from Sethian is first-order accurate. Second-order accurate finite difference adaptations are presented in [Set99b]. A drawback of using re-initialization in level set methods is the difficulty in controlling the change of the surface position. To address this issue several finite difference-based approaches have been introduced [AS99]. Alternative strategies to extend or reinitialize a level set function are based on employing time dependent PDEs generating stationary solutions that satisfy the Eikonal equation, cf. [SF99; TE00]. These methods, however, have certain disadvantages such as parameter selections and substantial computational costs in a narrow band setting. Elliptic-PDE-based approaches [BK13; Xue+21] circumvent temporal progressions towards stationary solutions but require nonlinear problem-solving and the construction of artificial boundary conditions. To the best of our knowledge, none of these techniques from the literature offer rigorous error estimates that would fit the finite element analysis of the entire narrow band discretization method.

In this chapter another approach for the extension of $\phi_h^{n+1}$ on $\Omega_{\mathrm{Tr}}^n$ is proposed. This extension method is based on a finite element $L^2-$ or $H^1-$projection combined with a *ghost-penalty method*. The ghost-penalty method is widely used to enhance the stability of finite element formulations [Bur10; Bur+14]. More recently, it was also used and analysed as an implicit extension procedure for an Eulerian finite element formulation of PDEs posed in time-dependent domains, cf. [LO19; WRL21]. We will use the ghost-penalty similarly as an explicit extension operator. The resulting extension procedure can be formulated as a *linear variational problem* in a narrow band around the surface. This makes it both computationally efficient and amenable to rigorous error analysis.

An important consideration in any narrow band level set method is the selection of appropriate artificial boundary conditions for the level set equation at the inflow boundary of the transport domain. The extension method we propose allows flexibility in selecting domains for local projection and ghost-penalty stabilization. With a suitable choice of these domains, it is possible to achieve higher order approximations of the zero level of $\phi$, i.e., the surface, even when numerical boundary data possess only lower-order accuracy.

The remainder of this chapter is structured as follows. In Section 6.2, we outline our assumptions regarding the level set function and describe the framework of the narrow band discretization method. Section 6.3 addresses the issue of numerical boundary conditions at the inflow boundary of the narrow band. In Section 6.4, we explain the methods employed for spatial and temporal discretization of the level set equation. The extension method is explained, and accuracy bounds are derived in Section 6.5, where we also present results from numerical experiments with this extension method. In Section 6.6, we combine the components resulting in a narrow band algorithm for discretizing the level set equation. Finally, Section 6.7 showcases the performance of this algorithm through numerical experiments.

## 6.2 Level set equation and structure of the narrow band method

In this section, we formulate the problem that we aim to solve and provide an overview of our narrow band level set method. We assume that the evolving surface is given by the zero level of a sufficiently smooth level set function $\phi$ defined in $\Omega \times [0, t_{\text{end}}]$ with $\Omega$ a polygonal domain that contains the surface $\Gamma(t)$ for $t \in [0, t_{\text{end}}]$. The surface is given by

$$\Gamma(t) = \{ \mathbf{x} \in \Omega \mid \phi(\mathbf{x}, t) = 0 \}.$$

Here, we assume the level set function $\phi$ to be close to a signed distance function in the sense that for fixed constants $c_1$, $c_2$ it fulfills

$$0 < c_1 \leq |\nabla \phi(\mathbf{x}, t)| \leq c_2 \quad \text{for } \mathbf{x} \in \mathcal{O}(\Gamma(t)),\ t \in [0, t_{\text{end}}],$$

where $\mathcal{O}(\Gamma(t)) = \Omega_\epsilon(t) := \{ \mathbf{x} \in \mathbb{R}^d \ : \ |\phi(\mathbf{x}, t)| < \epsilon \}$ denotes a neighborhood of the surface, with fixed width $\epsilon > 0$.

The geometric evolution of the surface is driven by the material flow field $\mathbf{u}$. The connection between the surface evolution in terms of the level set function and the velocity $\mathbf{u}$ is described in Section 2.6.1. The level set function is the unique solution of

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0 \quad \text{on} \bigcup_{t \in (0, t_{\text{end}}]} \mathcal{O}(\Gamma(t)) \times \{t\},$$

with a suitable initial condition $\phi(\mathbf{x}, 0) = \phi_0(\mathbf{x})$ for $\mathbf{x} \in \mathcal{O}(\Gamma(0))$.

It is important to observe that inflow boundary conditions are unnecessary in this setting due to the fact that the spatial boundary of the space-time neighborhood $\bigcup_{t \in (0, t_{\text{end}}]} \mathcal{O}(\Gamma(t)) \times \{t\}$ is a characteristic boundary. Our primary focus is the accurate recovery of $\Gamma(t)$ and not the accurate approximation of the function $\phi$ in the entire neighborhood $\bigcup_{t \in [0, t_{\text{end}}]} \mathcal{O}(\Gamma(t))$. Consequently, it suffices to obtain an accurate approximation of $\phi$ within a smaller subdomain that contains $\Gamma(t)$ for $t \in [0, t_{\text{end}}]$.

We outline the construction of our narrow–band method, starting with the time stepping procedure and the choice of the narrow bands $\Omega_{\text{Tr}}^n$. We chose a fixed time step size $\Delta t = t_{\text{end}}/N$ for some $N > 0$, but the method also works for varying time step sizes with minor modifications. The time nodes are denoted by $t_n = \Delta t n$, $n = 0, \ldots, N$ with $0 = t_0 < t_1 < \ldots < t_N = t_{\text{end}}$. We use the notations $f^n(\cdot) = f(\cdot, t_n)$ and $\Gamma^n = \Gamma(t_n)$ for time-dependent functions and surfaces. In the background domain $\Omega$ we assume a family of shape regular quasi-uniform simplicial triangulations $\{\mathcal{T}_h\}_{h>0}$. The subset of simplices that are "cut" by $\Gamma(t_n)$ is again denoted by $\mathcal{T}_\Gamma^n = \{ T \in \mathcal{T}_h \mid \text{meas}_{d-1}(T \cap \Gamma^n) > 0 \}$. We focus on narrow bands consisting of the "cut" elements with some layers of neighboring elements added. We introduce the notation to add layers of elements to a sub-triangulation $\tau_h \subset \mathcal{T}_h$ by

$$\mathcal{N}^1(\tau_h) = \mathcal{N}(\tau_h) := \left\{ T \in \mathcal{T}_h \mid T \cap \tau_h \neq \emptyset \right\} \text{ and } \mathcal{N}^j(\tau_h) = \mathcal{N}(\mathcal{N}^{j-1}(\tau_h)), \quad j \geq 2. \tag{6.2.1}$$

The narrow bands $\Omega_{\text{Tr}}^n$ for $1 \leq n \leq N$ are defined by adding $L_T$ layers of elements to the "cut" elements. They are given by

$$\Omega_{\text{Tr}}^n = \mathcal{N}^{L_T}(\mathcal{T}_\Gamma^n).$$

Thus, for $h$ sufficiently small we have the embedding $\mathcal{T}_\Gamma^n \subset \Omega_{\mathrm{Tr}}^n \subset \mathcal{O}(\Gamma(t_n))$. The number of element layers $L_T$ in the narrow band has to be chosen sufficiently large (or the time step size $\Delta t$ sufficiently small), such that $\Gamma^{n+1}$ is contained in $\Omega_{\mathrm{Tr}}^n$. This is a crucial condition for our method, and we will test it in each time step. See Fig. 6.2.1 for a 1D illustration. In Section 6.6 we discuss how this condition is handled in our algorithm.

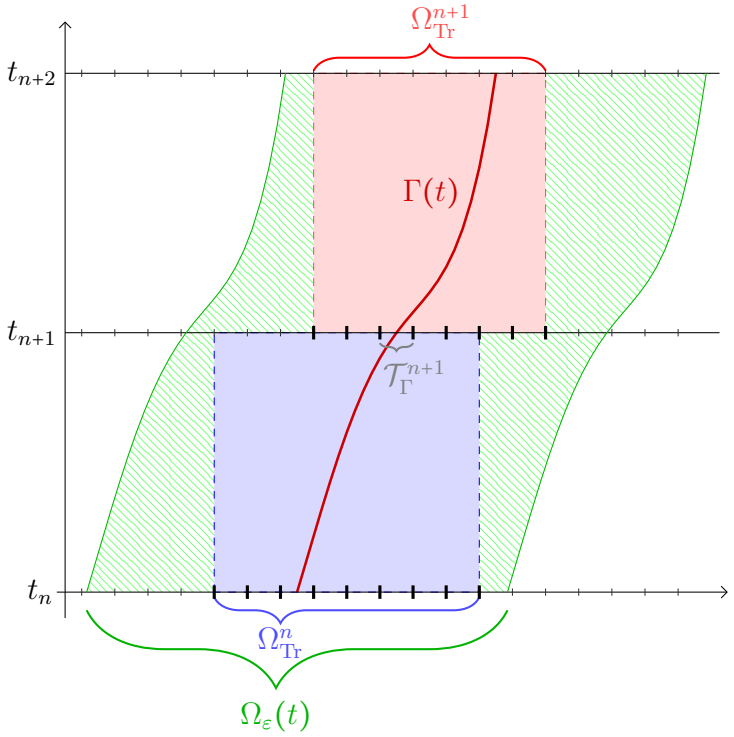



Figure 6.2.1: Sketch of narrow band $\mathcal{O}(\Gamma(t)) = \Omega_\epsilon(t)$ and successive time slabs of transport domains $\Omega_{\mathrm{Tr}}^n \times [t_n, t_{n+1}]$, $\Omega_{\mathrm{Tr}}^{n+1} \times [t_{n+1}, t_{n+2}]$

.

In an Eulerian framework, we (approximately) solve the level set equation on a space-time subdomain $\Omega_{\mathrm{Tr}}^n \times [t_n, t_{n+1}]$ and then proceed to the next space-time subdomain $\Omega_{\mathrm{Tr}}^{n+1} \times [t_{n+1}, t_{n+2}]$. It is evident that for a well-posed formulation of the level set equation on $\Omega_{\mathrm{Tr}}^n \times [t_n, t_{n+1}]$, we need initial data on $\Omega_{\mathrm{Tr}}^n$ for $t = t_n$ as well as boundary data on the inflow part of the piecewise planar boundary of $\Omega_{\mathrm{Tr}}^n$. The inflow boundary is defined by

$$\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}(t) = \{\, \mathbf{x} \in \partial\Omega_{\mathrm{Tr}}^n \;\mid\; \mathbf{u}(\mathbf{x},t) \cdot \mathbf{n}_{\Omega_T^n}(\mathbf{x}) < 0 \,\} \qquad \text{for } t \in [t_n, t_{n+1}],$$

where $\mathbf{n}_{\Omega_T^n}$ denotes the outward pointing unit normal on $\partial\Omega_{\mathrm{Tr}}^n$. For simplicity, in the remainder we assume that the inflow boundary is constant in each time interval, i.e., $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}(t)$ is independent of $t \in [t_n, t_{n+1}]$. We denote Dirichlet boundary data on this inflow boundary by $\phi_D^n$.

Given the narrow bands $\Omega_{\mathrm{Tr}}^n$ for $n = 0, \ldots, N-1$, that satisfy the condition $\Gamma_h^{n+1} \subset \Omega_{\mathrm{Tr}}^n$, we outline the structure of our narrow band method and the following sections: As a first step, Dirichlet boundary data $\phi_D^n$ on the inflow boundary $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$ are constructed. In Section 6.3, an approach for the construction is explained. Next, the level set equation is solved on the narrow band using a Galerkin DG discretization in space combined with

BDF in time as detailed in Section 6.4. Note that the choice of the discretization scheme is not crucial and the narrow band algorithm is also combinable with other transport schemes. The final extension step is performed using a ghost penalty technique, which is explained in Section 6.5.

In Algorithm 6.2.1, we summarize the structure of one time step of the narrow band method.

**Algorithm 6.2.1: One time step structure**

**Input:** $\phi_h^n, \mathbf{u}^n$ defined on $\Omega_{\mathrm{Tr}}^n$

**Procedure**

Step 1: $\phi_D^n \leftarrow$ boundary conditions on the inflow boundary $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$

Step 2: $\tilde{\phi}_h^{n+1} \leftarrow$ solve level set equation on $\Omega_{\mathrm{Tr}}^n \times [t_n, t_{n+1}]$

Step 3: $\phi_h^{n+1} \leftarrow$ extension of $\tilde{\phi}_h^{n+1}$ from $\Omega_{\mathrm{Tr}}^n$ to $\Omega_{\mathrm{Tr}}^{n+1}$

**end Procedure**

**Output:** $\phi_h^{n+1}$ defined on $\Omega_{\mathrm{Tr}}^{n+1}$

## 6.3 Construction of inflow boundary data

In this section, we discuss the construction of Dirichlet boundary data used in Step 1 in Algorithm 6.2.1. To construct Dirichlet boundary data on $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$ we use a (linear) mapping $B_n$ that defines $\phi_D^n = \phi_D^n(\mathbf{x}, t)$, $\mathbf{x} \in \partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$, $t \in [t_n, t_{n+1}]$ from the previous approximations $\phi_h^n, \phi_h^{n-1}, \ldots, \phi_h^{n-R+1}$. Here, $R$ is the number of approximations from previous time steps that are used to construct the boundary data. The choice of $R$ depends on the time stepping scheme (BDF(R)) that is used. To define $B_n$, we give two approaches. The first one resembles the straightforward idea used in equation (5.5.1). It consists of a simple extrapolation (in time) of the restriction of previous level set approximations to the inflow boundary. For a BDF(1) method, we use

$$\phi_D^n = B_n(\phi_h^n) := \phi_{h|\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}}^n. \tag{6.3.1}$$

If one uses BDF(2) it is natural to incorporate information of $\phi_h^{n-1}$ in the boundary data. An analogous choice of the boundary data is given by

$$\phi_D^n = B_n(\phi_h^n, \phi_h^{n-1}) := 2\phi_{h|\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}}^n - \phi_{h|\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}}^{n-1}. \tag{6.3.2}$$

These choices for the boundary data are independent of $t \in [t_n, t_{n+1}]$. If $\phi_h^n(\mathbf{x}) = \phi(\mathbf{x}, t_n)$ and $\phi_h^{n-1}(\mathbf{x}) = \phi(\mathbf{x}, t_{n-1})$ for $\mathbf{x} \in \Omega_{\mathrm{Tr}}^n$ hold, this construction yields boundary data that are first order (for BDF(1)) or second order (for BDF(2)) accurate in $\Delta t$. Along the same lines as in equation (5.5.1) one can also construct higher order accurate boundary data for BDF(R) schemes with $R > 2$.

A second, more accurate (in $\Delta t$) approximation of the boundary data is obtained by using the structure of the level set equation. For BDF(1) it is given by

$$\phi_D^n = B_n(\phi_h^n, \mathbf{u}^n) := \phi_{h|\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}}^n - (t - t_n)(\mathbf{u}^n \cdot \nabla\phi_h^n)_{|\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}}. \tag{6.3.3}$$

For $\phi_h^n = \phi^n$ (and thus $\nabla\phi_h^n = \nabla\phi^n$), this choice of boundary data is second order accurate in $\Delta t$. Along the same lines, one can construct higher order approximations for higher order BDF schemes. Both options (6.3.1) and (6.3.3) are natural choices in a BDF(1) time stepping procedure.

In Section 6.6, we will demonstrate that it is possible to achieve higher order discretization accuracy of the level set function close the surface, even when using low-order (in $\Delta t$) accurate Dirichlet boundary data. This phenomenon is related to the transport nature of the level set problem. In the transport equation, errors occurring on a level set defined by $|\phi(\cdot, t_n)| = ch$ do not propagate into the region formed by the level sets where $|\phi(\cdot, t)| < ch$ for all times $t \geq t_n$.

The default boundary data operator that we use in our numerical experiments is the one in (6.3.2) for BDF(2) or its higher order variants for other BDF(R) schemes. We will discuss the choice of the boundary data in Section 6.7. We observe that increasing the order (in time) of the boundary data operator does not necessarily lead to a better approximation of the surface, but increases the computational cost. In Table 6.7.4, the results of different boundary data operators are compared.

## 6.4 Discretization of the level set equation

This section introduces the method we use in step 2 of Algorithm 6.2.1. The method discretizes the level set equation (2.6.2) on one time slab $\Omega_{\mathrm{Tr}}^n \times [t_n, t_{n+1}]$. For simplicity, we drop the time slab superscript $n$ and write $\Omega_{\mathrm{Tr}}$, $\phi_h$, $\phi_D$ instead of $\Omega_{\mathrm{Tr}}^n$, $\phi_h^n$, $\phi_D^n$ in this section. The Dirichlet boundary data $\phi_D$ is assumed to be given on the inflow boundary $\partial\Omega_{\mathrm{Tr}}^{\mathrm{in}} = \partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$.

We employ a discontinuous Galerkin (DG) method for the spatial discretization and combine it with a backward differentiation formula (BDF) scheme for the time discretization. The DG method is detailed in [BDF16] and is briefly summarized in the following section. Unlike most of the methods in the literature, this DG method does not require the velocity $\mathbf{u}$ to be divergence-free, which is beneficial for applications that motivated this work. See Remark 6.4.2 for a discussion of this topic.

The discontinuous finite element space is given by

$$V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}}) := \{\, \psi \in L_2(\Omega_{\mathrm{Tr}}) \mid \psi|_T \in P_{k_g}(T) \;\; \forall T \in \mathcal{T}_h^{\mathrm{Tr}} \,\}, \tag{6.4.1}$$

where $\mathcal{T}_h^{\mathrm{Tr}} := \mathcal{T}_h \cap \Omega_{\mathrm{Tr}}$ is the set of tetrahedra in the narrow band $\Omega_{\mathrm{Tr}}$.

As common in DG methods, we use upwind fluxes and a suitable jump operator across the faces in $\mathcal{T}_h$. The boundary of an element $T \in \mathcal{T}_h^{\mathrm{Tr}}$ is split into its inflow and outflow part, given by

$$\partial T^-(t) := \left\{ \mathbf{x} \in \partial T \;:\; \mathbf{u}(\mathbf{x}, t) \cdot \mathbf{n}_T(\mathbf{x}) < 0 \right\},$$
$$\partial T^+(t) := \left\{ \mathbf{x} \in \partial T \;:\; \mathbf{u}(\mathbf{x}, t) \cdot \mathbf{n}_T(\mathbf{x}) \geq 0 \right\},$$

where $\mathbf{n}_T$ denotes the outward pointing unit normal on $T$. It holds $\partial T = \partial T^- \cup \partial T^+$.

Due to the discontinuity of $\phi_h \in V_{h,k_g}^{\mathrm{DG}}$ over the faces of the tetrahedra, the values of the finite element functions on the boundary of the tetrahedra are not uniquely defined. More precisely, let $\hat{T}$ be an adjacent tetrahedron to $T$, such that $T$ and $\hat{T}$ are connected

over the face $F$, see Figure 6.4.1 for a visualization. Then, $\mathbf{n}_T$ denotes the normal to $F$ pointing outward of $T$ (and into $\hat{T}$). For the evaluation of the finite element function $\phi_h$ on $F$, one can choose $\phi_{h|F} = \phi_{h|T}$ or $\phi_{h|F} = \phi_{h|\hat{T}}$.

We evaluate $\phi_h$ on $F$ depending on the direction of the flow field $\mathbf{u}$. Since the information is transported in the direction of the velocity field, it is a natural choice to take the value of $\phi_h$ that flows "into" the face. Here, "into" means that we take $\phi_{h|F} = \phi_{h|T}$ if the velocity $\mathbf{u}$ points from the tetrahedron $T$ into $F$ (and thus from $T$ into $\hat{T}$), i.e., we evaluate $\phi_h$ on $F$ as

$$\phi_h|_F := \begin{cases} \phi_{h|T}, & \mathbf{u} \cdot \mathbf{n}_T \geq 0, \\ \phi_{h|\hat{T}}, & \mathbf{u} \cdot \mathbf{n}_T < 0. \end{cases} \tag{6.4.2}$$

To simplify the notation, we keep $T$ as the primary tetrahedron if nothing else is stated and write $\phi_h = \phi_h|_T$ and $\hat{\phi}_h = \phi_h|_{\hat{T}}$.

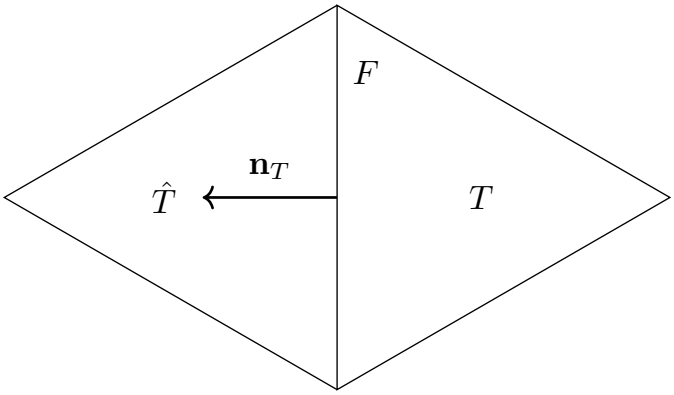



Figure 6.4.1: Adjacent Triangles $T$ and $\hat{T}$ sharing the face $F$.

We summarize the derivation of a semi-discrete DG finite element formulation of (2.6.2) from [BDF16]. To obtain a weak formulation, we replace $\phi$ with an approximation $\phi_h \in V_{h,k_g}^{\mathrm{DG}}$, multiply with a test function $\psi_h \in V_{h,k_g}^{\mathrm{DG}}$, and integrate over a tetrahedron $T \in \mathcal{T}_h^{\mathrm{Tr}}$ with $\mathbf{n}_T$ the outward pointing normal to $\partial T$. Since $\phi_h$ and $\psi_h$ are smooth on $T$, we can use Green's formula inside $T$. We replace $\phi_h$ with (6.4.2) in the boundary integral and get

$$\begin{aligned}
&\int_T (\psi_h \mathbf{u}) \cdot \nabla \phi_h \,\mathrm{d}\mathbf{x} = -\int_T \operatorname{div}(\psi_h \mathbf{u})\, \phi_h \,\mathrm{d}\mathbf{x} + \int_{\partial T} \phi_h \psi_h\, \mathbf{u} \cdot \mathbf{n}_T \,\mathrm{d}s \\
= \;& -\int_T \operatorname{div}(\psi_h \mathbf{u})\, \phi_h \,\mathrm{d}\mathbf{x} + \int_{\partial T^-} \hat{\phi}_h \psi_h\, \mathbf{u} \cdot \mathbf{n}_T \,\mathrm{d}s + \int_{\partial T^+} \phi_h \psi_h\, \mathbf{u} \cdot \mathbf{n}_T \,\mathrm{d}s \\
\overset{\text{Green}}{=} \;& \int_T \psi_h\, (\mathbf{u} \cdot \nabla \phi_h)\, \mathrm{d}\mathbf{x} - \int_{\partial T} \phi_h \psi_h\, (\mathbf{u} \cdot \mathbf{n}_T)\, \mathrm{d}s + \int_{\partial T^-} \hat{\phi}_h \psi_h\, \mathbf{u} \cdot \mathbf{n}_T \,\mathrm{d}s \\
&+ \int_{\partial T^+} \phi_h \psi_h\, \mathbf{u} \cdot \mathbf{n}_T \,\mathrm{d}s \\
= \;& \int_T \psi_h\, (\mathbf{u} \cdot \nabla \phi_h)\, \mathrm{d}\mathbf{x} - \int_{\partial T^-} \phi_h \psi_h\, (\mathbf{u} \cdot \mathbf{n}_T)\, \mathrm{d}s + \int_{\partial T^-} \hat{\phi}_h \psi_h\, \mathbf{u} \cdot \mathbf{n}_T \,\mathrm{d}s \\
= \;& \int_T \psi_h\, (\mathbf{u} \cdot \nabla \phi_h)\, \mathrm{d}\mathbf{x} - \int_{\partial T^-} \left(\phi_h - \hat{\phi}_h\right) \psi_h\, (\mathbf{u} \cdot \mathbf{n}_T)\, \mathrm{d}s.
\end{aligned}$$

On the boundary of the computational domain $\partial\Omega_{\mathrm{Tr}}$ we cannot use the values of $\phi_h$ from the neighboring elements (because there are no neighboring elements). Thus, we replace

$\hat{\phi}_h$ with the Dirichlet boundary data $\phi_D$ on $\partial\Omega_{\mathrm{Tr}}^{\mathrm{in}}$. We introduce the jump operator, given by

$$[\phi_h] = [\phi_h]_{|F} = \phi_{h|T} - \phi_{h|\hat{T}} = \phi_h - \hat{\phi}_h.$$

Summing over all tetrahedra in $\mathcal{T}_h^{\mathrm{Tr}} = \mathcal{T}_h \cap \Omega_{\mathrm{Tr}}$ leads to the semi-discrete formulation of (2.6.2), given by: Find $\phi_h(t) \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$ that satisfies $\phi_h(t_n) = \phi_h^n$ and

$$\begin{aligned}\sum_{T\in\mathcal{T}_h^{\mathrm{Tr}}}\Big(\int_T \frac{\partial\phi_h}{\partial t}\psi_h\,\mathrm{d}\mathbf{x} + \int_T (\mathbf{u}\cdot\nabla\phi_h)\psi_h\,\mathrm{d}\mathbf{x} - \int_{\partial T^-\setminus\partial\Omega_{\mathrm{Tr}}} [\phi_h]\psi_h(\mathbf{u}\cdot\mathbf{n}_T)\,\mathrm{d}s\Big)\\ -\int_{\partial\Omega_{\mathrm{Tr}}^{\mathrm{in}}} \phi_h\psi_h(\mathbf{u}\cdot\mathbf{n}_{\Omega_{\mathrm{Tr}}})\,\mathrm{d}s = -\int_{\partial\Omega_{\mathrm{Tr}}^{\mathrm{in}}} \phi_D\psi_h\left(\mathbf{u}\cdot\mathbf{n}_{\Omega_{\mathrm{Tr}}}\right)\mathrm{d}s\end{aligned} \tag{6.4.3}$$

for all $\psi_h \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$ for $t \in [t_n, t_{n+1}]$. Note that $\mathbf{u}$ and $\phi_D$ may depend on $t$. This formulation equals the one given in [BDF16, Definition 2.4.].

To obtain a fully discrete method, we discretize the time derivative of $\phi_h$. There are different schemes available, e.g., in [BDF16] a discontinuous Galerkin scheme is used for the spatial and the time discretization, in, e.g., [MRC06; DLP06] a discontinuous Galerkin method in space is combined with a Runge-Kutta time discretization scheme.

We aim to combine the transport equation for the level set function $\phi$ with the surface Navier–Stokes equation for the material velocity $\mathbf{u}$. Thus, we want to minimize the evaluations of $\mathbf{u}$, since for each evaluation a computationally expensive surface PDE has to be solved. For that reason, we use implicit BDF schemes that only need one new evaluation of $\mathbf{u}$ in each time step and are easily implemented within our finite element software. Specifically, the BDF(1) scheme (implicit Euler) leads to the following fully discrete problem within a single time interval: For given $\phi_h^n \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$, find $\tilde{\phi}_h^{n+1} \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$ such that

$$\begin{aligned}\sum_{T\in\mathcal{T}_h^{\mathrm{Tr}}}\Big(\int_T \frac{\tilde{\phi}_h^{n+1}-\phi_h^n}{\Delta t}\psi_h\,\mathrm{d}\mathbf{x} + \int_T \left(\mathbf{u}^{n+1}\cdot\nabla\tilde{\phi}_h^{n+1}\right)\psi_h\,\mathrm{d}\mathbf{x} - \int_{\partial T^-\setminus\partial\Omega_{\mathrm{Tr}}} [\tilde{\phi}_h^{n+1}]\psi_h\left(\mathbf{u}^{n+1}\cdot\mathbf{n}_T\right)\mathrm{d}s\Big)\\ -\int_{\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}} \tilde{\phi}_h^{n+1}\psi_h\left(\mathbf{u}^{n+1}\cdot\mathbf{n}_{\Omega_{\mathrm{Tr}}}\right)\mathrm{d}s = -\int_{\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}} \phi_D\,\psi_h\left(\mathbf{u}^{n+1}\cdot\mathbf{n}_{\Omega_{\mathrm{Tr}}}\right)\mathrm{d}s\end{aligned} \tag{6.4.4}$$

holds for all $\psi_h \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$. The extension of this method to higher order BDF (used in the numerical experiments in Section 6.7) is straightforward.

The extension method discussed in the next section applies to continuous finite element functions ($H^1$–conforming). However, from the DG method we obtain finite element approximations that in general are discontinuous across the edges/faces of the triangulation. Therefore, we use an Oswald-type quasi-interpolation operator that maps (with optimal order of accuracy) the DG finite element functions into the $H^1$-conforming Lagrange finite element space with the same polynomial degree. The Oswald interpolation operator $\Pi_h^k$ was also used in Section 4.1. See [GE17] for further details. Using this quasi-interpolation operator, we obtain a continuous approximation $\Gamma_h(t_n)$ of the zero level $\Gamma(t_n)$.

**Remark 6.4.1**

The specific choice of the discretization method for the level set equation in step 2 of Algorithm 6.2.1 is not crucial for the performance of our narrow band level set method. Given that the level set equation is a pure transport equation, a finite element discontinuous Galerkin (DG) technique is an appropriate choice. Related DG techniques can be found in [SVD06; vS08; CS89; CHS90]. Alternatively, a stabilized conforming finite element method, such as the streamline diffusion finite element method (cf. [RST08]), could be used. ◇

**Remark 6.4.2 (Divergence of flow field)**

In our case, the flow field $\mathbf{u}$ that transports the level set function is determined by the surface Navier–Stokes equations. These equations include an inextensibility condition ($\operatorname{div}_\Gamma \mathbf{u} = 0$ on $\Gamma$). The finite element scheme that we will introduce in the next chapter, constructs approximations $\mathbf{u}_h$ of $\mathbf{u}$ that are approximately constant in the normal direction ($\nabla \mathbf{u}_h \mathbf{n}_h \approx 0$) in a thin vicinity around the surface. Hence,

$$\begin{aligned}\operatorname{div} \mathbf{u}_h &= \operatorname{tr}(\nabla \mathbf{u}_h) \approx \operatorname{tr}(\nabla \mathbf{u}_h - \underbrace{\nabla \mathbf{u}_h \mathbf{n}_h}_{\approx 0} \mathbf{n}_h^T) = \operatorname{tr}\left(\nabla \mathbf{u}_h \mathbf{P}_h\right) = \operatorname{tr}(\mathbf{P}_h \nabla \mathbf{u}_h \mathbf{P}_h) = \operatorname{tr}\left(\nabla_{\Gamma_h} \mathbf{u}_h\right) \\ &= \operatorname{div}_{\Gamma_h} \mathbf{u}_h \approx 0.\end{aligned}$$

Thus, the flow (which is extended constant in normal direction) is divergence-free. However, in the application of the method the inextensibility and the extension are only enforced weakly. In addition, the divergence of the extended velocity depends strongly on the choice of the extension method. One could also use different extension methods that do not conserve the divergence free property. Thus, the level set function is transported by a finite element approximation of the flow field $\mathbf{u}_h$ which does not satisfy $\operatorname{div} \mathbf{u}_h = 0$ in $\Omega_{\mathrm{Tr}}^n$ pointwise. For that reason, it is important to use a numerical scheme that does not need the velocity field to be divergence-free. ◇

**Remark 6.4.3 (Analysis of transport method)**

We do not give an analysis of the method presented in (6.4.4) and refer to the literature. In [BDF16] the semi-discrete discretization scheme (6.4.3) on a fixed polygonal domain $\Omega \subset \mathbb{R}^d$ (instead of our $h$ dependent domain $\Omega_{\mathrm{Tr}}$) and a fully discrete formulation using a DG method in time is analysed also for the higher order case. The fully discrete method is similar to our method (6.4.4) in case of a first-order approximation of the time derivative. In [BDF16, Theorem 3.6, 3.7], the authors derive a suboptimal error bound for the $L^2$-norm of the level set function for the semi and fully discrete formulation of order $h^{k_g+\frac{1}{2}}$. However, in their numerical experiments, they observe an optimal order of convergence indicating that the error bounds in the analysis are not sharp. In other literature, different discretization schemes have been analysed. A space-time discontinuous Galerkin method for an advection-diffusion problem was analysed in [SVD06; vS08]. The discontinuous Galerkin method combined with a Runge-Kutta time discretization for the transport problem in a conservative form (i.e., with a divergence-free flow field) has been analysed, e.g., in [CS89; CHS90]. ◇

## 6.5 Extension of the level set function

In this section we describe and analyse a particular extension method for step 3 of Algorithm 6.2.1, the extension of $\tilde{\phi}_h^{n+1}$ defined on $\Omega_{\mathrm{Tr}}^n$ to $\phi_h^{n+1}$ defined on $\Omega_{\mathrm{Tr}}^{n+1}$.

One approach to extend the level set function from the literature is the so-called Fast Marching Method (FMM), cf. [GR11; LOX18]. It uses a greedy algorithm to calculate an approximative distance to the surface in each mesh point. The method is computationally expensive on unstructured grids and achieving higher order accuracy using the FMM is difficult.

Thus, we will not use the FMM and propose an extension method that uses a *ghost-penalty technique*. Several variants of the ghost-penalty method are known in the literature. The original version, known as local projection stabilization (LPS), was introduced in [Bur10]. Another variant is the normal derivative jump stabilization, extensively discussed in works such as [Bur+14; BHL15; Mas+14] for its first-order version, and in [Leh17; SW14] for higher order versions. In this work, we restrict ourselves to the so-called "volumetric jump" or "direct" formulation of the ghost penalty stabilization. It was derived in [Pre18] and further investigated and analysed, e.g., in [LO19; WRL21]. We refer to [LO19] for a comparison of different ghost penalty techniques. To the best of our knowledge, there is no existing literature where the ghost penalty technique is applied within the framework of a narrow band level set method.

We derive and analyse our extension method in a more general setting. Assume an unknown sufficiently smooth function $\phi$ defined on a polygonal domain $\Omega$ with an approximation $\tilde{\phi} \in H^1(\Omega_{\mathrm{Pr}})$ that is only known in $\Omega_{\mathrm{Pr}} \subset \Omega$. We extend $\tilde{\phi}$ from the projection-subdomain $\Omega_{\mathrm{Pr}}$ to some extension-domain $\Omega_{\mathrm{Ex}}$ that satisfies $\Omega_{\mathrm{Pr}} \subset \Omega_{\mathrm{Ex}} \subset \Omega$. The difference between $\Omega_{\mathrm{Pr}}$ and its extension is denoted by $\Omega_{\mathrm{Diff}} := \Omega_{\mathrm{Ex}} \backslash \Omega_{\mathrm{Pr}}$. The result of the extension method will be a continuous finite element function $\phi_h \in H^1(\Omega_{\mathrm{Ex}})$. We will use an $L^2$- or $H^1$-projection inside $\Omega_{\mathrm{Pr}}$. Thus, $\tilde{\phi}$ and $\phi_h$ will not coincide in $\Omega_{\mathrm{Pr}}$. In Section 6.6, we apply this method for an extension from $\Omega_{\mathrm{Tr}}^n$ to $\Omega_{\mathrm{Tr}}^{n+1}$ in step 3 of Algorithm 6.2.1, where $\tilde{\phi}$ corresponds to a finite element approximation of the level set function $\phi^n$.

In our setting, both domains ($\Omega_{\mathrm{Pr}}$ and $\Omega_{\mathrm{Ex}}$) consist of unions of simplices from $\mathcal{T}_h$, more specifically layers of tetrahedra around the surface $\Gamma_h$. These form tubular neighborhoods of width $\sim h$ around $\Gamma_h$. We want $\Omega_{\mathrm{Ex}}$ to be an extension of the domain $\Omega_{\mathrm{Pr}}$ by a few additional layers of elements, cf. Fig. 6.5.1. However, this specific structure of $\Omega_{\mathrm{Pr}}$ and $\Omega_{\mathrm{Ex}}$ is not required in the analysis of the extension method. Instead, the following more general assumption is made.

**Assumption 6.5.1**

We assume that $\Omega_{\mathrm{Pr}}$ consists of a subset of elements $T \in \mathcal{T}_h$ such that it is path-connected and such that any $T \subset \Omega_{\mathrm{Pr}}$ has a common face (3D) or edge (2D) with another $T \subset \Omega_{\mathrm{Pr}}$. The union of simplices $\Omega_{\mathrm{Ex}}$ is such that it contains $\Omega_{\mathrm{Pr}}$ and for any $T \in \Omega_{\mathrm{Diff}}$ there is a path of length at most $ch$, with a uniform constant $c$, that is completely contained in $\Omega_{\mathrm{Ex}}$ and connects $T$ with an element contained in $\Omega_{\mathrm{Pr}}$.

For simplicity, we identify a set of simplices with the domain defined by the union of the elements. The space of *continuous* finite element functions on a triangulated domain $\tau_h \subset \mathcal{T}_h$ is denoted by

$$V_{h,k_g}(\tau_h) := \{\psi_h \in C(\tau_h) : \psi_{h|T} \in P_{k_g}(T) \ \forall T \in \tau_h\}$$

with the same $k_g \geq 1$ as in (6.4.1). The set of ghost penalty faces $\mathcal{F}_h^{\mathrm{GP}}$ is given by

$$\mathcal{F}_h^{\mathrm{GP}} := \left\{ F \subset \partial T \mid T \in \Omega_{\mathrm{Diff}},\ F \notin \partial\Omega_{\mathrm{Ex}} \right\} \cup \left\{ F \subset \partial T \mid T \in \Omega_{\mathrm{Pr}},\ T \cap \partial\Omega_{\mathrm{Pr}} \neq \emptyset \right\}.$$

This definition ensures that all faces inside the "new" elements $\Omega_{\mathrm{Diff}}$ and their "inner neighbors" in $\Omega_{\mathrm{Pr}}$ are included. The faces on $\partial\Omega_{\mathrm{Ex}}$ are not included, since these do not have two neighboring elements in $\Omega_{\mathrm{Ex}}$. See Figure 6.5.1 for an illustration of $\Omega_{\mathrm{Pr}}$, $\Omega_{\mathrm{Ex}}$ ($\Omega_{\mathrm{Diff}}$), and the corresponding faces $\mathcal{F}_h^{\mathrm{GP}}$.

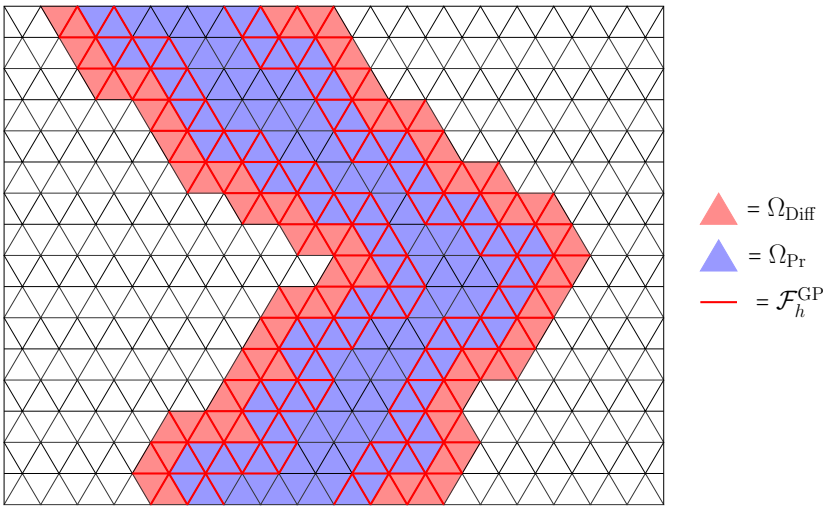



Figure 6.5.1: Sketch of the projection domain ($\Omega_{\mathrm{Pr}}$), the extension domain ($\Omega_{\mathrm{Ex}}$), and the corresponding faces $\mathcal{F}_h^{\mathrm{GP}}$.

For a face $F = T_1 \cap T_2 \in \mathcal{F}_h^{\mathrm{GP}}$, the corresponding patch is defined as $\omega(F) = T_1 \cup T_2$. For a finite element function $\psi \in V_{h,k_g}(\omega(F))$, we denote $\psi_i = \mathcal{E}^P \psi_{|T_i}$ using the canonical polynomial extension operator $\mathcal{E}^P : P_{k_g}(T) \to P_{k_g}(\mathbb{R}^d)$. The ghost penalty bilinear form is formulated as

$$s_h^{(\alpha)}(\phi, \psi) := \gamma^{\mathrm{ext}} h^{-\alpha} \sum_{F \in \mathcal{F}_h^{\mathrm{GP}}} \int_{\omega(F)} (\phi_1 - \phi_2)(\psi_1 - \psi_2)\, \mathrm{d}\mathbf{x}, \quad \text{for } \phi, \psi \in V_{h,k_g}(\Omega_{\mathrm{Ex}}), \tag{6.5.1}$$

with a parameter $\gamma^{\mathrm{ext}} > 0$ and an exponent $\alpha$.

For the error analysis, the ghost penalty bilinear form $s_h^{(\alpha)}(\cdot,\cdot)$ has to be well defined for function from $L^2(\Omega_{\mathrm{Ex}})$ and not only for finite element functions. For that reason, we define $\psi_i = \mathcal{E}^P \Pi_{T_i} \psi_{|T_i}$ for $i = 1, 2$, where $\Pi_{T_i}$ represents the $L^2(T_i)$-projection into $P_{k_g}(T_i)$, $i = 1, 2$, cf. [LO19]. If $\psi \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$, it holds $\Pi_{T_i} \psi_{|T_i} = \psi_{|T_i}$. Consequently, this allows to extend the bilinear form (6.5.1) to $L^2(\Omega_{\mathrm{Ex}}) \times L^2(\Omega_{\mathrm{Ex}})$.

This operator will be employed to define values of $\phi_h$ on the domain $\Omega_{\mathrm{Diff}}$. For obvious reasons, we want to maintain proximity between values of $\phi_h$ and those of $\phi$ on $\Omega_{\mathrm{Pr}}$. To achieve this, we use a projection on $\Omega_{\mathrm{Pr}}$ and define the extension bilinear forms on $V_{h,k_g}(\Omega_{\mathrm{Ex}}) \times V_{h,k_g}(\Omega_{\mathrm{Ex}})$ as

$$a_h^{\mathrm{ext}}(\phi_h, \psi_h) := (\phi_h, \psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\phi_h, \psi_h)$$

$$\text{and} \quad a_{1,h}^{\mathrm{ext}}(\phi_h, \psi_h) := (\phi_h, \psi_h)_{1,\Omega_{\mathrm{Pr}}} + s_h^{(2)}(\phi_h, \psi_h),$$

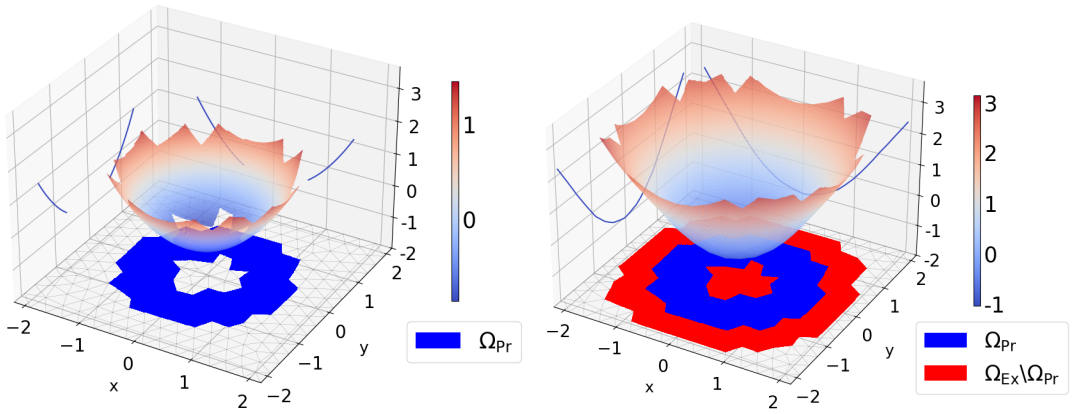
where $(\cdot,\cdot)_{\Omega_{\mathrm{Pr}}}$ and $(\cdot,\cdot)_{1,\Omega_{\mathrm{Pr}}}$ denote the $L^2$- and $H^1$-inner product on $\Omega_{\mathrm{Pr}}$, respectively. The choice of scaling with exponents $\alpha = 0$ and $\alpha = 2$ in these bilinear forms stems from numerical experiments and error analysis from forthcoming sections.

The corresponding extension problems are as follows: Given a function $\tilde{\phi}$ defined on $\Omega_{\mathrm{Pr}}$, determine $\phi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ such that

$$a_h^{\mathrm{ext}}(\phi_h,\psi_h) = (\tilde{\phi},\psi_h)_{\Omega_{\mathrm{Pr}}} \qquad \text{holds for all} \quad \psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}}), \tag{6.5.2a}$$

$$a_{1,h}^{\mathrm{ext}}(\phi_h,\psi_h) = (\tilde{\phi},\psi_h)_{1,\Omega_{\mathrm{Pr}}} \qquad \text{holds for all} \quad \psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}}). \tag{6.5.2b}$$

The extension method from equation (6.5.2a) is illustrated in Figure 6.5.2 for a simple 2D example. The input is a level set function of a sphere on $\Omega_{\mathrm{Pr}}$ (marked in blue), and the output is the extended level set function on $\Omega_{\mathrm{Ex}}$ (marked in red). The extension is performed with the parameters $\gamma^{\mathrm{ext}} = 1$. We observe a smooth transition from the known values on $\Omega_{\mathrm{Pr}}$ to the extended values on $\Omega_{\mathrm{Ex}}$.




(a) Input of extension method

(b) Output of extension method

Figure 6.5.2: Example of the extension of level set function of a sphere in 2D with $k_g = 1$.

In Section 6.6, we apply the first extension method in the context of the narrow band level set method, cf. step 3 of the Algorithm 6.2.1. Both formulations will be analysed in the next section.

### 6.5.1 Error analysis of the extension.

In this section, we derive optimal error estimates for the extension methods on $\Omega_{\mathrm{Ex}}$ in the $L^2$- and $H^1$-norm denoted by $\|\cdot\|_{\Omega_{\mathrm{Ex}}}$ and $\|\cdot\|_{1,\Omega_{\mathrm{Ex}}}$, respectively. We also use the energy norms given by $\|\cdot\|_a = \sqrt{a_h^{\mathrm{ext}}(\cdot,\cdot)}$ and $\|\cdot\|_{1,a} = \sqrt{a_{1,h}^{\mathrm{ext}}(\cdot,\cdot)}$. We start the analysis by citing useful results from the literature. In Theorem 6.5.3 we derive the error estimates for the $L^2$-based extension method (6.5.2a). Theorem 6.5.4 provides the error estimates for the $H^1$-based extension method (6.5.2b). Again, we use the notation $a \lesssim b$ that means that the inequality $a \leq c\,b$ holds with a constant $c$ which is independent of $h$ and the position of $\Gamma$ in the background mesh.

In the following lemma, we summarize some useful results from [LO19]. Therefore, we denote the Lagrange interpolation operator into the finite element space $V_{h,k_g}$ by $I_h$.

**Lemma 6.5.2**

$\psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ and $\psi \in H^{k_g+1}(\Omega_{\mathrm{Ex}})$ satisfy

$$\|\psi_h\|^2_{\Omega_{\mathrm{Ex}}} \lesssim \|\psi_h\|^2_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\psi_h,\psi_h), \tag{6.5.3}$$

$$\|\nabla\psi_h\|^2_{\Omega_{\mathrm{Ex}}} \lesssim \|\nabla\psi_h\|^2_{\Omega_{\mathrm{Pr}}} + s_h^{(2)}(\psi_h,\psi_h), \tag{6.5.4}$$

$$s_h^{(0)}(\psi,\psi) \lesssim h^{2k_g+2}\|\psi\|^2_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}, \tag{6.5.5}$$

$$s_h^{(0)}(\psi - I_h\psi, \psi - I_h\psi) \lesssim h^{2k_g+2}\|\psi\|^2_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}. \tag{6.5.6}$$

**Proof**

See [LO19, Lemma 5.2] for the proof of inequalities (6.5.3)–(6.5.4) and [LO19, Lemma 5.5] for the proof of inequalities (6.5.5)–(6.5.6). □

Using (6.5.3) and (6.5.4) we conclude that

$$a_h^{\mathrm{ext}}(\psi_h,\psi_h) \gtrsim \|\psi_h\|^2_{\Omega_{\mathrm{Ex}}} \quad \text{for all } \psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}}), \tag{6.5.7}$$

$$a_{1,h}^{\mathrm{ext}}(\psi_h,\psi_h) \gtrsim \|\psi_h\|^2_{1,\Omega_{\mathrm{Ex}}} \quad \text{for all } \psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}}) \tag{6.5.8}$$

hold. Thus, the bilinear forms $a_h^{\mathrm{ext}}(\cdot,\cdot)$ and $a_{1,h}^{\mathrm{ext}}(\cdot,\cdot)$ are elliptic on $V_{h,k_g}(\Omega_{\mathrm{Ex}})$ and the equations (6.5.2a) and (6.5.2b) are uniquely solvable.

For the error analysis we assume a $\phi \in H^{k_g+1}(\Omega_{\mathrm{Ex}})$, and $\tilde{\phi}$ used in (6.5.2) is meant to be an approximation of this $\phi$ on $\Omega_{\mathrm{Pr}}$. Since $\phi$ is defined on $\Omega_{\mathrm{Ex}}$, the error of interest $\phi - \phi_h$ is well defined on $\Omega_{\mathrm{Ex}}$.

We start with the error analysis of the $L^2$-based extension method (6.5.2a).

**Theorem 6.5.3**

Let $\phi \in H^{k_g+1}(\Omega_{\mathrm{Ex}})$ be given and $\phi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ the solution of (6.5.2a). The following stability and error estimates hold

$$\|\phi_h\|_a \leq \left(\|\tilde{\phi}\|^2_{\Omega_{\mathrm{Pr}}} - s_h^{(0)}(\phi_h,\phi_h)\right)^{\frac{1}{2}} \leq \|\tilde{\phi}\|_{\Omega_{\mathrm{Pr}}}, \tag{6.5.9}$$

$$\|\phi - \phi_h\|_{\Omega_{\mathrm{Ex}}} \lesssim \|\phi - \tilde{\phi}\|_{\Omega_{\mathrm{Pr}}} + h^{k_g+1}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}, \tag{6.5.10}$$

$$\|\phi - \phi_h\|_{1,\Omega_{\mathrm{Ex}}} \lesssim h^{-1}\|\phi - \tilde{\phi}\|_{\Omega_{\mathrm{Pr}}} + h^{k_g}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}. \tag{6.5.11}$$

**Proof**

Let $\phi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ be the solution of problem (6.5.2a). Using Cauchy Schwarz and the Binomial theorem yields

$$\|\phi_h\|_a^2 = a_h^{\mathrm{ext}}(\phi_h,\phi_h) = (\tilde{\phi},\phi_h)_{\Omega_{\mathrm{Pr}}} \leq \|\tilde{\phi}\|_{\Omega_{\mathrm{Pr}}}\|\phi_h\|_{\Omega_{\mathrm{Pr}}} \leq \frac{1}{2}\|\tilde{\phi}\|^2_{\Omega_{\mathrm{Pr}}} + \frac{1}{2}\left(\|\phi_h\|_a^2 - s_h^{(0)}(\phi_h,\phi_h)\right).$$

From this inequality, the stability estimate follows. Let $\hat{\phi}_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ be such that

$$a_h^{\mathrm{ext}}(\hat{\phi}_h, \psi_h) = (\phi, \psi_h)_{\Omega_{\mathrm{Pr}}} \quad \text{for all } \psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}}).$$

Hence, it holds $\|\phi - \phi_h\|_{\Omega_{\mathrm{Ex}}} \leq \|\phi_h - \hat{\phi}_h\|_{\Omega_{\mathrm{Ex}}} + \|\phi - \hat{\phi}_h\|_{\Omega_{\mathrm{Ex}}}$. We derive bounds for the two terms on the right-hand side. For the first one we use (6.5.7), the identity $a_h^{\mathrm{ext}}(\phi_h - \hat{\phi}_h, \psi_h) = (\tilde{\phi} - \phi, \psi_h)_{\Omega_{\mathrm{Pr}}}$ and the stability bound yielding

$$\|\phi_h - \hat{\phi}_h\|_{\Omega_{\mathrm{Ex}}}^2 \lesssim a_h^{\mathrm{ext}}(\phi_h - \hat{\phi}_h, \phi_h - \hat{\phi}_h) \leq (\tilde{\phi} - \phi, \phi_h - \hat{\phi}_h)_{\Omega_{\mathrm{Pr}}} \leq \|\tilde{\phi} - \phi\|_{\Omega_{\mathrm{Pr}}} \|\phi_h - \hat{\phi}_h\|_{\Omega_{\mathrm{Pr}}}. \tag{6.5.12}$$

To bound the second term, we use (6.5.7) and note that

$$\begin{aligned} \|\phi - \hat{\phi}_h\|_{\Omega_{\mathrm{Ex}}} &\leq \|\phi - I_h\phi\|_{\Omega_{\mathrm{Ex}}} + \|I_h\phi - \hat{\phi}_h\|_{\Omega_{\mathrm{Ex}}} \\ &\lesssim h^{k_g+1} \|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})} + \|I_h\phi - \hat{\phi}_h\|_{\Omega_{\mathrm{Ex}}} \\ &\lesssim h^{k_g+1} \|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})} + \|I_h\phi - \hat{\phi}_h\|_a \end{aligned} \tag{6.5.13}$$

holds. We need a bound for the last term ($\|I_h\phi - \hat{\phi}_h\|_a$). Using

$$\begin{aligned} a_h^{\mathrm{ext}}(\phi - \hat{\phi}_h, \psi_h) &= a_h^{\mathrm{ext}}(\phi, \psi_h) - a_h^{\mathrm{ext}}(\hat{\phi}_h, \psi_h) \\ &= (\phi, \psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\phi, \psi_h) - a_h^{\mathrm{ext}}(\hat{\phi}_h, \psi_h) \\ &= (\phi, \psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\phi, \psi_h) - (\phi, \psi_h)_{\Omega_{\mathrm{Pr}}} \\ &= s_h^{(0)}(\phi, \psi_h) \quad \text{for all } \psi_h \in V_{h,k_g}, \end{aligned} \tag{6.5.14}$$

and the Cauchy Schwarz inequality, we get

$$\begin{aligned} \|I_h\phi - \hat{\phi}_h\|_a^2 &= a_h^{\mathrm{ext}}(I_h\phi - \phi, I_h\phi - \hat{\phi}_h) + a_h^{\mathrm{ext}}(\phi - \hat{\phi}_h, I_h\phi - \hat{\phi}_h) \\ \overset{(6.5.14)}{=} & \; a_h^{\mathrm{ext}}(I_h\phi - \phi, I_h\phi - \hat{\phi}_h) + s_h^{(0)}(\phi, I_h\phi - \hat{\phi}_h) \\ \leq & \; \|I_h\phi - \phi\|_a \|I_h\phi - \hat{\phi}_h\|_a + \sqrt{s_h^{(0)}(\phi, \phi)} \|I_h\phi - \hat{\phi}_h\|_a. \end{aligned}$$

Dividing by $\|I_h\phi - \hat{\phi}_h\|_a$ yields

$$\|I_h\phi - \hat{\phi}_h\|_a \leq \|I_h\phi - \phi\|_a + \sqrt{s_h^{(0)}(\phi, \phi)}. \tag{6.5.15}$$

With the help of (6.5.6) and (6.5.5) we obtain

$$\begin{aligned} \|I_h\phi - \hat{\phi}_h\|_a &\leq \|I_h\phi - \phi\|_{\Omega_{\mathrm{Pr}}} + \sqrt{s_h^{(0)}(I_h\phi - \phi, I_h\phi - \phi)} + \sqrt{s_h^{(0)}(\phi, \phi)} \\ &\lesssim h^{k_g+1} \|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}. \end{aligned} \tag{6.5.16}$$

Combining this estimate with the results in (6.5.12) and (6.5.13) yields the bound in the $L^2$-norm (6.5.10). We derive the estimate in the $H^1$-norm. We start with the triangle inequality $\|\phi - \phi_h\|_{1,\Omega_{\mathrm{Ex}}} \leq \|\phi - \hat{\phi}_h\|_{1,\Omega_{\mathrm{Ex}}} + \|\hat{\phi}_h - \phi_h\|_{1,\Omega_{\mathrm{Ex}}}$ and derive bounds for both terms. For the first term we use an interpolation property of $I_h\phi$, a finite element

inverse inequality, estimates (6.5.7), and (6.5.16) and get

$$\begin{aligned}\|\phi - \hat{\phi}_h\|_{1,\Omega_{\mathrm{Ex}}} &\leq \|\phi - I_h\phi\|_{1,\Omega_{\mathrm{Ex}}} + \|I_h\phi - \hat{\phi}_h\|_{1,\Omega_{\mathrm{Ex}}} \lesssim h^{k_g}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})} + h^{-1}\|I_h\phi - \hat{\phi}_h\|_{\Omega_{\mathrm{Ex}}} \\ &\lesssim h^{k_g}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})} + h^{-1}\|I_h\phi - \hat{\phi}_h\|_a \lesssim h^{k_g}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}.\end{aligned}$$

For the second term we use a finite element inverse equality and (6.5.12) to obtain

$$\|\hat{\phi}_h - \phi_h\|_{1,\Omega_{\mathrm{Ex}}} \lesssim h^{-1}\|\hat{\phi}_h - \phi_h\|_{\Omega_{\mathrm{Ex}}} \lesssim h^{-1}\|\tilde{\phi} - \phi\|_{\Omega_{\mathrm{Pr}}}.$$

Combining these results yields the bound (6.5.11). □

Now, we give an error analysis for the extension problem (6.5.2b).

**Theorem 6.5.4**

Let $\phi \in H^{k_g+1}(\Omega_{\mathrm{Ex}})$ be given and $\phi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ the solution of (6.5.2b). The following stability and error estimates hold

$$\begin{aligned}\|\phi_h\|_{1,a} &\leq \left(\|\tilde{\phi}\|^2_{1,\Omega_{\mathrm{Pr}}} - s_h^{(2)}(\phi_h,\phi_h)\right)^{\frac{1}{2}} \leq \|\tilde{\phi}\|_{1,\Omega_{\mathrm{Pr}}}, \\ \|\phi - \phi_h\|_{1,\Omega_{\mathrm{Ex}}} &\lesssim \|\phi - \tilde{\phi}\|_{1,\Omega_{\mathrm{Pr}}} + h^{k_g}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}.\end{aligned}$$

**Proof**

The proof uses the same arguments as the proof of Theorem 6.5.3. Again, we define $\hat{\phi}_h$ as the solution of (6.5.2b) with $\tilde{\phi}$ replaced by $\phi$. Along the same lines as (6.5.12), we obtain

$$\|\phi_h - \hat{\phi}_h\|_{1,\Omega_{\mathrm{Ex}}} \lesssim \|\tilde{\phi} - \phi\|_{1,\Omega_{\mathrm{Pr}}}.$$

Applying the triangle inequality, an interpolation property of $I_h\phi$ and (6.5.8), we obtain the estimate

$$\|\phi - \hat{\phi}_h\|_{1,\Omega_{\mathrm{Ex}}} \leq \|\phi - I_h\phi\|_{1,\Omega_{\mathrm{Ex}}} + \|I_h\phi - \hat{\phi}_h\|_{1,\Omega_{\mathrm{Ex}}} \lesssim h^{k_g}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})} + \|I_h\phi - \hat{\phi}_h\|_{1,a}.$$

Repeating the estimates (6.5.14)–(6.5.15) with $a_h^{\mathrm{ext}}$ and $\|\cdot\|_a$ replaced by $a_{1,h}^{\mathrm{ext}}$ and $\|\cdot\|_{1,a}$, respectively, we get

$$\|I_h\phi - \hat{\phi}_h\|_{1,a} \leq \|I_h\phi - \phi\|_{1,a} + \sqrt{s_h^{(2)}(\phi,\phi)}.$$

With the help of (6.5.5) and (6.5.6) we obtain

$$\|I_h\phi - \hat{\phi}_h\|_{1,a} \leq \|I_h\phi - \phi\|_{1,\Omega_{\mathrm{Pr}}} + \sqrt{s_h^{(2)}(I_h\phi - \phi, I_h\phi - \phi)} + \sqrt{s_h^{(2)}(\phi,\phi)} \leq h^{k_g}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})},$$

and so the desired error estimate. □

**Remark 6.5.5**

We compare the results of the extension using the $L^2$- and the $H^1$-projection. The error estimates in the $H^1$-norm from Theorems 6.5.3 and 6.5.4 are independent of the choice of the projection. Both are of the same optimal order. Besides the errors in the $H^1$-norm, we also obtain optimal error bounds in the $L^2$-norm for the $L^2$-projection, but not for

the $H^1$-projection.

We compare the errors measured in the energy norm for both methods in a more realistic setting. We extend a finite element function. Let $\tilde{\phi}_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ be a given finite element approximation of $\phi \in H^{k_g+1}(\Omega_{\mathrm{Ex}})$. In case of the narrow band level set algorithm, $\tilde{\phi}_h$ is the result of step 2 of Algorithm 6.2.1 and $\phi$ is the exact level set function. The numerical solutions of (6.5.2a) and (6.5.2b) with this $\tilde{\phi}_h$ are denoted by $\phi_h, \phi_{1,h} \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$, respectively. The corresponding errors are given by

$$e_h := \tilde{\phi}_h - \phi_h \quad \text{and} \quad e_{1,h} := \tilde{\phi}_h - \phi_{1,h}.$$

Note that $e_h, e_{1,h} \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$. For all $\psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ it holds

$$\begin{aligned} a_h^{\mathrm{ext}}(e_h, \psi_h) &= a_h^{\mathrm{ext}}(\tilde{\phi}_h, \psi_h) - a_h^{\mathrm{ext}}(\phi_h, \psi_h) \\ &= (\tilde{\phi}_h, \psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\tilde{\phi}_h, \psi_h) - a_h^{\mathrm{ext}}(\phi_h, \psi_h) \\ &= s^{(0)}(\tilde{\phi}_h, \psi_h) = h^2 s^{(2)}(\tilde{\phi}_h, \psi_h) \\ &= h^2 \left( s^{(2)}(\tilde{\phi}_h, \psi_h) + (\tilde{\phi}_h, \psi_h)_{1,\Omega_{\mathrm{Pr}}} - a_{1,h}^{\mathrm{ext}}(\phi_{1,h}, \psi_h) \right) \\ &= h^2 a_{1,h}^{\mathrm{ext}}(e_{1,h}, \psi_h). \end{aligned}$$

In this equation, we replace $\psi_h$ by $e_h$ and use the inverse inequality

$$h^2 a_{1,h}^{\mathrm{ext}}(e_h, e_h) \lesssim a_h^{\mathrm{ext}}(e_h, e_h)$$

to get

$$\|e_h\|_a^2 = h^2 a_{1,h}^{\mathrm{ext}}(e_{1,h}, e_h) \lesssim c \, \|e_{1,h}\|_a \|e_h\|_a$$

and thus

$$\|\tilde{\phi}_h - \phi_h\|_a \lesssim \|\tilde{\phi}_h - \phi_{1,h}\|_a$$

holds. Hence, measured in the energy norm $\|\cdot\|_a$, up to a constant $c$, the error using the $L^2$-projection is always smaller than the error using the $H^1$-projection. ◇

The following reasons support the use of the $L^2$-projection over the $H^1$-projection in the application of the extension method:

- The $L^2$-projection has slightly lower computational costs than the $H^1$-projection.
- We have optimal error bounds in the $L^2$– and the $H^1$-norm for the $L^2$-projection, but only in the $H^1$-norm for the $H^1$-projection.
- The error in the energy norm is smaller (up to a constant) for the $L^2$-projection than for the $H^1$-projection, see Remark (6.5.5).

Thus, in the application of the extension method in the transport algorithm, we will use the $L^2$-projection.

In Section 6.5.3, we present results of numerical experiments that confirm the optimal approximation properties for both extension methods (based on the $L^2$- and $H^1$-projection). These numerical experiments also show that there are no significant differences in the approximation quality between the methods based on $L^2$- and $H^1$-projection.

### 6.5.2 Long-time behavior.

We also examine another stability aspect of the extension method. In the narrow band level set method, the extension is applied repeatedly in *each time step*, cf. Section 6.2. Thus, it is important to assess its behavior when applied repeatedly ($n$ times). For large $n$, this provides insight into the long-term behavior of the method.

In this section, we restrict the treatment to the $L^2$-projection. Similar results hold for the method based on $H^1$-projection.

For a smooth given $\phi$, we consider $\phi_h^0 := I_h\phi$ and define a sequence of approximations $(\phi_h^n)_{n\in\mathbb{N}}$ as follows: Given $\phi_h^n \in V_{h,k_g}(\Omega_{\mathrm{Pr}})$ define $\phi_h^{n+1} \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ such that it holds

$$a_h^{\mathrm{ex}}(\phi_h^{n+1},\psi_h) = (\phi_h^{n+1},\psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\phi_h^{n+1},\psi_h) = (\phi_h^n,\psi_h)_{\Omega_{\mathrm{Pr}}} \tag{6.5.17}$$

for all $\psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$.

We use the notation $\|\psi\|_s := \sqrt{s_h^{(0)}(\psi,\psi)}$ and note that for all $\psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$, the sequence $(\phi_h^n)_{n\in\mathbb{N}}$ satisfies

$$\begin{aligned} &(\phi_h^{n+1}-\phi_h^n,\psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\phi_h^{n+1},\psi_h) = 0 \\ \text{and}\quad &(\phi_h^n-\phi_h^0,\psi_h)_{\Omega_{\mathrm{Pr}}} = -\sum_{k=1}^{n} s_h^{(0)}(\phi_h^k,\psi_h). \end{aligned} \tag{6.5.18}$$

In the following lemma, we show two long-time stability results.

**Lemma 6.5.6**

The following bounds hold with a constant $c$ independent of $n$ and $h$:

$$\|\phi_h^n\|_{\Omega_{\mathrm{Pr}}} \le \|\phi_h^0\|_{\Omega_{\mathrm{Pr}}}, \tag{6.5.19}$$

$$\|\phi_h^n-\phi_h^0\|_{\Omega_{\mathrm{Pr}}} \le c\min\{1,\sqrt{n}h^{k_g+1}\}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}. \tag{6.5.20}$$

**Proof**

We replace $\psi_h$ by $\phi_h^{n+1}$ in (6.5.17) and obtain

$$\|\phi_h^{n+1}\|_{\Omega_{\mathrm{Pr}}}^2 + \|\phi_h^{n+1}\|_s^2 \le \|\phi_h^n\|_{\Omega_{\mathrm{Pr}}}\|\phi_h^{n+1}\|_{\Omega_{\mathrm{Pr}}} \le \frac{1}{2}\|\phi_h^n\|_{\Omega_{\mathrm{Pr}}}^2 + \frac{1}{2}\|\phi_h^{n+1}\|_{\Omega_{\mathrm{Pr}}}^2,$$

which yields (6.5.19).

Now, we replace $\psi_h$ by $\phi_h^{n+1}-\phi_h^n$ in the first identity in (6.5.18), use the Cauchy Schwarz inequality and get

$$\begin{aligned} \|\phi_h^{n+1}\|_s^2 &= s_h^{(0)}(\phi_h^{n+1},\phi_h^{n+1}) \le \|\phi_h^{n+1}-\phi_h^n\|_{\Omega_{\mathrm{Pr}}}^2 + s_h^{(0)}(\phi_h^{n+1},\phi_h^{n+1}) \\ &= s_h^{(0)}(\phi_h^{n+1},\phi_h^n) \le \|\phi_h^{n+1}\|_s^2\|\phi_h^n\|_s^2 \\ &= \frac{1}{2}\|\phi_h^{n+1}\|_s^2 + \frac{1}{2}\|\phi_h^n\|_s^2. \end{aligned}$$

This yields $\|\phi_h^{n+1}\|_s \le \|\phi_h^n\|_s$ and so $\|\phi_h^n\|_s \le \|\phi_h^0\|_s$. Hence, properties (6.5.5) and (6.5.6)

imply

$$\|\phi_h^n\|_s \leq \|\phi^0\|_s \leq \|I_h\phi - \phi\|_s + \|\phi\|_s \leq ch^{k_g+1}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}. \tag{6.5.21}$$

Using $\psi_h = \phi_h^n - \phi_h^0$ in the second equality of (6.5.18) and $\|\phi_h^{n+1}\|_s \leq \|\phi_h^n\|_s$, we get

$$\begin{aligned}
\|\phi_h^n - \phi_h^0\|_{\Omega_{\mathrm{Pr}}}^2 &= -\sum_{k=1}^{n} s_h^{(0)}(\phi_h^k, \phi_h^n - \phi_h^0) \\
&\leq \sum_{k=1}^{n} \|\phi_h^k\|_s \left(\|\phi_h^n\|_s + \|\phi_h^0\|_s\right) \\
&\leq n \max_{1\leq k\leq n} \|\phi_h^k\|_s \left(\|\phi_h^n\|_s + \|\phi_h^0\|_s\right) \\
&\leq 2n\|\phi_h^0\|_s^2.
\end{aligned}$$

Combining this bound with (6.5.21) yields (6.5.20) with $c\sqrt{n}h^{k_g+1}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}$ at the right-hand side. This can be complemented with $\|\phi_h^n - \phi_h^0\|_{\Omega_{\mathrm{Pr}}} \leq 2\|\phi_h^0\|_{\Omega_{\mathrm{Pr}}}$, which follows from a triangle inequality and (6.5.19). Finally, note that the initial approximation $\phi_h^0$ satisfies $\|\phi_h^0\|_{\Omega_{\mathrm{Pr}}} = \|I_h\phi\|_{\Omega_{\mathrm{Pr}}} \leq c\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Ex}})}$. □

The result in (6.5.20) is expected to be sharp in terms of dependence on $n$, since the initial growth of $\|\phi_h^n - \phi_h^0\|_{\Omega_{\mathrm{Pr}}}$ with $\sqrt{n}$ is observed in numerical experiments, cf. Section 6.5.3.

**Remark 6.5.7**

We note that the choice $\phi_h^0 := I_h\phi$ slightly simplified the analysis, but causes no loss of generality. Indeed, if $\tilde{\phi}_h^n$ is the sequence defined as in (6.5.17), with some general $\phi_h^0 = \tilde{\phi}$, then using (6.5.9) we obtain that $\tilde{\phi}_h^n$ and $\phi_h^n$ stay fullfil

$$\|\tilde{\phi}_h^n - \phi_h^n\|_a \leq \|\tilde{\phi} - I_h\phi\|_{\Omega_{\mathrm{Pr}}} \leq \|\tilde{\phi} - \phi\|_{\Omega_{\mathrm{Pr}}} + ch^{k_g+1}\|\phi\|_{H^{k_g+1}(\Omega_{\mathrm{Pr}})}$$

which shows, that they uniformly close in $n$. ◇

### 6.5.3 Numerical experiments with the extension method.

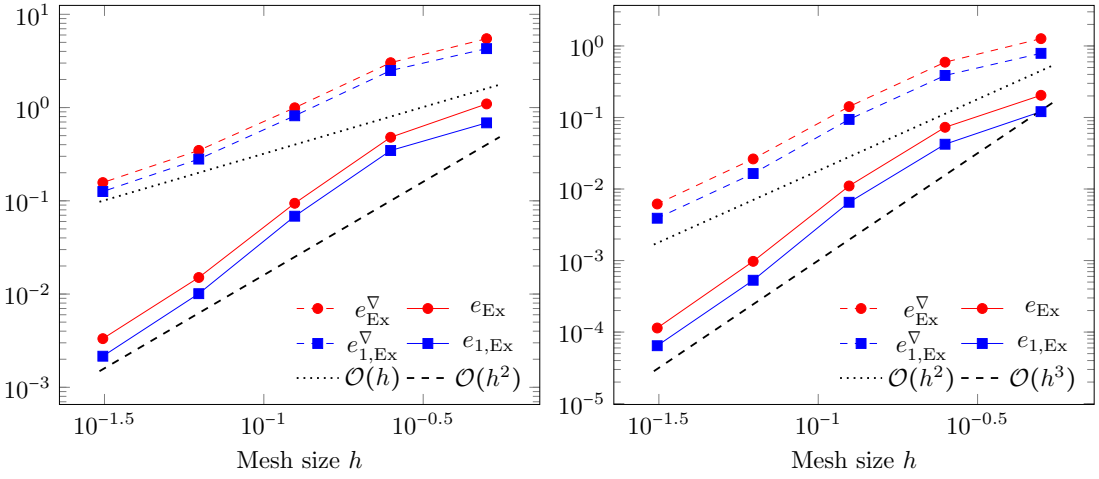
We present the findings from numerical experiments using the ghost penalty extension method as defined in (6.5.2). The exact level set function in all experiments is given by $\phi(\mathbf{x}) := (x_1 - x_3^2)^2 + x_2^2 + x_3^2 - 1$. The corresponding zero level set delineates a geometry referred to as "Kite", which has also been employed in [BR20; Dzi88]. A visualization of this surface is provided in Figure 6.7.1.

The computational domain is chosen as $\Omega = [-\frac{5}{3}, \frac{5}{3}]^3$, and we take a uniform tetrahedral mesh characterized by the initial mesh size $h_0 = 0.5$. In each refinement step, the mesh size is halved. Consistent with the error analysis, we use the parameter values $\alpha = 0$ for the $L^2$-projection and $\alpha = 2$ for the $H^1$-projection. The coefficient $\gamma^{\mathrm{ext}}$ is chosen as $\gamma^{\mathrm{ext}} = 1$ if not stated otherwise. We define the domain $\Omega_{\mathrm{Pr}}$ as the elements intersected by the surface, denoted by $\mathcal{T}_\Gamma$, along with two layers of neighboring elements added, i.e., $\Omega_{\mathrm{Pr}} = \mathcal{N}^2(\mathcal{T}_\Gamma)$. For the input function, we use an Oswald-type interpolation of $\phi$ in the finite element space $V_{h,k_g}(\Omega_{\mathrm{Pr}})$ to get $\tilde{\phi} = \Pi_h^{k_g}\phi$. The solution derived from the extension based on the $L^2$-projection is denoted by $\phi_h$ (refer to equation (6.5.2a)), while that based on the $H^1$-projection is labeled as $\phi_{1,h}$ (refer to equation (6.5.2b)). For the polynomial degree in the finite element space $V_{h,k_g}$ we consider $k_g = 1$ and $k_g = 2$. The extension elements are selected to include either one or two layers of neighbors surrounding $\Omega_{\mathrm{Pr}}$, resulting in $\Omega_{\mathrm{Ex}} = \mathcal{N}(\Omega_{\mathrm{Pr}})$ or $\Omega_{\mathrm{Ex}} = \mathcal{N}^2(\Omega_{\mathrm{Pr}})$.

We investigate the $L^2$- and the $h^1$-errors for the $L^2$- and the $H^1$-projection. Since the domains $\Omega_{\mathrm{Pr}}$ and $\Omega_{\mathrm{Ex}}$ depend on the mesh size $h$, we use scaled integrals given by $\fint_{\Omega_{\mathrm{Ex}}} f \,\mathrm{d}\mathbf{x} = \int_{\Omega_{\mathrm{Ex}}} f \,\mathrm{d}\mathbf{x} / \|1\|^2_{\Omega_{\mathrm{Ex}}}$. The error quantities read

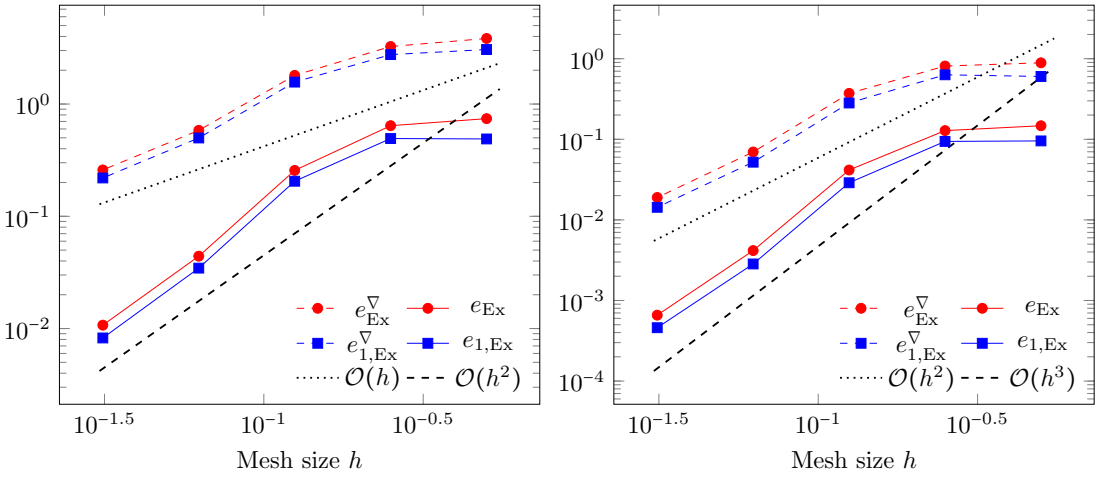
$$
\begin{aligned}
\left(e_{\mathrm{Ex}}\right)^2 &:= \fint_{\Omega_{\mathrm{Ex}}} \left(\phi - \phi_h\right)^2 \mathrm{d}\mathbf{x}, & \left(e^{\nabla}_{\mathrm{Ex}}\right)^2 &:= \fint_{\Omega_{\mathrm{Ex}}} \nabla\left(\phi - \phi_h\right)^2 \mathrm{d}\mathbf{x}, \\
\left(e_{1,\mathrm{Ex}}\right)^2 &:= \fint_{\Omega_{\mathrm{Ex}}} \left(\phi - \phi_{1,h}\right)^2 \mathrm{d}\mathbf{x}, & \left(e^{\nabla}_{1,\mathrm{Ex}}\right)^2 &:= \fint_{\Omega_{\mathrm{Ex}}} \nabla\left(\phi - \phi_{1,h}\right)^2 \mathrm{d}\mathbf{x}.
\end{aligned}
$$

The results of the experiments with one extension layer ($\Omega_{\mathrm{Ex}} = \mathcal{N}(\Omega_{\mathrm{Pr}})$) are shown in Figure 6.5.3 and with two layers ($\Omega_{\mathrm{Ex}} = \mathcal{N}^2(\Omega_{\mathrm{Pr}})$) in Figure 6.5.4.




(a) $k_g = 1$ (b) $k_g = 2$

Figure 6.5.3: Ghost penalty extension of the level set function on one extension layer, $\Omega_{\mathrm{Ex}} = \mathcal{N}^1(\Omega_{\mathrm{Pr}})$.




(a) $k_g = 1$ (b) $k_g = 2$

Figure 6.5.4: Ghost penalty extension of the level set function on two extension layers, $\Omega_{\mathrm{Ex}} = \mathcal{N}^2(\Omega_{\mathrm{Pr}})$.

We observe convergence rates $\mathcal{O}(h^2)$ for the $L^2$-errors and $\mathcal{O}(h)$ for the $h^1$-errors in the case $k_g = 1$. For $k_g = 2$ we observe convergence rates $\mathcal{O}(h^3)$ for the $L^2$-errors and

$\mathcal{O}(h^2)$ for the $h^1$-errors. These (optimal) orders of convergence are consistent with the error analysis in Theorems 6.5.3 and 6.5.4. Note that the errors both for the $L^2$- and $H^1$-projection are very similar.

Increasing the polynomial degree up to $k_g = 4$ in the extension method, we observe errors close to machine accuracy ($\sim 10^{-11}$) for the $L^2$– and the $H^1$–projection. Our method is (close to) exact in this setting, since the exact level set function is a polynomial of degree four in this example.

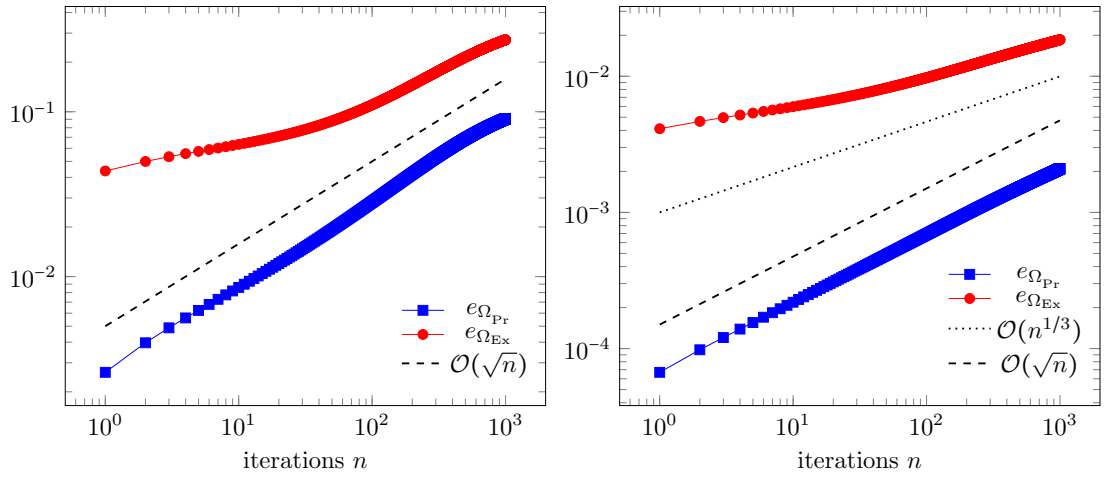
Now, we consider the repeated application of the extension method. We validate the theoretical results from Lemma 6.5.6 and the estimate (6.5.20). We start with the initial function $\phi_h^0 = \Pi_h^{k_g}\phi$ and determine $(\phi_h^n)_{\mathbb{N}} \subset V_{h,k_g}(\Omega_{\mathrm{Ex}})$ such that

$$(\phi_h^{n+1}, \psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\phi_h^{n+1}, \psi_h) = (\phi_h^n, \psi_h)_{\Omega_{\mathrm{Pr}}} \qquad \text{for all } \psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$$

holds. As in the experiments above we use the Kite-geometry and choose $\Omega_{\mathrm{Pr}} = \mathcal{N}^2(\mathcal{T}_\Gamma)$ and $\Omega_{\mathrm{Ex}} = \mathcal{N}^2(\Omega_{\mathrm{Pr}})$. The polynomial degrees in the finite element space are chosen as $k_g = 1$ and $k_g = 2$ and the mesh size is taken as $h = 2^{-4}$. The error quantities are defined as

$$e_{\Omega_{\mathrm{Pr}}} = \frac{\|\phi_h^n - \phi_h^0\|_{\Omega_{\mathrm{Pr}}}}{\|1\|_{\Omega_{\mathrm{Pr}}}}, \quad e_{\Omega_{\mathrm{Ex}}} = \frac{\|\phi_h^n - \phi_h^0\|_{\Omega_{\mathrm{Ex}}}}{\|1\|_{\Omega_{\mathrm{Ex}}}},$$

The results for $0 \le n \le 1000$ are shown in Figure 6.5.5.




(a) $k_g = 1$ (b) $k_g = 2$

Figure 6.5.5: Long–time behavior of extension method: Fixed mesh size $h$ and varying iteration number $n$.

For $k_g = 1$ we observe for $e_{\Omega_{\mathrm{Pr}}}$ a $\sqrt{n}$ dependence on $n$, consistent with (6.5.20). For $k_g = 2$ it seems that the dependence is milder.

In the next experiment, we test the robustness of the extension procedure with respect to the weight parameter $\gamma^{\mathrm{ext}}$. We present results using the 2D version of the Kite-geometry, defined by the zero level of the level set function $\phi(\mathbf{x}) := (x_1 + x_2^2)^2 + x_2^2 - 1$. The ghost penalty technique is employed to extend the Oswald-type interpolation of $\phi$ from $\Omega_{\mathrm{Pr}} = \mathcal{N}^1(\mathcal{T}_\Gamma)$ to $\Omega_{\mathrm{Ex}} = \mathcal{N}(\Omega_{\mathrm{Pr}})$ using polynomial degree $k_g = 1$. In the presented results, the $L^2$–projection is used, but similar outcomes are observed with the $H^1$ projection.

We vary the parameter $\gamma^{\text{ext}}$ from 0.01 to 50, and investigate both the $L^2$– and the $h^1$–errors, that are shown in Figure 6.5.6.

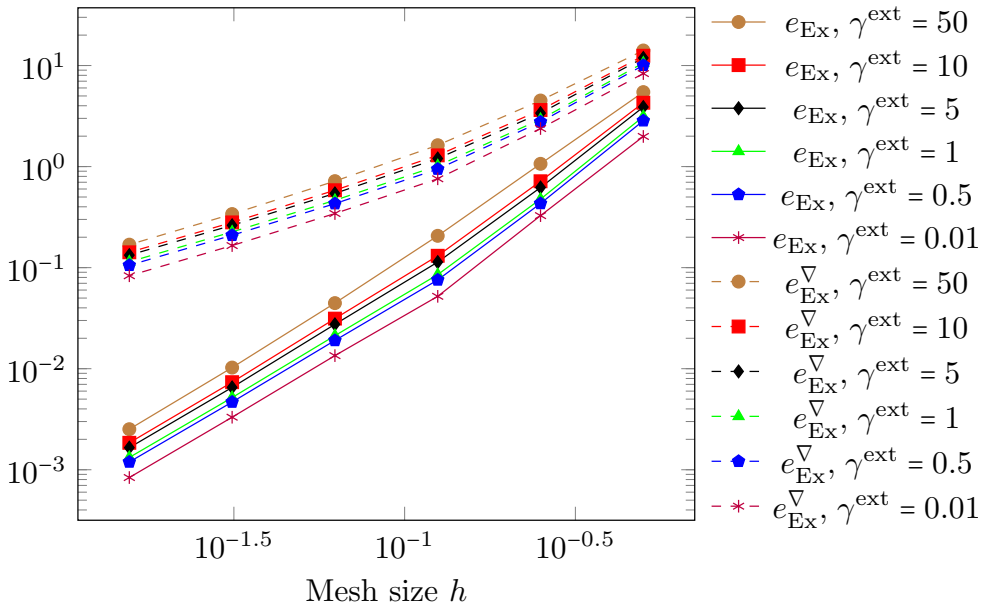



Figure 6.5.6: Different values for the weight parameter $\gamma^{\text{ext}}$ in the level set extension method.

We observe that the asymptotical behavior of our method does not depend on the choice of $\gamma^{\text{ext}} \in [0.01, 50]$. Furthermore, smaller values appear to perform slightly better than larger ones. For simplicity, we will use $\gamma^{\text{ext}} = 1$ for the rest of this chapter, as varying this parameter does not significantly enhance the extension solution.

## 6.6 A level set narrow band algorithm

We have introduced and described all ingredients needed in the narrow band level set transport as outlined in Algorithm 6.2.1. In this section, we discuss some implementation aspects and present more explanation on the method. For simplicity, we assume the velocity field $\mathbf{u}$ is given in a sufficiently large narrow band (cf. Remark 6.6.2).

An important aspect of the method is the choice of the domains $\Omega^n_{\text{Tr}}$ where the level set equation is solved in the time interval $[t_n, t_{n+1}]$ and the subdomain $\Omega^n_{\text{Pr}}$ where the projection in the extension method (6.5.2) is applied. We stick to transport domains $\Omega_{\text{Tr}}$ that consist of layers of elements around $\mathcal{T}^n_\Gamma$, the domain formed by the tetrahedra cut by $\Gamma^n_h$, i.e., it holds $\Omega^n_{\text{Tr}} = \mathcal{N}^{L_T}(\mathcal{T}^n_\Gamma)$, with some $L_T > 0$. Since we use a fixed time step size $\Delta t$, we have to take $\Omega_{\text{Tr}}$ large enough such that $\Gamma^{n+1}_h$ (the zero level of $\phi_h(\cdot, t_{n+1})$) is included in $\Omega^n_{\text{Tr}}$ (see Fig. 6.2.1).

Motivated by the maximal displacement of $\Gamma$ in one time slab $I_n = [t_n, t_{n+1}]$ given by $\Delta t \max_{t \in I_n} \|V_\Gamma\|_{L^\infty(\Gamma(t))}$, we choose $L_T \approx 1 + 4\frac{\Delta t}{h}\|V_\Gamma\|_{L^\infty(\Omega_\Gamma)}$ but at least $L_T = 3$. Here, $V_\Gamma$ denotes the surface velocity (in normal direction). Since the flow $\mathbf{u}$ is defined in a narrow band of the zero level set, $V_\Gamma$ is defined on $\mathcal{T}_\Gamma$. We use an approximation of $\|V_\Gamma\|_{L^\infty(\Omega_\Gamma)}$ instead of $\|V_\Gamma\|_{L^\infty(\Gamma)}$ since it is much easier to compute in our finite element software. In each step, we check if $L_T$ was taken large enough. If not, which happens only in

very rare cases, we redo the transport step with a larger transport domain (we add one layer). This procedure is repeated until the condition is satisfied. Using this strategy, we observe that $3 \leq L_T \leq 6$ holds in our experiments. Thus, we have few element layers in the transport domain and our method is efficient and applicable with $\mathbf{u}$ defined "close" to the zero level set.

At each time step, the level set equation is solved on a narrow band domain of width $\sim h$. The level set approximation at $t = t_{n+1}$, denoted by $\tilde{\phi}_h^{n+1}$ and defined on $\Omega_{\mathrm{Tr}}^n$, is then extended to $\phi_h^{n+1}$ defined on $\Omega_{\mathrm{Tr}}^{n+1}$ using the extension method (6.5.2). There are different possible choices for the projection domain $\Omega_{\mathrm{Pr}}^{n+1}$ in the extension method. Clearly the projection domain $\Omega_{\mathrm{Pr}}^{n+1}$ has to be a subdomain of both the current transport domain $\Omega_{\mathrm{Tr}}^n$ where $\tilde{\phi}_h^{n+1}$ is given and the next transport domain $\Omega_{\mathrm{Tr}}^{n+1}$ where $\phi_h^{n+1}$ has to be defined. Thus, the maximal possible projection domain is $\Omega_{\mathrm{Pr}}^{n+1} = \Omega_{\mathrm{Tr}}^n \cap \Omega_{\mathrm{Tr}}^{n+1}$. However, this choice leads to suboptimal results in the numerical experiments. The reason is that we do not have access to exact boundary data on the inflow boundary $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$ of $\Omega_{\mathrm{Tr}}^n$. This is a key challenge in narrow band level set methods: obtaining accurate numerical boundary data on the inflow boundary. An inaccuracy of boundary data at $t = t_n$ on the inflow boundary $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$ can significantly affect the solution of the level set equation in $\Omega_{\mathrm{Tr}}^n \times [t_n, t_{n+1}]$. However, due to the transport nature of the flow, such errors remain (essentially) on the same level line. For instance, an error at a point $\mathbf{z}$ on $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$ with $\phi(\mathbf{z}, t_n) = ch$ is transported along a trajectory $\mathbf{z}(t)$ with $\phi(\mathbf{z}(t), t_n) = ch$ for $t \in [t_n, t_{n+1}]$. Therefore, these boundary errors do not enter the domain $\{\mathbf{z} \in \Omega_{\mathrm{Tr}}^n \mid |\phi(\mathbf{z}, t_{n+1})| < ch\}$. Although this property does not hold exactly for the discrete approximation $\phi_h$, it is expected to hold approximately for an accurate discretization of the level set equation. Motivated by this observation, we use a *smaller* projection domain $\Omega_{\mathrm{Pr}}^{n+1}$ than the maximal one $\Omega_{\mathrm{Tr}}^n \cap \Omega_{\mathrm{Tr}}^{n+1}$.

Obvious candidates are the domain formed by elements cut by $\Gamma_h^{n+1}$, namely $\mathcal{T}_\Gamma^{n+1}$, the "cut" elements and their direct neighbors $\mathcal{N}^{L_P}(\mathcal{T}_\Gamma^{n+1})$ with some $L_P < L_T$, or the smallest set of simplices containing a level set $h$-tube $\{\mathbf{x} \in \mathbb{R}^d \mid |\tilde{\phi}_h^{n+1}(\mathbf{x})| \leq ch\}$, with some $c > 0$. In the numerical experiments, we use $\Omega_{\mathrm{Pr}} = \mathcal{N}^{L_T}(\mathcal{T}_\Gamma^{n+1})$, since this domain is independent of the scaling of $\phi_h$ and is easy to compute. Important effects of this choice of the projection domain are illustrated in numerical experiments in Section 6.7, cf. Figure 6.7.9.

If a BDF(R) scheme with $R > 1$ is used, we do not have enough input level set approximations for the BDF scheme in the first $R-1$ time steps. In this case, we use lower order BDF schemes with smaller time steps for the first $R-1$ time steps. This is a common approach in practice and is also used in the numerical experiments in Section 6.7.

An important feature of this approach is that higher order approximations can be easily achieved by using a higher order time and space discretization method for the level set equation and a higher order finite element space in the extension method. Although no rigorous error analysis of the entire algorithm is available yet, rigorous error bounds for the three components (numerical boundary data, space and time discretization of the level set equation, and the extension method) that form the algorithm have been derived.

Note that in this approach, we do *not* use any re-initialization of the level set function. In numerical experiments (see Section 6.7), we observed that for cases with smoothly evolving level sets, our approach appears to work satisfactory even without re-initialization.

Summarizing, we obtain Algorithm 6.6.1.

**Algorithm 6.6.1: Narrow band level set transport**

1: Parameters: $k_g$, $\Delta t$, $h$, $L_P$
2: $\phi_h^0 \leftarrow$ Initialization
3: **Procedure**
4: **for** $n$ from 1 to $N$ **do**:
5: $L_T \leftarrow \max\left\{3, [1 + 4\frac{\Delta t}{h}\|V_\Gamma\|_{L^\infty(\Omega_\Gamma)}]\right\}$
6: $\Omega_{\mathrm{Tr}}^n \leftarrow \mathcal{N}^{L_T}(\mathcal{T}_{\tilde{\Gamma}}^n)$
7: **if** $n > 1$ **then**
8: $\Omega_{\mathrm{Pr}}^n \leftarrow \mathcal{N}^{L_P}(\mathcal{T}_{\tilde{\Gamma}}^n)$
9: $\phi_h^n \leftarrow$ extension step (6.5.2a) of $\tilde{\phi}_h$ to $\Omega_{\mathrm{Tr}}^n$
10: **for** $i$ from 1 to $R-1$ **do** ▷ for BDF(R) method
11: $\phi_h^{n-i} \leftarrow$ extension step (6.5.2a) of $\phi_h^{n-i}$ to $\Omega_{\mathrm{Tr}}^n$
12: **end for**
13: **end if**
14: $\phi_D^n \leftarrow$ determine boundary data on $\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}$, e.g., (6.3.2) for BDF(2)
15: $\tilde{\phi}_h \leftarrow$ transport step on $\Omega_{\mathrm{Tr}}^n \times [t_n, t_{n+1}]$ using (6.4.4) and project into $V_{h,k_g}(\Omega_{\mathrm{Tr}}^n)$
16: **if** $\mathcal{N}^{L_P}(\mathcal{T}_{\tilde{\Gamma}}^n) \not\subset \Omega_{\mathrm{Tr}}^n$ **then**
17: $\Omega_{\mathrm{Tr}}^n \leftarrow \mathcal{N}(\Omega_{\mathrm{Tr}}^n)$, redo extension if $n > 1$ and go to line 14
18: **end if**
19: **end for**
20: **end Procedure**

We comment on the algorithm. In line 5, we use the rounding operator $[\cdot]$, since $L_T$ has to be an integer. The extension step in lines 7 to 13 is not used in the first time step, since $\phi_h^0$ is given on the whole computational domain. Since $\phi_h^n$ will be extended to $\Omega_{\mathrm{Tr}}^n$ in line 9 in time step $n$, it has to be extended once again to $\Omega_{\mathrm{Tr}}^{n+1}$ in the following step if a BDF(R) scheme with $R > 1$ is used. That is why the extension of the "old" level set approximations $\phi_h^{n-i}$ in line 10 to 12 is needed. In line 16, we check if the transport domain was large enough. If not, we add one layer of elements to the transport domain and redo the transport step.

**Remark 6.6.2 (Extrapolation of the velocity)**

In this chapter, we assume the flow field $\mathbf{u}$ is known in the narrow band, but this is usually not the case in practice. The velocity field will be determined by a surface Navier–Stokes equation, in the next chapter. We will use TraceFEM for the discretization of the velocity, which will lead to an approximation of the velocity field known in a narrow band. In such cases, there is a strong coupling between the PDE on the surface (which depends on $\Gamma$ and determines $\mathbf{u}$) and the level set equation (which depends on $\mathbf{u}$ and determines $\Gamma$). To obtain a reasonable approximation of $\mathbf{u}^{n+1}$ for use in (6.4.4), one can extrapolate the velocity in time based on previous time steps, as in [EGK22]. If

the velocity is defined in a narrow band around the surface (e.g., from TraceFEM), the extension method (6.5.2) can extend it to a wider band, or an implicit extension can be used as suggested in [LOX18]. ◇

## 6.7 Numerical experiments with the narrow band algorithm

We give results of numerical experiments to illustrate the performance of the narrow band level set algorithm from Algorithm 6.6.1. All examples are implemented in `NGSolve/netgen`, cf. [Sch97; Sch14] with the add-on `ngsxfem`, cf. [Leh+].

We give numerical results for the level set equation (2.6.2) on narrow bands in 2D and 3D. If not stated otherwise the time interval is taken as $I = [0,1]$. To obtain optimal order convergence, we use BDF(R) schemes with $R = k_g + 1$ for the time discretization. To get sufficiently good initial values for the BDF(2) or BDF(3) scheme, we initialize $\phi_h^0$ as an interpolation of the exact level set function and start with BDF(1) time steps with a sufficiently small time step size. If not stated otherwise we start with 10 BDF(1) time steps with time step size $\frac{\Delta t}{10}$, and then switch to the BDF(R) scheme with time step size $\Delta t$.

In case of the BDF(2) method, for the inflow boundary condition we use (6.3.2). If BDF(3) is used, we apply an analogous time extrapolation using the values at $t \in \{t_n, t_{n-1}, t_{n-2}\}$.

The narrow band scheme is applied on the cut elements with $L_T$ layers of neighboring elements added, $\Omega_{\mathrm{Tr}}^n := \mathcal{N}^{L_T}(\mathcal{T}_\Gamma^n)$, where $L_T$ depends on $\Delta t$, $h$, and $\|V_\Gamma\|_{L^\infty(\Omega_\Gamma)}$. For the projection domain in the extension method, we use $\Omega_{\mathrm{Pr}}^n = \mathcal{N}^{L_P}(\mathcal{T}_\Gamma^n)$, unless stated otherwise. We use $L_P \in \{1,2\}$. At each time step, we check that the projection elements for the next time step are contained in $\Omega_{\mathrm{Tr}}^n$. For the ghost penalty extension, we use the $L^2$-projection with $\alpha = 0$ and $\gamma^{\mathrm{ext}} = 1$.

We are interested in the error quantities

$$(e_\Gamma)^2 := \Delta t \sum_{n=1}^{N} \oint_{\Gamma_h^n} (\phi^n)^2 \,\mathrm{d}s, \qquad (e_{\Omega_{\mathrm{Tr}}})^2 := \Delta t \sum_{n=1}^{N} \oint_{\Omega_{\mathrm{Tr}}^n} \left(\phi^n - \phi_h^n\right)^2 \mathrm{d}\mathbf{x},$$
$$\text{and} \quad e_\Gamma^\infty = \max_{1\le n\le N} \|\phi^n\|_{L^\infty(\Gamma_h^n)},$$

where $\|\phi^n\|_{L^\infty(\Gamma_h^n)}$ is an approximation of the maximal (absolute) value of the exact level set function $\phi^n$ on $\Gamma_h^n$.

Note that the integrals over $\Gamma_h^n$ cannot be computed exactly when using polynomials of degree $k_g > 1$, and standard quadrature rules cannot be applied. For this reason, we use the so-called parametric approach described in Chapter 4 to approximate the error quantities with sufficiently high accuracy. The quantities $e_\Gamma$ and $e_\Gamma^\infty$ are a measure for the approximation error in $\Gamma_h^n \approx \Gamma(t_n)$.

We present results for the following test cases:

- *Deforming kite.* In this study, we investigate the deformation of a kite geometry into a sphere in three-dimensional space (denoted as kite($3D$)) and into a circle in two-dimensional space (denoted as kite($2D$)), as illustrated in Figure 6.7.1. We present results for linear polynomials combined with a BDF(2) scheme, and the

BDF(3) scheme combined with $k_g = 2$. Additionally, for the case of kite($2D$) using BDF(2) and $k_g = 1$, we analyse the influence of varying inflow boundary conditions on the results. Furthermore, we investigate the differences between using a normal flow field ($\mathbf{u}_T = 0$) and a flow field with a tangential component ($\mathbf{u}_T \neq 0$).

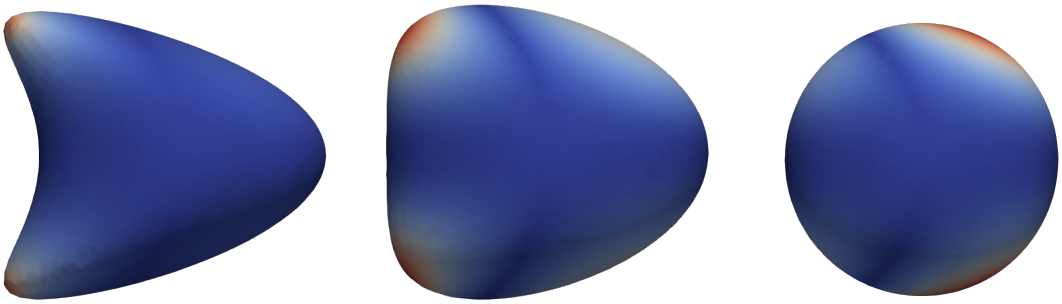
- *Ball in a rotating flow field.* We examine a spherical object in a rotating flow field, represented in both three dimensions (sphere) and two dimensions (circle), which completes a full rotation as depicted in Figure 6.7.6. The analysis includes error quantities for BDF(2) with $k_g = 1$ and BDF(3) with $k_g = 2$. In the two-dimensional scenario, we analyse the implications of using a larger projection domain in the extension procedure. In contrast, for the three-dimensional scenario, we focus on investigating properties related to volume conservation.

- *Strongly deforming sphere.* Lastly, we explore a more complex and less smooth example involving a sphere subjected to significant deformation into a thin tube due to vortex flow dynamics. Various configurations are considered, and accuracy results are presented for both BDF(2) with $k_g = 1$ and BDF(3) with $k_g = 2$.

The computational domain is chosen as $\Omega = [-\frac{7}{3}, \frac{7}{3}]^d$ for the first two geometries and as $\Omega = [0, 1]^3$ for the deforming sphere example.

**Deforming kite (2D, 3D)**

We start with the deformation of a "Kite"-geometry into a sphere (in 3D) that was also used in Section 6.5.3. The exact level set function of the kite is given by $\phi(\mathbf{x}, t = 0) = (x_1 + x_2^2)^2 + x_2^2 - 1$ in 2D and $\phi(\mathbf{x}, t = 0) = (x_1 - x_3^2)^2 + x_2^2 + x_3^2 - 1$ in 3D. We use a convex combination to deform the kite into a circle (or a sphere) with level set function $\phi(\mathbf{x}, t = 1) = x_1^2 + x_2^2 - 1$, or $\phi(\mathbf{x}, t = 1) = x_1^2 + x_2^2 + x_3^2 - 1$ in 2D and 3D, respectively. The geometry evolution for the 3D case is illustrated in Figure 6.7.1.



(a) $t = 0$ (b) $t = 0.5$ (c) $t = 1$

Figure 6.7.1: Snapshots of Kite geometry deforming into a sphere. The coloring shows the values of $|u_N|$ from blue (low) to red (high).

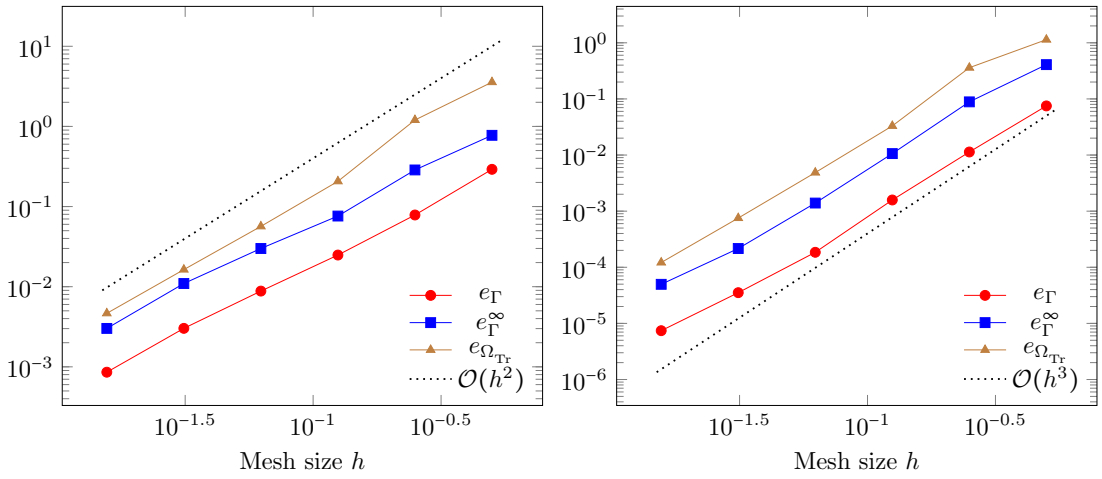
The velocity field $\mathbf{u}$ used in the level set equation has a normal component that is determined by the level set evolution. From the transport equation (2.6.2) we get

$$\mathbf{u} \cdot \mathbf{n} = V_\Gamma = -\frac{\partial \phi / \partial t}{\mathbf{n} \cdot \nabla \phi}.$$

To illustrate that our approach is not restricted to cases with normal velocity fields, we add a tangential component $\mathbf{u}_T$ to $V_\Gamma \mathbf{n}$ to obtain the full flow field $\mathbf{u}$. The tangential

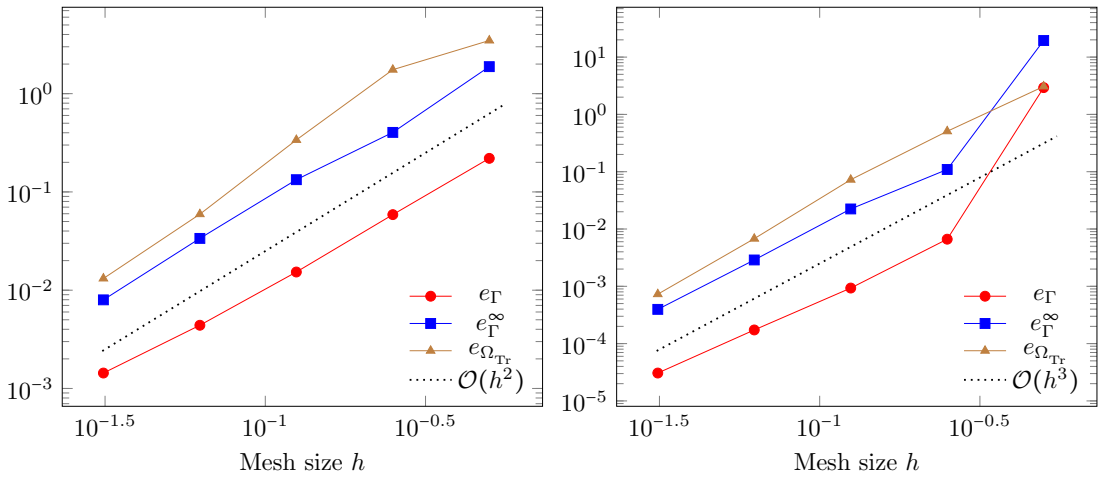
component of the velocity does not change the evolution of the level set function in (2.6.2). Thus, it should not change the accuracy of the level set approximation. We use $\mathbf{u}_T = \mathbf{curl}_\Gamma(x_1x_2x_3)$ as a tangential component with the curl operator defined as in [BR20; Reu18] in 3D and the tangential component of $(x_2, -x_1)^T$ in the 2D case.

The mesh size start as $h_0 = 0.5$, the initial time step size is taken as $\Delta t_0 = 0.1$. The mesh size $h$ and the time step size $\Delta t$ are halved it in each refinement step. For the projection domain, we use one layer of neighboring elements to the cut elements, i.e., $\Omega^n_{\mathrm{Pr}} = \mathcal{N}(\mathcal{T}^n_\Gamma)$. The results are shown in Fig. 6.7.2 for the 2D case and in Fig. 6.7.3 for the 3D case.




(a) BDF(2), $k_g = 1$ (b) BDF(3), $k_g = 2$

Figure 6.7.2: Kite deforming to a circle in 2D.




(a) BDF(2), $k_g = 1$ (b) BDF(3), $k_g = 2$

Figure 6.7.3: Kite deforming to a sphere in 3D.

We observe convergence rates that are close to the optimal rates of $\mathcal{O}(h^2)$ for $k_g = 1$ and $\mathcal{O}(h^3)$ for $k_g = 2$ for all error measures in two and three dimensions.

We examine the effect of the choice of the inflow boundary data, cf. Section 6.3. Constructing higher order boundary data is not straightforward. Since we are interested in

the influence of the boundary data on the accuracy of the surface approximation and not of the level set approximation in the whole narrow band, the more important error measure is $e_\Gamma$, but we also display the errors $e_{\Omega_{\text{Tr}}}$. We consider the kite geometry that deforms to a circle in $2D$ with BDF(2) scheme, $k_g = 1$, and $L_P = 1$. For the boundary data we use the first order (in time) boundary data from (6.3.1), the second order boundary values from (6.3.2) and a fourth order version of the boundary data similar to (6.3.3). The results are shown in Table 6.7.4.

| | $e_\Gamma$ | | | $e_{\Omega_{\text{Tr}}}$ | | |
|---|---|---|---|---|---|---|
| $h$ | Low | Medium | High | Low | Medium | High |
| 0.5 | 0.16 | 0.16 | 0.17 | 2.78 | 2.82 | 3.87 |
| 0.25 | $5.63 \cdot 10^{-2}$ | $5.63 \cdot 10^{-2}$ | $5.64 \cdot 10^{-2}$ | 1.04 | 1.06 | 2.84 |
| 0.125 | $1.78 \cdot 10^{-2}$ | $1.78 \cdot 10^{-2}$ | $1.78 \cdot 10^{-2}$ | 0.18 | 0.18 | 0.18 |
| 0.0625 | $7.46 \cdot 10^{-3}$ | $7.45 \cdot 10^{-3}$ | $7.47 \cdot 10^{-3}$ | $5.05 \cdot 10^{-2}$ | $5.08 \cdot 10^{-2}$ | $5.06 \cdot 10^{-2}$ |
| 0.0313 | $2.61 \cdot 10^{-3}$ | $2.61 \cdot 10^{-3}$ | $2.61 \cdot 10^{-3}$ | $1.5 \cdot 10^{-2}$ | $1.51 \cdot 10^{-2}$ | $1.5 \cdot 10^{-2}$ |
| 0.0156 | $7.84 \cdot 10^{-4}$ | $7.84 \cdot 10^{-4}$ | $7.83 \cdot 10^{-4}$ | $4.39 \cdot 10^{-3}$ | $4.42 \cdot 10^{-3}$ | $4.4 \cdot 10^{-3}$ |

Figure 6.7.4: Errors using different boundary values for level set equation of a kite deforming to a circle.

We observe almost no difference in both errors for all choices of the boundary values except that the error in the narrow band $e_{\Omega_{\text{Tr}}}$ is worse for the higher order boundary data for the first two mesh refinement levels. The larger errors for the higher order boundary data may occur because these boundary values are only higher order accurate in time when the previous level set approximations are sufficiently accurate in the $H^1$–norm in space. In contrast to the medium and lower order boundary values, the gradient of the level set approximations from previous time steps are used for the higher order boundary values. We expect that for the relatively coarse mesh sizes, these gradients are not accurate enough and thus, the boundary values are worse than the choices independent of the gradients, even if they are higher order in time. The results demonstrate that due to the choice of a suitable (sufficiently small) projection domain in the extension method we obtain higher order accuracy in the surface approximation even with low order accurate inflow boundary data. Based on these results, we will stick to the "medium" order boundary data (6.3.2) for the rest of the chapter.

As a last experiment for the kite geometry, we investigate the robustness of the method with respect to a change in the tangential component of the flow field. The exact transport equation (2.6.2) is independent of the tangential component of the flow field, since $\mathbf{u}_T \cdot \nabla\phi = 0$ holds, where $\mathbf{u}_T$ denotes the tangential component of the velocity, cf. (2.1.1). Thus, we expect our method to be robust with respect to the choice of the tangential component. We use the kite geometry that deforms to a circle in $2D$ with BDF(2) scheme, $k_g = 1$, and $L_P = 1$, but similar results hold for higher order approximations. We compare the results for a normal flow field ($\mathbf{u}_T = 0$) and a flow field with a tangential component ($\mathbf{u}_T \neq 0$). The tangential component is taken as in the previous experiments. The norms of the different flows on the surface read

$$\sqrt{\Delta t \sum_{n=1}^{N} \oint_{\Gamma_h^n} \mathbf{u} \cdot \mathbf{u} \, \mathrm{d}s} \approx 0.98 \qquad \text{and} \qquad \sqrt{\Delta t \sum_{n=1}^{N} \oint_{\Gamma_h^n} \mathbf{u}_N \cdot \mathbf{u}_N \, \mathrm{d}s} \approx 0.25.$$

Thus, we see that the tangential component has a significant impact on the $L^2$–norm of the flow field. The errors on the surface and in the transport domain are shown in Figure 6.7.5.

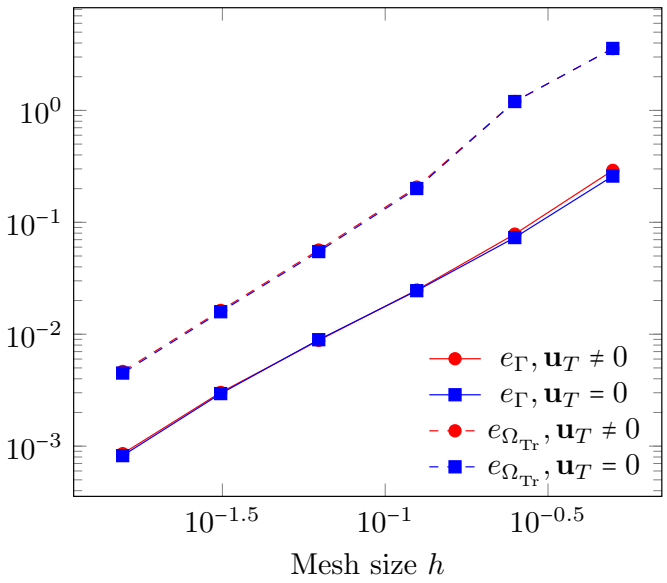
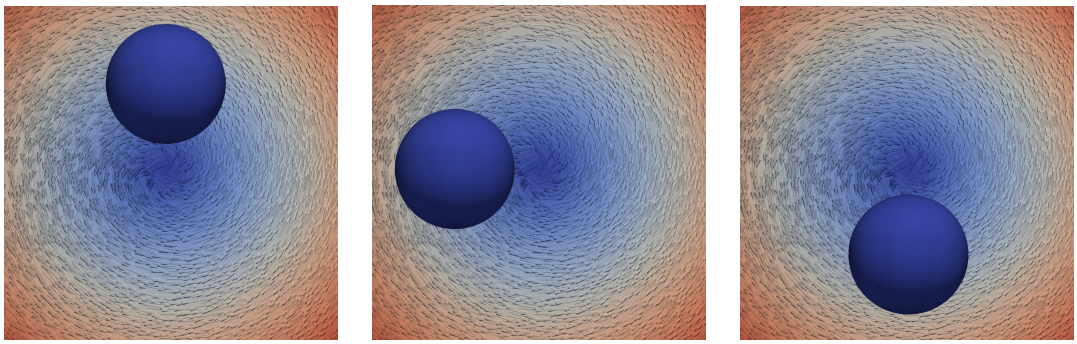



Figure 6.7.5: Errors in the level set approximation of the Kite geometry with $k_g = 1$ using different flow fields.

We observe that the errors are almost identical for both flow fields. This is in accordance with our expectations, since the tangential component of the flow field does not influence the evolution of the level set function. The results indicate that our method is robust with respect to the choice of the tangential component of the flow field.

**Ball in a rotating flow field (2D, 3D)**

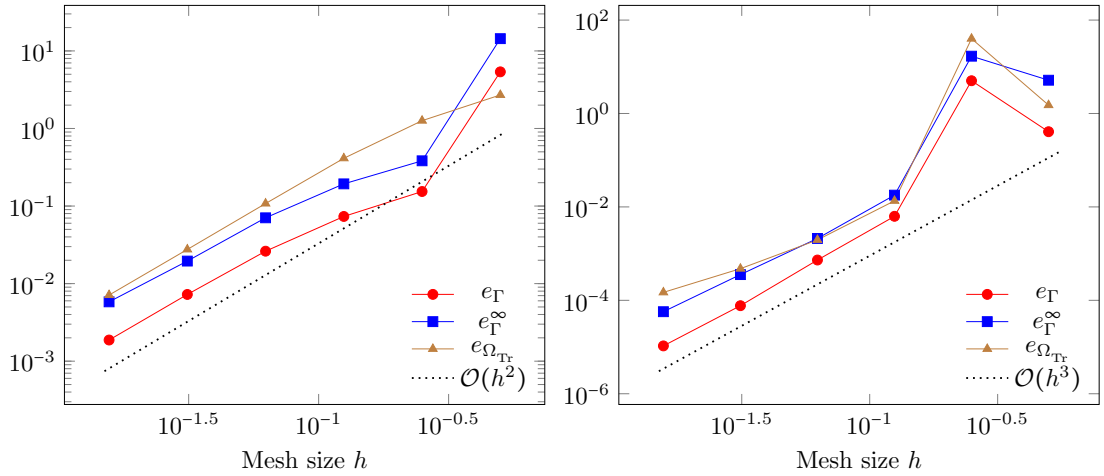
As the next geometry example, we consider a sphere (or circle) that makes a full rotation around the origin for $t \in [0, 1]$. The velocity field of the rotation is given by $\mathbf{u}(\mathbf{x}) = 2\pi(-x_2, x_1)^T$ in 2D and $\mathbf{u}(\mathbf{x}) = 2\pi(-x_2, x_1, 0)^T$ in 3D. It is independent of the time $t$. The exact level set function reads $\phi(\mathbf{x}, t) = (x_1 - \cos(2\pi t))^2 + (x_2 - \sin(2\pi t))^2 - \frac{1}{2}$ in 2D and is given by $\phi(\mathbf{x}, t) = (x_1 - \cos(2\pi t))^2 + (x_2 - \sin(2\pi t))^2 + x_3^2 - \frac{1}{2}$ in 3D. The geometry evolution is shown in Figure 6.7.6 in 3D.



(a) $t = 0.25$ (b) $t = 0.5$ (c) $t = 0.75$

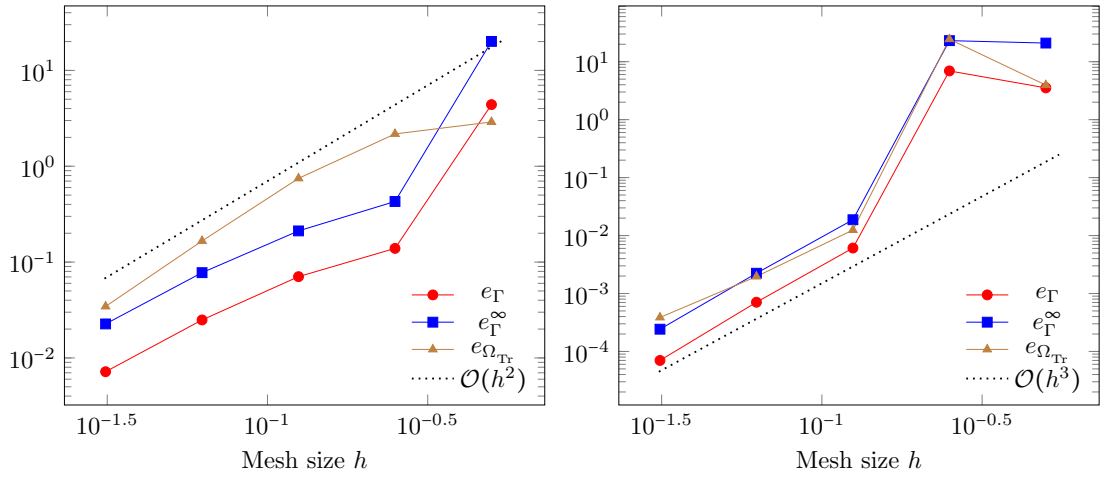
Figure 6.7.6: Snapshots of a sphere making a full rotation around the origin.

We start with a test of the convergence rates in 2D and 3D. The coarsest mesh and time step sizes are $h_0 = 0.5$ and $\Delta t = 0.1$, and we halve both in each refinement step. Again, we use the low order level set approximation $k_g = 1$ with BDF(2) and the higher order approximation $k_g = 2$ with a BDF(3) for the time discretization. The projection domain is chosen as $\Omega^n_{\mathrm{Pr}} = \mathcal{N}^{L_P}(\mathcal{T}^n_\Gamma)$ with two layers of elements added ($L_P = 2$). The results are shown in Fig. 6.7.7 for the 2D case and in Fig. 6.7.8 for the 3D case.




(a) $k_g = 1$ (b) $k_g = 2$

Figure 6.7.7: Errors of level set approximation of a circle rotating around the origin in 2D.



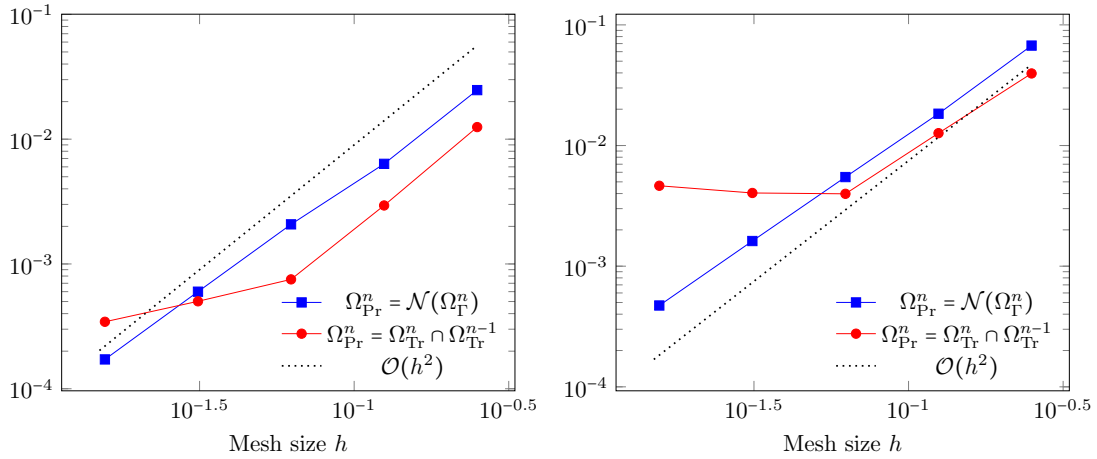

(a) $k_g = 1$ (b) $k_g = 2$

Figure 6.7.8: Errors of level set approximation of a sphere rotating around the origin in 3D.

We observe that the asymptotic error behavior is quite similar for one dimensional and two dimensional surfaces (embedded in $\mathbb{R}^2$ or $\mathbb{R}^3$). In all simulations, we get optimal convergence rates of $\mathcal{O}(h^2)$ for linear polynomials ($k_g = 1$) combined with BDF(2) and $\mathcal{O}(h^3)$ for quadratic polynomials ($k_g = 2$) combined with BDF(3) for all error measures after the second mesh refinement step. On the second mesh level ($h = 0.25$, $\Delta t = 0.05$), we observe smaller errors than expected for linear polynomials ($k_g = 1$) and higher errors

for quadratic polynomials ($k_g = 2$). It seems that this behavior is pre asymptotic and the optimal convergence rates are reached after the second refinement, leading to smaller errors for higher order level set approximations. The results indicate that the method also yields satisfactory results for a stronger movement of the surface.

In the next experiment, we investigate how a "too large" projection domain in the ghost penalty extension method affects the accuracy of the discretization method. We compare results where the projection domain consists of one layer added to the cut elements ($\Omega_{\mathrm{Pr}} = \mathcal{N}^1(\mathcal{T}_\Gamma)$) with the maximal possible projection domain ($\Omega^n_{\mathrm{Pr}} = \Omega^n_{\mathrm{Tr}} \cap \Omega^{n+1}_{\mathrm{Tr}}$). Again, we use first order polynomials in space ($k_g = 1$) and a BDF(2) method in time. We use a smaller initial time step of $\Delta t_0 = 0.05$ and a small time interval with $I = [0, 0.1]$. Increasing the time step size or the end time, leads to a break of our code when the maximal projection domain is used. The surface errors ($L^2$- and $L^\infty$-errors) for the 2D case are shown in Figure 6.7.9.




(a) $L^2$ surface errors $e_\Gamma$

(b) $L^\infty$ surface errors $e^\infty_\Gamma$

Figure 6.7.9: Errors of level set approximation of a circle rotating around the origin in 2D while using different projection domains in the extension method with $k_g = 1$.

We observe that using the maximal projection domain ($\Omega^n_{\mathrm{Pr}} = \Omega^n_{\mathrm{Tr}} \cap \Omega^{n+1}_{\mathrm{Tr}}$) leads to less accurate results, and we do not obtain an (almost) optimal rate of convergence in this case. After two refinement steps, we even lose convergence for both measured errors. This supports our choice of $\Omega_{\mathrm{Pr}} = \mathcal{N}^{L_P}(\mathcal{T}_\Gamma)$ for the projection domain in the extension method. We believe that this behavior arises because the errors caused by inexact inflow boundary values are transported inside the transport domain and the extension with a small projection domain neglects these errors, while the choice of the maximal projection domain does not. For more explanation, see the heuristics discussed in Section 6.6.

In the final experiment, we investigate volume conservation in 3D. Since the velocity field that rotates the sphere is divergence-free ($\operatorname{div} \mathbf{u} = 0$), the volume enclosed by the surface is conserved over time in the continuous problem. We measure the enclosed volume of the numerical approximation $\Gamma_h(t_n)$ in each time step for the lower ($k_g = 1$) and higher ($k_g = 2$) order narrow band scheme with mesh size $h = 0.0625$. The results are shown in Figure 6.7.10, and show that using higher order polynomials ($k_g = 2$), leads to a strong increase in the accuracy of the volume conservation. Note that we do not enforce volume conservation by a Lagrange multiplier or a similar method.

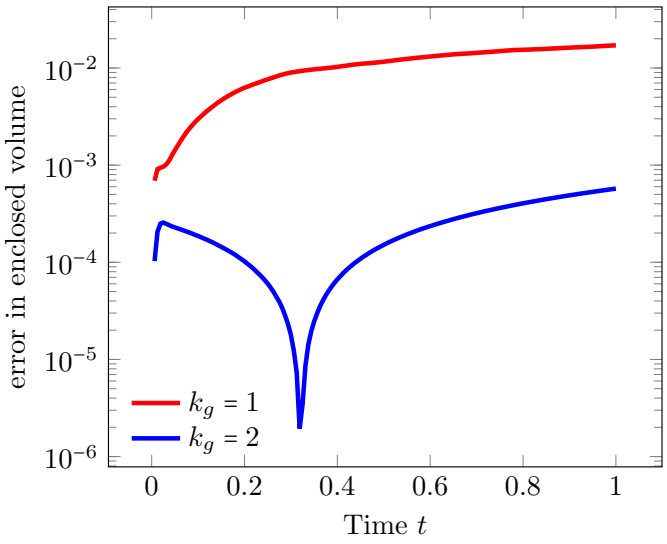



Figure 6.7.10: Errors in enclosed volume of level set approximation of a sphere rotating around the origin in 3D.

### Strongly deforming sphere (3D)

As a last geometry, we consider a sphere that is subjected to significant deformation into a thin tube due to vortex flow dynamics. As an initial surface we take a sphere with radius 0.15 centered at $(0.5, 0.75, 0.5)$. A corresponding level set function is given by $\phi(\mathbf{x}, t = 0) = (x_1 - 0.5)^2 + (x_2 - 0.75)^2 + (x_3 - 0.5)^2 - 0.15^2$. The sphere is transported and deformed by the velocity field that given by

$$\mathbf{u}(\mathbf{x}, t) = \cos\left(\frac{\pi}{t_{\text{end}}} t\right) \begin{pmatrix} -2\sin(\pi x_1)^2 \sin(\pi x_2)\cos(\pi x_2) \\ 2\sin(\pi x_2)^2 \sin(\pi x_1)\cos(\pi x_1) \\ 0 \end{pmatrix}, \quad t \in [0, t_{\text{end}}].$$

Due to the periodic $t$-behavior of this velocity field the exact surface will deform until $t = t_{\text{end}}/2$ is reached and then return to the initial spherical shape at $t = t_{\text{end}}$. Thus, the level set function satisfies $\phi(\mathbf{x}, t = 0) = \phi(\mathbf{x}, t = t_{\text{end}})$. The surface exhibits a spiral-like deformation. Increasing $t_{\text{end}}$ results in stronger deformed shapes of $\Gamma(t)$ at the time point of maximal deformation ($t = t_{\text{end}}/2$). Similar test cases are widely used to test interface capturing and interface tracking methods, cf. [DLP06; MRC06]. In Figure 6.7.11 we show resulting deformations of the sphere for $t_{\text{end}} = 2$ and $t_{\text{end}} = 4$.

(a) $t_{\text{end}} = 2$, $t = 0.5$ (b) $t_{\text{end}} = 2$, $t = 1$ (c) $t_{\text{end}} = 4$, $t = 1$ (d) $t_{\text{end}} = 4$, $t = 2$

Figure 6.7.11: Snapshots of the deformation of a sphere in a vortex.

We apply the transport scheme on the cut elements augmented by four layers of neighboring elements, while the projection domain in the extension method consists of the cut elements with two layers of neighboring elements added, i.e., $L_T = 4$ and $L_P = 2$. We use the BDF(2) and BDF(3) schemes combined with finite elements of degree $k_g = 1$ and $k_g = 2$, respectively. The time step size is taken as $\Delta t = h$. The initial mesh and time step sizes are taken as $h_0 = \Delta t_0 = 2^{-4} = 0.0625$ and are halved in each refinement step. In this example the exact level set function is not known for $t \notin \{0, t_{\text{end}}\}$. As a measure of accuracy we use

$$|e_{\Gamma,N}|^2 = \oint_{\Gamma_h(t_{\text{end}})} \phi^2 \, \mathrm{d}s,$$

which quantifies the deviation of the discrete zero level set from the spherical shape at the final time $t = t_{\text{end}}$. We present results of our method applied to this deforming sphere example for $t_{\text{end}} = 2$ in Figure 6.7.12.

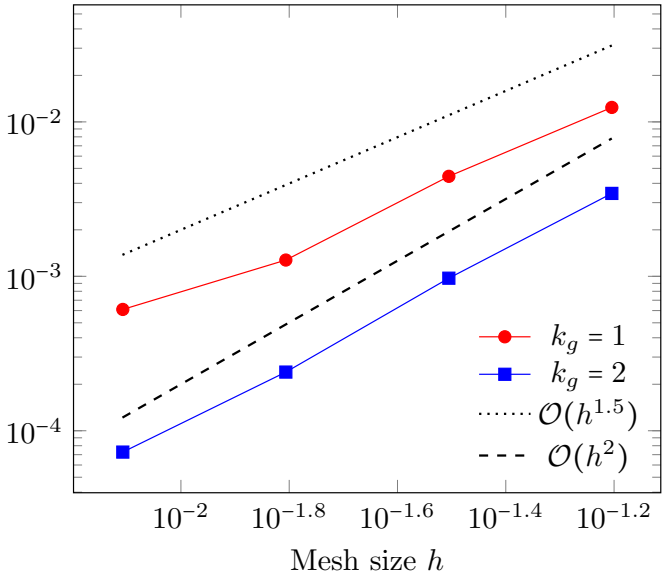



Figure 6.7.12: Surface error at the end time point $t_{\text{end}} = 2$ ($e_{\Gamma,N}$) for the strongly deforming sphere.

We observe a convergence rate of (approximately) $\mathcal{O}(h^{1.5})$ for $k_g = 1$ and second order convergence for $k_g = 2$.

We show the values of $e_{\Gamma,N}$ for different end times $t_{\text{end}}$ in Table 6.7.13. For certain combinations of end times $t_{\text{end}}$ and refinement levels, we cannot report errors due to insufficient mesh resolution (the surface vanishes in these examples).

| $h$ | $t_{\text{end}} = 1$ | | | | $t_{\text{end}} = 2$ | | | | $t_{\text{end}} = 3$ | | $t_{\text{end}} = 4$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $k = 1$ | | $k = 2$ | | $k = 1$ | | $k = 2$ | | $k = 1$ | | $k = 1$ | |
| | $e_{\Gamma,N}$ | rate | $e_{\Gamma,N}$ | rate | $e_{\Gamma,N}$ | rate | $e_{\Gamma,N}$ | rate | $e_{\Gamma,N}$ | rate | $e_{\Gamma,N}$ | rate |
| $6.3 \cdot 10^{-2}$ | $4.2 \cdot 10^{-3}$ | – | $2 \cdot 10^{-3}$ | – | $1.2 \cdot 10^{-2}$ | – | $3.4 \cdot 10^{-3}$ | – | – | – | – | – |
| $3.1 \cdot 10^{-2}$ | $1.5 \cdot 10^{-3}$ | 1.4 | $3.7 \cdot 10^{-4}$ | 2.5 | $4.4 \cdot 10^{-3}$ | 1.5 | $9.7 \cdot 10^{-4}$ | 1.8 | $9.9 \cdot 10^{-3}$ | – | $1.6 \cdot 10^{-2}$ | – |
| $1.6 \cdot 10^{-2}$ | $4 \cdot 10^{-4}$ | 2 | $6.3 \cdot 10^{-5}$ | 2.5 | $1.3 \cdot 10^{-3}$ | 1.8 | $2.4 \cdot 10^{-4}$ | 2 | $3.1 \cdot 10^{-3}$ | 1.7 | $6 \cdot 10^{-3}$ | 1.4 |
| $7.8 \cdot 10^{-3}$ | $1.3 \cdot 10^{-4}$ | 1.6 | $1 \cdot 10^{-5}$ | 2.7 | $6.1 \cdot 10^{-4}$ | 1.1 | $7.3 \cdot 10^{-5}$ | 1.7 | $1.1 \cdot 10^{-3}$ | 1.5 | $1.9 \cdot 10^{-3}$ | 1.6 |

Figure 6.7.13: Surface errors ($e_{\Gamma,N}$) for different final times $t_{\text{end}}$ for the deforming drop.

By taking sufficiently small $h$ and $k_g = 1$, we can handle the case $t_{\text{end}} = 4$ in which a very strong deformation occurs. For $t_{\text{end}} = 3$ and $t_{\text{end}} = 4$ we do not have results for

$k_g = 2$ since the zero level of the level set function leaves the transport domain $\Omega^n_{\mathrm{Tr}}$ for the given time step size $\Delta t = h$ and the algorithm breaks down. As expected, the errors are smaller for smaller $t_{\mathrm{end}}$ values. The convergence rates in the table vary between 1 and 2 for $k_g = 1$ and between 1.5 and 2.5 for $k_g = 2$.

The suboptimal rates in this experiment can be explained by the rather poor mesh resolution. We do not use an adaptive mesh refinement strategy and the mesh resolution is too low to resolve the strong deformations of the sphere. In Figure 6.7.14 we show a zoom of the narrow tail of the deformed sphere for $t_{\mathrm{end}} = 4$, $t \approx 2$ and $k_g = 1$.

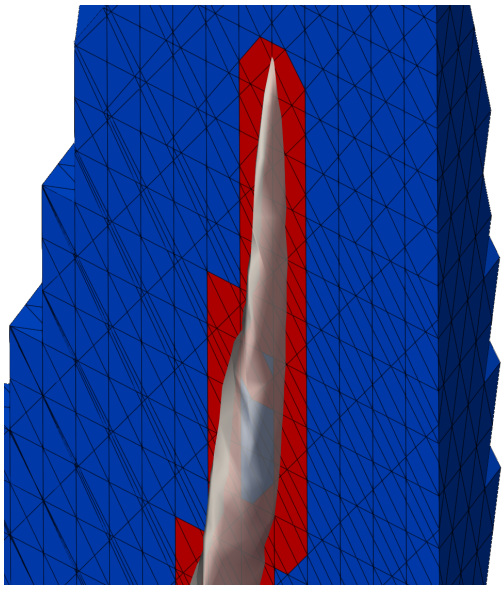


Figure 6.7.14: Zoom of the narrow tail of the deformed sphere, $h = 2^{-6}$.

The mesh shown is for $h = 2^{-6} \approx 0.016$ (second refinement level). The cut elements are marked in red and the transport elements in blue. As one can see from this figure, we have a poor resolution of the part of the surface that is most strongly deformed. Thus, the error in the surface approximation is therefore larger than in the previous experiments. To overcome this problem one could use an adaptive mesh refinement strategy, which is a topic of further research. On the other hand, one may conclude that even with this relatively poor resolution we obtain fairly good results, which indicates that our method has good robustness properties with respect to strong shape deformations.

# Chapter 7

# Discretization of Surface Navier–Stokes Equations

## 7.1 Introduction

In the final chapter of the thesis, we put all derived ingredients from previous chapters together to get an algorithm that discretizes the surface Navier–Stokes equations (SNS). We do not assume a known level set evolution or a known surface (normal) velocity and solve the Navier–Stokes system as a free surface problem. This is a highly nonlinear problem. Besides the nonlinearity occurring in the material derivative of the velocity which is easy to handle, the surface Navier–Stokes equations are highly nonlinear due to the coupling between the surface evolution and the velocity field. The surface Navier–Stokes PDE is posed on the evolving surface $\Gamma(t)$, but the evolution of the surface depends on the solution $\mathbf{u}$ of the SNS equations. Thus, there exists a strong coupling between the velocity $\mathbf{u}$ and the surface $\Gamma(t)$ making the surface Navier–Stokes system a difficult numerical challenge.

The level set equation is solved on a transport domain $\Omega_{\mathrm{Tr}}$ consisting of layers of neighboring elements of the cut elements. The resulting level set function is extended to the next transport domain, using a variant of the method introduced in Chapter 6. We introduce a new method to extend the velocity off the surface. The method is based on the volume normal derivative stabilization introduced in Chapter 4. The surface Navier–Stokes equations are solved using the TraceFEM method introduced in Chapter 4 combined with a BDF scheme to discretize the time derivative. We also introduce a method to construct a smooth normal approximation on the surface. The method is based on the $L^2$-projection of a discontinuous normal vector approximation and a ghost penalty stabilization. We validate our algorithm in various numerical experiments.

The remainder of this chapter is organized as follows: We recall the Navier-Stokes problem and give a short overview over the algorithm in Section 7.2. In the following sections, we introduce the velocity extension method (Section 7.3), the level set transport and extension (Section 7.4), the normal construction method (Section 7.5), and the TraceFEM formulation for the surface Navier-Stokes equations (Section 7.6). In Section 7.7, we summarize the full algorithm to solve the surface Navier-Stokes equations. Finally, we show convergence properties of the method in numerical experiments and illustrate physical phenomena of the numerical solution when no outer force term is acting in Section 7.8.

## 7.2 Problem formulation and structure of the method

In this section, we repeat the problem formulation of the surface Navier–Stokes equations and give an overview of the method to solve the system numerically. We outline the used algorithm and the main components of the method.

We recall the Navier-Stokes system from Problem 3.4.2: For initial values $\mathbf{u}^0$, $\Gamma^0$, viscosity coefficient $\mu_0$ and force term $\mathbf{f}$ find the velocity $\mathbf{u}$, the pressure $p$, and the parametrization of the surface $\mathbf{X}$ satisfying

$$\begin{cases} \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \nabla_\Gamma p - p\kappa\mathbf{n} = \mathbf{f} & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t), \\ \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(\mathbf{x},t) = \mathbf{u}(\mathbf{X}(\mathbf{x},t),t), \quad \mathbf{X}(\mathbf{x},0) = \mathbf{x} \in \Gamma^0. \end{cases} \tag{7.2.1}$$

In this system, the mean curvature $\kappa$ of the surface $\Gamma(t)$ occurs explicitly. This is a geometric quantity that is hard to approximate with higher order accuracy. For example, when a level set function $\phi_1 \in V_{h,1}$ of polynomial degree one is used, the canonical normal approximation $\mathbf{n}_1 = \frac{\nabla\phi_1}{\|\nabla\phi_1\|}$ is constant on each tetrahedron $T \in \mathcal{T}_h$. The simplest approximation of the mean curvature is then given by $\kappa_1 = \operatorname{div}_{\Gamma_h} \mathbf{n}_1 = 0$, since the mean curvature consists of second derivatives of the level set function. Thus, we do not have convergence of the mean curvature approximation when using the lowest order polynomials for the level set function. It is possible to construct the mean curvature vector $\kappa\mathbf{n}$ using Lemma 2.5.6 and a weak formulation. However, this approach needs careful stabilization and another linear system has to be solved in each time step increasing the computational costs.

For this reason, we reformulate the surface Navier–Stokes system to avoid the explicit use of the mean curvature. We use Lemma 2.5.5 that states

$$\operatorname{div}_\Gamma(p\mathbf{P}) = \nabla_\Gamma p - p\kappa\mathbf{n},$$

for a scalar function $p$. With this equation, we can rewrite the first equation from the Navier-Stokes system as

$$\dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \operatorname{div}_\Gamma(p\mathbf{P}) = \mathbf{f}.$$

As explained in Section 2.6, we use the level set function $\phi$ to describe the surface $\Gamma(t)$. The surface is then given by

$$\Gamma(t) = \{\mathbf{x} \mid \phi(\mathbf{x},t) = 0\}$$

and the last equation of the system is replaced by the level set equation describing the evolution of the level set function given by

$$\partial_t\phi + \nabla\phi \cdot \mathbf{u} = 0,$$

cf. Section 2.6.1 for more details. This equation requires the velocity field $\mathbf{u}$ to be defined on a neighborhood around a surface and not only on $\Gamma(t)$. The surface Navier-Stokes equations define the values of $\mathbf{u}$ on $\Gamma(t)$, and we have some freedom to define the

velocity field outside $\Gamma(t)$. It is known from the literature, that a constant velocity along normals to the surface is desirable for the level set equation, cf. e.g., [Set99a, Section 11.3] or [AS99]. More precisely, in [BS25] it is shown, that solving the level set transport equation with a velocity that is constant along the surface normal and a signed distance function as initial condition leads to a signed distance function as the solution at each time point. Thus, it holds

$$\begin{cases} \partial_t \phi + \nabla \phi \cdot \mathbf{u} = 0, \\ \phi(\mathbf{x}, 0) = d(\mathbf{x}, 0) \end{cases} \quad \Rightarrow \quad \phi(\mathbf{x}, t) = d(\mathbf{x}, t),$$

when the level set transport equation is solved exactly with a velocity field that is constant in normal direction. This motivates the choice of a constant extension along normals for $\mathbf{u}$. The extension can be formulated by the equation

$$\nabla \mathbf{u} \mathbf{n} = 0 \quad \text{in } \mathcal{O}(\Gamma),$$

with $\mathcal{O}(\Gamma)$ being a small tubular neighborhood around $\Gamma(t)$. When solving the Navier–Stokes system, we use a TraceFEM formulation with a volume normal derivative stabilization term added. This stabilization leads to a velocity field that is approximately constant along the normal vector approximation $\mathbf{n}_h$ on the cut domain $\Omega_\Gamma$. In Section 7.3, we introduce a method to extend the velocity from $\Omega_\Gamma$ to $\mathcal{O}(\Gamma)$. The level set function $\phi$ is then transported with this extended velocity field which is also denoted by $\mathbf{u}$.

This leads to the following Navier–Stokes problem.

**Problem 7.2.1: Surface Navier–Stokes equations**

For given initial values $\mathbf{u}^0$, $\phi^0$, viscosity coefficient $\mu_0$, time interval $I = [0, t_{\text{end}}]$, and force term $\mathbf{f}$ find $\mathbf{u}$, $p$, and $\phi$ satisfying

$$\begin{cases} \dot{\mathbf{u}} - 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) + \operatorname{div}_\Gamma(p\mathbf{P}) = \mathbf{f} & \text{on } \Gamma(t), \\ \operatorname{div}_\Gamma \mathbf{u} = 0 & \text{on } \Gamma(t), \\ \nabla \mathbf{u} \mathbf{n} = 0 & \text{in } \mathcal{O}(\Gamma), \\ \partial_t \phi + \nabla \phi \cdot \mathbf{u} = 0 & \text{in } \mathcal{O}(\Gamma). \end{cases} \tag{7.2.2}$$

for $t \in I$, with $\Gamma(t) = \{\mathbf{x} \mid \phi(\mathbf{x}, t) = 0\}$ and $\mathcal{O}(\Gamma)$ denotes a small tubular neighborhood around $\Gamma(t)$.

We emphasize that there is no well-posedness result for this problem. We even do not have conditions that guarantee the existence of a solution to this problem. The nonlinearity between the level set function defining the surface and the velocity field defined on the surface makes this problem difficult to analyse. Since the system is derived by suitable physical arguments, we expect the system to be uniquely solvable when the initial values are chosen sufficiently smooth and the end time $t_{\text{end}}$ is small enough. In the numerical experiments, we will show in various examples that our method converges to a manufactured exact solution.

We give a short overview of the method to solve the surface Navier–Stokes equations numerically. We use a fixed time step size $\Delta t$ and time nodes $t_n = n\Delta t$ satisfying $0 = t_0 < t_1 < \ldots < t_N = t_{\text{end}}$. Again, we use the notations $f^n(\cdot) = f(\cdot, t_n)$ and $\Gamma_h^n = \Gamma_h(t_n)$ for time-dependent functions and surfaces. We assume a given polygonal background domain $\Omega$ that includes the surface $\Gamma(t)$ for each time point $t \in [0, t_{\text{end}}]$. In this background domain $\Omega$ we use a family of shape regular quasi-uniform simplicial triangulations $\{\mathcal{T}_h\}_{h>0}$ with maximal mesh size $h$. The subset of simplices that are "cut" by $\Gamma(t_n)$ is again denoted by $\mathcal{T}_\Gamma^n = \{ T \in \mathcal{T}_h \mid \text{meas}_2(T \cap \Gamma^n) > 0 \}$. In the development of the discretization method for the level set equation in Chapter 6, we introduced the projection domain $\Omega_{\text{Pr}} = \mathcal{N}^{L_P}(\mathcal{T}_\Gamma)$ and the transport domain $\Omega_{\text{Tr}} = \mathcal{N}^{L_T}(\mathcal{T}_\Gamma)$ consisting of $\mathcal{T}_\Gamma$ augmented with layers of neighboring elements. See equation (6.2.1) for a definition of the narrow band layers and Figure 6.5.1 for a visualization.

The level set equation is solved on $\Omega_{\text{Tr}}$ using the DG scheme from Chapter 6. As explained in Section 6.2, we need to extend the level set function from $\Omega_{\text{Pr}}^n$ to $\Omega_{\text{Tr}}^n$ to construct initial and boundary values for the level set equation in the following time step. The transport domain $\Omega_{\text{Tr}}$ has to be chosen large enough such that

$$\Omega_\Gamma^{n+1} \subset \Omega_{\text{Tr}}^n \quad \text{for all } 0 < n < N \tag{7.2.3}$$

holds. If this relation is violated, the surface leaves the transport domain, and the algorithm breaks down. In our implementation, we check (7.2.3) in each time step.

On the surface, we use a TraceFEM as introduced in Chapter 4 to construct a velocity field $\mathbf{u}_h$. The used volume normal derivative stabilization from Section 4.2, leads to an approximation $\mathbf{u}_h^n$ defined on the cut elements $\mathcal{T}_\Gamma^n$. In Section 7.3, we introduce a method to extend the velocity off the surface. Using this extended velocity, we solve the level set equation. Moreover, relation (7.2.3), states that the extended velocity from the previous time step is well defined on $\Gamma_h$ and thus, we can use a BDF scheme for the time discretization in the TraceFEM to solve the PDE on the surface.

The structure of the method (for one time step) is as follows:

**Algorithm 7.2.2: One time step structure**

**Input:** $\phi_h^n$ defined on $\Omega_{\text{Tr}}^{n-1}$, $\mathbf{u}_h^n$ defined on $\Omega_\Gamma^{n-1}$

**Procedure**

Step 1: $\phi_h^n, \mathbf{u}_h^n$ defined on $\Omega_{\text{Tr}}^n \leftarrow$ extension of input functions

Step 2: $\phi_h^{n+1} \leftarrow$ solve level set equation on $\Omega_{\text{Tr}}^n \times [t_n, t_{n+1}]$

Step 3: $\mathbf{n}_h^{n+1} \leftarrow$ smooth normal approximation

Step 4: $\mathbf{u}_h^{n+1} \leftarrow$ solve Navier-Stokes system with TraceFEM on $\Omega_\Gamma^n$

**end Procedure**

In step 3 of the algorithm, we construct a smooth approximation of the normal vector $\mathbf{n}$ on the surface. The canonical normal approximation which was used in the previous chapters is discontinuous across facets of the triangulation which leads to problems when discretizing the (full) surface Navier–Stokes system.

In many transport algorithms known from the literature, one has to reinitialize the level set function after some time steps to keep it "close" to a signed distance function. When we start with a level set function that is close to a signed distance function in our method, we do not need to reinitialize. Extending the velocity $\mathbf{u}$ constantly along the normal vector $\mathbf{n}_h$ seems to be a suitable choice of the velocity to keep the level set function close to a signed distance function. Moreover, the choice of the projection domain $\Omega_{\mathrm{Pr}}$ in the level set extension method is crucial to avoid error accumulation at the boundary of the transport domain, see Section 6.2 for more details.

## 7.3 Velocity extension

In this section, we introduce a method to extend the velocity which is used in Step 1 of Algorithm 7.2.2. There are different extension procedures used in the literature. It is possible to use the extension via a ghost penalty method as in Chapter 6. Also, the *Fast Marching Method* can be applied to extend a vector-valued function. As described in Section 7.2, a velocity that is constant along the normal is desirable for the level set transport equation. There are different ways to construct a velocity field that solves the surface Navier-Stokes system (first two equations of (7.2.2)) and is constant in normal direction off the surface (third equation of (7.2.2)). In Chapter 4 we introduced a stabilization term added to the TraceFEM discretization, namely the volume normal derivative stabilization. This term is used to stabilize the method, but it also extends the velocity off the surface. This stabilization is achieved by adding the bilinear form

$$s_h(\mathbf{u},\mathbf{v}) = \rho \int_{\Omega_h} \nabla \mathbf{u}\mathbf{n}_h \cdot \nabla \mathbf{v}\mathbf{n}_h \,\mathrm{d}\mathbf{x}$$

to the TraceFEM formulation. Here, $\mathbf{n}_h$ is a normal vector approximation and $\Omega_h$ is some domain consisting of tetrahedra that contains the surface $\Gamma$. Adding this bilinear form leads to a velocity that is approximately constant along the normal $\mathbf{n}_h$ (in a weak sense), cf. Section 4.2.2. This (implicit) extension method is used in Chapter 5 to extend the velocity off the surface for the tangential surface Navier–Stokes equations.

We will use the idea of the volume normal derivative term but decouple the extension from the surface PDE. Since the volume normal derivative stabilization is used to stabilize the TraceFEM method in (7.6.2), the output of step 4 of Algorithm 7.2.2 is a velocity field $\mathbf{u}_h^n$ that is defined on the domain $\Omega_\Gamma$ formed by the cut elements. Thus, we construct a method to extend this velocity field to a domain $\Omega_{\mathrm{Ex}}$ satisfying $\Omega_\Gamma \subset \Omega_{\mathrm{Ex}} \subset \Omega$. In Algorithm 7.2.2 the extension domain will be chosen as the subsequent transport domain, i.e., $\Omega_{\mathrm{Ex}} = \Omega_{\mathrm{Tr}}^n$.

The extension is done componentwise using the bilinear form

$$s_h^{\mathrm{ext}}(u,v) := \int_{\Omega_{\mathrm{Ex}}\setminus\Omega_\Gamma} (\nabla u \cdot \mathbf{n}_h)(\nabla v \cdot \mathbf{n}_h)\,\mathrm{d}\mathbf{x},$$

where $\mathbf{n}_h = \nabla\phi_h/\|\nabla\phi_h\|$ denotes the canonical normal approximation defined on $\Omega_{\mathrm{Ex}}$.

We formulate vector-valued finite element functions as $\mathbf{u}_h = \sum_i u_{h,i}\mathbf{e}_i \in \mathbf{V}_{h,k_\mathbf{u}}$, where $(\mathbf{e}_i)_{i=1,2,3}$ denotes the standard Cartesian basis and $\mathbf{V}_{h,k_\mathbf{u}}$ the finite element space introduced in Chapter 4.

The extension procedure reads: For a given $\mathbf{u}_h^{\text{in}} \in \mathbf{V}_{h,k_\mathbf{u}}(\Omega_\Gamma)$, find the extended velocity $\mathbf{u}_h^{\text{out}} \in \mathbf{V}_{h,k_\mathbf{u}}(\Omega_{\text{Ex}} \backslash \Omega_\Gamma)$ such that for $i = 1, 2, 3$ it holds

$$\begin{cases} s_h^{\text{ext}}(u_{h,i}^{\text{out}}, v) = 0 & \text{for all } v \in V_{h,k_\mathbf{u}}(\Omega_{\text{Ex}} \backslash \Omega_\Gamma), \\ u_{h,i}^{\text{out}} = u_{h,i}^{\text{in}} & \text{on } \partial\Omega_\Gamma. \end{cases} \tag{7.3.1}$$

**Remark 7.3.1 (Well-posedness of the extension problem)**

In [Lou21, Section 5.1.1], it is shown that the continuous version of the extension problem (7.3.1) can be rewritten as an anisotropic elliptic diffusion problem, that reads

$$\begin{cases} \operatorname{div}\left(\mathbb{T}\nabla \mathbf{u}^{\text{out}}\right) = 0 & \text{in } \Omega_{\text{Ex}} \backslash \Omega_\Gamma, \\ \mathbf{u}^{\text{out}} = \mathbf{u}^{\text{in}} & \text{on } \partial\Omega_\Gamma, \end{cases} \tag{7.3.2}$$

with $\mathbb{T} = \mathbf{n} \otimes \mathbf{n} = \mathbf{n}\mathbf{n}^T$ denoting a tensor valued diffusion coefficient. Adding suitable Neumann boundary conditions on the boundary $\partial\Omega_{\text{Ex}} \backslash \partial\Omega_\Gamma$ makes (7.3.2) a well-posed problem, as long as the normal vector $\mathbf{n}$ is unambiguously defined on $\Omega_{\text{Ex}}$. Since we assume $\Gamma$ to be sufficiently smooth and the width of $\Omega_{\text{Ex}}$ scales with $h$, the normal vector $\mathbf{n}$ is well-defined on $\Omega_{\text{Ex}}$ for sufficiently small values of $h$. ◇

We summarize the method in the following algorithm.

**Procedure** DoVelocityExtension
**Input:**
- cut and extension domain: $\Omega_\Gamma \subset \Omega_{\text{Ex}}$
- velocity on $\Omega_\Gamma$: $\mathbf{u}_h \in \mathbf{V}_{h,k_\mathbf{u}}(\Omega_\Gamma)$
- normal on $\Omega_{\text{Ex}}$: $\mathbf{n}$

**Output:**
- velocity on $\Omega_{\text{Ex}}$: $\mathbf{u}_h \in \mathbf{V}_{h,k_\mathbf{u}}(\Omega_{\text{Ex}})$

**end Procedure**

With some abuse of notation the extended velocity will also be denoted as $\mathbf{u}_h$.

We test the extension method in a numerical experiment, using a surface Stokes system on a stationary sphere as explained in [Bra+22]. The exact velocity $\mathbf{u}$ (taken from [Bra+22]) is extended constantly in normal direction off the surface. We are interested in the quality of our extension method.

Firstly, we extend the exact velocity field $\mathbf{u}$ from $\Omega_\Gamma$ to $\Omega_{\text{Ex}} = \mathcal{N}(\Omega_\Gamma)$ and denote the extended velocity by $\mathbf{u}^{\text{ext}}$. We measure the difference between the exact velocity and the extended velocity on the extension domain $\Omega_{\text{Ex}}$ by $e_{\Omega_{\text{Ex}}} := \|\mathbf{u} - \mathbf{u}^{\text{ext}}\|_{\Omega_{\text{Ex}}} / \|1\|_{\Omega_{\text{Ex}}}$. Moreover, we consider the $L^2$-norm of $\nabla \mathbf{u}^{\text{ext}} \mathbf{n}_h$ which measures the normal derivative of the extended velocity field. This quantity is denoted by $e_{\nabla^\mathbf{n}} := \|\nabla \mathbf{u}^{\text{ext}} \mathbf{n}_h\|_{\Omega_{\text{Ex}}} / \|1\|_{\Omega_{\text{Ex}}}$. The $L^2$-norm on a domain $U$ is denoted by $\|\cdot\|_U$. The scaling of the norms by the factor $\|1\|_{\Omega_{\text{Ex}}}$ is needed, since the extension domain $\Omega_{\text{Ex}}$ depends on the mesh size $h$.

We employ a second, more realistic, test case. In Algorithm 7.2.2, our extension method is applied to a function constructed by a TraceFE method. Thus, we use the TraceFE

method explained and analysed in [Jan+21] with P2P1–Taylor–Hood finite elements and a level set approximation with $k_g = 2$ to solve the surface Stokes system. The output of the TraceFEM is given by $\mathbf{u}_h$. Since a volume normal derivative stabilization term is added to the TraceFEM formulation, the velocity field $\mathbf{u}_h$ is approximately constant along the normal vector $\mathbf{n}_h$ on the cut elements. Now, we use our method to extend $\mathbf{u}_h$ from $\Omega_\Gamma$ to $\Omega_{\mathrm{Ex}}$ and denote the extended velocity by $\mathbf{u}_h^{\mathrm{ext}}$. We measure the errors $e^h_{\Omega_{\mathrm{Ex}}} := \|\mathbf{u} - \mathbf{u}_h^{\mathrm{ext}}\|_{\Omega_{\mathrm{Ex}}}/\|1\|_{\Omega_{\mathrm{Ex}}}$ and $e^h_{\nabla^{\mathbf{n}}} := \|\nabla \mathbf{u}_h^{\mathrm{ext}} \mathbf{n}_h\|_{\Omega_{\mathrm{Ex}}}/\|1\|_{\Omega_{\mathrm{Ex}}}$. The results of both extension procedures are shown in Figure 7.3.1.

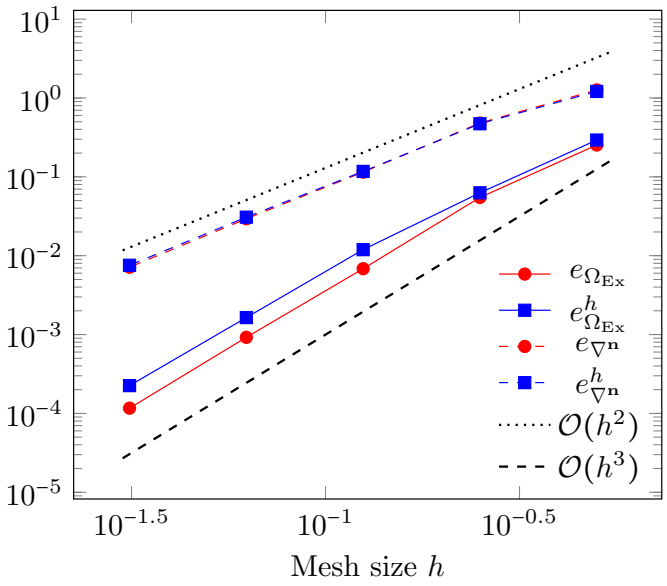



Figure 7.3.1: Errors of steady surface Stokes equations using an explicit volume normal derivative extension.

We observe optimal orders of convergence for all error quantities. The errors $e_{\Omega_{\mathrm{Ex}}}$ and $e^h_{\Omega_{\mathrm{Ex}}}$ converge with convergence rate $\mathcal{O}(h^3)$ and the errors $e_{\nabla^{\mathbf{n}}}$ and $e^h_{\nabla^{\mathbf{n}}}$ converge with convergence rate $\mathcal{O}(h^2)$. Note that the error $e_{\nabla^{\mathbf{n}}}$ is almost identical to the error $e^h_{\nabla^{\mathbf{n}}}$ which indicates that the extension method works similar for the exact velocity field and the finite element approximation constructed by the TraceFEM. We conclude, that our method is able to extend the velocity field off the surface with optimal convergence rates.

In Chapter 5 we used the volume normal derivative stabilization to extend the velocity field off the surface onto an extension domain implicitly in the TraceFEM formulation. This method leads to a similar accuracy of the resulting approximation, but it is computationally expensive. In our experiments, we observed that the decoupling of the extension from the surface PDE leads to up to six times faster computations in the example of the surface Stokes system. This is due to the fact that the explicit extension method can be applied component wise which reduces the size of the linear system to be solved. Moreover, the extension method can be applied to different domains without the need of solving the surface PDE again. When using the extension via the stabilization term added to the TraceFEM formulation, one has to solve the surface PDE again when changing the extension domain. This is not the case when using the explicit extension method we presented in this section. For these reasons, we use the explicit extension method in the following experiments.

## 7.4 Transport and extension of the level set function

In this section, we summarize the methods to solve the level set equation on the transport domain $\Omega_{\mathrm{Tr}}$ and to extend the level set function. Both methods were introduced in Chapter 6.

### Transport of the level set equation

The level set equation is solved on $\Omega_{\mathrm{Tr}}$ using a DG scheme combined with a BDF method. For a BDF(1) method it reads: For given $\phi_h^n \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$, find $\phi_h^{n+1} \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$ such that

$$\sum_{T\in\mathcal{T}_h^{\mathrm{Tr}}}\Bigg(\int_T \frac{\phi_h^{n+1}-\phi_h^n}{\Delta t}\psi_h\,\mathrm{d}\mathbf{x} + \int_T \left(\mathbf{u}^{n+1}\cdot\nabla\phi_h^{n+1}\right)\psi_h\,\mathrm{d}\mathbf{x} - \int_{\partial T^-\setminus\partial\Omega_{\mathrm{Tr}}} [\phi_h^{n+1}]\psi_h\left(\mathbf{u}^{n+1}\cdot\mathbf{n}_T\right)\mathrm{d}s\Bigg)$$
$$- \int_{\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}} \phi_h^{n+1}\psi_h\left(\mathbf{u}^{n+1}\cdot\mathbf{n}_{\Omega_{\mathrm{Tr}}}\right)\mathrm{d}s = -\int_{\partial\Omega_{\mathrm{Tr}}^{n,\mathrm{in}}} \phi_D\,\psi_h\left(\mathbf{u}^{n+1}\cdot\mathbf{n}_{\Omega_{\mathrm{Tr}}}\right)\mathrm{d}s$$

holds for all $\psi_h \in V_{h,k_g}^{\mathrm{DG}}(\Omega_{\mathrm{Tr}})$, cf. (6.4.4). See Section 6.4 for more details. The Dirichlet boundary data $\phi_D$ is constructed from the previous level set functions $\phi_h^n,\dots,\phi_h^{n-R+1}$ using a linear extrapolation as explained in (6.3.2). Here, $R$ denotes the order of the BDF method used in the transport scheme.

The input and output of the transport method are summarized in the following algorithm.

**Procedure** DoTransportStep
**Input:**
- transport domain $\Omega_{\mathrm{Tr}}$
- velocity $\mathbf{u}$ on $\Omega_{\mathrm{Tr}}$
- previous level set functions $\phi_h^n,\dots,\phi_h^{n-R+1} \in V_{h,k_g}(\Omega_{\mathrm{Tr}})$

**Output:**
- level set function $\phi_h^{n+1} \in V_{h,k_g}(\Omega_{\mathrm{Tr}})$

**end Procedure**

### Extension of the level set equation

The extension method is introduced in Chapter 6, where a ghost penalty technique is combined with an $L^2$–projection. It is given in (6.5.2): For given projection and extension domains holding $\Omega_{\mathrm{Pr}} \subset \Omega_{\mathrm{Ex}}$ and an input function $\phi_h^{\mathrm{in}} \in V_{h,k_g}(\Omega_{\mathrm{Pr}})$ defined on $\Omega_{\mathrm{Pr}}$, determine $\phi_h^{\mathrm{out}} \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$ that satisfies

$$(\phi_h^{\mathrm{out}},\psi_h)_{\Omega_{\mathrm{Pr}}} + s_h^{(0)}(\phi_h^{\mathrm{out}},\psi_h) = (\phi_h^{\mathrm{in}},\psi_h)_{\Omega_{\mathrm{Pr}}} \qquad \text{for all} \quad \psi_h \in V_{h,k_g}(\Omega_{\mathrm{Ex}}),$$

with

$$s_h^{(0)}(\phi,\psi) = \gamma^{\mathrm{ext}} \sum_{F\in\mathcal{F}_h^{\mathrm{GP}}} \int_{\omega(F)} (\phi_1-\phi_2)(\psi_1-\psi_2)\,\mathrm{d}\mathbf{x}.$$

Here, $\mathcal{F}_h^{\mathrm{GP}}$ denotes the set of facets in the extension domain $\Omega_{\mathrm{Ex}}$, $\omega(F)$ is a facet patch and $\gamma^{\mathrm{ext}} > 0$ a weight parameter. See Section 6.5 for more details and an error analysis of the method.

The extended level set function does not satisfy $\phi_h^{\mathrm{out}} = \phi_h^{\mathrm{in}}$ on $\Omega_{\mathrm{Pr}}$, but the difference $\|\phi_h^{\mathrm{out}} - \phi_h^{\mathrm{in}}\|_{L^2(\Omega_{\mathrm{Pr}})}$ converges with optimal order. Following Lemma 6.5.6 and the results from the numerical experiment in Figure 6.5.5, the error on the projection domain is slowly increasing when the extension is applied multiple times. Since the extension method will be applied in each time step of the surface Navier–Stokes algorithm 7.2.2 with an only slightly different projection domain, we expect that the use of the $L^2$-projection adds a small error to the level set function on the projection domain and thus on the discrete representation of the surface. To avoid this error, we modify the extension method. We improve the method such that $\phi_h^{\mathrm{out}} = \phi_h^{\mathrm{in}}$ on $\Omega_{\mathrm{Pr}}$ holds. We replace the $L^2$–projection by a Dirichlet boundary condition: the extended function $\phi_h^{\mathrm{out}}$ should satisfy $\phi_h^{\mathrm{out}} = \phi_h^{\mathrm{in}}$ on the boundary $\partial\Omega_{\mathrm{Pr}}$ (similar to the velocity extension) and the system is solved on the degrees of freedom on $\Omega_{\mathrm{Ex}} \cap \Omega_{\mathrm{Pr}}$ only. On the degrees of freedom of the projection domain, the values of the input function $\phi^{\mathrm{in}}$ are copied to the extended function $\phi_h^{\mathrm{out}}$. Thus, the extended function equals the input function on $\Omega_{\mathrm{Pr}}$ and the size of the linear system is reduced since only the DOFs on $\Omega_{\mathrm{Ex}} \cap \Omega_{\mathrm{Pr}}$ are involved. This extension does not need any parameter $\gamma^{\mathrm{ext}}$. We expect that the error analysis from Section 6.5 can be adapted to this modified extension method, leading to optimal convergence rates.

We summarize the extension in the following algorithm.

**Procedure** DoLevelSetExtension

**Input:**

- extension domain $\Omega_{\mathrm{Ex}}$
- projection domain $\Omega_{\mathrm{Pr}} \subset \Omega_{\mathrm{Ex}}$
- level set function $\phi_h^{\mathrm{in}} \in V_{h,k_g}(\Omega_{\mathrm{Pr}})$

**Output:**

- level set function $\phi_h^{\mathrm{out}} \in V_{h,k_g}(\Omega_{\mathrm{Ex}})$

**end Procedure**

Again, with some abuse of notation, we will denote the input and the output of the extension method as $\phi_h$ in the remainder.

We test the modified extension method for $k_g = 1$, using the kite geometry described by the level set function $\phi(\mathbf{x}) = (x_1 - x_3^2)^2 + x_2^2 + x_3^2 - 1$. See Figure 6.7.1 for a visualization of the geometry. As projection and extension domains, we use one layer of neighboring elements to the cut elements ($\Omega_{\mathrm{Pr}} = \mathcal{N}(\Omega_\Gamma)$) and three layers ($\Omega_{\mathrm{Ex}} = \mathcal{N}^3(\Omega_\Gamma)$). The input function $\phi_h$ is given by an Oswald-type quasi interpolation of the exact level set function $\phi$ on the projection domain. We perform the extension using the new method, utilizing the Dirichlet boundary values (with the solution denoted by $\phi_{\mathrm{Bnd}}$) and the old method, utilizing the $L^2$-projection on the projection domain (with the solution denoted by $\phi_{L^2}$).

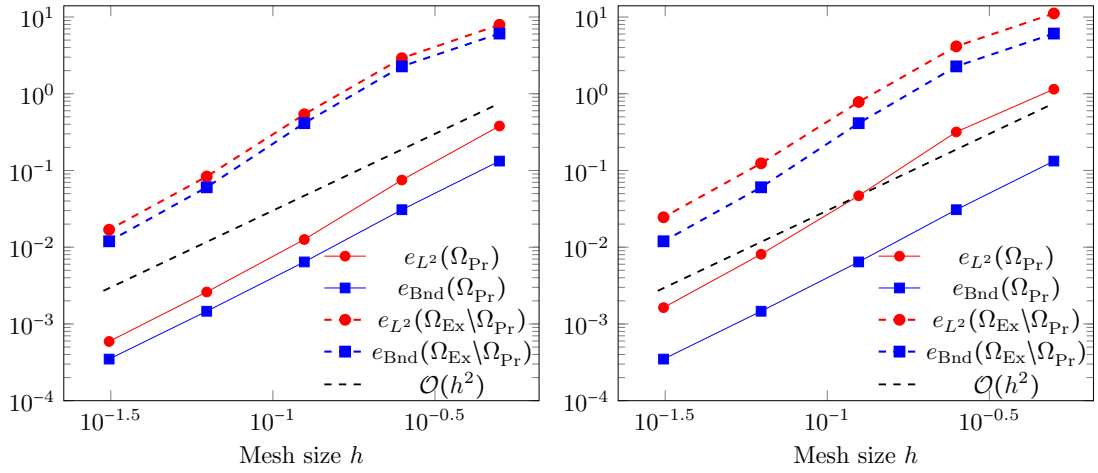
The errors of interest are given by

$$e_{\mathrm{Bnd}}(\omega) := \frac{\|\phi_{\mathrm{Bnd}} - \phi\|_\omega}{\|1\|_\omega}, \qquad e_{L^2}(\omega) := \frac{\|\phi_{L^2} - \phi\|_\omega}{\|1\|_\omega},$$

with $\omega = \Omega_{\mathrm{Pr}}$ to measure the errors on the projection domain and $\omega = \Omega_{\mathrm{Ex}} \backslash \Omega_{\mathrm{Pr}}$ to measure the errors on the extension domain.

We perform two tests: In the first test, we apply the extension methods once, to see if the modified method obtains optimal order of convergence. In the second test, we apply the extension methods 25 times, to simulate the situation of repeatedly applying the extension method in the Navier–Stokes algorithm. A similar experiment was performed in Section 6.5.3.

The errors are shown in Figure 7.4.1 with the left-hand plot showing the errors after a single extension step and the right-hand plot showing the errors after 25 applications of the extension methods.




(a) Applied the extensions once.

(b) Applied the extensions 25 times.

Figure 7.4.1: Errors of extension of the kite level set function using the $L^2$-projection or the boundary values with $k_g = 1$.

We observe optimal error convergence rates of at least $\mathcal{O}(h^2)$ for each error quantity. The errors obtained by the modified method ($e_{\mathrm{Bnd}}$) are slightly smaller that the errors obtained by the $L^2$-projection method ($e_{L^2}$) both on the projection domain and on the extension domain. Moreover, we observe a slight increase of the error on the extension domain when applying the method based on the $L^2$-projection multiple times and a large increase of this error on the projection domain. Note that the $\mathcal{O}(h^2)$ reference line has the same values in both figures. Meanwhile, the errors (both on the extension and on the projection domain) of the modified method do not increase when applying the method repeatedly. This indicates that no error accumulates when applying the modified extension method multiple times. Since the extension method is applied in each time step of the Navier–Stokes algorithm and an accumulation of the error in $\phi_h$ leads to an accumulation of an error in the position of $\Gamma_h$, we conclude that the modified extension method is more suitable for our purpose. The modified extension method is also slightly computationally cheaper since the size of the linear system is reduced.

## 7.5 Improved normal approximation

In this section, we introduce a method to construct a smooth normal approximation. The normal vector $\mathbf{n}$ is a crucial quantity in the Navier–Stokes system. It is used to construct the tangential projection $\mathbf{P} = \mathbf{I} - \mathbf{n}\mathbf{n}^T$ that is used to calculate the surface differential operators like the surface gradient $\nabla_\Gamma \mathbf{u} = \mathbf{P}\nabla\mathbf{u}\mathbf{P}$, the surface divergence $\operatorname{div}_\Gamma \mathbf{u} = \operatorname{tr}(\mathbf{P}\nabla\mathbf{u}\mathbf{P})$, and the surface rate of strain tensor $\mathbf{E}_\Gamma(\mathbf{u}) = \frac{1}{2}(\nabla_\Gamma \mathbf{u} + \nabla_\Gamma^T \mathbf{u})$.

The canonical way to approximate the normal vector is given by the normalized gradient of the level set function, i.e.,

$$\mathbf{n}_h = \frac{\nabla \phi_h}{\|\nabla \phi_h\|}.$$

This approximation is the standard normal approximation used in the literature on TraceFEM, e.g., in [BR20]. We use it to define $\mathbf{P}_h$ and the surface differential operators when solving the tangential surface Navier–Stokes system (TSNS) in Chapter 5. This normal approximation converges with order $\mathcal{O}(h^{k_g})$ in the $L^2$–norm if the approximation of the level set function converges with optimal rate of $\mathcal{O}(h^{k_g+1})$ in the $L^2$–norm.

In contrast to the TSNS system, the (full) surface Navier–Stokes equations (SNS) contain a derivative of the tangential projector $\mathbf{P}$ in the term $\operatorname{div}_\Gamma(p\mathbf{P})$. Thus, in the SNS system, the discretization of $\mathbf{P}$ (and the normal vector $\mathbf{n}$) plays a more important role. Also, if we use a different formulation of the SNS system, e.g., the one from (7.2.1), where no derivative of the tangential projector appears, we still need a good approximation of the normal vector to define the mean curvature $\kappa = \operatorname{div}_\Gamma \mathbf{n}$.

In the numerical experiments, we observe that the canonical normal approximation is not sufficiently accurate to approximate the SNS system. One reason might be that the canonical normal approximation is discontinuous across elements of the triangulation. We introduce a method to construct a smooth finite element approximation of the normal vector on the surface which is more accurate than the canonical approximation.

We use an $L^2$–projection on the surface of the canonical normal approximation and stabilize it with a ghost penalty term. In Section 4.2, we introduced the following notations: the canonical extension of a polynomial $\mathcal{E}_T : P_k(T) \to P_k(\mathbb{R}^3)$, the extensions $\mathbf{v}_i = \mathcal{E}_{T_i}\mathbf{v}|_{T_i}$ for $\mathbf{v} \in \mathbf{V}_{h,k_g}$, and the facet patch $\omega_F := T_1 \cup T_2$. We introduce the bilinear form

$$s_{h,\mathbf{n}}(\mathbf{n}, \mathbf{v}) = \int_{\Gamma_h} \mathbf{n} \cdot \mathbf{v} \, \mathrm{d}s + \rho_{\mathbf{n}} \sum_{F \in \mathcal{F}_h^\Gamma} \int_{\omega_F} (\mathbf{n}_1 - \mathbf{n}_2) \cdot (\mathbf{v}_1 - \mathbf{v}_2) \, \mathrm{d}\mathbf{x},$$

with $\mathcal{F}_h^\Gamma := \{F = T_1 \cap T_2 \mid T_1, T_2 \in \mathcal{T}_\Gamma, T_1 \neq T_2, \operatorname{meas}_2(F) > 0\}$. The improved normal approximation is given by: Find $\widetilde{\mathbf{n}}_h \in \mathbf{V}_{h,k_g}(\Omega_\Gamma)$ such that

$$s_{h,\mathbf{n}}(\widetilde{\mathbf{n}}_h, \mathbf{v}) = \int_{\Gamma_h} \mathbf{n}_h \cdot \mathbf{v} \, \mathrm{d}s \quad \text{for all } \mathbf{v} \in \mathbf{V}_{h,k_g}(\Omega_\Gamma),$$

where $\mathbf{n}_h = \nabla\phi_h / \|\nabla\phi_h\|$ denotes the canonical normal vector approximation defined on the surface $\Gamma_h$.

Note, that this normal approximation is defined on the cut domain $\Omega_\Gamma$. We will use it to define the surface differential operators and the tangential projection $\mathbf{P}$. Since they are only evaluated on the surface $\Gamma_h$, it is sufficient to define the improved normal on $\Omega_\Gamma$.

The method for normal construction is summarized in the following algorithm.

**Procedure** CalculateSurfaceNormal
**Input:**
- surface $\Gamma_h$
- discontinuous normal $\mathbf{n}_h = \frac{\nabla \phi_h}{\|\nabla \phi_h\|}$ on $\Gamma_h$

**Output:**
- $\widetilde{\mathbf{n}}_h \in \mathbf{V}_{h,k_g}(\Omega_\Gamma)$

**end Procedure**

We investigate the quality of our method in a numerical experiment. We employ the introduced method and compare the quality of the "improved" normal $\tilde{e}_h := \|\widetilde{\mathbf{n}}_h - \mathbf{n}\|_{\Gamma_h}$ with the quality of the canonical normal $e_h := \|\mathbf{n}_h - \mathbf{n}\|_{\Gamma_h}$, where $\mathbf{n}$ denotes the exact normal. The initial mesh size is chosen as $h_0 = 0.5$ (halved in each refinement), the polynomial degree as $k_g = 1$, and the stabilization parameter as $\rho_{\mathbf{n}} = \frac{1}{h^3}$. We use the kite and torus geometries as introduced in Sections 5.5 and 7.8. The level set approximation used to construct the canonical normal is given by an Oswald type interpolation of the exact level set function. The results are shown in Figure 7.5.1.

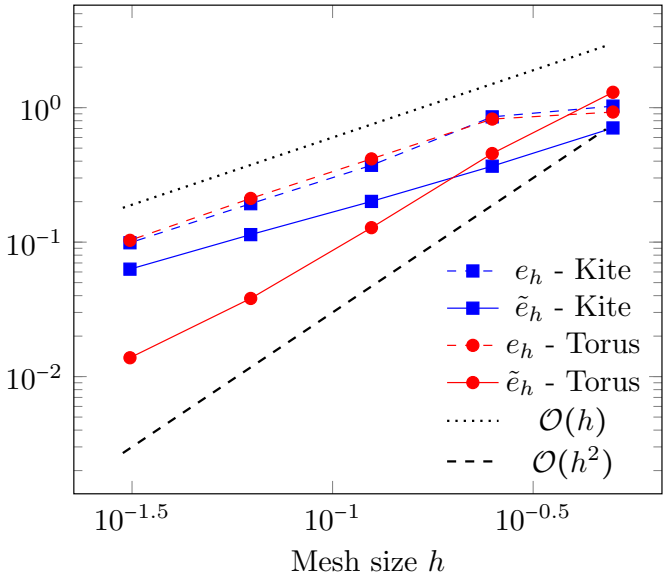



Figure 7.5.1: Errors of improved and canonical normal computation on a kite and a torus geometry.

We observe optimal rates of at least $\mathcal{O}(h)$ for the improved and the canonical normal approximation, which is expected since the level set approximation is of order $\mathcal{O}(h^2)$ and the normal approximations consist of first derivatives of the level set approximation. More interesting, we notice that the absolute errors in the improved normal approximations are indeed smaller than the absolute errors in the canonical normal approximation. For the torus example, the error is reduced by a factor of 10 on the finest mesh level, and we observe convergence rates of $\mathcal{O}(h^2)$. This improvement of accuracy may be explained by the fact, that the improved normal approximation is a finite element function

of degree $k_g$, while the canonical normal approximation is a piecewise polynomial of degree $k_g - 1$. Thus, the improved normal approximation is continuous across facets of the triangulation and has a higher degree. The result indicates, that our new method is an improvement over the canonical normal approximation concerning the accuracy of the normal vector.

In the numerical experiments in Section 7.8, we use the improved normal approximation to construct a converging scheme for the surface Navier–Stokes equations. We observed, that the canonical normal approximation leads to a non-converging scheme in our examples and thus, the improved normal is a crucial ingredient for the method.

## 7.6 TraceFEM for the surface Navier–Stokes equations

In this section, we derive a TraceFEM formulation for the surface Navier–Stokes equation in one time step, which will be used in Step 4 of Algorithm 7.2.2. We start with the Navier–Stokes system (7.2.2) and derive a weak formulation. By multiplying the left-hand side of the first equation of (7.2.2) with a sufficiently smooth test function $\mathbf{v}$, and applying integration by parts, we get

$$\begin{aligned}
&\int_{\Gamma(t)} \overset{\bullet}{\mathbf{u}} \cdot \mathbf{v} \,\mathrm{d}s - 2\mu_0 \int_{\Gamma(t)} \operatorname{div}_\Gamma \mathbf{E}_\Gamma(\mathbf{u}) \cdot \mathbf{v} \,\mathrm{d}s + \int_{\Gamma(t)} \operatorname{div}_\Gamma(p\mathbf{P}) \cdot \mathbf{v} \,\mathrm{d}s \\
&= \int_{\Gamma(t)} (\partial_t \mathbf{u} + \nabla \mathbf{u}\mathbf{u}) \cdot \mathbf{v} \,\mathrm{d}s + 2\mu_0 \int_{\Gamma(t)} \mathbf{E}_\Gamma(\mathbf{u}) : \mathbf{E}_\Gamma(\mathbf{v}) \,\mathrm{d}s - \int_{\Gamma(t)} p \underbrace{\mathbf{P} : \nabla_\Gamma \mathbf{v}}_{=\operatorname{tr}(\mathbf{P}^T \nabla_\Gamma \mathbf{v})} \,\mathrm{d}s \qquad (7.6.1) \\
&= \int_{\Gamma(t)} (\partial_t \mathbf{u} + \nabla \mathbf{u}\mathbf{u}) \cdot \mathbf{v} + 2\mu_0 \mathbf{E}_\Gamma(\mathbf{u}) : \mathbf{E}_\Gamma(\mathbf{v}) - p \operatorname{div}_{\Gamma_h} \mathbf{v} \,\mathrm{d}s.
\end{aligned}$$

To obtain a TraceFEM formulation, we use the Taylor-Hood finite element spaces $\mathbf{V}_{h,k_\mathbf{u}}$ for the velocity and $V_{h,k_\mathbf{u}-1}$ for the pressure with some $k_\mathbf{u} \geq 2$. The finite element spaces are defined as in Chapter 4. Motivated by (7.6.1), we introduce the bilinear forms

$$\begin{aligned}
a_h(\widetilde{\mathbf{u}}_h; \mathbf{u}_h, \mathbf{v}_h) &:= \int_{\Gamma_h} \nabla \mathbf{u}_h \widetilde{\mathbf{u}}_h \cdot \mathbf{v}_h + 2\mu_0 \mathbf{E}_{\Gamma_h}(\mathbf{u}_h) : \mathbf{E}_{\Gamma_h}(\mathbf{v}_h) \,\mathrm{d}s \\
\text{and} \quad b_h(\mathbf{u}_h, q_h) &:= \int_{\Gamma_h} \operatorname{div}_{\Gamma_h} \mathbf{u}_h \cdot q_h \,\mathrm{d}s,
\end{aligned}$$

where $\widetilde{\mathbf{u}}_h$ is some approximation of $\mathbf{u}$ used to linearize the system.

As explained in Section 4.2, we need to stabilize the method to fulfill the *Ladyzhenskaya–Babuška–Brezzi* (LBB) condition and to obtain a well conditioned linear system. We use the volume normal derivative stabilizations from [Bur+18] for the velocity and the pressure which are given by

$$s_h^\mathbf{u}(\mathbf{u}, \mathbf{v}) := \rho_\mathbf{u} \int_{\Omega_\Gamma} \nabla \mathbf{u} \widetilde{\mathbf{n}}_h \cdot \nabla \mathbf{v} \widetilde{\mathbf{n}}_h \,\mathrm{d}\mathbf{x} \quad \text{and} \quad s_h^p(p, q) := \rho_p \int_{\Omega_\Gamma} (\nabla p \cdot \widetilde{\mathbf{n}}_h)(\nabla q \cdot \widetilde{\mathbf{n}}_h) \,\mathrm{d}\mathbf{x},$$

with some stabilization parameters $\rho_\mathbf{u} \sim h^{-1}$ and $\rho_p \sim h$. Note, that in contrast to Chapter 5, we apply the stabilization on the cut domain $\Omega_\Gamma$ only and not on an extended domain. We perform the extension of the velocity explicitly using the method from Section 7.3 in Step 1 of Algorithm 7.2.2.

The partial time derivative is discretized using the backward Euler method. The Trace-

FEM formulation for BDF(1) is given by: Find $(\mathbf{u}_h^n, p_h^n) \in \mathbf{V}_{h,k_\mathbf{u}} \times V_{h,k_\mathbf{u}-1}$ such that

$$\begin{cases} \displaystyle\int_{\Gamma_h^n} \frac{\mathbf{u}_h^n - \mathbf{u}_h^{n-1}}{\Delta t} \cdot \mathbf{v} \,\mathrm{d}s + a_h(\mathbf{u}_h^{n-1}; \mathbf{u}_h^n, \mathbf{v}_h) - b_h(\mathbf{v}_h, p_h^n) + s_h^\mathbf{u}(\mathbf{u}_h^n, \mathbf{v}_h) = \int_{\Gamma_h^n} \mathbf{f}^n \cdot \mathbf{v}_h \,\mathrm{d}s, \\ -b_h(\mathbf{u}_h^n, q_h) + s_h^p(p_h^n, q_h) = 0, \end{cases} \tag{7.6.2}$$

holds for all $(\mathbf{v}_h, q_h) \in \mathbf{V}_{h,k_\mathbf{u}} \times V_{h,k_\mathbf{u}-1}$. Note that in this equation, the surface differential operators $\nabla_{\Gamma_h}$ and $\mathrm{div}_{\Gamma_h}$ are defined using the normal vector $\widetilde{\mathbf{n}}_h$ as constructed in the previous section and the previous velocity $\mathbf{u}_h^{n-1}$ has to be well defined on the surface $\Gamma_h^n$. The extension to higher–order BDF($R$) methods is possible as explained in Remark 7.6.1.

A short overview of the input and output of the TraceFEM for the surface Navier–Stokes equations is given in the following algorithm.

**Procedure** DoNavierStokesStep

**Input:**

- velocities from previous time steps $\mathbf{u}_h^{n-1}, \dots, \mathbf{u}_h^{n-R} \in \mathbf{V}_{h,k_\mathbf{u}}(\Omega_{\mathrm{Tr}}^{n-1})$
- outer force $\mathbf{f}^n$
- surface $\Gamma_h^n$ with its cut elements $\mathcal{T}_\Gamma^n$
- normal approximations $\mathbf{n}_h^n$ and $\widetilde{\mathbf{n}}_h^n$

**Output:**

- $\mathbf{u}_h^n \in \mathbf{V}_{h,k_\mathbf{u}}(\Omega_\Gamma^n)$

**end Procedure**

**Remark 7.6.1 (Higher order BDF version)**

As in the TraceFEM for the tangential surface Navier–Stokes equations, cf. Remark 5.3.3, the method can be extended to higher order BDF schemes to discretize the time derivative. The time difference in (7.6.2) is then replaced by

$$\partial_t \mathbf{u} \approx \frac{3\mathbf{u}^n - 4\mathbf{u}^{n-1} + \mathbf{u}^{n-2}}{2\Delta t} \qquad \text{for BDF(2)}$$

$$\text{and} \quad \partial_t \mathbf{u} \approx \frac{11\mathbf{u}^n - 18\mathbf{u}^{n-1} + 9\mathbf{u}^{n-2} - 2\mathbf{u}^{n-3}}{6\Delta t} \qquad \text{for BDF(3).}$$

For the linearization of the inertia term we replace $\widetilde{\mathbf{u}}_h^n = \mathbf{u}_h^{n-1}$ by

$$\widetilde{\mathbf{u}}_h^n = 2\mathbf{u}_h^{n-1} - \mathbf{u}_h^{n-2} \qquad \text{for BDF(2)}$$

$$\text{and} \quad \widetilde{\mathbf{u}}_h^n = 3\mathbf{u}_h^{n-1} - 3\mathbf{u}_h^{n-2} + \mathbf{u}_h^{n-3} \qquad \text{for BDF(3).}$$

Note that when a higher order BDF scheme is used, not only the velocity $\mathbf{u}^{n-1}$ has to be defined on $\Gamma^n$, but also the velocity $\mathbf{u}^{n-2}$ for BDF(2) and $\mathbf{u}^{n-2}$ and $\mathbf{u}^{n-3}$ for BDF(3) have to be defined on $\Gamma^n$. We will use the extension procedure from Section 7.3 to extend the used velocities. ◇

## 7.7 The full surface Navier–Stokes discretization method

In this section, we summarize the whole algorithm for the surface Navier–Stokes system. We use the methods introduced in the previous sections to perform the extension, transport the level set, construct an improved normal approximation, and solve the Navier–Stokes system on the surface. The needed input of the method is given by

- $\phi^0$, $\mathbf{u}^0$: initial level set function and velocity (defined in a neighborhood of the initial surface),
- $\Omega \subset \mathbb{R}^3$: polygonal domain such that $\Gamma(t) \subset \Omega$ holds,
- $t_{\text{end}}$: end time,
- $\mathbf{f}$: outer force term,
- $\mu_0$: viscosity coefficient.

The parameters of the algorithm are given by

- $k_{\mathbf{u}}$, $k_g$: polynomial degrees for velocity and level set function,
- $R$: order of BDF scheme for level set equation and Navier–Stokes,
- $\Delta t$, $h$: time step and mesh size,
  - $N = \left\lceil \frac{t_{\text{end}}}{\Delta t} \right\rceil$: number of time steps,
- $L_P$: layers used in projection in level set extension,
- $\rho_{\mathbf{u}}$ and $\rho_p$: stabilization parameters for velocity and pressure in TraceFEM
- $\rho_{\mathbf{n}}$: stabilization parameter for the normal approximation

The number of layers used to define the transport domain depends on the maximal displacement of $\Gamma$ in one time step, given by $\Delta t \max_{t\in[t_n,t_{n+1}]} \|V_\Gamma(t)\|_{L^\infty(\Gamma)}$. We choose it as

$$L_T = \max\{1 + 4\frac{\Delta t}{h}\|\mathbf{u}_h^n \cdot \mathbf{n}_h^n\|_{L^\infty(\Omega_\Gamma)}, 3\},$$

to ensure that $\Omega_{\text{Tr}}^n$ is large enough to contain the surface $\Gamma_h^{n+1}$ and the projection domain $\Omega_{\text{Pr}}^n$. This choice is further motivated and explained in Section 6.6.

**Remark 7.7.1**

It is known from the literature (cf. e.g. [BR20; ORS24]), that the stabilization parameters $\rho_{\mathbf{u}}$ and $\rho_p$ should be chosen as $\rho_{\mathbf{u}} = c_{\rho_{\mathbf{u}}} h^{-1}$ and $\rho_p = c_{\rho_p} h$ and that varying the constants $c_{\rho_{\mathbf{u}}}$, $c_{\rho_p}$ does not have a huge impact on the quality of the approximation. From numerical experiments and [Lou21], we follow that $\rho_{\mathbf{n}}$ should be chosen as $\rho_{\mathbf{n}} = c_{\rho_{\mathbf{n}}} h^{-3}$. In the numerical experiments we will stick to the values

$$\rho_{\mathbf{u}} = \frac{1}{h}, \quad \rho_p = h, \quad \text{and} \quad \rho_{\mathbf{n}} = \frac{1}{h^3},$$

for the stabilization parameters. ◇

We summarize the whole method to approximate the surface Navier–Stokes system in the following algorithm.

**Algorithm 7.7.2: Surface Navier–Stokes algorithm**

1: **Procedure**
2: **for** $n$ from 0 to $N-1$ **do**:
3: $L_T \leftarrow \max\left\{3, [1 + 4\frac{\Delta t}{h}\|\mathbf{u}_h^n \cdot \mathbf{n}_h^n\|_{L^\infty(\Omega_\Gamma)}]\right\}$
4: $\Omega_{\mathrm{Tr}}^n \leftarrow \mathcal{N}^{L_T}(\Omega_\Gamma^n)$
5: **if** $n > 0$ **then**
6: $\Omega_{\mathrm{Pr}}^n \leftarrow \mathcal{N}^{L_P}(\Omega_\Gamma^n)$
7: $\phi_h^n \leftarrow \mathrm{DoLevelSetExtension}(\Omega_{\mathrm{Tr}}^n, \Omega_{\mathrm{Pr}}^n, \phi_h^n)$
8: $\mathbf{u}_h^n \leftarrow \mathrm{DoVelocityExtension}(\Omega_\Gamma^n, \Omega_{\mathrm{Tr}}^n, \mathbf{u}_h^n, \mathbf{n}_h^n)$
9: **for** $i$ from 1 to $R-1$ **do**
10: $\phi_h^{n-i} \leftarrow \mathrm{DoLevelSetExtension}(\Omega_{\mathrm{Tr}}^n, \Omega_{\mathrm{Pr}}^n, \phi_h^{n-i})$
11: $\mathbf{u}_h^{n-i} \leftarrow \mathrm{DoVelocityExtension}(\Omega_{\mathrm{Tr}}^{n-1} \cap \Omega_{\mathrm{Tr}}^n, \Omega_{\mathrm{Tr}}^n, \mathbf{u}_h^{n-i}, \mathbf{n}_h^n)$
12: **end for**
13: **end if**
14: $\widetilde{\mathbf{u}}_h^n \leftarrow$ extrapolation in time of $\mathbf{u}_h^i$ for $i = n, \ldots, n-R+1$
15: $\phi_h^{n+1} \leftarrow \mathrm{DoTransportStep}(\Omega_{\mathrm{Tr}}^n, \widetilde{\mathbf{u}}_h^n, \phi_h^n, \ldots, \phi_h^{n-R+1})$
16: $\Gamma_h^{n+1} \leftarrow \{\mathbf{x} \mid \phi_h^{n+1}(\mathbf{x}) = 0\}$
17: $\mathbf{n}_h^{n+1} \leftarrow \nabla\phi_h^{n+1} / \|\nabla\phi_h^{n+1}\|$
18: $\widetilde{\mathbf{n}}_h \leftarrow \mathrm{CalculateSurfaceNormal}(\Gamma_h^{n+1}, \mathbf{n}_h^{n+1})$
19: $\mathbf{u}_h^{n+1} \leftarrow \mathrm{DoNavierStokesStep}(\mathbf{u}_h^n, \ldots, \mathbf{u}_h^{n-R+1}, \mathbf{f}^{n+1}, \Gamma_h^{n+1}, \mathbf{n}_h^{n+1}, \widetilde{\mathbf{n}}_h)$
20: **end for**
21: **end Procedure**

With some abuse of notation, we denote the extended functions by the same symbol as the original function, e.g., $\phi_h^n$ denotes the level set approximation on the projection domain $\Omega_{\mathrm{Pr}}^n$ and the transport domain $\Omega_{\mathrm{Tr}}^n$ in line 7 of the algorithm.

### 7.7.1 Illustration of the algorithm

We illustrate one time step of Algorithm 7.7.2 in the following figures using a one dimensional example, to provide an overview of the method and the domains used ($\Omega_{\mathrm{Tr}}$, $\Omega_{\mathrm{Pr}}$, and $\Omega_\Gamma$). In all figures, the $x$-axis denotes the space variable and the $y$-axis the time variable. For each time point, the surface consists of a single point, i.e. $\Gamma(t) \in \mathbb{R}$ for all $t \in [0, t_{\mathrm{end}}]$. We consider the case of a BDF(1) scheme and time step $n > 0$. The input data of one time step is given by the level set approximation $\phi_h^n$ defined on the previous transport domain $\Omega_{\mathrm{Tr}}^{n-1}$ and the velocity approximation $\mathbf{u}_h^n$ defined on the cut domain $\Omega_\Gamma^n$.

We start by illustrating line 7 of the algorithm, where the level set function $\phi_h^n$ is extended to the transport domain $\Omega_{\mathrm{Tr}}^n$. For this extension, we use the given approximation $\phi_h^n$ on the projection domain $\Omega_{\mathrm{Pr}}^n$ which is smaller than the transport domain $\Omega_{\mathrm{Tr}}^{n-1}$, where $\phi_h^n$ is defined. As explained in Chapter 6, the choice of the smaller projection domain is important to prevent error accumulation induced by low order boundary data.

We illustrate the extension of $\phi_h^n$ from the projection domain (blue) to the transport domain (red) in Figure 7.7.1.

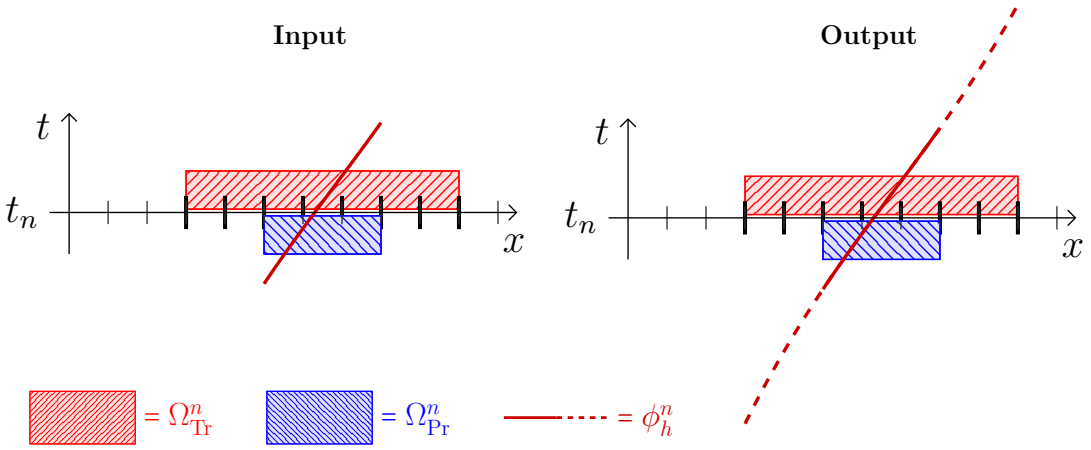



Figure 7.7.1: Line 7 of Algorithm 7.7.2: Level set extension. The level set function $\phi_h^n$ is extended from $\Omega_{\mathrm{Pr}}^n$ to the transport domain $\Omega_{\mathrm{Tr}}^n$.

For the velocity extension, we use the maximal possible projection domain, namely the cutted elements $\Omega_\Gamma^n$. Since no boundary values are needed in the surface Navier–Stokes solver, we do not have to choose a smaller projection domain. The velocity extension is shown in Figure 7.7.2.

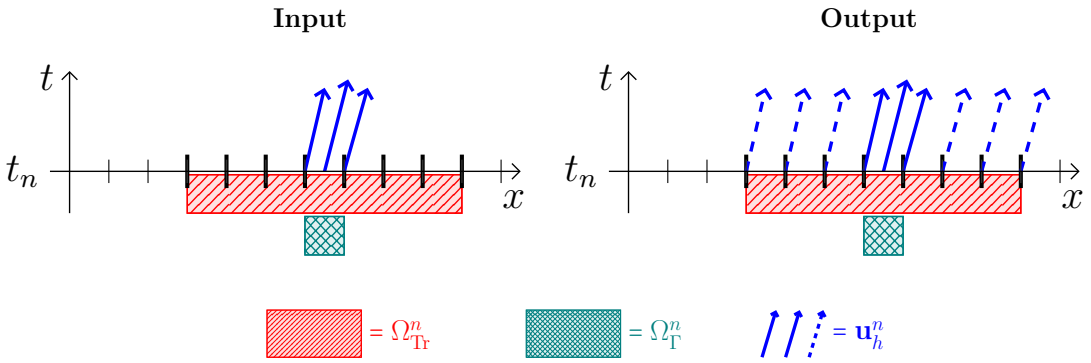



Figure 7.7.2: Line 8 of Algorithm 7.7.2: Velocity extension. The velocity function $\mathbf{u}_h^n$ is extended from $\Omega_\Gamma^n$ to the transport domain $\Omega_{\mathrm{Tr}}^n$.

Now, we have the level set approximation $\phi_h^n$ and the velocity approximation $\mathbf{u}_h^n$ defined on the transport domain $\Omega_{\mathrm{Tr}}^n$. In line 15 of Algorithm 7.7.2, we solve the transport equation to obtain the new level set approximation $\phi_h^{n+1}$ on the transport domain $\Omega_{\mathrm{Tr}}^n$. We use an extrapolation in time of the previous level set approximations $\phi_h^n, \dots, \phi_h^{n-R+1}$ to obtain Dirichlet boundary values. In line 14 of Algorithm 7.7.2, we extrapolate the velocity approximations $\mathbf{u}_h^n, \dots, \mathbf{u}_h^{n-R+1}$ to obtain a velocity function $\widetilde{\mathbf{u}}_h^n$ that is used in

the transport step. The extrapolation is done by taking a linear combination of the previous velocity approximations. In the case of a BDF(1) scheme, we simply use the previous velocity approximation, i.e. $\widetilde{\mathbf{u}}_h^n = \mathbf{u}_h^n$ holds. The transport step is illustrated in Figure 7.7.3.

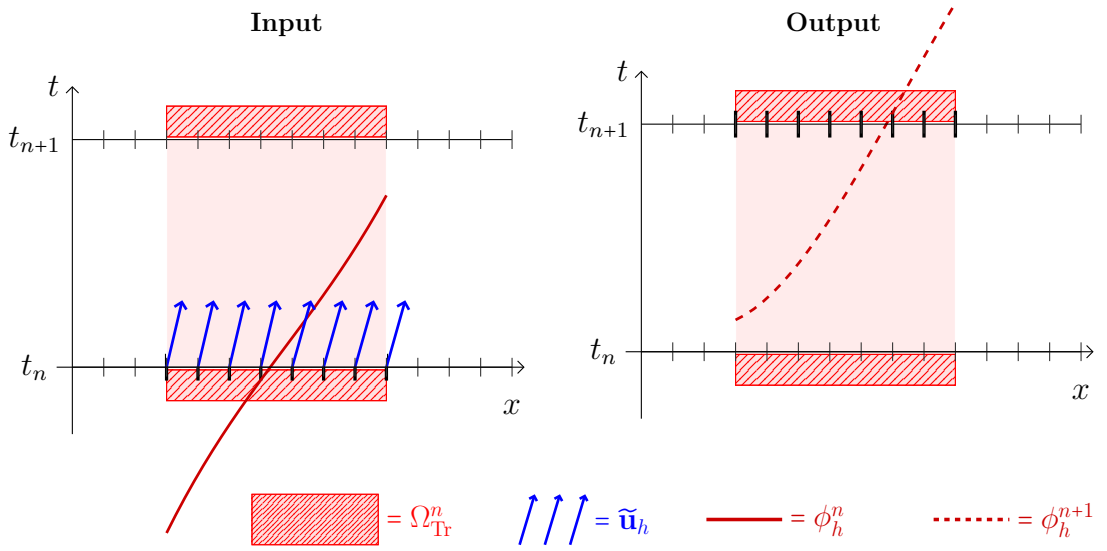



Figure 7.7.3: Line 15 of Algorithm 7.7.2: Transport step. The level set approximation $\phi_h^{n+1}$ is constructed by solving the transport equation on the transport domain $\Omega_{\mathrm{Tr}}^n$.

The resulting level set approximation $\phi_h^{n+1}$, defines the surface $\Gamma_h^{n+1} = \{\mathbf{x} \mid \phi_h^{n+1}(\mathbf{x}) = 0\}$ and the cut domain $\Omega_\Gamma^{n+1}$. On the surface $\Gamma_h^n$, we solve the surface Navier–Stokes system in line 19 of Algorithm 7.7.2. This method requires the previous velocity approximation $\mathbf{u}_h^n$ given on the surface $\Gamma_h^n$. Since we extended this velocity approximation to the transport domain $\Omega_{\mathrm{Tr}}^n$ (see Figure 7.7.2), it is well defined on $\Gamma_h^{n+1}$. The used stabilization in the TraceFEM leads to a resulting velocity approximation $\mathbf{u}_h^{n+1}$ defined on the cut domain $\Omega_\Gamma^{n+1}$. We visualize the surface Navier–Stokes step in Figure 7.7.4.

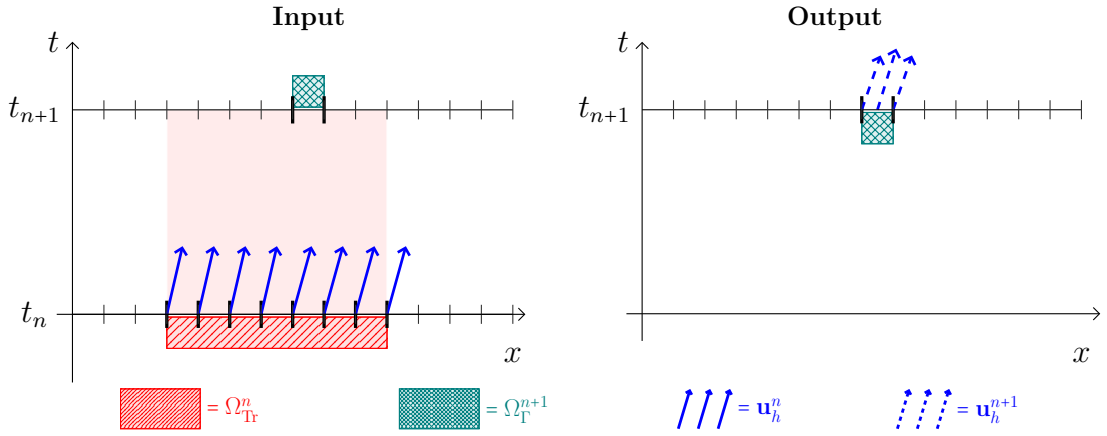



Figure 7.7.4: Line 19 of Algorithm 7.7.2: Navier–Stokes step. The velocity $\mathbf{u}_h^{n+1}$ is constructed by solving the surface Navier–Stokes system on the cut domain $\Omega_\Gamma^{n+1}$.

The resulting velocity approximation $\mathbf{u}_h^{n+1}$ and the level set approximation $\phi_h^{n+1}$ are used as input in the next time step.

## 7.8 Numerical experiments

In this section, we present numerical experiments to validate the method and to illustrate some (mathematical and physical) phenomena connected to the surface Navier–Stokes system. We start with simple validation examples, where the velocity is constant in space and time and the surface is moved with constant speed in a constant direction. As a next step, we consider more complex examples, where the velocity is not constant anymore, and the surface deforms over time. In the last validation example, we solve the Navier–Stokes system on a surface that exhibits a topological change. In all validation examples, a manufactured solution is available, and we calculate the accuracy of our numerical approximation.

After the validation of our algorithm, we present some examples to investigate the behavior of the numerical solution of the surface Navier–Stokes system. In the first example, in Section 7.8.3, we consider the continuous dependence of the solution on the initial data. In the last examples, we investigate the influence of the viscosity coefficient $\mu_0$ on the solution of the surface Navier–Stokes system.

Throughout all experiments, we use the parameters $k_{\mathbf{u}} = 2$ ($k_p = 1$), $k_g = 1$, and employ BDF(2) time discretization schemes. To obtain the second initial values for the BDF(2) scheme, we start with two time steps with time step size $\frac{\Delta t}{2}$, and apply a BDF(1) scheme. In the extension method, we use $L_P = 1$. The linear system resulting from discretizing the transport step is solved using a stabilized BCGS method with an incomplete LU decomposition as a preconditioner. All other linear systems are solved using a direct solver from the Intel MKL library. If not stated otherwise, the time interval is taken as $I = [0, 1]$, the evolving surfaces are embedded in the domain $\Omega = [-\frac{7}{3}, \frac{7}{3}]^3$, the viscosity is set to $\mu_0 = 0.05$. The initial mesh size is $h = 0.5$. The mesh size $h$ and the time step size $\Delta t$ are halved in each refinement step. We implement our methods using the finite element library `Netgen/NGSolve` [Sch97; Sch14] with the add-on `ngsxfem` [Leh+]. In the validation examples, we use Maple [Map] for computing the exact force terms corresponding to our manufactured solutions. [Sch25] contains the code used to perform the numerical experiments. The error metrics of interest are given by

$$(e_{\mathbf{u}})^2 := \Delta t \sum_{n=1}^{N} \|\mathbf{u}_h^n - \mathbf{u}^n\|_{\Gamma_h^n}^2, \qquad (e_{1,\mathbf{u}})^2 := \Delta t \sum_{n=1}^{N} \|\nabla_{\Gamma_h} \mathbf{u}_h^n - \nabla_{\Gamma_h} \mathbf{u}^n\|_{\Gamma_h^n}^2,$$
$$(e_p)^2 := \Delta t \sum_{n=1}^{N} \|p_h^n - p^n\|_{\Gamma_h^n}^2, \qquad (e_\Gamma)^2 := \Delta t \sum_{n=1}^{N} \|\phi^n\|_{\Gamma_h^n}^2 / \|1\|_{\Gamma_h^n}^2,$$

where $\|\cdot\|_{\Gamma_h^n}$ denotes the $L^2$-norm on the surface $\Gamma_h^n$. To construct time dependent level set functions, we start with the following exact stationary level set functions:

- $\phi_{\text{Sphere}}(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 - 1$,
- $\phi_{\text{Bicon}}(\mathbf{x}) = (x_1^2 + x_2^2 - 0.3)^2 + x_3^2 - 0.4$,
- $\phi_{\text{Torus}}(\mathbf{x}) = (\sqrt{x_1^2 + x_2^2} - 1)^2 + x_3^2 - 0.5^2$,
- $\phi_{\text{Kite}}(\mathbf{x}) = (-x_3^2 - x_1)^2 + x_2^2 + x_3^2 - 1$.

The first surface is a sphere, the second one is called a biconcave shape, the third one is a torus, and the last one is called 'kite'-geometry. We construct time dependent surfaces.

The first evolutions are the ones of surfaces (sphere, biconcave shape, and torus) slowly moving to the right and not changing the shape. In addition, we consider the evolution of a biconcave shape and a torus transforming into a sphere. The corresponding level set functions are given by

- $\phi_{\mathrm{S}\rightarrow}(\mathbf{x},t) = \phi_{\mathrm{Sphere}}(x_1 - 0.2t, x_2, x_3)$,
- $\phi_{\mathrm{B}\rightarrow}(\mathbf{x},t) = \phi_{\mathrm{Bicon}}(x_1 - 0.2t, x_2, x_3)$,
- $\phi_{\mathrm{T}\rightarrow}(\mathbf{x},t) = \phi_{\mathrm{Torus}}(x_1 - 0.2t, x_2, x_3)$,
- $\phi_{\mathrm{B}\rightarrow\mathrm{S}}(\mathbf{x},t) = (1-t)\cdot\phi_{\mathrm{Bicon}}(\mathbf{x}) + t\cdot\phi_{\mathrm{Sphere}}(\mathbf{x})$,
- $\phi_{\mathrm{T}\rightarrow\mathrm{S}}(\mathbf{x},t) = (1-t)\cdot\phi_{\mathrm{Torus}}(\mathbf{x}) + t\cdot\phi_{\mathrm{Sphere}}(\mathbf{x})$.

### 7.8.1 Validation examples I

We start the validation with a simple case, where the exact velocity is constant, and the exact pressure is zero. We choose a constant (in space) velocity $\mathbf{u}^0$ as the initial velocity and $\mathbf{f} = 0$ as the force term. It holds $\nabla_\Gamma \mathbf{u}^0 = \mathbf{E}_\Gamma \mathbf{u}^0 = \mathrm{div}_\Gamma\, \mathbf{u}^0 = 0$. Consequently, the exact pressure is zero ($p = 0$), and the velocity remains constant in both space and time, i.e., $\mathbf{u}(\mathbf{x},t) = \mathbf{u}^0$ holds for all $t \geq 0$. The exact solution of the velocity and the pressure are independent of the initial level set function and thus also independent of the surface and its curvature. Since the exact velocity is constant, the exact surface is transported constantly in the direction of the (fixed) velocity.

The initial velocity $\mathbf{u}^0 = (0.2, 0, 0)^T$ and level set functions $\phi^0 \in \{\phi_{\mathrm{Sphere}}, \phi_{\mathrm{Bicon}}, \phi_{\mathrm{Torus}}\}$ lead to the corresponding exact level set functions given by $\phi_{\mathrm{S}\rightarrow}$, $\phi_{\mathrm{B}\rightarrow}$, and $\phi_{\mathrm{T}\rightarrow}$, respectively.

This system is solved exactly (apart from rounding errors) using our method. The errors $e_{\mathbf{u}}$, $e_{1,\mathbf{u}}$, and $e_p$ are within machine accuracy limits, since the solution is contained in the finite element spaces. The error in the level set function converges at an order between 1.5 and 2, which is nearly optimal since we use $k_g = 1$. These results demonstrate that our method achieves optimal accuracy for this case.

### 7.8.2 Validation examples II

In the more complex validation examples, we consider three examples. The first one is a sphere moving to the right, where a (time-dependent) divergence free manufactured velocity field is used. The second example is a biconcave shape transforming into a sphere, where we show that our method is able to deal with varying curvature of the surface. The last example is a torus transforming into a sphere, where a topological change occurs, which makes this example more challenging. In all examples, the exact pressure is given by $p(\mathbf{x},t) = x_1 x_2 x_3$.

#### Sphere moving to the right

We use the exact level set function $\phi_{\mathrm{S}\rightarrow}$, which describes a sphere evolving with constant velocity to the right. The curvature of the surface remains constant in both space and time.

We define a tangential divergence-free velocity as $\widetilde{\mathbf{u}}_T = \mathbf{P}(\mathbf{n} \times \nabla_\Gamma \psi)$ with the stream function $\psi(\mathbf{x},t) = x_1 x_2 - 2t$. This function was also employed in the numerical experiments presented in Chapter 5. We add the constant velocity $\mathbf{u}_c = (0.2, 0, 0)^T$ that aligns

with the desired evolution of the surface $\Gamma(t)$. The full velocity is given by $\mathbf{u} = \widetilde{\mathbf{u}}_T + \mathbf{u}_c$, satisfying $\operatorname{div}_\Gamma \mathbf{u} = 0$. We choose the initial time step size as $\Delta t_0 = 0.25$. The results of our method are shown in Figure 7.8.1.

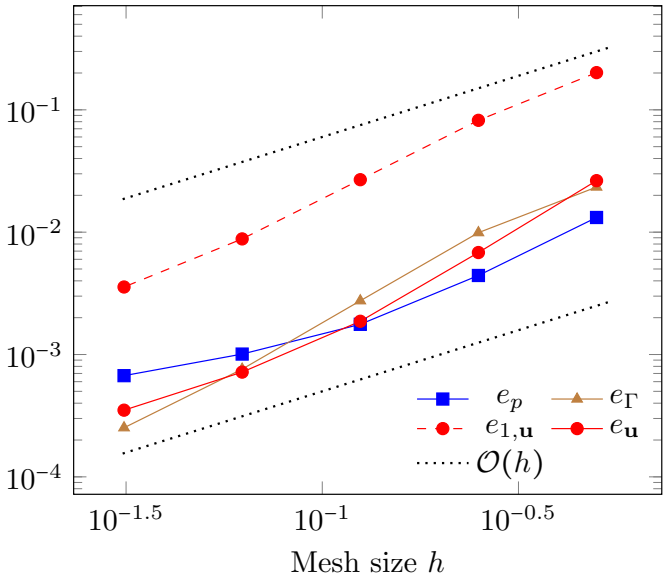



Figure 7.8.1: Navier–Stokes convergence test with sphere moving to the right.

The results demonstrate convergence of our method across all displayed error metrics. The convergence orders vary between 1 and 2 for the velocity and the level set function and between 0.5 and 1.5 for the pressure. The $h^1$-error of the velocity is larger than the $L^2$-error, which is an expected behavior.

The convergence rates are not optimal, which may be due to the strong nonlinearity between the velocity and the level set function. We use a simple extrapolation of the velocity field for the transport step, which may not be sufficient to achieve optimal convergence rates. The results indicate that our method is capable of solving the surface Navier–Stokes system with a moving surface, even in the presence of strong nonlinearity.

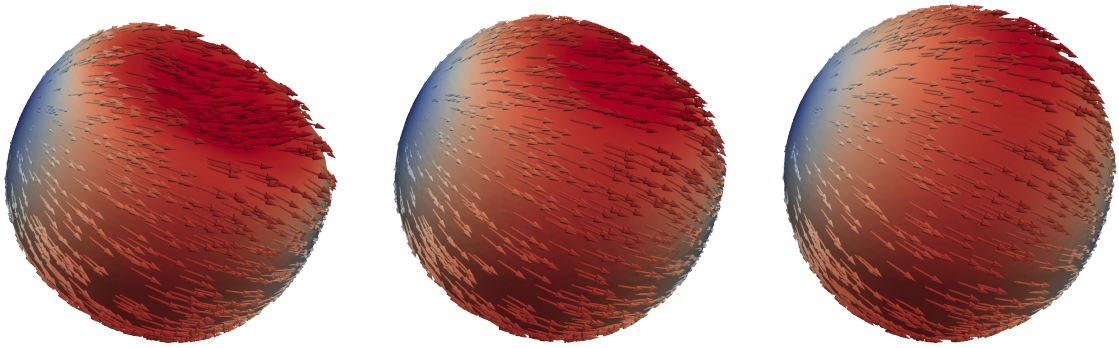
**Biconcave shape transforming into a sphere**

In the next example, we present a convergence test on a surface with varying curvature. We select the exact level set function $\phi = \phi_{\mathrm{B}\to\mathrm{S}}$ which describes a biconcave shape that deforms into a sphere. Since the surface evolution is prescribed by the level set function in this manufactured example, we can express the normal component of the velocity in terms of the level set function by

$$\mathbf{u} \cdot \mathbf{n} = u_N = V_\Gamma = -\frac{\partial_t \phi}{\mathbf{n} \cdot \nabla \phi}.$$

The tangential component of the velocity is chosen as $\mathbf{u}_T = \mathbf{P}\mathbf{e}_1$ with $\mathbf{e}_1 = (1,0,0)^T$. This velocity does not satisfy the divergence-free condition, i.e., $\operatorname{div}_\Gamma \mathbf{u} \neq 0$ holds. To obtain a closed system, we substitute the second equation ($\operatorname{div}_\Gamma \mathbf{u} = 0$) of the Navier–Stokes system (7.2.2) with the modified equation $\operatorname{div}_\Gamma \mathbf{u} = q$ where $q$ represents the divergence of the manufactured velocity.

The surface evolution and the velocity are shown in Figure 7.8.2.



(a) $t = 0$ (b) $t = 0.5$ (c) $t = 1$

Figure 7.8.2: Snapshots of biconcave shape deforming into a sphere. The glyphs represent the velocity field $\mathbf{u}$. The coloring is done by the value of $\|\mathbf{u}\|_2$ from blue (low) to red (high).

The initial time step size is chosen as $\Delta t_0 = 0.1$ and halved in each refinement step. Figure 7.8.3 displays the results of the convergence test.

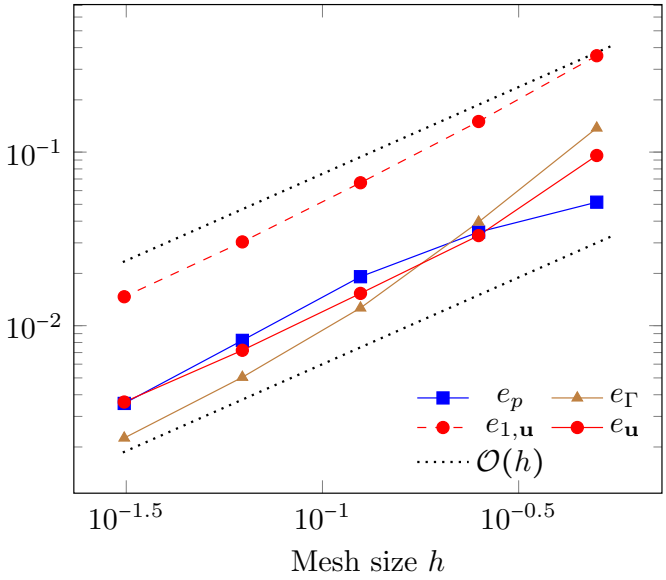



Figure 7.8.3: Navier–Stokes convergence test with a biconcave surface deforming to a sphere.

We observe that the method is convergent with order one in all error quantities. Errors exceed those seen previously due to varying curvature present in this example. This experiment indicates, that the proposed method is able to approximate the solution of the Navier–Stokes system on a surface with varying curvature.

Gradient of level set function

We use the example of the biconcave shape deforming into a sphere, to take a closer look at the level set approximation $\phi_h$ close to the surface. In Chapter 4, where we introduced the TraceFEM, we discussed the necessity of a level set function that is "close" to a signed distance function. This is important for the trace finite element method, as it allows us

accurate determination of the zero level set and thus the surface. If the level set function becomes too steep or too flat, the numerical approximation of the surface and surface integrals deteriorates. In existing literature, one often employs reinitializing methods for the level set functions after some time steps, to maintain proximity to a signed distance function. These reinitialization techniques are often computationally expensive, and the zero level set is often not preserved exactly during the reinitialization process. To avoid this, we choose a tailor made velocity field extension (normal aligned), that should (as the velocity in the transport step) keep the level set function close to a signed distance function over time. See the explanation in Section 7.2. One way to measure the quality of the level set function, with respect to the signed distance property, is via the Eikonal equation

$$\|\nabla\phi\|_2 = 1,$$

which is fulfilled for signed distance functions. We measure the quality of our level set function (in terms of approximating a signed distance function) by the following error quantity

$$e_{\nabla\phi,T} := \fint_T \|\nabla\phi\|_2 \,\mathrm{d}\mathbf{x}, \quad \text{for } T \in \mathcal{T}_h.$$

For a level set function $\phi$ that fulfills the Eikonal equation, it holds $e_{\nabla\phi,T} = 1$ for all $T \in \mathcal{T}_h$.

In Figure 7.8.4, we show the maximal and minimal size of $e_{\nabla\phi,T}$ on the cut elements $\mathcal{T}_\Gamma$ and on the elements within the transport domain $\Omega_{\mathrm{Tr}}$ for each time step. We use a mesh with resolution $h = 2^{-5}$.

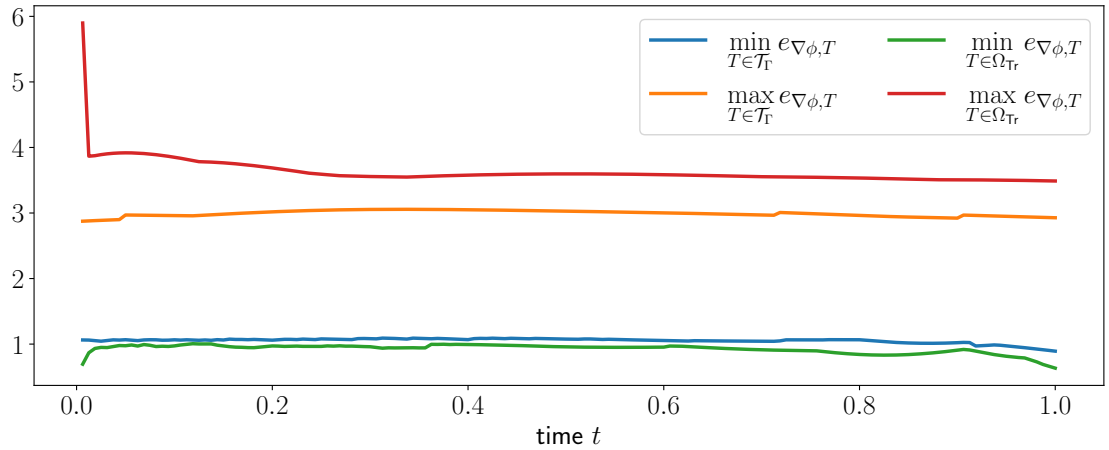
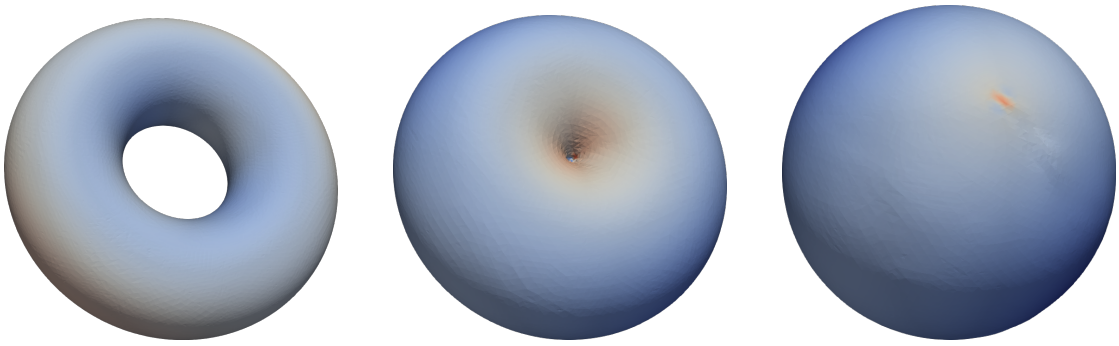



Figure 7.8.4: Eikonal error $e_{\nabla\phi,T}$ on $\Omega_\Gamma$ and $\Omega_{\mathrm{Tr}}$ over time.

The results show that the error quantity $e_{\nabla\phi,T}$ is between 1 and 3 for all time steps on the cut elements and between $\sim 0.8$ and 4 on the elements of the transport domain after the first time step. In the literature, cf. [GR11], one usually employs reinitialization techniques if the condition $\frac{1}{c} \leq e_{\nabla\phi,T} \leq c$ for some $T$ with a constant $c \in [5, 10]$ is violated. Thus, no reinitialization would be used in this example. We emphasize, that the deviation of the level set function from a signed distance function is not increasing over time. This indicates that our method is able to solve the level set equation without the need of reinitialization, and we have a stable implementation concerning the level set quality for this smooth example.

### Torus transforming into a sphere

In the last example with a manufactured solution, we test if our method is able to perform topology changes. In general, a topology change is a challenging problem for numerical methods. Surface finite element methods (SFEM), which are Lagrangian approaches, are not suitable to discretize topology changes, since the surface mesh needs to be split or merged. In contrast, trace finite element methods (which are Eulerian) are in general suited for topology changes, cf. the recent paper [OR25]. We test if our method is able to solve the Navier–Stokes system on a surface that undergoes a topology change. We use the exact level set function $\phi = \phi_{\mathrm{T}\to\mathrm{S}}$ which describes a torus (genus 1) that deforms into a sphere (genus 0), cf. Figure 7.8.5 for a visualization.



(a) $t = 0$ (b) $t \approx 0.5$ (c) $t = 1$

Figure 7.8.5: Snapshots of $\Gamma_h$ describing a torus deforming into a sphere.

The normal component of the velocity is given by the predefined surface evolution. Again, we choose the tangential component as $\mathbf{u}_T = \mathbf{P}\mathbf{e}_1$. The initial time step size is taken as $\Delta t_0 = 0.1$ and the divergence free condition is again replaced as in the previous example.

The results of the convergence test are shown in Figure 7.8.6.

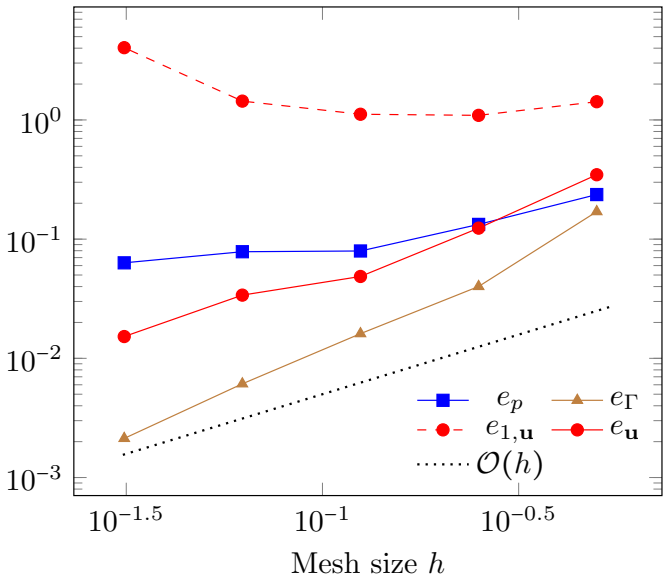



Figure 7.8.6: Navier–Stokes convergence test with a torus surface deforming to a sphere.

Once more, we observe convergence of order close to one for the $L^2-$ error of the velocity ($e_{\mathbf{u}}$) and the surface error ($e_\Gamma$). However, the errors in the gradient of the velocity ($e_{1,\mathbf{u}}$) and in the pressure ($e_p$) do not converge. Notably higher errors arise than in the previous examples. We assume the loss of convergence and the higher errors are caused by the topology change. See [OR25] for a discussion of the challenges arising from topology changes.

### 7.8.3 Continuous dependence on initial values

In this section, we test the dependence of the solution on initial values. As we mentioned when introducing the surface Navier–Stokes system, there is no theory available for the well-posedness of the system. Thus, we do not know if the solution is continuously depending on the initial values. If there is no continuous dependence, small perturbations (e.g., caused by the numerical scheme), may strongly increase over time and lead to a blow-up of the solution; thus constructing stable numerical schemes becomes more complicated. For the surface Stokes system, it is known, that perturbations in the initial values are damped over time. We test if similar results hold for the surface Navier–Stokes system.

We use the three examples from the previous section. The exact level set function is taken as $\phi \in \{\phi_{S\to}, \phi_{B\to S}, \phi_{T\to S}\}$ with the exact velocity and pressure functions as described in the previous examples. We disturb the initial level set function or the initial velocity by adding an error with high frequency. The initial error is given by

$$e_{\text{Init}}(\mathbf{x}) = \frac{\sin(50\pi x_1 x_2 x_3)}{10^2}.$$

We run two test cases. In the first one we perturb the initial level set function. We solve the system with the exact ($\phi^0_{\text{Ex}} = \phi(0)$) and the disturbed ($\phi^0_{\text{dis}} = \phi(0) + e_{\text{Init}}$) level set function and the exact velocity as initial value. The error of interest is given by

$$e^n_\phi = \|\phi^n_{\text{Ex}}\|_{L^2(\Gamma^n_{\text{dis}})}, \quad \text{for } n = 0, \dots, N,$$

where $\Gamma_{\text{dis}}$ is the surface generated by the disturbed level set function $\phi_{\text{dis}}$. In Table 7.8.7, we show the errors in the initial functions at $n = 0$ and the errors at the final time step $N$ and the growth rate (rate $= e^N_\phi / e^0_\phi$).

| | $\phi_{S\to}$ | | | $\phi_{B\to S}$ | | | $\phi_{T\to S}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| $h$ | $e^0_\phi$ | $e^N_\phi$ | rate | $e^0_\phi$ | $e^N_\phi$ | rate | $e^0_\phi$ | $e^N_\phi$ | rate |
| 0.5 | $6.1\cdot10^{-3}$ | $8.8\cdot10^{-3}$ | 1.4 | $5.9\cdot10^{-3}$ | $8.3\cdot10^{-2}$ | 14.1 | $6.98\cdot10^{-3}$ | $1.51\cdot10^{-2}$ | 2.2 |
| 0.25 | $5.1\cdot10^{-3}$ | $1.1\cdot10^{-3}$ | 0.2 | $5.4\cdot10^{-3}$ | $3.2\cdot10^{-3}$ | 0.6 | $7.83\cdot10^{-3}$ | $4.27\cdot10^{-3}$ | 0.5 |
| 0.125 | $8\cdot10^{-3}$ | $1.5\cdot10^{-3}$ | 0.2 | $9\cdot10^{-3}$ | $4\cdot10^{-3}$ | 0.4 | $1.09\cdot10^{-2}$ | $1\cdot10^{-2}$ | 0.9 |
| 0.063 | $1.5\cdot10^{-2}$ | $4.2\cdot10^{-3}$ | 0.3 | $1.5\cdot10^{-2}$ | $6.8\cdot10^{-3}$ | 0.4 | $1.66\cdot10^{-2}$ | $6.13\cdot10^{-3}$ | 0.4 |
| 0.031 | $2.2\cdot10^{-2}$ | $1.4\cdot10^{-2}$ | 0.7 | $2.1\cdot10^{-2}$ | $1.6\cdot10^{-2}$ | 0.7 | $2.47\cdot10^{-2}$ | $1.59\cdot10^{-2}$ | 0.6 |

Figure 7.8.7: Comparison of the initial errors and the errors at the final time in the geometry when perturbing the initial values.

In the second test case, we perturb the initial velocity instead of the initial level set function. We solve the system with the exact ($\mathbf{u}^0_{\text{Ex}} = \mathbf{u}^0$) and the disturbed ($\mathbf{u}^0_{\text{dis}} = \mathbf{u}^0 + e_{\text{Init}} \cdot (1,1,1)^T$) velocity as initial value and the exact level set function. The

error of interest is given by

$$e_{\mathbf{u}}^n = \|\mathbf{u}_{\mathrm{Ex}}^n - \mathbf{u}_{\mathrm{dis}}^n\|_{L^2(\Gamma_{\mathrm{Ex}}^n)}, \quad \text{for } n = 0, \dots, N,$$

where $\Gamma_{\mathrm{Ex}}^n$ is the surface generated using the exact initial values. We show the errors in the initial functions at time step $n = 0$ and the errors at the final time step $n = N$ and the growth rate (rate $= e_{\mathbf{u}}^N / e_{\mathbf{u}}^0$) in Table 7.8.8.

| | $\phi_{\mathrm{S}\to}$ | | | $\phi_{\mathrm{B}\to\mathrm{S}}$ | | | $\phi_{\mathrm{T}\to\mathrm{S}}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| $h$ | $e_{\mathbf{u}}^0$ | $e_{\mathbf{u}}^N$ | rate | $e_{\mathbf{u}}^0$ | $e_{\mathbf{u}}^N$ | rate | $e_{\mathbf{u}}^0$ | $e_{\mathbf{u}}^N$ | rate |
| 0.5 | $3 \cdot 10^{-2}$ | $1.4 \cdot 10^{-2}$ | 0.5 | $2.9 \cdot 10^{-2}$ | $1.5 \cdot 10^{-2}$ | 0.5 | $4.18 \cdot 10^{-2}$ | 0.11 | 2.6 |
| 0.25 | $3.5 \cdot 10^{-2}$ | $1 \cdot 10^{-2}$ | 0.3 | $3.6 \cdot 10^{-2}$ | $1 \cdot 10^{-2}$ | 0.3 | $4.4 \cdot 10^{-2}$ | $1.3 \cdot 10^{-2}$ | 0.3 |
| 0.125 | $5 \cdot 10^{-2}$ | $1.2 \cdot 10^{-2}$ | 0.2 | $4.6 \cdot 10^{-2}$ | $1.4 \cdot 10^{-2}$ | 0.3 | $5.21 \cdot 10^{-2}$ | $6.82 \cdot 10^{-2}$ | 1.3 |
| 0.063 | $4.4 \cdot 10^{-2}$ | $2.2 \cdot 10^{-2}$ | 0.5 | $4.2 \cdot 10^{-2}$ | $9.7 \cdot 10^{-3}$ | 0.2 | $5.75 \cdot 10^{-2}$ | $2.25 \cdot 10^{-2}$ | 0.4 |
| 0.031 | $4.5 \cdot 10^{-2}$ | $6.1 \cdot 10^{-2}$ | 1.4 | $4.2 \cdot 10^{-2}$ | $3.2 \cdot 10^{-2}$ | 0.8 | $5.65 \cdot 10^{-2}$ | 0.13 | 2.2 |

Figure 7.8.8: Comparison of the initial errors and the errors at the final in the velocity time when perturbing the initial values.

From Tables 7.8.7 and 7.8.8, we observe that for most examples and mesh sizes, the difference between the solution using the exact initial errors and the ones using the disturbed initial errors is decreasing over time for all examples. The growth rate is below one for almost all examples and all mesh resolutions after the first refinement. In the examples where an increase of the error is observed, the growth rate is still not larger than 2.2 after the first refinement, so the error does not blow up significantly.

These experiments suggest that the underlying smooth problem could possibly be well-posed, meaning that the solution might depend continuously on the initial values.

### 7.8.4 Influence of the viscosity coefficient $\mu_0$

We have shown, that we have a stable method to discretize the surface Navier–Stokes system. Now, we investigate the behavior of the velocity and the surface, when no outer force term is present, i.e., $\mathbf{f} = 0$ holds. For the best of our knowledge, there are no experimental studies available in the literature for this setting. In the related works where the surface Navier–Stokes system is discretized, cf. [KV23; RV18], an additional bending term is added to stabilize the system. This bending term is not present in this section.

We investigate the influence of the viscosity coefficient $\mu_0$ on the solution of the surface Navier–Stokes system. We use different initial level set functions $\phi^0$ and different initial velocities $\mathbf{u}^0$ and show the resulting surfaces for different values for $\mu_0$. There are no exact solutions available in these examples, thus we can only measure the error in the surface divergence $\mathrm{div}_{\Gamma_h} \mathbf{u}_h$ of the velocity which should vanish. In each experiment in this section, we use a mesh with a maximum mesh size of $h = 2^{-5} = 0.03125$ and time step size $\Delta t = 0.2 \cdot 2^{-4} = 0.0125$ if not stated otherwise. The viscosity coefficient $\mu_0$ is varied between 0.01 and 100. The time interval is taken as $I = [0, 1]$. In all figures, the glyphs represent the velocity field $\mathbf{u}_h$. The coloring is done by the value of $\|\mathbf{u}_h\|_2$ from blue (low) to red (high).

We show results for different choices of initial surfaces and velocities. The initial surfaces represent a sphere or a kite geometry, the corresponding level set functions are

taken as $\phi^0 \in \{\phi_{\text{Sphere}}, \phi_{\text{Kite}}\}$. The initial velocities are given by $\mathbf{u}^0 \in \{\mathbf{u}_{\circlearrowleft}, \mathbf{u}_{\rightarrow}\}$ where $\mathbf{u}_{\circlearrowleft} = (-x_2, x_1, 0)^T$ is a rotation around the $x_3$-axis which is tangential to the sphere (given by the zero level of $\phi_{\text{Sphere}}$). The discontinuous velocity $\mathbf{u}_{\rightarrow}$ is locally pointing to the right for $x_1 > 0.5$ and zero otherwise, i.e., it is given by

$$\mathbf{u}_{\rightarrow} = \begin{cases} (0.5, 0, 0)^T, & \text{if } x_1 > 0.5 \\ (0, 0, 0)^T, & \text{else.} \end{cases}$$

**Example 1: Sphere with rotation**

In the first test case, we choose $\phi^0 = \phi_{\text{Sphere}}$ and $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$. For this example, we examine the convergence rates of the surface divergence in the approximated velocity field. We investigate the $L^2L^2$–divergence error, which is given by

$$e^2_{\text{div}_\Gamma} := \Delta t \sum_{n=1}^{N} \| \operatorname{div}_{\Gamma_h} \mathbf{u}_h^n \|^2_{\Gamma_h^n} / \|1\|^2_{\Gamma_h^n}.$$

In Figure 7.8.9, we show the $L^2L^2$–divergence error for initial mesh and time step sizes $h_0 = 0.25$ and $\Delta t_0 = 0.1$.

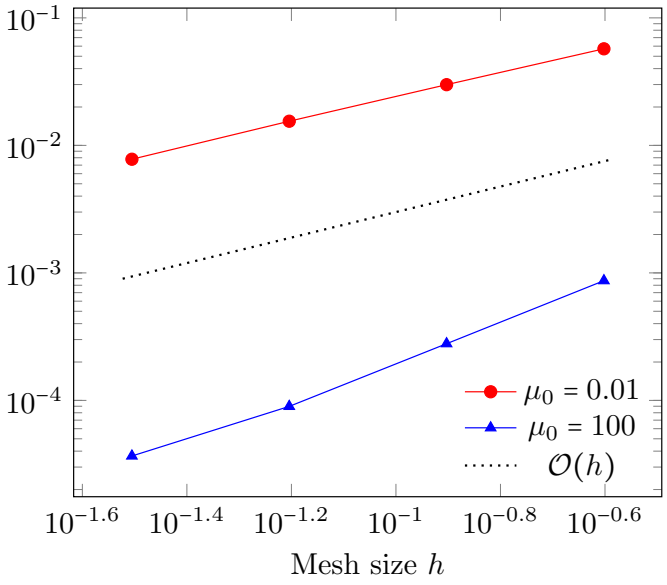



Figure 7.8.9: $L^2L^2$–divergence error $e_{\text{div}_\Gamma}$ for $\phi^0 = \phi_{\text{Sphere}}$, $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$, and $\mu_0 \in \{0.01, 100\}$.

The results show that the divergence error is converging with convergence rate $\mathcal{O}(h)$, which is expected, since we used a linear level set approximation to define $\Gamma_h$ and $\operatorname{div}_{\Gamma_h}$. The errors are larger for the low viscosity coefficient $\mu_0 = 0.01$ compared to the high viscosity coefficient $\mu_0 = 100$. We assume that the larger errors are caused by the higher curvature (as shown in the following figures) that results from the lower viscosity.

The convergence of the surface divergence error suggests that our method generally is converging to a solution of the surface Navier–Stokes system. When we combine this finding with the validation results from previous sections, we can conclude that our method is likely able to accurately approximate the solution of the surface Navier-

Stokes system. Therefore, we can expect that the behavior shown in the following figures represents this solution.

We take a look at the evolution of the surface over time. Snapshots of the surface and the velocity field at different time steps for $\mu_0 = 0.01$ are shown in Figure 7.8.10 and for $\mu_0 = 100$ in Figure 7.8.11.

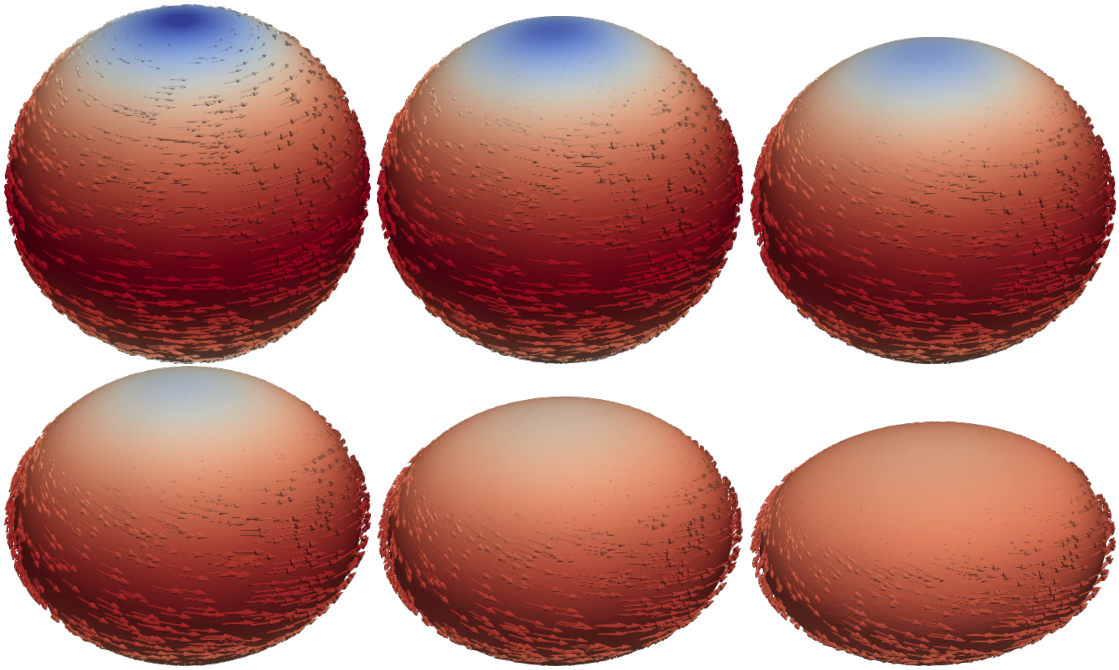


Figure 7.8.10: Surface and velocity over time for $\phi^0 = \phi_{\text{Sphere}}$, $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$, and $\mu_0 = 0.01$.

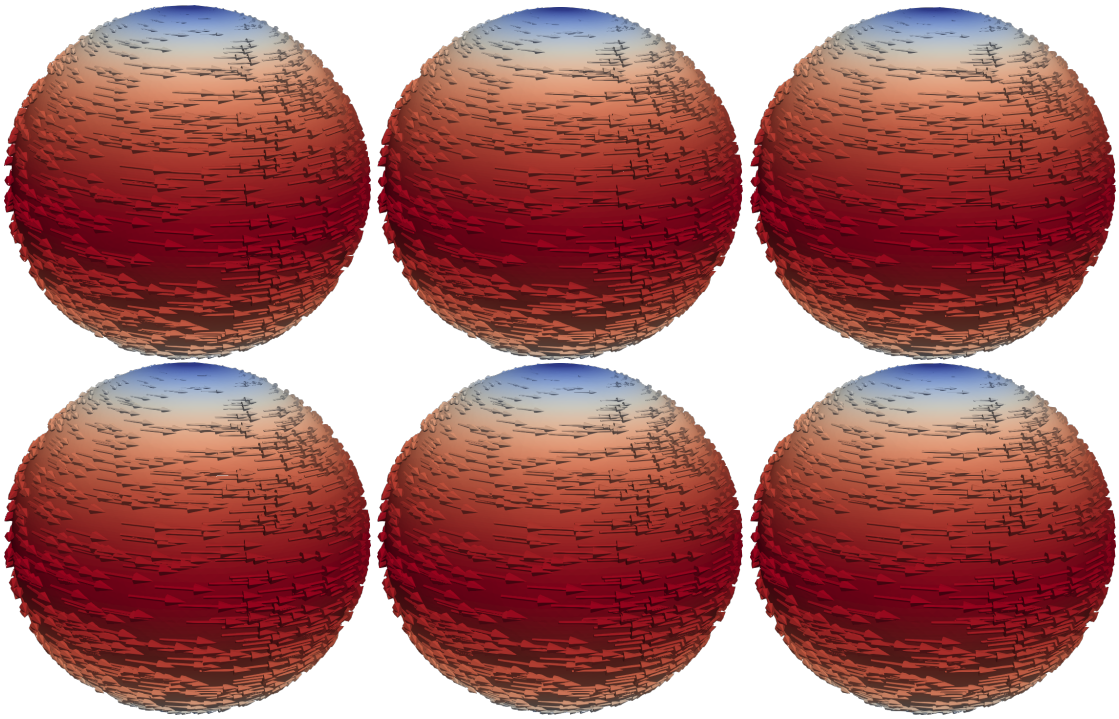


Figure 7.8.11: Surface and velocity over time for $\phi^0 = \phi_{\text{Sphere}}$, $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$, and $\mu_0 = 100$.

We observe that in the low viscosity case ($\mu_0 = 0.01$), the surface becomes "flatter" over time, i.e., the curvature decreases close to the $x_3$-axis and increases in regions further away from the $x_3$-axis. The deformation can be explained by the acting of "centrifugal-like" forces induced by the rotation $\mathbf{u}_{\circlearrowleft}$. The initial velocity field is tangential to the surface, but it does not stay tangential over time and the surface is deforming. This illustrates, that in the surface Navier–Stokes system, normal velocities can arise, even if the initial velocity field is purely tangential.

Increasing the viscosity coefficient to $\mu_0 = 100$ leads to a different behavior of the surface. We observe that the surface is only deforming very slowly, i.e., the curvature is nearly constant over time. The velocity field is also nearly constant over time. When we compare the initial surface with the final one, we see that the surface is a bit "flatter" than the initial surface, but the deformation is very small.

Also, we compare the final surfaces (at $t = t_{\text{end}} = 1$) for varying viscosity coefficients $\mu_0 \in \{0.01, 0.1, 1, 10, 100\}$. We take a "slice" of the surface at $x_2 = 0$ to be able to compare different surfaces in one single plot. The results are shown in Figure 7.8.12.

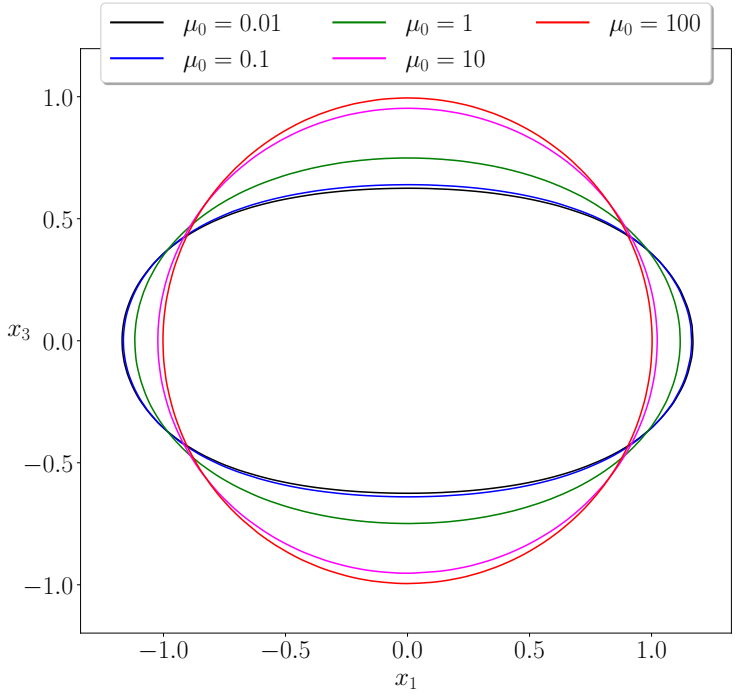



Figure 7.8.12: Slice of surfaces at $t = 1$ for different viscosity coefficients $\mu_0$ for initial values $\phi^0 = \phi_{\text{Sphere}}$ and $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$.

We observe that the dependence of the surface deformation on the viscosity coefficient is "anti-proportional" in the sense that decreasing the viscosity leads to a stronger deformation of the final surface. It seems like the strength of the deformation depends continuously on the viscosity coefficient $\mu_0$.

**Example 2: Kite with rotation**

In the second test case, we choose the initial values $\phi^0 = \phi_{\text{Kite}}$ and $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$. The initial velocity field $\mathbf{u}^0$ is not tangential to the surface in this case and the initial surfaces

has varying curvature. The results for the low viscosity case $\mu_0 = 0.01$ are shown in Figure 7.8.13.

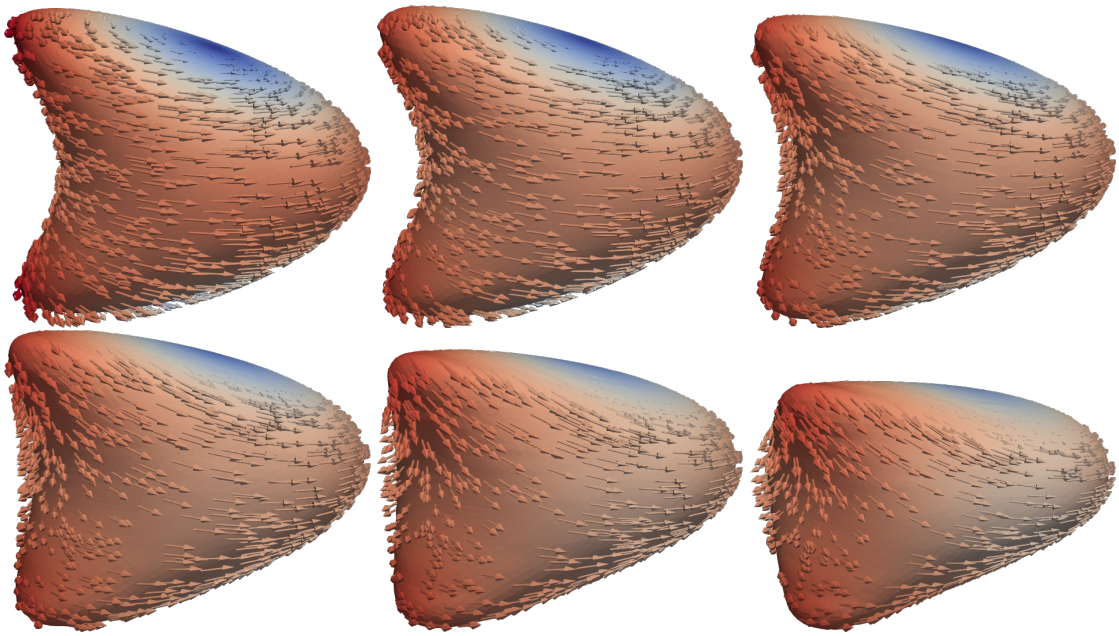


Figure 7.8.13: Surface and velocity over time for $\phi^0 = \phi_{\text{Kite}}$, $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$, and $\mu_0 = 0.01$.

Increasing the viscosity coefficient to $\mu_0 = 100$ leads to the results shown in Figure 7.8.14.

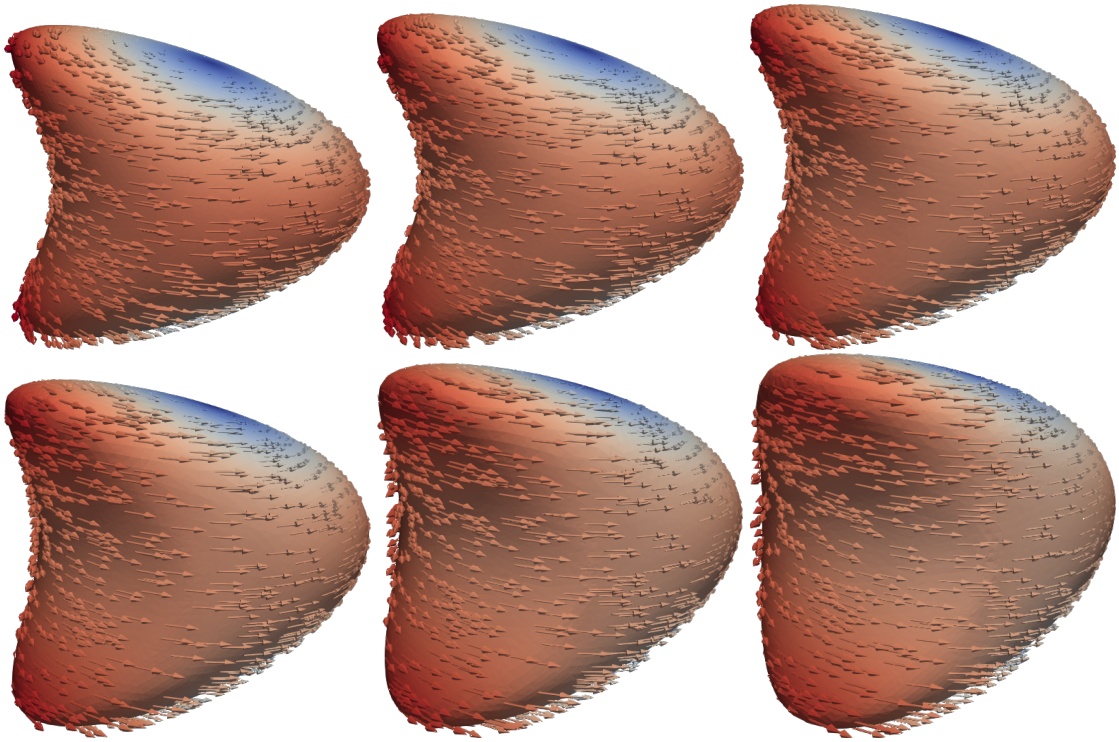


Figure 7.8.14: Surface and velocity over time for $\phi^0 = \phi_{\text{Kite}}$, $\mathbf{u}^0 = \mathbf{u}_{\circlearrowleft}$, and $\mu_0 = 100$.

We observe similar behavior as in the first example. In the low viscosity case ($\mu_0 = 0.01$), the surface becomes "flatter" over time, e.g. the curvature on the left side of the initial

surface is getting smaller. We observe that the surface is rotating around the $x_3$-axis. This rotation was difficult to observe in the first test case, since the initial surface was an axis-symmetric sphere. The figures indicate a local influence of the surface curvature on the velocity field. On the left side of the initial surface, the curvature is higher than on the right side. We see that the velocity in the regions with higher curvature behaves differently over time from the velocity in the regions with lower curvature. While the velocity stays close to the initial rotation in the smoother regions of $\Gamma_h$, it changes the direction in the regions with higher curvature. This indicates the influence of the curvature on the velocity field.

In the high viscosity case ($\mu_0 = 100$), the surface is only slightly deformed, i.e., the curvature is nearly constant over time. The velocity field is also nearly constant over time. The surface is rotating around the $x_3$-axis, but it does not deform significantly.

**Example 3: Sphere with velocity pointing to the right**

In the third test case, we choose $\phi^0 = \phi_{\text{Sphere}}$ and $\mathbf{u}^0 = \mathbf{u}_{\rightarrow}$. The velocity field $\mathbf{u}_{\rightarrow}$ is locally pointing to the right for $x_1 > 0.5$. Thus, the velocity is nonzero only on the right side of the initial surface.

For the low viscosity coefficient $\mu_0 = 0.01$, the results are shown in Figure 7.8.15.

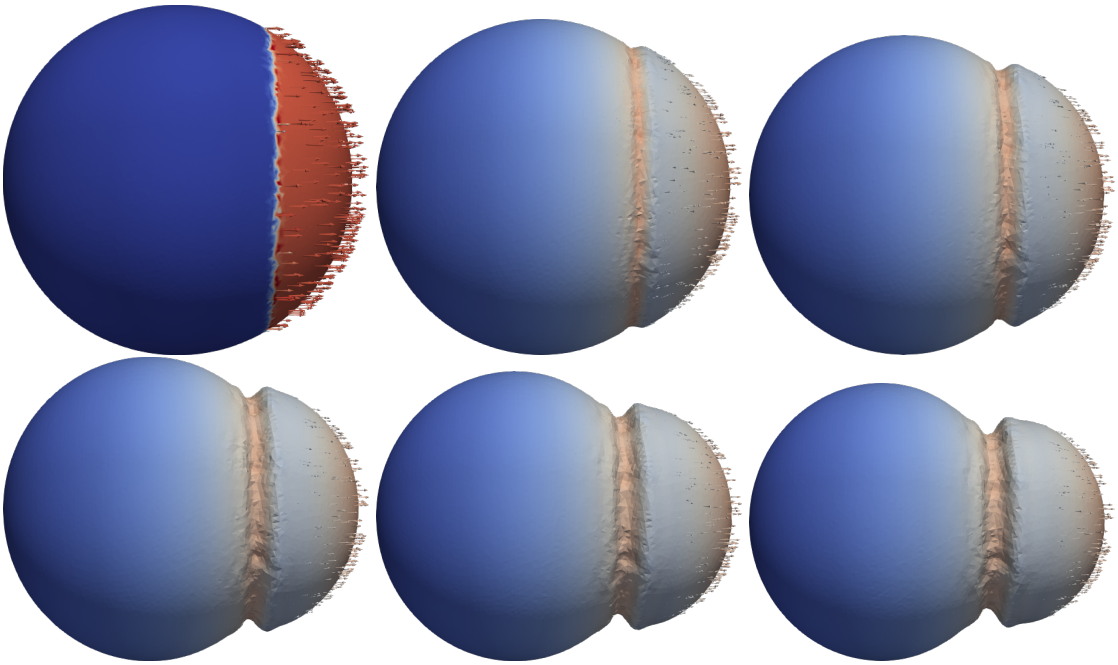


Figure 7.8.15: Surface and velocity over time for $\phi^0 = \phi_{\text{Sphere}}$, $\mathbf{u}^0 = \mathbf{u}_{\rightarrow}$, and $\mu_0 = 0.01$.

We observe that the surface is moving to the right, where the far right side of the surface is moving faster than the left side. This leads to a deformation of the surface. One can distinguish two parts of the surface: the right part, which is moving faster to the right, and the left part, which is almost spherical and stays nearly unchanged. The norm of the velocity field is getting smaller over time, but the velocity still points to the right at the right-hand side of the surface and is almost zero on the left-hand side. In another experiment, we have seen that increasing the final time leads to a separation of the surface into two parts, where the right smaller part is moving faster than the left part.

The results for the high viscosity coefficient $\mu_0 = 100$ are shown in Figure 7.8.16.

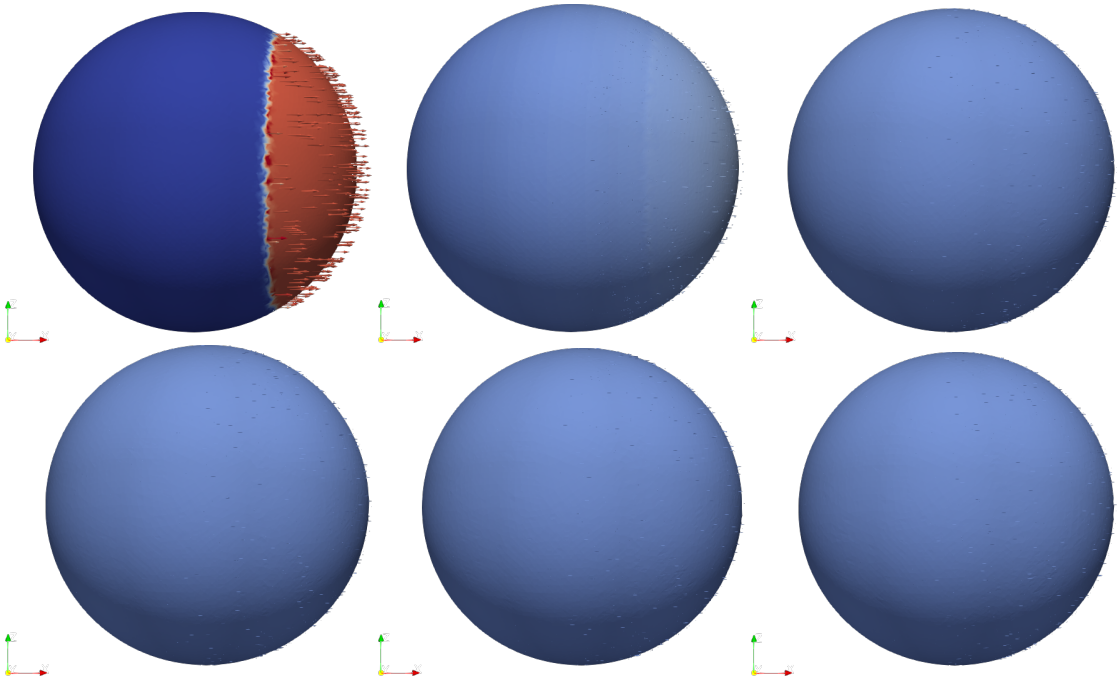


Figure 7.8.16: Surface and velocity over time for $\phi^0 = \phi_{\text{Sphere}}$, $\mathbf{u}^0 = \mathbf{u}_{\rightarrow}$, and $\mu_0 = 100$.

We see that the velocity is nearly constant after a short time (the second image is taken at the first time step at $t = 0.0125$). After the first time step, the velocity is constantly pointing to the right on the whole surface, and we do not see any local difference anymore. This constant velocity field leads to a constant movement of the surface to the right and no deformation of the surface. The surface keeps its spherical shape and is slowly moving to the right.

**Summary and heuristic explanation**

We describe a heuristic that explains the behavior of the surface and the velocity in the shown examples. Summarizing the similarities in the behavior of the solutions from the test cases, we observe that the surface is in general deformed strongly for the small viscosity coefficient $\mu_0 = 0.01$, while its shape is nearly unchanged for the large viscosity coefficient $\mu_0 = 100$. For example, in the second test case ($\phi_{\text{Kite}}$, $\mathbf{u}_{\circlearrowleft}$), the surface keeps its kite-like shape for $\mu_0 = 100$ and is only rotated by the velocity field. When the small viscosity coefficient $\mu_0 = 0.01$ is used, the surface is also rotated, but additionally it is deformed strongly, such that the final surface is much "flatter" than the initial one.

The change (over time) in the velocity obtained by the surface Navier–Stokes system, can be expressed by the first equation of (7.2.2) with $\mathbf{f} = 0$, given by

$$\dot{\mathbf{u}} = 2\mu_0 \operatorname{div}_\Gamma \mathbf{E}_\Gamma \mathbf{u} - \operatorname{div}_\Gamma(p\mathbf{P}).$$

Thus, for a large $\mu_0 \gg 1$, the first term ($\operatorname{div}_\Gamma \mathbf{E}_\Gamma \mathbf{u}$) dominates the evolution of $\mathbf{u}$ and for small $\mu_0 \ll 1$, the second term ($\operatorname{div}_\Gamma(p\mathbf{P})$) dominates. We first consider the case for high viscosity coefficients $\mu_0 \gg 1$. The dominating term is the diffusion term $\operatorname{div}_\Gamma \mathbf{E}_\Gamma \mathbf{u}$, that leads to a dissipation of the kinetic energy. In the third test case ($\phi_{\text{Sphere}}$, $\mathbf{u}_{\rightarrow}$),

the velocity field has huge local differences at $t = 0$, i.e., it is zero on the left-hand side of the surface and non-zero on the right-hand side. The diffusion term smooths the velocity field over time, such that the velocity field becomes nearly constant over the whole surface after one time step. When the velocity field is constant (in space), it stays unchanged by the surface Navier–Stokes system (cf. Section 7.8.1), and thus the surface moved to the right without changing its shape. In the first two test cases, the initial velocity field $\mathbf{u}_{\circlearrowleft}$ satisfies $\mathbf{E}_\Gamma \mathbf{u}_{\circlearrowleft} = 0$, i.e., $\mathbf{u}_{\circlearrowleft}$ is a so-called "Killing vector field". Thus, the diffusion term $\operatorname{div}_\Gamma \mathbf{E}_\Gamma \mathbf{u}$ has no influence on the velocity, and it stays nearly constant over time. This leads to a rotation of the surface around the $x_3$-axis, which is an axis of symmetry of the sphere ($\phi = \phi_{\text{Sphere}}$).

In the case of small viscosity coefficients $\mu_0 \ll 1$, the second term ($\operatorname{div}_\Gamma(p\mathbf{P}) = \nabla_\Gamma p - p\kappa\mathbf{n}$ with $\kappa$ being the mean curvature of $\Gamma(t)$) dominates the evolution of the velocity field. It takes the surface curvature into account. Thus, we have a stronger influence of the surface curvature on the velocity. We clearly observe this in the second test case ($\phi_{\text{Kite}}$, $\mathbf{u}_{\circlearrowleft}$), where the surface is deformed strongly, and the velocity depends on the local curvature of the surface.

# Chapter 8

# Summary and Outlook

The final chapter of this thesis is organized into two parts. In Section 8.1, we summarize the key findings and contributions made in the research on the discretization of surface Navier-Stokes equations. Potential future research directions and open problems are discussed in Section 8.2.

## 8.1 Summary

In this doctoral thesis, we explored a TraceFEM-based discretization of Navier-Stokes equations on evolving surfaces. Our primary focus was on the development of novel computational techniques that integrate advanced concepts from TraceFEM for surface PDEs, BDF methods for discrete time derivatives, DG schemes for level set transport equations, and ghost penalty methods for extending level set functions. The motivation for this work stemmed from the increasing relevance of modeling fluid flows on deformable surfaces in various scientific and engineering applications, such as biological membranes.

We began by establishing the mathematical framework for describing evolving surfaces by introducing two coordinate systems. The local (curvilinear) coordinate system is well-suited for modeling more complex fluid properties, while the global (Cartesian) coordinate system is more convenient for numerical computations. Both coordinate systems are commonly used in the literature. The different bases lead to different representations of surface differential operators, such as the surface gradient, the surface divergence, and the covariant derivative. In Chapter 2, we provided a comprehensive overview of these operators and demonstrated relations between them in the local and global coordinate system. These relations are essential for comparing partial differential equations formulated in the different coordinate systems. We also presented a level set approach to describe evolving surfaces as the zero level of a scalar function, which is crucial for using TraceFEM to discretize surface PDEs.

In Chapter 3, we closely examined the derivation of surface Navier–Stokes equations. As a foundation for our numerical methods, we summarized various derivations known from the literature and based on different physical principles, such as conservation laws of surface mass and momentum, volume conservation laws combined with thin film limit techniques, or an energetic variational approach. The derivations were performed using

either local or global coordinate systems, resulting in strongly different looking equations. We summarized the main ingredients and techniques used in these derivations, we compared the resulting equations, and highlighted their similarities and differences.

The derivations led to two systems: the tangential surface Navier-Stokes system (TSNS) and the full surface Navier-Stokes system (SNS). The TSNS is characterized by a known surface where only the tangential component of the velocity and the pressure are unknown; within the SNS framework both surface geometry and flow field (including its normal and tangential components) are unknowns. We showed that all compared derivations lead to an identical TSNS system, a result that confirms consistency across different approaches.

Our numerical approach is based on the Trace Finite Element Method (TraceFEM), introduced in Chapter 4. TraceFEM serves as a standard tool for discretizing surface PDEs using a background mesh that does not align with the surface itself. This method allows flexible handling of complex geometries without requiring remeshing when surfaces evolve over time. We presented fundamental concepts related to TraceFEM including definitions of finite element spaces, an isoparametric mapping to obtain higher order surface approximations, along with construction techniques for discrete geometric quantities. Due to arbitrary cuts between background mesh and surface, instabilities in the discretization and the linear discrete systems can arise. Therefore, we introduced stabilization techniques such as a volume normal derivative stabilization. We also discussed an extension of TraceFEM to handle time-dependent surfaces using an implicit extension procedure.

In Chapter 5, we adapted the TraceFEM framework specifically to discretizing tangential surface Navier-Stokes equations (TSNS). We presented a novel numerical method combining TraceFEM with implicit extensions alongside backward differentiation formulas (BDF) used to discretize the time derivative. Furthermore, we summarized main results of a full error and stability analysis, which shows optimal convergence rates in suitable energy norms.

As described in Chapter 4, for applying TraceFEM it is very natural to represent surfaces implicitly as the zero level set of scalar functions. In the case of the full surface Navier-Stokes equations (SNS), the evolution of this level set function, describing the evolution of the surface, is an unknown of the system. Thus, we need to solve a transport equation for the level set function. In Chapter 6, we formulated a discontinuous Galerkin (DG) method for the transport equation that is known from the literature. We explained the need for a suitable extension method for the level set function. Since the material velocity of the surface can only be approximated in a narrow band around the surface, we need to extend the level set function in each time step. We introduced a new ghost penalty technique that allows us to extend the level set function from a narrow band around the surface to a wider domain. We presented an error and stability analysis of the ghost penalty technique, and combined the extension method with the DG method to obtain an efficient narrow band level set method. To demonstrate the accuracy of our transport method, we performed several numerical experiments that validate the theoretical results and show optimal convergence rates.

In the final Chapter 7, we combined previously developed methods into one framework to discretize the full surface Navier-Stokes equations (SNS). The main ingredients of our method are a TraceFEM based discretization of the surface PDE in space, a BDF method

for the time derivative, a narrow band level set method for the evolution of the surface, a volume normal derivative based extension method for the velocity, and a ghost penalty based method to construct a smooth discrete normal approximation. These methods are combined to obtain a fully discrete numerical scheme for the SNS system. The main difficulties in the discretization arise from the highly nonlinear coupling between $\mathbf{u}$ and $\Gamma(t)$. The surface PDE posed on $\Gamma(t)$ defines the velocity $\mathbf{u}$, which is used in the level set transport equation that defines the evolution of the surface $\Gamma(t)$. We presented various numerical experiments to demonstrate the accuracy of our method. We observed that our method achieves convergence for different surface geometries and flow conditions, including geometries with a topological change. Moreover, we showcased the behavior of the solution of the SNS system when no outer force is present. As far as we know, this is the first time that a fully discrete numerical method for the full surface Navier-Stokes equations based on TraceFEM has been presented. The method is capable of handling complex geometries and evolving surfaces without requiring remeshing or reinitializing.

In summary, this work advances understanding of fluid dynamics on evolving surfaces by providing new theoretical insights and practical computational tools. Especially, our method to discretize the surface Navier–Stokes system, handling the strong nonlinearity and the coupling of the surface PDE with the level set transport equation, forms a new contribution in the field. The developed TraceFEM-based approach allows for flexible and efficient simulations of fluid flows on deformable surfaces, which is crucial for applications in various scientific and engineering domains.

## 8.2 Outlook

We briefly discuss a few topics that we consider to be of interest for further research in this field.

### Analysis

The authors of [ORZ22] prove the existence and uniqueness of a weak solution for the tangential surface Navier-Stokes equations (TSNS), confirming that the TSNS system is well-posed. There are, however, no such results for the full surface Navier-Stokes system (SNS) in the literature. The nonlinear interaction between the surface PDE and surface evolution presents major challenges in proving well-posedness. Future research could focus on developing a suitable formulation of the SNS system and proving existence and uniqueness of solutions. This would be a major step forward in understanding its mathematical properties.

For the TSNS system, we demonstrated optimal convergence rates through error analysis in natural energy norms in Chapter 5. However, our analysis lacks $L^2$-error bounds for the pressure. Experiments in Chapter 5 suggest that the pressure converges optimally in the $L^2$-norm, but a proof is missing. For the full surface Navier-Stokes system (SNS), an error analysis for our TraceFEM-based discretization remains an open problem. As far as we know, also for other discretization methods, such as surface finite element methods (SFEM), no error analysis exists for the full SNS system.

It is well known that for unfitted finite element methods such as the TraceFEM there is an issue with algebraic stability. For the Laplace-Beltrami equation, it is known that the spectral condition number of the resulting stiffness matrix is bounded by $\mathcal{O}(h^{-2})$ if the

volume normal derivative stabilization in the trace finite element formulation is used. We expect that similar results can be derived for the trace finite element methods for the surface Navier–Stokes system, but there is no such theoretical result available yet.

**Improvement of numerical methods**

The strong nonlinearity of the surface Navier-Stokes equations poses challenges for both analysis and numerical simulation. We used a simple approach by extrapolating velocity from previous time steps as input for the level set equation. Future research might develop more advanced techniques to handle nonlinearity more effectively, such as iterating between solving surface PDE and the level set equation.

Discretizing surface Navier-Stokes equations leads to large linear systems posing computational challenges. We used direct solvers from the Intel MKL to solve the linear systems. Due to high computational cost and memory needs, these solvers become impractical as the size of the linear systems grows. A solution of very large sparse systems, which is more efficient in terms of computational and memory effort, requires the application of iterative solvers and tailor-made preconditioners. Developing suitable preconditioners and iterative solvers specifically designed for these large systems could significantly enhance solver efficiency and increase the applicability of the method to larger and more complex problems.

A crucial ingredient of our algorithms is the numerical discretization of the surface normal vector. Depending on the formulation of the problem and on the chosen discretization scheme, the approximation of the normal vector plays an import role. For example, in the TSNS system, the normal vector approximation used for the penalization of the normal component has to have a higher order of accuracy than the one needed for the approximation of the normal vector used to define the surface differential operators. A comprehensive study of the influence of the normal vector approximation on the accuracy of the numerical solution (for the TSNS and for the SNS) is missing in the literature.

While we focused on TraceFEM to discretize the PDEs on the surfaces in this thesis, there are other tailor-made methods for surface PDEs, e.g., surface Finite Element Methods (SFEM). It would be interesting to conduct comparative studies between different methods. Understanding their relative strengths and weaknesses when applied to evolving surfaces will improve our understanding of the problem and can help to validate the different methods.

**Other applications**

Experimental validation is crucial to further extend applicability of our computational framework. Validating numerical results against physical experiments involving fluid flows on deformable surfaces can reveal model limitations guiding refinements in modeling and numerical techniques.

As a next step, one might explore the surface Navier-Stokes equations on dynamic surfaces within a two-phase flow context. Unfitted finite element techniques have been employed for the numerical simulation of two-phase flows involving Navier-Stokes equations in the bulk phases, while also considering the influence of a viscous surface fluid or the movement of a surfactant at the interface (see [GR11; RZ13]). The numerical modeling of comparable two-phase flow scenarios utilizing Lagrangian tracking approaches for

the surface, such as surface finite elements and/or Arbitrary Lagrangian–Eulerian finite element methods, has been examined in [BGN15]. Initially, it is necessary to develop a model that connects the surface Navier-Stokes problem with the Navier-Stokes flow in the surrounding medium.

Another extension of our model could be a *two-phase* surface Navier–Stokes system. One way to model these type of systems is to couple the surface Navier-Stokes equations with a Cahn–Hilliard equation on the surface, which describes the evolution of a phase field. This approach has been used in [Bac+23].

In [KV23], the authors present a method for solving the surface Navier-Stokes equations combined with a constraint for the enclosed volume. This method is based on a Lagrange multiplier approach, which enforces the volume constraint during the simulation. Our method does not include such a constraint, and it would be interesting to extend our framework to incorporate a volume constraint. This could be particularly relevant for applications where the volume of the fluid domain is critical, such as in some biological systems.

In conclusion, while this thesis has made a step forward in understanding and computing fluid flows on evolving surfaces through novel methodologies, numerous challenges lie ahead that will deepen our understanding of fluid dynamics.

# Appendix A

## Chapter 2: Basic Notations and Differential Operators

### Chain rule

We give a representation of the chain rule for vector-valued functions, which is known from the literature.

**Theorem A.1: Chain rule**

Let $U \subset \mathbb{R}^n, V \subset \mathbb{R}^m$ open, $F : U \to \mathbb{R}^m$ differentiable in $a \in U$ with $F(U) \subset V$ and $G : V \to \mathbb{R}^p$ differentiable in $b = F(a)$. Then, $H := G \circ F : U \to \mathbb{R}^p$ is a differentiable function in $a$ satisfying

$$\nabla(G \circ F)(a) = (\nabla G)(F(a))(\nabla F)(a),$$

i.e.,

$$\frac{\partial}{\partial x_j} H(a) \cdot \mathbf{e}_i = \sum_{k=1}^{m} \left[ \frac{\partial}{\partial y_k} G(F(a)) \cdot \mathbf{e}_i \right] \left[ \frac{\partial}{\partial x_j} F(a) \cdot \mathbf{e}_k \right]$$

holds for $1 \le i \le p,\ 1 \le j \le n$.

### Proof of equation (2.2.3)

We give a proof of equation (2.2.3). Let $\mathbf{A} = \hat{A}_{ij}(\mathbf{g}^i \otimes \mathbf{g}^j) = \hat{A}^{ij}(\mathbf{g}_i \otimes \mathbf{g}_j) = A_{ij}(\mathbf{e}_i \otimes \mathbf{e}_j)$ be a matrix. The trace of the matrix in Cartesian coordinates is defined as $\operatorname{tr}(\mathbf{A}) := A_{ii}$. A direct calculation gives

$$\begin{aligned}\operatorname{tr}(\mathbf{A}) &= A_{kk} = \mathbf{A}_{ij}(\mathbf{e}_i \cdot \mathbf{e}_k)(\mathbf{e}_j \cdot \mathbf{e}_k) = \mathbf{e}_k \cdot [A_{ij}(\mathbf{e}_i \otimes \mathbf{e}_j)\mathbf{e}_k] = \mathbf{e}_k \cdot (\mathbf{A}\mathbf{e}_k) \\ &= (\mathbf{g}^i \cdot \mathbf{e}_k)\mathbf{g}_i \cdot (\mathbf{A}(\mathbf{g}_j \cdot \mathbf{e}^k)\mathbf{g}^j) = (\mathbf{e}_k \cdot \mathbf{g}^i)(\mathbf{e}_k \cdot \mathbf{g}_j)\mathbf{g}_i \cdot (\mathbf{A}\mathbf{g}^j) = (\mathbf{g}^i \cdot \mathbf{g}_j)\mathbf{g}_i \cdot (\mathbf{A}\mathbf{g}^j) \\ &= \mathbf{g}_i \cdot (\mathbf{A}\mathbf{g}^i).\end{aligned}$$

Analogously, we get $\operatorname{tr}(\mathbf{A}) = \mathbf{g}^i \cdot \mathbf{A}\mathbf{g}_i$. In addition, it holds

$$\operatorname{tr}(\mathbf{A}) = \mathbf{g}^i \cdot \mathbf{A}\mathbf{g}_i = \mathbf{g}^i \cdot \hat{A}_{kl}(\mathbf{g}^k \otimes \mathbf{g}^l)\mathbf{g}_i = g^{ik}\hat{A}_{kl}\delta^l_i = \hat{A}^i_i,$$

which completes the proof of (2.2.3).

### Proof of Lemma 2.2.4

We give a proof of Lemma 2.2.4. We start with the representation of the projection in mixed components. Since it holds $\mathbf{P}\mathbf{g}_\beta = \mathbf{g}_\beta$ and $\mathbf{P}\mathbf{n} = \mathbf{P}\mathbf{g}_3 = 0$, it follows

$$P^i_\beta = \mathbf{g}^i \cdot (\mathbf{P}\mathbf{g}_\beta) = \mathbf{g}^i \cdot \mathbf{g}_\beta = \delta^i_\beta, \quad \text{and} \quad P^i_3 = \mathbf{g}^i \cdot (\mathbf{P}\mathbf{g}_3) = 0.$$

We prove the representation of the Weingarten mapping in local components. From $\mathbf{H}\mathbf{n} = 0$, it follows immediately that $\hat{H}^i_3 = \mathbf{g}^i \cdot \mathbf{H}\mathbf{g}_3 = 0$ and $\hat{H}^3_i = 0$. Using the relation $(\partial^\Gamma_\alpha \mathbf{n} \cdot \mathbf{g}_\beta)\mathbf{g}^\beta = (\partial^\Gamma_\alpha \mathbf{n} \cdot \mathbf{g}_\beta)\mathbf{g}^\beta + \partial^\Gamma_\alpha \mathbf{n} \cdot \mathbf{n}\mathbf{n} = \partial^\Gamma_\alpha \mathbf{n}$ yields

$$\mathbf{H} = \nabla_\Gamma \mathbf{n} = \mathbf{g}^\alpha \otimes \partial^\Gamma_\alpha \mathbf{n} = \mathbf{g}^\alpha \otimes [(\partial^\Gamma_\alpha \mathbf{n} \cdot \mathbf{g}_\beta)\mathbf{g}^\beta] = (\partial^\Gamma_\alpha \mathbf{n} \cdot \mathbf{g}_\beta)(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta)$$

Thus, it follows $\hat{H}_{\alpha\beta} = (\partial^\Gamma_\alpha \mathbf{n} \cdot \mathbf{g}_\beta)$. The identities from the lemma follow from Definition 2.2.5 as well as the equation $\partial^\Gamma_\alpha \mathbf{g}_\beta \cdot \mathbf{g}_3 = \partial^\Gamma_\alpha \left(\mathbf{g}_\beta \cdot \mathbf{g}_3\right) - \mathbf{g}_\beta \cdot \partial^\Gamma_\alpha \mathbf{g}_3 = -\mathbf{g}_\beta \cdot \partial^\Gamma_\alpha \mathbf{g}_3$. Finally, a straight–forward computation yields

$$\begin{aligned}\hat{H}^\beta_\alpha &= g^\beta \cdot (\mathbf{H}\mathbf{g}_\alpha) = [(\mathbf{g}^\beta \cdot \mathbf{g}^\sigma)\mathbf{g}_\sigma] \cdot (\mathbf{H}g_\alpha) = (\mathbf{g}^\beta \cdot \mathbf{g}^\sigma)(\mathbf{g}_\sigma \cdot \mathbf{H}g_\alpha) \\ &= g^{\beta\sigma}\hat{H}_{\sigma\alpha}.\end{aligned}$$

### Proof of equation (2.2.5)

We give a proof of the first equality in (2.2.5). The second equality can be derived in the same way. The product rule, (2.2.4), and the symmetry of the Christoffel symbols yield

$$\begin{aligned}\partial^\Gamma_\gamma \mathbf{T} &= \partial^\Gamma_\gamma \hat{T}_{\alpha\beta}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) + \hat{T}_{\alpha\beta}\left[(\partial^\Gamma_\gamma \mathbf{g}^\alpha \otimes \mathbf{g}^\beta) + (\mathbf{g}^\alpha \otimes \partial^\Gamma_\gamma \mathbf{g}^\beta)\right] \\ &= \partial^\Gamma_\gamma \hat{T}_{\alpha\beta}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) + \hat{T}_{\alpha\beta}\left[(-\Gamma^\alpha_{\gamma\mu}\mathbf{g}^\mu - \hat{H}^\alpha_\gamma \mathbf{g}^3) \otimes \mathbf{g}^\beta + \mathbf{g}^\alpha \otimes (-\Gamma^\beta_{\gamma\mu}\mathbf{g}^\mu - \hat{H}^\beta_\gamma \mathbf{g}^3)\right] \\ &= \left[\partial^\Gamma_\gamma \hat{T}_{\alpha\beta} - \hat{T}_{\mu\beta}\Gamma^\mu_{\gamma\alpha} - \hat{T}_{\alpha\mu}\Gamma^\mu_{\gamma\beta}\right](\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) - \hat{T}_{\alpha\beta}\hat{H}^\alpha_\gamma(\mathbf{g}^3 \otimes \mathbf{g}^\beta) - \hat{T}_{\alpha\beta}\hat{H}^\beta_\gamma(\mathbf{g}^\alpha \otimes \mathbf{g}^3) \\ &= \hat{T}_{\alpha\beta|\gamma}(\mathbf{g}^\alpha \otimes \mathbf{g}^\beta) - \hat{T}_{\alpha\beta}\hat{H}^\alpha_\gamma(\mathbf{g}^3 \otimes \mathbf{g}^\beta) - \hat{T}_{\alpha\beta}\hat{H}^\beta_\gamma(\mathbf{g}^\alpha \otimes \mathbf{g}^3).\end{aligned}$$

### Proof of Lemma 2.2.10

We give a proof of Lemma 2.2.10. The first equality follows using the definition of the divergence and Theorem 2.2.6. It holds

$$\begin{aligned}\widehat{\operatorname{div}}_\Gamma \mathbf{v} = \partial^\Gamma_\alpha \mathbf{v} \cdot \mathbf{g}^\alpha &= \left[\left(\hat{v}^\beta{}_{|\alpha} + \hat{H}^\beta_\alpha \hat{v}_3\right)\mathbf{g}_\beta + \left(\hat{v}^3{}_{|\alpha} - \hat{H}_{\alpha\beta}\hat{v}^\beta\right)\mathbf{g}_3\right] \cdot \mathbf{g}^\alpha \\ &= \left(\hat{v}^\beta{}_{|\alpha} + \hat{H}^\beta_\alpha \hat{v}_3\right)\mathbf{g}_\beta \cdot \mathbf{g}^\alpha \\ &= \hat{v}^\alpha{}_{|\alpha} + \hat{H}^\alpha_\alpha \hat{v}_3.\end{aligned}$$

We represent $\mathbf{T}$ in local coordinates as $\mathbf{T} = \hat{T}^{\alpha\beta}(\mathbf{g}_\alpha \otimes \mathbf{g}_\beta)$. From the definition of the divergence and Theorem 2.2.6, we get the following relation for the divergence of a tensor-valued function $\mathbf{T}$:

$$
\begin{aligned}
\widehat{\operatorname{div}}_\Gamma \mathbf{T} &= (\partial_\alpha^\Gamma \mathbf{T})^T \mathbf{g}^\alpha = \left[\hat{T}^{\gamma\beta}_{\ \ |\alpha}(\mathbf{g}_\gamma \otimes \mathbf{g}_\beta) - \hat{T}^{\gamma\beta}\hat{H}_{\alpha\gamma}(\mathbf{g}_3 \otimes \mathbf{g}_\beta) - \hat{T}^{\gamma\beta}\hat{H}_{\alpha\beta}(\mathbf{g}_\gamma \otimes \mathbf{g}_3)\right]^T \mathbf{g}^\alpha \\
&= \left[\hat{T}^{\gamma\beta}_{\ \ |\alpha}(\mathbf{g}_\beta \otimes \mathbf{g}_\gamma) - \hat{T}^{\gamma\beta}\hat{H}_{\alpha\gamma}(\mathbf{g}_\beta \otimes \mathbf{g}_3) - \hat{T}^{\gamma\beta}\hat{H}_{\alpha\beta}(\mathbf{g}_3 \otimes \mathbf{g}_\gamma)\right]\mathbf{g}^\alpha \\
&= \hat{T}^{\gamma\beta}_{\ \ |\alpha}\mathbf{g}_\beta(\underbrace{\mathbf{g}_\gamma \cdot \mathbf{g}^\alpha}_{\delta_\gamma^\alpha}) - \hat{T}^{\gamma\beta}\hat{H}_{\alpha\gamma}\mathbf{g}_\beta(\underbrace{\mathbf{g}_3 \cdot \mathbf{g}^\alpha}_{=0}) - \hat{T}^{\gamma\beta}\hat{H}_{\alpha\beta}\mathbf{g}_3(\underbrace{\mathbf{g}_\gamma \cdot \mathbf{g}^\alpha}_{=\delta_\gamma^\alpha}) \\
&= \hat{T}^{\alpha\beta}_{\ \ |\alpha}\mathbf{g}_\beta - \hat{T}^{\alpha\beta}\hat{H}_{\alpha\beta}\mathbf{g}_3,
\end{aligned}
$$

which proves the first equality. The second equality follows with the same arguments.

**Proof of Lemma 2.5.1**

We give a proof of the differential rules from Lemma 2.5.1.

Equations (i) to (iii), and (v) to (viii) are proven by direct calculations given by

$$
\begin{aligned}
&\text{(i)} && \nabla_\Gamma(p \cdot q)_i = P_{ij}\partial_j(p \cdot q) = P_{ij}\partial_j pq + P_{ij}\partial_j qp = (\nabla_\Gamma p)_i \cdot q + (\nabla_\Gamma q)_i \cdot p, \\
&\text{(ii)} && \nabla_\Gamma(q \cdot \mathbf{v})_{ij} = P_{ik} \cdot (\partial_l q \cdot v_k + q \cdot \partial_l v_k) \cdot P_{lj} = (\mathbf{P}\mathbf{v})_i(\nabla_\Gamma q)_j + q \cdot (\nabla_\Gamma \mathbf{v})_{ij} \\
&&& \qquad = (\mathbf{P}\mathbf{v}\nabla_\Gamma^T q + \nabla_\Gamma \mathbf{v} q)_{ij}, \\
&\text{(iii)} && \nabla_\Gamma \mathbf{v} = \nabla_\Gamma(\mathbf{v}_T + v_N\mathbf{n}) = \nabla_\Gamma \mathbf{v}_T + \nabla_\Gamma(v_N\mathbf{n}) \overset{\text{(ii)}}{=} \nabla_\Gamma \mathbf{v}_T + v_N\nabla_\Gamma \mathbf{n} + \mathbf{P}\mathbf{n}\nabla_\Gamma^T u_N \\
&&& \qquad = \nabla_\Gamma \mathbf{v}_T + \mathbf{H}v_N, \\
&\text{(v)} && \operatorname{div}_\Gamma(q\mathbf{v}) = P_{lk} \cdot \partial_k q \cdot v_l + P_{lk}\partial_k v_l \cdot q = (\nabla_\Gamma q)_l \cdot v_l + \operatorname{div}_\Gamma \mathbf{v} \cdot q = \nabla_\Gamma q \cdot \mathbf{v} + q \operatorname{div}_\Gamma \mathbf{v}, \\
&\text{(vi)} && \operatorname{div}_\Gamma(q\mathbf{T})_i = \operatorname{div}_\Gamma(q\mathbf{T}\mathbf{e}_i) \overset{\text{(v)}}{=} \nabla_\Gamma q \cdot (\mathbf{T}\mathbf{e}_i) + q \operatorname{div}_\Gamma(\mathbf{T}\mathbf{e}_i) \\
&&& \qquad = \left(\mathbf{T}^T\nabla_\Gamma q\right) \cdot \mathbf{e}_i + q \operatorname{div}_\Gamma(\mathbf{T})_i, \\
&\text{(vii)} && \nabla_\Gamma(v_i)_j = (\mathbf{P}\nabla v_i)_j = P_{jk}(\nabla v_i)_k = P_{jk}(\nabla \mathbf{v})_{ik} = (\nabla \mathbf{v}\mathbf{P})_{ij}.
\end{aligned}
$$

Equation (iv) is proven by using the splitting $\mathbf{w} = \mathbf{w}_T + w_N\mathbf{n}$. It holds

$$
\nabla_\Gamma(\mathbf{v} \cdot \mathbf{w})_i = \nabla_\Gamma(\mathbf{v} \cdot (\mathbf{w}_T + w_N\mathbf{n}))_i = \nabla_\Gamma(\mathbf{v} \cdot \mathbf{w}_T)_i + \nabla_\Gamma(v_N \cdot w_N)_i.
$$

For the first addend, we get

$$
\begin{aligned}
\nabla_\Gamma(\mathbf{v} \cdot \mathbf{w}_T)_i &= P_{ij}\partial_j\big(\mathbf{v} \cdot (\mathbf{P}\mathbf{w})\big) = P_{ij}\partial_j\big((\mathbf{P}\mathbf{v})_k(\mathbf{P}\mathbf{w})_k\big) \\
&= P_{ij}\partial_j(\mathbf{v}_T)_k \cdot (\mathbf{P}\mathbf{w})_k + P_{ij}(\mathbf{P}\mathbf{v})_k \cdot \partial_j(\mathbf{w}_T)_k \\
&= P_{ij}\partial_j(\mathbf{v}_T)_k P_{kl} w_l + P_{ij}\partial_j(\mathbf{w}_T)_k P_{kl} v_l \\
&= (\nabla_\Gamma \mathbf{v}_T)_{li} w_l + (\nabla_\Gamma \mathbf{w}_T)_{li} v_l = (\nabla_\Gamma^T \mathbf{v}_T \mathbf{w} + \nabla_\Gamma^T \mathbf{w}_T \mathbf{v})_i
\end{aligned}
$$

and for the second addend, we get $\nabla_\Gamma(v_N \cdot w_N) = w_N\nabla_\Gamma v_N + v_N\nabla_\Gamma w_N$ using equation (i). This completes the proof.

# Appendix B

## Chapter 5: Discretization of Tangential Surface Navier–Stokes Equations

For completeness, we proof technical auxiliary results and error estimates used in Chapter 5: Discretization of Tangential Surface Navier–Stokes Equations.

### B.1 Preliminaries

We give some auxiliary results that are used in the proofs below.

Differentiating the identities $\mathbf{v}^\ell(\mathbf{y}) = \mathbf{v}^\ell(\mathbf{p}^n(\mathbf{y}))$ with the lifted function $\mathbf{v}^\ell$ one obtains the useful transformation relation

$$\nabla \mathbf{v}^\ell(\mathbf{y}) = \nabla \mathbf{v}^\ell(\mathbf{p}^n(\mathbf{y}))(\mathbf{P}(\mathbf{y}) - d(\mathbf{y})\mathbf{H}(\mathbf{y})), \quad \mathbf{v} \in H^1(\Gamma_h^n)^3, \tag{B.1.1}$$

with $d$ the signed distance function for $\Gamma^n$.

We define $\mathbf{B} = \mathbf{B}(\mathbf{x}, t) = \mathbf{P}(\mathbf{I} - d\mathbf{H})\mathbf{P}_h$ for $\mathbf{x} \in \Gamma_h(t)$. Geometry approximation assumptions from Section 5.3.2 imply that $\mathbf{B}$ is invertible on the range of $\mathbf{P}$ (for $h$ small enough) and the following uniform in time and discretization parameters estimates hold

$$\begin{aligned} \|\mathbf{B}\|_{L^\infty(\Gamma_h)} \leq C, \quad \|\mathbf{P}_h\mathbf{B}^{-1}\mathbf{P}\|_{L^\infty(\Gamma_h)} \leq C, \\ \|\mathbf{P}_h\mathbf{B}^{-1}\mathbf{P} - \mathbf{P}_h\mathbf{P}\|_{L^\infty(\Gamma_h)} \leq Ch^{k_g+1}, \quad \|1 - |\det(\mathbf{B})|\,\|_{L^\infty(\Gamma_h)} \leq Ch^{k_g+1} \end{aligned} \tag{B.1.2}$$

cf. [HLL19].

### B.2 Proof of consistency error bounds

We give bounds for the consistency errors defined in (5.4.9). Combining these results proves Corollary 5.4.7. All terms from (5.4.9) except $I_3$ have been considered in consistency analyses of TraceFEM in the literature, [LOX18; JR20; Jan+21]. In the next lemma we collect results which are essentially known and then treat the term $I_3$ in Lemma B.2.

**Lemma B.1**

The following uniform estimates hold

$$|I_1| \lesssim (\Delta t + h^{k_g})\|\mathbf{v}_h\|_{\Gamma_h^n} \tag{B.2.1}$$

$$|I_2| \lesssim h^{k_g}\rho_{\mathbf{u}}{}^{\frac{1}{2}}\|\mathbf{n}_h \cdot \nabla \mathbf{v}_h\|_{\Omega_\delta^n} \tag{B.2.2}$$

$$|I_4| \lesssim h^{k_g}\left(\|\mathbf{v}_h\|_{H^1(\Gamma_h^n)} + h^{-1}\|\widetilde{\mathbf{n}}_h \cdot \mathbf{v}_h\|_{\Gamma_h^n}\right) \tag{B.2.3}$$

$$|I_6| \lesssim h^{k_g+1}\tau\|\widetilde{\mathbf{n}}_h \cdot \mathbf{v}_h\|_{\Gamma_h^n} \lesssim h^{k_g}\tau^{\frac{1}{2}}\|\widetilde{\mathbf{n}}_h \cdot \mathbf{v}_h\|_{\Gamma_h^n} \tag{B.2.4}$$

$$|I_7| \lesssim h^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n} \tag{B.2.5}$$

where $k_g + 1$ is the order of geometry recovery defined in (4.1.3) and (5.3.6).

**Proof**

Componentwise application of the arguments presented in the proof of [LOX18, Lemma 11], combined with the estimate $\|\mathbf{P}_h - \mathbf{P}\|_{L^\infty(\Gamma_h^n)} \lesssim h^{k_g}$, leads to the derivation of the bound (B.2.1). Moreover, in the same proof, the result (B.2.2) is established, using equation (5.3.9). Recall that $E_{\Gamma_h}(\mathbf{P}_h\mathbf{v}_h)$ represents $E_{\Gamma_h}(\mathbf{v}_h) - (\mathbf{n}_h \cdot \mathbf{v}_h)\mathbf{H}_h$, cf. Remark 5.3.4. Equation (B.2.3) can be proven by [JR20, Lemma 5.15 and equation (5.42)] combined with norm equivalences from [JR20, Lemma 5.14] and geometric approximation inequalities from Section 5.3.2. Similarly, inequality (B.2.4) can be derived using similar arguments, employing the second equation from (4.1.5), as outlined in the proof of [JR20, Lemma 5.18]. The bound (B.2.5) is obtained analogously. □

The nonlinear term requires a more careful handling, which is carried out below.

**Lemma B.2**

It holds

$$|I_3| \lesssim (\Delta t + h^{k_g})\|\mathbf{v}_h\|_{H^1(\Gamma_h^n)} \quad \text{and} \quad |I_5| \lesssim h^{k_g}\|\mathbf{v}_h\|_{H^1(\Gamma_h^n)}.$$

**Proof**

We begin with the bound for $I_5$. Using definition $\mathbf{H} = \mathbf{P}\nabla\mathbf{n}\mathbf{P}$, equality $\mathbf{P}\mathbf{u}^n = \mathbf{u}^n$ and the product rule given by $\nabla(\mathbf{w}\cdot\mathbf{n})^T = \mathbf{w}\cdot\nabla\mathbf{n} + \mathbf{n}\cdot\nabla\mathbf{w}$ we get

$$\begin{aligned}\int_{\Gamma^n} w_N^n \mathbf{v}_h^\ell \cdot \mathbf{H}\mathbf{u}^n \,\mathrm{d}s &= \int_{\Gamma^n} w_N^n \nabla(\mathbf{P}\mathbf{v}_h^\ell \cdot \mathbf{n})\cdot\mathbf{u}^n \,\mathrm{d}s - \int_{\Gamma^n} w_N^n \mathbf{n}\cdot\nabla(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n \,\mathrm{d}s \\ &= \int_{\Gamma^n} w_N^n \nabla_\Gamma(\mathbf{v}_h^\ell \cdot \mathbf{P}\mathbf{n})\cdot\mathbf{u}^n \,\mathrm{d}s - \int_{\Gamma^n} w_N^n \mathbf{n}\cdot\nabla(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n \,\mathrm{d}s = -\int_{\Gamma^n} w_N^n \mathbf{n}\cdot\nabla(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n \,\mathrm{d}s.\end{aligned} \tag{B.2.6}$$

Using the definition $\mathbf{H}_h = \nabla_{\Gamma_h}\overline{\mathbf{n}}_h = \mathbf{P}_h\nabla\overline{\mathbf{n}}_h^\ell\mathbf{P}_h$, the transformation formula (B.1.1) applied to $\overline{\mathbf{n}}_h$, $\|d\|_{L^\infty(\Gamma_h^n)} \lesssim h^{k_g+1}$, $\|\mathbf{P} - \mathbf{P}_h\|_{L^\infty(\Gamma_h^n)} \lesssim h^{k_g}$, the bound for the surface measure change in (B.1.2) and the smoothness of $\mathbf{u}$ we obtain

$$\left|\int_{\Gamma_h^n} w_N^n\left[\mathbf{v}_h\cdot(\mathbf{H}_h\mathbf{u}^n)\right]\mathrm{d}s_h - \int_{\Gamma^n} w_N^n\left[\mathbf{v}_h^\ell\cdot(\mathbf{P}\nabla\overline{\mathbf{n}}_h^\ell\mathbf{P}\mathbf{u}^n)\right]\mathrm{d}s\right| \lesssim h^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n}. \tag{B.2.7}$$

We apply partial integration as in (B.2.6), which yields

$$\int_{\Gamma^n} w_N^n \left[\mathbf{v}_h^\ell \cdot (\mathbf{P}\nabla \overline{\mathbf{n}}_h^\ell \mathbf{P}\mathbf{u}^n)\right] \mathrm{d}s \\ = \int_{\Gamma^n} w_N^n \nabla_\Gamma(\mathbf{v}_h^\ell \cdot \mathbf{P}\overline{\mathbf{n}}_h^\ell)\cdot \mathbf{u}^n \,\mathrm{d}s \;\; - \int_{\Gamma^n} w_N^n \left[\overline{\mathbf{n}}_h^\ell \cdot (\nabla(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n)\right] \mathrm{d}s.$$

For the first term on the right-hand side we apply partial integration and using the identity $\|\mathbf{P}\overline{\mathbf{n}}_h^\ell\|_{L^\infty(\Gamma^n)} = \|\mathbf{P}(\overline{\mathbf{n}}_h^\ell - \mathbf{n})\|_{L^\infty(\Gamma^n)} \lesssim h^{k_g}$, cf. (5.3.6), we get

$$\left|\int_{\Gamma^n} w_N^n \nabla_\Gamma(\mathbf{v}_h^\ell \cdot \mathbf{P}\overline{\mathbf{n}}_h^\ell)\cdot \mathbf{u}^n \,\mathrm{d}s\right| = \left|\int_{\Gamma^n} (\mathbf{v}_h^\ell \cdot \mathbf{P}\overline{\mathbf{n}}_h^\ell)\,\mathrm{div}_\Gamma(w_N^n \mathbf{u}^n)\,\mathrm{d}s\right| \lesssim h^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n}. \tag{B.2.8}$$

With the results derived above we get, using again (5.3.6),

$$\begin{aligned} |I_5| &\le \left|\int_{\Gamma^n} w_N^n \mathbf{n}\cdot\nabla(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n \,\mathrm{d}s - \int_{\Gamma^n} w_N^n \overline{\mathbf{n}}_h^\ell \cdot \nabla(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n \,\mathrm{d}s\right| + h^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n} \\ &= \left|\int_{\Gamma^n} w_N^n (\mathbf{n}-\overline{\mathbf{n}}_h^\ell)\cdot\nabla(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n \,\mathrm{d}s\right| + h^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n} \lesssim h^{k_g}\|\mathbf{v}_h\|_{H^1(\Gamma_h^n)}, \end{aligned}$$

which completes the proof for $I_5$.

Now, we derive the bound for $I_3$. Recall that the spatial-normal extension of $\mathbf{u}$ to $\mathcal{O}(\mathcal{S})$ is denoted by $\mathbf{u}$, too. We start with estimating the differences between the quadratic nonlinear terms, i.e., the third and fourth terms in $I_3$, and the corresponding linearized ones on the *exact* surface at time $t_n$. For the fourth term in $I_3$ we use the smoothness of $\mathbf{u}$ and get

$$\begin{aligned} \left|\int_{\Gamma^n} \left[\nabla_\Gamma(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^n\right]\cdot\mathbf{u}^n - \left[\nabla_\Gamma(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^{n-1}\right]\cdot\mathbf{u}^n \,\mathrm{d}s\right| &= \left|\int_{\Gamma^n}\left[\nabla_\Gamma(\mathbf{P}\mathbf{v}_h^\ell)\int_{t_{n-1}}^{t_n}\mathbf{u}_t\,\mathrm{d}t\right]\cdot\mathbf{u}^n\,\mathrm{d}s\right| \\ &\le \Delta t\,\|\nabla_\Gamma \mathbf{P}\mathbf{v}_h^\ell\|_{L^2(\Gamma^n)} \lesssim \Delta t\|\mathbf{v}_h^\ell\|_{H^1(\Gamma^n)} \lesssim \Delta t\|\mathbf{v}_h\|_{H^1(\Gamma_h^n)}. \end{aligned}$$

With very similar arguments applied on the third term in $I_3$, we obtain

$$\left|\int_{\Gamma^n}\left[\nabla_\Gamma \mathbf{u}^n\mathbf{u}^n\right]\cdot\mathbf{v}_h^\ell\,\mathrm{d}s - \int_{\Gamma^n}\left[\nabla_\Gamma\mathbf{u}^n\mathbf{u}^{n-1}\right]\cdot\mathbf{v}_h^\ell\,\mathrm{d}s\right| \lesssim \Delta t\|\mathbf{v}_h\|_{\Gamma_h^n}.$$

For the approximate surface $\nabla_{\Gamma_h}\mathbf{P}_h\mathbf{w}$ represents $\nabla_{\Gamma_h}\mathbf{w} - (\mathbf{n}_h\cdot\mathbf{w})\mathbf{H}_h$. We now compare the linearized terms on the *approximate* surface $\Gamma_h^n$ (first and second one in $I_3$) with the corresponding ones on the exact surface. For the second term in $I_3$ we get

$$\begin{aligned} &\left|\int_{\Gamma_h^n}\left[\nabla_{\Gamma_h}(\mathbf{P}_h\mathbf{v}_h)\mathbf{u}^{n-1}\right]\cdot\mathbf{u}^n\,\mathrm{d}s_h - \int_{\Gamma^n}\left[\nabla_\Gamma(\mathbf{P}\mathbf{v}_h^\ell)\mathbf{u}^{n-1}\right]\cdot\mathbf{u}^n\,\mathrm{d}s\right| \\ \le &\underbrace{\left|\int_{\Gamma_h^n}\left[\nabla_{\Gamma_h}\mathbf{v}_h\mathbf{u}^{n-1}\right]\cdot\mathbf{u}^n\,\mathrm{d}s_h - \int_{\Gamma^n}\left[\nabla_\Gamma\mathbf{v}_h^\ell\mathbf{u}^{n-1}\right]\cdot\mathbf{u}^n\,\mathrm{d}s\right|}_{=:\,J_1} \\ &+ \underbrace{\left|\int_{\Gamma_h^n}(\mathbf{v}_h\cdot\mathbf{n}_h)\left[\mathbf{u}^{n-1}\cdot(\mathbf{H}_h\mathbf{u}^n)\right]\mathrm{d}s_h - \int_{\Gamma^n}(\mathbf{v}_h^\ell\cdot\mathbf{n})\left[\mathbf{u}^{n-1}\cdot(\mathbf{H}\mathbf{u}^n)\right]\mathrm{d}s\right|}_{=:\,J_2} \end{aligned}$$

For the term $J_1$ we use $\nabla_{\Gamma_h}\mathbf{v}_h = \mathbf{P}_h\nabla\mathbf{v}_h^\ell\mathbf{P}_h$, the transformation relation (B.1.1) applied

to $\mathbf{v}_h$, the relation $\|\mathbf{P}-\mathbf{P}_h\|_{L^\infty(\Gamma_h^n)} \lesssim h^{k_g}$ and the bound in (B.1.2) for the change in surface measure. Thus, we get

$$J_1 \lesssim h^{k_g}\|\mathbf{v}_h^\ell\|_{H^1(\Gamma^n)} \lesssim h^{k_g}\|\mathbf{v}_h\|_{H^1(\Gamma_h^n)}.$$

For the term $J_2$ we proceed as in the derivation of the bound for $I_5$. As in (B.2.6) we obtain

$$\int_{\Gamma^n}(\mathbf{n}\cdot\mathbf{v}_h^\ell)\left[\mathbf{u}^{n-1}\cdot(\mathbf{H}\mathbf{u}^n)\right]\mathrm{d}s = -\int_{\Gamma^n}(\mathbf{n}\cdot\mathbf{v}_h^\ell)\left[\mathbf{n}\cdot(\nabla(\mathbf{P}\mathbf{u}^{n-1})\mathbf{u}^n)\right]\mathrm{d}s. \tag{B.2.9}$$

In the other term in $J_2$ we replace $\mathbf{n}_h$ by $\mathbf{n}$, with error bounded by $Ch^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n}$. With the same arguments as in (B.2.7)–(B.2.8) the resulting term $\int_{\Gamma_h^n}(\mathbf{n}\cdot\mathbf{v}_h)\mathbf{u}^{n-1}\cdot\mathbf{H}_h\mathbf{u}^n\,\mathrm{d}s_h$ can be replaced by $-\int_{\Gamma^n}(\mathbf{n}\cdot\mathbf{v}_h^\ell)\overline{\mathbf{n}}_h^\ell\cdot\nabla(\mathbf{P}\mathbf{u}^{n-1})\mathbf{u}^n\,\mathrm{d}s$ with error bounded by $Ch^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n}$. Comparing the latter term with the one on the right-hand side in (B.2.9) we get, using the assumption (5.3.6), a bound $Ch^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n}$. Summarizing we get

$$J_1 + J_2 \lesssim h^{k_g}\|\mathbf{v}_h\|_{H^1(\Gamma_h^n)}.$$

For the remaining first term in $I_3$ we use similar arguments. First note

$$\begin{aligned}\left|\int_{\Gamma_h^n}\left[\nabla_{\Gamma_h}(\mathbf{P}_h\mathbf{u}^n)\mathbf{u}^{n-1}\right]\cdot\mathbf{v}_h\,\mathrm{d}s_h - \int_{\Gamma^n}\left[\nabla_\Gamma\mathbf{u}^n\mathbf{u}^{n-1}\right]\cdot\mathbf{v}_h^\ell\,\mathrm{d}s\right| \\ \leq \left|\int_{\Gamma_h^n}\left[\nabla_{\Gamma_h}\mathbf{u}^n\mathbf{u}^{n-1}\right]\cdot\mathbf{v}_h\,\mathrm{d}s_h - \int_{\Gamma^n}\left[\nabla_\Gamma\mathbf{u}^n\mathbf{u}^{n-1}\right]\cdot\mathbf{v}_h^\ell\,\mathrm{d}s\right| \\ + \left|\int_{\Gamma_h^n}(\mathbf{n}_h\cdot\mathbf{u}^n)\left[\mathbf{u}^{n-1}\cdot(\mathbf{H}_h\mathbf{v}_h)\right]\mathrm{d}s_h\right|.\end{aligned}$$

The first term of the right-hand side can be bounded by $Ch^{k_g}\|\mathbf{v}_h\|_{\Gamma_h^n}$ using the same arguments as for $J_1$ above. For the second term we also obtain such a bound using $\mathbf{n}_h\cdot\mathbf{u}^n = (\mathbf{n}_h-\mathbf{n})\cdot\mathbf{u}^n$ and $\|\mathbf{n}_h-\mathbf{n}\|_{L^\infty(\Gamma_h^n)} \lesssim h^{k_g}$. Combining these estimates we obtain the desired result. □

# Bibliography


[A96] Sethian J. A. "A fast marching level set method for monotonically advancing fronts." In: *Proceedings of the National Academy of Sciences of the United States of America* 93.4 (1996), pp. 1591–1595. DOI: 10.1073/pnas.93.4.1591 (cit. on pp. 107, 109).

[AD00] Luigi Ambrosio and Norman Dancer. *Calculus of Variations and Partial Differential Equations*. Ed. by Giuseppe Buttazzo, Antonio Marino, and M. K. V. Murthy. Springer Berlin Heidelberg, 2000. DOI: 10.1007/978-3-642-57186-2 (cit. on p. 32).

[AD09] Marino Arroyo and Antonio DeSimone. "Relaxation dynamics of fluid membranes." In: *Physical Review E* 79.3 (2009), p. 031915 (cit. on pp. 1, 3, 35, 56).

[Ago98] Valeri Agoshkov. *Boundary Value Problems for Transport Equations*. Birkhäuser Boston, 1998. DOI: 10.1007/978-1-4612-1994-1 (cit. on p. 33).

[AS14] M. Arienti and Mark Sussman. "An embedded level set method for sharp-interface multiphase simulations of diesel injectors." In: *Journal of Multiphase Flow* 59 (2014), pp. 1–14 (cit. on p. 108).

[AS99] David Adalsteinsson and James A. Sethian. "The fast construction of extension velocities in level set methods." In: *Journal of Computational Physics* 148 (1999), pp. 2–22 (cit. on pp. 109, 143).

[Bac+23] Elena Bachini et al. "Derivation and simulation of a two-phase fluid deformable surface model." In: *Journal of Fluid Mechanics* 977 (Dec. 2023). DOI: 10.1017/jfm.2023.943. URL: http://dx.doi.org/10.1017/jfm.2023.943 (cit. on p. 179).

[BDF16] Eva Bezchlebová, Vít Dolejší, and Miloslav Feistauer. "Discontinuous Galerkin method for the solution of a transport level-set problem." In: *Computers & Mathematics with Applications* 72.3 (2016), pp. 455–480. DOI: 10.1016/j.camwa.2016.04.033. URL: https://www.sciencedirect.com/science/article/pii/S0898122116302267 (cit. on pp. 7, 108, 113–116).

[BDL20] Andrea Bonito, Alan Demlow, and Martin Licht. "A Divergence-Conforming Finite Element Method for the Surface Stokes Equation." In: *SIAM Journal on Numerical Analysis* 58.5 (2020), pp. 2764–2798. DOI: 10.1137/19M1284592 (cit. on pp. 2, 21, 78).

[BDN20] Andrea Bonito, Alan Demlow, and Ricardo H. Nochetto. "Finite element methods for the Laplace–Beltrami operator." In: *Geometric Partial Differential Equations - Part I*. Elsevier, 2020, pp. 1–103. DOI: 10.1016/bs.hna.2019.06.002 (cit. on p. 2).

[Ber+01] Marcelo Bertalmio et al. "Variational Problems and Partial Differential Equations on Implicit Surfaces." In: *Journal of Computational Physics* 174.2 (2001), pp. 759–780. DOI: 10.1006/jcph.2001.6937. URL: https://www.sciencedirect.com/science/article/pii/S0021999101969372 (cit. on p. 59).

[BFM22] Erik Burman, Stefan Frei, and Andre Massing. "Eulerian time-stepping schemes for the non-stationary Stokes equations on time-dependent domains." In: *Numerische Mathematik* (2022), pp. 1–56 (cit. on pp. 6, 77, 94).

[BGN15] John W. Barrett, Harald Garcke, and Robert Nürnberg. "Stable finite element approximations of two-phase flow with soluble surfactant." In: *Journal of Computational Physics* 297 (Sept. 2015), pp. 530–564. DOI: 10.1016/j.jcp.2015.05.029. URL: http://dx.doi.org/10.1016/j.jcp.2015.05.029 (cit. on p. 179).

[BH82] Alexander N. Brooks and Thomas J.R. Hughes. "Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations." In: *Computer Methods in Applied Mechanics and Engineering* 32.1 (1982), pp. 199–259. DOI: 10.1016/0045-7825(82)90071-8. URL: https://www.sciencedirect.com/science/article/pii/0045782582900718 (cit. on p. 108).

[BHL15] Erik Burman, Peter Hansbo, and Mats Larson. "A stabilized cut finite element method for partial differential equations on surfaces: The Laplace-Beltrami operator." In: *Computer Methods in Applied Mechanics and Engineering* 285 (Mar. 2015), pp. 188–207. DOI: 10.1016/j.cma.2014.10.044 (cit. on pp. 68, 117).

[BK13] Christopher Basting and Dmitri Kuzmin. "A minimization-based finite element formulation for interface-preserving level set reinitialization." In: *Computing* 95 (Suppl 1).5 (2013), pp. 13–25. DOI: 10.1007/s00607-012-0259-z (cit. on p. 109).

[BMS04] F. Brezzi, L. D. Marini, and E. Süli. "Discontinuous Galerkin methods for first-oder hyperbolic problems." In: *Mathematical Models and Methods in Applied Sciences* 14.12 (2004), pp. 1893–1903 (cit. on pp. 7, 108).

[Bot92] Dieter Bothe. "Multivalued differential equations on graphs." In: *Nonlinear Analysis: Theory, Methods & Applications* 18.3 (1992), pp. 245–252. DOI: 10.1016/0362-546X(92)90062-J. URL: https://www.sciencedirect.com/science/article/pii/0362546X9290062J (cit. on p. 12).

[BPS05] Dieter Bothe, Jan Prüss, and Gieri Simonett. "Well-posedness of a Two-phase Flow with Soluble Surfactant." In: *Nonlinear Elliptic and Parabolic Problems: A Special Tribute to the Work of Herbert Amann*. Ed. by Haim Brezis, Michel Chipot, and Joachim Escher. Basel: Birkhäuser Basel, 2005, pp. 37–61. DOI: 10.1007/3-7643-7385-7_3 (cit. on p. 12).

[BQS10] Erik Burman, Alfio Quarteroni, and Benjamin Stamm. "Interior penalty continuous and discontinuous finite element approximations of hyperbolic equations." In: *Journal of Scientific Computing* 43 (2010), pp. 293–312 (cit. on pp. 7, 108).

[BR20] Philip Brandner and Arnold Reusken. "Finite element error analysis of surface Stokes equations in stream function formulation." In: *ESAIM: M2AN* 54.6 (2020), pp. 2069–2097. DOI: 10.1051/m2an/2020044 (cit. on pp. 2, 60, 125, 133, 151, 155).

[Bra+22] Philip Brandner et al. "Finite element discretization methods for velocity-pressure and stream function formulations of surface Stokes equations." In: *SIAM Journal on Scientific Computing* 44.4 (2022), A1807–A1832 (cit. on pp. 60, 78, 146).

[BRS22] Philip Brandner, Arnold Reusken, and Paul Schwering. "On derivations of evolving surface Navier–Stokes equations." In: *Interfaces Free Bound.* 24 (Oct. 2022), pp. 533–563. DOI: 10.4171/IFB/483. URL: https://ems.press/journals/ifb/articles/7790227 (cit. on pp. 8, 11, 19, 35, 40).

[BS25] Dieter Bothe and Kohei Soga. "Mathematical Analysis of the Velocity Extension Level Set Method." In: (2025). DOI: 10.2139/ssrn.5232128. URL: http://dx.doi.org/10.2139/ssrn.5232128 (cit. on p. 143).

[Bur+14] Erik Burman et al. "CutFEM: Discretizing geometry and partial differential equations." In: *International Journal for Numerical Methods in Engineering* 104.7 (2014), pp. 472–501. DOI: doi:10.1002/nme.4823 (cit. on pp. 60, 68, 109, 117).

[Bur+16] Erik Burman et al. "Full gradient stabilized cut finite element methods for surface partial differential equations." In: *Computer Methods in Applied Mechanics and Engineering* 310 (Oct. 2016), pp. 278–296. DOI: 10.1016/j.cma.2016.06.033 (cit. on p. 70).

[Bur+18] Erik Burman et al. "Cut Finite Element Methods for Partial Differential Equations on Embedded Manifolds of Arbitrary Codimensions." In: *ESAIM: Mathematical Modelling and Numerical Analysis* 52 (June 2018). DOI: 10.1051/m2an/2018038 (cit. on pp. 69, 153).

[Bur10] Erik Burman. "Ghost penalty." In: *Comptes rendus mathematique* 348.21-22 (Oct. 2010), pp. 1217–1220. DOI: doi:10.1016/j.crma.2010.10.006 (cit. on pp. 67, 68, 109, 117).

[CCD17] Chi-Hin Chan, Magdalena Czubak, and Marcelo M. Disconzi. "The formulation of the Navier–Stokes equations on Riemannian manifolds." In: *Journal of Geometry and Physics* 121 (Nov. 2017), pp. 335–346. DOI: 10.1016/j.geomphys.2017.07.015 (cit. on p. 56).

[CHS90] Bernardo Cockburn, Suchung Hou, and Chi-Wang Shu. "The Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws. IV: The Multidimensional Case." In: *Mathematics of Computation* 54.190 (1990), pp. 545–581. URL: http://www.jstor.org/stable/2008501 (cit. on pp. 108, 116).

[Cia13] Philippe G. Ciarlet. *Linear and Nonlinear Functional Analysis with Applications*. USA: Society for Industrial and Applied Mathematics, 2013, pp. XIV + 832 (cit. on pp. 14–18, 47).

[CM79] H. Cohen and A. I. Murdoch. "Symmetry considerations for material surfaces." In: *Archive for Rational Mechanics and Analysis* 72 (Mar. 1979). DOI: 10.1007/BF00250737 (cit. on p. 11).

[CS89] Bernardo Cockburn and Chi-Wang Shu. "TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws II: General Framework." In: *Mathematics of Computation* 52.186 (1989), pp. 411–435. URL: http://www.jstor.org/stable/2008474 (cit. on pp. 108, 116).

[DD07] Alan Demlow and Gerhard Dziuk. "An Adaptive Finite Element Method for the Laplace–Beltrami Operator on Implicitly Defined Surfaces." In: *SIAM Journal on Numerical Analysis* 45.1 (2007), pp. 421–442. DOI: 10.1137/050642873 (cit. on p. 2).

[DE08] Gerhard Dziuk and Charles. M. Elliott. "An Eulerian approach to transport and diffusion on evolving implicit surfaces." In: *Computing and Visualization in Science* 13.1 (July 2008), pp. 17–28. DOI: 10.1007/s00791-008-0122-0 (cit. on p. 60).

[DE13] Gerhard Dziuk and Charles M. Elliott. "Finite element methods for surface PDEs." In: *Acta Numerica* 22 (2013), pp. 289–396. DOI: 10.1017/S0962492913000056 (cit. on pp. 2, 12, 20, 22, 28, 31, 59, 61).

[Dec+09] Klaus Deckelnick et al. "An h-narrow band finite-element method for elliptic equations on implicit surfaces." In: *IMA Journal of Numerical Analysis* 30.2 (Mar. 2009), pp. 351–376. DOI: 10.1093/imanum/drn049 (cit. on p. 60).

[Dem09] Alan Demlow. "Higher-Order Finite Element Methods and Pointwise Error Estimates for Elliptic Problems on Surfaces." In: *SIAM Journal on Numerical Analysis* 47.2 (2009), pp. 805–827. DOI: 10.1137/070708135 (cit. on p. 59).

[DER14] Klaus Deckelnick, Charles M. Elliott, and Thomas Ranner. "Unfitted Finite Element Methods Using Bulk Meshes for Surface Partial Differential Equations." In: *SIAM Journal on Numerical Analysis* 52.4 (Jan. 2014), pp. 2137–2162. DOI: 10.1137/130948641 (cit. on p. 69).

[DF04] Vít Dolejší and M. Feistauer. "A semi-implicit discontinuous Galerkin finite element method for the numerical solution of inviscid compressible flow." In: *Journal of Computational Physics* 198.2 (2004), pp. 727–746. DOI: 10.1016/j.jcp.2004.01.023. URL: https://www.sciencedirect.com/science/article/pii/S0021999104000609 (cit. on p. 108).

[DF15] Vít Dolejší and Miloslav Feistauer. *Discontinuous Galerkin Method: Analysis and Applications to Compressible Flow*. Springer International Publishing, 2015. DOI: 10.1007/978-3-319-19267-3 (cit. on pp. 7, 108).

[Dim+06] Rumiana Dimova et al. "A practical guide to giant vesicles. Probing the membrane nanoregime via optical microscopy." In: *Journal of Physics: Condensed Matter* 18.28 (2006), S1151 (cit. on p. 1).

[DLP06] Daniele A. Di Pietro, Stefania Lo Forte, and Nicola Parolini. "Mass preserving finite element implementations of the level set method." In: *Applied Numerical Mathematics* 56.9 (2006). Numerical Methods for Viscosity Solutions and Applications, pp. 1179–1195. DOI: 10.1016/j.apnum.2006.03.003. URL: https://www.sciencedirect.com/science/article/pii/S0168927406000432 (cit. on pp. 115, 138).

[Dzi88] Gerhard Dziuk. "Finite Elements for the Beltrami operator on arbitrary surfaces." In: *Partial differential equations and calculus of variations*. Ed. by S. Hildebrandt and R. Leis. Vol. 1357. Lecture Notes in Mathematics. Springer, Berlin, Heidelberg, 1988, pp. 142–155 (cit. on pp. 59, 70, 105, 125).

[EGK22] Charles M. Elliott, Harald Garcke, and Balázs Kovács. "Numerical analysis for the interaction of mean curvature flow and diffusion on closed surfaces." In: *Numerische Mathematik* 151 (4 Aug. 2022), pp. 873–925. DOI: 10.1007/s00211-022-01301-3 (cit. on p. 130).

[EHZ07] Weinan E, Dan Hu, and Pingwen Zhang. "Continuum theory of a moving membrane." In: *Physical Review E* 75 (4 Apr. 2007), p. 041605. DOI: 10.1103/PhysRevE.75.041605. URL: https://link.aps.org/doi/10.1103/PhysRevE.75.041605 (cit. on pp. 1, 3, 18, 23, 27, 28, 35–41, 52, 56).

[EM70] David G Ebin and Jerrold Marsden. "Groups of diffeomorphisms and the motion of an incompressible fluid." In: *Annals of Mathematics* 92.1 (July 1970), pp. 102–163. DOI: 10.2307/1970699 (cit. on pp. 1, 35, 56).

[EV15] Charles M. Elliott and Chandrasekhar Venkataraman. "Error analysis for an ALE evolving surface finite element method." In: *Numerical Methods for Partial Differential Equations* 31.2 (Mar. 2015), pp. 459–499. DOI: 10.1002/num.21930 (cit. on p. 59).

[Fri18] Thomas-Peter Fries. "Higher-order surface FEM for incompressible Navier–Stokes flows on manifolds." In: *International Journal for Numerical Methods in Fluids* 88.2 (2018), pp. 55–78. DOI: 10.1002/fld.4510 (cit. on pp. 2, 21, 59, 78).

[FŠ04] M. Feistauer and K. Švadlenka. "Discontinuous Galerkin method of lines for solving nonstationary singularly perturbed linear problems." In: *Journal of Numerical Mathematics* 12.2 (2004), pp. 97–117. DOI: 10.1515/156939504323074504 (cit. on p. 108).

[GE17] J.-L. Guermond and A. Ern. "Finite element quasi-interpolation and best approximation." In: *ESAIM: Mathematical Modelling and Numerical Analysis* 51 (2017), pp. 1367–1385 (cit. on p. 115).

[GKL17] Yoshikazu Giga, Hajime Koba, and Chun Liu. "Energetic Variational approaches for incompressible fluid systems on an evolving surface." In: *Quarterly of Applied mathematics* 75 (June 2017), pp. 359–389. DOI: 10.1090/qam/1452 (cit. on pp. 1, 3, 21, 25, 35–37, 41–43, 55).

[GLR18] Jörg Grande, Christoph Lehrenfeld, and Arnold Reusken. "Analysis of a High-Order Trace Finite Element Method for PDEs on Level Set Surfaces." In: *SIAM Journal on Numerical Analysis* 56.1 (2018), pp. 228–255. DOI: 10.1137/16M1102203 (cit. on pp. 2, 67–70).

[GM75] Morton E. Gurtin and A. Ian Murdoch. "A continuum theory of elastic material surfaces." In: *Archive for Rational Mechanics and Analysis* 57.4 (Dec. 1975), pp. 291–323. DOI: 10.1007/bf00261375 (cit. on p. 11).

[GR11] Sven Gross and Arnold Reusken. *Numerical Methods for Two-phase Incompressible Flows*. Springer Berlin Heidelberg, 2011. DOI: 10.1007/978-3-642-19686-7 (cit. on pp. 27, 28, 67, 108, 117, 163, 178).

[Gre05] John B. Greer. "An Improvement of a Recent Eulerian Method for Solving PDEs on General Geometries." In: *Journal of Scientific Computing* 29.3 (Dec. 2005), pp. 321–352. DOI: 10.1007/s10915-005-9012-5 (cit. on p. 60).

[Gro+18] Sven Gross et al. "A trace finite element method for vector-Laplacians on surfaces." In: *SIAM Journal on numerical analysis* 56.4 (2018), pp. 2406–2429 (cit. on p. 78).

[HLL19] Peter Hansbo, Mats G Larson, and Karl Larsson. "Analysis of finite element methods for vector Laplacians on surfaces." In: *IMA Journal of Numerical Analysis* 40.3 (Apr. 2019), pp. 1652–1701. DOI: 10.1093/imanum/drz018 (cit. on pp. 78, 85, 185).

[HLP23] Fabian Heimann, Christoph Lehrenfeld, and Janosch Preuß. "Geometrically Higher Order Unfitted Space-Time Methods for PDEs on Moving Domains." In: *SIAM Journal on Scientific Computing* 45.2 (Mar. 2023), B139–B165. DOI: 10.1137/22m1476034 (cit. on p. 61).

[HW96] Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II*. Springer Berlin Heidelberg, 1996. DOI: 10.1007/978-3-642-05221-7 (cit. on p. 71).

[Jan+21] Thomas Jankuhn et al. "Error analysis of higher order trace finite element methods for the surface Stokes equation." In: *Journal of Numerical Mathematics* 29.3 (2021), pp. 245–267 (cit. on pp. 78, 147, 185).

[JNP84] Claes Johnson, Uno Nävert, and Juhani Pitkäranta. "Finite element methods for linear hyperbolic problems." In: *Computer Methods in Applied Mechanics and Engineering* 45.1 (1984), pp. 285–312. DOI: 10.1016/0045-7825(84)90158-0. URL: https://www.sciencedirect.com/science/article/pii/0045782584901580 (cit. on p. 108).

[JOR18] Thomas Jankuhn, Maxim Olshanskii, and Arnold Reusken. "Incompressible fluid problems on embedded surfaces: Modeling and variational formulations." In: *Interfaces and Free Boundaries* 20 (Oct. 2018), pp. 353–377. DOI: 10.4171/IFB/405 (cit. on pp. 1, 3, 20, 21, 25, 27, 28, 35–37, 40, 41, 52, 54–56, 60).

[JR20] Thomas Jankuhn and Arnold Reusken. "Trace finite element methods for surface vector-Laplace equations." In: *IMA Journal of Numerical Analysis* 41.1 (May 2020), pp. 48–83. DOI: 10.1093/imanum/drz062 (cit. on pp. 2, 78, 85, 88, 91, 185, 186).

[Keb+14] Felix C Keber et al. "Topology and dynamics of active nematic vesicles." In: *Science* 345.6201 (2014), pp. 1135–1139 (cit. on p. 1).

[KKV23] Veit Krause, Eric Kunze, and Axel Voigt. "A surface finite element method for the Navier–Stokes equations on evolving surfaces." In: *PAMM* 23.3 (Sept. 2023). DOI: 10.1002/pamm.202300014 (cit. on pp. 2, 59, 78).

[KMS99] G. Kossioris, Ch. Makridakis, and P.E. Souganidis. "Finite volume schemes for Hamilton-Jacobi equations." In: *Numerische Mathematik* 83 (1999), pp. 427–442. DOI: 10.1007/s002110050457 (cit. on p. 107).

[KV23] Veit Krause and Axel Voigt. "A numerical approach for fluid deformable surfaces with conserved enclosed volume." In: *Journal of Computational Physics* 486 (2023), p. 112097. DOI: 10.1016/j.jcp.2023.112097. URL: https://www.sciencedirect.com/science/article/pii/S0021999123001924 (cit. on pp. 2, 166, 179).

[LDM14] Curtis Lee, John Dolbow, and Peter J Mucha. "A narrow-band gradient-augmented level set method for multiphase incompressible flow." In: *Journal of Computational Physics* 273 (2014), pp. 12–37 (cit. on p. 108).

[Leh+] Christoph Lehrenfeld et al. *ngsxfem: Add-on to NGSolve for geometrically unfitted finite element discretizations*. Journal of Open Source Software, 6(64), 3237. DOI: 10.21105/joss.03237 (cit. on pp. 7, 65, 99, 131, 159).

[Leh16] Christoph Lehrenfeld. "High order unfitted finite element methods on level set domains using isoparametric mappings." In: *Computer Methods in Applied Mechanics and Engineering* 300 (2016), pp. 716–733 (cit. on pp. 6, 60, 62, 63).

[Leh17] Christoph Lehrenfeld. "A Higher Order Isoparametric Fictitious Domain Method for Level Set Domains." In: *Geometrically Unfitted Finite Element Methods and Applications*. Ed. by Stéphane P. A. Bordas et al. Springer International Publishing, 2017, pp. 65–92. DOI: 10.1007/978-3-319-71431-8 (cit. on p. 117).

[LL22] Christoph Lehrenfeld and Yimin Lou. "An Eulerian finite element method for PDEs in time-dependent domains." In: *SIAM Journal on Numerical Analysis* 60.4 (2022), pp. 2069–2098. DOI: 10.1137/21M142126X (cit. on pp. 61, 70, 75).

[LO19] Christoph Lehrenfeld and Maxim Olshanskii. "An Eulerian finite element method for PDEs in time-dependent domains." In: *ESAIM: M2AN* 53.2 (2019), pp. 585–614. DOI: 10.1051/m2an/2018068 (cit. on pp. 68, 109, 117–120).

[Loc13] Eva Loch. "The level set method for capturing interfaces with applications in two-phase flow problems." PhD thesis. Sept. 2013 (cit. on p. 33).

[Lou21] Yimin Lou. "High-order Unfitted Discretizations for Partial Differential Equations Coupled with Geometric Flow." dissertation. Georg-August-Universität Göttingen, 2021 (cit. on pp. 146, 155).

[LOX18] Christoph Lehrenfeld, Maxim A. Olshanskii, and Xianmin Xu. "A Stabilized Trace Finite Element Method for Partial Differential Equations on Evolving Surfaces." In: *SIAM Journal on Numerical Analysis* 56.3 (Jan. 2018), pp. 1643–1672. DOI: 10.1137/17m1148633 (cit. on pp. 6, 61, 70, 75, 92, 117, 131, 185, 186).

[LR17] Christoph Lehrenfeld and Arnold Reusken. "Analysis of a high-order unfitted finite element method for elliptic interface problems." In: *IMA Journal of Numerical Analysis* 38.3 (Aug. 2017), pp. 1351–1387. DOI: 10.1093/imanum/drx041 (cit. on p. 64).

[Map] Maplesoft. https://de.maplesoft.com/, (cit. on pp. 99, 159).

[Mas+14] André Massing et al. "A Stabilized Nitsche Fictitious Domain Method for the Stokes Problem." In: *Journal of Scientific Computing* 61.3 (2014), pp. 604–628. DOI: 10.1007/s10915-014-9838-9 (cit. on pp. 68, 117).

[Miu17] Tatsu-Hiko Miura. "On singular limit equations for incompressible fluids in moving thin domains." In: *Quarterly of Applied Mathematics* 76.2 (Dec. 2017), pp. 215–251. DOI: 10.1090/qam/1495 (cit. on pp. 1, 3, 21, 25, 35–37, 43–46).

[MRC06] Emilie Marchandise, Jean-François Remacle, and Nicolas Chevaugeon. "A quadrature-free discontinuous Galerkin method for the level set equation." In: *Journal of Computational Physics* 212.1 (2006), pp. 338–357. DOI: 10.1016/j.jcp.2005.07.006. URL: https://www.sciencedirect.com/science/article/pii/S0021999105003244 (cit. on pp. 108, 115, 138).

[MT01] Marius Mitrea and Michael Taylor. "Navier-Stokes equations on Lipschitz domains in Riemannian manifolds." In: *Mathematische Annalen* 321.4 (Dec. 2001), pp. 955–987. DOI: 10.1007/s002080100261 (cit. on pp. 1, 35, 56).

[NC17] Long Cu Ngo and Hyoung Gwon Choi. "Efficient direct re-initialization approach of a level set method for unstructured meshes." In: *Computers & Fluids* 154 (2017), pp. 167–183 (cit. on p. 108).

[NNV19] Michael Nestler, Ingo Nitschke, and Axel Voigt. "A finite element approach for vector- and tensor-valued surface PDEs." In: *Journal of Computational Physics* 389 (2019), pp. 48–61. DOI: https://doi.org/10.1016/j.jcp.2019.03.006 (cit. on pp. 18, 37, 59).

[NRV19] Ingo Nitschke, Sebastian Reuther, and Axel Voigt. "Hydrodynamic interactions in polar liquid crystals on evolving surfaces." In: *Phys. Rev. Fluids* 4 (4 Apr. 2019), p. 044002. DOI: 10.1103/PhysRevFluids.4.044002. URL: https://link.aps.org/doi/10.1103/PhysRevFluids.4.044002 (cit. on pp. 1–3, 19, 35–37, 40, 46–52).

[NRV20] Ingo Nitschke, Sebastian Reuther, and Axel Voigt. "Liquid crystals on deformable surfaces." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 476.2241 (2020), p. 20200313. DOI: 10.1098/rspa.2020.0313. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rspa.2020.0313 (cit. on pp. 1, 3, 35, 36).

[OF03] Stanley Osher and Ronald Fedkiw. "Level Set Methods and Dynamic Implicit Surfaces." In: *Applied Mechanics Reviews* 57.3 (June 2003), B15–B15. DOI: 10.1115/1.1760520 (cit. on pp. 31, 108).

[Ols+18] Maxim A. Olshanskii et al. "A finite element method for the surface Stokes problem." In: *SIAM Journal on Scientific Computing* 40.4 (2018), A2492–A2518 (cit. on pp. 2, 78, 85).

[Ons31a] Lars Onsager. "Reciprocal Relations in Irreversible Processes. I." In: *Phys. Rev.* 37 (4 Feb. 1931), pp. 405–426. DOI: 10.1103/PhysRev.37.405. URL: https://link.aps.org/doi/10.1103/PhysRev.37.405 (cit. on p. 42).

[Ons31b] Lars Onsager. "Reciprocal Relations in Irreversible Processes. II." In: *Phys. Rev.* 38 (12 Dec. 1931), pp. 2265–2279. DOI: 10.1103/PhysRev.38.2265. URL: https://link.aps.org/doi/10.1103/PhysRev.38.2265 (cit. on p. 42).

[OR17] Maxim A. Olshanskii and Arnold Reusken. "Trace Finite Element Methods for PDEs on Surfaces." In: *Geometrically Unfitted Finite Element Methods and Applications*. Ed. by Stéphane P. A. Bordas et al. Cham: Springer International Publishing, 2017, pp. 211–258 (cit. on pp. 2, 6).

[OR25] Maxim A. Olshanskii and Arnold Reusken. *Analysis of a finite element method for PDEs in evolving domains with topological changes*. 2025. URL: https://arxiv.org/abs/2504.14116 (cit. on pp. 164, 165).

[ORG09] Maxim Olshanskii, Arnold Reusken, and Jörg Grande. "A Finite Element Method for Elliptic Equations on Surfaces." In: *SIAM Journal on Numerical Analysis* 47 (Jan. 2009), pp. 3339–3358. DOI: 10.1137/080717602 (cit. on pp. 6, 60).

[ORS24] Maxim A. Olshanskii, Arnold Reusken, and Paul Schwering. "An Eulerian finite element method for tangential Navier–Stokes equations on evolving surfaces." In: *Mathematics of Computation* 93.349 (Dec. 2024), pp. 2031–2065. DOI: 10.1090/mcom/3931 (cit. on pp. 8, 77, 89, 91, 93, 96–98, 155).

[ORS25] Maxim A. Olshanskii, Arnold Reusken, and Paul Schwering. "A Narrow Band Finite Element Method for the Level Set Equation." In: *SIAM Journal on Scientific Computing* 47.2 (Mar. 2025), A762–A789. DOI: 10.1137/24m1674182 (cit. on pp. 8, 107, 108).

[ORX14] Maxim A Olshanskii, Arnold Reusken, and Xianmin Xu. "An Eulerian space–time finite element method for diffusion problems on evolving surfaces." In: *SIAM Journal on Numerical Analysis* 52.3 (2014), pp. 1354–1377 (cit. on p. 6).

[ORZ21] Maxim Olshanskii, Arnold Reusken, and Alexander Zhiliakov. "Inf-sup stability of the trace P2-P1 Taylor-Hood elements for surface PDEs." In: *Mathematics of Computation* 90.330 (2021), pp. 1527–1555 (cit. on pp. 78, 85, 93).

[ORZ22] Maxim A. Olshanskii, Arnold Reusken, and Alexander Zhiliakov. "Tangential Navier–Stokes equations on evolving surfaces: Analysis and simulations." In: *Mathematical Models and Methods in Applied Sciences* 32.14 (Dec. 2022), pp. 2817–2852. DOI: 10.1142/S0218202522500658 (cit. on pp. 2, 57, 61, 70, 77, 78, 91, 177).

[OS15] Maxim A. Olshanskii and Danil Safin. "A narrow-band unfitted finite element method for elliptic PDEs posed on surfaces." In: *Mathematics of Computation* 85.300 (Sept. 2015), pp. 1549–1570. DOI: 10.1090/mcom/3030 (cit. on p. 60).

[OS16] Maxim A Olshanskii and Danil Safin. "Numerical integration over implicitly defined domains for higher order unfitted finite element methods." In: *Lobachevskii Journal of Mathematics* 37.5 (2016), pp. 582–596 (cit. on p. 60).

[OY19] Maxim A Olshanskii and Vladimir Yushutin. "A Penalty Finite Element Method for a Fluid System Posed on Embedded Surface." In: *Journal of Mathematical Fluid Mechanics* 21.1 (2019), p. 14 (cit. on pp. 2, 78, 85).

[PR16] A. Petras and S.J. Ruuth. "PDEs on moving surfaces via the closest point method and a modified grid based particle method." In: *Journal of Computational Physics* 312 (May 2016), pp. 139–156. DOI: 10.1016/j.jcp.2016.02.024 (cit. on p. 60).

[Pre18] Janosch Preuß. "Higher order unfitted isoparametric space-time FEM on moving domains." MA thesis. Institute for Numerical and Applied Mathematics University of Göttingen, Feb. 2018 (cit. on pp. 69, 117).

[PS16] Jan Prüss and Gieri Simonett. *Moving Interfaces and Quasilinear Parabolic Evolution Equations*. Vol. 105. Birkhäuser Basel, Jan. 2016. DOI: 10.1007/978-3-319-27698-4 (cit. on pp. 14, 18, 19, 21, 22, 24, 25).

[PSW20] Jan Prüss, Gieri Simonett, and Mathias Wilke. "On the Navier–Stokes equations on surfaces." In: *Journal of Evolution Equations* 21.3 (Nov. 2020), pp. 3153–3179. DOI: 10.1007/s00028-020-00648-0 (cit. on pp. 19, 27, 56).

[Reu14] Arnold Reusken. "Analysis of trace finite element methods for surface partial differential equations." In: *IMA Journal of Numerical Analysis* 35.4 (Oct. 2014), pp. 1568–1590. DOI: 10.1093/imanum/dru047 (cit. on pp. 2, 60, 69, 70).

[Reu18] Arnold Reusken. "Stream function formulation of surface Stokes equations." In: *IMA Journal of Numerical Analysis* 40.1 (Oct. 2018), pp. 109–139. DOI: 10.1093/imanum/dry062 (cit. on p. 133).

[Reu24] Arnold Reusken. "Analysis of the Taylor-Hood surface finite element method for the surface Stokes equation." In: *Mathematics of Computation* (Aug. 2024). DOI: 10.1090/mcom/4008 (cit. on p. 59).

[RM08] Steven J. Ruuth and Barry Merriman. “A simple embedding method for solving partial differential equations on surfaces.” In: *Journal of Computational Physics* 227.3 (Jan. 2008), pp. 1943–1961. DOI: 10.1016/j.jcp.2007.10.009 (cit. on p. 60).

[RNV20] Sebastian Reuther, Ingo Nitschke, and Axel Voigt. “A numerical approach for fluid deformable surfaces.” In: *Journal of Fluid Mechanics* 900 (2020), R8. DOI: 10.1017/jfm.2020.564 (cit. on pp. 2, 46, 52, 56, 59).

[RST08] H.-G. Roos, M. Stynes, and L. Tobiska. *Robust Numerical Methods for Singularly Perturbed Differential Equations*. Springer Berlin Heidelberg, 2008 (cit. on p. 116).

[RV06] Andreas Rätz and Axel Voigt. “PDE’s on surfaces—a diffuse interface approach.” In: *Communications in Mathematical Sciences* 4.3 (2006), pp. 575–590. DOI: 10.4310/cms.2006.v4.n3.a5 (cit. on p. 2).

[RV18] Sebastian Reuther and Axel Voigt. “Solving the incompressible surface Navier–Stokes equation by surface finite elements.” In: *Physics of Fluids* 30 (2018) (cit. on pp. 59, 78, 166).

[RZ13] Arnold Reusken and Yuanjun Zhang. “Numerical simulation of incompressible two-phase flows with a Boussinesq–Scriven interface stress tensor.” In: *International Journal for Numerical Methods in Fluids* 73.12 (Aug. 2013), pp. 1042–1058. DOI: 10.1002/fld.3835. URL: http://dx.doi.org/10.1002/fld.3835 (cit. on p. 178).

[Say15] R.I. Saye. “High-order quadrature method for implicitly defined surfaces and volumes in hyperrectangles.” In: *SIAM Journal on Scientific Computing* 37.2 (2015), A993–A1019 (cit. on p. 60).

[Sch14] Joachim Schöberl. *C++11 Implementation of Finite Elements in NGSolve*. ASC Report 30/2014, Institute for Analysis and Scientific Computing, Vienna University of Technology. Sept. 2014. URL: https://ngsolve.org/ (cit. on pp. 7, 99, 131, 159).

[Sch25] Paul Schwering. *TraceFEM for full Surface Navier-Stokes equations*. Sept. 2025. DOI: 10.5281/zenodo.17120084. URL: https://doi.org/10.5281/zenodo.17120084 (cit. on p. 159).

[Sch97] Joachim Schöberl. “NETGEN - An advancing front 2D/3D-mesh generator based on abstract rules.” In: *Computing and Visualization in Science* 1.1 (July 1997), pp. 41–52. DOI: 10.1007/s007910050004 (cit. on pp. 7, 99, 131, 159).

[Scr60] L.E. Scriven. “Dynamics of a fluid interface Equation of motion for Newtonian surface fluids.” In: *Chemical Engineering Science* 12.2 (May 1960), pp. 98–108. DOI: 10.1016/0009-2509(60)87003-0 (cit. on p. 1).

[Set96] James A. Sethian. “Theory, algorithms, and applications of level set methods for propagating interfaces.” In: *Acta Numerica* 5 (1996), pp. 309–395. DOI: 10.1017/S0962492900002671 (cit. on pp. 31, 107).

[Set99a] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Second. Cambridge Monograph on Applied and Computational Mathematics. Cambridge Univeristy Press, 1999. URL: http://math.berkeley.edu/%5C~%7B%7Dsethian/2006/Publications/Book/2006/book%5C_1999.html (cit. on pp. 108, 143).

[Set99b] James A. Sethian. "Fast marching methods." In: *SIAM Review* 41 (1999), pp. 199–235 (cit. on p. 109).

[SF99] Mark Sussman and E. Fatemi. "An efficient interface preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow." In: *SIAM J. Sci. Comp.* 20 (1999), pp. 1165–1191 (cit. on p. 109).

[SR23] Hauke Sass and Arnold Reusken. "An accurate and robust Eulerian finite element method for partial differential equations on evolving surfaces." In: *Computers & Mathematics with Applications* 146 (2023), pp. 253–270. DOI: 10.1016/j.camwa.2023.06.040. URL: https://www.sciencedirect.com/science/article/pii/S0898122123002924 (cit. on p. 60).

[SR26] Hauke Sass and Arnold Reusken. "Analysis of a space-time unfitted finite element method for PDEs on evolving surfaces." To appear in: *Journal of Numerical Methods*. 2026. URL: https://arxiv.org/abs/2401.01215 (cit. on pp. 7, 61).

[SVD06] J.J. Sudirham, J.J.W. van der Vegt, and R.M.J. van Damme. "Space-time discontinuous Galerkin method for advection-diffusion problems on time-dependent domains." In: *Applied Numerical Mathematics* 56.12 (Dec. 2006), pp. 1491–1518. DOI: 10.1016/j.apnum.2005.11.003 (cit. on pp. 108, 116).

[SW13] Y Sudhakar and Wolfgang A Wall. "Quadrature schemes for arbitrary convex/concave volumes and integration of weak form in enriched partition of unity methods." In: *Computer Methods in Applied Mechanics and Engineering* 258 (2013), pp. 39–54 (cit. on p. 60).

[SW14] B. Schott and W. A. Wall. "A new face-oriented stabilized XFEM approach for 2D and 3D incompressible Navier–Stokes equations." In: *Computer Methods in Applied Mechanics and Engineering* 276 (2014), pp. 233–265. DOI: 10.1016/j.cma.2014.02.014 (cit. on pp. 68, 117).

[Tay92] Michael E Taylor. "Analysis on Morrey spaces and applications to Navier–Stokes and other evolution equations." In: *Communications in Partial Differential Equations* 17.9-10 (Jan. 1992), pp. 1407–1456. DOI: 10.1080/03605309208820892 (cit. on pp. 1, 56).

[TE00] A.-K. Tornberg and B. Engquist. "A finite element based level-set method for multiphase flow applications." In: *Comp. Vis. Sci.* 3 (2000), pp. 93–101 (cit. on p. 109).

[TMA19] Alejandro Torres-Sánchez, D. Millan, and Marino Arroyo. "Modelling fluid deformable surfaces with an emphasis on biological interfaces." In: *Journal of Fluid Mechanics* 872 (2019), pp. 218–271 (cit. on pp. 1, 3, 36).

[vS08] J.J.W. van der Vegt and J.J. Sudirham. "A space-time discontinuous Galerkin method for the time-dependent Oseen equations." In: *Applied Numerical Mathematics* 58.12 (2008), pp. 1892–1917. DOI: 10.1016/j.apnum.2007.11.010. URL: https://www.sciencedirect.com/science/article/pii/S0168927407001808 (cit. on pp. 108, 116).

[WRL21] Henry von Wahl, Thomas Richter, and Christoph Lehrenfeld. "An unfitted Eulerian finite element method for the time-dependent Stokes problem on moving domains." In: *IMA Journal of Numerical Analysis* 42.3 (July 2021), pp. 2505–2544. DOI: 10.1093/imanum/drab044 (cit. on pp. 6, 61, 70, 77, 94, 109, 117).

[XSL14] Shixin Xu, Ping Sheng, and Chun Liu. "An energetic variational approach for ion transport." In: *Communications in Mathematical Sciences* 12.4 (2014), pp. 779–789. DOI: 10.4310/cms.2014.v12.n4.a9 (cit. on p. 42).

[Xue+21] Tianju Xue et al. "A new finite element level set reinitialization method based on the shifted boundary method." In: *Journal of Computational Physics* 438 (2021), p. 110360 (cit. on pp. 108, 109).