The tetrahedral finite cell method for fluids: Immersogeometric analysis of turbulent flow around complex geometries

Fei Xu[a], Dominik Schillinger[b], David Kamensky[c], Vasco Varduhn[b], Chenglong Wang[a], Ming-Chen Hsu[a,*]

[a]*Department of Mechanical Engineering, Iowa State University, 2025 Black Engineering, Ames, IA 50011, USA*
[b]*Department of Civil, Environmental, and Geo- Engineering, University of Minnesota, 500 Pillsbury Drive S.E., Minneapolis, MN 55455, USA*
[c]*Institute for Computational Engineering and Sciences, The University of Texas at Austin, 201 East 24th St, Stop C0200, Austin, TX 78712, USA*

Abstract

We present a tetrahedral finite cell method for the simulation of incompressible flow around geometrically complex objects. The method immerses such objects into non-boundary-fitted meshes of tetrahedral finite elements and weakly enforces Dirichlet boundary conditions on the objects' surfaces. Adaptively-refined quadrature rules faithfully capture the flow domain geometry in the discrete problem without modifying the non-boundary-fitted finite element mesh. A variational multiscale formulation provides accuracy and robustness in both laminar and turbulent flow conditions. We assess the accuracy of the method by analyzing the flow around an immersed sphere for a wide range of Reynolds numbers. We show that quantities of interest such as the drag coefficient, Strouhal number and pressure distribution over the sphere are in very good agreement with reference values obtained from standard boundary-fitted approaches. We place particular emphasis on studying the importance of the geometry resolution in intersected elements. Aligning with the immersogeometric concept, our results show that the faithful representation of the geometry in intersected elements is critical for accurate flow analysis. We demonstrate the potential of our proposed method for high-fidelity industrial scale simulations by performing an aerodynamic analysis of an agricultural tractor.

Keywords: Immersed method, Complex geometry, Immersogeometric finite elements, Geometric accuracy in intersected elements, Weakly enforced boundary conditions, Tetrahedral finite cell method

1. Introduction

Immersed methods approximate the solution of boundary value problems on analysis meshes that do not necessarily conform to the boundary of the domain. Such methods have greater geometric flexibility than their boundary-fitted counterparts and circumvent the meshing obstacles that frequently impede analysis of problems posed on geometrically-complex domains. In the context of computational fluid dynamics (CFD), immersed methods have a long tradition that dates back at least to the Immersed Boundary Method developed by Peskin [1] to simulate cardiac mechanics and associated blood flow. Since then, the body of research on immersed methods area has undergone tremendous growth [2–5].

In the context of finite elements [6], several variants of immersed methods for fluids have been explored over the last decade. Löhner et al. [7–9] adapted kinetic and kinematic enforcement of boundary conditions used in immersed boundary methods [5] for use in adaptive nodal finite element grids. Glowinski et al. [10–12] simulated viscous flow interacting with rigid particles by forcing the rigid body motion in each particle sub-domain onto the overlapping fluid field via a distributed Lagrange multiplier field. Zhang, Liu and coworkers [13–16] proposed the Immersed Finite Element Method (IFEM) to use a flexible Lagrangian solid mesh that moves on top of a background Eulerian fluid mesh. This circumvented the major limitation of the immersed boundary method where the fiber-like one-dimentional structure carries mass but does not occupy volume, and opened the door to the immersed methods for fluid–structure interaction (FSI) problems. The concept of IFEM was extended recently by Casquero et al. [17] to use Non-Uniform Rational B-Splines (NURBS) as the basis functions to improve the robustness and accuracy of the immersed method for FSI.

In addition, several researchers designed immersed methods that resolve immersed boundaries and introduce weak coupling schemes for velocity and stress fields directly at the interface. Baaijens [18] and Parussini et al. [19, 20] combined the fictitious domain approach with Lagrange multiplier fields at the interface for immersed thin and volumetric structures. Gerstenberger, Wall and coworkers [21–23] combined Lagrange multiplier fields with interface enrichments of the velocity and pressure fields in the sense of the extended finite element method to ensure the separation of physical and fictitious domains. Rüberg and Cirak [24, 25] combined weak Nitsche-type coupling methods at the interface with Cartesian B-spline finite elements for moving boundary and FSI problems. Several groups also started to work on non-boundary-fitted FSI methods, where both the fluid and the solid domains are immersed [26–30].

This work presents an immersed method for solving incompressible flow problems on unstructured tetrahedral finite el-

*Corresponding author
Email address: jmchsu@iastate.edu (Ming-Chen Hsu)

The final publication is available at Computers & Fluids via http://dx.doi.org/10.1016/j.compfluid.2015.08.027

ement meshes. The proposed method combines a variational multiscale (VMS) formulation of incompressible flow [31–34], consistent weak enforcement of boundary conditions [35–38], and a geometrically-accurate representation of the fluid domain in the integration of the variational problem on elements that straddle the domain boundary. We emphasize the implications of the latter, highlighting the importance of accurately describing the geometry in intersected elements for obtaining accurate flow solutions. Several studies have shown that inaccurate quadrature in elements cut by domain boundaries introduces a geometry error, which prevents higher-order accuracy of immersed methods [39, 40]. Influenced by isogeometric analysis [41, 42], which has recently drawn broader recognition to the importance of eliminating geometric errors, we follow our previous work [43] in denoting immersed methods that accurately represent the geometry of the domain as *immersogeometric* methods.

A pioneering instantiation of the immersogeometric concept is the Finite Cell Method (FCM), introduced by Parvizian et al. [44] and Düster et al. [45]. The FCM represents the geometry of the domain in intersected elements by adaptive quadrature points, such that the geometric accuracy can be increased by adding additional levels of quadrature points, if needed. The adaptive quadrature scheme is based on the decomposition of each intersected element into sub-cells that can be efficiently organized in hierarchical tree data structures. Although this strategy leads to an increased number of quadrature points in intersected elements, its implementation is extremely robust and flexible; sub-cell decomposition can operate with almost any geometric model, ranging from boundary representations in computer aided geometric design to voxel representations obtained from medical imaging technologies [46].

Since its inception, significant efforts have been invested to further develop the FCM. Technical improvements include the weak imposition of boundary and coupling conditions [47, 48], local refinement schemes [49–53], and improved quadrature rules for intersected elements [39, 40, 54]. Furthermore, the FCM has been successfully applied for large deformation analysis [55, 56], thermoelasticity [57], homogenization [58], bone mechanics [59], topology optimization [60], and elastodynamics and wave propagation [61–63]. A concise summary of the FCM and related developments and applications can be found in the recent review article by Schillinger and Ruess [46]. In addition, there exists an open-source MATLAB code[1] that provides an easy-to-handle starting point for running numerical tests with the FCM [64].

Most prior work on the FCM used structured meshes of hexahedral elements, but this is not a necessary feature of the FCM. Varduhn et al. [65] recently applied the FCM with unstructured meshes of tetrahedral elements. The present contribution extends the tetrahedral FCM of [65] to simulations of incompressible flow, where the flexibility of unstructured tetrahedral meshes is useful for boundary layer refinement. One key feature of this study is that for all example problems considered, we employ a boundary-fitted finite element method based on the VMS formulation with weakly enforced boundary conditions to compute reference solutions. Corresponding boundary-fitted and immersogeometric analyses use tetrahedral meshes with the same refinement pattern and approximately the same number of degrees of freedom. Based on this comparison, we show that our immersogeometric method achieves results that are in very good accordance with standard boundary-fitted results in terms of key quantities of interest.

This paper is organized as follows. In Section 2, we describe the precise variational problem under consideration and our discrete formulation of it. Section 3 details the implementation of the key technical components, including a tree based element decomposition for geometrically accurate quadrature in intersected elements and an efficient point-location query for inside-outside tests. Section 4 focuses on the canonical benchmark of the flow around a sphere at Reynolds numbers of 100, 300 and 3700. We compare the results of immersogeometric analysis to boundary-fitted reference computations of this benchmark problem to demonstrate that our method accurately computes quantities of interest. Section 5 presents a detailed account of the accuracy of our method for the flow analysis of the full-scale tractor, illustrating the potential of immersogeometric analysis for high-fidelity aerodynamic analysis of industrial-scale problems. Section 6 draws conclusions and motivates future work.

2. Variational formulation and discretization

In this section, we summarize the variational formulation of the Navier–Stokes equations of incompressible flow and its spatial and temporal discretizations. We also briefly review the variational multiscale (VMS) method and the weak enforcement of boundary conditions. Note that the framework reviewed in this section equally holds for immersogeometric and boundary-fitted finite element methods.

2.1. Governing equations of incompressible flow

Let Ω (subsets of $\mathbb{R}^d, d \in \{2, 3\}$) denote the spatial domain and Γ be its boundary. The incompressible Navier–Stokes equations on Ω can be written as

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\mathbf{u} - \mathbf{f}\right) - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} = \mathbf{0}\,, \tag{1}$$

$$\boldsymbol{\nabla}\cdot\mathbf{u} = 0\,, \tag{2}$$

where ρ, $\mathbf{u}$, and $\mathbf{f}$ are the density of the fluid, the velocity of the fluid and the external force per unit mass, respectively. The stress and strain-rate tensors are defined respectively as

$$\boldsymbol{\sigma}(\mathbf{u}, p) = -p\,\mathbf{I} + 2\mu\,\boldsymbol{\varepsilon}(\mathbf{u})\,, \tag{3}$$

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \frac{1}{2}\left(\boldsymbol{\nabla}\mathbf{u} + \boldsymbol{\nabla}\mathbf{u}^T\right)\,, \tag{4}$$

where p is the pressure, $\mathbf{I}$ is the identity tensor and μ is the dynamic viscosity. The problem (1)–(4) is accompanied by suitable boundary conditions, defined on the boundary of the fluid

[1] http://fcmlab.cie.bgu.tum.de

domain, $\Gamma = \Gamma^D \cup \Gamma^N$:

$$\mathbf{u} = \mathbf{g} \qquad \text{on } \Gamma^D\,, \tag{5}$$

$$-p\,\mathbf{n} + 2\mu\,\boldsymbol{\varepsilon}(\mathbf{u})\,\mathbf{n} = \mathbf{h} \qquad \text{on } \Gamma^N\,, \tag{6}$$

where $\mathbf{g}$ denotes the prescribed velocity at the Dirichlet boundary Γ^D, $\mathbf{h}$ is the traction vector at the Neumann boundary Γ^N, and $\mathbf{n}$ is the outward unit normal.

2.2. Semi-discrete variational multiscale formulation

Consider a collection of disjoint elements $\{\Omega^e\}$, $\cup_e \Omega^e \subset \mathbb{R}^d$, with closures covering the fluid domain: $\Omega \subset \cup_e \overline{\Omega^e}$. Note that Ω^e is not necessarily a subset of Ω. Let $\mathcal{V}_u^h$ and $\mathcal{V}_p^h$ be the discrete velocity and pressure spaces of functions supported on these elements. The strong problem (1)–(6) may be recast in a weak form and posed over these discrete spaces to produce the following semi-discrete problem: Find $\mathbf{u}^h \in \mathcal{V}_u^h$ and $p^h \in \mathcal{V}_p^h$ such that for all $\mathbf{w}^h \in \mathcal{V}_u^h$ and $q^h \in \mathcal{V}_p^h$:

$$B^{\text{VMS}}\left(\{\mathbf{w}^h, q^h\}, \{\mathbf{u}^h, p^h\}\right) - F^{\text{VMS}}\left(\{\mathbf{w}^h, q^h\}\right) = 0\,. \tag{7}$$

The bilinear form B^{VMS} and the load vector F^{VMS} are given as

$$\begin{aligned}
B^{\text{VMS}}\left(\{\mathbf{w}^h, q^h\}, \{\mathbf{u}^h, p^h\}\right) &= \int_\Omega \mathbf{w}^h \cdot \rho\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla}\mathbf{u}^h\right) d\Omega \\
&+ \int_\Omega \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}\left(\mathbf{u}^h, p^h\right)\, d\Omega + \int_\Omega q^h\, \boldsymbol{\nabla}\cdot\mathbf{u}^h\, d\Omega \\
&- \sum_e \int_{\Omega^e\cap\Omega} \left(\mathbf{u}^h \cdot \boldsymbol{\nabla}\mathbf{w}^h + \frac{\boldsymbol{\nabla} q^h}{\rho}\right)\cdot \mathbf{u}'\, d\Omega \\
&- \sum_e \int_{\Omega^e\cap\Omega} p'\,\boldsymbol{\nabla}\cdot\mathbf{w}^h\, d\Omega \\
&+ \sum_e \int_{\Omega^e\cap\Omega} \mathbf{w}^h \cdot (\mathbf{u}'\cdot\boldsymbol{\nabla}\mathbf{u}^h)\, d\Omega \\
&- \sum_e \int_{\Omega^e\cap\Omega} \frac{\boldsymbol{\nabla}\mathbf{w}^h}{\rho} : (\mathbf{u}'\otimes\mathbf{u}')\, d\Omega \\
&+ \sum_e \int_{\Omega^e\cap\Omega} \left(\mathbf{u}'\cdot\boldsymbol{\nabla}\mathbf{w}^h\right)\overline{\tau}\cdot\left(\mathbf{u}'\cdot\boldsymbol{\nabla}\mathbf{u}^h\right)\, d\Omega\,,
\end{aligned} \tag{8}$$

and

$$F^{\text{VMS}}\left(\{\mathbf{w}^h, q^h\}\right) = \int_\Omega \mathbf{w}^h\cdot\rho\,\mathbf{f}\, d\Omega + \int_{\Gamma^N} \mathbf{w}^h\cdot\mathbf{h}\, d\Gamma\,, \tag{9}$$

where $\mathbf{u}'$ is defined as

$$\mathbf{u}' = -\tau_{\text{M}}\left(\rho\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{u}^h - \mathbf{f}\right) - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}\left(\mathbf{u}^h, p^h\right)\right)\,, \tag{10}$$

and p' is given by

$$p' = -\rho\,\tau_{\text{C}}\,\boldsymbol{\nabla}\cdot\mathbf{u}^h\,. \tag{11}$$

Equations (8)–(11) emanate from the VMS formulation of the Navier–Stokes equations of incompressible flow [34]. The terms integrated over element interiors would not appear in a Galerkin method based on the canonical weak form of incompressible Navier–Stokes. These additional terms may be interpreted both as stabilization and as a turbulence model [32–34, 66–69]. The stabilization parameters are

$$\tau_{\text{M}} = \left(\frac{C_t}{\Delta t^2} + \mathbf{u}\cdot\mathbf{G}\,\mathbf{u} + C_I\,\nu^2\,\mathbf{G}:\mathbf{G}\right)^{-1/2}\,, \tag{12}$$

$$\tau_{\text{C}} = (\tau_{\text{M}}\,\text{tr}\,\mathbf{G})^{-1}\,, \tag{13}$$

$$\overline{\tau} = (\mathbf{u}'\cdot\mathbf{G}\,\mathbf{u}')^{-1/2}\,, \tag{14}$$

where Δt is the time-step size, C_I is a positive constant derived from an appropriate element-wise inverse estimate [70–72], $\nu = \mu/\rho$ is the kinematic viscosity, $\mathbf{G}$ generalizes the notion of element size to physical elements mapped from a parametric parent element by $\mathbf{x}(\xi)$:

$$G_{ij} = \sum_{k=1}^{d} \frac{\partial \xi_k}{\partial x_i}\frac{\partial \xi_k}{\partial x_j}\,, \tag{15}$$

$\text{tr}\,\mathbf{G}$ is the trace of $\mathbf{G}$, and the parameter C_t is typically equal to 4 [34, 68].

Note that we allow the fluid domain boundary to intersect the finite elements. Hence, each intersected element consists of a fluid portion of Ω^e, over which the formulation is integrated, and a portion of Ω^e outside of the fluid domain, over which the integration is discarded. The boundary Γ is discretized into a number of surface elements. In a boundary-fitted method, these boundary elements naturally arise as the surfaces of finite elements adjacent to the boundary of the fluid domain. In an immersogeometric method, surface elements are defined independently of the background finite element mesh.

2.3. Variationally consistent weak boundary conditions

The standard way of imposing Dirichlet boundary conditions in Eq. (7) is to enforce them strongly by ensuring that they are satisfied by all trial solution functions. This is not feasible in immersed methods (and not always desirable in boundary-fitted ones (see, e.g., [38])). We replace the strong enforcement by weakly enforced Dirichlet boundary conditions in the sense of Nitsche's method [73] proposed by Bazilevs et al. [35–37]. The semi-discrete problem becomes

$$\begin{aligned}
&B^{\text{VMS}}\left(\{\mathbf{w}^h, q^h\}, \{\mathbf{u}^h, p^h\}\right) - F^{\text{VMS}}\left(\{\mathbf{w}^h, q^h\}\right) \\
&- \int_{\Gamma^D} \mathbf{w}^h\cdot\left(-p^h\,\mathbf{n} + 2\mu\,\boldsymbol{\varepsilon}(\mathbf{u}^h)\,\mathbf{n}\right)\, d\Gamma \\
&- \int_{\Gamma^D} \left(2\mu\,\boldsymbol{\varepsilon}(\mathbf{w}^h)\,\mathbf{n} + q^h\,\mathbf{n}\right)\cdot\left(\mathbf{u}^h - \mathbf{g}\right)\, d\Gamma \\
&- \int_{\Gamma^{D,-}} \mathbf{w}^h\cdot\rho\left(\mathbf{u}^h\cdot\mathbf{n}\right)\left(\mathbf{u}^h - \mathbf{g}\right)\, d\Gamma \\
&+ \int_{\Gamma^D} \tau_{\text{TAN}}^B\left(\mathbf{w}^h - \left(\mathbf{w}^h\cdot\mathbf{n}\right)\mathbf{n}\right)\cdot\left(\left(\mathbf{u}^h - \mathbf{g}\right) - \left(\left(\mathbf{u}^h - \mathbf{g}\right)\cdot\mathbf{n}\right)\mathbf{n}\right)\, d\Gamma \\
&+ \int_{\Gamma^D} \tau_{\text{NOR}}^B\left(\mathbf{w}^h\cdot\mathbf{n}\right)\left(\left(\mathbf{u}^h - \mathbf{g}\right)\cdot\mathbf{n}\right)\, d\Gamma \;=\; 0\,.
\end{aligned} \tag{16}$$

In the above equation, $\Gamma^{D,-}$ is the *inflow* part of Γ^D, on which $\mathbf{u}^h \cdot \mathbf{n} < 0$. The term integrated over $\Gamma^{D,-}$ increases the stability of the formulation without breaking variational consistency [35]. τ^B_{TAN} and τ^B_{NOR} are stabilization parameters that act on the tangential and normal components of the velocity at the boundary, respectively. They need to be chosen element-wise as a compromise between the following two requirements. If they are too large, they assume a penalty-type character, affecting the conditioning of the stiffness matrix and overshadowing the variational consistency. If they are too small, the solution of Eq. (16) is unstable. Bazilevs et al. [35–37] indicate that the stabilization parameters should scale like $\tau^B_{(\cdot)} = C^B_I \mu / h$, where h has dimensions of length and indicates the element size at the boundary and C^B_I is a dimensionless constant. In immersogeometric methods, the definitions of h and C^B_I depend on how the boundary intersects each element. References [48, 74] describe a method for computing optimal stabilization parameters in each intersected element. In this work, for simplicity, we use uniform values of τ^B_{TAN} and τ^B_{NOR}. These uniform values typically need to be higher than the element-wise stabilization parameters computed using the methods of [48, 74]. Overestimation of τ^B_{TAN} may lead to excessive penalization of flow slipping along the boundary. Although maybe counter-intuitive at first sight, some violation of the no-slip boundary condition is in fact desirable, in particular for coarse meshes, as it imitates the presence of a boundary layer [35–38, 75]. This allows for an accurate overall flow solution, even if the mesh size in wall-normal direction is relatively large. Also, over-penalization of solution differences between non-matching meshes tends to produce oscillatory coupling forces, as is well-known in computational contact mechanics [76–80]. High-fidelity surface traction calculations would therefore likely benefit from the methods of references [48, 74]. In the present work, though, we are principally interested in net forces on structures and turbulent flow characteristics, for which we found uniform penalty constants sufficient.

Remark 1. For immersogeometric methods, weakly enforced boundary conditions are particularly attractive as the additional Nitsche terms in Eq. (16) are formulated independently of the mesh. In contrast to strong enforcement, which relies on boundary-fitted meshes to impose Dirichlet boundary conditions on the discrete solution space, the Nitsche terms in Eq. (16) also hold for intersected elements, where the domain boundary does not coincide with element boundaries. All that is needed is a separate discretization of the domain boundary with independent boundary segments whose position in intersected elements is known or can be determined.

2.4. Time integration and iterative solution methods

We complete the discretization of Eq. (16) by a time integration scheme from the family of generalized-α integrators. Generalized-α methods were introduced by Chung and Hulbert [81] for structural dynamics and later extended to the unsteady Navier–Stokes problem by Jansen et al. [82]. The subset of generalized-α methods used in the current work is parameterized by a single number, ρ_∞, where $0 \leq \rho_\infty \leq 1$ (see Bazilevs et al. [83] for details). Following Bazilevs et al. [34], we use $\rho_\infty = 0.5$ for all computations presented in this paper. The generalized-α scheme is implicit and requires solution of a nonlinear algebraic problem at each time step. We directly apply Newton–Raphson iterations (with an approximate tangent) to converge the residual of this algebraic problem. For each Newton–Raphson iteration, the linear system is solved using a block-diagonal preconditioned GMRES method [84, 85].

3. Implementation of the tetrahedral finite cell method

The main challenge entailed by non-boundary-fitted meshes is the geometrically accurate evaluation of volume and surface integrals in intersected elements. These integrals directly emanate from the variational formulation (16). Our immersogeometric method largely draws on the FCM, which is briefly reviewed first. Specializing to tetrahedral elements, we detail the basic technology components, which are a volume quadrature method based on recursive subdivision of intersected elements and a surface quadrature method that uses an independent surface mesh. In addition, we briefly describe the implementation of an efficient point location query to determine whether a quadrature point is located inside or outside of the fluid domain. We finally outline an efficient workflow for the generation of immersogeometric meshes that combines our quadrature methods with an open-source mesh generator and locally refined boundary layers.

3.1. The finite cell method

The FCM is a technique for solving partial differential equations posed on complex geometries. For a summary of recent developments, we refer the interested reader to [46]. The FCM is based on the fictitious domain concept illustrated in Fig. 1. Its main idea is to extend the original fluid domain to a more tractable shape, e.g., a rectangular prism bounding the original domain. The FCM discretizes the embedding domain into

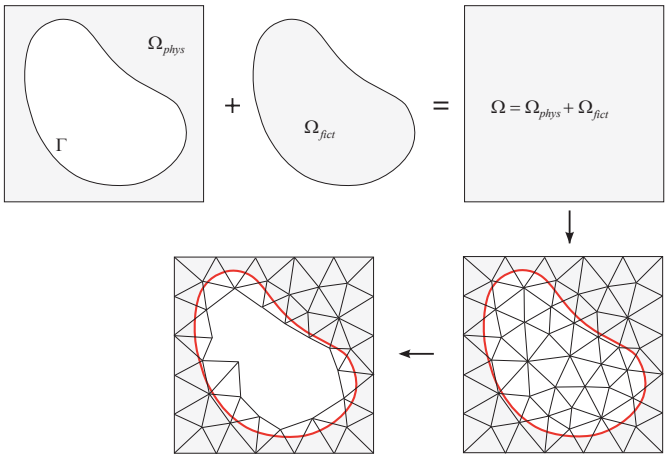


Figure 1: The physical domain of interest Ω_{phys} is extended by the fictitious domain Ω_{fict} into an embedding domain Ω to allow easy meshing of complex geometries. Elements without support in Ω_{phys} can be discarded from the mesh, since they do not contribute to the solution fields in the physical domain.

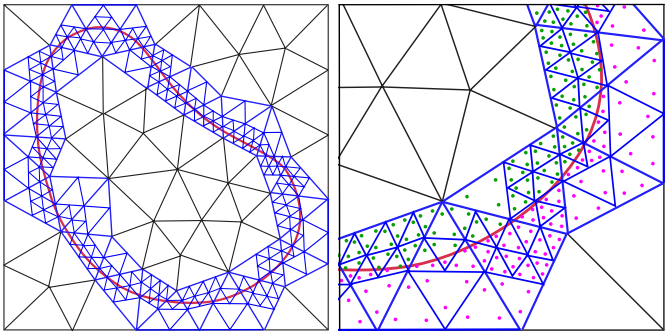


Figure 2: Quadrature scheme based on adaptive sub-cells (blue lines). Quadrature points within the fluid domain (marked in pink) are used in the numerical integration. Quadrature points outside (marked in green) are discarded.

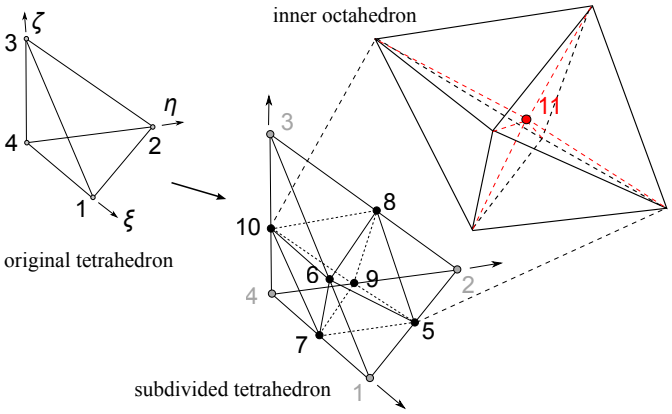


Figure 3: The rule used to subdivide a tetrahedron into sub-tetrahedra.

elements irrespective of the geometric boundary of potentially complex embedded objects. This introduces elements that are intersected by the geometric boundary, which creates complex integration domains in intersected elements. For Cartesian elements, Düster et al. [45] describe a method of automatically generating quadrature rules for finite cell computations by dividing intersected elements into sub-cells and applying standard quadrature rules within the sub-cells.

Our immersogeometric approach adapts the sub-cell based adaptive quadrature scheme of the Cartesian FCM to the tetrahedral case. Based on this scheme, we are able to evaluate arbitrary integration domains in intersected tetrahedral elements that arise in non-boundary-fitted discretizations of Eq. (8). The basic concept is based on the increase of quadrature points around geometric boundaries in each intersected cell, so that arbitrary integration domains that emanate from the intersecting boundary can be taken into account accurately. This is achieved by recursively splitting intersected cells into sub-tetrahedra. At each level, only those sub-tetrahedra that are intersected by the boundary are further split. This procedure leads to an aggregation of sub-tetrahedra of finer levels along the intersecting boundary. For each of the sub-tetrahedra, the standard 4-point quadrature rule for linear tetrahedral elements is applied. This keeps the amount of quadrature points per sub-tetrahedron constant and allows an easy calculation of the weights and local coordinates of the recursive quadrature points. For clarity, we illustrate the quadrature scheme based on adaptive sub-cells for triangles in 2D in Fig. 2. We emphasize that splitting is performed on the quadrature level only and does not affect the basis functions, which are still defined on the original tetrahedral element. The subdivision procedure into sub-tetrahedra and related algorithms are detailed in the following section.

3.2. Subdivision based adaptive quadrature of intersected tetrahedra

The decision of whether or not to subdivide an element or sub-tetrahedron would ideally depend on whether or not it is intersected by the immersed boundary. The decision of whether or not to include a quadrature point in the quadrature rule requires a second test, to determine whether a point is included in the domain of the partial differential equation. In certain cases, such as triangulated surfaces immersed in tetrahedral meshes, it is possible to devise an analytical surface–element intersection test. However, the necessity of an inside/outside test remains and, to reduce the number of assumptions required of the immersed boundary representation, we propose to approximate the surface–element intersection test using only the inside/outside test. This approximation can be made to within the resolution of the finest level of sub-tetrahedra by using the *bottom-up* approach detailed by Varduhn et al. [65]. The bottom-up approach applies an inside/outside test to all quadrature points of the finest level of subdivision, then combines groups of fully-included tetrahedral sub-cells into larger tetrahedral sub-cells wherever possible, to reduce the final number of quadrature points. This involves a costly preprocessing step to generate the quadrature rule.

In the terminology of [65], the present work uses a *top-down* approach to generate quadrature rules. This method invokes the inside/outside test fewer times while generating a quadrature rule, but does not always resolve the boundary geometry as precisely. The algorithm that we use to determine the set of quadrature points and weights for each tetrahedral element of the original mesh is to apply the following recursive procedure to an integration tetrahedron covering the entire element, with integer input $l \geq 0$ indicating the level of recursion:

1. Propose a set of N_q quadrature points (and associated weights) appropriate for integrating smooth functions over the current integration tetrahedron. These points may be obtained by transforming quadrature points from a reference tetrahedron and scaling the weights appropriately.
2. Count the numbers N_in and N_out of the vertices of the integration tetrahedron falling inside and outside of the immersed object. This requires four calls to the inside/outside test.
3. If $N_\mathrm{in} = 0$, $N_\mathrm{out} = 0$, or $l = 0$, do not recurse. If $N_\mathrm{out} > 0$, add the proposed quadrature points falling in the fluid domain to the quadrature rule. This requires N_q calls to the inside/outside test.
4. Otherwise, if $N_\mathrm{in} > 0$, $N_\mathrm{out} > 0$, and $l > 0$, discard the points proposed in Step 1, divide the integration tetrahedron into 12 children, as depicted in Fig. 3, and apply this

procedure recursively to each of the children, with input $(l-1)$.

Section 3.4 details our implementation of the inside/outside test required for Steps 2 and 3. The division of a parent sub-tetrahedron into 12 children for Step 4 may be stated precisely as follows. We start by defining the 4-tuple of parametric points $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) \in (\mathbb{R}^3)^4$: these are the sub-tetrahedron's vertices in the parametric space of the top-level tetrahedral element of the finite element mesh. Next, we introduce the additional points

$$\begin{aligned}
\mathbf{p}_5 &= \tfrac{1}{2}(\mathbf{p}_1 + \mathbf{p}_2), & \mathbf{p}_9 &= \tfrac{1}{2}(\mathbf{p}_2 + \mathbf{p}_4),\\
\mathbf{p}_6 &= \tfrac{1}{2}(\mathbf{p}_1 + \mathbf{p}_3), & \mathbf{p}_{10} &= \tfrac{1}{2}(\mathbf{p}_3 + \mathbf{p}_4),\\
\mathbf{p}_7 &= \tfrac{1}{2}(\mathbf{p}_1 + \mathbf{p}_4), & \mathbf{p}_{11} &= \tfrac{1}{4}(\mathbf{p}_1 + \mathbf{p}_2 + \mathbf{p}_3 + \mathbf{p}_4),\\
\mathbf{p}_8 &= \tfrac{1}{2}(\mathbf{p}_2 + \mathbf{p}_3), &&
\end{aligned}$$

in this parameter space. The 12 children are defined by 4-tuples of parametric verticies chosen from the set $\{\mathbf{p}_1, \ldots, \mathbf{p}_{11}\}$, as illustrated in Fig. 3:

$$\begin{aligned}
&1 - (\mathbf{p}_5, \mathbf{p}_7, \mathbf{p}_6, \mathbf{p}_1), && 7 - (\mathbf{p}_8, \mathbf{p}_{10}, \mathbf{p}_6, \mathbf{p}_{11}),\\
&2 - (\mathbf{p}_9, \mathbf{p}_5, \mathbf{p}_8, \mathbf{p}_2), && 8 - (\mathbf{p}_{10}, \mathbf{p}_9, \mathbf{p}_7, \mathbf{p}_{11}),\\
&3 - (\mathbf{p}_{10}, \mathbf{p}_8, \mathbf{p}_6, \mathbf{p}_3), && 9 - (\mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_{11}, \mathbf{p}_8),\\
&4 - (\mathbf{p}_7, \mathbf{p}_9, \mathbf{p}_{10}, \mathbf{p}_4), && 10 - (\mathbf{p}_9, \mathbf{p}_7, \mathbf{p}_{11}, \mathbf{p}_5),\\
&5 - (\mathbf{p}_6, \mathbf{p}_7, \mathbf{p}_5, \mathbf{p}_{11}), && 11 - (\mathbf{p}_7, \mathbf{p}_{10}, \mathbf{p}_{11}, \mathbf{p}_6).\\
&6 - (\mathbf{p}_5, \mathbf{p}_9, \mathbf{p}_8, \mathbf{p}_{11}), &&
\end{aligned}$$

This subdivision rule can be applied recursively to each child tetrahedron by substituting its 4-tuple of verticies for $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4)$.

Remark 2. Another choice of the subdivision scheme is to subdivide a tetrahedron into only eight sub-tetrahedra [86]. This generates fewer sub-tetrahedra at each level of recursion and reduces the cost of the preprocessing phase of the bottom-up method for generating the quadrature rule, which must access all tetrahedra at the finest level of subdivision. The proliferation of sub-tetrahedra at finer levels of subdivision is of less concern in top-down approaches to generating quadrature rules, and the 12-sub-tetrahedron method used here confers an additional benefit. In severely distorted finite elements, the ratio of the maximum edge length of a child sub-tetrahedron to the diameter of its parent sub-tetrahedron can be arbitrarily close to one when using the 8-sub-tetrahedron method. This could impede its ability to resolve discontinuities in the integrand, because the maximum length over which the integrand was assumed smooth would not decrease significantly. The introduction of an extra central vertex, namely $\mathbf{p}_{11}$ in the above discussion, eliminates this possibility in the 12-sub-tetrahedron subdivision.

3.3. Immersed surface integration

The weak boundary conditions formulated in Section 2.3 require that we evaluate surface integrals of traces of functions defined on the background finite element mesh. We follow the general approach of Düster et al. [45], by breaking the immersed surface into quadrature elements independently of the finite element mesh, and defining a quadrature rule on these elements. The points of these surface element quadrature rules must then be located in the parameter space of the tetrahedral finite elements in which they fall. This requires us to invert the mapping from the finite element parameter space to physical space. This inversion may be accomplished for general parametric elements by Newton iteration.

For the linear tetrahedral elements employed here, the mapping is affine, and its inverse can be computed in closed form. We could, in principle, invert the parameter-to-physical-space mapping for each background element until finding one in which the inverted parameters are within the element's parametric domain, but this procedure would be intractably costly to perform for every surface quadrature point. Instead, we compute bounding boxes for finite elements, then associate these bounding boxes with the leaves of a spatial octree that they intersect. For each surface quadrature point, we then recursively search for its containing finite element in the sub-tree containing the point. With this approach, we need only invert parameter-to-physical-space mappings of the subset of finite elements whose bounding boxes intersect the unique leaf of the octree that contains the surface quadrature point. Assuming that the element size of the finite element mesh is quasi-uniform in a mesh parameter, h, the octree should be constructed such that its coarsest level contains the entire set of finite elements and the leaf cells of its finest level have diameter $\Theta(h)$. When the finite element mesh is broken into subdomains for parallel computation, a separate octree may be generated for each subdomain, and the location of surface quadrature points in the finite element mesh may be carried out in parallel, with no communication overhead.

3.4. Simple-but-effective point location query

To determine whether a point $\mathbf{x}$ is inside or outside of a nominally closed surface, we employ ray tracing. We count the number of times, N, that a ray emanating from $\mathbf{x}$ intersects the surface. If N is odd, then $\mathbf{x}$ is inside of the surface and, if N is even, then $\mathbf{x}$ is outside of the surface. The operation of ray–surface intersection does not depend on the surface being closed, so this algorithm will execute even if the surface representation is not watertight. To make this procedure robust in the case of non-watertight surfaces and inexact floating-point arithmetic, one may cast several rays from $\mathbf{x}$ along different directions and return the inside/outside classification given by the majority of these rays. In the present work, we have taken care to ensure that the surface representations are closed, and we make inside/outside decisions using only one ray per point.

The operation of ray–surface intersection has been investigated very thoroughly by the computer graphics community [87]. A number of high-performance implementations with sophisticated optimizations are publicly available, e.g., the Manta interactive ray tracer [88]. In the present work, we have implemented a simple-but-effective strategy which we have found sufficiently fast for our purposes. First, we break the immersed surface into primitives (viz. triangles, in the present work) and sort these primitives into an octree hierarchy of bounding boxes as a preprocessing step. We then recursively intersect rays with occupied sub-trees of the octree. This avoids many unnecessary ray–primitive intersection operations

relative to the brute force approach of testing each ray against each surface primitive.

3.5. Generating adaptive non-boundary-fitted meshes

The immersogeometric concept based on the FCM is independent of a specific basis and can be used with any basis function technology and element type. The main motivation for using tetrahedral elements is their ability to provide locally refined three-dimensional discretizations, which is required here for boundary layer resolution. In contrast to hexahedral elements, there exist generally valid refinement algorithms for tetrahedra that work in any situation without restrictions. The availability of a large number of advanced tetrahedral meshing tools [89] motivates the integration of such a tool to generate an initial unstructured tetrahedral mesh. In many immersed situations, conforming to the boundary of a simple geometry (e.g., a rectangular box as the embedding domain) is the only restricting requirement for generating a mesh; the discretization process is extremely fast, even for a very large number of elements. At the same time, we can make use of advanced algorithms for mesh regularization and smoothing to ensure high-quality tetrahedral elements. We present a workflow based on an open-source mesh generator Gmsh [90] to efficiently generate non-boundary-fitted adaptive tetrahedral meshes.

In immersogeometric analysis, the whole embedding domain, including physical and fictitious domain, is discretized irrespective of the immersed boundary. The actual geometry of the immersed object is not explicitly needed for the mesh generation. Instead, one only needs to specify how the mesh should be graded in terms of element size over the embedding domain, with finer element sizes in the area of boundary layers. Gmsh enables control of the local mesh size via special functions that can be used in the input script. The functions we use include "Point", "Attractor", "Threshold", "Box", and "Min". "Point" specifies the location where a specific mesh size will be enforced. "Attractor" specifies the mesh size within a "Threshold" distance to a geometrical entity such as "Point". We use these functions to set the element sizes in the vicinity of the immersed boundary. Since elements can arbitrarily intersect with the immersed boundary, that is, the mesh conformity does not need to be enforced, this procedure is significantly less demanding and much more reliable than conforming mesh generation. In addition, we use "Box" to specify the element size inside of a parallelepiped to locally (and uniformly) refine zones where the flow is expected to be more complex. Finally, when several mesh size control functions are active at the same location, "Min" is used to resolve this overconstraint by choosing the mesh size to be the minimum of the sizes specified in those functions. For more details of the Gmsh functions, the reader is referred to [90].

4. Benchmark example: Flow around a sphere

The flow around a sphere at Reynolds numbers $Re = 100$ and 300 for laminar flow and $Re = 3700$ for turbulent flow constitute canonical test cases, for which a large number of reference results are available in the literature (see, e.g., [91–95] for laminar flow and [96–98] for turbulent flow). We use this example as a first benchmark to assess the accuracy of our immersogeometric method in both laminar and turbulent flow regimes. In addition to reference values from the literature, we compute further reference results using the same variational framework with standard boundary-fitted tetrahedral meshes that are comparable in terms of overall mesh resolution and boundary layer mesh grading.

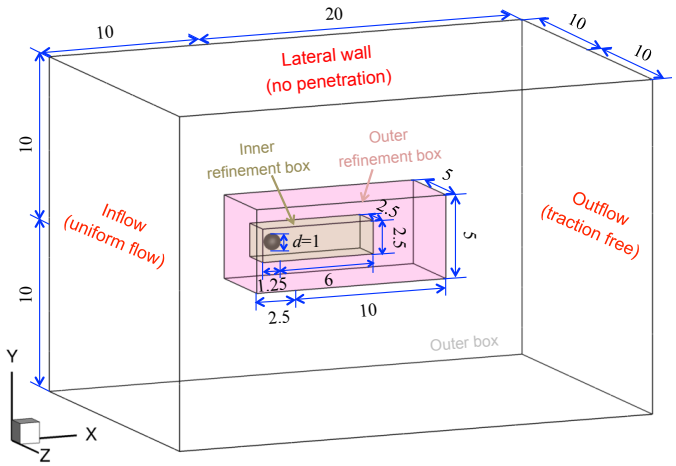


Figure 4: Computational domain, boundary conditions, and the immersed sphere. The refinement boxes, where finer element sizes are used, are also shown in the figure.

4.1. Problem set-up

Figure 4 illustrates the dimensions of the computational domain, the location and size of the immersed sphere, and the boundary conditions. We note that for this example all indications of size are non-dimensional. The radius of the sphere, the inflow velocity and the fluid density are all one, so that the Reynolds number $Re = \mu^{-1}$ is defined as the inverse of the viscosity (and vice versa). The inlet boundary condition and the slip boundary condition on the lateral faces are strongly enforced, while the no-slip/no-penetration condition $\mathbf{u} = \mathbf{0}$ on the surface of the sphere is enforced weakly as described in Section 2.3 .

As there is no analytical flow solution to this problem, we use characteristic quantities of interest that are widely used in fluid mechanics to compare solutions. The drag coefficient is computed as $C_D = 2F_D/(\rho U^2 A)$, where F_D is the drag force, ρ is the fluid density, U is the inflow speed, and A is the frontal area of the sphere. We note that we evaluate the drag force using the variationally consistent conservative definition of traction [75, 99] in the following form:

$$\begin{aligned}\mathbf{t}^h = &-\boldsymbol{\sigma}^h\mathbf{n} - \rho\left\{\mathbf{u}^h\cdot\mathbf{n}\right\}_-\left(\mathbf{u}^h-\mathbf{g}\right)\\ &+\tau^B_{\mathrm{TAN}}\left(\left(\mathbf{u}^h-\mathbf{g}\right)-\left(\left(\mathbf{u}^h-\mathbf{g}\right)\cdot\mathbf{n}\right)\mathbf{n}\right)\\ &+\tau^B_{\mathrm{NOR}}\left(\left(\mathbf{u}^h-\mathbf{g}\right)\cdot\mathbf{n}\right)\mathbf{n}\,,\end{aligned} \tag{17}$$

where $\{\,\cdot\,\}_-$ denotes the negative part of the bracketed quantity, that is, $\{\mathcal{A}\}_- = \mathcal{A}$ if $\mathcal{A} < 0$ and $\{\mathcal{A}\}_- = 0$ if $\mathcal{A} \geq 0$. In our case, the sphere is stationary, hence we have $\mathbf{g} = \mathbf{0}$. Parametric

Table 1: Element sizes in the boundary-fitted mesh for laminar flow around a sphere.

Mesh	Total number of elements	Near sphere element size	Inner refinement box element size	Outer refinement box element size	Outer box element size
BM0	229,694	0.02	0.2	$0.8/\sqrt{2}$	1.2
BM1	1,710,898	0.01	0.1	$0.4/\sqrt{2}$	1.0
BM2	8,519,435	0.005	0.05	$0.2/\sqrt{2}$	0.8

Table 2: Element sizes in the immersogeometric mesh for laminar flow around a sphere.

Mesh	Total number of elements	Near sphere element size	Inner refinement box element size	Outer refinement box element size	Outer box element size
IM0	304,330	0.02	0.2	$0.8/\sqrt{2}$	1.2
IM1	1,833,434	0.01	0.1	$0.4/\sqrt{2}$	1.0
IM2	9,041,302	0.005	0.05	$0.2/\sqrt{2}$	0.8

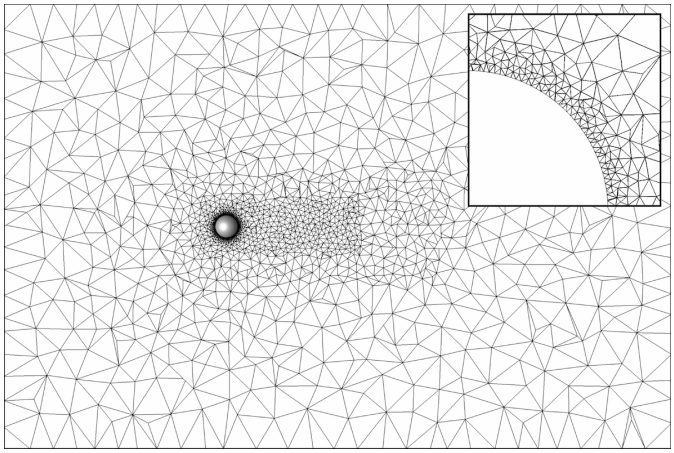


Figure 5: Central section through the coarsest boundary-fitted mesh (BM0).

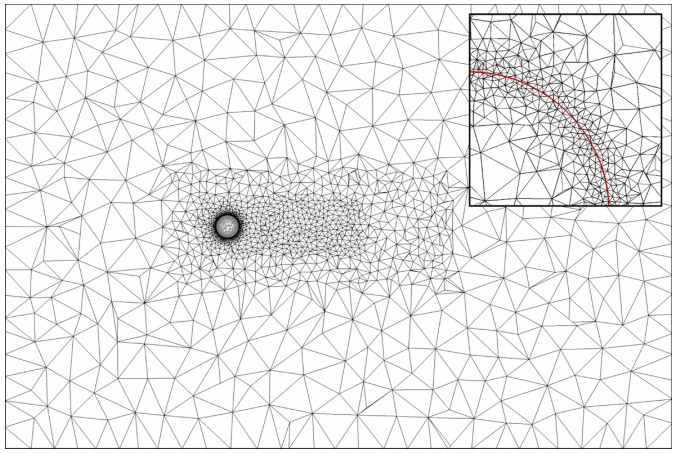


Figure 6: Central section through the coarsest immersogeometric mesh (IM0). Note that the elements within the immersed sphere without support in the fluid domain are removed before analysis.

studies in [43] found that quantities of interest for flow over a 2D cylinder were relatively insensitive to the precise values of the stabilizing penalties τ^B_{NOR} and τ^B_{TAN}. In the computations of this paper, we set $\tau^B_{\mathrm{NOR}} = \tau^B_{\mathrm{TAN}} = 10^3$.

The non-dimensional length of the recirculation bubble is computed as L/d, where d is the diameter of the sphere and L is the length from the rear end of the sphere to the point where the velocity in x-direction changes sign. For $Re = 300$ and 3700 vortex shedding occurs, which can be characterized by the Strouhal number defined as $St = fd/U$, with f being the frequency of vortex shedding. For the turbulent case, we record the drag history over time, and use the Lomb-Scargle periodogram technique to extract the frequency (see [98] for details).

4.2. Mesh design and boundary layer resolution

The proper design of the fluid mesh with a suitable resolution of boundary layers is a key requirement for successful high-fidelity analysis in CFD. Our immersogeometric method does not eliminate mesh design considerations, as locations for adaptive boundary layer resolution and the degree of the mesh grading still need to be specified. However, non-boundary-fitted meshes speed up the generation of reliable CFD meshes significantly, since boundary layer elements do not have to conform to (potentially very complex) surfaces of immersed objects.

In this section, we summarize the boundary-fitted and immersogeometric tetrahedral mesh generations for the sphere benchmark. For the laminar cases at $Re = 100$ and 300, we consider three sets of boundary-fitted meshes denoted by BM0, BM1 and BM2. The mesh statistics and the characteristic element sizes used in the different areas are detailed in Table 1. Figure 5 illustrates the coarsest boundary-fitted mesh (BM0), for which the central section in x-direction is shown. We observe that the local refinement is performed around the sphere, where we expect sharp boundary layers, and in the wake region behind the sphere, where we expect the formation of vortices.

Using the workflow described in Section 3.5, we generate immersogeometric tetrahedral meshes denoted by IM0, IM1 and IM2 for the laminar flow cases. The three meshes have comparable mesh resolution to their boundary-fitted counterparts. Table 2 shows detailed mesh characteristics for all three immersogeometric meshes and a central cut of the coarsest one (IM0) is shown in Fig. 6.

Comparing the number of elements, we observe that the immersogeometric meshes have slightly more elements than the corresponding boundary-fitted meshes. This is due to the elements that are located within the domain of the immersed sphere. Since in our immersogeometric method all elements

Table 3: Element sizes used in the immersogeometric and boundary-fitted mesh generations for turbulent flow case.

Near sphere element size	Inner refinement box element size	Outer refinement box element size	Outer box element size
0.004	0.04	$0.16/\sqrt{2}$	0.8

without support in the fluid domain will be removed in a post-processing step and are not taken into account during the analysis, the effective number of elements (and hence the effective number of degrees of freedom) is equivalent for corresponding immersogeometric and boundary-fitted meshes. We note that the surface of the immersed sphere needs to be discretized as well. We use a simple triangulation of the surface, whose element size is half of the volume element size near the sphere. We perform standard triangular quadrature on the surface triangles to evaluate the weak boundary condition terms in the variation formulation (16).

For the turbulent case at $Re = 3700$, we consider only one mesh resolution design for generating the immersogeometric and corresponding boundary-fitted meshes. The mesh design is based on the experience reported by Bazilevs et al. [98], in which the same VMS formulation with weakly enforced boundary conditions presented in Section 2 was employed. Note that prismatic elements were used for the boundary layer mesh in [98]. In our cases, for the sake of consistency between immersogeometric and boundary-fitted meshes, only tetrahedral elements are considered and we keep the element heights in the boundary layer comparable to those reported in [98]. Table 3 summarizes the characteristic element sizes at different locations in the computational domain shown in Fig. 4. The resulting immersogeometric mesh consists of 12,068,115 tetrahedral elements (11,019,886 effective elements, including 922,404 intersected elements) and the boundary-fitted counterpart consists of 10,911,263 elements.

Remark 3. CFD meshes often include thin layers of anisotropic prismatic or hexahedral elements near no-slip boundaries to resolve the sharp gradients in wall normal direction. In the tangential direction, the element size can be relatively large. This reduces the total number of elements relative to isotropic tetrahedral meshes, such as those used in this paper. However, tetrahedral elements offer us great geometrical flexibility, allowing efficient and automatic mesh generation. It is also possible to generate high-quality anisotropic boundary layer meshes using only tetrahedral elements (see, e.g., [100–102]), but we have not attempted to apply this technology in the present work.

4.3. Immersogeometric results for laminar flow

For the sphere example at moderate Reynolds numbers, we obtain laminar flow that over time reaches a stable state. $Re = 100$ leads to an axisymmetric steady flow pattern and $Re = 300$ leads to a periodic flow pattern in the wake of the sphere. We note that in all our computations, we use a constant time-step size of $\Delta t = 0.01$, which ensures that there are more than 500 steps within one period for the $Re = 300$ case. As

Table 4: Different flow characteristics, computed with our immersogeometric method and different levels l of adaptive quadrature sub-cells, and with a boundary-fitted mesh of comparable size.

	$Re = 100$		$Re = 300$	
	C_D	L/d	C_D	St
IM2 ($l = 0$)	1.142	0.858	0.719	0.139
IM2 ($l = 2$)	1.093	0.856	0.662	0.135
BM2	1.093	0.857	0.661	0.135

initial condition, we apply a constant flow field corresponding to the uniform inflow velocity over the entire fluid domain.

4.3.1. The importance of accurate geometry resolution

We test the influence of the accuracy with which we resolve the geometry in intersected elements. To this end, we focus on the finest immersogeometric mesh (IM2) and vary the number of sub-cell levels l in the adaptive quadrature scheme. Taking more sub-cell levels into account increases the accuracy of the domain integration, which is directly linked to geometric accuracy. We focus here on the effect of the domain integration in intersected elements, since we know from earlier studies (see, e.g., [39]) that it has a significantly larger impact on the solution than the accuracy of the surface integral. We ensure a sufficiently fine resolution of the surface discretization of the sphere by using a triangulation with half the characteristic length of the finest tetrahedral elements.

Table 4 shows the results of the flow characteristics for $Re = 100$ and 300 obtained from computations without adaptive quadrature, with two levels of adaptive quadrature sub-cells, and with a boundary-fitted mesh of comparable mesh resolution. The drag coefficient depends on the flow near the sphere boundary and appears to be particularly sensitive to quadrature error, exhibiting variations of up to 10%. The non-dimensional recirculation length L/d and the Strouhal number mainly depend on the flow pattern in the wake of the sphere, which are less sensitive to the near boundary flow. Comparing the quantities with results obtained from boundary-fitted computations, we clearly see that an increased geometric resolution is mandatory to achieve accurate flow solutions.

Increasing the number of adaptive sub-cell levels becomes expensive for larger l, because the number of quadrature points increases exponentially [56]. To find a suitable compromise between computational cost and geometric accuracy, we gradually increase the number of levels from $l = 0$ to $l = 4$ for the $Re = 100$ case. Figure 7 plots the corresponding drag coefficient versus the number of sub-cell levels. We observe that from $l = 0$ to $l = 1$ there is a huge improvement. We still obtain an improvement from $l = 1$ to $l = 2$, but the difference between $l = 2$ and $l = 4$ is very small. Taking into account the increase in computing time (see Table 5 and Remark 4), we conclude that $l = 2$ levels of adaptive sub-cells represent a good balance between computational cost and geometric accuracy for the present immersogeometric method.

Remark 4. The computations reported in Table 5 are carried out in a parallel computing environment on the Lonestar Linux cluster [103] at the Texas Advanced Computing Center

Table 5: Total computing time required to run 50 time steps with different levels of adaptive quadrature sub-cells on the same mesh (IM2).

	$l = 0$	$l = 1$	$l = 2$	$l = 4$
Time (s)	~ 323	~ 347	~ 442	~ 3521

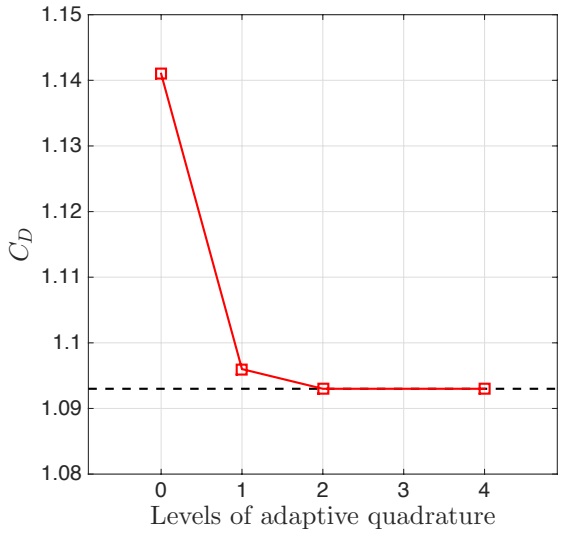


Figure 7: Drag coefficient C_D, computed with our immersogeometric method and different levels of adaptive quadrature sub-cells for flow at $Re = 100$. We include the corresponding boundary-fitted result as reference (dashed line).

(TACC) [104]. The system consists of 1,888 compute nodes, each with two Intel Xeon X5680 3.33GHz hex-core processors and 24GB of memory. A description of our parallelization strategy and a demonstration of its strong linear scaling can be found in Hsu et al. [105]. The mesh is partitioned into 480 subdomains using METIS [106], and each subdomain is assigned to a processor core. For the computations in Table 5, we use two Newton iterations per time step, with 80 and 250 GMRES iterations for the first and second Newton iterations, respectively, and record the time required to run 50 time steps.

4.3.2. *Convergence and mesh independence study*

To get an idea of the overall accuracy of the flow solution that we can achieve with our immersogeometric method, we compare the immersogeometric results in terms of the drag coefficient, the non-dimensional length of the recirculation bubble and the Strouhal number with reference values that we computed with boundary-fitted meshes as well as corresponding values reported in the literature. This comparison is done for the complete series of meshes with increasing mesh density that we defined in Section 4.2. This also allows for a mesh independence study for both the immersogeometric and boundary-fitted cases. Table 6 shows the reference values obtained with the boundary-fitted discretizations BM0, BM1 and BM2 at Reynolds numbers $Re = 100$ and 300. We also show the maximum and minimum range of values for these quantities that we found in the CFD literature, specifically consulting these articles [91–95]. Table 7 shows the corresponding quantities obtained with the immersogeometric method and meshes IM0, IM1 and IM2 that are of comparable mesh density and grading. The immersogeometric computations are based on

Table 6: Mesh independence study for boundary-fitted discretization.

	$Re = 100$	
	C_D	L/d
BM0	1.131	0.790
BM1	1.094	0.855
BM2	1.093	0.857
Literature	1.060–1.096	0.850–0.880
	$Re = 300$	
	C_D	St
BM0	0.715	0.123
BM1	0.662	0.136
BM2	0.661	0.135
Literature	0.634–0.671	0.134–0.137

Table 7: Mesh independence study for immersogeometric discretization with $l = 2$ levels of adaptive quadrature sub-cells.

	$Re = 100$		$Re = 300$	
	C_D	L/d	C_D	St
IM0	1.141	0.767	0.714	0.123
IM1	1.095	0.855	0.662	0.134
IM2	1.093	0.856	0.662	0.135

$l = 2$ levels of adaptive quadrature sub-cells to ensure the accurate resolution of the geometry in intersected elements.

The overall convergence behavior of the computed quantities is equivalent in both immersogeometric and boundary-fitted cases. Comparing the results between the different mesh sizes within each method, we see that the results obtained with the finest meshes are sufficiently converged and can be considered as mesh independent. A comparison of the values in Tables 6 and 7 shows that with a comparable mesh resolution, our immersogeometric method achieves the same accuracy as the boundary-fitted method.

4.4. *Immersogeometric results for turbulent flow*

For assessing the accuracy of our immersogeometric method for turbulent flows, we increase the Reynolds number in the current benchmark to $Re = 3700$. For this configuration and Reynolds number, there occurs a laminar flow separation near the equator of the sphere and a transition to turbulence in the wake of the sphere [97]. We compare the immersogeometric results in terms of key quantities of interest with reference values obtained from our boundary-fitted computations, as well as with Direct Numerical Simulation (DNS) results reported by Rodriguez et al. [97] and VMS results computed by Bazilevs et al. [98].

In our immersogeometric and boundary-fitted computations, we use a time-step size of 0.0015. Figure 8 shows the immersogeometric result of instantaneous vortical structures by visualizing the isosurfaces of $\lambda_2 = -0.1, -0.5$ and -1. λ_2 is the second largest eigenvalue of the tensor $\mathbf{S}^2 + \mathbf{\Omega}^2$, where $\mathbf{S}$ and $\mathbf{\Omega}$ are the symmetric and antisymmetric components of $\nabla\mathbf{u}$, respectively. The vortex core is defined as the region where $\lambda_2 < 0$ [107]. The figure illustrates the three-dimentional chaotic nature of turbulent flow in the wake of the sphere. Figure 9 shows the contours of the instantaneous out-of-plane vorticity component on a planar cut, computed with both methods.

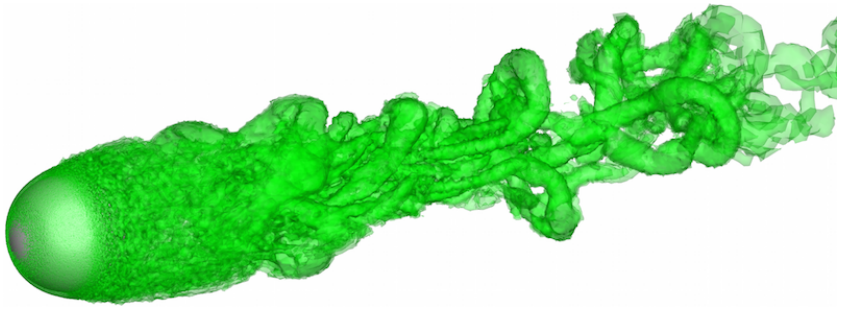
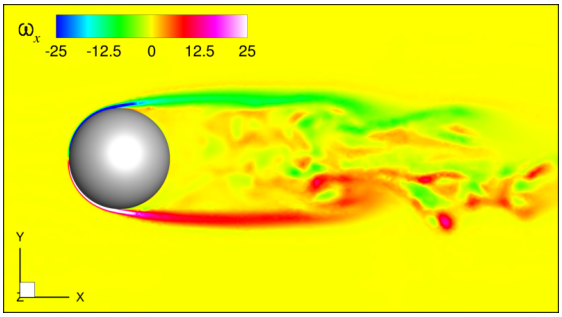
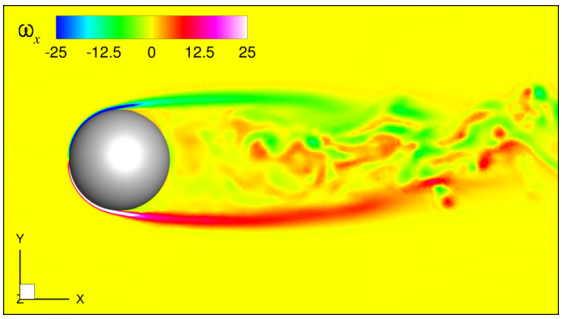


Figure 8: Visualization of the immersogeometric result of instantaneous vortical structures in the wake of the sphere for turbulent flow at Re = 3700.

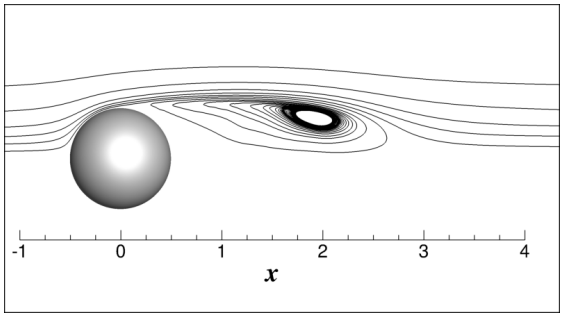


(a) Boundary-fitted result.

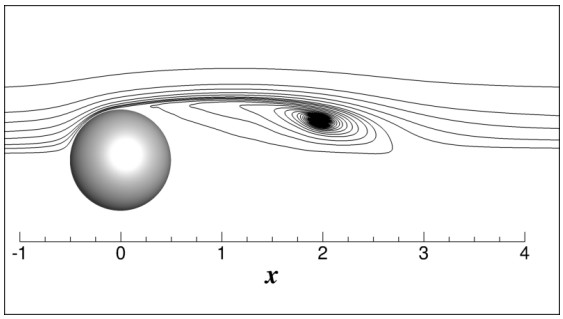


(b) Immersogeometric result.

Figure 9: Contour of instantaneous out-of-plane vorticity component on a planar cut.



(a) Boundary-fitted result.



(b) Immersogeometric result.

Figure 10: Time-averaged velocity streamlines on a planar cut.

The results illustrate the laminar shear layer and its separation from the sphere. They also show the break-up of the shear layer, its transition to turbulence and the turbulent wake, which characterize the flow at sub-critical Reynolds numbers [97].

We report further the time-averaged quantities of interest in Table 8, computed with our immersogeometric and boundary-fitted methods, and compare them with reference values in the literature [97, 98]. We investigate the time-averaged drag coefficient $\overline{C}_D$, the Strouhal number $\overline{St}$, and the non-dimensional length $\overline{L}/d$ of the recirculation bubble evaluated from the rear end of the sphere. In addition, we compute the time-averaged pressure coefficient $\overline{C}_{pb}$ at an azimuthal angle of 180°, which corresponds to the rearmost point of the sphere in the main flow direction. Time averaging is performed when the flow solution has converged to a quasi-steady state. Figure 10 shows the mean velocity streamlines on a planar cut, from which the time-averaged recirculation bubble can be seen.

Table 8: Comparison of time-averaged quantities of interest for turbulent flow at Re = 3700.

	$\overline{C}_D$	$\overline{L}/d$	$\overline{St}$	$\overline{C}_{pb}$
Immersogeometric (l = 0)	0.399	2.26	0.205	-0.254
Immersogeometric (l = 1)	0.397	2.26	0.208	-0.258
Immersogeometric (l = 2)	0.393	2.27	0.218	-0.217
Boundary-fitted	0.393	2.27	0.217	-0.215
DNS (Rodriguez et al. [97])	0.394	2.28	0.215	-0.207
VMS (Bazilevs et al. [98])	0.392	2.28	0.221	-0.207

4.4.1. *The importance of accurate geometry resolution*

We assess the role of accurate geometry resolution in immersogeometric analysis of turbulent flow by varying the number of levels l of adaptive quadrature sub-cells in intersected elements. We observe in Table 8 that the immersogeometric results converge to the boundary-fitted reference values when l is increased from 0 to 2, i.e., under the refinement of adaptive quadrature sub-cells. We also find that all quantities obtained

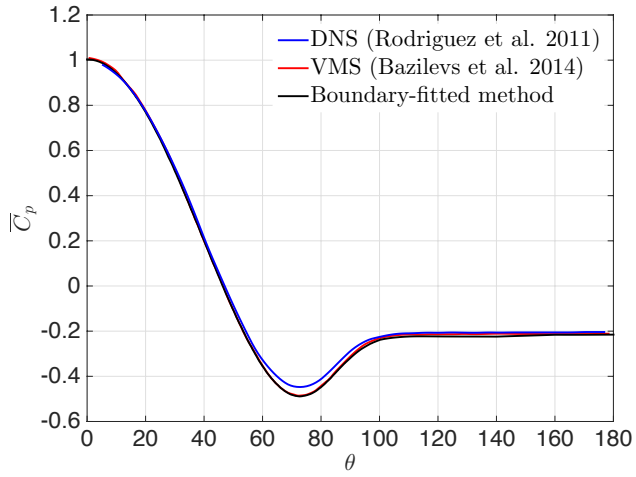


Figure 11: Time-averaged pressure coefficient evaluated along the upper crown line of the sphere. We compare our reference boundary-fitted result with results from the literature [97, 98].

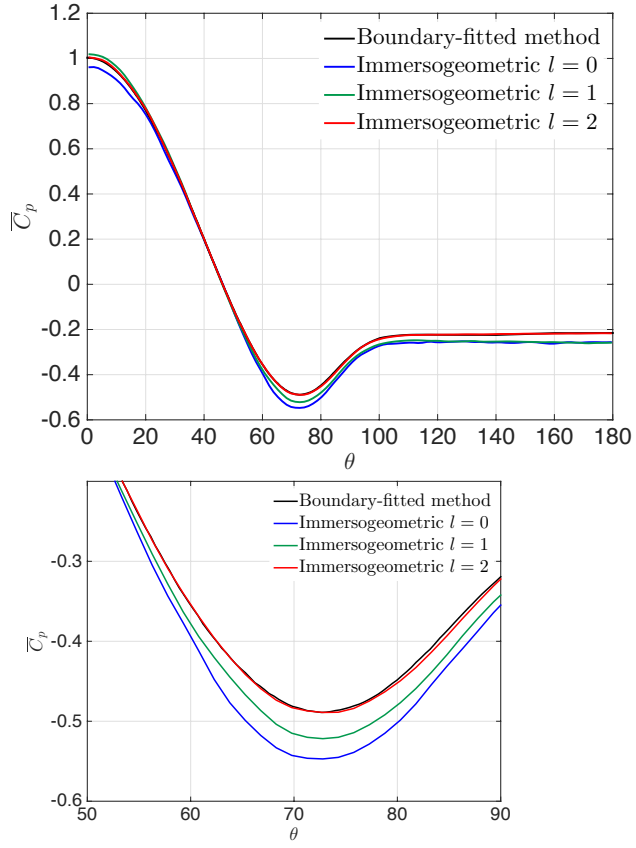


Figure 12: Comparison of the time-averaged pressure coefficient computed with our immersogeometric method with different levels l of adaptive quadrature sub-cells and our boundary-fitted method as reference.

with $l = 2$ are in good agreement with the values reported in the literature.

Figures 11 and 12 plot the distribution of the time-averaged pressure coefficient over the upper crown line of the sphere along the main flow direction computed with different methods. Figure 11 shows that our boundary-fitted simulation result is in very good accordance with results from the literature.

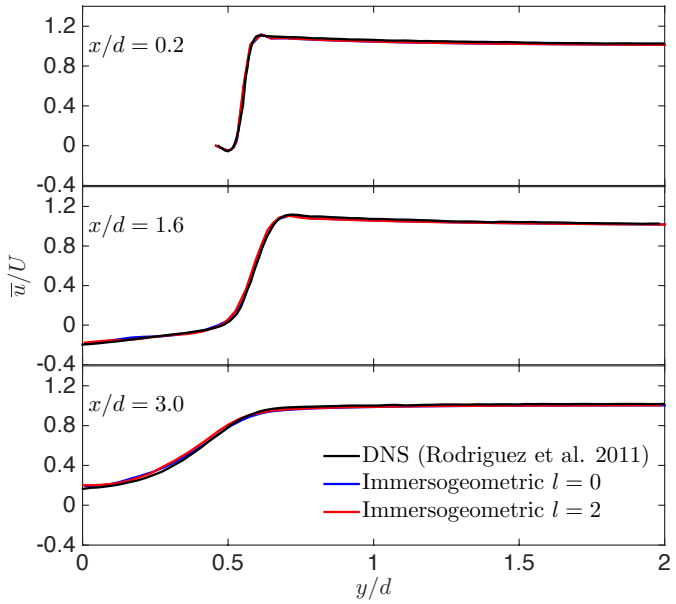


Figure 13: Mean streamwise velocity profiles at three different locations in the wake. DNS [97] result is plotted as reference.

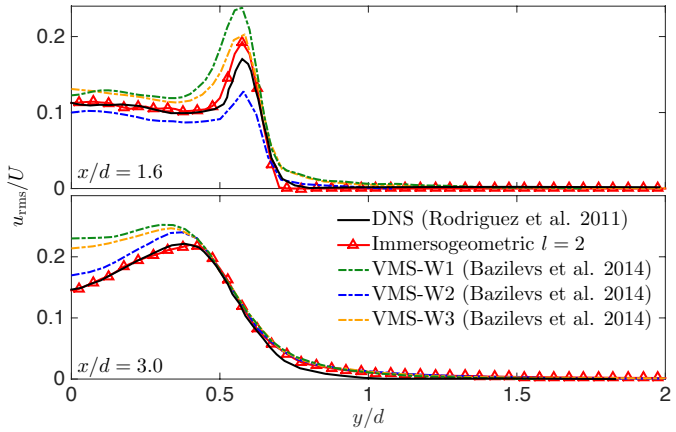
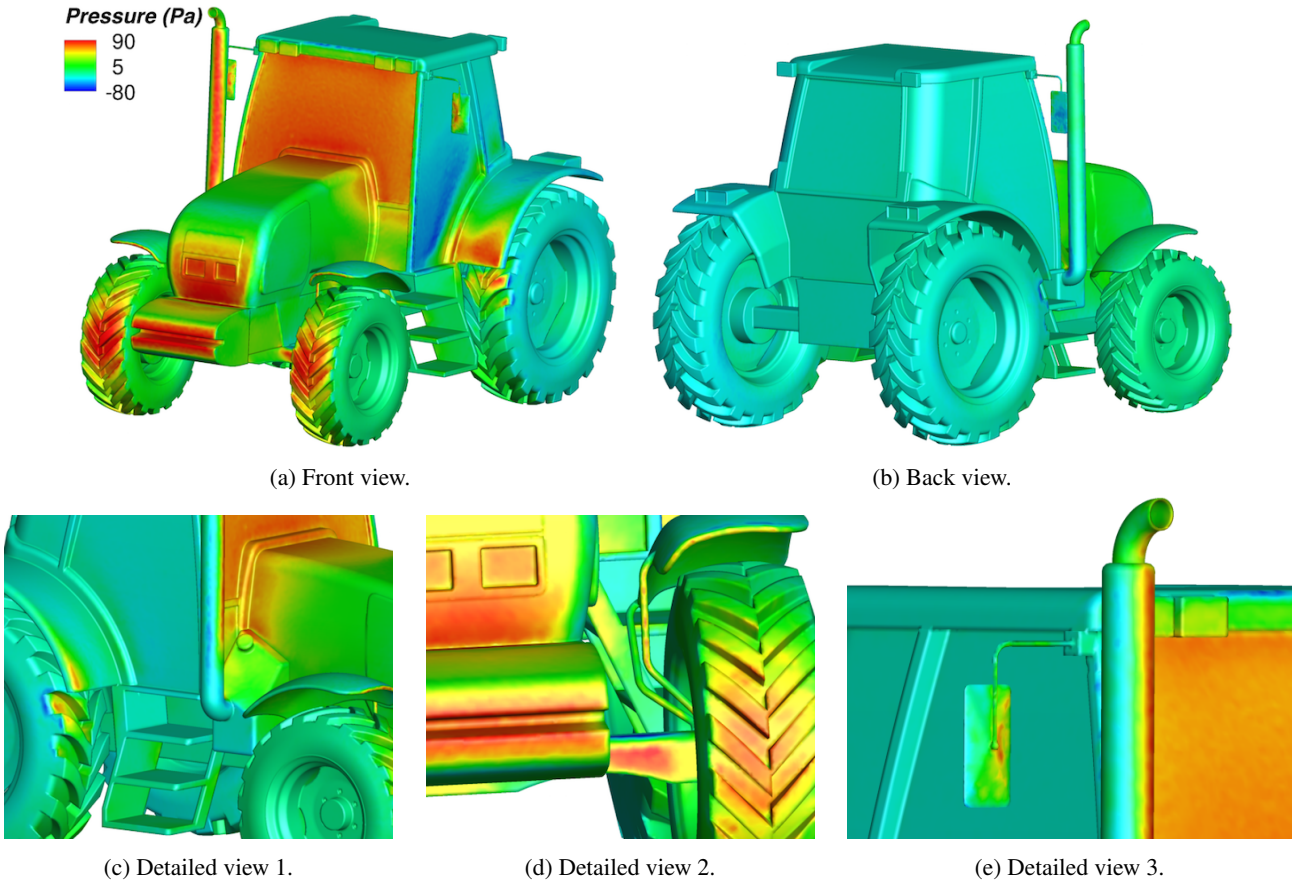


Figure 14: RMS of the streamwise velocity fluctuations at two different locations in the sphere wake. DNS result [97] and VMS results for three different time windows [98] are plotted as references.

We note that even though the same variational formulation and characteristic element sizes are used in our simulation and that of Bazilevs et al. [98], different element types were used in the boundary layer mesh (see Section 4.2), which result in the slight difference between our results.

Figures 12a and 12b show that results obtained with our immersogeometric method match the result of our boundary-fitted method well, while clear convergence towards the boundary-fitted reference can be observed under the increased levels of adaptive quadrature sub-cells. This confirms that a faithful representation of the geometry in terms of accurate volume quadrature in intersected elements is a key requirement for obtaining accurate flow results with our immersogeometric method.

Figure 13 compares the profiles of the mean streamwise velocity at three different positions in the wake computed by DNS [97] and our immersogeometric method with $l = 0$ and



(a) Front view.

(b) Back view.

(c) Detailed view 1.

(d) Detailed view 2.

(e) Detailed view 3.

Figure 15: Time-averaged pressure distribution over the tractor surface, computed with the immersogeometric mesh.

$l = 2$ levels of adaptive quadrature sub-cells. We observe that our results and the DNS results are in very good agreement, even when the adaptive quadrature is not applied ($l = 0$). This is due to the fact that the streamwise velocities compared are away from the immersed boundary, and as a result, are not sensitive to the accuracy of the geometry resolution. Figure 14 shows the root-mean-square (RMS) of the streamwise velocity fluctuations at two different locations in the sphere wake. DNS result [97] and VMS results for three different time windows [98] are plotted along with our immersogeometric result ($l = 2$). Bazilevs et al. [98] observed that the RMS of the streamwise velocity fluctuations is sensitive to the time window selection in the very near wake. In our case, a window of 300 time units is used for the evaluation. The comparison with the references clearly shows the accuracy of our immersogeometric method based on VMS turbulence model and weakly enforced boundary conditions for turbulent flow problems.

5. Industrial scale example: Turbulent flow around a tractor

Aerodynamic analysis of vehicles based on standard CFD tools with boundary-fitted meshes is a standard practice in the automotive industry. However, there are several important challenges that hamper the efficient use of standard CFD tools. Typical vehicle designs lead to very complex fluid domain boundaries. This constitutes a major obstacle for the transfer of fluid domains into boundary-fitted computational meshes, since it requires labor-intensive intermediate steps such as decomposition of large geometric models, geometry clean-up, and mesh manipulation. An example is the agricultural tractor shown in Fig. 15, which incorporates many geometrically complex details. In the context of tractor design, aerodynamic analysis help designers determine how air flow around the tractor can contribute to the cooling of the engines at low driving speeds. In this section, we will use the tractor to demonstrate how immersion of complex geometries into a non-boundary-fitted mesh can alleviate many challenges of standard boundary-fitted mesh generation in the context of large-scale industrial applications. We also verify the accuracy of our immersogeometric method by comparison with standard boundary-fitted meshes of comparable size and refinement pattern.

5.1. Tractor set-up

The dimensions of the tractor, the surrounding fluid domain, and the boundary conditions are illustrated in Fig. 16. We specify a uniform inflow with a streamwise velocity of 11.176 m/s (25 mph) ahead of the tractor. This velocity corresponds to the typical driving speed of an agricultural tractor. For simplicity, we assume a no-slip boundary condition on the ground. At a constant temperature of 300 K (26.85°C), the density and dynamic viscosity of the air are 1.177 kg/m^3 and 1.846×10^{-5}

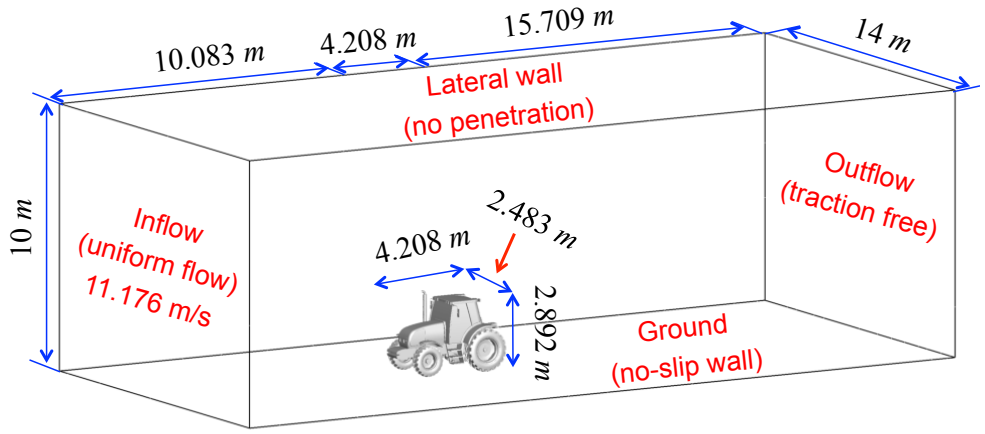


Figure 16: Dimensions and boundary conditions of the flow domain and the immersed tractor geometry.

kg/(m·s), respectively. The characteristic length of the tractor is defined as the distance from the leading to the rear end in the direction of the main flow and is 4.208 m. The Reynolds number is approximately 3×10^6, which corresponds to highly turbulent flow. The geometry of the tractor is described by a boundary representation, which was designed with a CAD tool and exported in STL-format.

5.2. *Generating immersogeometric and boundary-fitted meshes*

We employ the workflow described in Section 3.5 to generate an adaptive immersogeometric mesh of the tractor, leveraging the refinement capabilities of the open-source mesh generator Gmsh [90]. We locally refine the tetrahedral mesh close to the tractor surface and the ground surface for capturing boundary layers, and behind the tractor to capture the wake. The final immersogeometric mesh consists of 11,489,570 linear tetrahedral elements (9,367,179 effective elements) and is shown in Figs. 17a and 17b. In all (1,953,920) intersected elements we add two levels of adaptive quadrature sub-cells to accurately integrate the volume integrals. Sufficiently accurate integration in intersected elements is required to faithfully capture the geometry of the tractor, which is a key requirement for obtaining accurate flow solutions with our immersogeometric method.

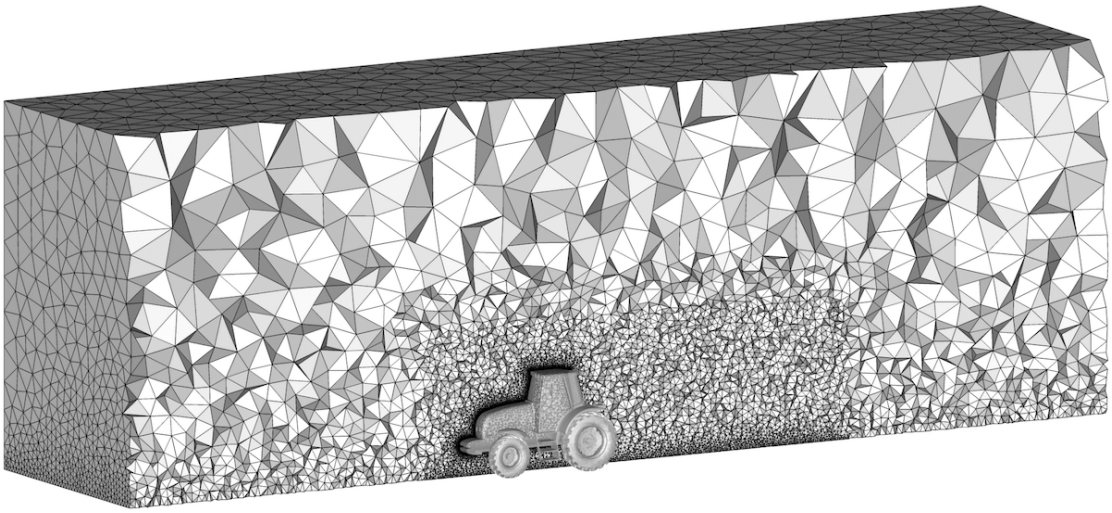
To obtain simulation results based on standard boundary-fitted meshes for comparison, we use a commercial mesh generator ANSA [108] due to its robustness in generating boundary-fitted meshes for complex geometries. We ensure that the local refinement pattern close to the tractor and ground surfaces and in the wake of the tractor is comparable to the immersogeometric mesh. A zoom of our boundary-fitted mesh can be seen in Fig. 17c. We note that due to the fine geometric details of the tractor, local refinement of the surface mesh in many of the regions is necessary to be able to fit the fine-scale geometry with boundary-conforming tetrahedral elements. The quality of the volume mesh depends highly on the quality of the surface mesh used for its generation. The total number of elements in the boundary-fitted mesh is 10,854,275, which is comparable to the immersogeometric mesh.

We note that a significant advantage of the immersogeometric workflow is its geometric flexibility. For example, it enables us to impose a uniform mesh size along the immersed tractor surface regardless of fine-scale geometric features. We can therefore easily control the mesh size resolution independently of the geometry, for example, to obtain a coarser mesh for fast preliminary design studies. This is not possible in boundary-fitted analysis, where a coarser mesh requires geometry operations first to remove all geometric features that are of finer scale than the targeted minimum element length.

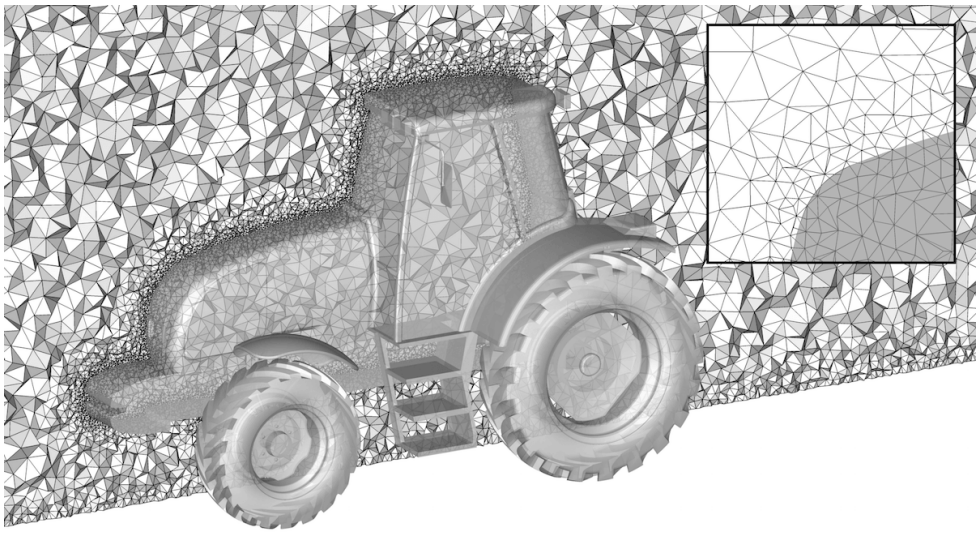
5.3. *Comparison of immersogeometric and boundary-fitted results*

In both immersogeometric and boundary-fitted computations, we use a time-step size of 3×10^{-4} s. Both meshes are partitioned into 504 subdomains, each assigned to a processor core. The computations are carried out on the Lonestar Linux cluster described in Remark 4. Figure 15 shows the time-averaged static pressure over the tractor surface, computed with our immersogeometric method. We observe that the pressure distribution over geometric details of the tractor surface is captured well. Figure 18 illustrates the instantaneous vortical structures of the highly turbulent flow around the tractor by visualizing the isosurfaces of $\lambda_2 = -1.5, -2$ and -5. Snapshots of the velocity magnitude on planar cuts at different heights above the ground at the same time instant as Figure 18 are shown in Figure 19. Detailed flow features such as flow around the pipe and mirrors can be clearly seen.

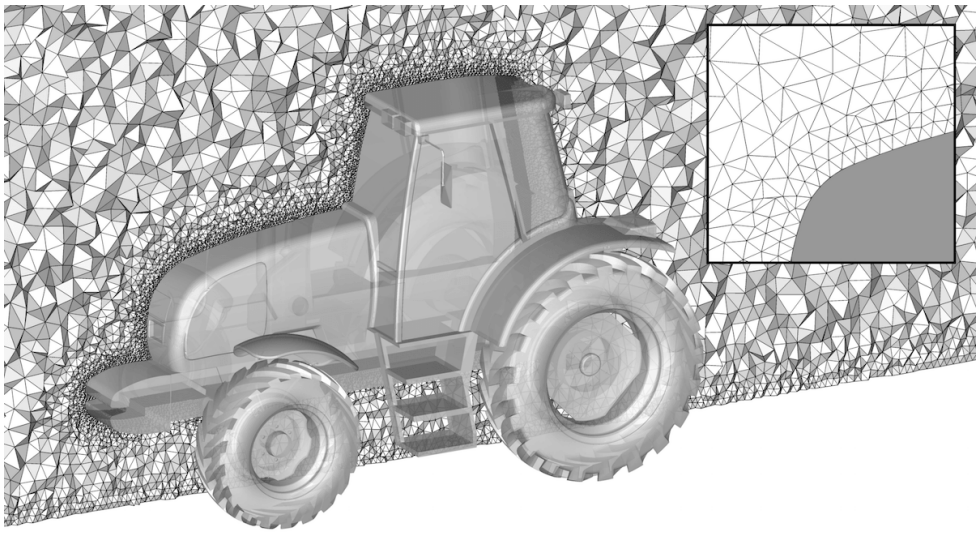
To compare the immersogeometric and boundary-fitted results, we first compute the time-averaged drag coefficient $\overline{C}_D = 2\overline{F}_D/\rho U^2 A$, where U is the inflow velocity, $\overline{F}_D$ is the time-averaged drag force, and $A = 5.942$ m^2 is the area of the frontal tractor surface projected onto a plane perpendicular to the main flow direction. The values of $\overline{C}_D$ are 0.851 and 0.838 for the immersogeometric and boundary-fitted computations, respectively. The results are in good agreement between the two methods, with a deviation of 1.55%. We note that the reported drag coefficients are in good accordance with those of observed in similar industry benchmarks. For example, the drag coefficient



(a) Overall view of the immersogeometric mesh.



(b) Zoom of the immersogeometric mesh.



(c) Zoom of the standard boundary-fitted mesh.

Figure 17: Locally refined tetrahedral meshes of the fluid domain for aerodynamic analysis of the tractor. We show the mesh cut along a plane in flow direction.

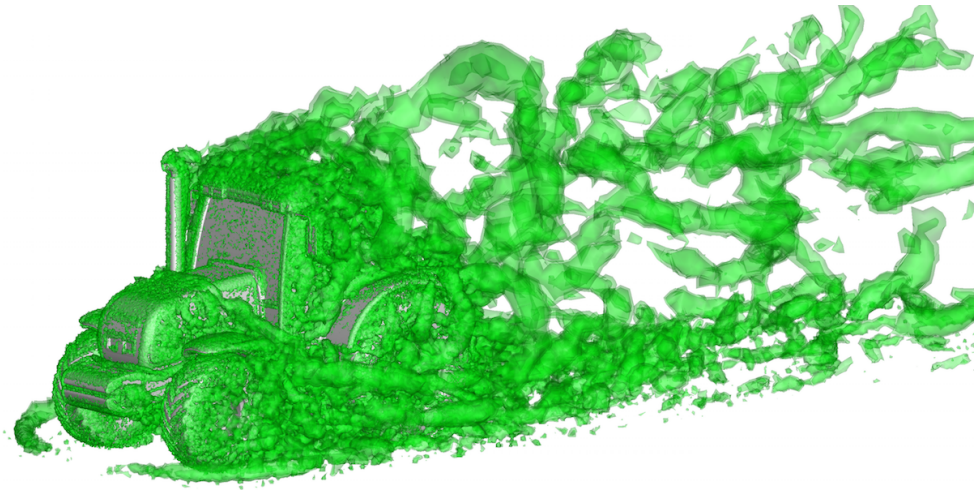
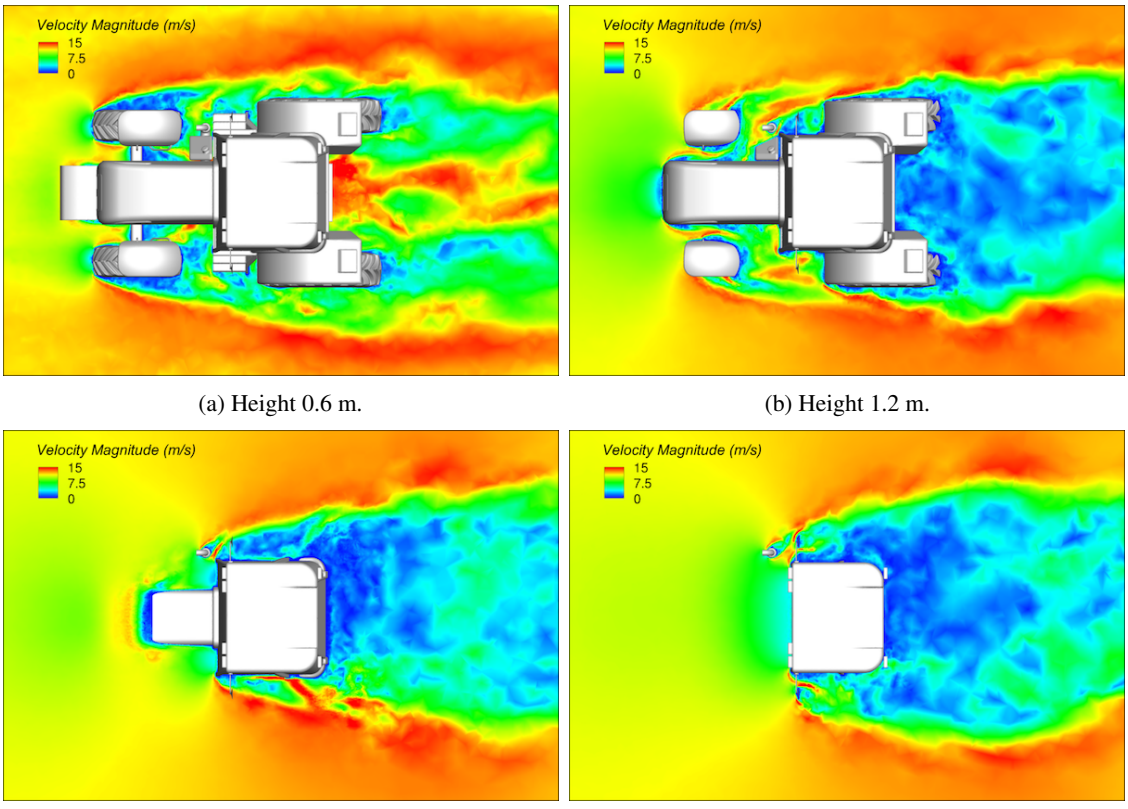


Figure 18: Visualization of the immersogeometric result of instantaneous vortical structures for turbulent flow around the tractor.



(a) Height 0.6 m.

(b) Height 1.2 m.

(c) Height 1.8 m.

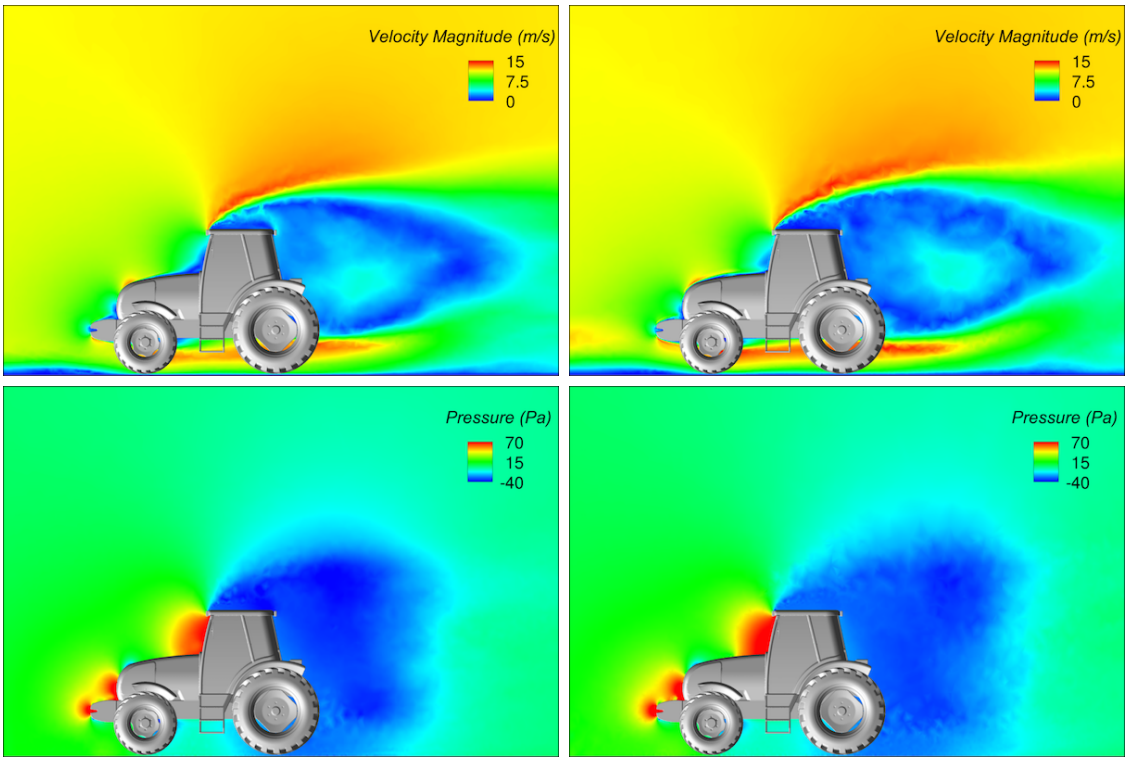
(d) Height 2.4 m.

Figure 19: Snapshots of the instantaneous velocity magnitude on planar cuts at different heights above the ground, computed with our immersogeometric method.

of heavy vehicles traveling at 25 mph is in the range of 0.7–0.9 [109].

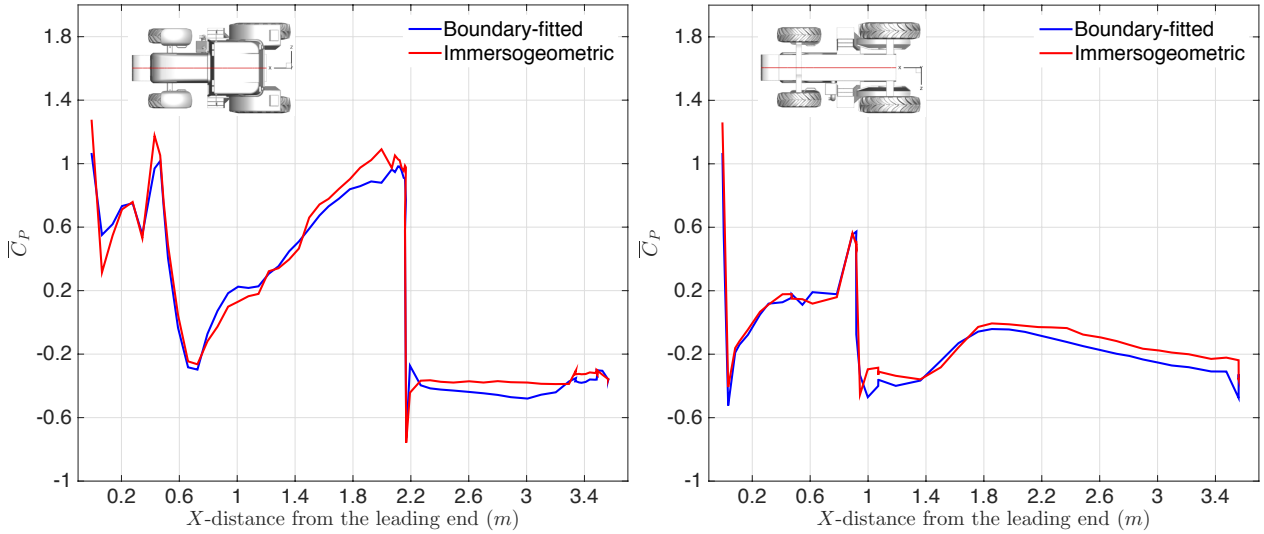
Figure 20 displays time-averaged velocity and pressure fields on a planar cut. We observe that immersogeometric results are in excellent accordance with the boundary-fitted results. To assess the accuracy of the immersogeometric results at single points of the surface, we plot the distribution of the time-averaged pressure coefficient $\overline{C}_P = 2(\overline{P} - P_\infty)/\rho U^2$ along curves over the tractor surface. The results are plotted for a curve over the top surface and one over the bottom surface of the tractor in Figs. 21a and 21b, respectively. Overall good agreement between the two methods is observed. This shows that our immersogeometric method is able to achieve accurate flow solutions near the boundary of an immersed object, where all elements are intersected, for high Reynolds number turbulent flow problems.

As discussed in Section 4.3.1, the use of integration subcells improves the immersogeometric solution quality, but



(a) Boundary-fitted. (b) Immersogeometric.

Figure 20: Time-averaged velocity and pressure fields on a planar cut.



(a) $\overline{C}_P$ distribution on the top surface. (b) $\overline{C}_P$ distribution on the bottom surface.

Figure 21: Time-averaged pressure coefficient $\overline{C}_P$ plotted along two curves over the tractor surface.

comes at the cost of additional computation during the finite element assembly procedures. The additional computations are in the form of quadrature over elements, and do not require any communication between subdomains. The impact of this additional cost on wall clock time may therefore be mitigated by simply partitioning the mesh into more subdomains (each assigned to a processor core). To demonstrate this, we perform a scalability study of our immersogeometric method using the tractor example presented in this section. For each parallel test, we compute 200 time steps, each with three Newton iterations. 100, 100 and 250 GMRES iterations are used for the first, second and third Newton iterations, respectively. The computations are carried out on the Lonestar Linux cluster described in Remark 4. The results are shown in Figure 22 and demonstrate nearly ideal scaling. Although the immersogeometric case takes more wall clock time than the boundary-fitted case

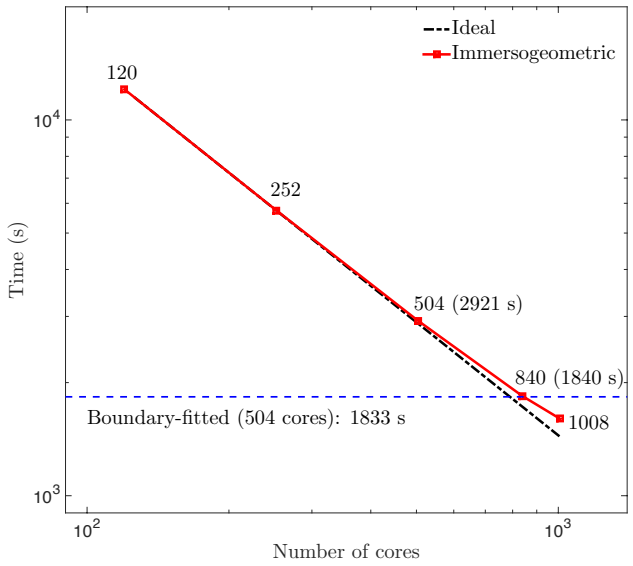


Figure 22: Scalability study for the example of turbulent flow around a tractor.

when 504 partitions are used for both, equivalent times may be achieved by simply partitioning the immersogeometric mesh into more subdomains. As the cost of access to supercomputing resources drops, it is not computer time but rather the time of human analysts—which is often spent on mesh design—that dominates overall analysis cost. We therefore believe that it is useful to investigate numerical methods that require moderate increases in computation but drastically simplify mesh generation.

6. Conclusions and future work

In this paper, we presented an immersogeometric framework for analyzing incompressible flows around geometrically complex objects immersed in non-boundary-fitted tetrahedral finite element meshes. The main components of this framework are the variational multiscale method, the weak enforcement of boundary conditions, an adaptive quadrature scheme for the integration of intersected elements, and the local refinement of areas with boundary layers.

We examined in detail two representative example problems: flow around a sphere and aerodynamic analysis of a tractor. We showed that the immersogeometric solutions were in good agreement with reference solutions, both in terms of characteristic parameters such as the drag coefficient and the Strouhal number, and in terms of near-boundary solution features such as the pressure distribution plotted over surface lines of immersed objects. We also demonstrated that such agreement is *not* achieved without the faithful representation of surface geometry provided by our approach. The tractor analysis indicates that our immersogeometric method can greatly simplify the mesh generation process for industrial turbulent flow problems without sacrificing accuracy.

Some future research directions in immersogeometric CFD include:

- Development of advanced quadrature schemes for intersected elements that are geometrically faithful and computationally efficient.
- Improved weak boundary and coupling conditions that limit the dependence on stabilization parameters and maintain a good conditioning of the system matrix.
- Boundary layer refinement strategies with anisotropic approximation power.
- Extension of the methods from this paper to higher-order finite element spaces.
- Efficient treatment of moving immersed boundaries.

The last of these points is of particular importance to immersogeometric FSI analysis [43, 110, 111].

Acknowledgments

We thank the Texas Advanced Computing Center (TACC) at the University of Texas at Austin and the Minnesota Supercomputing Institute (MSI) for providing HPC resources that have contributed to the research results reported within this paper. F. Xu, C. Wang and M.-C. Hsu were partially supported by the ARO grant No. W911NF-14-1-0296. This support is gratefully acknowledged.

References

[1] C. S. Peskin. Flow patterns around heart valves: A numerical method. *Journal of Computational Physics*, 10(2):252–271, 1972.

[2] R. Scardovelli and S. Zaleski. Direct numerical simulation of free-surface and interfacial flow. *Annual Review of Fluid Mechanics*, 31(1):567–603, 1999.

[3] C. S. Peskin. The immersed boundary method. *Acta Numerica*, 11:479–517, 2002.

[4] R. Mittal and G. Iaccarino. Immersed boundary methods. *Annual Review of Fluid Mechanics*, 37:239–261, 2005.

[5] F. Sotiropoulos and X. Yang. Immersed boundary methods for simulating fluid–structure interaction. *Progress in Aerospace Sciences*, 65:1–21, 2014.

[6] T. J. R. Hughes. *The Finite Element Method. Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[7] R. Löhner, J. D. Baum, E. Mestreau, D. Sharov, C. Charman, and D. Pelessone. Adaptive embedded unstructured grid methods. *International Journal for Numerical Methods in Engineering*, 60(3):641–660, 2004.

[8] R. Löhner, R. J. Cebral, F. E. Camelli, S. Appanaboyina, J. D. Baum, E. L. Mestreau, and O. A. Soto. Adaptive embedded and immersed unstructured grid techniques. *Computer Methods in Applied Mechanics and Engineering*, 197:2173–2197, 2008.

[9] R. Löhner. *Applied Computational Fluid Dynamics Techniques: An Introduction Based on Finite Element Methods, Second Edition*. John Wiley & Sons, Chichester, 2008.

[10] R. Glowinski, T.-W. Pan, T. I. Hesla, and D. D. Joseph. A distributed Lagrange multiplier/fictitious domain method for particulate flows. *International Journal of Multiphase Flow*, 25(5):755–794, 1999.

[11] R. Glowinski, T. W. Pan, T. I. Hesla, D. D. Joseph, and J. Périaux. A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: Application to particulate flow. *Journal of Computational Physics*, 169(2):363–426, 2001.

[12] R. Glowinski and Y. Kuznetsov. Distributed Lagrange multipliers based on fictitious domain method for second order elliptic problems. *Computer Methods in Applied Mechanics and Engineering*, 196:1498–1506, 2007.

[13] L. Zhang, A. Gerstenberger, X. Wang, and W. K. Liu. Immersed finite element method. *Computer Methods in Applied Mechanics and Engineering*, 193:2051–2067, 2004.

[14] W. K. Liu, D. W. Kim, and S. Tang. Mathematical foundations of the immersed finite element method. *Computational Mechanics*, 39(3):211–222, 2007.

[15] X. S. Wang, L. T. Zhang, and W. K. Liu. On computational issues of immersed finite element methods. *Journal of Computational Physics*, 228(7):2535–2551, 2009.

[16] X. Wang and L. T. Zhang. Modified immersed finite element method for fully-coupled fluid–structure interactions. *Computer Methods in Applied Mechanics and Engineering*, 267:150–169, 2013.

[17] H. Casquero, C. Bona-Casas, and H. Gomez. A NURBS-based immersed methodology for fluid–structure interaction. *Computer Methods in Applied Mechanics and Engineering*, 284:943–970, 2015.

[18] F. P. T. Baaijens. A fictitious domain/mortar element method for fluid–structure interaction. *International Journal for Numerical Methods in Fluids*, 35(7):743–761, 2001.

[19] L. Parussini. Fictitious domain approach via Lagrange multipliers with least squares spectral element method. *Journal of Scientific Computing*, 37(3):316–335, 2008.

[20] L. Parussini and V. Pediroda. Fictitious domain approach with *hp*-finite element approximation for incompressible fluid flow. *Journal of Computational Physics*, 228(10):3891–3910, 2009.

[21] A. Gerstenberger and W. A. Wall. Enhancement of fixed-grid methods towards complex fluid–structure interaction applications. *International Journal for Numerical Methods in Fluids*, 57:1227–1248, 2008.

[22] A. Gerstenberger and W. A. Wall. An embedded Dirichlet formulation for 3D continua. *International Journal for Numerical Methods in Engineering*, 82:537–563, 2010.

[23] S. Shahmiri, A. Gerstenberger, and W. A. Wall. An XFEM-based embedding mesh technique for incompressible viscous flows. *International Journal for Numerical Methods in Fluids*, 65:166–190, 2011.

[24] T. Rüberg and F. Cirak. Subdivision-stabilised immersed B-spline finite elements for moving boundary flows. *Computer Methods in Applied Mechanics and Engineering*, 209–212:266–283, 2012.

[25] T. Rüberg and F. Cirak. A fixed-grid b-spline finite element technique for fluid–structure interaction. *International Journal for Numerical Methods in Fluids*, 74(9):623–660, 2014.

[26] J. Baiges and R. Codina. The fixed-mesh ALE approach applied to solid mechanics and fluid–structure interaction problems. *International Journal for Numerical Methods in Engineering*, 81:1529–1557, 2010.

[27] T. Wick. Fully Eulerian fluid–structure interaction for time-dependent problems. *Computer Methods in Applied Mechanics and Engineering*, 255:14–26, 2013.

[28] T. Richter and T. Wick. Finite elements for fluid–structure interaction in ALE and fully Eulerian coordinates. *Computer Methods in Applied Mechanics and Engineering*, 199:2633–2642, 2010.

[29] C. Hesch, A. J. Gil, A. Arranz Carreño, and J. Bonet. On continuum immersed strategies for fluid–structure interaction. *Computer Methods in Applied Mechanics and Engineering*, 247-248:51–64, 2012.

[30] T. Richter. A fully Eulerian formulation for fluid–structure-interaction problems. *Journal of Computational Physics*, 233:227–240, 2013.

[31] T. J. R. Hughes, L. Mazzei, and K. E. Jansen. Large eddy simulation and the variational multiscale method. *Computing and Visualization in Science*, 3:47–59, 2000.

[32] T. J. R. Hughes, L. Mazzei, A. A. Oberai, and A. Wray. The multiscale formulation of large eddy simulation: Decay of homogeneous isotropic turbulence. *Physics of Fluids*, 13:505–512, 2001.

[33] T. J. R. Hughes, G. Scovazzi, and L. P. Franca. Multiscale and stabilized methods. In E. Stein, R. de Borst, and T. J. R. Hughes, editors, *Encyclopedia of Computational Mechanics*, Volume 3: Fluids, chapter 2. John Wiley & Sons, 2004.

[34] Y. Bazilevs, V. M. Calo, J. A. Cottrel, T. J. R. Hughes, A. Reali, and G. Scovazzi. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 197:173–201, 2007.

[35] Y. Bazilevs and T. J. R. Hughes. Weak imposition of Dirichlet boundary conditions in fluid mechanics. *Computers & Fluids*, 36:12–26, 2007.

[36] Y. Bazilevs, C. Michler, V. M. Calo, and T. J. R. Hughes. Weak Dirichlet boundary conditions for wall-bounded turbulent flows. *Computer Methods in Applied Mechanics and Engineering*, 196:4853–4862, 2007.

[37] Y. Bazilevs, C. Michler, V. M. Calo, and T. J. R. Hughes. Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes. *Computer Methods in Applied Mechanics and Engineering*, 199:780–790, 2010.

[38] M.-C. Hsu, I. Akkerman, and Y. Bazilevs. Wind turbine aerodynamics using ALE–VMS: Validation and the role of weakly enforced boundary conditions. *Computational Mechanics*, 50:499–511, 2012.

[39] A. Stavrev. *The Role of Higher-Order Geometry Approximation and Accurate Quadrature in NURBS Based Immersed Boundary Methods*. Master's Thesis, Technische Universität München, 2012.

[40] L. Kudela. *Highly Accurate Subcell Integration in the Context of The Finite Cell Method*. Master's Thesis, Technische Universität München, 2013.

[41] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[42] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, Chichester, 2009.

[43] D. Kamensky, M.-C. Hsu, D. Schillinger, J. A. Evans, A. Aggarwal, Y. Bazilevs, M. S. Sacks, and T. J. R. Hughes. An immersogeometric variational framework for fluid–structure interaction: Application to bioprosthetic heart valves. *Computer Methods in Applied Mechanics and Engineering*, 284:1005–1053, 2015.

[44] J. Parvizian, A. Düster, and E. Rank. Finite cell method: *h*- and *p*- extension for embedded domain methods in solid mechanics. *Computational Mechanics*, 41:122–133, 2007.

[45] A. Düster, J. Parvizian, Z. Yang, and E. Rank. The finite cell method for three-dimensional problems of solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 197:3768–3782, 2008.

[46] D. Schillinger and M. Ruess. The Finite Cell Method: A review in the context of higher-order structural analysis of CAD and image-based geometric models. *Archives of Computational Methods in Engineering*, 22(3):391–455, 2015.

[47] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn, and E. Rank. Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method. *International Journal for Numerical Methods in Engineering*, 95(10):811–846, 2013.

[48] M. Ruess, D. Schillinger, A. I. Özcan, and E. Rank. Weak coupling for isogeometric analysis of non-matching and trimmed multi-patch geometries. *Computer Methods in Applied Mechanics and Engineering*, 269:46–71, 2014.

[49] D. Schillinger and E. Rank. An unfitted *hp* adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry. *Computer Methods in Applied Mechanics and Engineering*, 200(47–48):3358–3380, 2011.

[50] D. Schillinger, A. Düster, and E. Rank. The *hp-d* adaptive finite cell method for geometrically nonlinear problems of solid mechanics. *International Journal for Numerical Methods in Engineering*, 89:1171–1202, 2012.

[51] D. Schillinger, L. Dedè, M. A. Scott, J. A. Evans, M. J. Borden, E. Rank, and T. J. R. Hughes. An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 249-250:116–150, 2012.

[52] M. Joulaian and A. Düster. Local enrichment of the finite cell method for problems with material interfaces. *Computational Mechanics*, 52:741–762, 2013.

[53] N. Zander, T. Bog, S. Kollmannsberger, D. Schillinger, and E. Rank. Multi-level *hp*-adaptivity: high-order mesh adaptivity without the difficulties of constraining hanging nodes. *Computational Mechanics*, 55:499–517, 2015.

[54] Z. Yang, M. Ruess, S. Kollmannsberger, A. Düster, and E. Rank. An efficient integration technique for the voxel-based finite cell method. *International Journal for Numerical Methods in Engineering*, 91:457–471, 2012.

[55] D. Schillinger, S. Kollmannsberger, R.-P. Mundani, and E. Rank. The finite cell method for geometrically nonlinear problems of solid mechanics. *IOP Conference Series: Material Science and Engineering*, 10:012170, 2010.

[56] D. Schillinger, M. Ruess, N. Zander, Y. Bazilevs, A. Düster, and E. Rank. Small and large deformation analysis with the *p*- and B-spline versions of the Finite Cell Method. *Computational Mechanics*, 50(4):445–478, 2012.

[57] N. Zander, S. Kollmannsberger, M. Ruess, Z. Yosibash, and E. Rank. The Finite Cell Method for linear thermoelasticity. *Computers & Mathematics with Applications*, 64(11):3527–3541, 2012.

[58] A. Düster, H.-G. Sehlhorst, and E. Rank. Numerical homogenization of heterogeneous and cellular materials utilizing the finite cell method. *Computational Mechanics*, 50:413–431, 2012.

[59] M. Ruess, D. Tal, N. Trabelsi, Z. Yosibash, and E. Rank. The finite cell method for bone simulations: Verification and validation. *Biomechanics and Modeling in Mechanobiology*, 11(3):425–437, 2012.

[60] J. Parvizian, A. Düster, and E. Rank. Topology optimization using the finite cell method. *Optimization and Engineering*, 13:57–78, 2012.

[61] C. Willberg, S. Duczek, J. M. Vivar Perez, D. Schmicker, and U. Gabbert. Comparison of different higher order finite element schemes for the simulation of Lamb waves. *Computer Methods in Applied Mechanics and Engineering*, 241–244:246–261, 2012.

[62] S. Duczek, M. Joulaian, A. Düster, and U. Gabbert. Numerical analysis of Lamb waves using the finite and spectral cell methods. *International Journal for Numerical Methods in Engineering*, 99(1):26–53, 2014.

[63] M. Joulaian, S. Duczek, U. Gabbert, and A. Düster. Finite and spectral cell method for wave propagation in heterogeneous materials. *Computational Mechanics*, 54(1):661–675, 2014.

[64] N. Zander, T. Bog, M. Elhaddad, R. Espinoza, H. Hu, A. F. Joly, C. Wu, P. Zerbe, A. Dster, S. Kollmannsberger, J. Parvizian, M. Ruess, D. Schillinger, and E. Rank. FCMLab: A finite cell research toolbox for MATLAB. *Advances in Engineering Software*, 74:49–63,, 2014.

[65] V. Varduhn, M.-C. Hsu, M. Ruess, and D. Schillinger. The tetrahedral finite cell method: Higher-order immersogeometric analysis on adaptive non-boundary-fitted meshes. *Submitted to International Journal for Numerical Methods in Engineering*, 2015.

[66] A. N. Brooks and T. J. R. Hughes. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32:199–259, 1982.

[67] T. E. Tezduyar. Stabilized finite element formulations for incompressible flow computations. *Advances in Applied Mechanics*, 28:1–44, 1992.

[68] T. E. Tezduyar and Y. Osawa. Finite element stabilization parameters computed from element matrices and vectors. *Computer Methods in Applied Mechanics and Engineering*, 190:411–430, 2000.

[69] M.-C. Hsu, Y. Bazilevs, V. M. Calo, T. E. Tezduyar, and T. J. R. Hughes. Improving stability of stabilized and multiscale formulations in flow simulations at small time steps. *Computer Methods in Applied Mechanics and Engineering*, 199:828–840, 2010.

[70] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Cambridge University Press, Sweden, 1987.

[71] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods, 2nd ed.* Springer, Berlin, 2002.

[72] A. Ern and J. L. Guermond. *Theory and Practice of Finite Elements*. Springer, Berlin, 2004.

[73] J. Nitsche. Uber ein variationsprinzip zur losung von Dirichlet-problemen bei verwendung von teilraumen, die keinen randbedingungen unterworfen sind. *Abh. Math. Univ. Hamburg*, 36:9–15, 1971.

[74] A. Embar, J. Dolbow, and I. Harari. Imposing Dirichlet boundary conditions with Nitsche's method and spline-based finite elements. *International Journal for Numerical Methods in Engineering*, 83(7):877–898, 2010.

[75] Y. Bazilevs and I. Akkerman. Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual-based variational multiscale method. *Journal of Computational Physics*, 229:3402–3414, 2010.

[76] N. Kikuchi. A smoothing technique for reduced integration penalty methods in contact problems. *International Journal for Numerical Methods in Engineering*, 18(3):343–350, 1982.

[77] J. D. Sanders, J. E. Dolbow, and T. A. Laursen. On methods for stabilizing constraints over enriched interfaces in elasticity. *International Journal for Numerical Methods in Engineering*, 78:1009–1036, 2009.

[78] F. Liu and R. I. Borja. Stabilized low-order finite elements for frictional contact with the extended finite element method. *Computer Methods in Applied Mechanics and Engineering*, 199:2456–2471, 2010.

[79] L. De Lorenzis, İ. Temizer, P. Wriggers, and G. Zavarise. A large deformation frictional contact formulation using NURBS-based isogeometric analysis. *International Journal for Numerical Methods in Engineering*, 87:1278–1300, 2011.

[80] R. A. Sauer and L. De Lorenzis. A computational contact formulation based on surface potentials. *Computer Methods in Applied Mechanics and Engineering*, 253(0):369–395, 2013.

[81] J. Chung and G. M. Hulbert. A time integration algorithm for structural dynamics with improved numerical dissipation: The generalized-α method. *Journal of Applied Mechanics*, 60:371–75, 1993.

[82] K. E. Jansen, C. H. Whiting, and G. M. Hulbert. A generalized-α method for integrating the filtered Navier-Stokes equations with a stabilized finite element method. *Computer Methods in Applied Mechanics and Engineering*, 190:305–319, 2000.

[83] Y. Bazilevs, V. M. Calo, T. J. R. Hughes, and Y. Zhang. Isogeometric fluid–structure interaction: theory, algorithms, and computations. *Computational Mechanics*, 43:3–37, 2008.

[84] Y. Saad and M. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal of Scientific and Statistical Computing*, 7:856–869, 1986.

[85] F. Shakib, T. J. R. Hughes, and Z. Johan. A multi-element group preconditioned GMRES algorithm for nonsymmetric systems arising in finite element analysis. *Computer Methods in Applied Mechanics and Engineering*, 75:415–456, 1989.

[86] C. Pozrikidis. *Introduction to Finite and Spectral Element Methods Using MATLAB, Second Edition*. Taylor & Francis Group, Boca Raton, FL, 2014.

[87] I. Wald, W. R. Mark, J. Günther, S. Boulos, T. Ize, W. Hunt, S. G. Parker, and P. Shirley. State of the art in ray tracing animated scenes. In *Proceedings of the Annual Conference of the European Association for Computer Graphics*, pages 89–116, 2007.

[88] J. Bigler, A. Stephens, and S. G. Parker. Design for parallel interactive ray tracing systems. In *Proceedings of the IEEE Symposium on Interactive Ray Tracing 2006*, pages 187–196, 2006.

[89] S. J. Owen. A survey of unstructured mesh generation technology. In *Proceedings of the 7th International Meshing Roundtable*, pages 239–267, Sandia National Lab, 1998.

[90] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[91] T. A. Johnson and V. C. Patel. Flow past a sphere up to a Reynolds number of 300. *Journal of Fluid Mechanics*, 378:19–70, 1999.

[92] R. Mittal. A Fourier–Chebyshev spectral collocation method for simulating flow past spheres and spheroids. *International journal for numerical methods in fluids*, 30(7):921–937, 1999.

[93] S. Marella, S. Krishnan, H. Liu, and H. S. Udaykumar. Sharp interface Cartesian grid method I: An easily implemented technique for 3D moving boundary computations. *Journal of Computational Physics*, 210(1):1–31, 2005.

[94] J.-I. Choi, R. C. Oberoi, J. R. Edwards, and J. A. Rosati. An immersed boundary method for complex incompressible flows. *Journal of Computational Physics*, 224(2):757–784, 2007.

[95] R. Mittal, H. Dong, M. Bozkurttas, F. M. Najjar, A. Vargas, and A. von Loebbecke. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. *Journal of computational physics*, 227(10):4825–4852, 2008.

[96] G. Yun, D. Kim, and H. Choi. Vortical structures behind a sphere at subcritical Reynolds numbers. *Physics of Fluids*, 18(1):015102–14, 2006.

[97] I. Rodriguez, R. Borell, O. Lehmkuhl, C. D. Perez Segarra, and A. Oliva. Direct numerical simulation of the flow over a sphere at Re = 3700. *Journal of Fluid Mechanics*, 679:263–287, 2011.

[98] Y. Bazilevs, J. Yan, M. de Stadler, and S. Sarkar. Computation of the flow over a sphere at Re = 3700: A comparison of uniform and turbulent inflow conditions. *Journal of Applied Mechanics*, 81(12):121003, 2014.

[99] E. H. van Brummelen, K. G. van der Zee, V. V. Garg, and S. Prudhomme. Flux evaluation in primal and dual boundary-coupled problems. *Journal of Applied Mechanics*, 79:010904, 2011.

[100] R. V. Garimella and M. S. Shephard. Boundary layer mesh generation for viscous flow simulations. *International Journal for Numerical Methods in Engineering*, 49(1-2):193–218, 2000.

[101] X. Li, M. S. Shephard, and M. W. Beall. 3D anisotropic mesh adaptation by mesh modification. *Computer Methods in Applied Mechanics and Engineering*, 194(48-49):4915–4950, 2005.

[102] Y. Zhang, W. Wang, X. Liang, Y. Bazilevs, M.-C. Hsu, T. Kvamsdal, R. Brekken, and J.G. Isaksen. High-fidelity tetrahedral mesh generation from medical imaging data for fluid–structure interaction analysis of cerebral aneurysms. *Computer Modeling in Engineering & Sciences*, 42:131–150, 2009.

[103] TACC Lonestar User Guide. https://portal.tacc.utexas.edu/user-guides/lonestar. 2015.

[104] Texas Advanced Computing Center (TACC). http://www.tacc.utexas.edu. 2015.

[105] M.-C. Hsu, I. Akkerman, and Y. Bazilevs. High-performance computing of wind turbine aerodynamics using isogeometric analysis. *Computers & Fluids*, 49:93–100, 2011.

[106] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1999.

[107] J. Jeong and F. Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285:69–94, 1995.

[108] BETA CAE Systems S.A. – ANSA pre-processor. http://www.beta-cae.gr/ansa.htm. 2015.

[109] V. Malviya, R. Mishra, and J. D. Fieldhouse. CFD investigation of a novel fuel-saving device for articulated tractor-trailer combinations. *Engineering Applications of Computational Fluid Mechanics*, 3(4):587–607, 2009.

[110] M.-C. Hsu, D. Kamensky, Y. Bazilevs, M. S. Sacks, and T. J. R. Hughes. Fluid–structure interaction analysis of bioprosthetic heart valves: significance of arterial wall deformation. *Computational Mechanics*, 54(4):1055–1071, 2014.

[111] M.-C. Hsu, D. Kamensky, F. Xu, J. Kiendl, C. Wang, M. C. H. Wu, J. Mineroff, A. Reali, Y. Bazilevs, and M. S. Sacks. Dynamic and fluid–structure interaction simulations of bioprosthetic heart valves using parametric design with T-splines and Fung-type material models. *Computational Mechanics*, 55:1211–1225, 2015.