CutFEM and ghost stabilization techniques for higher order space-time discretizations of the Navier–Stokes equations

Mathias Anselmann*1, Markus Bause*

* Helmut Schmidt University, Faculty of Mechanical Engineering, Holstenhofweg 85, 22043 Hamburg, Germany

Abstract. We propose and analyze computationally a new fictitious domain method, based on higher order space-time finite element discretizations, for the simulation of the nonstationary, incompressible Navier–Stokes equations on evolving domains. The physical domain is embedded into a fixed computational mesh such that arbitrary intersections of the moving domain’s boundaries with the background mesh occur. The potential of such cut finite element techniques for higher order space-time finite element methods has rarely been studied in the literature so far and deserves further elucidation. The key ingredients of the approach are the weak formulation of Dirichlet boundary conditions by Nitsche’s method, the flexible and efficient integration over all types of intersections of cells by moving boundaries and the spatial extension of the discrete physical quantities to the entire computational background mesh including fictitious (ghost) subdomains of fluid flow. Thereby, an expensive remeshing and adaptation of the sparse matrix data structure are avoided and the computations are accelerated. To prevent spurious oscillations caused by irregular intersections of mesh cells, a penalization, defining also implicitly the extension to ghost domains, is added. These techniques are embedded in an arbitrary order, discontinuous Galerkin discretization of the time variable and an inf-sup stable discretization of the spatial variables. The parallel implementation of the matrix assembly is described. The optimal order convergence properties of the algorithm are illustrated in a numerical experiment for an evolving domain. The well-known 2d benchmark of flow around a cylinder as well as flow around moving obstacles with arising cut cells and fictitious domains are considered further.

1Corresponding author: anselmann@hsu-hh.de
1 Introduction

Cut finite element methods (CutFEM) are subject to active current research. They are suitable when it comes to solving partial differential equations on evolving domains with moving boundaries. They have been analyzed and studied numerically for parabolic problems, using low order finite elements and BDF time stepping schemes; cf. e.g. [1]. This analysis was recently extended to the time-dependent Stokes problem using, for the spatial discretization, lowest order Taylor-Hood elements [2] or equal-order elements along with stabilization [3], combined with BDF time-stepping schemes. For instance, we refer to [2, Thm. 5.26] for a rigorous error estimate. Also, the simulation of incompressible flow on complex domains [4, 5, 6, 7] or even the simulation of coupled problems of multi-physics, for instance of fluid–structure interaction, with moving interfaces [8, 9, 10] becomes feasible with CutFEM based techniques. The appreciable advantage of the CutFEM approach is, that all the computations are done on a time-independent, fixed background mesh. However, mesh cells then are intersected in an arbitrary manner by moving boundaries of the domain or inner interfaces, for instance, if fluid-structure interaction is studied. In contrast to the well-known Arbitrary–Lagrangian–Eulerian (ALE) method (cf. [11, 12, 13]), no transformation of the physical domain to a fixed computational reference domain is induced by CutFEM and the computational mesh is not necessarily fitted to the outer boundaries of the domain or, for instance, the boundaries of enclosed rigid domains if flow around obstacles is studied. In classical ALE approaches, larger motions and deformations of the domains lead to a poor quality of the non-transformed, physical mesh (cf., e.g. [14, 15, 16]), since the computational mesh has to track and resolve moving boundaries or interfaces. This mesh deformation impacts the transformation of the model equations to the reference domain and, thereby, the stability of the overall ALE approach. To preserve the mesh quality and stability of the transformation, remeshing becomes inevitable, combined with the necessity to project discrete solutions from one mesh to another, which is computationally expensive and can cause an accumulation of projection errors (cf. [17]). Fixed-mesh ALE approaches (cf. [18]) try to overcome this issue by combining a fixed background mesh with the ALE concept. In each time step, the discrete functions are projected onto a fixed background mesh, so that additional degrees of freedom, used to preserve the mesh quality by introducing supplementary grid nodes, are naturally defined by the ALE approach.

The CutFEM approach overcomes the ALE transformation of the model equations to a reference domain. In CutFEM, the model is discretized on a fixed background mesh in its native formulation. The CutFEM methodology is based on three key ingredients: Firstly, boundary conditions of Dirichlet type are imposed in a weak variational form by using Nitsche’s method, cf. [19, 20, 21, 22, 23, 24]. Secondly, a flexible and efficient approach to integrate over all types of mesh cells resulting from intersections of computational grid cells by moving boundaries, referred to as cut cells, is needed. The efficient integration over such cut cells requires the computation of volume integrals over portions of the underlying finite element cells. This integration should not disturb the convergence order of the underlying space-time discretization scheme. One way to do this is to divide the cut cells into sub-elements and to apply a standard quadrature formula on the resulting triangulation; cf. [25, 26]. However, in this approach the mesh information has to be recomputed or rearranged for assembling the algebraic system, which counteracts the methodology and advantages of CutFEM techniques. There exist some remedies for this drawback, for instance, by using a refined sub-mesh for the integration over the cut cells, without adjusting the underlying mesh and degrees of freedom.
(cf. [27, 28]), which however is still a challenging task, in particular in the case of three space dimensions. An alternative is given by using the divergence theorem and transforming the volume integrals to surface integrals (line integrals in the two-dimensional case). The latter ones can then be computed by a moment fitting method (cf. [29]) or suitable quadrature formulas (cf. [30]). Our approach proposes a different technique, that deviates from the previous ones. We compute iterated integrals of multivariate calculus over all portions of cells intersected by the domain’s boundaries by applying (iterated) one-dimensional quadrature formulas. This results in a flexible and efficient algorithm.

The third and last ingredient of CutFEM techniques for evolving domains is the extension of discrete functions to fictitious or ghost subdomains or the whole computational mesh, respectively, along with the ghost penalization of the discrete variational equations; cf. [3, 2, 31, 5, 1]. The stabilization of the variational problem aims at reducing unphysical and spurious oscillations that are due to unavoidable irregular (small) cuts of mesh cells by moving boundaries. A strong analogy to the discretization of convection-dominated problems (cf. [32]) can be observed. Further, the stabilization also aims at improving the conditioning of the resulting algebraic system; cf. [33, 23, 1]. Proposed stabilization techniques are based on suitable extensions of the discrete functions to some neighborhood of the flow domain; cf. [2, 34]. We refer to this extension of the flow domain as a fictitious or ghost subdomain, since the physical unknowns are not defined in the domain’s extension by some mathematical problem. Usually, the prolongation of the discrete functions is restricted to some thin layer adjacent to the domain’s boundary; cf. [1, 2, 35, 36]. However, in our approach we extend all quantities to the entire computational mesh. More precisely, for our prototype model problem of flow around a moving rigid obstacle in a pipe (cf. Fig. 2.1) this means that the velocity and pressure variable are extended to the whole rigid obstacle. The appreciable advantage of this augmented extension is that the underlying degrees of freedom and mesh information are fixed over the whole simulation time. This frees us from an expensive remeshing, a redistribution of degrees of freedom and a rebuilding or updating of the sparse system matrix data structure. For all time steps of the considered time interval, a non-condensed system matrix for all degrees of freedom of the whole computational grid, as the union of the flow domain and fictitious or ghost subdomains, is computed. No adaption of the matrix’s data structure with respect to active degrees of freedom of the flow domain and non-active or fictitious degrees of freedom of the ghost subdomains, varying in time due to the evolving domain, is required. As concerns the stabilization itself, our approach differs from the popular derivative jump ghost penalization technique used in, e.g., [37, 35, 38, 5, 8]. In this approach jumps in the (higher order) derivatives are penalized over facets which requires the computationally expensive evaluation of all higher order derivatives up to the maximum polynomial degree. We adopt a computationally cheaper and more direct concept of ghost penalization suggested in [39, 1] for linear convection-diffusion-reaction equations and generalized in [2] to the Stokes system. In this approach, no derivatives are required and the penalization is not depending on the polynomial degree of the spatial finite element spaces, just on some scaling factors in terms of negative powers of the spatial mesh size and the viscosity. This direct stabilization is carefully adapted to the mathematical setting of the Navier–Stokes system with two unknown variables and their non-equal order, inf-sup stable approximation in space. The stability of the proposed choice of the scaling factors in terms of the spatial mesh size is demonstrated.

Finally, we mention that an arbitrary order, discontinuous Galerkin method is applied for the discretization of the time variable. In [37, 40], a piecewise linear and discontinuous in
time discretization, that is combined with CutFEM techniques, is developed. The integration over the cut cells follows similar ideas as used here, but differs by subdividing the flow region of the cut cells into triangles (in two space dimensions) and integrating over these triangles whereas iterated integration over the (partially) curved subcell is applied here. Our motivation for using a discontinuous approximation in time is to overcome some difficulty in the pressure approximation of equal-order and continuous in time discretizations of Navier–Stokes solutions. Precisely, the latter lacks from the availability of an (continuous and discrete) initial value for the pressure variable that is not provided explicitly by the Navier–Stokes system, but needed for the unique definition of the full pressure trajectory. We note that discrete pressure approximations of a continuous in time approximation, for instance in the Gauss quadrature nodes of a Gauss quadrature formula in time, become accessible without any initial pressure values, which however is not sufficient to compute the complete pressure trajectory. The discontinuous discretization in time can be applied in a natural way to time-dependent domains and combined with the extension of discrete functions to fictitious domains and the integration over cut cells. By using a discontinuous test basis we reduce the space-time formulation to a time-marching scheme. Appreciable advantage of this approach is that algebraic systems of reasonable dimension are obtained. However, for higher order time discretizations the block structure of the resulting algebraic system is still complex and requires highly efficient algebraic solvers. Geometric multigrid methods have proved to be very efficient, when it comes to solving the Navier–Stokes equations. For non-evolving domains, a suitable preconditioning technique for the Navier–Stokes system based on a geometric multigrid method is analyzed computationally in [43]. Its extension to evolving domains and CutFEM techniques is still an ongoing research and beyond the scope of this work.

In our numerical studies we consider a sequence of four test problems of increasing complexity to analyze the stability and performance properties of the proposed approach. Firstly, its convergence properties are analyzed for a time-dependent domain. Secondly, the well-know DFG benchmark of flow around a cylinder in two space dimensions is considered. Even though the domain is time-independent, it’s worth to analyze the accuracy of the proposed approach for this well-understood setting and to compare it with discretizations based on body-fitted techniques. Thereby, our integration over cut cells, the extension of discrete functions to fictitious (ghost) subdomains of the computational mesh and the ghost penalization are evaluated. In the third and fourth experiment, time-independent domains are studied again. Flow around moving rigid bodies is investigated and illustrated for two different settings.

This work is organized as follows. In Sec. 2, the prototype model problem and the notation are introduced. In Sec. 3 our space-time finite element approach for simulating the Navier–Stokes system on evolving domains is presented. In Sec. 4 practical aspects of the parallel implementation of the algorithms in the deal.II library and the linear algebra package Trilinos are addressed. In Sec. 5 the numerical results for a sequence of test problems are presented. We end with a summary and an outlook in Sec. 6.
2 Mathematical problem and notation

2.1 Mathematical problem

Without loss of generality, in this work we study a prototype model problem that is sketched in Fig. 2.1. Restricting ourselves to considering incompressible viscous flow around a moving, undeformable, rigid body in a pipe is only done in order to simplify the notation, reduce technical ballast and focus on the essential features of the numerical techniques. The problem is of high interest in practice and can be regarded as a test problem for more sophisticated applications, for instance, as a building block for fluid-structure interaction. By \( \Omega = (0, L) \times (0, W) \), with some \( L > 0 \) and \( W > 0 \), we denote the rectangular pipe of length \( L \) and width \( W \). Let \( T > 0 \) be the final time. About the rigid body \( \Omega_r(t) \) we make the following assumption.

Assumption 2.1. The smooth motion of the bounded rigid body \( \Omega_r^t := \Omega_r(t) \), with positive measure \( \text{meas}_2(\Omega_r^t) > 0 \) and sufficiently smooth boundary \( \Gamma_r^t := \partial \Omega_r^t(t) \), for \( t \in [0, T] \), is supposed to be given. Contact between the pipe boundary \( \Gamma_p := \Gamma_i \cup \Gamma_w \cup \Gamma_o \) and the rigid body is assumed not to occur such that \( \Omega_r^t \subset \Omega \) and, with some suitable constant \( \delta > 0 \),

\[
\text{dist}(\Gamma_r^t, \Gamma_p) \geq \delta > 0, \quad \text{for } t \in [0, T].
\]

By \( \Omega_f^t := \Omega_f(t) = \Omega \setminus \Omega_r^t \), with \( \Omega_f^t \subset \Omega \), we denote the open domain filled with fluid. For \( \Omega_f^t \), with \( t \in [0, T] \), and \( I := (0, T] \) we consider solving the incompressible Navier–Stokes system

\[
\begin{align*}
\partial_t v + (v \cdot \nabla)v - \nu \Delta v + \nabla p &= f & \text{in } \Omega_f^t \times I, \quad (2.1a) \\
\nabla \cdot v &= 0 & \text{in } \Omega_f^t \times I, \quad (2.1b) \\
v &= 0 & \text{on } \Gamma_w \times I, \quad (2.1c) \\
v &= g_i & \text{on } \Gamma_i \times I, \quad (2.1d) \\
v &= g_r & \text{on } \Gamma_r^t \times I, \quad (2.1e) \\
\nu \nabla v \cdot n - np &= 0 & \text{on } \Gamma_o \times I, \quad (2.1f) \\
v(0) &= v_0 & \text{in } \Omega_f^t. \quad (2.1g)
\end{align*}
\]

In (2.1), the velocity field \( v \) and the pressure \( p \) are the unknown variables. In (2.1a), the parameter \( \nu > 0 \) denotes the fluid’s viscosity and the right-hand side function \( f \) is a given...
external force. In (2.11), the field \( n \) is the outer unit normal vector. In (2.12), the function \( v_0 \) denotes the prescribed initial velocity. On \( \Gamma_i \), an inflow flow profile is prescribed by (2.1d). The boundary parts \( \Gamma_w \) are fixed walls with no-slip boundary (2.1c), and \( \Gamma_o \) represents an outflow boundary that is modeled by the do-nothing boundary condition (2.1f); cf. [47]. The boundary condition (2.1e) ensures that the fluid follows the prescribed rigid body’s motion \( g_r \) on \( \Gamma_t \). Here, \( g_r \) is the prescribed material velocity of the particles from \( \Gamma_t \). To simplify the notation, we put

\[
\begin{align*}
  g &= \begin{cases} 
    0 & \text{on } \Gamma_w \times I, \\
    g_i & \text{on } \Gamma_i \times I, \\
    g_r & \text{on } \Gamma_t \times I
  \end{cases}
\end{align*}
\]

and, with \( \Gamma_D := \Gamma_i \cup \Gamma_w \cup \Gamma_t \), we rewrite the conditions (2.1c) to (2.1e) by

\[
v = g \quad \text{on } \Gamma_D \times I.
\]

We assume, that a unique (weak) solution of the system (2.1) exists. This implies that sufficient smoothness conditions about the evolving flow domain are satisfied. For the existence and uniqueness of solutions to the Stokes and Navier–Stokes system on time-dependent domains we refer to, e.g., [3, 48, 49]. In particular, we refer to [48, p. 156, Thm. 1, p. 164, Cor.] for the existence of local in time solutions to (2.1) or solutions for sufficiently small data. Even though solutions to (2.1) might lack (due to the polygonal pipe boundary) higher order regularity, that is needed to expect an optimal order convergence behavior of higher order discretization methods, members of such families are applied here. In the numerical approximation of partial differential equations it is widely accepted that higher order methods achieve accurate results on computationally feasible grids, even if the solution of the mathematical problem is not sufficiently smooth on the whole space-time domain. This also applies to the lack of regularity of Navier–Stokes solutions for \( t \to 0 \) under realistic assumptions about the data of the problem (cf. [50, 51, 52]) which is not considered here.

### 2.2 Function spaces and forms

Here, we introduce the functions spaces that are used in this work to present our space-time CutFEM approach for the Navier–Stokes system (2.1). For the analysis of solutions to (2.1) or the discrete scheme a more sophisticated framework is needed. The proof of well-posedness and error analyses are more involved than in the case of fixed domains. In particular, a diffeomorphism mapping the evolving domain to a reference domain and the transformation of the model equations to an equivalent system on the reference domain by the diffeomorphism are utilized strongly. For further details we refer to, e.g., [4] for the Stokes system and to [53] for an abstract framework for parabolic partial differential equations on evolving spaces.

By \( L^2(S) \) we denote the function space of square integrable functions on a domain \( S \) while \( H^1(S) \) is the usual Sobolev space of functions in \( L^2(S) \) which have first order weak derivatives in \( L^2(S) \). Further, \( \langle \cdot, \cdot \rangle_S \) is the standard inner product of \( L^2(S) \). We define the subspace of \( L^2(S) \) with mean zero \( L^2_0(S) := \{ v \in L^2(S) \mid \int_S v \, dx = 0 \} \) and the subspace of \( H^1(S) \) of functions with zero boundary values (in the sense of traces) on the Dirichlet portion \( \Gamma \subset \partial S \) of the boundary \( \partial S \) of \( S \) as \( H^1_0,\Gamma(S) \). Finally, by \( H^{1/2}(\Gamma) \) we denote the space of all traces on \( \Gamma \subset \partial S \) of functions in \( H^1(S) \). For vector-valued functions we write those spaces bold.
For the weak problem formulation we introduce the semi-linear form $A_S : (H^1(S) \times L^2_0(S)) \times (H^1_{0,\Gamma}(S) \times L^2_0(S)) \to \mathbb{R}$ by

$$A_S((v, p), (\psi, \xi)) := \langle (v \cdot \nabla)v, \psi \rangle_S + \nu \langle \nabla v, \nabla \psi \rangle_S - \langle p, \nabla \cdot \psi \rangle_S + \langle \nabla \cdot v, \xi \rangle_S$$

(2.2)

for $(v, p) \in H^1(S) \times L^2_0(S)$ and $(\psi, \xi) \in H^1_{0,\Gamma}(S) \times L^2_0(S)$. For given $f \in L^2(S)$ we introduce the linear form $L : H^1(S) \to \mathbb{R}$ by

$$L_S(\psi; f) := \langle f, \psi \rangle_S$$

(2.3)

for $\psi \in H^1_{0,\Gamma}(S)$.

Further, we define the spaces

$$V_I := \left\{ v \in L^2(I; H^1(\Omega^f_{t})) \mid \partial_t v \in L^2(I; L^2(\Omega^f_{t})) \right\},$$

$$V_{0,I} := \left\{ v \in L^2(I; H^1_{0,\Gamma_D}(\Omega^f_{t})) \right\}$$

(2.4)

and

$$L^2_{0,I} := L^2(I; L^2_0(\Omega^f_{t})), \quad L^\infty_I := L^\infty(I; H^1(\Omega^f_{t})).$$

(2.5)

In the first of the definitions in (2.4), the weak partial derivative with respect to the time variable $t$ is defined as an element of $L^2(I; L^2(\Omega^f_{t}))$ for the space-time flow domain $\Omega_{f,I} := \bigcup_{0 \leq t \leq T} \Omega^f_{t} \times \{t\}$. The definitions (2.4) and (2.5) are sufficient here. For an abstract framework to treat evolving spaces we refer to [53]. Finally, we let

$$V^I_{\text{div}} := \left\{ v \in H^1(\Omega^f_{t}) \mid \nabla \cdot v = 0 \text{ in } \Omega^f_{t} \right\}.$$

### 2.3 Space discretization

Let $T_h = \{K\}$ be a family of shape-regular, structured decompositions of the pipe $\Omega$ (cf. Fig. 2.1) into (open) quadrilaterals $K$ with maximum cell size $h$. Precisely, $T_h$ is the computational background mesh. It is not fitted to the boundary of the moving rigid body $\Omega^f_{t}$; cf. Fig. 2.2. We make the following assumptions.

**Assumption 2.2.** The family $T_h = \{K\}$ satisfies:

1. The background mesh is independent of the time $t$.
2. The mesh is cartesian, such that the cells are aligned along the coordinate lines.
3. Each face of a quadrilateral is cut at most once by the rigid body

The opportunity to use structured meshes facilitates their generation which is an appreciable advantage of CutFEM. The third of the assumptions is made in order to simplify the implementation when it comes to the integration over cut cells. For time-independent domains, this condition is always fulfilled, if the mesh size is chosen sufficiently small with respect to the size of the rigid body. For evolving domains, this condition can be violated, if $\Omega^f_{t}$ has a curved
boundary. In our implementation we ensure, that the condition of item 3 is always fulfilled, cf. Sec. 4 for details.

The time-dependent set of mesh cells \( K \in T_h \) that are subsets of the fluid domain \( \Omega^f_t \) is denoted by \( T^f_{h,f} \), the mesh cells \( K \in T_h \) that are subsets of the rigid domain \( \Omega^r_t \) is denoted by \( T^r_{h,r} \). The set of cut cells \( K \in T_h \) such that \( K \cap \Omega^f_t \neq \emptyset \) and \( K \cap \Omega^r_t \neq \emptyset \) is denoted by \( T^c_{h,c} \); cf. Fig. 2.2c.

For some \( r \in \mathbb{N} \), let \( H^r_h = H^r_h(\Omega) \) be the finite element space given by

\[
H^r_h = \{ v_h \in C(\Omega) \mid v_h|_T \in Q_r(K) \forall K \in T_h \},
\]

(2.6)

where \( Q_r(K) \) is the space defined by the multilinear reference mapping of polynomials on the reference element with maximum degree \( r \) in each variable. For the numerical experiments that are presented in Sec. 5 we used the Taylor–Hood family of inf-sup stable finite element pairs for the space discretization. These elements can be replaced by any other type of inf-sup stable elements (cf. [54]), when the proposed ghost penalty stabilization of Sec. 3.2 is applied. For brevity, we restrict ourselves to the presentation of the Taylor–Hood family. For some natural number \( r \geq 2 \) and with (2.6) we then put

\[
V_h = H^r_h \times H^{r-1}_h, \quad Q_h = H^{r-1}_h.
\]

The space of weakly divergence free functions is denoted by

\[
V^\text{div} = \{ v_h \in V_h \mid \langle \nabla \cdot v_h, q_h \rangle = 0 \text{ for all } q_h \in Q_h \}.
\]

Finally, we define the spaces

\[
V_{I,h} := \{ v_h \in L^2(I; V_h) \mid \partial_t v \in L^2(I; V_h) \}, \quad L^2_{0,I,h} := L^2(I; Q_h).
\]

### 2.4 Time discretization

For the time discretization, we decompose the time interval \( I = (0,T] \) into \( N \) subintervals \( I_n = (t_{n-1}, t_n], \ n = 1, \ldots, N \), where \( 0 = t_0 < t_1 < \cdots < t_{N-1} < t_N = T \) such that \( I = \bigcup_{n=1}^N I_n \)
and $I_n \cap I_m = \emptyset$ for $n \neq m$, $m, n = 1, \ldots, N$. We put $\tau = \max_{n=1,\ldots,N} \tau_n$ with $\tau_n = t_n - t_{n-1}$. Further, the set $\mathcal{M}_\tau := \{I_1, \ldots, I_N\}$ of time intervals is called the time mesh. For a Banach space $B$ of functions defined on the time-independent domain $\Omega$ and any $k \in \mathbb{N}_0$, we let

$$\mathbb{P}_k(I_n; B) = \left\{ w_{\tau} : I_n \rightarrow B \mid w_{\tau}(t) = \sum_{j=0}^k W_j \tau^j \forall t \in I_n, \ W_j \in B \ \forall j \right\}. \quad (2.7)$$

For an integer $k \in \mathbb{N}_0$, we put

$$X^k(\tau) := \left\{ w_{\tau} \in L^2(I; B) \mid w_{\tau}|_{I_n} \in \mathbb{P}_k(I_n; B) \ \forall I_n \in \mathcal{M}_\tau, \ w_{\tau}(0) \in B \right\}. \quad (2.8)$$

3 Space–time finite element discretization with CutFEM

Here we introduce our approximation of the Navier–Stokes system (2.1) by combining CutFEM and variational time discretization techniques. Firstly, we introduce the variational formulation of the Navier–Stokes system (2.1). The next subsection is devoted to the discretization in space. Nitsche’s method [21] is applied to incorporate Dirichlet boundary conditions in a weak form. Further, the ghost penalty stabilization along with the extension of the discrete in space functions to fictitious subdomains of fluid flow, i.e. to the domain of the moving rigid body here, are introduced. Together, the stabilization and the extension provide an implicit definition of the discrete functions on the background mesh for the time-independent pipe domain $\Omega$. The stabilization reduces spurious oscillations by irregular cuts of finite elements by the moving rigid body $\Omega_t$ and improves the conditioning of the resulting algebraic system; cf. [1]. Finally, the discretization in time by the discontinuous Galerkin method is done and the resulting fully discrete problem is presented. In this work, the space-time finite element approach is recovered as a time marching scheme by the choice of a discontinuous test basis. This is done with the perspective of getting linear systems of reasonable size, even though the systems still have a complex block structure due to the higher order time discretization; cf. [24].

3.1 Variational formulation of the continuous problem

A sufficiently regular solution of the Navier–Stokes system (2.1) satisfies the following variational problem that provides the basis for space-time finite element discretizations.

**Problem 3.1** (Continuous problem). Let $f \in L^2(\Omega_f, I)$ and $v_0 \in V^0_{\text{div}}$ be given. Let $\hat{g} \in V_I \cap L^\infty_I$ denote a prolongation of the boundary values on $\Gamma^f_D$ to $\Omega^f_I$, such that $\hat{g} = g$ on $\Gamma^f_D \times I$. Find $(v, p) \in (V_I \cap L^\infty_I) \times L^2_{0, I}$, with $v \in \hat{g} + V_0, I$, such that $v(0) = v_0$ and for all $\phi := (\psi, \xi) \in V_0, I \times L^2_{0, I}$,

$$\int_0^T \langle \partial_t v, \psi \rangle_{\Omega^f_I} + A_{\Omega^f_I} ((v, p), (\psi, \xi)) \ dt = \int_0^T L_{\Omega^f_I}(\psi; f) \ dt. \quad (3.1)$$

**Remark 3.1.** Regarding the well-posedness of Problem 3.1 we note the following.
• In [48], p. 156, Thm. 1], the existence and uniqueness of local in time weak solutions to Eqs. (2.1a), (2.1b), (2.1g) equipped with homogeneous Dirichlet boundary conditions is shown under sufficient regularity assumptions about the data and motion of the evolving domain. The technical details of the assumptions about the domain motion are skipped here. In particular, \( v \in L^2_T \) and \( \partial_t v \in L^2(\Omega_T) \) is ensured on a sufficiently small time interval. Further, in [48] p. 164, Cor.] the existence of weak solutions is proved under smallness assumptions about the data.

• For evolving domains, the functional analytical setting for proving the existence of weak solutions differs from the one used for time-independent domains. Stricter assumptions about the data and domain are required. Then, a velocity field with time derivative \( \partial_t v \in L^2(\Omega_T) \) about the right-hand side function \( f \) is made here. Further regularity conditions about \( f \) are supposed in [48] for the proof of existence and uniqueness.

• For \( v \in V_I \cap L^\infty_T \) and \( p \in L^2_{0,t} \) the left-hand side terms in (3.1) are well-defined. The existence of the sufficiently smooth extension \( \hat{g} \) is tacitly assumed here. This requires smoothness assumptions about the evolving boundary of the fluid domain and assumptions about polygonal portions of the boundary. We skip the technical details of the assumptions since we focus on the discrete scheme here. On the discrete finite element level we avoid such extensions by employing Nitsche’s method.

3.2 Semidiscretization in space with weak enforcement of boundary conditions by Nitsche’s method and ghost penalty stabilization

To incorporate Dirichlet boundary conditions on unfitted meshes we use Nitsche’s method and follow [21, 55]. Here, the Dirichlet boundary conditions are enforced in a weak form by adding face integrals to the variational problem. We also refer to [24] where Nitsche’s method is applied to a higher order variational time discretization of the Navier–Stokes system on time-independent domains. Degrees of freedom assigned to Dirichlet portions of the boundary are now treated as unknowns of the variational problem. They are no longer enforced by the underlying function space and their implementation into the algebraic system, as it is usually done, with subsequent condensation of the algebraic equations.

Let \( t \in [0, T] \). For the treatment of Dirichlet boundary conditions by Nitsche’s method (cf. [21]) we introduce the bilinear form \( B_{\Gamma_D} : H^{1/2}(\Gamma_D) \times (V_h \times Q_h) \to \mathbb{R} \) by

\[
B_{\Gamma_D}(w, (\psi_h, \xi_h)) := - \langle w, \nu \nabla \psi_h \cdot n + \xi_h n \rangle_{\Gamma_D} + \gamma_1 \nu \langle h^{-1} w, \psi_h \rangle_{\Gamma_D} + \gamma_2 \langle h^{-1} w \cdot n, \psi_h \cdot n \rangle_{\Gamma_D}
\]

(3.2)

for \( w \in H^{1/2}(\Gamma_D) \) and \( (\psi_h, \xi_h) \in V_h \times Q_h \), where \( \gamma_1 > 0 \) and \( \gamma_2 > 0 \) are numerical (tuning) parameters for the penalization. In [6, 55, 51], it is demonstrated computationally that their choice in the interval \((10, 100)\) leads to robust results. In our simulation presented in Sec. 5 we put \( \gamma_1 = \gamma_2 = 35 \). The discrete semilinear form \( A_h : (V_h \times Q_h) \times (V_h \times Q_h) \to \mathbb{R} \) is then given by

\[
A_h((v_h, p_h), (\psi_h, \xi_h)) := A_{\Gamma_D}((v_h, p_h), (\psi_h, \xi_h)) - \langle \nu \nabla v_h \cdot n - p_h n, \psi_h \rangle_{\Gamma_D} + B_{\Gamma_D}(v_h, \phi_h)
\]

(3.3)
for \((v_h, p_h) \in V_h \times Q_h\) and \((\psi_h, \xi_h) \in V_h \times Q_h\). The linear form \(L_h : (V_h \times Q_h) \to \mathbb{R}\) is defined by

\[
L_h((\psi_h, \xi_h); f, g) := L_{\Omega_f}(\psi_h; f) + B_{\Gamma_D}(g, (\psi_h, \xi_h))
\]

for \((\psi_h, \xi_h) \in V_h \times Q_h\). In (3.3) and (3.4), the forms \(A_h\) and \(L_h\) defined in (2.2) and (2.3), respectively, are extended naturally to test functions of \(V_h\) with nonhomogeneous Dirichlet conditions.

Remark 3.2.

- We note that the forms (3.2) and (3.3) are introduced for (spatially) discrete functions that are defined on the time-independent background mesh \(\mathcal{T}_h\) of the pipe \(\Omega\). In (3.2) and (3.3), the integration is done over the evolving fluid domain \(\Omega_f^t\) and its boundary part \(\Gamma_D^t\) where Dirichlet boundary conditions are prescribed. The domain of the rigid body \(\Omega_r^t\) is considered as a fictitious (ghost) flow domain. In the following, the discrete fluid velocity \(v_h\) and pressure \(p_h\) are defined implicitly in the ghost domain \(\Omega_r^t\) by the (semi-) discrete variational problem that is augmented by a ghost penalty stabilization exploiting an extension of the discrete functions to \(\Omega_r^t\).

- We comment on the different boundary terms in the forms (3.2) and (3.3). The second term on the right-hand side of (3.3) reflects the natural boundary condition, making the weak imposition of the boundary conditions consistent. The first term on the right-hand side of (3.2) is introduced to preserve the symmetry properties of the continuous system. The last two terms are penalizations, that ensure the stability of the discrete system. In the inviscid limit \(\nu = 0\), the last term amounts to a “no-penetration” condition. Thus, the form (3.2) provides a natural weighting between boundary terms corresponding to viscous effects \((v = g)\), convective behavior \(((v \cdot n)^{v} = (g \cdot n)^{g})\) and inviscid behavior \((v \cdot n = g \cdot n)\).

To control irregular cuts of finite elements by the evolving domain and extend the fluid velocity and pressure functions from \(\Omega_f^t\) to \(\Omega_r^t\), and thereby to the entire, time-independent background domain \(\Omega\), we introduce a discrete ghost penalty operator \(S_h\). This stabilization then expands the semi-linear form (3.3) by additional terms. Our stabilization modifies an approach that was firstly introduced in [1] for convection-diffusion-reaction equations and, then, considered in [2] for Stokes problems to the Navier–Stokes system (2.1). This approach offers the appreciable advantage over further implementations of the ghost penalty method (cf. [1]) that no higher order spatial derivatives have to be computed. Thereby, it leads to reduced computational costs. The twofold motivation of the ghost penalty stabilization is sketched in Fig. 3.1 and Fig. 3.2 respectively. The first aim of the ghost stabilization \(S_h\), defined in (3.6), is to extend the solution from the physical fluid domain \(\Omega_f^t\) to the rigid domain \(\Omega_r^t\). This is illustrated in Fig. 3.1. To assemble the algebraic system, i.e. the Jacobian matrix and right-hand-side function of the Newton linearization that is applied here, the discrete solution that is already computed for some time point \(\tilde{t}_{n-1}\) in the corresponding fluid domain \(\Omega_f^{\tilde{t}_{n-1}}\) needs to be evaluated in the fluid domain \(\Omega_f^{\tilde{t}_n}\) at time \(\tilde{t}_n\). For instance, in the case of the lowest-order discontinuous Galerkin time discretization the time points \(\tilde{t}_{n-1}\) and \(\tilde{t}_n\) correspond to the midpoints of the subintervals \(I_{n-1}\) and \(I_n\) such that \(\tilde{t}_{n-1} = (t_{n-2} + t_{n-1})/2\) and \(\tilde{t}_n = (t_{n-1} + t_{n})/2\). In the case of higher order discontinuous Galerkin time discretizations, more time nodes or degrees of freedom in time are involved, but the extension problem applies similarly. Due to the motion of the rigid body \(\Omega_r^t\), the discrete velocities and pressure values at time \(\tilde{t}_{n-1}\) are not necessarily defined in
those parts of the rigid domain $\Omega^{t_{n-1}}_{r}$ that belong to the fluid domain $\Omega^{t_{n}}_{f}$ at time $t_{n}$ such that an appropriate extension of the solution at time $t_{n-1}$ to the rigid domain at time $t_{n-1}$ or to the entire time-independent domain $\Omega$, as done in this work, becomes indispensable. For this reason, we denote the domain of the rigid body as the fictitious or ghost domain of fluid flow, since discrete velocities and pressures are not defined here by means of a physical problem, but by some artificial extension only.

![Figure 3.1](image_url)

Figure 3.1: A moving rigid domain at two different time points $t_{n-1}$ and $t_{n}$ with computational background mesh.

The second aim of the stabilization is to reduce spurious and unphysical oscillations. They are due to unavoidable irregular or tiny cuts of finite elements by the moving rigid domain. This is illustrated in Fig. 3.2a. Here, we refer to the color range indicating the instability. The irregular cuts lead to steep gradients and oscillations in the computed solution profile. This is observation is related to the well-known challenges in the numerical approximation of convection-dominated transport. By steep gradients in the solution profile, the condition number of the corresponding linear algebraic system increases strongly which prevents their efficient solution by iterative methods. This problem of instability is sketched in Fig. 3.2b. For the sake of simplicity, flow around a fixed (non-moving) cylinder is simulated on a computational background mesh with arising cut cells. Here, no extension of the discrete solution to the rigid body is applied. In cut cells, the integration is done over the fluid part of the cell only. In Fig. 3.2a, an instability of the discrete solution in the cut cells is observed. By usage of ghost penalty stabilization that is proposed in the following, a smooth extension is obtained; cf. Fig. 3.2c. We explicitly note, that this extension admits no reasonable physical interpretation, since it is not based on any mathematical model but represents a numerically motivated approach.

In the sequel, we extend the ghost penalty approach that is proposed in [1] for linear transport equations and extended in [2] to Stokes problems to the case of the Navier–Stokes system. However, our stabilization differs from the one that is suggested in [2]. In [2] the stabilization covers a boundary strip, i.e. the cut cells and cells of the fluid and the rigid domain adjacent to the boundary $\Gamma^{t}_{r}$. In the analysis, this is exploited to prove the error estimates. Here, we apply the combined stabilization and extension on the submeshes of the cut cells and the cells covering the rigid body (fictitious domain) $\Omega^{t}_{r}$ and, thus, to the entire domain $\Omega$ of our problem setting (cf. Fig. 2.1). This increases the efficiency of the parallel implementation (cf. Sec. 4). Our numerical experiments (cf. Sec. 5) indicate the admissibility of this modified ghost penalty stabilization and extension. For the problem setting of Fig. 2.1 we denote by the time-dependent submesh $T^{t}_{h,s}$ the set of all cut cells and all cells that are entirely in the rigid domain,

$$T^{t}_{h,s} := T^{t}_{h,r} \cup T^{t}_{h,c} = \{ K \in T_{h} \mid K \subset \Omega^{t}_{r} \text{ or } K \cap \Omega^{t}_{f} \neq \emptyset \neq K \cap \Omega^{t}_{r} \} .$$
(a) Irregular tiny cut cells of the background mesh due to the non body-fitted triangulation.

(b) Solution without ghost-penalty stabilization and extension. Cells $\in T_{th,r}$ are colored black and were inactive.

(c) Solution with ghost-penalty stabilization and fictitious domain extension to rigid body $\Omega^r_t$.

Figure 3.2: Effect of ghost-penalty stabilization and extension.

The set of all the faces that are common to two cells $K_1$ and $K_2$ of the submesh $T_{th,s}$ is defined by

$$F_{th} := \{ K_1 \cap K_2 \mid K_1 \in T_{th,s}, K_2 \in T_{th,s}, K_1 \neq K_2, \text{ meas}_1(K_1 \cap K_2) > 0 \}.$$  

(3.5)

For $F \in F_{th}$, we denote by $\omega_F = K_1 \cup K_2$ the patch of the two adjacent elements $K_1$ and $K_2$ with the common face $F$. Then we define the bilinear form $S_{Fh} : (V_h \times Q_h) \times (V_h \times Q_h) \to \mathbb{R}$ by

$$S_{Fh}((v_h,p_h),(\psi_h,\xi_h)) = \sum_{F \in F_{th}} \gamma_v \left( \frac{1}{H^2} \right) \langle (\mathcal{E}v_{h|K_1} - \mathcal{E}v_{h|K_2}), (\mathcal{E}\psi_{h|K_1} - \mathcal{E}\psi_{h|K_2}) \rangle_{\omega_F}$$

$$+ \gamma_p \left( \frac{1}{H^2} \right) \langle (\mathcal{E}p_{h|K_1} - \mathcal{E}p_{h|K_2}), (\mathcal{E}\xi_{h|K_1} - \mathcal{E}\xi_{h|K_2}) \rangle_{\omega_F},$$

(3.6)

with numerical parameters $\gamma_v, \gamma_p > 0$. The stabilization (3.6) is based on [2]. However, in this work it is applied to the cut and rigid body cells given by $T_{th,s}$. This approach differs from [2] which is motivated by our aim to use a time-independent background mesh for computational efficiency. The performed computations (cf. Sec. 5) illustrate the robustness of the extension and stabilization. The restriction of the ghost penalty stabilization to a strip around the fluid domain is studied in a further work; cf. [57]. A superiority of the latter stabilization is not observed. In Eq. (3.6), $\mathcal{E}$ is the canonical extension of a polynomial function of degree $r$ in each variable, that is defined on one element of the face patch $\omega_F$, to the whole patch; cf. [1]. Thus, we have that

$$\mathcal{E} : Q_r(K_i) \to Q_r(\omega_F), \quad \text{for } i \in \{1, 2\},$$

(3.7)

with $(\mathcal{E} u)_{K_i} = u$ for $u \in Q_r(K_i)$. In the case of vector-valued functions, the extension (3.7) is applied component wise. To illustrate the extension (3.7), Fig. 3.3a shows the extended function $\mathcal{E}v_1$ of a scalar valued Lagrange $Q_1$ finite element Lagrangian basis function $v_1$ that
is defined on the quadrilateral $K_1$ and has the value 1 in the grid node $(1, 0)^\top$. In Fig. 3.3b we illustrate the application of the stabilization operator (3.6) for the cut cell $K_1$. This cell has four faces, that all belong to the set $F_h^t$. For each of these faces we apply the stabilization (3.6) along with the extension (3.7). We start with the face $F_2$, marked by a full red line. The corresponding face patch, built from the cells $K_1$ and $K_2$, i.e. $\omega_F = K_1 \cup K_2$, is colored in purple in Fig. 3.3b. After evaluating the extension (3.6) on this first face patch we continue with the remaining three face patches, that are respectively built from the faces marked by the dashed red line. Proceeding in this way with all faces in $F_h^t$, we implicitly extend a discrete velocity $v_h$ or pressure $p_h$ to $\Omega_h^f$ by adding the stabilization (3.6) to the space discretization. The implicit definition of the extension of the discrete functions comes through the fact that the function values in the extended (ghost) domain $\Omega_h^f$ are obtained by the solution of the algebraic system and is not explicitly prescribed.

(a) Canonical extension $\mathcal{E}v_1$ of a Lagrange basis function from $K_1$ to $\omega_F = K_1 \cup K_2$.

(b) Patchwise extension to a fictitious domain for a cut cell $K_1$, with tiny, irregular cut.

Figure 3.3: Patchwise extension of ghost penalty stabilization.

Finally, we define the stabilized semilinear form $A^s_h : (V_h \times Q_h) \times (V_h \times Q_h) \to \mathbb{R}$ by

$$A^s_h((v_h, p_h), (\psi_h, \xi_h)) := A_h((v_h, p_h), (\psi_h, \xi_h)) + S^h_{F^t_h}((v_h, p_h), (\psi_h, \xi_h))$$  \hspace{1cm} (3.8)

for $(v_h, p_h) \in V_h \times Q_h$ and $(\psi_h, \xi_h) \in V_h \times Q_h$.

**Remark 3.3.** We note that $A^s_h$ is defined on the time-independent bulk space $V_h \times Q_h$. The volume integrals of $A_h$ (cf. (3.3)) in (3.8) and of $L_h$ in (3.4) are computed over the time-dependent fluid domain $\Omega_h^f$. Thus, an integration over the fluid portion of the cut cells has to be provided. This is done in Sec. 4. The penalty stabilization $S^h$ (cf. (3.6)) is computed over time-dependent patches that are subsets either of the fluid and the rigid domain or of the rigid one only.

**Remark 3.4.** Since we extend the solution of the physical fluid domain $\Omega_f$ to the whole computational domain $\Omega$, the choice of the time step size is in principle not restricted by a geometric CFL condition as it arises in [58]. This is confirmed by numerical experiments. Nevertheless, the choice of the time step size affects the performance and robustness of the ghost penalty stabilization. Its sensitivity depends on the geometrical and flow parameters like the diameter of the rigid domain and the fluid viscosity such that the time step size should be chosen reasonably.
We are now in a position to define a semidiscrete approximation of the system (2.1).

**Problem 3.2.** Let \( f \in L^2(\Omega_{f,t}) \) and \( v_{0,h} \in V_h^{\text{div}} \) be given. Find \((v_h, p_h) \in V_{I,h} \times L^2_{0,I,h}\), such that \( v_h(0) = v_{0,h} \) and for all \((\psi, \xi) \in L^2(I; V_h) \times L^2_{0,I,h}\):

\[
\int_0^T \langle \partial_t v_h, \psi \rangle_{\Omega_t} + A^h_v ((v_h, p_h), (\psi_h, \xi_h)) \, dt = \int_0^T L_h(\psi; f, g) \, dt . \tag{3.9}
\]

### 3.3 Fully discrete problem

For the discretization in time we use the discontinuous Galerkin method with piecewise polynomials in time of order \( k \in \mathbb{N}_0 \). We note that continuous in time (or even continuously differentiable in time) variational discretizations of partial differential equations are known to be more efficient if the number of degrees of freedom in time is measured versus the convergence rate of the time discretization if the underlying basis functions in time and quadrature formulas are chosen properly. For this observation we refer to, e.g. [41, 59, 60]. The superiority of the continuous in time families is then due the fact that some degrees of freedom in time are directly obtained by continuity constraints and do not have to be computed as parts of the algebraic system; cf. [24, 61]. However, if the Stokes or Navier–Stokes system is considered, the application of continuous in time variational discretization becomes more involved. This is due to the fact that no initial value for the pressure variable is given by the mathematical model. However, such initial value is required for the unique definition of the pressure trajectory as long as equal order in time discretizations of the velocity and pressure variable are desired. Applying the discontinuous in time Galerkin method, frees us from an initial value for the pressure. Moreover, stronger stability properties of the discretization are ensured since discontinuous Galerkin methods are known to be strongly \( A \)-stable.

In order to keep this work self-contained, we briefly present a formal derivation of the discontinuous Galerkin discretization for the abstract evolution problem

\[
\partial_t v + A v = f, \quad v(0) = v_0 , \tag{3.10}
\]

as an equality in the dual space \( V^*(t) \) of a time-dependent Hilbert space \( V(t) \) of functions defined on an evolving domain \( \Omega^t \) for \( t \in [0, T] \); cf. [53]. In the formal derivation we tacitly assume that the solution of (3.10) satisfies all the additional conditions that are required such that the arising terms are well-defined. For the derivation of the discontinuous Galerkin method we use the Reynold’s transport theorem that reads as

\[
\frac{d}{dt} \int_{\Omega^t} v \cdot \psi \, dx = \int_{\partial \Omega^t} \partial_t v \cdot \psi \, ds + \int_{\Omega^t} \partial_t (v \cdot \psi) \, dx + \int_{\partial \Omega^t} (v \cdot \psi)(w \cdot n) \, ds . \tag{3.11}
\]

Here, the vector field \( w(x, t) \) is the material velocity of the particles from the boundary \( \partial \Omega^t \), and \( n \) is the outer unit normal vector. Substituting (3.10) into (3.11) yields that

\[
\int_0^T \frac{d}{dt} \langle v, \psi \rangle_{\Omega^t} - \langle v, \partial_t \psi \rangle_{\Omega^t} - \langle v \cdot \psi, w \cdot n \rangle_{\partial \Omega^t} + \langle A v, \psi \rangle_{\Omega^t} \, dt = \int_0^T \langle f, \psi \rangle_{\Omega^t} \, dt . \tag{3.12}
\]
With the fundamental theorem of calculus and for test functions $\psi$ with $\psi(T) = 0$ we get that

$$-\langle v(0), \psi(0) \rangle_{\Omega^0} - \int_0^T \langle v, \partial_t \psi \rangle_{\Omega^t} + \langle v \cdot \psi, w \cdot n \rangle_{\partial \Omega^t} - \langle Av, \psi \rangle_{\Omega^t} \, dt = \int_0^T \langle f, \psi \rangle_{\Omega^t} \, dt. \quad (3.13)$$

Rewriting the integrals in (3.13) as a sum over the subintervals of the time mesh $\mathcal{M}_\tau$, we get that

$$-\langle v(0), \psi(0) \rangle_{\Omega^0} - \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle v, \partial_t \psi \rangle_{\Omega^t} + \langle v \cdot \psi, w \cdot n \rangle_{\partial \Omega^t} - \langle Av, \psi \rangle_{\Omega^t} \, dt = \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle f, \psi \rangle_{\Omega^t} \, dt. \quad (3.14)$$

From (3.11) we derive that

$$-\int_{t_{n-1}}^{t_n} \langle v, \partial_t \psi \rangle_{\Omega^t} \, dt = -\int_{t_{n-1}}^{t_n} \frac{d}{dt} \langle v, \psi \rangle_{\Omega^t} \, dt + \int_{t_{n-1}}^{t_n} \langle \partial_t v, \psi \rangle_{\Omega^t} \, dt + \int_{t_{n-1}}^{t_n} \langle v \cdot \psi, w \cdot n \rangle_{\partial \Omega^t} \, dt.$$ 

Applying the fundamental theorem of calculus to the first term on the right-hand side of this identity and substituting the resulting equation into (3.14) shows that

$$\sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle \partial_t v, \psi \rangle_{\Omega^t} + \langle Av, \psi \rangle_{\Omega^t} \, dt + \sum_{n=1}^{N-1} \left[ \langle v, \psi \rangle_{\Omega^t} \right]_{n} + \langle \psi(0^+), \psi(0) \rangle_{\Omega^0} = \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle \phi; f \rangle_{\Omega^t} \, dt + \langle v_0, \psi(0) \rangle_{\Omega^0}, \quad (3.15)$$

with the jump operator $[\cdot]_n$ at $t_n$, defined as $[g]_n := g(t_n^+) - g(t_n^-)$ with the one-sided limits $g(t_n^+) = \lim_{t \to t_n^+} g(t)$. For a smoothly evolving domain $\Omega^t$, that is assumed here, the one-sided limits of $\Omega^t$ at the time nodes $t_n$ coincide. We note that (3.15) continues to be well-defined for functions that are differentiable piecewise in time (with respect to the time mesh $\mathcal{M}_\tau$) only. Thus, (3.15) can be solved within the space (2.8) of piecewise polynomials in time.

Applying this concept of discontinuous Galerkin time discretization to the semidiscrete Problem 3.2 yields the following fully discrete problem.

**Problem 3.3.** Let $f \in L^2(\Omega_f, I)$ and $v_{0,h} \in V^\text{div}_h$ be given. For $n = 1, \ldots, N$, and given $v_{\tau,h|I_{n-1}} \in P_k(I_{n-1}; V_h)$ for $n > 1$ and $v_{\tau,h|I_{n-1}} := v_{0,h}$ for $n = 1$, find $(v_{\tau,h}, p_{\tau,h}) \in P_k(I_n; V_h) \times P_k(I_n; Q_h)$, such that for all $(\psi_{\tau,h}, \xi_{\tau,h}) \in P_k(I_n; V_h) \times P_k(I_n; Q_h)$,

$$\int_{t_{n-1}}^{t_n} \langle \partial_t v_{\tau,h}, \psi_{\tau,h} \rangle_{\Omega^t} + A_h((v_{\tau,h}, p_{\tau,h}), (\psi_{\tau,h}, \xi_{\tau,h})) \, dt + \langle v_{\tau,h}(t_{n-1}^+), \psi_{\tau,h}(t_{n-1}^+) \rangle_{\Omega^t} = \int_{t_{n-1}}^{t_n} L_h(\psi_{\tau,h}; f, g) \, dt + \langle v_{\tau,h}(t_{n-1}^-), \psi_{\tau,h}(t_{n-1}^-) \rangle_{\Omega^t}. \quad (3.16)$$

**Remark 3.5.**

- In Problem 3.3 a smoothly evolving domain is assumed such that $\Omega^{t_{n-1}^+} = \Omega^{t_{n-1}^-}$ is satisfied. All volume integrals that arise in the forms of Eq. (3.16) are computed over the fluid domain $\Omega^t_f$. 

16
To solve the algebraic counterpart of Eq. (3.16) we use an inexact Newton method. To enhance the range of convergence of the standard Newton method, a linesearch is applied to damp the length of a Newton step. Further, a "dogleg approach" (cf., e.g. [62]), that belongs to the class of trust-region methods and offers the advantage that also the search direction, not just its length, can be adapted to the nonlinear solution process, was implemented. Both schemes require the computation of the Jacobian matrix of the algebraic counterpart of Eq. (3.16). In the dogleg method multiple matrix-vector products with the Jacobian matrix have to be computed. Since the Jacobian matrix is stored as a sparse matrix, this matrix–vector product can be computed at low computational costs. From the point of view of convergence, both methods yield a superlinear convergence behavior. In our numerical examples of Sec. 5 both modifications of Newton’s method lead to comparable results. We did not observe any convergence problems for these nonlinear solvers.

To solve the linear system of the Newton iteration, we use the parallel, sparse direct solver SuperLU_DIST [63]. The development of an efficient geometric multigrid preconditioner for the CutFEM approach is still an ongoing work. For our geometric multigrid preconditioning technique for space-time finite element discretizations of the Navier–Stokes problem on fitted spatial spaces for time-independent domains we refer to [43].

### 3.4 Algebraic in time formulation

In this section we derive a semi-algebraic formulation of Problem 3.3. The algebraic counterpart of Eq. (3.16) is presented with respect to the time variable only. The restriction to the time-discretization is done due to the challenges related to the evolving domain. The presentation of the fully algebraic counterpart of Eq. (3.16) is skipped here for brevity. The transformation of the space discretization into an algebraic form follows the usual steps of finite element methods. For the derivation of the nonlinear algebraic system of a similar higher order space-time finite element approach to the Navier–Stokes system on time-independent domains and the application of Newton’s method for the system’s linearization we refer to [24].

In Problem 3.3 numerical integration in time by the $m$-point Gauss quadrature is still applied. For $g \in \left\{ f : (0, T) \rightarrow L^2(\Omega) \mid f \in C(I_n; L^2(\Omega)), \forall I_n \in M_T \right\}$ this formula is given by

$$Q_n(g) = \sum_{\mu=1}^{m} w_\mu g(t_{n,\mu}) \approx \int_{I_n} g(t) \, dt,$$

where $t_{n,\mu} \in I_n$, for $\mu = \ldots, m$, are the integration points and $w_\mu > 0$ the weights. The number $m \in \mathbb{N}$ of quadrature points, and thereby the order of exactness $2m-1$ the quadrature formula (3.17), is chosen such that $m \geq (3k+1)/2$ is satisfied. Then, the integration in time of the discrete semilinearform $A_h((v_{r,h},p_{r,h}),(\psi_{r,h},\xi_{r,h}))$ on the left-hand side of (3.16) is done exactly. In Fig. 3.4 we illustrate the distribution of the space-time integration points for a one-dimensional setting in space and a single cell. Two quadrature nodes are used for the time domain. This corresponds to the application of the dG(1) scheme in time. Six quadrature nodes are used for spatial integration, which arises in the spatial approximation by piecewise quadratic functions.
Further, we define a temporal basis \( \{ \chi_l \}_{l=0}^k \subset \mathbb{P}_k(I_n; \mathbb{R}) \) by the conditions

\[
\chi_l(t^{GR}_{n,\mu}) = \delta_{l,\mu}, \quad \text{for} \quad l, \mu = 0, \ldots, k,
\]

for the Gauss–Radau quadrature nodes \( t^{GR}_{n,\mu} \) of the subinterval \( I_n \). Expanding the discrete solution \( (v_\tau|_{I_n}, p_\tau|_{I_n}) \in \mathbb{P}_k(I_n; \mathbb{V}_h) \times \mathbb{P}_k(I_n; \mathbb{Q}_h) \) on \( I_n \) in terms of the basis functions yields

\[
v_{\tau,h}|_{I_n}(x,t) = \sum_{l=0}^k v_n,l(x) \chi_l(t) \quad \text{and} \quad p_{\tau,h}|_{I_n}(x,t) = \sum_{l=0}^k p_n,l(x) \chi_l(t),
\]

(3.18)

for \( t \in I_n \) and with coefficient functions \( v_n,l \in \mathbb{V}_h \) and \( p_n,l \in \mathbb{Q}_h \). Appreciable advantage of the usage of the Gauss–Radau quadrature points for the construction of the temporal basis is that the quantity \( v_{\tau,h}|_{I_n-1}(x,t^{n-1}_n) \) in (3.16), that is due to the discontinuous Galerkin scheme, is a degree of freedom of the time discretization and, thus, is available without further computational costs.

Then, Problem 3.3 can be recovered in the following form.

**Problem 3.4.** Let \( f \in L^2(\Omega_{f,I}) \) and \( v_{0,h} \in \mathbb{V}_h^{\text{div}} \) be given. For \( n = 1, \ldots, N \), and given \( v_{\tau,h}|_{I_n-1} \in \mathbb{P}_k(I_{n-1}; \mathbb{V}_h) \) for \( n > 1 \) and \( v_{\tau,h}|_{I_1}(t^{n-1}_n) := v_{0,h} \) for \( n = 1 \), find \( (v_{\tau,h}, p_{\tau,h}) \in \mathbb{P}_k(I_n; \mathbb{V}_h) \times \mathbb{P}_k(I_n; \mathbb{Q}_h) \), such that for all \( j \in \{0 \ldots k\} \) and \( (\psi_h, \xi_h) \in \mathbb{V}_h \times \mathbb{Q}_h \),

\[
\sum_{\mu=1}^m w_\mu \left[ \langle \partial_t v_{\tau,h}|_{I_n}(t_{n,\mu}), \psi_h \chi_j(t_{n,\mu}) \rangle_{\Omega_t} + A_h^j \left( \langle v_{\tau,h}|_{I_n}(t_{n,\mu}), p_{\tau,h}|_{I_n}(t_{n,\mu}) \rangle, (\psi_h \chi_j(t_{n,\mu}), \xi_h \chi_j(t_{n,\mu})) \rangle \right] \right]

+ \langle v_{\tau,h}|_{I_n-1}(t^{n-1}_n), \psi_h \chi_j(t^{n-1}_n) \rangle_{\Omega_{n-1}^t} = \sum_{\mu=1}^m w_\mu \left[ \langle L_h(\psi_h \chi_j(t_{n,\mu}); f, g) \rangle \right] + \langle v_{\tau,h}|_{I_{n-1}}(x,t^{n-1}_n), \psi_h(x) \chi_j(t^{n-1}_n) \rangle_{\Omega_{n-1}^t},
\]

(3.19)

To illustrate the structure of Eq. (3.19), we review the terms in Eq. (3.19) more in detail. Exemplarily, this is done for some of them. Using the expansions given in (3.18), we obtain
that
\[
\langle \partial_t \mathbf{v}_{\tau,h}|_{I_n}(t_{n,\mu}), \psi_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f}^t = \sum_{l=0}^k \langle \mathbf{v}_{n,l} \partial_t \chi_l(t_{n,\mu}), \psi_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f}^t, \tag{3.20}
\]

\[
\langle \nabla \mathbf{v}_{\tau,h}|_{I_n}(t_{n,\mu}), \nabla \psi_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f} = \sum_{l=0}^k \langle \nabla \mathbf{v}_{n,l} \chi_l(t_{n,\mu}), \nabla \psi_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f}, \tag{3.21}
\]

\[
\langle p_{\tau,h}|_{I_n}(t_{n,\mu}), \nabla \cdot \psi_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f} = \sum_{l=0}^k \langle p_{n,l} \chi_l(t_{n,\mu}), \nabla \cdot \psi_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f}. \tag{3.22}
\]

Thus, Problem 3.4 aims at computing the coefficient functions \( \{v_{n,l}, p_{n,l}\}_{l=0}^k \), with \( \{v_{n,l}, p_{n,l}\} \in V_h \times Q_h \) for \( l = 0, \ldots, k \). Clearly, the pair \( \{v_{n,l}, p_{n,l}\} \in V_h \times Q_h \) yields the fully discrete approximation in the Gauss–Radau quadrature node \( t_{n,l} \in I_n \). The coefficient functions are elements of the bulk finite element spaces \( V_h \) and \( Q_h \), respectively, and are thus defined on the computational background mesh \( \mathcal{T}_h \) of \( \Omega \). In the rigid body domain they are defined implicitly by means of the stabilization and extension operator \( S_{F_{\tau}} \) introduced in Eq. (3.6).

4 Implementational aspects

In this section we address key aspects of the implementation of the presented CutFEM higher order space-time approach with arbitrary polynomial degree in time and space. We use a software architecture that is built upon the C++ library deal.II [45] combined with the linear algebra package Trilinos [46] under a Message Passing Interface (MPI) parallelization. All routines for the assembly of the Newton-linearized algebraic system and the linear algebraic solver are parallelized and can be run on standard multicore/processor architectures. Here, the mesh is partitioned into the number of MPI processes and each of these processes owns the cells and corresponding matrix and vector entries of its partition (and some additional overhead). Basically, the simulation can be run with an arbitrary number of MPI processes. For the simulations of this work we distributed exactly one MPI process to one physical CPU core.

Now we firstly address the integration over cut cells which is an important ingredient of the practical realization of the CutFEM approach. Then, key blocks of the code are discussed.

4.1 Integration over cut cells

In Problem 3.3 or 3.4, respectively, the stabilized discrete form \( A^h_{\tau} \), that is defined in (3.8) along with (3.3) and (2.2), evokes the computation of integrals over the fluid domain. Due to the usage of unfitted meshes, integrals over the portions of the cut cells, that are filled with fluid, have thus to be evaluated. Fig. 4.1 illustrates schematically the types of cell intersections that can be induced by the motion of the rigid body. For quadrilateral finite elements, that we use in our implementation, the fluid cell portions can be pentagons, general quadrilaterals or triangles. We recall that we have assumed (cf. Assumption 2.2), that each face of the...
quadrilateral is cut at most once by the boundary of the rigid body (cf. Fig. 4.1). The type of the cut cell is determined by evaluating the levelset function, for \( t \in (0, T) \),

\[
\theta(x, t) \begin{cases} 
< 0 & \forall x \in \Omega^+_t, \\
= 0 & \forall x \in \partial \Omega^+_t, \\
> 0 & \forall x \in \Omega^-_t 
\end{cases}
\]  

(4.1)

in the corners of each cell.

![Types of arising cut cells](a) (b) (c)

Figure 4.1: Types of arising cut cells with gray shaded rigid body and white shaded fluid domain.

In order to integrate over such cut cells we apply iterated one-dimensional numerical integration schemes. This is illustrated in Fig. 4.2 and reads as

\[
\int \int_{K_{\text{fluid}}} f(x_1, x_2) \, d(x_1, x_2) \approx \text{meas}_2(K_{\text{fluid}}) \sum_{\mu_1=1}^{M_1} \sum_{\mu_2=1}^{M_2} \omega_{\mu_1} \omega_{\mu_2} f(x_{1,\mu_1}, x_{2,\mu_2}),
\]  

(4.2)

where \( \omega_{\mu_1} \) and \( x_{1,\mu_1} \), for \( \mu_1 = 1, \ldots, M_1 \), as well as \( \omega_{\mu_2} \) and \( x_{2,\mu_2} \), for \( \mu_2 = 1, \ldots, M_2 \), denote the weights and quadrature nodes of the quadrature formula for the respective coordinate direction and \( K_{\text{fluid}} \) is the fluid portion of the cut cell \( K \). Here, we use the Gauss quadrature formula. The number \( M_1 \) and \( M_2 \) of quadrature points in the coordinate directions, and thus the degree of exactness of the quadrature rules, can be chosen independently of each other, the number \( M_2 \) of quadrature nodes in \( x_2 \) direction can even depend on \( \mu_1 \), i.e., on the quadrature node \( x_{1,\mu_1} \) in \( x_1 \) direction. We note that the integration by numerical quadrature is restricted to the fluid portions of the cut cells by adapting the interval lengths in either coordinate direction. The flexible choice of the degree of exactness of the iterated integration formulas then allows an accurate integration over cut cells. In the implementation, the direction \( x_1 \) and \( x_2 \) of the iterated integration is adapted to the respective element. Precisely, the direction of the largest element face (i.e., longest side in two space dimensions) bounding the fluid portion of the cut cell (cf. Fig. 4.2), is chosen for the outer summation in Eq. 4.2. For this we recall that in this work the assumption was made that the evolving domain and thus the position of the moving boundary \( \Gamma_t^r \) (cf. Fig. 2.1) is prescribed explicitly and, therefore, is computable; cf. Assumption 2.1.

Further, a structured grid of elements aligned to the coordinate lines was supposed (cf. Subsec. 2.3). In our implementation, the quadrature points of each cut cell are computed in each time step of Problem 3.3 before the assembly of the algebraic system is done. In the assembly routine, these points are then used to build a custom quadrature rule using deal.II routines.

Regarding the accuracy of the presented iterated integration we note the following.
Remark 4.1. The numerical results presented in Sec. 5 and [57] show that the space-time convergence behavior of the CutFEM approach is not deteriorated by the application of the iterated numerical integration on the cut cells. This also holds if irregular (tiny) cuts are present.

Remark 4.2. For rigid domains $\Omega_t^r$ with curved boundary, a cut scenario as sketched in Fig. 4.3 can occur. Here, the face $F_3$ of the cell is cut twice. If $\Omega_t$ is time-independent, such a scenario is avoided by using the mesh sufficiently fine. Nevertheless, when the rigid domain $\Omega_t^r$ evolves in time, such cell intersections might arise. In our implementation, it is avoided by the following algorithm. We allow only the types of cell cuts that are illustrated in Fig. 4.2. To ensure the conformity of the arising cuts with these patterns, we evaluate the levelset function (4.1) not only in the corners of each cell, but also in the $n$ Gauss-Lobatto points on each face of the corresponding cell. This is sketched in Fig. 4.3 for $n = 5$, which we use in all of the simulations presented in Sec. 5. If the levelset function coincides in both corners of the face, the values in the interior Gauss-Lobatto nodes of this face are checked to coincide with these value as well. If this condition is fulfilled, it is assumed that this face is not cut. Otherwise, the face is supposed to be cut more than once. For time-independent domains, either a mesh refinement or repositioning of the rigid domain $\Omega_t$ is performed. For evolving domains, a simple algorithm that slightly adjusts the time step size for this single time step in order to avoid such a scenario is applied. We explicitly note that cut scenarios as illustrated in Fig. 4.3 are very unlikely to occur, especially when the mesh size is chosen sufficiently fine.

Remark 4.3. The integration over the boundary $\Gamma_t^r$ is performed using a standard parametrization of the circular rigid domain $\Omega_t^r$. With this we transform the surface integral into a 1D integral that we numerically evaluate using a Gaussian quadrature rule.

4.2 Key code blocks

Here, we describe the key blocks of the implementation of our CutFEM approach. A flowchart of the code is given in Fig. 4.4. After execution of the program and before the loop of the time marching process of Problem 3.3 is started, a nodal Lagrangian time basis $\{\chi_l\}_{l=0}^k$ is computed...
with respect to the \((k+1)\) Gauss–Radau quadrature nodes by solving the corresponding interpolation problem. Actually, this is done on the reference interval \(\hat{I} = [0,1]\), such that \(\{\hat{\chi}_l\}_{l=0}^k\), with \(\hat{\chi}_l \in P_k(\hat{I}; \mathbb{R})\), satisfies
\[
\hat{\chi}_l(\hat{t}_\mu) = \delta_{l,\mu}, \quad l, \mu = 0, \ldots, k,
\]
with the Kronecker symbol \(\delta_{l,\mu}\) and the \((k+1)\) Gauss–Radau quadrature nodes \(\{\hat{t}_\mu\}_{\mu=0}^k\) of \(\hat{I}\). The resulting linear system is solved by using Linear Algebra PACKage (LAPACK) routines. Since the support points are non-equally distributed, the spectral condition number \(\kappa_2\) of the resulting Vandermonde matrix is feasible even for high values of \(k\). This is illustrated in Fig. 4.5. In each subinterval of the time marching process of Problem 3.3, numerical quadrature is applied for the integration in the time and space domain. Firstly, the subinterval and its quadrature nodes are incremented to \(I_n\) and \(t_{n,\mu}\), for \(\mu = 0, \ldots, k\). For the representation of the rigid domain \(\Omega^r_t\) we use the levelset function \(\theta(x, t)\), defined in (4.1). By evaluating this function we identify whether a finite element cell belongs at time \(t_{n,\mu}\) to the set of fluid cells \(T_{h,f}\), rigid cells \(T_{h,r}\) or cut cells \(T_{h,c}\). In subroutines, we mark each cell of the computational
background grid by evaluating the levelset function in the edges of each cell. For cut cells, we pre-compute the quadrature points and weights for the (fluid) volume and surface integrals (line integrals in two space dimensions) and store them into lookup tables. In the MPI based implementation these tasks scale perfectly with the number of available processes.

**Remark 4.4.** We note that the levelset function (4.1) is not discretized or computed by the approximation of a suitable transport equation. The direct evaluation of (4.1) becomes feasible due to the Assumption 2.1 that the motion of the rigid body is prescribed. Thereby, further discretization errors are avoided.

However, the parallelized code needs some mechanism to prevent that the ghost penalty operator defined in Eq. (3.6) is applied twice in the interface zone of MPI process boundaries of the mesh partition. For this we use a simple master-slave approach and let the process with the higher process number assemble the ghost penalty operator over patches $\omega_F$ at process partition boundaries. This problem is sketched in Fig. 4.6 for the patch $\omega_F$, built from the cells $K_1$ and $K_2$. Cell $K_1$ of the patch belongs to process 0 and cell 2 to process 1. Without a control of the parallel assembly of the ghost penalty stabilization, both processes would assemble the contributions of Eq. (3.6), such that the stabilization would be applied twice. To avoid this, the process with the higher process number (process 1 in this case) assembles the ghost penalty stabilization.

In the next block the linear system of the Newton iteration (cf. Remark 3.5) is assembled. For the integration over fluid cut cells iterated integration, as sketched in Subsec. 4.1, is applied. Then, the resulting linear system is solved by the parallel, sparse direct solver SuperLU_DIST [63] and, unless the stopping criteria is satisfied, the Newton iteration is continued by reassembling the linear system. After convergence of the Newton iteration, a parallel assembly and data output of the computed higher order solution for the subinterval $I_n$ terminates the time step.
Figure 4.6: Assembly of the ghost penalty stabilization at partition boundaries with computational mesh $\mathcal{T}_b$ and colored processor ownerships of mesh cells.

5 Numerical results

In this section we present the results of four numerical experiments to illustrate the performance and accuracy properties of the proposed CutFEM approach. We start with a space-time convergence test with a moving $\Omega^\epsilon_t$, to validate all aspects of the implementation. In the second experiment, the popular DFG benchmark of flow around a cylinder (cf. [44]) is considered. A careful comparative study of the CutFEM approach on background meshes with the results obtained for body fitted meshes is done. In the last two test cases we investigate the quality of the ghost penalty stabilization and extension of (3.6) on an evolving domain and demonstrate its application in a realistic example. Throughout all simulations we set the viscosity to $\nu = 0.001$.

In all the examples, the Newton solver uses an absolute value of the residual smaller than $1 \times 10^{-10}$ as the stopping criteria. In Sec. 5.2 it is checked additionally if the initial residual of a time step is decreased by a factor of 1000. For all our numerical experiments we set the Nitsche penalty parameters of Eq. (3.2) to $\gamma_1 = \gamma_2 = 35$, which is motivated by the numerical experiments in [6, 4]. The ghost penalty parameters of Eq. (3.6) are set to $\gamma_v = \gamma_p = 10^{-2}$, except in Sec. 5.4 where we set $\gamma_v = \gamma_p = 1$.

5.1 Experimental order of convergence

The aim of this first numerical experiment is to verify all components of our approach, in particular the integration over cut cells as well as the ghost penalty stabilization, by a numerical space-time convergence study. We consider the problem setting sketched in Fig. 5.1a with a time-dependent domain $\Omega^\epsilon_t$. We put $\Omega \times I = (0, 1)^2 \times (0, 1]$. The rigid body is modeled by a circular domain of radius of $r = 0.1$ and an initial position of its center at $\mathbf{x}_r(0) = (0.5, 0.5)^\top$. The motion of the center is prescribed by

$$\mathbf{x}_r(t) = \mathbf{x}_r(0) + (A \cdot \sin(\omega \cdot t), 0)^\top,$$

(5.1)
with $A = 0.2$ and $\omega = 1$. We prescribe the initial value $v_0$ and right-hand side function $f$ on $\Omega_f \times I$ in such a way, that the solution of the Navier–Stokes system on $\Omega_f$ is given by

$$v_e(x, t) := \left( \cos(x_2 \pi) \cdot \sin(t) \cdot \sin(x_1 \pi)^2 \cdot \sin(x_2 \pi), \right.$$

$$\left. - \cos(x_1 \pi) \cdot \sin(t) \cdot \sin(x_2 \pi)^2 \cdot \sin(x_1 \pi) \right),$$

$$p_e(x, t) := \cos(x_2 \pi) \cdot \sin(t) \cdot \sin(x_1 \pi) \cdot \cos(x_1 \pi) \cdot \sin(x_2 \pi).$$

On the inner fluid boundary we prescribe the condition $g_r = v_e$. On the outer boundary we use a homogeneous Dirichlet condition. Table 5.1 shows the computed errors and experimental orders of convergence (EOC) for two different combinations of space-time finite elements, based on the Taylor–Hood family in space. In the first experiment, the polynomial order in time is adapted to the spatial approximation of the pressure variable. It is chosen to $k = 1$ such that optimal second order of convergence in time and space is obtained for the pressure variable with discrete values in $H^1$. In the second experiment, the polynomial order in time is put to $k = 2$ such that optimal third order of convergence in time and space is obtained for the pressure variable with discrete values in $H^2 \times H^2$. For this we recall that a non-equal order, inf-sup stable discretization in space by the Taylor–Hood family of elements is used here. For the definition of the discrete function spaces we refer to (2.6) and (2.7), respectively. In Table 5.1, the expected convergence of optimal order is documented for both experiments.

![Figure 5.1: Problem setup and pressure field at the end of the simulation.](image)

Table 5.1: Errors and experimental order of convergence for $\tau_0 = 1.0$ and $h_0 = 1/(2\sqrt{2})$.

| $\tau$ | $h$ | $\|e^v\|_{L^2(L^2)}$ EOC | $\|e^p\|_{L^2(L^2)}$ EOC | $\|e^v\|_{L^2(L^2)}$ EOC | $\|e^p\|_{L^2(L^2)}$ EOC |
|--------|-----|-------------------------|-------------------------|-------------------------|-------------------------|
| $\tau_0/2^0$ | $h_0/2^0$ | 4.314e–2 | - | 1.476e–2 | - | 3.420e–2 | - | 1.340e–2 | - |
| $\tau_0/2^1$ | $h_0/2^1$ | 8.602e–3 | 2.33 | 3.047e–3 | 2.28 | 7.833e–3 | 2.13 | 2.590e–3 | 2.37 |
| $\tau_0/2^2$ | $h_0/2^2$ | 2.063e–3 | 2.06 | 6.964e–4 | 2.13 | 9.459e–4 | 3.05 | 5.605e–4 | 2.21 |
| $\tau_0/2^3$ | $h_0/2^3$ | 4.882e–3 | 2.08 | 1.853e–4 | 1.91 | 1.151e–4 | 3.04 | 1.354e–4 | 2.05 |
| $\tau_0/2^4$ | $h_0/2^4$ | 1.221e–4 | 2.00 | 4.600e–5 | 2.01 | 1.428e–5 | 3.01 | 3.385e–5 | 2.00 |

Remark 5.1. The setting of this example was chosen in such a way, that the propagation of the boundary of $\Gamma^t_r$ is less than one cell in each time step, in order to preserve a CFL like...
condition. We didn’t observe any decrease in the convergence rate by violating this condition. For instance, choosing $h_0 = 1/(4\sqrt{2})$, leads to comparable EOCs, but violates the assumption that the boundary of $\Gamma_t^+$ propagates over a distance less than one cell within a time step.

5.2 Time periodic flow around a cylinder

The aim of the second numerical example is to analyze the effects of usage of cut cells for the space discretization on background meshes along with the extension to a rigid domain $\Omega_r$ and the application of the stabilization introduced in (3.6). For this, we use the well-known DFG benchmark setting of flow around a cylinder, defined in [44]. Although the domain is non-evolving, evaluating the performance properties of the approach for this flow benchmark is of high interest. We compare the numerical results with the ones obtained for simulations on a body-fitted meshes that is highly pre-adapted to the cylinder. Quantities of interest and comparison in the simulations are the drag and lift coefficient of the flow on the circular cross section (cf. [44]). With the drag and lift forces $F_D$ and $F_L$ on the rigid circle $S$ given by

$$F_D = \int_S \left( \nu \frac{\partial v_t}{\partial n} n_y - P n_x \right) dS,$$
$$F_L = -\int_S \left( \nu \frac{\partial v_t}{\partial n} n_x - P n_y \right) dS,$$  (5.2)

where $n$ is the normal vector on $S$ and $v_t$ is the tangential velocity $t = (n_y, -n_x)^\top$, the drag and lift coefficient $c_D, c_L$ are defined by means of

$$c_D = \frac{2}{U^2 L} F_D,$$
$$c_L = \frac{2}{U^2 L} F_L.$$  (5.3)

According to [44], we set the boundary condition on the inflow boundary $\Gamma_i$ as $g_i(x, y, t) = \left( \frac{4 \cdot 1.5 \cdot y(0.41-y)}{0.41^2}, 0 \right)^\top$. This leads to a Reynold’s number of $Re = 100$ and a time-periodic flow behavior. The space–time discretization is done in the discrete spaces $P_1(I_n; H_2^h)^2 \times P_1(I_n; H_1^h)$ on each time interval $I_n$. For the CutFEM approach we study a sequence of successive mesh refinements in space. For the developed flow and the computation of the drag and lift coefficient as quantities of physical interest we use the time step size $\tau = 5e^{-3}$.

Figure 5.2: Portion of problem setting and computed velocity profile for CutFEM approach (Fig. 5.2a) and fitted mesh approximation (Fig. 5.2b).

Fig. 5.2 shows the computed flow profile and spatial mesh of a CutFEM simulation and a computation done on a highly pre-adapted, body-fitted mesh. In the latter case, high accuracy
for the drag and lift coefficient is obtained (cf. [44]). The CutFEM simulation is done for a sequence of successive refinement steps of the entire background mesh (cf. Table 5.2). Table 5.2 shows the computed drag and lift coefficient as well as their frequency. Precisely, the latter denotes the frequency of the oscillation of the lift coefficient \(c_L\), which is computed at a time instant \(t_0\) where the lift coefficient \(c_L\) is smallest and ends at a time instant \(t_1 = t_0 + \frac{1}{f}\) when \(c_L\) is smallest again. Exemplarily, Fig. 5.3 illustrates such a cycle for the drag and the lift coefficient. To simplify the interpretation of the graphs, the starting point of the monitoring cycle is set to \(t = 0\).

Table 5.2: Computed minimum and maximum drag–lift coefficients and frequency for a sequence of successively refined CutFEM background meshes and an adapted, body-fitted mesh.

| Method         | DoFs | \(\min(c_L)\) | \(\max(c_L)\) | \(\min(c_D)\) | \(\max(c_D)\) | \(f_L\) |
|----------------|------|----------------|----------------|----------------|----------------|--------|
| cut\(_4\)      | 24000| -0.8814        | 0.9147         | 3.0507         | 3.1157         | 2.8433 |
| cut\(_5\)      | 94086| -0.9127        | 0.7626         | 3.0308         | 3.0810         | 3.0146 |
| cut\(_6\)      | 372486| -0.9759        | 0.9101         | 3.0848         | 3.1391         | 3.0274 |
| cut\(_7\)      | 1482246| -1.0177        | 0.9856         | 3.1638         | 3.2272         | 3.0183 |
| Adapted, body-fitted | 502464| -1.0190        | 0.9843         | 3.1625         | 3.2259         | 3.0184 |

Comparing the results of the either approaches, CutFEM on background meshes versus adapted body-fitted mesh, we observe that the goal quantities of the CutFEM simulations nicely converge to the results computed on a manually pre-adapted and body-fitted mesh. Whereas the CutFEM background mesh is refined uniformly, the body-fitted mesh is highly adapted to the problem setting and flow profile. In particular, the region around the cylinder is strongly resolved by the spatial mesh whereas the CutFEM mesh remains much coarser in this region, even on the finest refinement level. In Fig. 5.2 both meshes are shown. The diameter of a quadratic cut cell of the Cut\(_7\) configuration is \(4.530e^{-3}\). The diameter of a rectangular cell of the body-fitted triangulation in the neighborhood of \(\Omega_r\) is \(2.495e^{-3}\). Therefore, the superiority of the body-fitted approach for this test case with non-evolving domain is obvious. Nevertheless, Table 5.2 and Fig. 5.3 nicely show the numerical convergence of the CutFEM simulations for a successive refinement of the background mesh to the expected solution in terms of the confirmed drag and lift coefficient and their frequency (cf. [44]). This confirms the accuracy of the proposed CutFEM approach in the case of non-evolving domains.

### 5.3 Dynamic Poiseuille flow

In this numerical experiment the accuracy of the presented CutFEM approach is analyzed for an evolving domain. Precisely, we study the perturbation of a Poiseuille flow profile by the motion of an enclosed moving rigid body (cf. Fig. 2.1) and whether application of the ghost penalty operator leads to a disturbance of the solution. The test configuration is chosen in such a way, that the Poiseuille profile is preserved in the pipe, even though a moving rigid body is present. We than analyze if the Poiseuille flow is recovered by the CutFEM approach.
Figure 5.3: Computed drag and lift coefficients for the example of Sec. 5.2 and different time discretization schemes with basic time step size $\tau = 5e^{-3}$.

The computational domain is $\Omega = (0, l_x) \times (-l_y, l_y)$, with $l_x = 3, l_y = 0.5$, we let $I = (0, 40]$. A parabolic Poiseuille flow profile is prescribed at the left inflow boundary, that is given by

$$v_p(x, y) = U_{in}(l_y^2 - y^2, 0) \top$$

with $U_{in} = 1$. The initial value is $v_0 = 0$. The extension of (5.4) to the entire pipe is the Poiseuille flow and satisfies the Navier–Stokes equations along with the pressure $p_p = -2\nu U_{in}(x - l_x)$. The rigid body is modeled by a circular domain of radius of $r = 0.2$ and an initial position of its center at $x_r(0) = (1.545, 0) \top$. The motion of the ball’s center is prescribed by eq. (5.1), with $A = 0.8$ and $\omega = 0.2$. On the boundary of the rigid body the Dirichlet condition

$$v = g_r := v_p$$

is prescribed. After a transition from the zero initial state, the Poiseuille flow profile is expected to develop, and it is not perturbed by the rigid body motion due to the choice of the boundary condition (5.5). Clearly, even though the rigid body is moving inside the fluid by means of (5.1), its impact on the fluid flow is hidden by means of the boundary condition (5.5).

We compute fully discrete approximations in $(\mathbb{P}_1(I_n; H^2_h))^2 \times \mathbb{P}_1(I_n; H^1_h)$, for $n = 1, \ldots, N$, for the time step size $\tau = 0.1$ and a background mesh with 223 750 space-time degrees of freedom in each time step. The mesh and the computed flow profile at the final simulation
time \( T = 40 \) are illustrated in Fig. 5.4. This solution is fully converged. In fact, the Poiseuille profile is nicely recovered by the simulation. Fig. 5.5 shows the cross sections plots in \( y \) and \( x \) coordinate directions, respectively, of the flow and pressure profile computed by CutFEM. The position of the rigid body domain is gray-shaded. In this domain the velocity and pressure values are computed by the extension and ghost penalty stabilization operator defined in (3.6). The computed profiles and the Poiseuille profile nicely coincide which clearly demonstrates the accuracy of the suggested CutFEM approach and efficiency of the combined extension and stabilization (3.6).

Figure 5.5: Cross section plots of velocity in \( y \)-direction and pressure in \( x \)-direction and Poiseuille profile at \( T = 40 \) with gray-shaded rigid body domain.

5.4 Dynamic flow around moving cylinder

In the last numerical example we illustrate the stability and performance properties of the proposed CutFEM for a problem of higher interest in practice. Precisely, we simulate dynamic
flow around a moving (two-dimensional) ball. The background domain is $\Omega = (0, 3) \times (0, 1)$, and we let $I = (0, 29)$. At the left inflow boundary we prescribe the parabolic inflow profile

$$g_i(x, t) = \begin{cases} (6ty(1-y), 0)^T, & t \leq 1, \\ (6y(1-y), 0)^T, & t > 1. \end{cases} \quad (5.6)$$

For the rigid body, a ball with a radius $r = 0.2$ and an initial position of its center at $x_r(0) = (1.545, 0.6)^T$ is chosen. For a non-moving ball, the flow setting would result in a Reynolds number of $Re = 400$. The motion of the ball’s center is prescribed by eq. (5.1), with $A = 0.8$ and $\omega = 0.5$. On the boundary $\Gamma^k$ of the ball, the no-slip condition $v = v_r = (A\omega \cos(\omega t), 0)^T$ for a moving, rigid domain is applied, such that the fluid and ball velocity coincide on their interface, cf. [64, p. 47]. We use the time step size $\tau = 0.01$ and a uniform structured background mesh with $h = \frac{1}{64\sqrt{2}}$, which results in 889,862 degrees of freedom in each time interval. With this setup the rigid interface crosses at max one spatial mesh cell. Discrete solutions are computed in $((P_1(I_n; H^2_h))^2 \times P_1(I_n; H^1_h))$. Fig. 5.6 and Fig. 5.7 illustrate the computed profiles of velocity and the pressure at time $t = 10.12$ and $t = T$, respectively. The figures show that by the application of the ghost penalty stabilization and extension of the discrete solution to the ghost rigid domain, defined by (3.6), lead to stable approximations of the flow problem on a background mesh. The color legend of Fig. 5.7 shows that reasonable values for the velocity and pressure variable are obtained in the ghost subdomain of the rigid ball. We note that all kind of cuts (cf. Subsec. 4.1), including irregular ones with small portions, arise in the simulation. Unphysical oscillations that are due to irregular cuts of finite elements or insufficient extensions to the rigid domain, are strongly reduced and do not perturb the ambient fluid flow.
6 Summary and outlook

In this work a CutFEM approach for the numerical simulation of fluid flow on evolving domain was presented. The physical domain was embedded into a fixed computational background
mesh. Discontinuous Galerkin methods and inf-sup stable pairs of finite element spaces were used for the discretization of the time and space variables, respectively. Ghost penalty stabilization for the treatment of irregular cuts along with an extension of the discrete solution to the ghost rigid domain was suggested further. The numerical performance properties of the parallel implementation of the family of schemes were illustrated. Thereby, the accuracy and stability of the approach was confirmed. Here, a parallel direct linear solver was applied for the sake of simplicity. The integration of cuts cells into our geometric multigrid preconditioner 43 for GMRES iterations for solving the Newton linearized Navier–Stokes is an ongoing work. In this work, the motion of the rigid body was prescribed explicitly. For the future, more sophisticated applications in that the motion of the body is no longer prescribed explicitly (cf., e.g., 28) such that the interface has to be computed simultaneously or is given by, for instance fluid-structure interaction, are also in the scope of our interest.

References

[1] C. Lehrenfeld and M. Olshanskii, “An Eulerian finite element method for PDEs in time-dependent domains,” ESAIM: Mathematical Modelling and Numerical Analysis, vol. 53, pp. 585–614, Mar. 2019.

[2] H. von Wahl, T. Richter, and C. Lehrenfeld, “An unfitted Eulerian finite element method for the time-dependent Stokes problem on moving domains,” preprint, Feb. 2020.

[3] E. Burman, S. Frei, and A. Massing, “Eulerian time-stepping schemes for the non-stationary Stokes equations on time-dependent domains,” Dec. 2020.

[4] M. Winter, B. Schott, A. Massing, and W. Wall, “A Nitsche cut finite element method for the Oseen problem with general Navier boundary conditions,” Computer Methods in Applied Mechanics and Engineering, vol. 330, pp. 220–252, Mar. 2018.

[5] E. Burman and P. Hansbo, “Fictitious domain methods using cut elements: III. A stabilized Nitsche method for Stokes’ problem,” ESAIM: Mathematical Modelling and Numerical Analysis, vol. 48, pp. 859–874, May 2014.

[6] B. Schott and W. Wall, “A new face-oriented stabilized XFEM approach for 2D and 3D incompressible Navier–Stokes equations,” Computer Methods in Applied Mechanics and Engineering, vol. 276, pp. 233–265, July 2014.

[7] S. Court, M. Fournié, and A. Lozinski, “A fictitious domain approach for the Stokes problem based on the extended finite element method,” International Journal for Numerical Methods in Fluids, vol. 74, pp. 73–99, 2014.

[8] B. Schott, C. Ager, and W. Wall, “A monolithic approach to fluid-structure interaction based on a hybrid Eulerian-ALE fluid domain decomposition involving cut elements,” International Journal for Numerical Methods in Engineering, vol. 119, pp. 208–237, July 2019.

[9] S. Court and M. Fournié, “A fictitious domain finite element method for simulations of fluid–structure interactions: The Navier–Stokes equations coupled with a moving solid,” Journal of Fluids and Structures, vol. 55, pp. 398–408, May 2015.
[10] E. Burman, M. A. Fernández, and S. Frei, “A Nitsche-based formulation for fluid-structure interactions with contact,” *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 54, pp. 531–564, Mar. 2020.

[11] T. J. Hughes, W. K. Liu, and T. K. Zimmermann, “Lagrangian-Eulerian finite element formulation for incompressible viscous flows,” *Computer Methods in Applied Mechanics and Engineering*, vol. 29, pp. 329–349, Dec. 1981.

[12] J. Donea, S. Giuliani, and J. Halleux, “An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions,” *Computer Methods in Applied Mechanics and Engineering*, vol. 33, pp. 689–723, Sept. 1982.

[13] H. Braess and P. Wriggers, “Arbitrary Lagrangian Eulerian finite element analysis of free surface flow,” *Computer Methods in Applied Mechanics and Engineering*, vol. 190, pp. 95–109, Oct. 2000.

[14] W. A. Wall, A. Gerstenberger, P. Gamnitzer, C. Forster, and E. Ramm, “Large Deformation Fluid-Structure Interaction – Advances in ALE Methods and New Fixed Grid Approaches,” in *Fluid-Structure Interaction* (H.-J. Bungartz and M. Schäfer, eds.), vol. 53, pp. 195–232, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[15] M. Nazem, D. Sheng, and J. P. Carter, “Stress integration and mesh refinement for large deformation in geomechanics,” *International Journal for Numerical Methods in Engineering*, vol. 65, pp. 1002–1027, Feb. 2006.

[16] J. L. Farinatti Aymone, “Mesh motion techniques for the ALE formulation in 3D large deformation problems: MESH MOTION TECHNIQUES,” *International Journal for Numerical Methods in Engineering*, vol. 59, pp. 1879–1908, Apr. 2004.

[17] A. Brenner, E. Bänsch, and M. Bause, “A priori error analysis for finite element approximations of the Stokes problem on dynamic meshes,” *IMA Journal of Numerical Analysis*, vol. 34, pp. 123–146, Jan. 2014.

[18] J. Baiges and R. Codina, “The fixed-mesh ALE approach applied to solid mechanics and fluid–structure interaction problems,” *International Journal for Numerical Methods in Engineering*, vol. 81, no. 12, pp. 1529–1557, 2010.

[19] J. Nitsche, “Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind,” *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, vol. 36, pp. 9–15, July 1971.

[20] E. Burman and P. Hansbo, “Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method,” *Applied Numerical Mathematics*, vol. 62, pp. 328–341, Apr. 2012.

[21] R. Becker, “Mesh adaptation for Dirichlet flow control via Nitsche’s method,” *Communications in Numerical Methods in Engineering*, vol. 18, no. 9, pp. 669–680, 2002.

[22] T. Boiveau and E. Burman, “A penalty-free Nitsche method for the weak imposition of boundary conditions in compressible and incompressible elasticity,” *IMA Journal of Numerical Analysis*, vol. 36, pp. 770–795, Apr. 2016.
A. Massing, M. G. Larson, A. Logg, and M. E. Rognes, “A Stabilized Nitsche Fictitious Domain Method for the Stokes Problem,” *Journal of Scientific Computing*, vol. 61, pp. 604–628, Dec. 2014.

M. Anselmann and M. Bause, “Higher order Galerkin–collocation time discretization with Nitsche’s method for the Navier–Stokes equations,” *Mathematics and Computers in Simulation*, p. S0378475420303827, Nov. 2020.

H. Xie, Z. Li, and Z. Qiao, “A Finite Element Method for Elasticity Interface Problems with Locally Modified Triangulations,” *International Journal of Numerical Analysis and Modeling*, vol. 8, no. 2, pp. 189–200, 2011.

E. S. Gawlik and A. J. Lew, “High-order finite element methods for moving boundary problems with prescribed boundary evolution,” *Computer Methods in Applied Mechanics and Engineering*, vol. 278, pp. 314–346, Aug. 2014.

T. Carraro and S. Wetterauer, “On the implementation of the eXtended Finite Element Method (XFEM) for interface problems,” *Archive of Numerical Software*, vol. 4, no. 2, pp. 1–23, 2016.

S. Frei, T. Richter, and T. Wick, “LocModFE: Locally modified finite elements for approximating interface problems in deal.II,” *Software Impacts*, vol. 8, p. 100070, May 2021.

A. Massing, M. G. Larson, and A. Logg, “Efficient Implementation of Finite Element Methods on Nonmatching and Overlapping Meshes in Three Dimensions,” *SIAM Journal on Scientific Computing*, vol. 35, pp. C23–C47, Jan. 2013.

Y. Sudhakar, J. Moitinho de Almeida, and W. A. Wall, “An accurate, robust, and easy-to-implement method for integration over arbitrary polyhedra: Application to embedded interface methods,” *Journal of Computational Physics*, vol. 273, pp. 393–415, Sept. 2014.

E. Burman and M. A. Fernández, “Continuous interior penalty finite element method for the time-dependent Navier–Stokes equations: Space discretization and convergence,” *Numerische Mathematik*, vol. 107, pp. 39–77, July 2007.

H.-G. Roos, M. Stynes, L. Tobiska, and H.-G. Roos, *Robust Numerical Methods for Singularly Perturbed Differential Equations: Convection-Diffusion-Reaction and Flow Problems*. No. 24 in Springer Series in Computational Mathematics, Berlin: Springer-Verlag, 2nd ed ed., 2008.

E. Burman, “Ghost penalty,” *Comptes Rendus Mathematique*, vol. 348, pp. 1217–1220, Nov. 2010.

E. Burman and M. A. Fernández, “An unfitted Nitsche method for incompressible fluid–structure interaction using overlapping meshes,” *Computer Methods in Applied Mechanics and Engineering*, vol. 279, pp. 497–514, Sept. 2014.

P. Hansbo, M. G. Larson, and S. Zahedi, “A cut finite element method for coupled bulk-surface problems on time-dependent domains,” *Computer Methods in Applied Mechanics and Engineering*, vol. 307, pp. 96–116, Aug. 2016.
[36] E. Burman, D. Elfverson, P. Hansbo, M. G. Larson, and K. Larsson, “A cut finite element method for the Bernoulli free boundary value problem,” *Computer Methods in Applied Mechanics and Engineering*, vol. 317, pp. 598–618, Apr. 2017.

[37] T. Frachon and S. Zahedi, “A cut finite element method for incompressible two-phase Navier–Stokes flows,” *Journal of Computational Physics*, vol. 384, pp. 77–98, May 2019.

[38] E. Burman, S. Claus, P. Hansbo, M. G. Larson, and A. Massing, “CutFEM: Discretizing geometry and partial differential equations,” *International Journal for Numerical Methods in Engineering*, vol. 104, pp. 472–501, Nov. 2015.

[39] J. Preuß, *Higher Order Unfitted Isoparametric Space-Time FEM on Moving Domains*. MA thesis, University of Göttingen, Feb. 2018.

[40] S. Zahedi, “A Space-Time Cut Finite Element Method with Quadrature in Time,” in *Geometrically Unfitted Finite Element Methods and Applications* (S. P. A. Bordas, E. Burman, M. G. Larson, and M. A. Olshanskii, eds.), vol. 121, pp. 281–306, Cham: Springer International Publishing, 2017.

[41] S. Hussain, F. Schieweck, and S. Turek, “An efficient and stable finite element solver of higher order in space and time for nonstationary incompressible flow,” *International Journal for Numerical Methods in Fluids*, vol. 73, pp. 927–952, Dec. 2013.

[42] V. John and L. Tobiska, “Numerical performance of smoothers in coupled multigrid methods for the parallel solution of the incompressible Navier–Stokes equations,” *International Journal for Numerical Methods in Fluids*, vol. 33, no. 4, pp. 453–473, 2000.

[43] M. Anselmann and M. Bause, “A geometric multigrid method for space-time finite element discretizations of the Navier-Stokes equations and its application to 3d flow simulation,” submitted, arXiv:2107.10561, July 2021.

[44] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher, “Benchmark Computations of Laminar Flow Around a Cylinder,” in *Flow Simulation with High-Performance Computers II* (E. H. Hirschel, K. Fujii, B. van Leer, M. A. Leschziner, M. Pandolfi, A. Rizzi, B. Roux, and E. H. Hirschel, eds.), vol. 48, pp. 547–566, Wiesbaden: Vieweg+Teubner Verlag, 1996.

[45] D. Arndt, W. Bangerth, B. Blais, T. C. Clevenger, M. Fehling, A. V. Grayver, T. Heister, L. Heltai, M. Kronbichler, M. Maier, P. Munch, J.-P. Pelteret, R. Rastak, I. Tomas, B. Turcksin, Z. Wang, and D. Wells, “The deal.II library, Version 9.2,” *Journal of Numerical Mathematics*, vol. 28, pp. 131–146, Sept. 2020.

[46] The Trilinos Project Team, *The Trilinos Project Website*, July 2020.

[47] J. G. Heywood, R. Rannacher, and S. Turek, “Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations,” *International Journal for Numerical Methods in Fluids*, vol. 22, no. 5, pp. 325–352, 1996.

[48] R. Salvi, “On the Navier-Stokes equations in non-cylindrical domains: On the existence and regularity,” *Mathematische Zeitschrift*, vol. 199, pp. 153–170, June 1988.

[49] D. N. Bock, “On the Navier-Stokes equations in noncylindrical domains,” *Journal of Differential Equations*, vol. 25, pp. 151–162, Aug. 1977.
[50] J. G. Heywood and R. Rannacher, “Finite Element Approximation of the Nonstationary Navier–Stokes Problem. I. Regularity of Solutions and Second-Order Error Estimates for Spatial Discretization,” SIAM Journal on Numerical Analysis, vol. 19, pp. 275–311, Apr. 1982.

[51] M. Bause, “On optimal convergence rates for higher-order Navier–Stokes approximations. I. Error estimates for the spatial discretization,” IMA Journal of Numerical Analysis, vol. 25, pp. 812–841, Oct. 2005.

[52] F. Sonner and T. Richter, “Second Order Pressure Estimates for the Crank–Nicolson Discretization of the Incompressible Navier–Stokes Equations,” SIAM Journal on Numerical Analysis, vol. 58, pp. 375–409, Jan. 2020.

[53] A. Alphonse, C. Elliott, and B. Stinner, “An abstract framework for parabolic PDEs on evolving spaces,” Portugaliae Mathematica, vol. 72, no. 1, pp. 1–46, 2015.

[54] V. John, Finite Element Methods for Incompressible Flow Problems. New York, NY: Springer Berlin Heidelberg, 2016.

[55] J. Benk, M. Ulbrich, and M. Mehl, “The Nitsche Method of the Navier Stokes Equations for Immersed and Moving Boundaries,” in Seventh International Conference on Computational Fluid Dynamics, 2012.

[56] C. Ager, B. Schott, M. Winter, and W. Wall, “A Nitsche-based cut finite element method for the coupling of incompressible fluid flow with poroelasticity,” Computer Methods in Applied Mechanics and Engineering, vol. 351, pp. 253–280, July 2019.

[57] M. Anselmann and M. Bause, “Numerical convergence of discrete extensions in a space-time finite element, fictitious domain method for the Navier-Stokes equations,” submitted to PAMM, July 2021.

[58] B. Schott, C. Ager, and W. A. Wall, “Monolithic cut finite element–based approaches for fluid-structure interaction,” International Journal for Numerical Methods in Engineering, vol. 119, no. 8, pp. 757–796, 2019.

[59] S. Hussain, F. Schieweck, and S. Turek, “Higher order Galerkin time discretizations and fast multigrid solvers for the heat equation,” Journal of Numerical Mathematics, vol. 19, Jan. 2011.

[60] U. Köcher and M. Bause, “Variational Space–Time Methods for the Wave Equation,” Journal of Scientific Computing, vol. 61, pp. 424–453, Nov. 2014.

[61] M. Anselmann and M. Bause, “Numerical Study of Galerkin–Collocation Approximation in Time for the Wave Equation,” in Mathematics of Wave Phenomena (W. Dörfler, M. Hochbruck, D. Hundertmark, W. Reichel, A. Rieder, R. Schnaubelt, and B. Schörkhuber, eds.), pp. 15–36, Cham: Springer International Publishing, 2020.

[62] R. P. Pawlowski, J. P. Simonis, H. F. Walker, and J. N. Shadid, “Inexact Newton Dogleg Methods,” SIAM Journal on Numerical Analysis, vol. 46, pp. 2112–2132, Jan. 2008.

[63] X. S. Li and J. W. Demmel, “SuperLU_DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems,” ACM Transactions on Mathematical Software, vol. 29, pp. 110–140, June 2003.
[64] T. Richter, *Fluid-Structure Interactions: Models, Analysis and Finite Elements*. No. 118 in Lecture Notes in Computational Science and Engineering, Cham: Springer, 2017.