

Cut finite element methods and ghost stabilization techniques for space-time discretizations of the Navier–Stokes equations

Mathias Anselmann | Markus Bause

Faculty of Mechanical Engineering,
Helmut Schmidt University, Hamburg,
Germany

Correspondence

Mathias Anselmann, Faculty of
Mechanical Engineering, Helmut Schmidt
University, Holstenhofweg 85, 22043
Hamburg, Germany.
Email: anselmann@hsu-hh.de

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.

KEYWORDS

discontinuous Galerkin, finite element, immersed boundary, incompressible flow, Navier–Stokes, partial differential equations

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; compare for example, Reference 1. This analysis was recently extended to the time-dependent Stokes problem using, for the spatial discretization, lowest order Taylor-Hood elements² or equal-order elements along with stabilization,³ combined with BDF time-stepping schemes. For instance, we refer to Reference 2 (Thm. 5.26) for a rigorous error estimate. Also, the simulation of incompressible flow on complex domains⁴⁻⁷ or even the simulation of coupled problems of multi-physics, for instance of fluid-structure interaction, with moving interfaces⁸⁻¹⁰ 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. References 11-13), 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., References 14-16), 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. Reference 17). Fixed-mesh ALE approaches (cf. Reference 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: First, boundary conditions of Dirichlet type are imposed in a weak variational form by using Nitsche's method, compare References 19-24. Second, 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; compare References 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. References 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. Reference 29) or suitable quadrature formulas (cf. Reference 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; compare References 1-3,5,31. 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. Reference 32) can be observed. Further, the stabilization also aims at improving the conditioning of the resulting algebraic system; compare References 1,23,33. Proposed stabilization techniques are based on suitable extensions of the discrete functions to some neighborhood of the flow domain; compare References 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

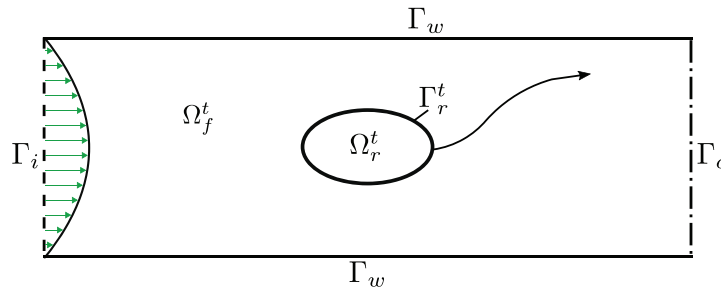


FIGURE 1 Problem setting and corresponding notation with pipe's boundary $\Gamma_p := \Gamma_i \cup \Gamma_w \cup \Gamma_o$ [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/doi/10.1002/nd.5074)]

prolongation of the discrete functions is restricted to some thin layer adjacent to the domain's boundary; compare References 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. Figure 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, for example, References 5,8,35,37,38. 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 References 1,39 for linear convection-diffusion-reaction equations and generalized in Reference 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 References 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.^{41,42} For non-evolving domains, a suitable preconditioning technique for the Navier–Stokes system based on a geometric multigrid method is analyzed computationally in Reference 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. First, its convergence properties are analyzed for a time-dependent domain. Second, the well-know DFG benchmark⁴⁴ 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 illustrated for two different settings.

This work is organized as follows. In Section 2, the prototype model problem and the notation are introduced. In Section 3, our space-time finite element approach for simulating the Navier–Stokes system on evolving domains is presented. In Section 4, practical aspects of the parallel implementation of the algorithms in the *deal.II* library⁴⁵ and the linear algebra package *Trilinos*⁴⁶ are addressed. In Section 5, the numerical results for a sequence of test problems are presented. We end with a summary and an outlook in Section 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 Figure 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 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 $\Gamma_r^t := \partial\Omega_r(t)$, 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 $\overline{\Omega_r^t} \subset \Omega$ 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 $\Omega_f^t := \Omega_f(t) = \Omega \setminus \overline{\Omega_r^t}$, with $\Omega_f^t \subset \Omega$, we denote the open domain filled with fluid. For Ω_f^t , with $t \in [0, T]$, and $I := (0, T]$ we consider solving the incompressible Navier–Stokes system

$$\partial_t \mathbf{v} + (\mathbf{v} \cdot \nabla) \mathbf{v} - \nu \Delta \mathbf{v} + \nabla p = \mathbf{f} \quad \text{in } \Omega_f^t \times I, \quad (1a)$$

$$\nabla \cdot \mathbf{v} = 0 \quad \text{in } \Omega_f^t \times I, \quad (1b)$$

$$\mathbf{v} = \mathbf{0} \quad \text{on } \Gamma_w \times I, \quad (1c)$$

$$\mathbf{v} = \mathbf{g}_i \quad \text{on } \Gamma_i \times I, \quad (1d)$$

$$\mathbf{v} = \mathbf{g}_r \quad \text{on } \Gamma_r^t \times I, \quad (1e)$$

$$\nu \nabla \mathbf{v} \cdot \mathbf{n} - np = \mathbf{0} \quad \text{on } \Gamma_o \times I, \quad (1f)$$

$$\mathbf{v}(0) = \mathbf{v}_0 \quad \text{in } \Omega_f^t. \quad (1g)$$

In (1), the velocity field \mathbf{v} and the pressure p are the unknown variables. In (1a), the parameter $\nu > 0$ denotes the fluid's viscosity and the right-hand side function \mathbf{f} is a given external force. In (1f), the field \mathbf{n} is the outer unit normal vector. In (1g), the function \mathbf{v}_0 denotes the prescribed initial velocity. On Γ_i , an inflow flow profile is prescribed by (1d). The

boundary parts Γ_w are fixed walls with no-slip boundary (1c), and Γ_o represents an outflow boundary that is modeled by the do-nothing boundary condition (1f); compare Reference 47. The boundary condition (1e) ensures that the fluid follows the prescribed rigid body's motion \mathbf{g}_r on Γ_r^t . Here, \mathbf{g}_r is the prescribed material velocity of the particles from Γ_r^t . To simplify the notation, we put

$$\mathbf{g} = \begin{cases} \mathbf{0} & \text{on } \Gamma_w \times I, \\ \mathbf{g}_i & \text{on } \Gamma_i \times I, \\ \mathbf{g}_r & \text{on } \Gamma_r^t \times I \end{cases}$$

and, with $\Gamma_D^t := \Gamma_i \cup \Gamma_w \cup \Gamma_r^t$, we rewrite the conditions (1c) to (1e) by

$$\mathbf{v} = \mathbf{g} \text{ on } \Gamma_D^t \times I.$$

We assume, that a unique (weak) solution of the system (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, for example, References 3,48,49. In particular, we refer to Reference 48 (p. 156, Thm. 1, p. 164, Cor.) for the existence of local in time solutions to (1) or solutions for sufficiently small data. Even though solutions to (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 $t \rightarrow 0$ under realistic assumptions about the data of the problem (cf. References 50–52) 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 (1). For the analysis of solutions to (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, for example, Reference 3 for the Stokes system and to Reference 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 $S \subset \mathbb{R}^2$ 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 $L_0^2(S) := \{v \in L^2(S) \mid \int_S v \, dx = 0\}$ 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 ∂S of S as $H_{0,\Gamma}^1(S)$. 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 $A_S : (\mathbf{H}^1(S) \times L_0^2(S)) \times (\mathbf{H}_{0,\Gamma}^1(S) \times L_0^2(S)) \rightarrow \mathbb{R}$ by

$$A_S((\mathbf{v}, p), (\boldsymbol{\psi}, \xi)) := \langle (\mathbf{v} \cdot \nabla) \mathbf{v}, \boldsymbol{\psi} \rangle_S + \nu \langle \nabla \mathbf{v}, \nabla \boldsymbol{\psi} \rangle_S - \langle p, \nabla \cdot \boldsymbol{\psi} \rangle_S + \langle \nabla \cdot \mathbf{v}, \xi \rangle_S, \quad (2)$$

for $(\mathbf{v}, p) \in \mathbf{H}^1(S) \times L_0^2(S)$ and $(\boldsymbol{\psi}, \xi) \in \mathbf{H}_{0,\Gamma}^1(S) \times L_0^2(S)$. For given $\mathbf{f} \in L^2(S)$ we introduce the linear form $L : \mathbf{H}^1(S) \rightarrow \mathbb{R}$ by

$$L_S(\boldsymbol{\psi}; \mathbf{f}) := \langle \mathbf{f}, \boldsymbol{\psi} \rangle_S, \quad (3)$$

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

Further, we let

$$\mathbf{V}_{\text{div}}^t := \left\{ \mathbf{v} \in \mathbf{H}^1(\Omega_f^t) \mid \nabla \cdot \mathbf{v} = 0 \text{ in } \Omega_f^t \right\},$$

and define the spaces

$$\begin{aligned} V_I &:= \left\{ \mathbf{v} \in L^2(I; \mathbf{H}^1(\Omega_f^t) \cap \mathbf{V}_{\text{div}}^t) \mid \partial_t \mathbf{v} \in L^2(I; L^2(\Omega_f^t)) \right\}, \\ V_{0,I} &:= \left\{ \mathbf{v} \in L^2(I; \mathbf{H}_{0,\Gamma_D}^1(\Omega_f^t)) \right\}. \end{aligned} \quad (4)$$

and

$$L_{0,I}^2 := L^2\left(I; L_0^2(\Omega_f^t)\right), \quad L_I^\infty := L^\infty\left(I; \mathbf{H}^1(\Omega_f^t)\right). \quad (5)$$

In the first of the definitions in (4), the weak partial derivative with respect to the time variable t is defined as an element of $L^2(\Omega_I)$ for the space-time flow domain $\Omega_I := \bigcup_{0 \leq t \leq T} \Omega_f^t \times \{t\}$. The definitions (4) and (5) are sufficient here. For an abstract framework to treat evolving spaces we refer to Reference 53.

2.3 | Space discretization

Let $\mathcal{T}_h = \{K\}$ be a family of shape-regular, structured decompositions of the pipe Ω (cf. Figure 1) into (open) quadrilaterals K with maximum cell size h . Precisely, \mathcal{T}_h is the computational background mesh. It is not fitted to the boundary of the moving rigid body Ω_r^t ; compare Figure 2. We make the following assumptions.

Assumption 2. The family $\mathcal{T}_h = \{K\}$ 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, compare Remark 9.

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 Ω_r^t has a curved boundary. In our implementation we ensure, that the condition of item 3 is always fulfilled, compare Section 4 for details.

The time-dependent set of mesh cells $K \in \mathcal{T}_h$ that are subsets of the fluid domain Ω_f^t is denoted by $\mathcal{T}_{h,f}^t$, the mesh cells $K \in \mathcal{T}_h$ that are subsets of the rigid domain Ω_r^t is denoted by $\mathcal{T}_{h,r}^t$. The set of cut cells $K \in \mathcal{T}_h$ such that $K \cap \Omega_f^t \neq \emptyset$ and $K \cap \Omega_r^t \neq \emptyset$ is denoted by $\mathcal{T}_{h,c}^t$; compare Figure 2C.

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

$$H_h^{(r)} = \left\{ v_h \in C(\bar{\Omega}) \mid v_h|_T \in \mathbb{Q}_r(K) \ \forall K \in \mathcal{T}_h \right\}, \quad (6)$$

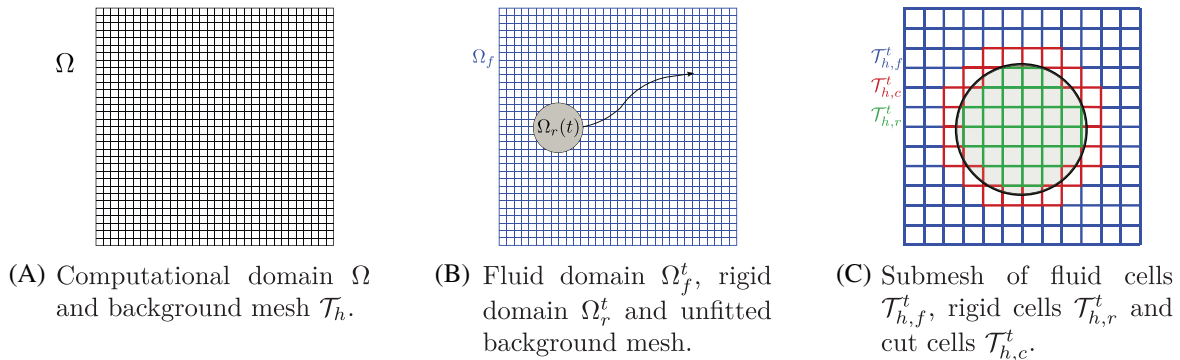


FIGURE 2 CutFEM mesh topology with computational background mesh and submeshes of fluid, cut, and rigid cells [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/doi/10.1002/nd.5074)]

where $\mathbb{Q}_r(K)$ 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 Section 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. Reference 54), when the proposed ghost penalty stabilization of Section 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 (6) we then put

$$\mathbf{V}_h = H_h^{(r)} \times H_h^{(r)}, \quad Q_h = H_h^{(r-1)}.$$

The space of weakly divergence free functions is denoted by

$$\mathbf{V}_h^{\text{div}} = \{\mathbf{v}_h \in \mathbf{V}_h \mid \langle \nabla \cdot \mathbf{v}_h, q_h \rangle = 0 \text{ for all } q_h \in Q_h\}.$$

Finally, we define the spaces

$$V_{I,h} := \{\mathbf{v}_h \in L^2(I; \mathbf{V}_h) \mid \partial_t \mathbf{v} \in L^2(I; \mathbf{V}_h)\}, \quad L_{0,I,h}^2 := L^2(I; Q_h).$$

2.4 | Time discretization

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

$$\mathbb{P}_k(I_n; B) = \left\{ w_\tau : I_n \rightarrow B \mid w_\tau(t) = \sum_{j=0}^k W_j t^j \forall t \in I_n, W_j \in B \forall j \right\}. \quad (7)$$

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

$$X_\tau^k(B) := \{w_\tau \in L^2(I; B) \mid w_\tau|_{I_n} \in \mathbb{P}_k(I_n; B) \forall I_n \in \mathcal{M}_\tau, w_\tau(0) \in B\}. \quad (8)$$

3 | SPACE-TIME FINITE ELEMENT DISCRETIZATION WITH CUTFEM

Here we introduce our approximation of the Navier–Stokes system (1) by combining CutFEM and variational time discretization techniques. First, we introduce the variational formulation of the Navier–Stokes system (1). The next subsection is devoted to the discretization in space. Nitsche’s method²¹ 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, that is, 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 Ω . The stabilization reduces spurious oscillations by irregular cuts of finite elements by the moving rigid body Ω_f^t and improves the conditioning of the resulting algebraic system; compare Reference 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; compare Reference 24.

3.1 | Variational formulation of the continuous problem

A sufficiently regular solution of the Navier–Stokes system (1) satisfies the following variational problem that provides the basis for space-time finite element discretizations.

Problem 1 (Continuous problem). Let $\mathbf{f} \in L^2(\Omega_I)$ and $\mathbf{v}_0 \in \mathbf{H}_{0,\Gamma_D^t}^1(\Omega_f^0) \cap \mathbf{V}_{\text{div}}^0$ be given. Let $\hat{\mathbf{g}} \in V_I \cap L_I^\infty$ denote a prolongation of the boundary values on Γ_D^t to Ω_f^t , such that $\hat{\mathbf{g}} = \mathbf{g}$ on $\Gamma_D^t \times I$. Find $(\mathbf{v}, p) \in (V_I \cap L_I^\infty) \times L_{0,I}^2$, with $\mathbf{v} \in \hat{\mathbf{g}} + V_{0,I}$, such that $\mathbf{v}(0) = \mathbf{v}_0$ and for all $\boldsymbol{\varphi} := (\boldsymbol{\psi}, \xi) \in V_{0,I} \times L_{0,I}^2$,

$$\int_0^T \langle \partial_t \mathbf{v}, \boldsymbol{\psi} \rangle_{\Omega_f^t} + A_{\Omega_f^t}((\mathbf{v}, p), (\boldsymbol{\psi}, \xi)) \, dt = \int_0^T L_{\Omega_f^t}(\boldsymbol{\psi}; \mathbf{f}) \, dt. \quad (9)$$

Remark 1. Regarding the well-posedness of Problem 1 we note the following.

- In Reference 48 (p. 156, Thm. 1), the existence and uniqueness of local in time weak solutions to Equations (1a), (1b), (1g) equipped with homogeneous Dirichlet boundary conditions (or Problem 1, respectively) 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, $\mathbf{v} \in L_I^\infty$ and $\partial_t \mathbf{v} \in L^2(\Omega_I)$ is ensured on a sufficiently small time interval. Further, in Reference 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 $\partial_t \mathbf{v} \in L^2(\Omega_I)$ instead of the distribution $\partial_t \mathbf{v} \in L^2(I; H^{-1}(\Omega))$ for time-independent domains Ω is obtained. For this reason, the stricter assumption $\mathbf{f} \in L^2(\Omega_I)$ about the right-hand side function \mathbf{f} is made here. Further regularity conditions about \mathbf{f} are supposed in Reference 48 for the proof of existence and uniqueness.
- For $\mathbf{v} \in V_I \cap L_I^\infty$ and $p \in L_{0,I}^2$ the left-hand side terms in (9) are well-defined. The existence of the sufficiently smooth extension $\hat{\mathbf{g}}$ 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 Reference 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. References 21,55) we introduce the bilinear form $B_{\Gamma_D^t} : \mathbf{H}^{1/2}(\Gamma_D^t) \times (\mathbf{V}_h \times Q_h) \rightarrow \mathbb{R}$ by

$$B_{\Gamma_D^t}(\mathbf{w}, (\boldsymbol{\psi}_h, \xi_h)) := -\langle \mathbf{w}, \nu \nabla \boldsymbol{\psi}_h \cdot \mathbf{n} + \xi_h \mathbf{n} \rangle_{\Gamma_D^t} + \gamma_1 \nu \langle h^{-1} \mathbf{w}, \boldsymbol{\psi}_h \rangle_{\Gamma_D^t} + \gamma_2 \langle h^{-1} \mathbf{w} \cdot \mathbf{n}, \boldsymbol{\psi}_h \cdot \mathbf{n} \rangle_{\Gamma_D^t}, \quad (10)$$

for $\mathbf{w} \in \mathbf{H}^{1/2}(\Gamma_D^t)$ and $(\boldsymbol{\psi}_h, \xi_h) \in \mathbf{V}_h \times Q_h$, where $\gamma_1 > 0$ and $\gamma_2 > 0$ are numerical (tuning) parameters for the penalization, compare Remark 5. The discrete semilinear form $A_h : (\mathbf{V}_h \times Q_h) \times (\mathbf{V}_h \times Q_h) \rightarrow \mathbb{R}$ is then given by

$$A_h((\mathbf{v}_h, p_h), (\boldsymbol{\psi}_h, \xi_h)) := A_{\Omega_f^t}((\mathbf{v}_h, p_h), (\boldsymbol{\psi}_h, \xi_h)) - \langle \nu \nabla \mathbf{v}_h \cdot \mathbf{n} - p_h \mathbf{n}, \boldsymbol{\psi}_h \rangle_{\Gamma_D^t} + B_{\Gamma_D^t}(\mathbf{v}_h, \boldsymbol{\varphi}_h), \quad (11)$$

for $(\mathbf{v}_h, p_h) \in \mathbf{V}_h \times Q_h$ and $(\boldsymbol{\psi}_h, \xi_h) \in \mathbf{V}_h \times Q_h$. The linear form $L_h : (\mathbf{V}_h \times Q_h) \rightarrow \mathbb{R}$ is defined by

$$L_h((\boldsymbol{\psi}_h, \xi_h); \mathbf{f}, \mathbf{g}) := L_{\Omega_f^t}(\boldsymbol{\psi}_h; \mathbf{f}) + B_{\Gamma_D^t}(\mathbf{g}, (\boldsymbol{\psi}_h, \xi_h)), \quad (12)$$

for $(\boldsymbol{\psi}_h, \xi_h) \in \mathbf{V}_h \times Q_h$. In (11) and (12), the forms A_h and L_h of (2) and (3), respectively, are extended naturally to test functions of \mathbf{V}_h with nonhomogeneous Dirichlet conditions.

Remark 2.

- We note that the forms (10) and (11) are introduced for (spatially) discrete functions that are defined on the time-independent background mesh \mathcal{T}_h of the pipe Ω . In (10) and (11), the integration is done over the evolving fluid domain Ω_f^t and its boundary part Γ_D^t where Dirichlet boundary conditions are prescribed. The domain of the rigid body Ω_r^t is considered as a fictitious (ghost) flow domain. In the following, the discrete fluid velocity \mathbf{v}_h and pressure p_h are defined implicitly in the ghost domain Ω_r^t by the (semi-) discrete variational problem that is augmented by a ghost penalty stabilization exploiting an extension of the discrete functions to Ω_r^t .
- We comment on the different boundary terms in the forms (10) and (11). The second term on the right-hand side of (11) reflects the natural boundary condition, making the weak imposition of the boundary conditions consistent. The first term on the right-hand side of (10) 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 (10) provides a natural weighting between boundary terms corresponding to viscous effects ($\mathbf{v} = \mathbf{g}$), convective behavior ($(\mathbf{v} \cdot \mathbf{n}) - \mathbf{v} = (\mathbf{g} \cdot \mathbf{n}) - \mathbf{g}$) and inviscid behavior ($\mathbf{v} \cdot \mathbf{n} = \mathbf{g} \cdot \mathbf{n}$).

To control irregular cuts of finite elements by the evolving domain and extend the fluid velocity and pressure functions from Ω_f^t to Ω_r^t , and thereby to the entire, time-independent background domain Ω , we introduce a discrete ghost penalty operator S_h . This stabilization then expands the semi-linear form (11) by additional terms. Our stabilization modifies an approach that was first introduced in Reference 1 for convection-diffusion-reaction equations and, then, considered in Reference 2 for Stokes problems to the Navier–Stokes system (1). This approach offers the appreciable advantage over further implementations of the ghost penalty method (cf. Reference 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 Figures 3 and 4 respectively. The first aim of the ghost stabilization S_h , defined in (14), is to extend the solution from the physical fluid domain Ω_f to the rigid domain Ω_r . To assemble the algebraic system, that is, 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 . Due to the motion of the rigid body Ω_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 $\Omega_r^{\tilde{t}_{n-1}}$ that belong to the fluid domain $\Omega_f^{\tilde{t}_n}$ at time \tilde{t}_n such that an appropriate extension of the solution at time \tilde{t}_{n-1} to the rigid domain at time \tilde{t}_{n-1} or to the entire time-independent domain Ω 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.

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 Figure 4A. 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 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 Figure 4B. 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 Figure 4B, an instability of the discrete solution in the cut cells is

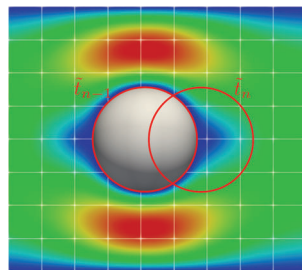


FIGURE 3 A moving rigid domain at two different time points \tilde{t}_{n-1} and \tilde{t}_n with computational background mesh [Colour figure can be viewed at wileyonlinelibrary.com]

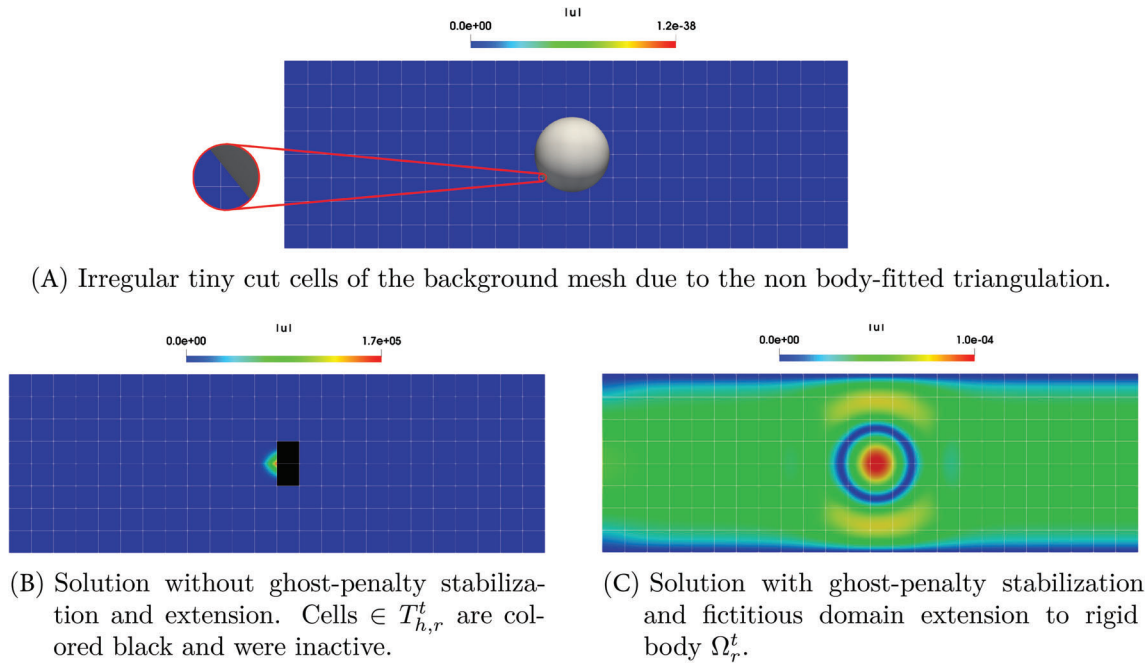


FIGURE 4 Effect of ghost-penalty stabilization and extension [Colour figure can be viewed at wileyonlinelibrary.com]

observed. By usage of ghost penalty stabilization that is proposed in the following, a smooth extension is obtained; compare Figure 4C. 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.

Our stabilization differs from the approach that is suggested in Reference 2. In Reference 2 the stabilization covers a boundary strip, that is, the cut cells and cells of the fluid and the rigid domain adjacent to the boundary Γ_r^t . 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) Ω_r^t and, thus, to the entire domain Ω of our problem setting (cf. Figure 1). This increases the efficiency of our parallel implementation (cf. Section 4). Our numerical experiments (cf. Section 5) indicate the admissibility of this modified ghost penalty stabilization. For the problem setting of Figure 1 we denote by the time-dependent submesh $\mathcal{T}_{h,s}^t$ the set of all cut cells and all cells that are entirely in the rigid domain,

$$\mathcal{T}_{h,s}^t := \mathcal{T}_{h,r}^t \cup \mathcal{T}_{h,c}^t = \left\{ K \in \mathcal{T}_h \mid K \subset \Omega_r^t \text{ or } K \cap \Omega_r^t \neq \emptyset \neq K \cap \Omega_f^t \right\}.$$

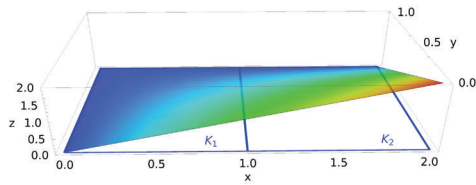
The set of all the faces that are common to two cells K_1 and K_2 of $\mathcal{T}_{h,s}^t$ is defined by

$$F_h^t := \left\{ \overline{K_1} \cap \overline{K_2} \mid K_1 \in \mathcal{T}_{h,s}^t, K_2 \in \mathcal{T}_{h,s}^t, K_1 \neq K_2, \text{meas}_1(\overline{K_1} \cap \overline{K_2}) > 0 \right\}. \quad (13)$$

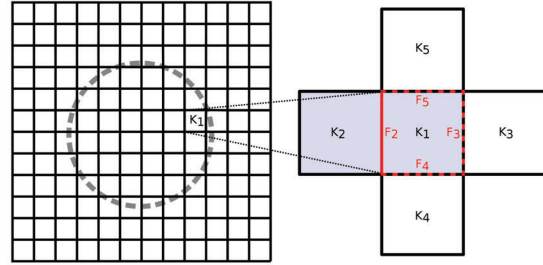
For $F \in F_h^t$, we denote by $\omega_F = K_1 \cup K_2$ the patch of the two adjacent elements K_1 and K_2 with the common face F . We define the bilinear form $S_{F_h^t} : (\mathbf{V}_h \times Q_h) \times (\mathbf{V}_h \times Q_h) \rightarrow \mathbb{R}$ by

$$\begin{aligned} S_{F_h^t}((\mathbf{v}_h, p_h), (\psi_h, \xi_h)) &= \sum_{F \in F_h^t} \gamma_v \frac{1}{h^2} \langle (\mathcal{E} \mathbf{v}_h|_{K_1} - \mathcal{E} \mathbf{v}_h|_{K_2}), (\mathcal{E} \psi_h|_{K_1} - \mathcal{E} \psi_h|_{K_2}) \rangle_{\omega_F} \\ &\quad + \gamma_p \langle (\mathcal{E} p|_{K_1} - \mathcal{E} p|_{K_2}), (\mathcal{E} \xi|_{K_1} - \mathcal{E} \xi|_{K_2}) \rangle_{\omega_F}, \end{aligned} \quad (14)$$

with numerical parameters $\gamma_v, \gamma_p > 0$. The stabilization (14) is based on Reference 2. However, in this work it is applied to the cut and rigid body cells given by $\mathcal{T}_{h,s}^t$. Our performed computations (cf. Section 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; compare Reference 56. A superiority of the latter stabilization is not observed. In Equation (14), \mathcal{E} is



(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 5 Patchwise extension of ghost penalty stabilization [Colour figure can be viewed at wileyonlinelibrary.com]

the canonical extension of a polynomial function of degree r in each variable, that is defined on one element of the face patch ω_F , to the whole patch; compare Reference 1. Thus, we have that

$$\mathcal{E} : \mathbb{Q}_r(K_i) \rightarrow \mathbb{Q}_r(\omega_F), \quad \text{for } i \in \{1, 2\}, \quad (15)$$

with $(\mathcal{E}u)|_{K_i} = u$ for $u \in \mathbb{Q}_r(K_i)$. In the case of vector-valued functions, the extension (15) is applied component wise. To illustrate the extension (15), Figure 5A shows the extended function $\mathcal{E}v_1$ of a scalar-valued finite element Lagrange basis function $v_1 \in \mathbb{Q}_1(K_1)$ that is defined on the quadrilateral K_1 and has the value 1 in the grid node $(1, 0)^T$. In Figure 5B, we illustrate the application of the stabilization operator (14) 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 (14) along with the extension (15). 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 , that is, $\omega_F = K_1 \cup K_2$, is colored in purple in Figure 5B. After evaluating the extension (14) 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 \mathbf{v}_h or pressure p_h to Ω_r^t by adding the stabilization (14) 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 Ω_r^t are obtained by the solution of the algebraic system and is not explicitly prescribed.

Finally, we define the stabilized semilinear form $A_h^s : (\mathbf{V}_h \times Q_h) \times (\mathbf{V}_h \times Q_h) \rightarrow \mathbb{R}$ by

$$A_h^s((\mathbf{v}_h, p_h), (\boldsymbol{\psi}_h, \xi_h)) := A_h((\mathbf{v}_h, p_h), (\boldsymbol{\psi}_h, \xi_h)) + S_{F_h^t}((\mathbf{v}_h, p_h), (\boldsymbol{\psi}_h, \xi_h)), \quad (16)$$

for $(\mathbf{v}_h, p_h) \in \mathbf{V}_h \times Q_h$ and $(\boldsymbol{\psi}_h, \xi_h) \in \mathbf{V}_h \times Q_h$.

Remark 3. We note that A_h^s is defined on the time-independent bulk space $\mathbf{V}_h \times Q_h$. The volume integrals of A_h (cf. (11)) in (16) and of L_h in (12) are computed over the time-dependent fluid domain Ω_f^t . Thus, an integration over the fluid portion of the cut cells is required; compare Section 4. The penalty stabilization S_h (cf. (14)) is computed over time-dependent patches that are subsets either of the fluid and the rigid domain or of the rigid one only.

Remark 4. Since we extend the solution of the physical fluid domain Ω_f to the whole computational domain Ω , the choice of the time step size is not restricted by a geometric CFL condition as in Reference 57. 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.

Remark 5. We comment on the choice of the algorithmic (tuning) parameters γ_1 and γ_2 of the Nitsche formulation (cf. Equation (10)) as well as on the choice of γ_v and γ_p of the ghost penalty extension (cf. Equation (14)). In References 21,55, a dependence of γ_1 and γ_2 on the viscosity ν is outlined. In References 4,6,58, the magnitude of the velocity \mathbf{v} is used further for their computation in practice. Moreover, it is demonstrated by numerical experiments, that a choice of the algorithmic parameters in the interval (10,100) leads to robust results for a wide range of flow regimes. Next, we address the ghost penalty extension parameters γ_v and γ_p . In Reference 2, their dependence on the viscosity ν by the term $\nu + \nu^{-1}$ is derived from theoretical error estimates, with no guarantee of sharpness of the estimates. In References 57,59, a choice

depending further on the fluid's velocity is suggested. However, for numerical simulations an experimental adaptation is applied further. In our simulations presented in Section 5, the algorithmic parameters are fixed by experimental investigations (fitting) for the chosen Reynolds number or fluid viscosity, respectively. The appearance of algorithmic constants is in fact a weakness of the approach. Their optimal choice still remains an open problem. For this, we observe a strong analogy to residual based stabilizations (for instance SUPG stabilization of convection-dominated problems), where the problem of the optimal choice of the algorithmic parameters could not be solved completely up to now. Precisely, in our computations we put $\gamma_1 = \gamma_2 = 35$ and let $\gamma_v = \gamma_p = 1.0e1$, except for Section 5.4, where $\gamma_v = \gamma_p = 1.0e3$ is used. In our performed simulations, a mild sensitivity with respect to the choice of the algorithmic constants is observed only.

We are now in a position to define the semidiscrete approximation of the system (1).

Problem 2. Let $\mathbf{f} \in L^2(\Omega_I)$ and $\mathbf{v}_{0,h} \in \mathbf{V}_h^{\text{div}}$ be given. Find $(\mathbf{v}_h, p_h) \in V_{I,h} \times L^2_{0,I,h}$, such that $\mathbf{v}_h(0) = \mathbf{v}_{0,h}$ and for all $(\psi, \xi) \in L^2(I; V_h) \times L^2(0, T; Q_h)$,

$$\int_0^T \langle \partial_t \mathbf{v}_h, \psi_h \rangle_{\Omega^t} + A_h^s((\mathbf{v}_h, p_h), (\psi_h, \xi_h)) \, dt = \int_0^T L_h(\psi_h; \mathbf{f}, \mathbf{g}) \, dt. \quad (17)$$

This is the corresponding LaTeX code

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 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, for example, References 41,60,61. 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; compare Reference 24,62. 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

$$\partial_t \mathbf{v} + \mathbf{A}\mathbf{v} = \mathbf{f}, \quad \mathbf{v}(0) = \mathbf{v}_0, \quad (18)$$

as an equality in the dual space $V^*(t)$ of a time-dependent Hilbert space $V(t)$ of functions defined on an evolving domain Ω^t for $t \in [0, T]$; compare Reference 53. In the formal derivation we tacitly assume that the solution of (18) 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

$$\frac{d}{dt} \int_{\Omega^t} \mathbf{v} \cdot \boldsymbol{\psi} \, d\mathbf{x} = \int_{\Omega^t} \partial_t \mathbf{v} \cdot \boldsymbol{\psi} \, d\mathbf{x} + \int_{\Omega^t} \mathbf{v} \cdot \partial_t \boldsymbol{\psi} \, d\mathbf{x} + \int_{\partial\Omega^t} (\mathbf{v} \cdot \boldsymbol{\psi})(\mathbf{w} \cdot \mathbf{n}) \, ds. \quad (19)$$

Here, the vector field $\mathbf{w}(\mathbf{x}, t)$ is the material velocity of the particles from the boundary $\partial\Omega^t$, and \mathbf{n} is the outer unit normal vector. Substituting (18) into (19) yields that

$$\int_0^T \frac{d}{dt} \langle \mathbf{v}, \boldsymbol{\psi} \rangle_{\Omega^t} - \langle \mathbf{v}, \partial_t \boldsymbol{\psi} \rangle_{\Omega^t} - \langle \mathbf{v} \cdot \boldsymbol{\psi}, \mathbf{w} \cdot \mathbf{n} \rangle_{\partial\Omega^t} + \langle \mathbf{A}\mathbf{v}, \boldsymbol{\psi} \rangle_{\Omega^t} \, dt = \int_0^T \langle \mathbf{f}, \boldsymbol{\psi} \rangle_{\Omega^t} \, dt. \quad (20)$$

With the fundamental theorem of calculus and for test functions $\boldsymbol{\psi}$ with $\boldsymbol{\psi}(T) = \mathbf{0}$ we get that

$$-\langle \mathbf{v}(0), \boldsymbol{\psi}(0) \rangle_{\Omega^0} - \int_0^T \langle \mathbf{v}, \partial_t \boldsymbol{\psi} \rangle_{\Omega^t} + \langle \mathbf{v} \cdot \boldsymbol{\psi}, \mathbf{w} \cdot \mathbf{n} \rangle_{\partial\Omega^t} - \langle \mathbf{A}\mathbf{v}, \boldsymbol{\psi} \rangle_{\Omega^t} \, dt = \int_0^T \langle \mathbf{f}, \boldsymbol{\psi} \rangle_{\Omega^t} \, dt. \quad (21)$$

Rewriting the integrals in (21) as a sum over the subintervals of \mathcal{M}_τ , we get that

$$-\langle \mathbf{v}(0), \boldsymbol{\psi}(0) \rangle_{\Omega^0} - \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle \mathbf{v}, \partial_t \boldsymbol{\psi} \rangle_{\Omega^t} + \langle \mathbf{v} \cdot \boldsymbol{\psi}, \mathbf{w} \cdot \mathbf{n} \rangle_{\partial \Omega^t} - \langle \mathbf{A} \mathbf{u}, \boldsymbol{\psi} \rangle_{\Omega^t} dt = \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle \mathbf{f}, \boldsymbol{\psi} \rangle_{\Omega^t} dt. \quad (22)$$

From (19) we derive that

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

Applying the fundamental theorem of calculus to the first term on the right-hand side of this identity and substituting the resulting equation into (22) shows that

$$\begin{aligned} \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle \partial_t \mathbf{v}, \boldsymbol{\psi} \rangle_{\Omega^t} + \langle \mathbf{A} \mathbf{v}, \boldsymbol{\psi} \rangle_{\Omega^t} dt + \sum_{n=1}^{N-1} [\langle \mathbf{v}, \boldsymbol{\psi} \rangle_{\Omega^t}]_n + \langle \mathbf{v}(0^+), \boldsymbol{\psi}(0) \rangle_{\Omega^0} \\ = \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \langle \mathbf{f}, \boldsymbol{\psi} \rangle_{\Omega^t} dt + \langle \mathbf{v}_0, \boldsymbol{\psi}(0) \rangle_{\Omega^0}, \end{aligned} \quad (23)$$

with the jump operator $[\cdot]_n$ at t_n , defined as $[g]_n := g(t_n^+) - g(t_n^-)$ with the one-sided limits $g(t_n^\pm) = \lim_{t \rightarrow t_n^\pm} g(t)$. For a smoothly evolving domain Ω^t , that is assumed here, the one-sided limits of Ω^t at the time nodes t_n coincide. We note that (23) continues to be well-defined for functions that are differentiable piecewise in time (with respect to the time mesh \mathcal{M}_τ) only. Thus, (23) can be solved within the space (8) of piecewise polynomials in time.

Applying this concept to Problem 2 yields the following fully discrete problem.

Problem 3. Let $\mathbf{f} \in L^2(\Omega_f)$ and $\mathbf{v}_{0,h} \in \mathbf{V}_h^{\text{div}}$ be given. For $n = 1, \dots, N$, and given $\mathbf{v}_\tau, h_{|I_{n-1}} \in \mathbb{P}_k(I_{n-1}; \mathbf{V}_h)$ for $n > 1$ and $\mathbf{v}_\tau, h_{|I_{n-1}}(t_{n-1}^-) := \mathbf{v}_{0,h}$ for $n = 1$, find $(\mathbf{v}_{\tau,h}, p_{\tau,h}) \in \mathbb{P}_k(I_n; \mathbf{V}_h) \times \mathbb{P}_k(I_n; Q_h)$, such that for all $(\boldsymbol{\psi}_{\tau,h}, \xi_{\tau,h}) \in \mathbb{P}_k(I_n; \mathbf{V}_h) \times \mathbb{P}_k(I_n; Q_h)$,

$$\begin{aligned} \int_{t_{n-1}}^{t_n} \langle \partial_t \mathbf{v}_{\tau,h}, \boldsymbol{\psi}_{\tau,h} \rangle_{\Omega_f^t} + A_h^s((\mathbf{v}_{\tau,h}, p_{\tau,h}), (\boldsymbol{\psi}_{\tau,h}, \xi_{\tau,h})) dt + \langle \mathbf{v}_{\tau,h}(t_{n-1}^+), \boldsymbol{\psi}_{\tau,h}(t_{n-1}^+) \rangle_{\Omega_f^{t_{n-1}}} \\ = \int_{t_{n-1}}^{t_n} L_h(\boldsymbol{\psi}_{\tau,h}; \mathbf{f}, \mathbf{g}) dt + \langle \mathbf{v}_{\tau,h}(t_{n-1}^-), \boldsymbol{\psi}_{\tau,h}(t_{n-1}^+) \rangle_{\Omega_f^{t_{n-1}}}. \end{aligned} \quad (24)$$

Remark 6.

- The derivation (18) to (23) of the discontinuous Galerkin formulation is done on the evolving Eulerian domain by the usage of Reynolds transport theorem (19). This shows a strong relation to the transformation of the equations from Eulerian coordinates on moving domains to Lagrangian coordinates on fixed domains. The last term on the right-hand side of the Reynolds transport theorem (19) involves the particle velocity \mathbf{w} relating the material derivative d_t of the Arbitrary Lagrangian Eulerian formulation to the partial time derivative ∂_t by $d_t f(\mathbf{x}, t) = \partial_t f(\mathbf{x}, t) + \nabla f \cdot \mathbf{w}$.
- In Problem 3 a smoothly evolving domain is assumed such that $\Omega_f^{t_{n-1}^+} = \Omega_f^{t_{n-1}^-}$ holds. All volume integrals that arise in Equation (24) are computed over the fluid domain Ω_f^t . The integration over cells intersected by the domain's boundaries is addressed in Subsection 4.1.
- To solve the algebraic counterpart of Equation (24) 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., Reference 63), 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 Equation (24). 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 Section 5, both modifications of Newton's method lead to comparable results. We did not observe any convergence problems.

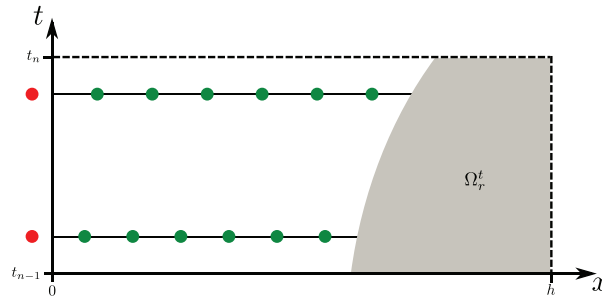


FIGURE 6 Space-time quadrature nodes for a one-dimensional problem and a single cut cell with a rigid domain Ω_r^t [Colour figure can be viewed at wileyonlinelibrary.com]

- To solve the linear system of the Newton iteration, we use the parallel, sparse direct solver SuperLU_DIST.⁶⁴ 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 Reference 43.
- Problem 3 corresponds to an Arbitrary–Lagrangian–Eulerian method in purely Eulerian coordinates.⁶⁵ So the fluid domain (which is in an ALE context usually called “spatial domain”) coincides with the reference domain and the mesh velocity is zero.⁶⁶

3.4 | Algebraic in time formulation

In this section we derive a semi-algebraic formulation of Problem 3. The algebraic counterpart of Equation (24) 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 Equation (24) 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 Reference 24.

In Problem 3, numerical integration in time by the m -point Gauss quadrature is still applied. For $g \in \{f : (0, T) \rightarrow L^2(\Omega) | f|_{I_n} \in C(I_n; L^2(\Omega)) \forall I_n \in \mathcal{M}_\tau\}$ this formula is given by

$$Q_n(g) = \sum_{\mu=1}^m w_\mu g(t_{n,\mu}) \approx \int_{I_n} g(t) dt, \quad (25)$$

where $t_{n,\mu} \in I_n$, for $\mu = 1, \dots, m$, 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 (25), is chosen such that $m \geq (3k + 1)/2$ is satisfied. Then, the integration in time of the discrete semilinear form $A_h((\mathbf{v}_{\tau,h}, p_{\tau,h}), (\boldsymbol{\psi}_{\tau,h}, \xi_{\tau,h}))$ on the left-hand side of (24) is done exactly. In Figure 6 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 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

$$\chi_l(t_{n,\mu}^{\text{GR}}) = \delta_{l,\mu}, \quad \text{for } l, \mu = 0, \dots, k,$$

for the Gauss–Radau quadrature nodes $\{t_{n,\mu}^{\text{GR}}\}_{\mu=0}^k$ of the subinterval I_n . Expanding the discrete solution $(\mathbf{v}_{\tau}|_{I_n}, p_{\tau}|_{I_n}) \in \mathbb{P}_k(I_n; \mathbf{V}_h) \times \mathbb{P}_k(I_n; Q_h)$ on I_n in terms of the basis functions yields

$$\mathbf{v}_{\tau,h}|_{I_n}(\mathbf{x}, t) = \sum_{l=0}^k \mathbf{v}_{n,l}(\mathbf{x}) \chi_l(t) \quad \text{and} \quad p_{\tau,h}|_{I_n}(\mathbf{x}, t) = \sum_{l=0}^k p_{n,l}(\mathbf{x}) \chi_l(t), \quad (26)$$

for $t \in I_n$ and with coefficient functions $\mathbf{v}_{n,l} \in \mathbf{V}_h$ and $p_{n,l} \in Q_h$. Appreciable advantage of the usage of the Gauss–Radau quadrature points for the construction of the temporal basis is that the quantity $\mathbf{v}_{\tau,h|I_{n-1}}(\mathbf{x}, t_{n-1}^-)$ in (24), 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 can be recovered in the following form.

Problem 4. Let $\mathbf{1}f \in L^2(\Omega_I)$ and $\mathbf{v}_{0,h} \in \mathbf{V}_h^{\text{div}}$ be given. For $n = 1, \dots, N$, and given $\mathbf{v}_{\tau,h|I_{n-1}} \in \mathbb{P}_k(I_{n-1}; \mathbf{V}_h)$ for $n > 1$ and $\mathbf{v}_{\tau,h|I_{n-1}}(t_{n-1}^-) := \mathbf{v}_{0,h}$ for $n = 1$, find $(\mathbf{v}_{\tau,h}, p_{\tau,h}) \in \mathbb{P}_k(I_n; \mathbf{V}_h) \times \mathbb{P}_k(I_n; Q_h)$, such that for all $j \in \{0 \dots k\}$ and $(\boldsymbol{\psi}_h, \xi_h) \in \mathbf{V}_h \times Q_h$,

$$\begin{aligned} \sum_{\mu=1}^m w_{\mu} \left[\langle \partial_t \mathbf{v}_{\tau,h|I_n}(t_{n,\mu}), \boldsymbol{\psi}_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f^{t_{n,\mu}}} + A_h^s \left((\mathbf{v}_{\tau,h|I_n}(t_{n,\mu}), p_{\tau,h|I_n}(t_{n,\mu})), (\boldsymbol{\psi}_h \chi_j(t_{n,\mu}), \xi_h \chi_j(t_{n,\mu})) \right) \right] \\ + \langle \mathbf{v}_{\tau,h|I_n}(t_{n-1}^+), \boldsymbol{\psi}_h \chi_j(t_{n-1}^+) \rangle_{\Omega_f^{t_{n-1}^+}} = \sum_{\mu=1}^m w_{\mu} \left[(L_h(\boldsymbol{\psi}_h \chi_j(t_{n,\mu}); \mathbf{f}, \mathbf{g})) \right] + \\ \langle \mathbf{v}_{\tau,h|I_{n-1}}(\mathbf{x}, t_{n-1}^-), \boldsymbol{\psi}_h(\mathbf{x}) \chi_j(t_{n-1}^+) \rangle_{\Omega_f^{t_{n-1}^+}}. \end{aligned} \quad (27)$$

To illustrate the structure of Equation (27), we review the terms in Equation (27) in detail. Exemplarily, this is done for some of them. Using the expansions given in (26), we get that

$$\langle \partial_t \mathbf{v}_{\tau,h|I_n}(t_{n,\mu}), \boldsymbol{\psi}_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f^{t_{n,\mu}}} = \sum_{l=0}^k \langle \mathbf{v}_{n,l} \partial_t \chi_l(t_{n,\mu}), \boldsymbol{\psi}_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f^{t_{n,\mu}}}, \quad (28)$$

$$\langle \nabla \mathbf{v}_{\tau,h|I_n}(t_{n,\mu}), \nabla \boldsymbol{\psi}_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f^{t_{n,\mu}}} = \sum_{l=0}^k \langle \nabla \mathbf{v}_{n,l} \chi_l(t_{n,\mu}), \nabla \boldsymbol{\psi}_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f^{t_{n,\mu}}}, \quad (29)$$

$$\langle p_{\tau,h|I_n}(t_{n,\mu}), \nabla \cdot \boldsymbol{\psi}_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f^{t_{n,\mu}}} = \sum_{l=0}^k \langle p_{n,l} \chi_l(t_{n,\mu}), \nabla \cdot \boldsymbol{\psi}_h \chi_j(t_{n,\mu}) \rangle_{\Omega_f^{t_{n,\mu}}}. \quad (30)$$

Thus, Problem 4 aims at computing the coefficient functions $\{\mathbf{v}_{n,l}, p_{n,l}\}_{l=0}^k$, with $\{\mathbf{v}_{n,l}, p_{n,l}\} \in \mathbf{V}_h \times Q_h$ for $l = 0, \dots, k$. Clearly, the pair $\{\mathbf{v}_{n,l}, p_{n,l}\} \in \mathbf{V}_h \times Q_h$ 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 \mathbf{V}_h and Q_h , respectively, and are thus defined on the computational background mesh \mathcal{T}_h of Ω . In the rigid body domain they are defined implicitly by means of the stabilization and extension operator $S_{F_h^t}$ introduced in Equation (14).

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). 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 first 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 or 4, respectively, the stabilized discrete form A_h^s , that is defined in (16) along with (11) and (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. Figure 7 illustrates schematically the types of cell

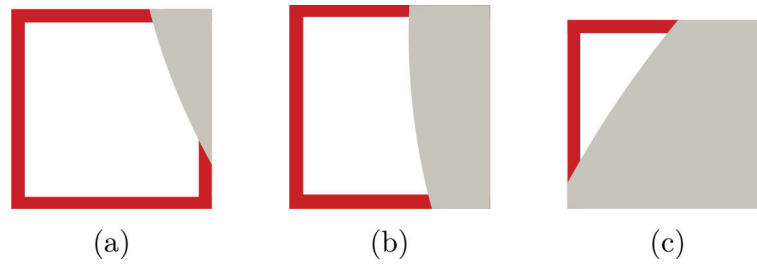


FIGURE 7 Types of arising cut cells with gray shaded rigid body and white shaded fluid domain. Ad cut quad can become a pentagon (a), a deformed quad (b) or a triangle (c). [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/doi/10.1002/nd.5074)]

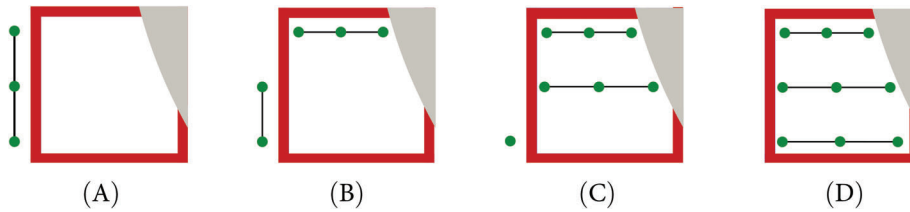


FIGURE 8 Quadrature nodes of iterated numerical integration over cut cells by Gauss quadrature formulas. At first a one dimensional quadrature rule is applied along the (axis parallel) longest face (A). Afterwards we iterate over each integration point and apply a one dimensional integration rule in the next axis direction (B) - (D) [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/doi/10.1002/nd.5074)]

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), that each face of the quadrilateral is cut at most once by the boundary of the rigid body (cf. Figure 7). The type of the cut cell is determined by evaluating the levelset function, for $t \in (0, T)$,

$$\theta(\mathbf{x}, t) \begin{cases} < 0 & \forall \mathbf{x} \in \Omega_r^t, \\ = 0 & \forall \mathbf{x} \in \partial\Omega_r^t, \\ > 0 & \forall \mathbf{x} \in \Omega_f^t \end{cases} \quad (31)$$

in the corners of each cell.

In order to integrate over such cut cells we apply iterated one-dimensional numerical integration schemes. This is illustrated in Figure 8 and reads as

$$\iint_{K_{\text{fluid}}} f(x_1, x_2) \, d(x_1, x_2) \approx \text{meas}_2(K_{\text{fluid}}) \sum_{\mu_1=1}^{M_1} \sum_{\mu_2=1}^{M_2} \omega_{\mu_1} \omega_{\mu_2} f(x_{1,\mu_1}, x_{2,\mu_2}), \quad (32)$$

where ω_{μ_1} and x_{1,μ_1} , for $\mu_1 = 1, \dots, M_1$, as well as ω_{μ_2} and x_{2,μ_2} , for $\mu_2 = 1, \dots, M_2$, denote the weights and quadrature nodes of the quadrature formula for the respective coordinate direction and K_{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 μ_1 , that is, on the quadrature node x_{1,μ_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. Figure 8), is chosen for the outer summation in Equation (32). For this we recall that in this work the assumption was made that the evolving domain and thus the position of the moving boundary Γ_r^t (cf. Figure 1) is prescribed explicitly and, therefore, is computable; compare Assumption 1. Further, a structured grid

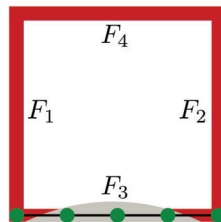


FIGURE 9 One face cut twice [Colour figure can be viewed at wileyonlinelibrary.com]

of elements aligned to the coordinate lines was supposed (cf. Section 2.3). In our implementation, the quadrature points of each cut cell are computed in each time step of Problem 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 7. The numerical results presented in Section 5 and Reference 56 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 8. The integration over the boundary Γ_r^t is performed using a standard parametrization of the circular rigid domain Ω_r^t . With this we transform the surface integral into a 1D integral that we numerically evaluate using a Gaussian quadrature rule.

Remark 9. For rigid domains Ω_r^t with curved boundary, a cut scenario as sketched in Figure 9 can occur. Here, the face F_3 of the cell is cut twice. If Ω_r is time-independent, such a scenario is avoided by using the mesh sufficiently fine. Nevertheless, when the rigid domain Ω_r^t 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 Figure 8. To ensure the conformity of the arising cuts with these patterns, we evaluate the levelset function (31) 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 Figure 9 for $n = 5$, which we use in all of the simulations presented in Section 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 Ω_r 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 Figure 9 are very unlikely to occur, especially when the mesh size is chosen sufficiently fine.

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 Figure 10. After execution of the program and before the loop of the time marching process of Problem 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 $\hat{\chi}_l \in \mathbb{P}_k(\hat{I}; \mathbb{R})$, satisfies

$$\hat{\chi}_l(\hat{t}_\mu) = \delta_{l,\mu}, \quad l, \mu = 0 \dots, k,$$

with the Kronecker symbol $\delta_{l,\mu}$ and the $(k+1)$ Gauss–Radau quadrature nodes $\{\hat{t}_\mu\}_{\mu=0}^l$ 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 κ_2 of the resulting Vandermonde matrix is feasible even for high values of k . This is illustrated in Figure 11. In each subinterval of the time marching process of Problem 3, numerical quadrature is applied for the integration in the time and space domain. First, the subinterval and its quadrature nodes are incremented to I_n and $t_{n,\mu}$,

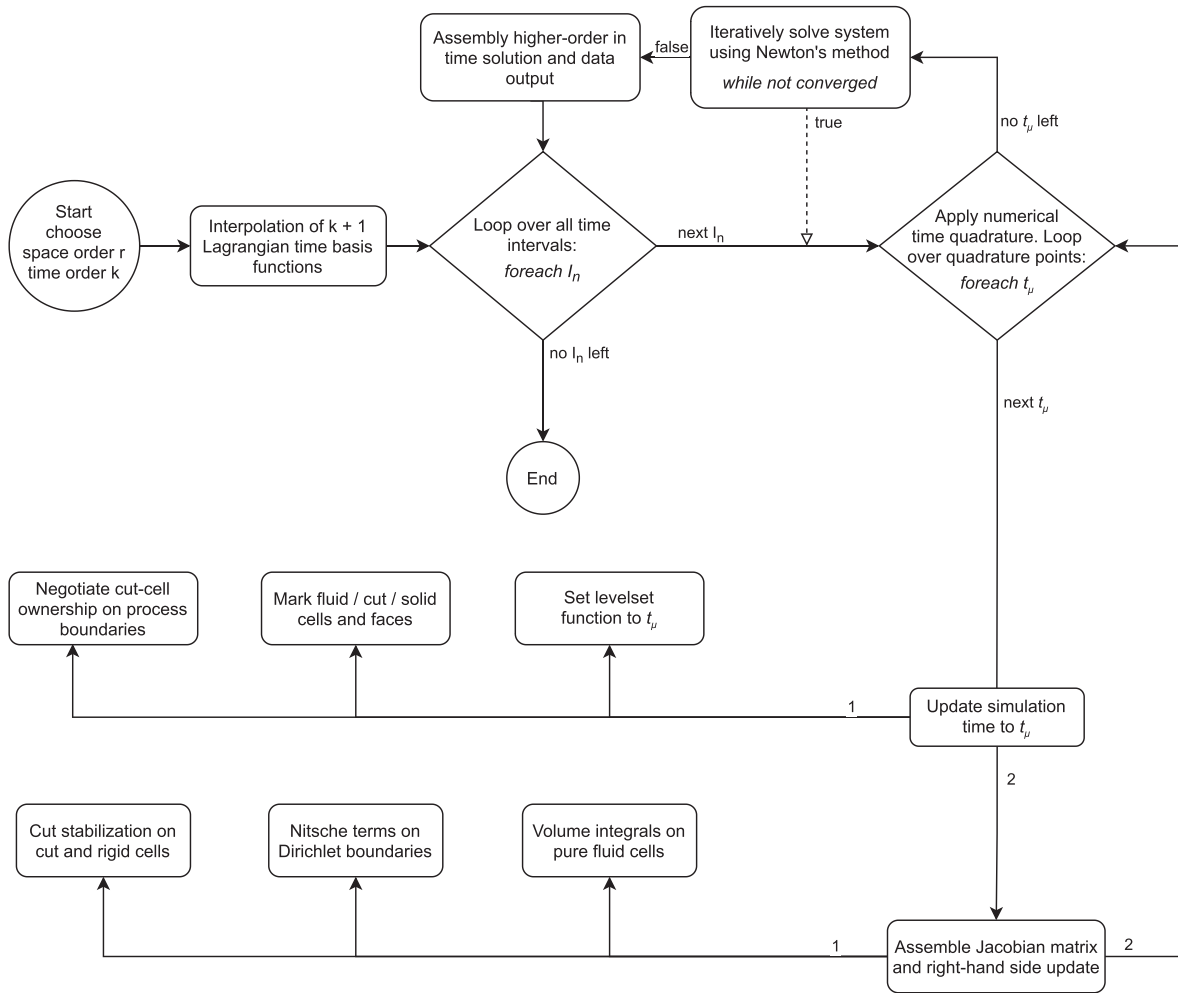


FIGURE 10 Key blocks of the CutFEM solver with arbitrary polynomial order in space and time

for $\mu = 0, \dots, k$. For the representation of the rigid domain Ω_r^t we use the levelset function $\theta(\mathbf{x}, t)$, defined in (31). By evaluating this function we identify whether a finite element cell belongs at time $t_{n,\mu}$ to the set of fluid cells $\mathcal{T}_{hf}^{t_{n,\mu}}$, rigid cells $\mathcal{T}_{hr}^{t_{n,\mu}}$ or cut cells $\mathcal{T}_{hc}^{t_{n,\mu}}$. 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 10. We note that the levelset function (31) is not discretized or computed by the approximation of a suitable transport equation. The direct evaluation of (31) becomes feasible due to the Assumption 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 Equation (14) 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 ω_F at process partition boundaries. This problem is sketched in Figure 12 for the patch ω_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 Equation (14), 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 6) is assembled. For the integration over fluid cut cells iterated integration, sketched in Section 4.1, is applied. Then, the resulting linear system is solved by the parallel,

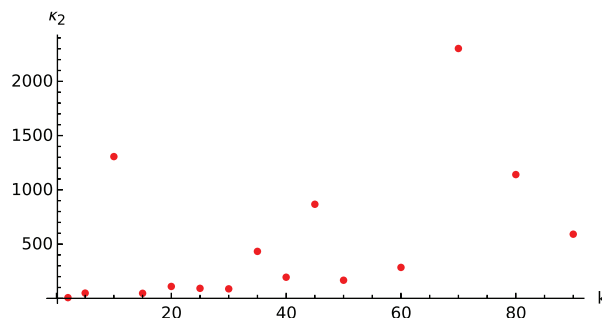


FIGURE 11 Spectral condition number κ_2 for the computation of the nodal $dG(k)$ time basis [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/doi/10.1002/nd.5074)]

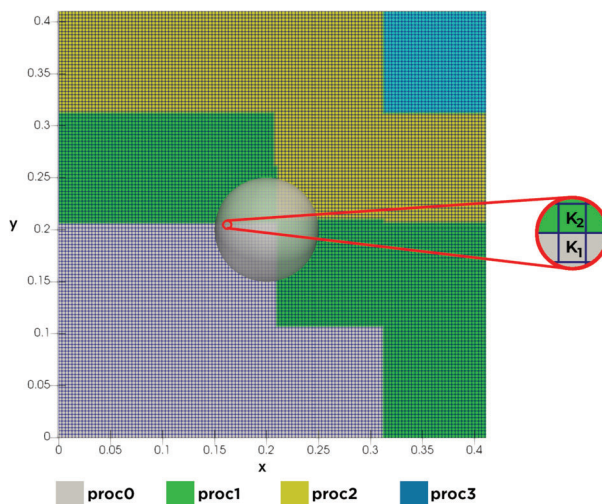


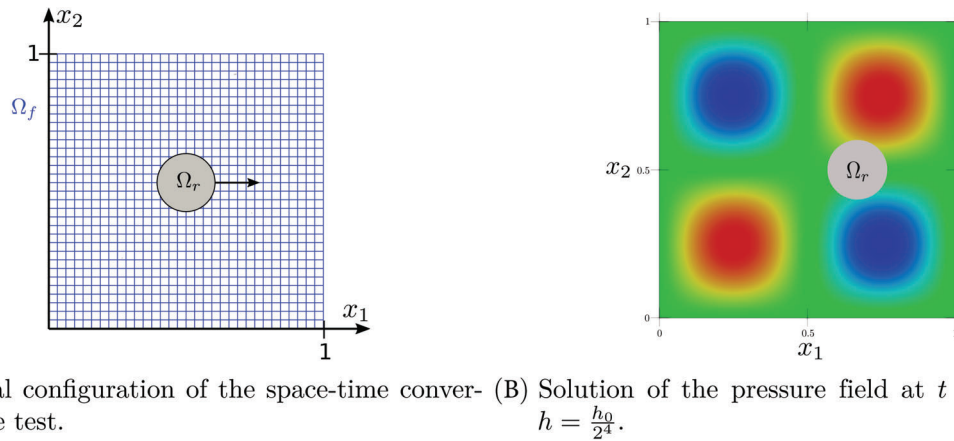
FIGURE 12 Assembly of the ghost penalty stabilization at partition boundaries with computational mesh \mathcal{T}_h and colored processor ownerships of mesh cells [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/doi/10.1002/nd.5074)]

sparse direct solver SuperLU_DIST⁶⁴ 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.

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 Ω_r^t , to validate all aspects of the implementation. In the second experiment, the popular DFG benchmark of flow around a cylinder (cf. Reference 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 (14) on an evolving domain and demonstrate it's application in a realistic example. Throughout, we set the viscosity to $\nu = 0.001$.

In all the examples, the Newton solver uses an absolute value of the residual smaller than 1×10^{-10} as the stopping criteria. In Section 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 Equation (10) to $\gamma_1 = \gamma_2 = 35$, which is motivated by Schott and Wall,⁶ Winter et al.⁴ and our computations. The ghost penalty parameters of Equation (14) were chosen experimentally and put to $\gamma_v = \gamma_p = 1.0e1$, except in Section 5.4, where we put $\gamma_v = \gamma_p = 1.0e3$.



(A) Initial configuration of the space-time convergence test. (B) Solution of the pressure field at $t = T$ with $h = \frac{h_0}{2^4}$.

FIGURE 13 Problem setup and pressure field at the end of the simulation [Colour figure can be viewed at wileyonlinelibrary.com]

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

τ	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
$\tau_0/2^0$	$h_0/2^0$	4.3136e-02	–	1.4756e-02	–	3.4198e-02	–	1.3397e-02	–
$\tau_0/2^1$	$h_0/2^1$	8.6019e-03	2.33	3.0469e-03	2.28	7.8334e-03	2.13	2.5895e-03	2.37
$\tau_0/2^2$	$h_0/2^2$	2.0628e-03	2.06	6.9638e-04	2.13	9.4588e-04	3.05	5.6048e-04	2.21
$\tau_0/2^3$	$h_0/2^3$	4.8824e-03	2.08	1.8527e-04	1.91	1.1506e-04	3.04	1.3538e-04	2.05
$\tau_0/2^4$	$h_0/2^4$	1.2206e-04	2.00	4.5999e-05	2.01	1.4283e-05	3.01	3.3846e-05	2.00
Elements $_{ I_n}$		$(\mathbb{P}_1(I_n; H_h^2))^2 \times \mathbb{P}_1(I_n; H_h^1)$				$(\mathbb{P}_2(I_n; H_h^2))^2 \times \mathbb{P}_2(I_n; H_h^1)$			

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 Figure 13A with a time-dependent domain Ω_r^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

$$\mathbf{x}_r(t) = \mathbf{x}_r(0) + (A \cdot \sin(\omega \cdot t), 0)^\top, \quad (33)$$

with $A = 0.2$ and $\omega = 1$. We prescribe the initial value \mathbf{v}_0 and right-hand side function \mathbf{f} on $\Omega_f \times I$ in such a way, that the solution of the Navier–Stokes system on Ω_f is given by

$$\begin{aligned} \mathbf{v}_e(\mathbf{x}, t) &:= \begin{pmatrix} \cos(x_2\pi) \cdot \sin(t) \cdot \sin(x_1\pi)^2 \cdot \sin(x_2\pi) \\ -\cos(x_1\pi) \cdot \sin(t) \cdot \sin(x_2\pi)^2 \cdot \sin(x_1\pi) \end{pmatrix}, \\ p_e(\mathbf{x}, t) &:= \cos(x_2\pi) \cdot \sin(t) \cdot \sin(x_1\pi) \cdot \cos(x_1\pi) \cdot \sin(x_2\pi). \end{aligned}$$

On the inner fluid boundary we prescribe the condition $\mathbf{g}_r = \mathbf{v}_e$. On the outer boundary we use a homogeneous Dirichlet condition. Table 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_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 $H_h^2 \times H_h^2$. 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 (6) and (7), respectively. The ghost penalty parameters of Equation (14) are set to $\gamma_v = 1.0001$ and $\gamma_p = 0.0001$. In Table 13, the expected convergence of optimal order is documented for both experiments.

Remark 11. The setting of this example was chosen in such a way, that the propagation of the boundary of Γ_r^t 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 Γ_r^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 Ω_r and the application of the stabilization introduced in (14). For this, we use the well-known DFG benchmark setting of flow around a cylinder, defined in Reference 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. Reference 44). With the drag and lift forces F_D and F_L on the rigid circle S given by

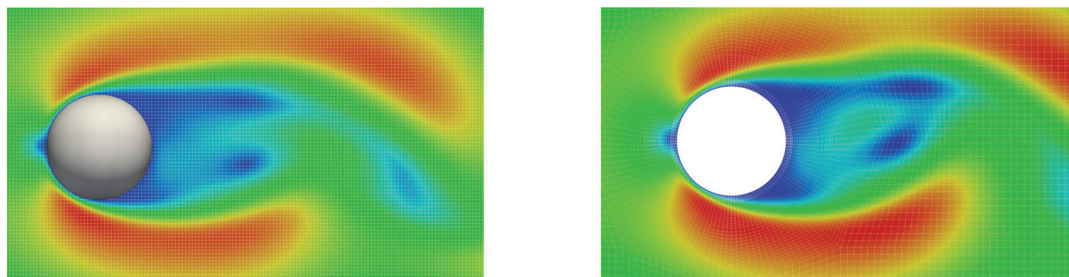
$$F_D = \int_S \left(v \frac{\partial v_t}{\partial \mathbf{n}} n_y - P n_x \right) dS, \quad F_L = - \int_S \left(v \frac{\partial v_t}{\partial \mathbf{n}} n_x - P n_y \right) dS, \quad (34)$$

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

$$c_D = \frac{2}{\bar{U}^2 L} F_D, \quad c_L = \frac{2}{\bar{U}^2 L} F_L. \quad (35)$$

According to Reference 44, we put $\mathbf{g}_i(x, y, t) = \left(\frac{4 \cdot 1.5 \cdot y(0.41 - y)}{0.41^2}, 0 \right)^\top$ on the inflow boundary Γ_i . 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 $\mathbb{P}_1(I_n; H_h^2) \times \mathbb{P}_1(I_n; H_h^1)$ on each time interval I_n . For the CutFEM approach we study a sequence of successive mesh refinements in space. For the simulation of the fully developed flow profile, i.e. after an initial time interval, and for the computation of the drag and lift coefficients as quantities of physical interest we use the time step size $\tau = 5e - 3$; cf. 44.

Figure 14 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. Reference 44). The CutFEM simulation is done for a sequence of successive refinement steps of the entire background mesh (cf. Table 2). Table 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

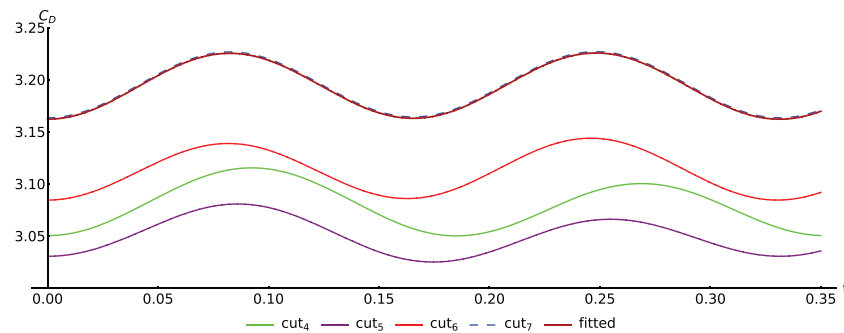


(A) Cut₇ configuration with computational mesh. (B) Fitted mesh, pre-adapted to the rigid body.

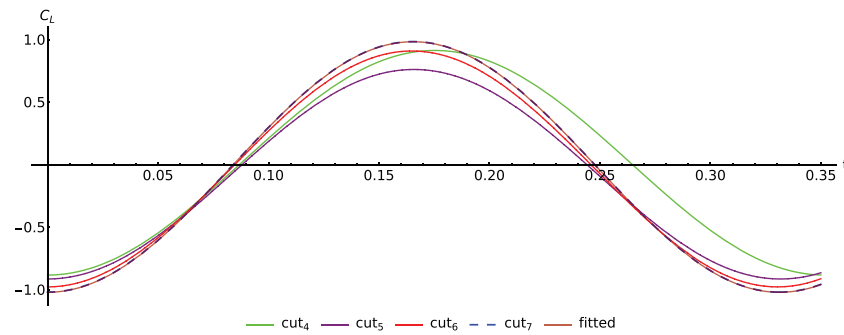
FIGURE 14 Portion of problem setting and computed velocity profile for CutFEM approach (A) and fitted mesh approximation (B) [Colour figure can be viewed at wileyonlinelibrary.com]

TABLE 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 _{I_n}	min(c_L)	max(c_L)	min(c_D)	max(c_D)	f_L
cut ₄	24,000	−0.881383	0.914682	3.05072	3.11567	2.84333
cut ₅	94,086	−0.912680	0.762557	3.03084	3.08099	3.01459
cut ₆	372,486	−0.975867	0.910129	3.08479	3.13906	3.02737
cut ₇	1,482,246	−1.01765	0.985596	3.16375	3.22723	3.01825
Adapted, body-fitted	502,464	−1.01899	0.984292	3.16245	3.22593	3.01844



(A) Drag coefficients c_D



(B) Lift coefficients c_L

FIGURE 15 Computed drag and lift coefficients for the example of Section 5.2 and different time discretization schemes with basic time step size $\tau = 5e - 3$ [Colour figure can be viewed at wileyonlinelibrary.com]

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, Figure 15 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 $t = 0$. 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 Figure 14 both meshes are shown. The diameter of a quadratic cut cell of the Cut₇ configuration is $4.530e-3$. The diameter of a rectangular cell of the body-fitted triangulation in the neighborhood of Ω_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 2 and Figure 15 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. Reference 44). This confirms the accuracy of the proposed CutFEM approach in the case of non-evolving domains.

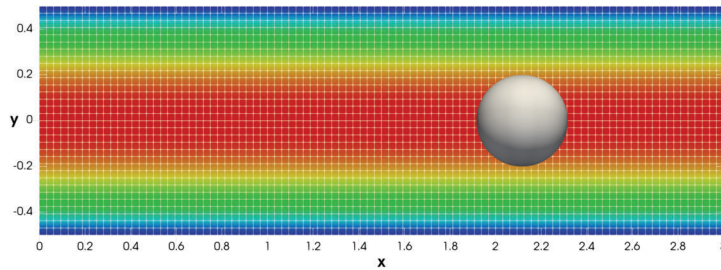


FIGURE 16 Problem setting and flow profile at $T = 40$ [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/doi/10.1002/nd.5074)]

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. Figure 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. Then we analyze if the Poiseuille flow is recovered by the CutFEM approach.

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

$$\mathbf{v}_p(x, y) = U_{\text{in}}(l_y^2 - y^2, 0)^\top \quad (36)$$

with $U_{\text{in}} = 1$. The initial value is $\mathbf{v}_0 = \mathbf{0}$. The extension of (36) to the entire pipe is the Poiseuille flow and satisfies the Navier–Stokes equations along with the pressure $p_p = -2\nu U_{\text{in}}(x - l_x)$. The rigid body is modeled by a circular domain of radius of $r = 0.2$ and an initial position of its center at $\mathbf{x}_r(0) = (1.545, 0)^\top$. The motion of the ball's center is prescribed by Equation (33), with $A = 0.8$ and $\omega = 0.2$. On the boundary of the rigid body the Dirichlet condition

$$\mathbf{v} = \mathbf{g}_r := \mathbf{v}_p, \quad (37)$$

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 (37). Clearly, even though the rigid body is moving inside the fluid by means of (33), its impact on the fluid flow is hidden by means of the boundary condition (37).

We compute fully discrete approximations in $(\mathbb{P}_1(I_n; H_h^2))^2 \times \mathbb{P}_1(I_n; H_h^1)$, for $n = 1, \dots, N$, 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 Figure 16. This solution is fully converged. In fact, the Poiseuille profile is nicely recovered by the simulation. Figure 17 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 (14). 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 (14).

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

$$\mathbf{g}_i(\mathbf{x}, t) = \begin{cases} (6ty(1-y), 0)^\top, & t \leq 1, \\ (6y(1-y), 0)^\top, & t > 1. \end{cases} \quad (38)$$

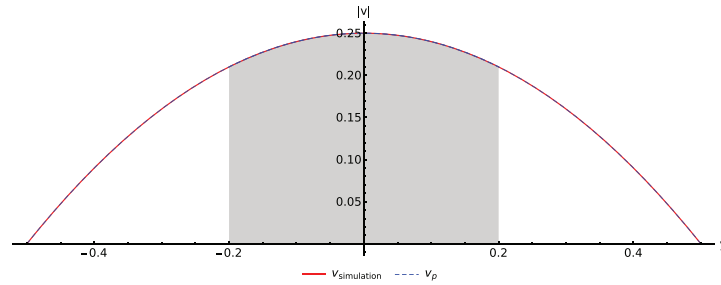
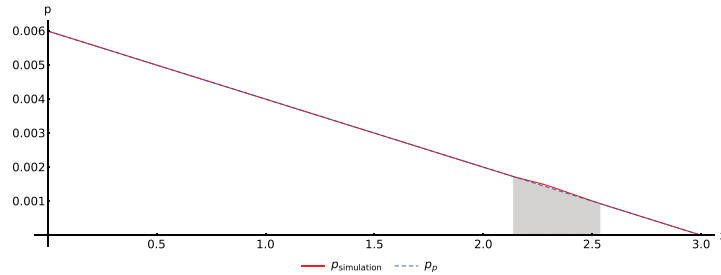
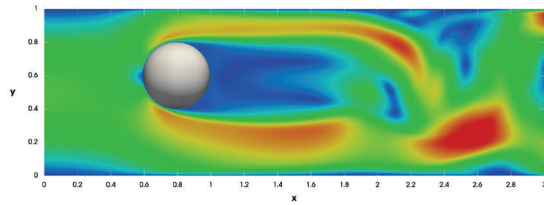
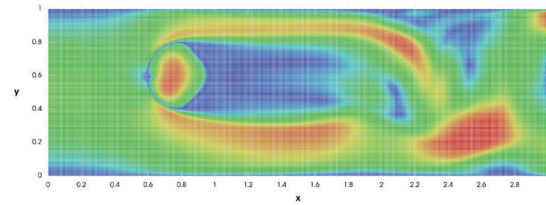
(A) Velocity along y -direction at $x = 2.34$ (B) Pressure along x -direction at $y = 0$

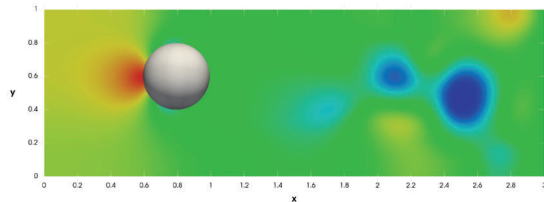
FIGURE 17 Cross section plots of velocity in y -direction and pressure in x -direction and Poiseuille profile at $T = 40$ with gray-shaded rigid body domain [Colour figure can be viewed at wileyonlinelibrary.com]



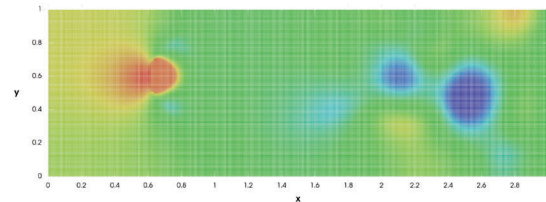
(A) Velocity field with overlay of moving rigid do-main.



(B) Velocity field with computational background mesh.



(C) Pressure field with overlay of moving rigid body.



(D) Pressure field with computational background mesh.

FIGURE 18 Solution of the CutFEM simulation at time $t = 10.12$ [Colour figure can be viewed at wileyonlinelibrary.com]

For the rigid body, a ball with a radius $r = 0.2$ and an initial position of its center at $\mathbf{x}_r(0) = (1.545, 0.6)^\top$ 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 Equation (33), with $A = 0.8$ and $\omega = 0.5$. On the boundary Γ_r^t of the ball, the no-slip condition $\mathbf{v} = \mathbf{v}_r = (A\omega \cos(\omega t), 0)^\top$ for a moving, rigid domain is applied, such that the fluid and ball velocity coincide on their interface, compare Reference 67 (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 $(\mathbb{P}_1(I_n; H_h^2))^2 \times \mathbb{P}_1(I_n; H_h^1)$. Figures 18 and 19 illustrate the computed profiles of velocity and the pressure at time $t = 10.12$ and $t = T$, respectively. The figures show that by the

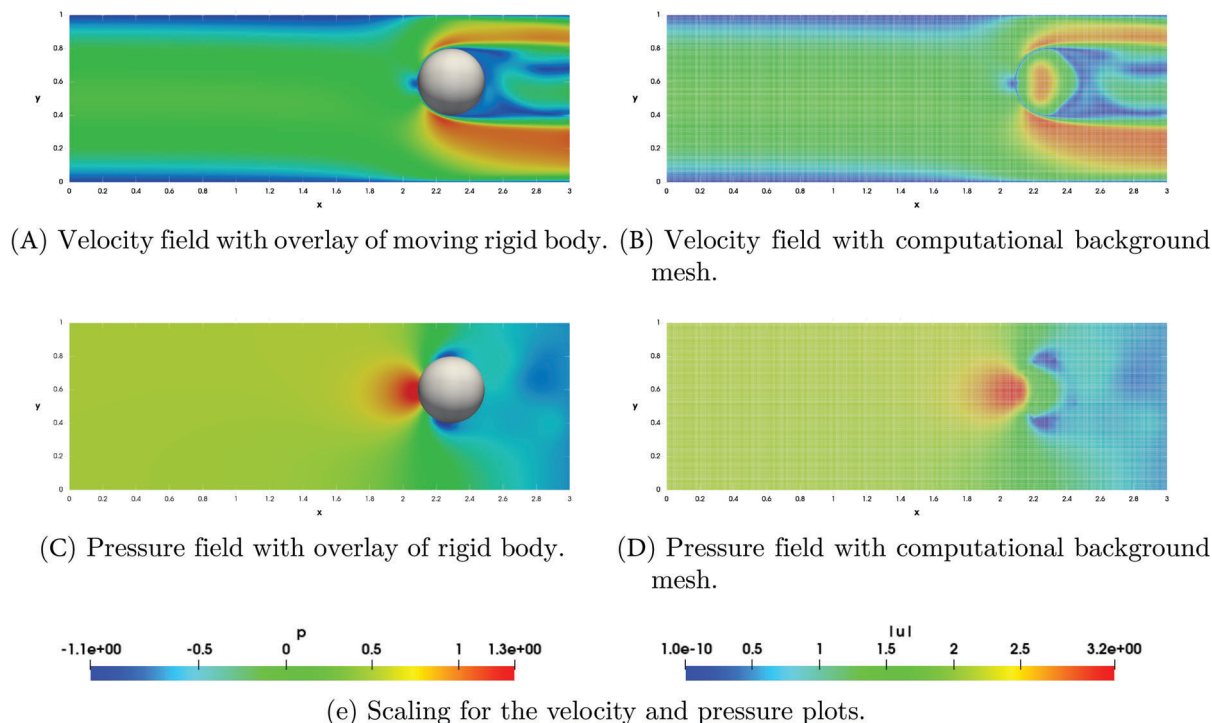


FIGURE 19 Solution of the CutFEM simulation at $t = T$ [Colour figure can be viewed at [wileyonlinelibrary.com](https://onlinelibrary.wiley.com/terms-and-conditions)]

application of the ghost penalty stabilization and extension of the discrete solution to the ghost rigid domain, defined by (14), lead to stable approximations of the flow problem on a background mesh. The color legend of Figure 19 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. Section 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 schemes were illustrated.

In this work, a parallel direct linear solver was applied for the sake of simplicity. The integration of the CutFEM methodology into our geometric multigrid preconditioner⁴³ for GMRES iterations for solving the Newton linearized Navier–Stokes equations is an ongoing work. In this work, the evolution in time of the flow domain’s boundaries was prescribed explicitly. For the future, more sophisticated applications in that the domains’ boundaries are no longer prescribed explicitly but are computed simultaneously (cf. e.g., Reference 28) will be focused. In particular, multi-physics systems like fluid-structure interaction with arising subproblems coupled with the fluid flow at an moving inner interface or free boundary value problems will be in the scope of our interest. Further, the proposed CutFEM techniques offer large potential for the simulation of flows in porous media with fractures or crack propagation. To enhance physical realism of the simulations, an extension of the proposed approach to three space dimensions is intended. The presented CutFEM approach offers the flexibility for its application on three-dimensional hexahedral meshes. Nevertheless, in three space dimensions the cut scenarios will be more diverse and involved and, possibly, demand for further generalizations like a

parametric representation of enclosed moving rigid bodies. We note that our approach exploits its strength for quadrilateral or hexahedral elements on Cartesian grids that are implemented in our finite element toolbox deal.II.⁴⁵ For simplicial meshes, different approaches, in particular for the integration over the cut cells, might be more advantageous.

ACKNOWLEDGEMENT

Open Access funding enabled and organized by Projekt DEAL.

DATA AVAILABILITY STATEMENT

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

REFERENCES

1. Lehrenfeld C, Olshanskii M. An Eulerian finite element method for PDEs in time-dependent domains. *ESAIM Math Model Numer Anal.* 2019;53:585-614.
2. von Wahl H, Richter T, Lehrenfeld C. An unfitted Eulerian finite element method for the time-dependent stokes problem on moving domains. *IMA J Numer Anal.* 2021;drab044.
3. Burman E, Frei S, Massing A. Eulerian time-stepping schemes for the non-stationary stokes equations on time-dependent domains; December 2020.
4. Winter M, Schott B, Massing A, Wall W. A Nitsche cut finite element method for the Oseen problem with general Navier boundary conditions. *Comput Methods Appl Mech Eng.* 2018;330:220-252.
5. Burman E, Hansbo P. Fictitious domain methods using cut elements: III. A stabilized Nitsche method for stokes' problem. *ESAIM Math Model Numer Anal.* 2014;48:859-874.
6. Schott B, Wall W. A new face-oriented stabilized XFEM approach for 2D and 3D incompressible Navier–Stokes equations. *Comput Methods Appl Mech Eng.* 2014;276:233-265.
7. Court S, Fournié M, Lozinski A. A fictitious domain approach for the stokes problem based on the extended finite element method. *Int J Numer Methods Fluids.* 2014;74:73-99.
8. Schott B, Ager C, Wall W. A monolithic approach to fluid-structure interaction based on a hybrid Eulerian-ALE fluid domain decomposition involving cut elements. *Int J Numer Methods Eng.* 2019;119:208-237.
9. Court S, Fournié M. A fictitious domain finite element method for simulations of fluid–structure interactions: the Navier–Stokes equations coupled with a moving solid. *J Fluids Struct.* 2015;55:398-408.
10. Burman E, Fernández MA, Frei S. A Nitsche-based formulation for fluid-structure interactions with contact. *ESAIM Math Model Numer Anal.* 2020;54:531-564.
11. Hughes TJ, Liu WK, Zimmermann TK. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Comput Methods Appl Mech Eng.* 1981;29:329-349.
12. Donea J, Giuliani S, Halleux J. An arbitrary Lagrangian-Eulerian finite element method for transient dynamic fluid-structure interactions. *Comput Methods Appl Mech Eng.* 1982;33:689-723.
13. Braess H, Wriggers P. Arbitrary Lagrangian Eulerian finite element analysis of free surface flow. *Comput Methods Appl Mech Eng.* 2000;190:95-109.
14. Wall WA, Gerstenberger A, Gammizter P, Förster C, Ramm E. Large deformation fluid-structure interaction – advances in ale methods and new fixed grid approaches. In: Bungartz H-J, Schäfer M, eds. *Fluid-Structure Interaction*. Vol 53. Springer; 2006:195-232.
15. Nazem M, Sheng D, Carter JP. Stress integration and mesh refinement for large deformation in geomechanics. *Int J Numer Methods Eng.* 2006;65:1002-1027.
16. Farinatti Aymone JL. Mesh motion techniques for the ALE formulation in 3D large deformation problems: MESH MOTION TECHNIQUES. *Int J Numer Methods Eng.* 2004;59:1879-1908.
17. Brenner A, Bänsch E, Bause M. A priori error analysis for finite element approximations of the Stokes problem on dynamic meshes. *IMA J Numer Anal.* 2014;34:123-146.
18. Baiges J, Codina R. The fixed-mesh ALE approach applied to solid mechanics and fluid–structure interaction problems. *Int J Numer Methods Eng.* 2010;81(12):1529-1557.
19. Nitsche J. 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.* 1971;36:9-15.
20. Burman E, Hansbo P. Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. *Appl Numer Math.* 2012;62:328-341.
21. Becker R. Mesh adaptation for Dirichlet flow control via Nitsche's method. *Commun Numer Methods Eng.* 2002;18(9):669-680.
22. Boiveau T, Burman E. A penalty-free Nitsche method for the weak imposition of boundary conditions in compressible and incompressible elasticity. *IMA J Numer Anal.* 2016;36:770-795.
23. Massing A, Larson MG, Logg A, Rognes ME. A stabilized Nitsche fictitious domain method for the stokes problem. *J Sci Comput.* 2014;61:604-628.
24. Anselmann M, Bause M. Higher order Galerkin–collocation time discretization with Nitsche's method for the Navier–Stokes equations. *Math Comput Simul.* 2020;S0378475420303827.

25. Xie H, Li Z, Qiao Z. A finite element method for elasticity interface problems with locally modified triangulations. *Int J Numer Anal Model*. 2011;8(2):189-200.
26. Gawlik ES, Lew AJ. High-order finite element methods for moving boundary problems with prescribed boundary evolution. *Comput Methods Appl Mech Eng*. 2014;278:314-346.
27. Carraro T, Wetterauer S. On the implementation of the extended finite element method (XFEM) for interface problems. *Arch Numer Softw*. 2016;4(2):1-23.
28. Frei S, Richter T, Wick T. LocModFE: locally modified finite elements for approximating interface problems in deal.II. *Softw Impacts*. 2021;8:100070.
29. Massing A, Larson MG, Logg A. Efficient implementation of finite element methods on nonmatching and overlapping meshes in three dimensions. *SIAM J Sci Comput*. 2013;35:C23-C47.
30. Sudhakar Y, Moitinho de Almeida J, Wall WA. An accurate, robust, and easy-to-implement method for integration over arbitrary polyhedra: application to embedded interface methods. *J Comput Phys*. 2014;273:393-415.
31. Burman E, Fernández MA. Continuous interior penalty finite element method for the time-dependent Navier–Stokes equations: space discretization and convergence. *Numer Math*. 2007;107:39-77.
32. 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, Springer-Verlag, 2nd, 2008.
33. Burman E. Ghost penalty. *Comptes Rendus Mathématique*. 2010;348:1217-1220.
34. Burman E, Fernández MA. An unfitted Nitsche method for incompressible fluid–structure interaction using overlapping meshes. *Comput Methods Appl Mech Eng*. 2014;279:497-514.
35. Hansbo P, Larson MG, Zahedi S. A cut finite element method for coupled bulk-surface problems on time-dependent domains. *Comput Methods Appl Mech Eng*. 2016;307:96-116.
36. Burman E, Elferverson D, Hansbo P, Larson MG, Larsson K. A cut finite element method for the Bernoulli free boundary value problem. *Comput Methods Appl Mech Eng*. 2017;317:598-618.
37. Frachon T, Zahedi S. A cut finite element method for incompressible two-phase Navier–Stokes flows. *J Comput Phys*. 2019;384:77-98.
38. Burman E, Claus S, Hansbo P, Larson MG, Massing A. CutFEM: discretizing geometry and partial differential equations. *Int J Numer Methods Eng*. 2015;104:472-501.
39. Preuß J. *Higher Order Unfitted Isoparametric Space-Time FEM on Moving Domains*. Master's thesis. University of Göttingen; February 2018.
40. Zahedi S. A space-time cut finite element method with quadrature in time. In: Bordas SPA, Burman E, Larson MG, Olshanskii MA, eds. *Geometrically Unfitted Finite Element Methods and Applications*. Vol 121. Springer International Publishing; 2017:281-306.
41. Hussain S, Schieweck F, Turek S. An efficient and stable finite element solver of higher order in space and time for nonstationary incompressible flow. *Int J Numer Methods Fluids*. 2013;73:927-952.
42. John V, Tobiska L. Numerical performance of smoothers in coupled multigrid methods for the parallel solution of the incompressible Navier–Stokes equations. *Int J Numer Methods Fluids*. 2000;33(4):453-473.
43. Anselmann M, Bause M. A geometric multigrid method for space-time finite element discretizations of the Navier-Stokes equations and its application to 3D flow simulation; Submitted, July 2021.
44. Schäfer M, Turek S, Durst F, Krause E, Rannacher R. Benchmark computations of laminar flow around a cylinder. In: Hirschel EH, Fujii K, van Leer B, et al., eds. *Flow Simulation with High-Performance Computers II*. Vol 48. Vieweg+Teubner Verlag; 1996:547-566.
45. Arndt D, Bangerth W, Blais B, et al. The deal. II library, version 9.2. *J Numer Math*. 2020;28:131-146.
46. The Trilinos Project Team. The Trilinos project website; July 2020.
47. Heywood JG, Rannacher R, Turek S. Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations. *Int J Numer Methods Fluids*. 1996;22(5):325-352.
48. R. Salvi, “On the Navier-Stokes equations in non-cylindrical domains: on the existence and regularity,” *Math Z*, vol. 199, pp. 153–170, June 1988.
49. Bock DN. On the Navier-Stokes equations in noncylindrical domains. *J Differ Equ*. 1977;25:151-162.
50. Heywood JG, Rannacher R. Finite element approximation of the nonstationary Navier–Stokes problem. I. regularity of solutions and second-order error estimates for spatial discretization. *SIAM J Numer Anal*. 1982;19:275-311.
51. Bause M. On optimal convergence rates for higher-order Navier–Stokes approximations. I. Error estimates for the spatial discretization. *IMA J Numer Anal*. 2005;25:812-841.
52. Sonner F, Richter T. Second order pressure estimates for the Crank–Nicolson discretization of the incompressible Navier–Stokes equations. *SIAM J Numer Anal*. 2020;58:375-409.
53. Alphonse A, Elliott C, Stinner B. An abstract framework for parabolic PDEs on evolving spaces. *Port Math*. 2015;72(1):1-46.
54. John V. *Finite Element Methods for Incompressible Flow Problems*. Springer Series in Computational Mathematics. Vol 51. Springer International Publishing; 2016.
55. Benk J, Ulbrich M, Mehl M. The Nitsche method of the Navier Stokes equations for immersed and moving boundaries. Proceedings of the 7th International Conference on Computational Fluid Dynamics; 2012.
56. Anselmann M, Bause M. Numerical convergence of discrete extensions in a spacetime finite element, fictitious domain method for the Navier-Stokes equations; submitted to PAMM, July 2021.
57. Schott B, Ager C, Wall WA. Monolithic cut finite element–based approaches for fluid-structure interaction. *Int J Numer Methods Eng*. 2019;119(8):757-796.

58. Ager C, Schott B, Winter M, Wall W. A Nitsche-based cut finite element method for the coupling of incompressible fluid flow with poroelasticity. *Comput Methods Appl Mech Eng*. 2019;351:253-280.
59. Massing A, Schott B, Wall W. A stabilized Nitsche cut finite element method for the Oseen problem. *Comput Methods Appl Mech Eng*. 2018;328:262-300.
60. Hussain S, Schieweck F, Turek S. Higher order Galerkin time discretizations and fast multi-grid solvers for the heat equation. *J Numer Math*. 2011;19:41-61.
61. Köcher U, Bause M. Variational space-time methods for the wave equation. *J Sci Comput*. 2014;61:424-453.
62. Anselmann M, Bause M. Numerical study of Galerkin-collocation approximation in time for the wave equation. In: Dörfler W, Hochbruck M, Hundertmark D, et al., eds. *Mathematics of Wave Phenomena*. Springer International Publishing; 2020:15-36.
63. Pawlowski RP, Simonis JP, Walker HF, Shadid JN. Inexact Newton Dogleg methods. *SIAM J Numer Anal*. 2008;46:2112-2132.
64. Li XS, Demmel JW. SuperLU_DIST: a scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Trans Math Softw*. 2003;29:110-140.
65. Fehn N, Heinz J, Wall WA, Kronbichler M. High-order arbitrary Lagrangian-Eulerian discontinuous Galerkin methods for the incompressible Navier-Stokes equations. *J Comput Phys*. 2021;430:110040.
66. Donea J, Huerta A, Ponthot J-P, Rodríguez-Ferran A. Arbitrary Lagrangian-Eulerian methods. In: Stein E, de Borst R, Hughes TJR, eds. *Encyclopedia of Computational Mechanics*. 2nd ed. John Wiley & Sons, Ltd; 2017:1-23.
67. Richter T. *Fluid-Structure Interactions: Models, Analysis and Finite Elements*. Lecture Notes in Computational Science and Engineering. Vol 118. Springer; 2017.

How to cite this article: Anselmann M, Bause M. Cut finite element methods and ghost stabilization techniques for space-time discretizations of the Navier-Stokes equations. *Int J Numer Meth Fluids*. 2022;94(7):775-802. doi: 10.1002/fld.5074