Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 404 (2023) 115742
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Direct immersogeometric fluid flow and heat transfer analysis of objects represented by point clouds

Aditya Balu[a], Manoj R. Rajanna[a,1], Joel Khristy[a,1], Fei Xu[b], Adarsh Krishnamurthy[a,*], Ming-Chen Hsu[a,*]

[a] Department of Mechanical Engineering, Iowa State University, 2043 Black Engineering, Ames, IA 50011, USA
[b] Ansys Inc., 807 Las Cimas Parkway, Austin, TX 78746, USA

Received 2 September 2022; received in revised form 23 October 2022; accepted 23 October 2022
Available online xxxx

Abstract

Immersogeometric analysis (IMGA) is a geometrically flexible method that enables one to perform multiphysics analysis directly using complex computer-aided design (CAD) models. While the IMGA approach is well-studied and has a remarkable advantage over traditional CFD, IMGA still requires a well-defined B-rep model to represent the geometry. Obtaining such a model can sometimes be equally as challenging as creating a body-fitted mesh. To address this issue, we develop a novel IMGA approach for the simulation of incompressible and compressible flows around complex geometries represented by point clouds in this work. The point cloud representation of geometries is a direct method for digitally acquiring geometric information using LiDAR scanners, optical scanners, or other passive methods such as multi-view stereo images. The point cloud object's geometry is represented using a set of unstructured points in the Euclidean space with (possible) orientation information in the form of surface normals. Due to the absence of topological information in the point cloud model, there are no guarantees for the geometric representation to be watertight or 2-manifold or to have consistent normals. To perform IMGA directly using point cloud geometries, we first develop a method for estimating the inside–outside information and the surface normals directly from the point cloud. We also propose a method to compute the Jacobian determinant for the surface integration (over the point cloud) necessary for the weak enforcement of Dirichlet boundary conditions. We validate these geometric estimation methods by comparing the geometric quantities computed from the point cloud with those obtained from analytical geometry and tessellated CAD models. In this work, we also develop thermal IMGA to simulate heat transfer in the presence of flow over complex geometries. The proposed framework is tested for a wide range of Reynolds and Mach numbers on benchmark problems of geometries represented by point clouds, showing the robustness and accuracy of the method. Finally, we demonstrate the applicability of our approach by performing IMGA on large industrial-scale construction machinery represented using a point cloud of more than 12 million points.

Keywords: Immersogeometric analysis; Point clouds; Geometric algorithms; Winding number; Weakly enforced Dirichlet boundary conditions; Nitsche's method

* Corresponding authors.
E-mail addresses: adarsh@iastate.edu (A. Krishnamurthy), jmchsu@iastate.edu (M.-C. Hsu).
[1] These two authors contributed equally to this work.

https://doi.org/10.1016/j.cma.2022.115742
0045-7825/

1. Introduction

In immersogeometric analysis (IMGA), a solid object is immersed into a non-boundary-fitted discretization of the background fluid domain that is used to solve flow physics using finite-element-based computational fluid dynamics (CFD) [1,2]. IMGA alleviates the labor-intensive and time-consuming geometry cleanup process needed to create a boundary-conforming fluid mesh to perform traditional CFD. For example, in boundary-fitted CFD mesh generation, small and thin geometric features often need to be manually defeatured, and the resulting gaps in the boundary representation (B-rep) of the solid model need to be filled to create a watertight surface representation of the solid object. This process can take up to several months for extensive industrial-scale simulations of flow over complex geometries such as tractors and trucks [3–6].

IMGA has been shown to be a practical method [2,7–9] for the simulation of incompressible flow (both laminar and turbulent) around geometrically complex objects in the context of a tetrahedral finite cell approach [10]. Since the fluid domain is meshed independently, defeaturing or removing small geometric features from the immersed object is no longer necessary. The method was extended by Xu et al. [11,12] to handle moving particles, by Zhu et al. [13] to consider free-surface flows, and by Saurabh et al. [14,15] to perform industrial scale large eddy simulations using adaptive octree meshes. Xu et al. [16] recently proposed a compressible-flow version of the IMGA formulation for the simulation of aircraft aerodynamics, and Hoang et al. [17] developed a skeleton-stabilized IMGA technique for the simulation of fluid flow through a porous medium. The immersogeometric approach has also been shown as an efficient method to solve computational fluid–structure interaction (FSI) problems for heart valve applications [18–26]. It is also flexible enough to be automated and placed in an optimization loop that searches for an optimal design [27]. A subset of the immersogeometric FSI functionality was recently implemented as the open-source library CouDALFISh [28,29].

We have previously developed the immersogeometric method to directly use the B-rep of the computer-aided design (CAD) model to perform fluid flow simulations. We were able to perform IMGA of flow over B-rep models represented using triangles [19], trimmed non-uniform rational B-splines (NURBS) [7], and finally analytic surfaces [8]. These approaches were able to handle complex geometry without needing any geometric defeaturing. However, they still worked based on the boundary representation of the well-defined CAD model (2-manifold, watertight, and with consistent normals). This paper extends the work to perform IMGA directly with point cloud representation of the solid geometry, which relaxes these restrictions (manifoldness and watertightness of the boundary representation).

In a point cloud representation, the boundary of the object is represented using a set of unstructured points in the Euclidean space with (possible) orientation information in the form of surface normals. Many geometric data acquisition methods, including optical laser-based scanners, LiDAR scanners, and even passive methods such as multi-view stereo, produce a point cloud representation of the geometry [30]. Further, there has been a high interest in performing flow simulations over as-manufactured or in-use objects rather than ideal as-designed objects for performing analysis (and to provide feedback) using digital twins [31–35]. In practice, we need the geometric information corresponding to the in-use physical object to analyze the flow over them. While it is possible to perform IMGA by first reconstructing the boundary surfaces and then using them for the analysis, reconstructing the surfaces to generate watertight and manifold solid models is by itself very challenging. In addition, such an approach would relax the restrictions on the B-rep CAD models obtained directly after design. The geometry no longer needs to be cleaned up to ensure it is watertight; points can be directly sampled from boundary surfaces to perform IMGA.

There are two main geometry processing steps for performing IMGA directly on point cloud representations. First, we need to perform an inside–outside test to conduct a point membership classification (PMC) on the background fluid mesh based only on the point cloud representation of the solid model. Second, we need to perform surface integration over the point cloud to impose the weak Dirichlet boundary conditions [36]. Surface integration requires estimating the surface normals and area for each point in the point cloud to be used as the Jacobian determinant for each surface element during integration. In addition, the fluid mesh needs to be adequately refined near the surface locations, i.e., the region around the points of the point cloud. In this paper, we exploit the prior art in geometric processing methods to achieve these steps.

Inside–outside evaluation for the background fluid mesh is necessary to identify points inside the solid geometry. Traditionally, a point membership classification is performed over the solid CAD model. However, these approaches are restricted to 2-manifold solids with a watertight surface representation. Since point clouds are not manifold

and do not have any inherent order, we propose using the generalized winding-number-based inside–outside testing approach to perform the point membership classification. This method relaxes the manifold and watertight requirements of the surface representation.

As mentioned earlier, the immersed objects used in previous immersogeometric methods consisted of tessellations, trimmed NURBS, or analytic surfaces. In the case of triangles, the Gaussian quadrature points were directly generated on the planar triangular surfaces [2]. In the case of trimmed NURBS, the NURBS parameterization was used to generate the quadrature points [7]. Similarly, the surface parametric equations were used to generate the Gaussian quadrature points for trimmed analytic surfaces [8]. In this work, it is convenient for the integration points to be co-located with the points of the point cloud. However, this approach requires calculating the Jacobian of the integration for each point in the point cloud. In this paper, we have developed a Voronoi projection-based area to compute the Jacobian determinant (or the effective area) associated with each point of the point cloud. In addition, we also need the surface normal corresponding to each point in the point cloud. We use a local hyperplane fitting algorithm to compute the surface normals.

In many industrial applications, fluid flow analyses are employed to verify and validate the efficacy of thermal control systems. The thermal analysis predicts the surface and ambient temperatures of critical components in an industrial product assembly. Previous work demonstrated the capability to simulate heat transfer using a variational multiscale method with weakly enforced Dirichlet conditions on conforming boundaries [37]. In this paper, we develop a thermal IMGA formulation to apply fixed temperature boundary conditions weakly on immersed point cloud surfaces, showing convection and conduction of thermal quantities in the flow. We first validate the proposed formulation using benchmark problems and later demonstrate the utility of thermal IMGA on a large vehicle assembly.

To summarize, the specific contributions in this paper include: (1) Methods to compute the surface Jacobian determinant and surface normals from the point cloud representation of a solid model. (2) Methods to compute the inside–outside information for classifying the quadrature points of the background fluid mesh. (3) A thermal IMGA formulation for modeling the heat transfer in flow over an immersed object. (4) Validation studies on the proposed methods to understand their efficacy in performing IMGA with a point cloud. (5) Demonstration of coupled fluid and thermal flow analysis on a large industrial-scale object represented using a point cloud.

This paper is organized as follows. We provide the mathematical formulation of IMGA for incompressible and compressible flows in Section 2. In Section 3, we provide details on processing the point cloud for computing the normals, Jacobian determinant, and inside–outside information. We also perform validation studies for the point cloud processing methods. Next, we provide the details of the flow and thermal validation studies in Section 4 and the application of the method to industrial-scale parts represented as point clouds in Section 5. Finally, in Section 6, we conclude our work and provide a few directions for future work.

2. Immersogeometric analysis

The immersogeometric flow analysis methodology consists of three main components: (1) The thermal fluid system is modeled using stabilized finite element methods for incompressible [37–39] and compressible [40–42] flows. (2) The Dirichlet boundary conditions imposed on the immersed objects are enforced weakly in the sense of Nitsche's method [2,16,43]. (3) To accurately capture the geometry of the flow domain, the concept of the Finite Cell Method (FCM) is employed in which the quadrature rules are adaptively refined [2,44,45]. These numerical ingredients are presented in this section.

2.1. Mathematical formulation

2.1.1. Variational multiscale formulation of the incompressible thermal fluid flow

Let Ω (subsets of $\mathrm{R}^d, d \in \{2, 3\}$) denote the spatial domain, and Γ be its boundary. In the context of immersogeometric flow analysis, the computational domain Ω consists of two exclusive parts, the physical domain Ω_{phys}, i.e., the fluid domain, and the fictitious domain Ω_{fict}, i.e., the domain enclosed by solid objects. Ω_{phys} and Ω_{fict} are separated by the immersed boundary Γ, as shown in Fig. 1. Consider a collection of disjoint elements $\{\Omega^e\}$, $\Omega^e \subset \mathrm{R}^d_e$, with closures covering the computational domain, $\Omega \subset \bigcup \Omega^e$, and let Γ_e be discretized into a collection of boundary elements $\{\Gamma^b\}$. In what follows, $\Omega^e_{\text{phys}} = \Omega^e \cap \Omega_{\text{phys}}$ and $\Omega^e_{\text{fict}} = \Omega^e \cap \Omega_{\text{fict}}$. Note that the finite-element discretization of the domain Ω is created without conforming to the geometry Γ, which greatly

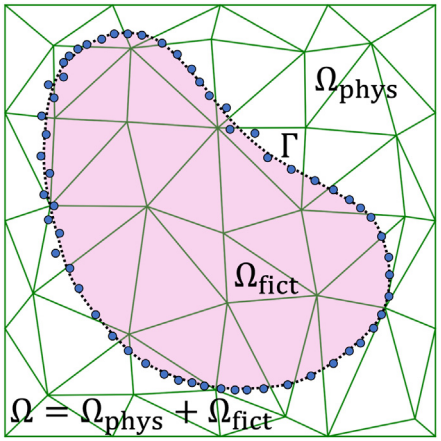


Fig. 1. An example of flow over an object. The object with boundary Γ is immersed into the domain Ω. The immersed boundary separates the domain Ω into a physical part Ω_{phys} and a fictitious part Ω_{fict}.

simplifies the mesh generation process. In what follows, a superscript h indicates that the variable is evaluated in the discrete space. The variational multiscale (VMS) discretization of the incompressible thermal fluid flow problem can be stated as: find the pressure p^h, fluid velocity u^h, and temperature T^h in the discrete solution space S^h, such that for all their corresponding test functions w_p^h, w^h, and w_T^h in the test function space V^h,

$$B^{IC}\left(\{w_p^h, w^h, w_T^h\}, \{p^h, u^h, T^h\}\right) - F^{IC}\left(\{w_p^h, w^h, w_T^h\}\right) = 0, \tag{1}$$

where

$$\begin{aligned}
B^{IC}&\left(\{w_p^h, w^h, w_T^h\}, \{p^h, u^h, T^h\}\right) = \\
&\int_{\Omega_{\text{phys}}} w^h \cdot \rho\left(\frac{\partial u^h}{\partial t} + u^h \cdot \nabla u^h\right) d\Omega + \int_{\Omega_{\text{phys}}} \boldsymbol{\varepsilon}(w^h) : \boldsymbol{\sigma}(u^h, p^h)\, d\Omega + \int_{\Omega_{\text{phys}}} w_p^h \nabla \cdot u^h \, d\Omega \\
&- \sum_e \int_{\Omega_{\text{phys}}^e} \left(\rho u^h \cdot \nabla w^h + \nabla w^h \cdot u'\right) d\Omega_p - \sum_e \int_{\Omega_{\text{phys}}^e} p' \nabla \cdot w^h \, d\Omega \\
&+ \sum_e \int_{\Omega_{\text{phys}}^e} \rho w^h \cdot (u' \cdot \nabla u^h)\, d\Omega - \sum_e \int_{\Omega_{\text{phys}}^e} \rho \nabla w^h : \left(u' \nabla u'\right) d\Omega \\
&+ \sum_e \int_{\Omega_{\text{phys}}^e} \rho\left(u' \cdot \nabla w^h\right) \tau \cdot \left(u' \cdot \nabla u^h\right) d\Omega \\
&+ \int_{\Omega_{\text{phys}}} w_T^h \rho c\left(\frac{\partial T^h}{\partial t} + u^h \cdot \nabla T^h\right) d\Omega + \int_{\Omega_{\text{phys}}} \nabla w_T^h \cdot \kappa \nabla T^h d\Omega \\
&+ \sum_e \int_{\Omega_{\text{phys}}^e} \nabla w_T^h \cdot \kappa_{DC}^{IC} \nabla T^h d\Omega \\
&- \sum_e \int_{\Omega_{\text{phys}}^e} \rho c\left(u^h \cdot \nabla w_T^h\right) T' d\Omega + \sum_e \int_{\Omega_{\text{phys}}^e} w_T^h \rho c\left(u' \cdot \nabla T^h\right) d\Omega \\
&- \sum_e \int_{\Omega_{\text{phys}}^e} \rho c \nabla w_T^h \cdot \left(u' T'\right) d\Omega,
\end{aligned} \tag{2}$$

and

$$F^{IC}\left(\{w_p^h, w^h, w_T^h\}\right) = \int_{\Omega_{\text{phys}}} w^h \cdot \rho f_{\text{buoy}}\, d\Omega + \int_{\Gamma_u^N} w^h \cdot h \, d\Gamma + \int_{\Gamma_T^N} w_T^h h_T \, d\Gamma. \tag{3}$$

In the above equations, ρ, c, and κ are the density, specific heat capacity, and thermal conductivity of the fluid, respectively. $\boldsymbol{\sigma}(u^h, p^h) = -p^h I + 2\mu\boldsymbol{\varepsilon}(u^h)$ and $\boldsymbol{\varepsilon}(u^h) = \frac{1}{2}(\nabla u^h + (\nabla u^h)^T)$ are the Cauchy stress and strain-rate

tensors of incompressible fluid, respectively, with μ being the dynamic viscosity. $\mathbf{I}$ is the $d \times d$ identity matrix if not otherwise specified. f_{buoy} is the buoyancy force per unit mass modeled using Boussinesq approximation as $-\beta\,(T - T_{ref})\,\mathbf{g}$, with β being the fluid thermal expansion coefficient, T_{ref} being the reference temperature, and $\mathbf{g}$ being the gravitational acceleration. h and h_T contain the prescribed traction and heat flux conditions, respectively, applied on Γ_u^N and Γ_T^N, which are the portions of Γ where the corresponding Neumann boundary conditions are applied. The fine-scale velocity, pressure, and temperature are defined by $u' = -\tau_M r_M/\rho$, $p' = -\rho\tau_C r_C$, and $T' = -\tau_E r_E/(\rho c)$, respectively, where

$$r_M = \rho\left(\frac{\partial u^h}{\partial t} + u^h \cdot \nabla u^h\right) - \nabla \cdot \boldsymbol{\sigma}_1\left(u^h, p^h\right) - \rho f_{buoy}\,, \tag{4}$$

$$r_C = \nabla \cdot u^h, \tag{5}$$

$$r_E = \rho c\left(\frac{\partial T^h}{\partial t} + u^h \cdot \nabla T^h\right) - \nabla \cdot \left(\kappa \nabla T^h\right). \tag{6}$$

In the above equations, the stabilization parameters are given by

$$\tau_M = \left(\frac{C_t}{\Delta t^2} + u^h \cdot G u^h + C_I \nu^2 G : G\right)^{-\frac{1}{2}}, \tag{7}$$

$$\tau_C = (\tau_M \mathrm{tr} G)^{-1}, \tag{8}$$

$$\tau_E = \left(\frac{C_t}{\Delta t^2} + u^h \cdot G u^h + C_I \alpha^2 G : G\right)^{-\frac{1}{2}}, \tag{9}$$

$$\bar{\tau} = \left(u' \cdot G u'\right)^{-\frac{1}{2}}, \tag{10}$$

where Δt is the time step size, the constants $C_t = 4$ and $C_I = 3$ are chosen from an appropriate element-wise inverse estimation [46,47], $\nu = \mu/\rho$ is the kinematic viscosity, $\alpha = \kappa/(\rho c)$ is the thermal diffusivity, and G contains the information about the element size derived from the element geometric mapping from the parametric parent element to physical coordinates $x(\xi)$. The components of G are defined as $G_{ij} = \frac{\partial \xi_k}{\partial x_i}\frac{\partial \xi_k}{\partial x_j}$. The term κ_{DC}^{IC} formulates a discontinuity capturing (DC) operator, which provides additional numerical stability to locations where temperature gradients are large. The DC stabilization parameter [48] is given by

$$\kappa_{DC}^{IC} = C_{DC}^{IC}\left(\nabla T^h \cdot G \nabla T^h\right)^{-\frac{1}{2}} |r_E|\,, \tag{11}$$

where C_{DC}^{IC} is a positive constant scaling the strength of the DC operator and is set to 0.5 in this paper.

The standard way of imposing Dirichlet boundary conditions is to enforce them strongly by ensuring that these conditions are satisfied by all trial solution functions, which is not feasible in immersed methods. Instead, the strong enforcement is replaced by weakly enforced Dirichlet boundary conditions originally introduced by Bazilevs et al. [36,49,50] for incompressible isothermal flows and later extended in Refs. [37,51,52] for thermal fluid flows. Let $\Gamma^D = \Gamma^D \cup \Gamma_u^D$ be the portions of Γ on which the velocity and temperature Dirichlet boundary conditions are applied. The semi-discrete problem of the thermal fluid system can now be stated as follows:

$$\begin{aligned}
&B^{IC}\left(\{w_p^h, w^h, w_T^h\}, \{p^h, u^h, T^h\}\right) - F^{IC}\left(\{w_p^h, w^h, w_T^h\}\right) \\
&- \sum_b \int_{\Gamma^b \cap \Gamma_u^D} w^h \cdot \left(-p^h n + 2\mu\boldsymbol{\varepsilon}(u^h) n\right) d\Gamma - \sum_b \int_{\Gamma^b \cap \Gamma_u^D} \left(w_p^h n + \tilde{\gamma}\, 2\mu\boldsymbol{\varepsilon}(w^h) n\right) \cdot \left(u^h - u_D\right) d\Gamma \\
&- \sum_b \int_{\Gamma^b \cap \Gamma_u^{D,-}} w^h \cdot \rho\left(u^h \cdot n\right)\left(u^h - u_D\right) d\Gamma + \sum_b \int_{\Gamma^b \cap \Gamma_u^D} w^h \cdot \tau_\mu^{IC}\left(u^h - u_D\right) d\Gamma \\
&- \sum_b \int_{\Gamma^b \cap \Gamma_T^D} w_T^h \kappa \nabla T^h \cdot n\, d\Gamma - \sum_b \int_{\Gamma^b \cap \Gamma_T^D} \tilde{\gamma}\, \kappa \nabla w^h \cdot n \left(T^h - T_D\right) d\Gamma \\
&- \sum_b \int_{\Gamma^b \cap \Gamma_T^{D,-}} w_T^h \rho c \left(u^h \cdot n\right)\left(T^h - T_D\right) d\Gamma + \sum_b \int_{\Gamma^b \cap \Gamma_T^D} w_T^h \tau_\kappa^{IC}\left(T^h - T_D\right) d\Gamma = 0.
\end{aligned} \tag{12}$$

In the above, u_D is the prescribed velocity on Γ_u^D, T_D is the prescribed temperature on Γ_T^D, n is the unit outward normal vector, and $\Gamma^{D,-}$ is the inflow part of Γ^D, on which $u^h \cdot n < 0$. The value of $\tilde{\gamma}$ can be selected

as 1 or −1 which determines whether Eq. (12) is a symmetric or non-symmetric type of Nitsche's method, respectively [16,36,53]. Finally, τ_{μ}^{IC} and τ_{κ}^{IC} are stabilization parameters that need to be estimated element-wise as a compromise between the conditioning of the stiffness matrix, variational consistency, and the stability of the formulation. The choice of $\tilde{\gamma}$ influences the performance of the weak boundary condition operator and the selection of $\tau^{IC}{}_{\mu}$ and $\tau^{IC}{}_{,\kappa}$ which will be discussed in detail in Section 2.1.3.

An important advantage of using weakly enforced Dirichlet boundary conditions is the release of the point-wise no-slip condition at the boundary of the fluid domain. This, in turn, allows the flow to slip slightly on the solid surface and imitates the presence of the thin boundary layer that typically needs to be resolved with spatial refinement. It was shown in Bazilevs et al. [50] and Hsu et al. [54] that weak boundary conditions allow for an accurate overall flow solution even if the mesh size in the wall-normal direction is relatively large. In the immersogeometric method, the fluid mesh is arbitrarily cut by the object boundary, leaving a boundary layer discretization of inferior quality compared to the boundary-fitted counterpart. However, it was shown in Xu et al. [2] that accurate flow solutions were obtained using the immersogeometric method with a mesh resolution and refinement pattern comparable to the boundary-fitted mesh used to obtain the reference values. We believe this is partially due to the use of weak-boundary-condition formulation.

2.1.2. Compressible flow formulation

The compressible-flow governing equations are discretized using a streamline upwind Petrov–Galerkin (SUPG) formulation [55–69] augmented by a DC operator [70–81]. In what follows, Roman indices take on values $\{1,\dots,d\}$, and summation convention on repeated indices is applied. In addition, we use $(\cdot)_{,t}$ to denote a partial time derivative, and we use $(\cdot)_{,i}$ to denote the spatial gradient. Let $Y = [p\ u\ T]^T$ denote the solution vector of pressure, velocity and temperature, and $W = [w_p\ w\ w_T]^T$ denote the test function vector of their respective test functions. The problem can be stated as follows: find $Y^h \in S^h$ such that for all $W^h \in V^h$,

$$B^C\left(W^h, Y^h\right) - F^C\left(W^h\right) = 0\,, \tag{13}$$

where

$$\begin{aligned} B^C\left(W^h, Y^h\right) = & \int_{\Omega_{\text{phys}}} W^h \cdot \left(A_0 Y^h_{,t} + A_i^{\text{adv}\backslash p} Y^h_{,i} + A_i^{sp} Y^h_{,i}\right) d\Omega \\ & - \int_{\Omega_{\text{phys}}} W^h_{,i} \cdot \left(A_i^p Y^h - K_{ij} Y^h_{,j}\right) d\Omega \\ & + \sum_e \int_{\Omega^e_{\text{phys}}} \left(\left(A_i + A_i^{sp}\right)^T W^h_{,i}\right) \cdot \left(A_0^{-1} \hat{\boldsymbol{\tau}}_{SUPG}\right) \text{Res}\left(Y^h\right) d\Omega \\ & + \sum_e \int_{\Omega^e_{\text{phys}}} \left(\hat{\kappa}_{DC} \hat{A}_0^h\right) Y^C_{,i} \, d\Omega, \end{aligned} \tag{14}$$

and

$$F^C\left(W^h\right) = \int_{\Omega} W^h \cdot S \, d\Omega + \int_{\Gamma_H} W^h \cdot H \, d\Gamma\,. \tag{15}$$

In the above, A's and K_{ij} are the Euler Jacobian matrices and the diffusivity matrix, respectively, whose specific definitions can be found in Appendix. Note that the superscript h for A's and K_{ij} are dropped for the clarity of notation, even though they are evaluated in the discrete space. H contains the prescribed fluid traction and heat flux boundary conditions, and Γ_H is the subset of Γ where H is specified. Res is the residual of the compressible-flow equations defined as

$$\text{Res}\left(Y^h\right) = A_0 Y^h_{,t} + \left(A_i + A_i^{sp}\right) Y^h_{,i} - \left(K_{ij} Y^h_{,j}\right)_{,i} - S\,. \tag{16}$$

The stabilization matrix $\hat{\boldsymbol{\tau}}_{SUPG}$ is defined as

$$\hat{\boldsymbol{\tau}}_{SUPG} = \left(\frac{C_t}{\Delta t^2} I + G_{ij} \left(\hat{A}_i + \hat{A}_i^{sp}\right)\left(\hat{A}_j + \hat{A}_j^{sp}\right) + C_l G_{ij} G_{kl} \hat{K}_{ik} \hat{K}_{lj} \right)^{-\frac{1}{2}}. \tag{17}$$

Note that in Eq (17), the identity matrix $\mathbb{I}$ is $(d+2)\times(d+2)$, and the notation $\hat{(\cdot)}$ on top of the matrices indicates that they are evaluated based on the governing equations using conservation variables. The term associated with $\hat{\kappa}^C_{DC}$ in Eq. (14) is a DC operator, where the DC stabilization parameter is given by

$$\hat{\kappa}^C_{DC} = C^C_{DC}\left(G_{ij}U^{h,T}_{,i}\tilde{A}_0^{-1}U^h_{,j}\right)^{-\frac{1}{2}}\left(\mathrm{Res}(Y^h)^T\tilde{A}_0^{-1}\mathrm{Res}(Y^h)\right)^{\frac{1}{2}}. \tag{18}$$

In the above, C^C_{DC} is a $O(1)$ positive constant ($C^C_{DC} = 0.5$ in this work), and $\tilde{A}_0^{-1}$ is the inverse of the zeroth Euler Jacobian of the transformation between the conservation and entropy variables (see Rajanna et al. [82, Appendix A]). Eq. (18) is an extension of the δ_{91} definition designed by Tezduyar and colleagues [58,83], where only the convective part of the full residual operator $\mathrm{Res}(Y^h)$ was employed. While the SUPG terms provide the necessary stability across a wide range of Reynolds numbers, the DC operator provides the necessary additional dissipation in the shock regions. Finally, the compressible-flow version of the weak-boundary-condition formulation is added to the weak form:

$$\begin{aligned}
&B^C(W^h,Y^h) - F^C(W^h)\\
&-\sum_b\int_{\Gamma^b\cap\Gamma_u^D} w^h\cdot\left(-p^h\mathbf{n}+\left(\lambda\hat{\mathbb{I}}\cdot u^h\right)\mathbf{n}+2\mu\boldsymbol{\varepsilon}(u^h)\mathbf{n}\right)d\Gamma\\
&-\sum_b\int_{\Gamma^b\cap\Gamma_u^D}\left(\rho^h w_p^h\mathbf{n}+\tilde{\gamma}\left(\left(\lambda\hat{\mathbb{I}}\cdot w^h\right)\mathbf{n}+2\mu\boldsymbol{\varepsilon}(w^h)\mathbf{n}\right)\right)\cdot\left(u^h-u_D\right)d\Gamma\\
&-\sum_b\int_{\Gamma^b\cap\Gamma_u^D} w^h\cdot\rho^h\left(u^h\cdot\mathbf{n}\right)\left(u^h-u_D\right)d\Gamma\\
&+\sum_b\int_{\Gamma^b\cap\Gamma_u^{D,-}} w^h\cdot\tau^C_\mu(u^h-u_D)\,d\Gamma+\sum_b\int_{\Gamma^b\cap\Gamma_u^D}\left(w^h\cdot\mathbf{n}\right)\tau^C_\lambda\left(\left(u^h-u_D\right)\cdot\mathbf{n}\right)d\Gamma\\
&-\sum_b\int_{\Gamma^b\cap\Gamma_u^D} w_T^h\,\kappa\hat{\mathbb{I}}T^h\cdot\mathbf{n}\,d\Gamma-\sum_b\int_{\Gamma^b\cap\Gamma_T^D}\tilde{\gamma}\,\kappa\hat{\mathbb{I}}w_T^h\cdot\mathbf{n}\left(T^h-T_D\right)d\Gamma\\
&-\sum_b\int_{\Gamma^b\cap\Gamma_T^{D,-}} w_T^h\rho^h c_v\left(u^h\cdot\mathbf{n}\right)\left(T^h-T_D\right)d\Gamma+\sum_b\int_{\Gamma^b\cap\Gamma_T^D} w_T^h\tau^C_\kappa(T^h-T_D)\,d\Gamma=0,
\end{aligned} \tag{19}$$

where λ is the second coefficient of viscosity ($\lambda=-2\mu/3$ based on Stokes' hypothesis), and τ^C, τ^C_μ, τ^C_λ, and τ^C_κ are the stabilization parameters of Nitsche's method. In this work, the compressible gas is assumed to be calorically perfect and the specific heats at constant volume and constant pressure can be defined as $c_v = R/(\gamma-1)$ and $c_p = \gamma R/(\gamma-1)$, respectively, where R is the ideal gas constant and γ is the heat capacity ratio. The pressure, density, and temperature are related through the ideal gas equation of state, $p=\rho RT$. In addition, thermal conductivity can be calculated by $\kappa = c_p\mu/Pr$, where Pr is the Prandtl number.

2.1.3. Stabilization parameter selection for weak boundary conditions

Eqs. (12) and (19) can be either the symmetric or non-symmetric form of Nitsche's method, depending on the value of $\tilde{\gamma}$ being selected as 1 or -1, respectively. The symmetric Nitsche method provides excellent accuracy and robustness when the stabilization parameters are properly estimated, which sometimes requires the solution of a local eigenvalue problem [84–88]. However, the arbitrarily intersected elements in immersed methods make obtaining accurate solutions to the eigenvalue problem challenging. An intuitive way to circumvent this difficulty is to use a uniform value that can be determined by a trial-and-error approach to simultaneously satisfy the requirements for numerical stability and conditioning of the system matrix [1,2]. Some delicately designed algorithms have been proposed for the local selection of stabilization parameters in intersected elements [86,89] and for preconditioning of the linear system [90–92]. However, even with the added algorithmic complexity, careful estimations of τ's do not necessarily result in improved accuracy in L^2 errors when compared with the uniform-valued stabilization parameters [53]. Therefore, in our IMGA CFD for incompressible flows, we set $\tau^{IC}_\mu = 10^3$ and $\tau^{IC}_\kappa = c\tau^{IC}_\mu$ to take advantage of this most straightforward but effective strategy.

Our previous numerical experiments suggest that, compared with incompressible flows, IMGA CFD for compressible flows is more sensitive to the oscillations around immersed boundaries caused by the uniform, large-value stabilization parameters. With this observation, the non-symmetric Nitsche type weak boundary condition

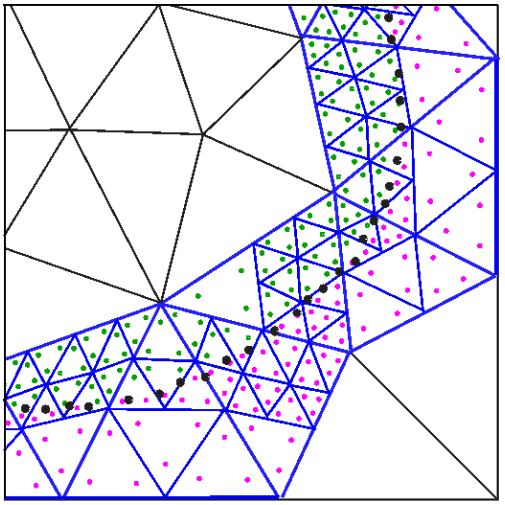


Fig. 2. Adaptive quadrature of mesh cells cut by the point cloud object boundary. Black points define the object's point cloud, green points are (inactive) quadrature points inside of the object, and the pink points are (active) quadrature points outside of the object. Two levels of subdivisions are shown here. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

operator [16,53,93–96] becomes an attractive alternative. The non-symmetric Nitsche formulation can be parameter-free [97–101], or with a stabilization added to reduce the oscillations near interfaces and to improve the L^2 accuracy [102–105]. Since the stabilization parameter in this case is not required to be larger than a specific lower-bound value to ensure stability, the estimation of the parameter becomes much simpler. In addition, the value of stabilization does not need to be very large, so it is less likely to overshadow the consistency term. Inspired by Wu et al. [27], we scale the stabilization parameters as $\tau^C = \mu\tau^C = 4\rho h^e/\Delta t_n$ and $\tau^C = c_v\tau^C$ for compressible-flow IMGA with non-symmetric Nitsche-type weak boundary conditions. Note that here $h^e = (n \cdot Gn)^{-1/2}$ is calculated from the full element, which greatly simplifies the evaluation in intersected elements.

2.2. Adaptive quadrature near point cloud surface

Accurate numerical integration over the physical domain Ω_{phys} with the presence of intersected elements is crucial for obtaining accurate simulation results in immersogeometric flow analysis. This is achieved in this paper via a subdivision-based adaptive quadrature rule in the context of the tetrahedral finite cell method [2]. Without changing the original background elements that support the basis functions, the method adaptively refines the quadrature distributions to create an aggregation of quadrature points in the vicinity of the immersed boundary. We refer to the elements used for generating adaptive quadrature rules as cells to avoid confusion with elements used for constructing basis functions. An intersected cell is recursively split into sub-cells until the sub-cell is completely inside Ω_{phys} or Ω_{fict}, or when the subdivision reaches a prescribed refinement level. Standard quadrature rules are applied to determine the set of quadrature points and weights within the sub-cells.

For clarity of visualization, Fig. 2 shows an illustration of the method in 2D based on adaptive sub-cells for triangles. The adaptive quadrature scheme needs two phases of point membership classification: the first classification is performed on the cell vertices to determine whether this cell is intersected, and the second classification is performed on the quadrature points to determine whether they should be active (inside Ω_{phys}) or inactive (inside Ω_{fict}). In the context of point cloud surface representation, a point membership classification method based on winding number will be presented in Section 3.3. For more details on the adaptive quadrature algorithms, see Xu et al. [2, Section 3.2].

3. Point cloud processing

Before we extend the IMGA for point cloud representation, we begin with a few notations for representing the point cloud and its associated differential quantities. A solid geometry is represented as a smooth manifold S

isometrically embedded in the Euclidean space $\mathbb{R}^d$, and the boundary surface of the solid geometry is represented as Γ (interacting with the domain Ω). A point cloud $P \subset \Gamma$ could be defined as a set of points sampled from the boundary surface, i.e. $P = \{p \in \Gamma\}$. We assume that Γ is 2-manifold (to ensure that S is solid) and piecewise smooth (for estimating the normals and curvatures). Further, P is sampled randomly from Γ, with the number of points denoted by $|P|$. For performing IMGA using point clouds, we present the following methods for (i) estimating the normal or the orientation of each point in the point cloud, (ii) enforcing Dirichlet boundary conditions, and (iii) performing point membership classification for any given domain point with respect to the boundary surface represented by point clouds for adaptive quadrature computations.

3.1. Normal computation

The raw point cloud P does not contain any information of the orientation (or the normal vector for each point in the point cloud). We estimate the normals from the raw (unoriented) point cloud by constructing a local surface proxy for each point and then estimating the normals for the surface proxy as illustrated in Fig. 3. For constructing this surface, we first identify the neighborhood of each point. For a given query point $p_i \in P$, where $i \in [1, |P|]$, we define the neighborhood N_i to be the k-nearest neighbors of the point in the set P. For fitting the local surface for each point p_i, we first transform all the neighboring points, $q \in N_i$, to a local coordinate system with the origin at p_i. Considering the points lie in $\mathbb{R}^3$, we now define a height function for $f(x, y)$:

$$f(x, y) = J_{B,n}(x, y) + O\left(\|(x, y)\|^{n+1}\right), \tag{20}$$

with

$$J_{B,n}(x, y) = \sum_{r=0}^{n} H_{B,r}(x, y), \qquad H_{B,r}(x, y) = \sum_{j=0}^{r} B_{r-j,j}\, x^{r-j} y^{j}. \tag{21}$$

Here, $B_{(\cdot,\cdot)}$ are different constants for a n-degree surface fit for the height function, $z = f(x, y)$. The neighboring points are fit with this height function by minimizing objective function of $\sum_{l=0}^{k} (f(x_l, y_l) - z_l)^2$. To simplify this problem, we write it in matrix representation, where the objective function is to minimize $\|MB - Z\|^2$. Here, M is the Vandermonde matrix $\left[1, x_l, y_l, x_l^2, \ldots, x_l y_l^{n-1}, y_l^n\right]_{l=1,\ldots,k}$ and B is the matrix with constants of the surface fit $\left[B_{0,0}, B_{0,1}, B_{1,0}, \ldots, B_{1,n-1}\right]$. We then find the solution to the linear system $MB = Z$, which is equivalent to fitting the best-fit surface. The guarantee for the existence of a unique solution and the accuracy of the solution to B as $O(h^{n-j+1})$ are proven in Cazals and Pouget [106]. Once the solution for B is obtained, the normal can be computed as follows:

$$n = \frac{\left(-B_{1,0}, -B_{0,1}, 1\right)}{\sqrt{1 + B_{1,0}^2 + B_{0,1}^2}}. \tag{22}$$

Note that when fitting a surface, we use a different and simpler approach by constructing a covariance matrix C, which can be defined as:

$$C = \begin{bmatrix} q_1 - \bar{p} & q_1 - \bar{p} \\ q_2 - \bar{p} & q_2 - \bar{p} \\ \vdots & \vdots \\ q_k - \bar{p} & q_k - \bar{p} \end{bmatrix}^T \begin{bmatrix} q_1 - \bar{p} & q_1 - \bar{p} \\ q_2 - \bar{p} & q_2 - \bar{p} \\ \vdots & \vdots \\ q_k - \bar{p} & q_k - \bar{p} \end{bmatrix} \tag{23}$$

Here, $\bar{p}$ represents the centroid of the neighboring points cluster N_i. We can estimate the normal vector for the plane at point p_i by computing the eigenvalues and eigenvectors of the covariance matrix C. The normal is defined as the eigenvector corresponding to the smallest eigenvalue [107]. Since the eigenvector corresponding to the smallest eigenvalue is also a principal component for the point cloud, this method is also known as principal component analysis (PCA)-based normal estimation.

Note that the method above does not guarantee that the calculated normals consistently point inward or outward with respect to the point cloud geometry. For a given point, we construct a minimum spanning tree consisting of the nearest neighbors and flip the calculated normal vector if its orientation is inconsistent with the neighbors [107].

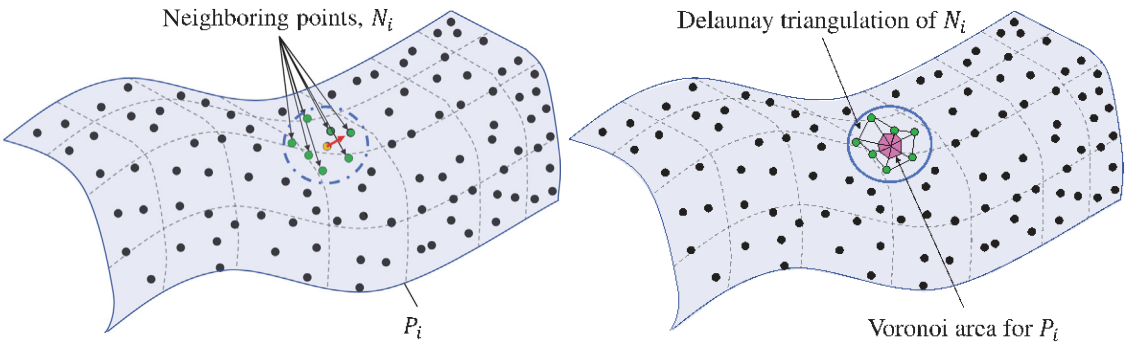


Fig. 3. Estimation of the normal and Jacobian determinant for each point on the surface. The image on the left shows the estimation of normals. First, we identify the k-nearest neighbors for each point. Then we fit a local surface using the neighbors, and the computed normal of the local surface is considered as the normal for that corresponding point. On the right, we show the estimation of the area (Jacobian) corresponding to each point. Using the k-nearest neighbors computed earlier, we first use Delaunay triangulation to obtain the local triangulation surrounding the query point. Then using the barycentric coordinates, we compute the Voronoi area corresponding to each point.

While the PCA-based normal estimation performs well (using plane fitting), for better accuracy, we can fit a quadratic surface to obtain better curvature and smoothness [106,108] (also commonly known as 2-jet fitting, or simply, jet fitting). In this paper, we compare both methods (PCA and jet fitting) to understand the advantages and disadvantages of both methods.

3.2. Point cloud surface integration rules

To weakly enforce the Dirichlet boundary conditions on Γ^D, we first discretize the boundary surface into elements. We then compute the weak-boundary-condition (or Nitsche) terms in Eq. (12) using the points of the point cloud and their corresponding Jacobian for each discrete element of the surface. For a point cloud represented using $\mathbb{P}$, we do not need any additional discretization, and hence we consider each point $p_i \in \mathbb{P}$ as a surface element[2] and its associated (single) quadrature point for performing the integration of the Nitsche terms over Γ^D. However, computing the Jacobian determinant for each quadrature point for performing the integration is not straightforward. While the surface area or the Jacobian determinant is not defined in a strict sense for a point-cloud-based representation, we define an effective surface area a_i for each point in the point cloud (i.e., point area or Voronoi area). This area can be associated with the geodesic Voronoi area of the point p_i on the boundary surface Γ. For some point cloud acquisition processes, this is already available from the sensors [109], and for others, this can be approximated as shown in Belkin et al. [110] and Barill et al. [111]. Similar to the approach used for computing the normals of the point cloud, we construct a local neighborhood for each point $p_i \in \mathbb{P}$ to obtain the geodesic Voronoi area a_i as shown in Fig. 3b.

The complete process can be detailed in three steps: (i) For each point p_i in the set of point cloud $\mathbb{P}$, we compute the neighborhood $\mathbb{N}_i$ to be the k-nearest neighbors. (ii) Using each set of neighborhood points $\mathbb{N}_i$, we perform Delaunay triangulation to result in a local surface $\mathbb{T}_i$. (iii) The dual graph of a Voronoi diagram $\mathbb{V}_i$ is the Delaunay triangulation $\mathbb{T}_i$. Therefore, computing the Voronoi area of p_i can be easily performed by taking the contribution area of each triangle in $\mathbb{T}_i$ to a_i. Once we obtain the area a_i corresponding to each quadrature point, we can integrate and obtain the weak-boundary-condition terms in Eq. (12).

3.3. Point membership classification

For IMGA, the attribution of a given point q in the domain Ω to the physical domain Ω_{phys} or fictitious domain Ω_{fict} is fundamental. In other words, we would have to determine if the point q is inside the solid S

[2] In this work, a surface element is represented by a point of the point cloud, which we sometimes refer to as a point cloud element.

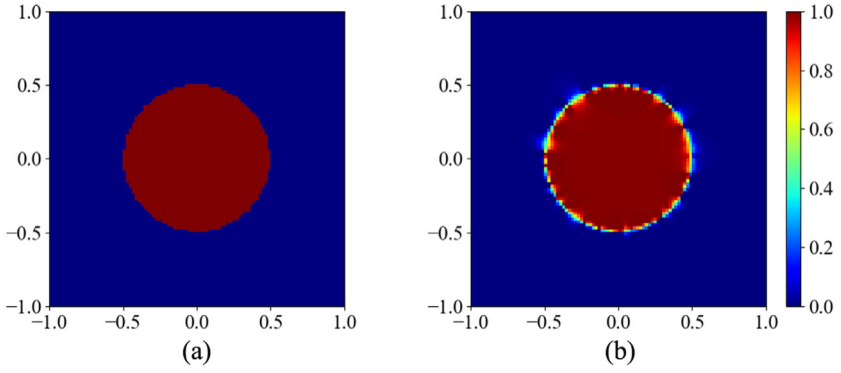


Fig. 4. Illustration of winding number for (a) a manifold representation of a circle and (b) a non-manifold representation of a circle. The circle is centered at (0.0, 0.0) and with a radius of 0.5. The non-manifold point cloud shown on the right is represented using 1000 points.

or outside it. Traditional B-rep CAD representations perform inside–outside testing by using ray-intersection-based methods. However, these approaches assume that the surface Γ representing the boundary of solid S is 2-manifold and watertight. When the CAD geometry is represented using a non-watertight surface representation or badly oriented normals, the ray-intersection-based methods fail to determine the membership correctly. Particularly, ray-intersection-based methods do not work for representations such as point clouds (which are non-manifold).

There has been recent work on defining inside–outside tests for arbitrary non-manifold geometries such as triangle soups and point clouds using generalized winding numbers [111,112]. The winding number $w(q, \Gamma) \in \mathbb{R}$ can be computed as:

$$w(q, \Gamma) = \frac{1}{4\pi} \int_{\Gamma} dA. \tag{24}$$

Here, A refers to the total solid angle subtended by Γ on q, dA refers to the differential solid angle subtended by each element of Γ on q. We then project the surface onto a unit sphere to obtain the surface integral and further simplify it for a discrete point cloud represented by P as follows:

$$w(q, \Gamma) = \frac{1}{4\pi} \int_{\Gamma} dA = \int_{\Gamma} \frac{(x - q) \cdot n}{4\pi \|x - q\|^3} dA = \sum_{i=1}^{|P|} a_i \frac{(p_i - q) \cdot n_i}{4\pi \|p_i - q\|^3} \tag{25}$$

p_i, n_i, and a_i refer to the point coordinates, the normal, and the geodesic Voronoi area, respectively, for the ith point in the point cloud P. Although the winding number is defined as a real number, for a 2-manifold, watertight representation of the geometry, the possible winding number value at any given point $q \in \mathbb{R}^3$ are positive integers. Further, if the surface does not have self-intersections, the winding number is in the binary set of $\{0, 1\}$. In other words, for any point $q \in \mathbb{R}^3$:

$$w(q, \Gamma) = \begin{cases} 1 & q \in \Omega_{phys} \\ 0 & q \notin \Omega_{phys}. \end{cases} \tag{26}$$

However, in the case of a non-manifold boundary representation such as a point cloud representation P, the winding number $w(q, \Gamma)$ is still zero for all $q \notin \Omega_{phys}$ with a continuous value between 0 and $s + 1$, where s is the number of self-intersections of the surface Γ. We illustrate how the winding number field for a non-manifold and manifold boundary representation differs in Fig. 4. Therefore, for attribution of a given point q to be inside or outside of Ω_{phys}, we threshold the winding number at 0.5, and any point with a smaller winding number is considered outside the physical domain, Ω_{phys}.

4. Validation

4.1. Validation of geometric quantities

Before performing IMGA using point clouds, we first validate the estimated geometric quantities to understand their accuracy in representing the geometry. We consider three different geometries represented by point clouds for

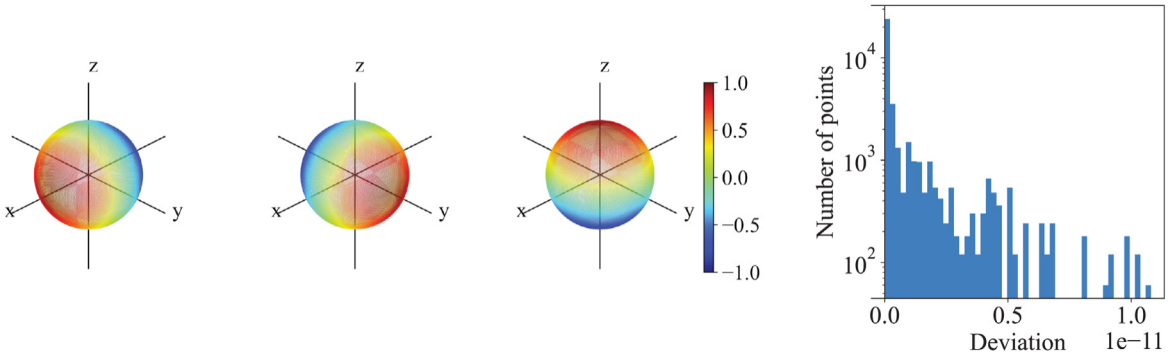


Fig. 5. The x, y, and z components of the normals estimated from 163,842 points sampled from an icosphere and the histogram of the deviation in the estimated normals (right).

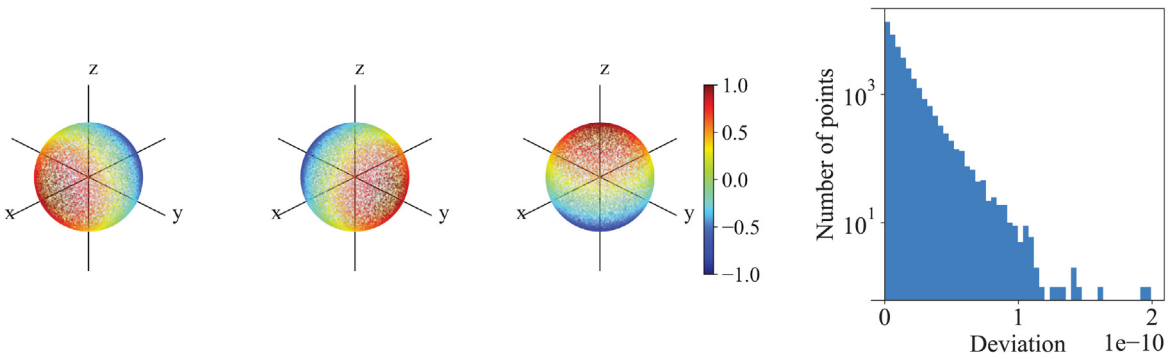


Fig. 6. The x, y, and z components of the normals estimated from 200,000 randomly sampled points and the histogram of the deviation in the estimated normals (right).

our study: (i) an icosahedron-based sampling of a sphere; (ii) a randomly sampled sphere; and (iii) a Fandisk model (a benchmark CAD geometry) containing sharp and smooth features [113].

4.1.1. Validation on icosphere and randomly sampled sphere

While our proposed method for estimating the geometric quantities required for IMGA works independent of the point sampling, for the first study, we consider the case of an icosphere. An icosphere is a set of points obtained by tessellating an icosahedron inscribed inside a sphere of unit radius [114]. Icosphere provides an ideal set of points sampled on a sphere such that there is a uniform distance between the points lying on the sphere. This geometric setup provides an ideal example for understanding the behavior of different parameters used to estimate geometric quantities. A more realistic geometry is a randomly sampled point cloud. We randomly generate points on a sphere by generating a set of random points and then projecting them to the sphere by dividing each point by its distance to the sphere center. Using this method, we can obtain an arbitrarily large number of points on the spherical surface.

For some methods used for acquiring the point cloud, the normal information is not available. Hence, we estimate the normals using the 2-jet fitting (mentioned above) for the point cloud. It takes 4.06 s to estimate normals for 163,842 points on the icosphere. Fig. 5 shows the visualization of the estimated normals for the icosphere. Here, we visualize the three components of the normal vector individually. We observe that each normal component is aligned with the corresponding primary axes. A similar visualization for a randomly sampled sphere with 200,000 points is shown in Fig. 6.

We also compute the deviation between the estimated and theoretical spherical normals. The theoretical normal of a given point on a unit sphere can be calculated by normalizing the vector from the sphere center to the point. The deviation is computed by subtracting the dot product between the estimated normal vector n_{esti} and the theoretical normal vector n_{theo} from unity ($1 - n_{theo} \cdot n_{esti}$). The maximum deviation in estimating the normals for 163,842

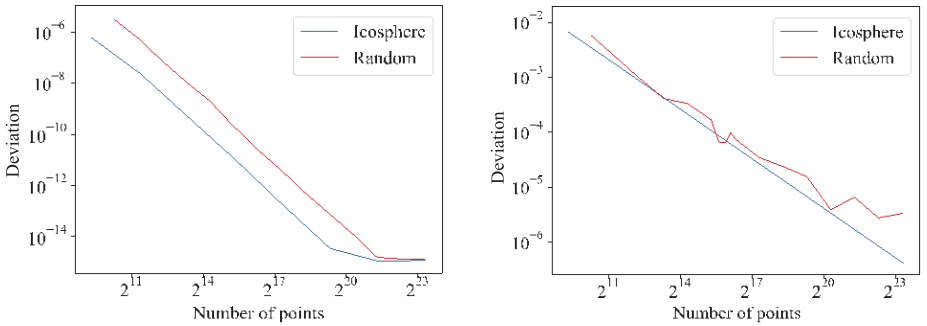


Fig. 7. Trends of deviation in normal estimation (left) and area estimation (right) with the increase in the number of points.

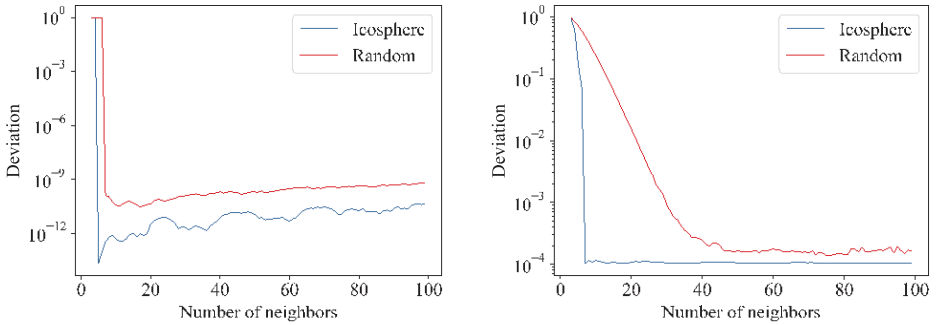


Fig. 8. Trends of deviation in normal estimation (left) and area estimation (right) with the increase in the number of neighboring points used for estimation.

points is 1.15×10^{-11}. A similar behavior can be observed for 200,000 randomly sampled in Fig. 6 with a slightly larger deviation of 2.00×10^{-10}.

Since the method for normal estimation relies on the local geometry, naturally, this deviation reduces with the number of points used to represent the geometry (hence capturing more intricate features of the geometry). In Fig. 7, we compare the mean deviation in normal estimation and deviation in total area estimation. For normal estimation, we use $1 - n_{theo} \cdot n_{esti}$ for estimating the deviation at each point and then use the mean deviation for comparison. However, we compare the sum of each point area for area estimation. The analytical value of the surface area of a sphere is used as the baseline for comparing the deviation. We see a linear trend for deviation of the geometric quantities with the increase in the number of points on the log scale. Also, the difference in mean deviation of normals obtained from the icosphere and randomly sampled sphere reduces with an increase in the number of points. The total area deviation for randomly sampled points is almost similar to the icosphere.

Another parameter affecting the estimation of geometric quantities is the number of neighbors used to estimate them. Fig. 8 shows the trend in the mean deviation of the estimated normals and the deviation in the estimated total surface area of the point cloud for different numbers of neighbors. The deviation is the minimum with an optimal number of neighbors (empirically close to 18–20 neighboring points, subject to change based on the number of points in the point cloud) and remains almost constant with a slight increase in the mean deviation of the estimated normals. This increase in deviation with an increase in the number of neighbors is because of adding more points that do not meaningfully contribute to the normal estimation. While a similar behavior is observed in normal estimation for a randomly sampled number of points, we also observe that random sampling increases the optimal number of neighbors to account for non-uniform distances between the points, specifically in point area estimation.

Another parameter to study is the degree of the surface used for fitting from the nearest neighbors. Recall that the linear plane fitting is equivalent to PCA-based fitting and is faster than the jet-fitting approach. While in terms of speed, the PCA-based plane fitting takes 0.79 s for normal estimation (compared to 4.06 s for 2-jet fitting), there is a difference of an order of magnitude in the estimation of the normals and consequently even in the area estimation. Fig. 9 shows the histogram of deviation in normals estimation for icosphere and randomly sampled points using

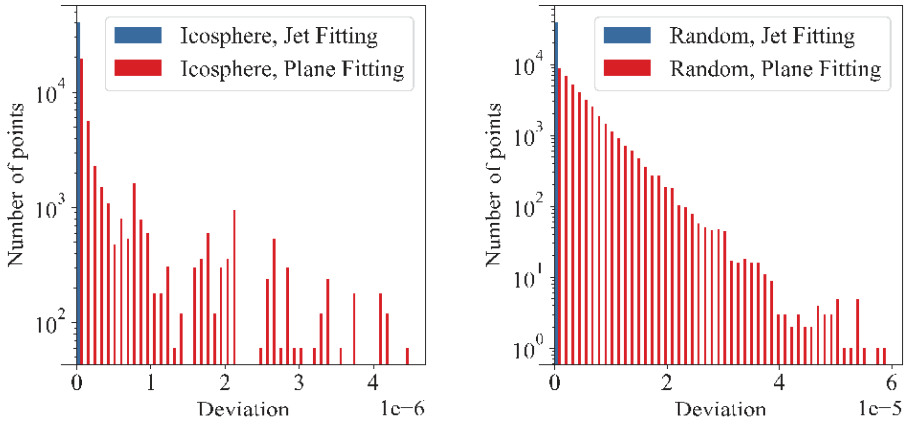


Fig. 9. Trends in the histogram of deviation in normals estimation from PCA-based estimation and 2-jet-based fitting for icosphere (left) and randomly sampled sphere (right).

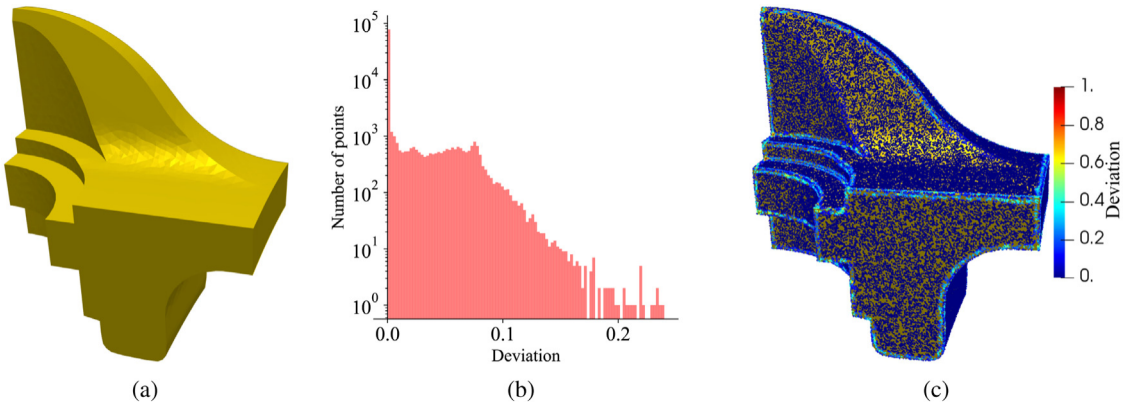


Fig. 10. Normal estimation on Fandisk model. (a) Surface representation of Fandisk model. (b) Variation of the maximum deviation in estimation of normals with the number of nearest neighbors used for fitting the surface. (c) Visualization of deviations of the normals at each point on the surface.

both PCA and 2-jet fitting. While there is a significant difference in the computational time and the deviation, the maximum deviation in the estimation is still less than 10^{-4}, and hence is still a viable method for estimating the quantities for IMGA. Both methods, with appropriately chosen neighbors and point cloud density, have very little deviation for the sphere.

4.1.2. Validation of normals estimation on a point cloud sampled from the Fandisk model

As mentioned above, the normal estimation depends on the nearest neighbors for a given point. This means there could be significant deviation when the geometry has sharp edges or very fine features. To test this, we use a benchmark CAD model (Fandisk model, shown in Fig. 10a). The Fandisk model consists of smooth and sharp features, thus enabling us to test the robustness of the normal and area estimation methods. While the experiments elucidated above can also be performed for the Fandisk model, we show just a few key results that provide insights for brevity.

We use the normals computed from the triangle faces of the benchmark CAD model to set a baseline for the normal vectors. This baseline preserves sharp features since the normals are completely different for adjacent triangles. However, such sharp features cannot be replicated using point clouds due to a lack of connectivity information. In Fig. 10b, we show the histogram of deviations in normal estimation. While most points have very little deviations, about 20% of the total points have some significant deviation. We visualize the deviations of the normals at each point on the surface in Fig. 10c. While most of the points at the center of each face have minimal

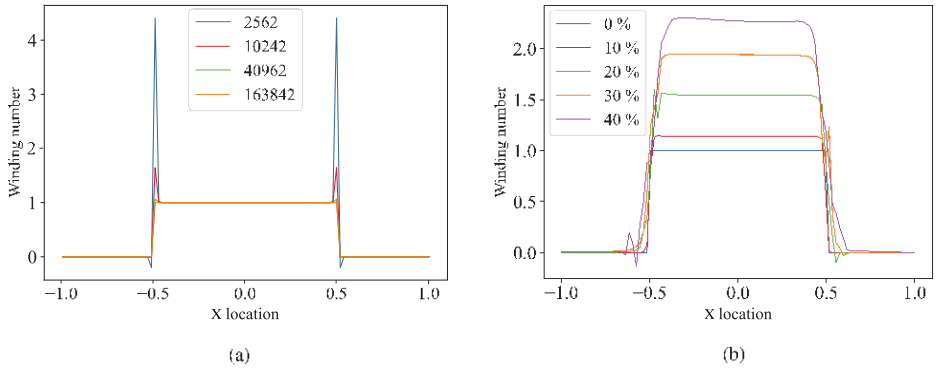


Fig. 11. Line-cuts of winding number along the x axis for a sphere with (a) a different number of points in the point cloud, and (b) with an increase in the noise for the sampled point cloud.

deviation, all the points near the edges show significant normal deviation, which is expected. Note that the deviation of 0.30 (the maximum deviation for a point) refers to a deviation of 45° between the two normal vectors, the average direction between perpendicular planes.

4.1.3. Validation of winding number computation

Since the winding number is a scalar field, we analyze the behavior by plotting the winding number at one point along the x axis. The ideal winding number for a sphere looks similar to the line-cut representing 163,842 points as shown in Fig. 11(a). The winding number at $x < -0.5$ and $x > 0.5$ for a sphere centered around origin with a diameter 1.0 is 0, and 1 for $-0.5 \leq x \leq 0.5$. When fewer points are sampled, the dipole moments (see Barill et al. [111] for a detailed discussion on dipole moments) do not cancel in the local region around the boundary, which leads to local self-intersections. However, we can resolve a watertight geometry without local self-intersections by using 0.5 as a threshold to classify a given point to be within the geometry.

4.1.4. Robustness to noise

Often point clouds acquired from actual sensors have significant noise associated with them. While understanding the efficacy of noise removal approaches is not in the scope of this paper, we still would like to ensure that the proposed method is robust to noise. To understand the effect of noise, we add a specified Gaussian noise at each point. Fig. 12 shows the increase in the deviation with an increase in the % noise added to the original point cloud. To see how this affects the geometric reconstruction, we obtain line cuts of the winding number as earlier in Fig. 11(b). As would be expected, adding more noise makes the reconstruction non-manifold. Fig. 13 shows the renderings of the reconstructed surface mesh of the sphere. With the increase in noise, many dimples are created on the surface that could change the aerodynamics of the shape; nevertheless, the shapes generated are all 2-manifold and watertight.

4.2. Incompressible flow around a point cloud sphere

We perform mesh convergence and validation studies using icosphere point clouds representing spheres. We create an icosphere by subdividing the faces of an icosahedron and projecting the resulting nodes onto the enclosing sphere. Though the icosphere does not achieve an ideal pattern of equilateral triangles, the generated geometry is equivalent to a well-defined 2-manifold surface mesh. From this tessellated icosphere, points are spawned at the center of each triangle element and collected into a point cloud. We begin with a description of the problem setup and background meshes used for the convergence studies and validations.

4.2.1. Problem setup

We follow the flow around a sphere study undertaken by Xu et al. [2] with the same domain dimensions and very similar background meshes. Note that the problem is non-dimensional, and the icosphere has a diameter of

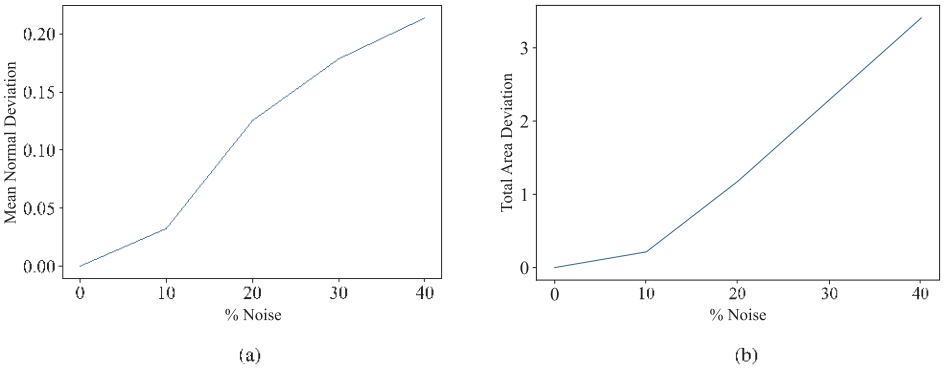


Fig. 12. The degradation in the normal estimation and Voronoi area estimation with increasing variance of noise.

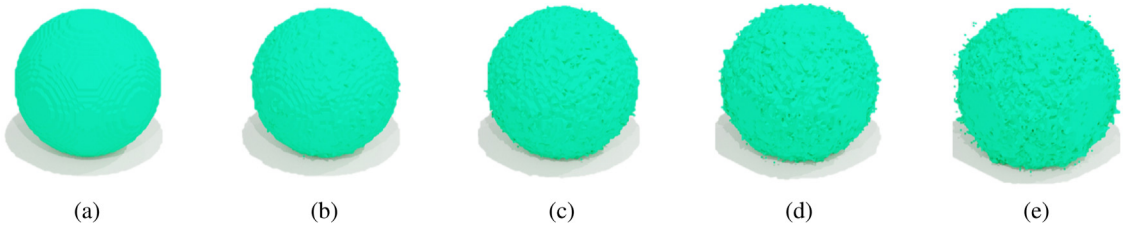


Fig. 13. Rendering of spheres obtained from reconstructed using winding number from (a) 0%, (b) 10%, (c) 20%, (d) 30%, and (e) 40% Gaussian noise in the point cloud.

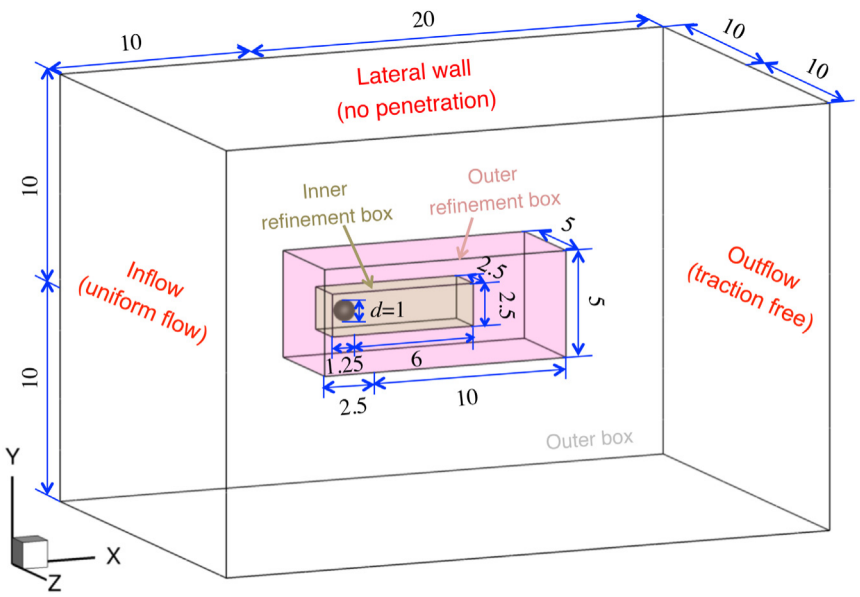


Fig. 14. Computational domain for the icosphere flow validations. The outer frame bounds the fluid domain, enclosing an outer refinement region, which itself encloses an inner refinement region. The icosphere point cloud resides at the origin.

one. The computational domain, boundary conditions, and immersed icosphere are shown in Fig. 14. A uniform inflow velocity of one in the x-direction is set at the inlet, and the no-penetration condition is set at the lateral walls, both enforced strongly. The outflow boundary is traction free. The no-slip condition is enforced weakly on the sphere point cloud using the methods described in the previous sections. The density of the fluid is set to one,

Table 1
Element sizes in tetrahedral meshes for simulating flow around a sphere.

Mesh	No. of elements	Cut element	Inner region	Outer region	Base domain
IM0	304,330	0.020	0.20	0.8/√2	1.2
IM1	1,833,434	0.010	0.10	0.4/√2	1.0
IM2	9,041,302	0.005	0.05	0.2/√2	0.8
BM2 [2]	8,519,435	0.005	0.05	0.2/√2	0.8

Table 2
Point cloud convergence for flow around a sphere with adaptive quadrature level 2 (AQ2) on background mesh IM0. Blank values indicate divergent solutions as a result of insufficient point cloud density.

Points	Re = 100		Re = 300		Re = 3700	
	C_D	L/d	$\overline{C}_D$	St	$\overline{C}_D$	St
642	3.531	1.627				
2562	1.404	1.092	1.104	0.105		
10,242	1.087	0.982	0.690	0.125	0.643	0.062
40,962	1.094	0.973	0.676	0.144	0.419	0.093
163,842	1.094	0.973	0.676	0.144	0.419	0.093
655,362	1.094	0.973	0.676	0.144	0.419	0.093

and the Reynolds number ($Re = \mu^{-1}$) is defined as the inverse of the viscosity. Non-dimensional mesh settings used for this problem are summarized in Table 1. IM0, IM1, and IM2 are immersogeometric tetrahedral meshes of increasing mesh density. Note that IM2 has a comparable mesh resolution to the boundary-fitted mesh BM2 used in Xu et al. [2], which we use as a reference. We consider $Re = 100$, 300 and 3700 in this study, and the time step sizes in the simulations are set to 1.0×10^{-2}, 1.0×10^{-3} and 1.0×10^{-3}, respectively.

As point clouds lack continuous surfaces to recursively refine surface quadrature points, a certain spatial density of point cloud relative to the background mesh's cut element size is required to prevent flow leakage. Hsu et al. [7] recommends a maximum ratio of 2 between the sizes of a tessellated surface element and the volume element cut by it. Before conducting validation studies, we derive an analogous ratio for point clouds with an icosphere geometry. This is found by converging flow quantities with increasingly dense point clouds on a fixed background mesh, specifically IM0.

4.2.2. Point cloud convergence with fixed mesh

We simulate flow over six different densities of point clouds on IM0 for $Re = 100$, 300, and 3700. The coarsest geometry has an average point spacing equal to about five times the cut element size, while the finest geometry has a point spacing of approximately 1/6 of IM0's cut element size. This tabulation aims to find the point cloud density at which our measured flow heuristics gain independence from point spacing. In Table 2, drag coefficient (C_D) and recirculation length (L/d) are listed for the steady-state $Re = 100$ solutions, whereas we report time-averaged drag coefficient ($\overline{C}_D$) and Strouhal number (St) for vortex shedding cases of the two other Reynolds numbers. 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. The drag force is evaluated using the variationally consistent conservative definition of traction [2,115]. The recirculation bubble length 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. The Strouhal number is computed as $St = f d/U$, where f is the frequency of vortex shedding.

As Re increases, we observe an increase in the minimum point cloud density required to achieve stable flow solutions. As indicated by the blank cells in Table 2, the coarsest point clouds exhibit divergent solutions at higher Reynolds numbers, even with small time step sizes. As Re increases, it becomes more necessary that each cut element encloses at least one icosphere point so that each cut element adequately "feels" the forcing contribution from the immersed point cloud to prevent flow leakage.

Table 3
Convergence study for flow around a sphere at Re = 100 with different mesh densities and adaptive quadrature levels.

Mesh	C_D			L/d		
	IM0	IM1	IM2	IM0	IM1	IM2
AQ0	0.927	0.960	0.988	1.010	0.848	0.857
AQ1	1.078	1.080	1.092	0.997	0.846	0.859
AQ2	1.094	1.091	1.092	0.982	0.853	0.858
AQ3	1.094	1.091	1.092	0.973	0.854	0.858
BM2 [2]			1.093			0.857

Table 4
Convergence study for flow around a sphere at Re = 300 with different mesh densities and adaptive quadrature levels.

Mesh	$\overline{C}_D$			St		
	IM0	IM1	IM2	IM0	IM1	IM2
AQ0	0.602	0.620	0.649	0.139	0.144	0.139
AQ1	0.675	0.643	0.657	0.144	0.139	0.136
AQ2	0.677	0.658	0.662	0.142	0.136	0.135
AQ3	0.676	0.659	0.662	0.144	0.135	0.135
BM2 [2]			0.661			0.135

Table 5
Convergence study for flow around a sphere at Re = 3700 with different mesh densities and adaptive quadrature levels. BF denotes the boundary-fitted solution from Xu et al. [2].

Mesh	$\overline{C}_D$			St		
	IM0	IM1	IM2	IM0	IM1	IM2
AQ0	0.562	0.468	0.399	0.083	0.164	0.219
AQ1	0.516	0.407	0.395	0.089	0.163	0.218
AQ2	0.419	0.402	0.394	0.093	0.160	0.218
BF [2]			0.393			0.217

The results in Table 2 show that convergence is consistently achieved when using a point cloud of 40,962 points. This density translates to an average point spacing valued at about 2/3 of IM0's cut element size. This agrees with the spatial density recommendations of Xu et al. [2], as this point cloud is the coarsest permutation that maintains a point cloud spacing smaller than IM0's cut element size. In subsequent simulations, we prescribe a point cloud density that ensures at least one point per cut element.

4.2.3. Mesh convergence and flow validation

Here, we perform convergence studies on mesh density and adaptive quadrature level and validate the flow quantities of interest at Re = 100, 300, and 3700 against those reported in Xu et al. [2]. The adaptive quadrature refinement cases are denoted as AQ followed by the level number. One icosphere cloud, with its average point spacing less than the cut element size of the finest mesh IM2, is used in all simulations in this section. The same flow qualities as in Section 4.2.2 are evaluated and the results are shown in Tables 3–5. For all cases, the convergence under adaptive quadrature refinement is clearly shown.

For Re = 100 and 300, data obtained above and including IM1 and AQ2 refinement levels are in excellent agreement with the reference values (BM2) reported in Xu et al. [2]. The results clearly show mesh convergence and demonstrate the importance of capturing the point cloud geometry in cut cells when integrating the background elements. For Re = 3700, the results show that IM2 is essential to obtaining accurate solutions. For this configuration and flow condition, there occurs a laminar flow separation near the equator of the sphere and a transition to turbulence in the wake of the object [116]. Compared to the drag coefficient, the Strouhal number

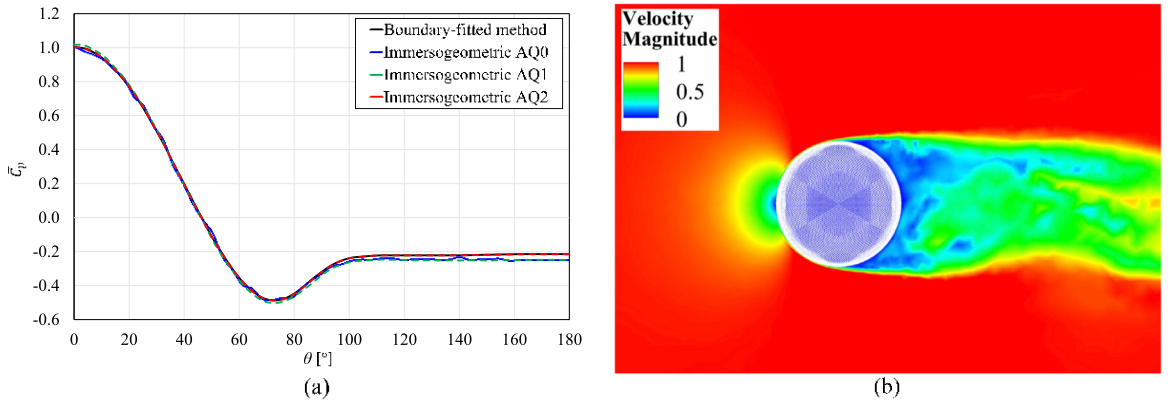


Fig. 15. (a) Time-averaged pressure coefficient along the upper crown line of the sphere obtained using IM2 with different AQ levels. Immersogeometric results are compared against the reference boundary-fitted result [2]. (b) Velocity magnitude contour on a planar cut around the sphere.

appears particularly sensitive to mesh density, unable to reach a closer value until IM2. In Fig. 15, the time-averaged pressure coefficient ($\overline{C}_p$) along the sphere's upper crown line is plotted for the IM2 case. $\theta = 0^\circ$ corresponds to the stagnation point and $\theta = 180^\circ$ is the trailing point of the sphere. AQ0 shows oscillatory behavior in C_p, signifying that further refined quadrature points are required in cut elements. AQ1 remedies the oscillation but largely follows the same pressure coefficient curve as AQ0. Finally, AQ2 produces a result that is in excellent agreement with the boundary-fitted reference [2]. It should be noted that the refinement levels, solution accuracy, and convergence behavior presented in this section are in full agreement with those in Xu et al. [2].

4.3. Buoyancy-driven flow

After the validation of isothermal fluid flow, we validate the thermal IMGA by simulating buoyancy-driven flows. Following the problem of natural convection in an enclosure with a heated sphere investigated by Yoon et al. [117], a static spherical boundary is situated within a sealed cubical enclosure filled with air, as shown in Fig. 16. Buoyant flow is driven by a temperature differential between the sphere and the enclosure walls. This case is chosen as it requires simulation of heat transfer involving immersed Dirichlet boundaries and accurate resolution and application of buoyant forces on fluids. The sphere's boundary, with a non-dimensional radius of $R = 0.2$, is immersed within the cubical simulation mesh. This cube has non-dimensional edge lengths of 1, with the coordinate system at the domain's center and coordinate axes parallel to the domain's edges. The gravity acts in the $-z$ direction. All boundaries are treated as no-slip walls. A high temperature $T_h = 1$ is applied on the sphere, while a low temperature $T_c = 0$ is applied on the cube domain surfaces. The normalized temperature can be defined as $\Theta = (T - T_c)/(T_h - T_c)$.

We simulate two cases at Rayleigh number $Ra = 1000$ with $\delta = 0$ and $\delta = 0.25$. Rayleigh number is defined as $Ra = g\beta L^3(T_h - T_c)/(\nu\alpha)$, where $g = 1$ is the gravitational acceleration, $\beta = 1$ is the thermal expansion coefficient, $L = 1$ is the length of the enclosure, ν is the kinematic viscosity, and α is the thermal diffusivity. δ refers to the z coordinate of the sphere where $x = 0$ and $y = 0$. For both cases, the Prandtl number, $Pr = \nu/\alpha$, of air is used ($Pr = 0.7$). With the aforementioned problem setup, one can adjust the value of ν or α to achieve the desired Ra number. Three meshes are tested with each of the two δ cases, with non-dimensional element sizes listed in Table 6. The mesh sizes on the cube domain surfaces and near the immersed sphere are the "base elements" and "cut elements" sizes, respectively, and a smooth transition is achieved in the space between the boundaries. A time step size of 5.0×10^{-3} is used in all simulations.

For each of the six simulation case permutations, the Nusselt number (Nu) is calculated at the top surface of the enclosure, referring to the wall perpendicular to the z-axis located at $z = 0.5$. The Nusselt number can be calculated by $Nu = \nabla\Theta \cdot n$, where n is the unit normal vector to the wall. In Fig. 17, we plot this value along a line between $(0.0, 0.0, 0.5)$ and $(0.5, 0.0, 0.5)$ and compare our results with the reference values reported by Yoon et al. [117].

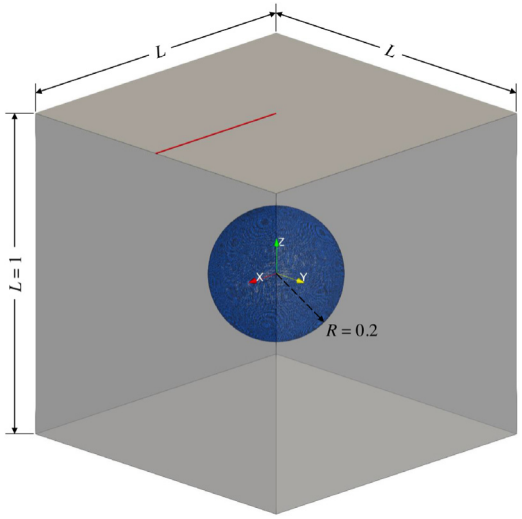


Fig. 16. Computational domain for the problem of natural convection in a sealed cube enclosure with a heated sphere. The red line indicates where Nusselt number is plotted to compare against reference data.

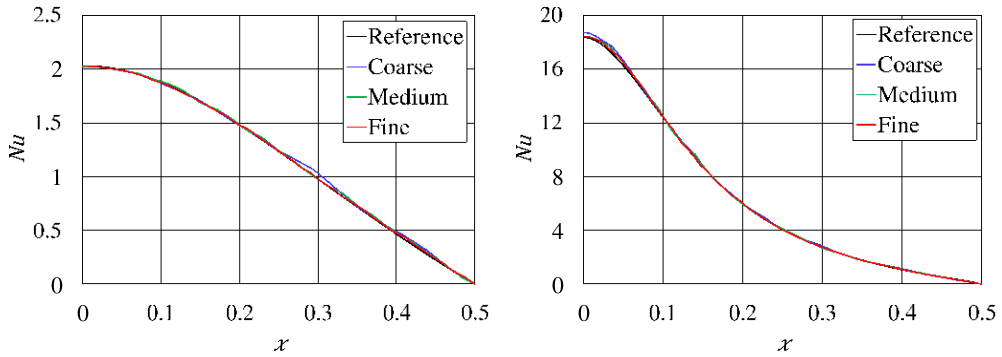


Fig. 17. Nusselt number along the centerline of the top surface of the sealed enclosure for δ = 0 (left) and for δ = 0.25 (right).

Table 6
Non-dimensional element size settings of each simulation mesh for buoyancy-driven flow validation.

Mesh	Cut elements	Base elements
Coarse	0.020	0.040
Medium	0.010	0.020
Fine	0.005	0.010

In the case of δ = 0, for the most part, even the coarse mesh is sufficient in reproducing the temperature gradient curve. All meshes accurately simulate boundary values relative to the reference solution. Unsurprisingly, the coarse mesh produces some notable solution oscillations. Grid independence is apparent after the medium mesh, where the medium and fine meshes improve the solution accuracy and track the reference data to a strong degree. Similarly, all three meshes with δ = 0.25 produce comparable solutions near the edge of the top surface. The coarse mesh solution noticeably deviates from the reference solution at the center of the top surface. Again, the coarse mesh exhibits some oscillations throughout the solution, whereas the medium and fine meshes are successful in reproducing the reference curve.

4.4. Compressible flow over a torpedo-shaped body

In this section, we simulate laminar flow around a torpedo-shaped body represented by the point cloud data in both the subsonic and supersonic regimes. The dimension of the geometry and the computational domain can be found in Wang et al. [8]. The point cloud representation of the torpedo-shaped body is shown in Fig. 18a. To perform the simulation at subsonic speed of $M = 0.8$, the inflow quantities are set to $p = 1.1161$, $\|u\| = 1.0$, and $T = 3.8713 \times 10^{-3}$. The dynamic viscosity μ is set to a constant value of 0.01. For the supersonic case of $M = 2.0$, the inflow quantities are set to $p = 0.1786$, $\|u\| = 1.0$, and $T = 6.1941 \times 10^{-4}$. The dynamic viscosity is determined from Sutherland's law: $\mu = (C_1 T^2)/(T + S)$, where $S = 1.406 \times 10^{-4}$ and $C_1 = 0.906$. For both the subsonic and supersonic cases, no-penetration and zero-heat flux boundary conditions are enforced on all the lateral boundaries of the domain. The outlet boundary is set to have the same total traction as the inlet. On the point cloud object, the velocity is set to zero, and the temperature is set as the stagnation temperature determined by $T_D = (1 + 0.5(\gamma - 1)M^2)T$; both conditions are enforced weakly. The heat capacity ratio γ is 1.4, the ideal gas constant $R = 288.293$, and the Prandtl number Pr is 0.72. Note that all quantities are dimensionless.

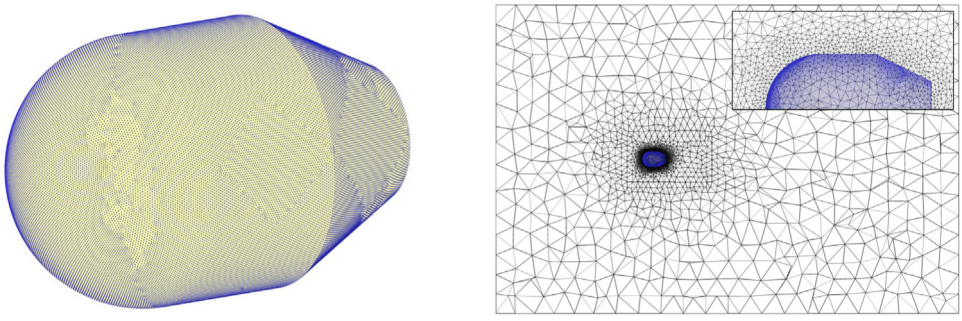
We perform a mesh refinement study to assess the performance of the compressible-flow point-cloud IMGA formulation. Simulations are carried out on three meshes named IM0, IM1, and IM2, from coarser mesh to finer mesh, respectively (see Wang et al. [8] for the notation and statistics of these meshes). To illustrate the mesh design, we plot a planar cut through the center of the coarsest mesh IM0 in Fig. 18b. The simulations are performed using a time step size of 0.005 until a steady state is reached. Figs. 19a and 20a show the Mach number contour plots computed on IM2 for the subsonic and supersonic cases, respectively. The pressure coefficient distributions along the upper crown line of the torpedo-shaped body as a function of the streamwise coordinate for different meshes for the subsonic and supersonic cases are plotted in Figs. 19b and 20b, respectively. The pressure coefficient results are also compared with the boundary-fitted computations using a comparable mesh resolution for both cases. The results demonstrate that IM1 and IM2 meshes produce converged solutions and show excellent agreement with the boundary-fitted computations. They are also in excellent agreement with those reported in Xu et al. [16]. Note that a two-level recursive adaptive quadrature rule is employed to faithfully capture the immersed geometry and produce an essentially converged solution.

5. Flow over an industrial vehicle

Extending upon the practical application of IMGA to industrial geometry in B-reps and tessellated formats [2,7,8,16], we demonstrate the compatibility of this method with remarkably detailed geometry in a point cloud format. The workflow for computational analysis of an industrial product design typically begins with a B-rep CAD model representing an industrial object to be manufactured. This B-rep model primarily exists for purposes other than CFD, meaning that the geometry is often unsuitable for boundary-fitted analysis. Significant effort is invested to either "clean up" the existing B-rep or create a surrogate version with limited complexity and airtight topology. Only then may boundary-fitted mesh generation begin, accompanied by its own set of difficulties. Hsu et al. [7] and Wang et al. [8] subverted these computational analysis roadblocks using IMGA on an agricultural tractor and a tractor-trailer truck in trimmed NURBS and analytic surface formats. Compared to the tractor and truck, the geometry of this demonstration is unique in that it is overwhelmingly composed of finite-thickness shells, one type of topology targeted by CAD cleanup operations.

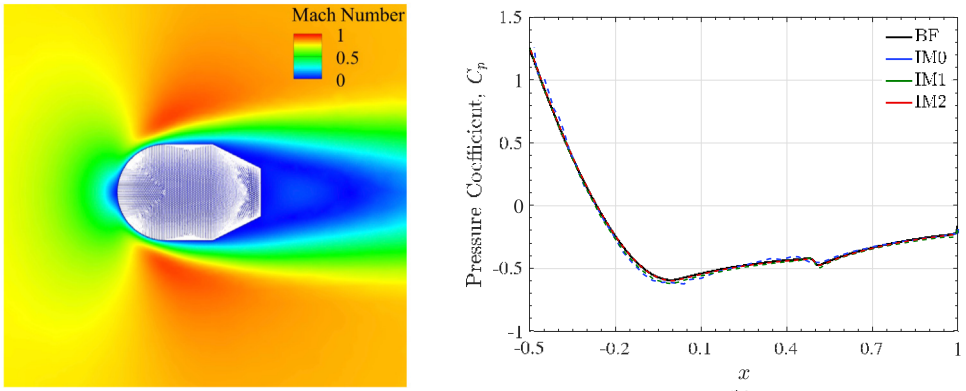
The geometry in this demonstration represents a medium-sized construction vehicle: a John Deere 544K Wheel Loader. Given that this is a low-speed utility vehicle, there is little concern for aerodynamic forces experienced by the vehicle, which is contrary to common vehicular CFD analysis, like that of a tractor-trailer truck. For this type of vehicle in an actual product analysis application, fluid simulation seeks to forecast how components, especially the engine and electronics, remain within thermal constraints throughout their continuous operation. Physics-coupled simulations are particularly useful to predict temperatures that the key components of the vehicle experience. Regarding the 544K demonstration that we subsequently exhibit, fluid flow similarly transpires within this low-speed incompressible flow regime, though we purposely avoid boundary conditions expressing resemblance to conditions within the operating range of the real-life vehicle. In this section, we demonstrate the utility of point-cloud IMGA applied to an industrial analysis workflow.

The generality of the point cloud format affords flexibility regarding the types of CAD formats that can be analyzed. The as-manufactured analysis advantage of point-cloud IMGA is fully realized with the virtualization



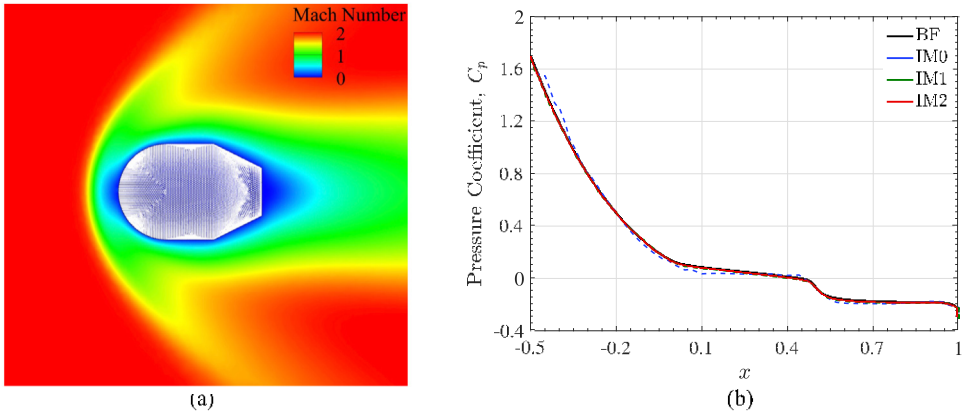
(a) (b)

Fig. 18. (a) Point cloud representation of the torpedo-shaped body. (b) IM0 mesh with a zoom on the region near the point cloud representation of a torpedo-shaped body.



(a) (b)

Fig. 19. (a) Mach number contours for the subsonic flow (M = 0.8) around a torpedo-shaped body. (b) Pressure coefficient along the upper crown line of the torpedo-shaped body as a function of the streamwise coordinate.



(a) (b)

Fig. 20. (a) Mach number contours for the supersonic flow (M = 2.0) around a torpedo-shaped body. (b) Pressure coefficient along the upper crown line of the torpedo-shaped body as a function of the streamwise coordinate.

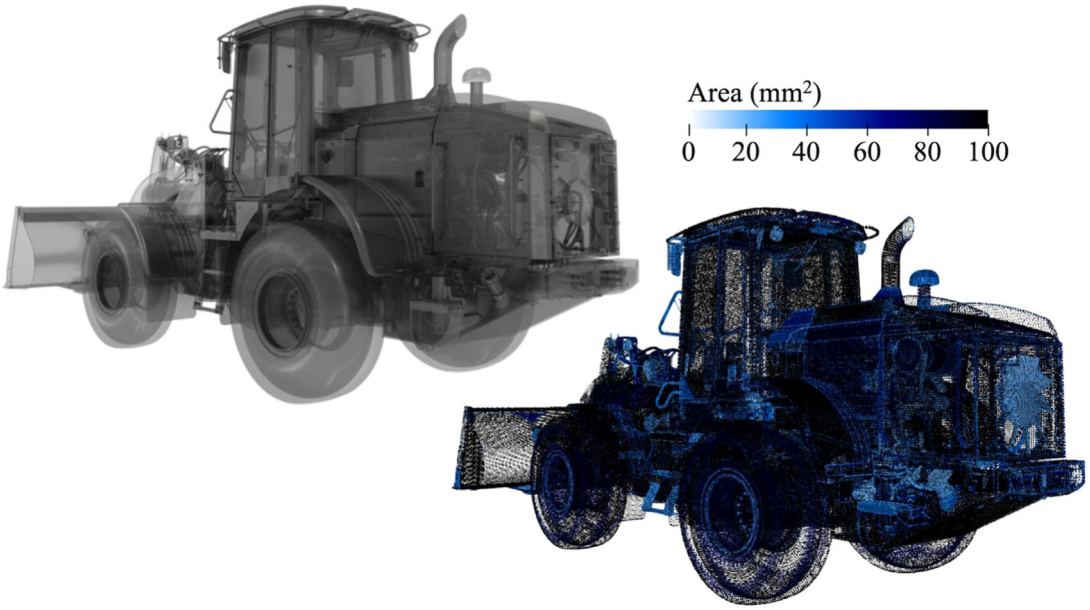


Fig. 21. Translation of John Deere 544K construction vehicle's surface representation to point cloud format. (Left) Translucent surfaces exposing complex details within the vehicle. (Right) Sampling of surface representation into points, each point colored by its scalar area. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of physical objects, but true as-designed analysis still presents worthwhile advantages over conventional analysis of simplified geometries. Lacking a practical 3D scan, we begin with the comprehensive CAD assembly used by Deere &Company for design and manufacturing purposes. This B-rep is ill-suited for boundary-fitted analysis as it is non-manifold with gaps, intersections, and collocations. Instead of converting it to watertight NURBS, analytic, or tessellated surfaces, the assembly is sampled into points with normal vectors and area scalars as in Fig. 21 in a matter of minutes. A significant amount of manual labor is saved by circumventing the geometry simplification process typically required for boundary-fitted analysis of such complex geometry. All impermeable features within the vehicle are retained; in the absence of support for sub-scale porosity, this preliminary demonstration omits porous screens and radiators. It should be noted that all parameters utilized to simulate the 544K are unrepresentative of reality, and presented simulation results make no claims regarding the performance of the actual product as it exists physically. This exercise demonstrates point-cloud IMGA's compatibility with multiphysics simulation using complex point clouds.

5.1. Geometric pre-processing accuracy

The vehicle's point cloud is discretized by sampling its B-rep surface model. We begin with a 0.025 m spacing between points, which returns a total of 4,041,875 point cloud elements. This operation filters small geometric features, automating an otherwise intensive manual task. Similar to the icosphere, sampling of the surface geometry provides normal vectors and area scalars as analytical values to compare the quality of pre-processing approximations. Point spacing is decreased to produce three more point clouds with greater element populations of 8,135,975 points, 11,855,474 points, and 27,937,410 points. Normals estimation is performed using 4 neighboring points, and a box plot of each point cloud's collection of normal vector deviations is displayed in Fig. 22.

Here, the performance of jet-fitting resembles the normal estimation on the Fandisk. A deviation of two indicates flipped normals, suggesting that the point cloud with 4 million elements contains an excess of incorrectly aligned normal vectors. Cazals and Pouget [106] point out that the ambiguity of normal vectors on flat surfaces is a weakness of normal vector calculation in its current form. Using a larger number of neighboring points for the normal estimation increases the error, which we theorize is due to the ubiquity of thin plates in this geometry. As the number of neighbors increases for a point on a thin plate, points on one side of the plate are grouped with

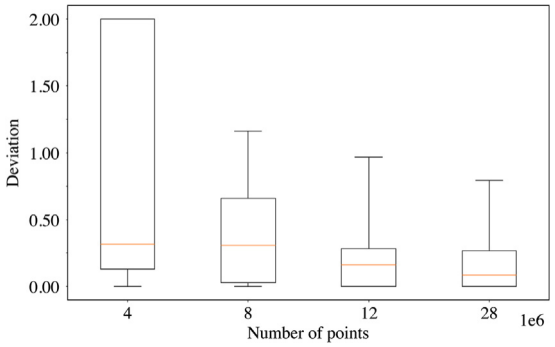


Fig. 22. Box plot of the deviation of calculated normals against analytical normal vectors of the John Deere 544K. The same procedure is performed on four point clouds of the 544K with varying point density.

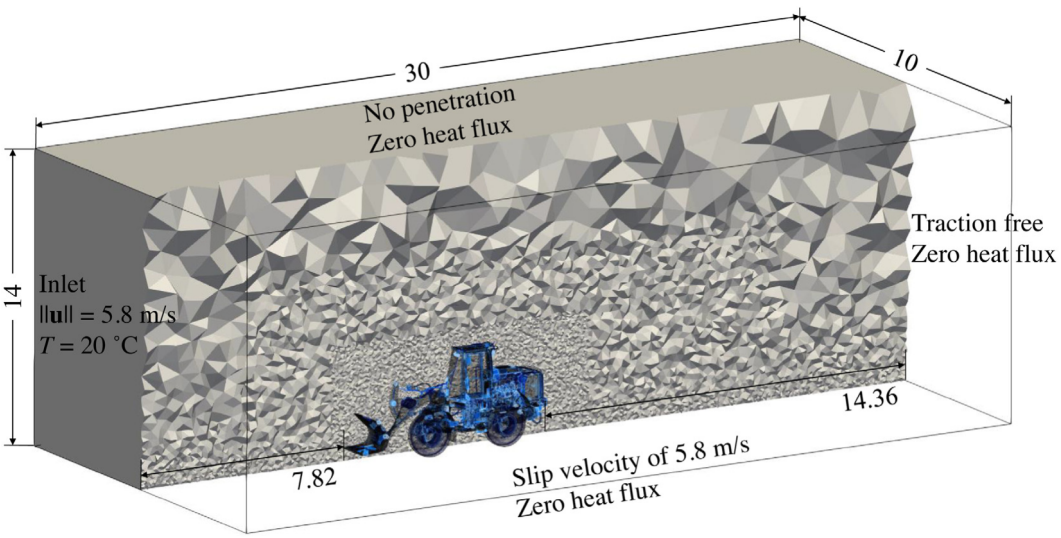


Fig. 23. Mesh refinement, domain dimensions (in meters), and boundary conditions for the flow analysis of John Deere 544K driving forwards.

points on the other side, malevolently affecting calculations for points on both sides. As the density of the point cloud increases, a point on one side of a plate is more likely to be in a neighborhood exclusively with points also on the same side of the plate. As will be discussed later, we find that the two coarsest point clouds are too sparse to produce reliable normals for stable flow simulations of John Deere 544K using point-cloud IMGA. The vehicle point cloud with 12 million elements, which has an average distance of 0.00625 m between points, is therefore used for the subsequent studies.

5.2. Problem setup

Fig. 23 visualizes a mid-plane clip of the domain mesh along with the boundary condition setup. A uniform flow velocity of 5.8 m/s is applied at the inlet, and the outlet is traction free. A slip velocity of 5.8 m/s is applied on the stationary ground boundary to produce the ground effect of a forward-moving vehicle in a stationary fluid domain. No-penetration conditions are applied on the top and lateral walls. The wheels of the construction vehicle are modeled as rotating no-slip walls, with the angular velocity matching the tire surface velocity with the floor's slip velocity at their common interface. The fan located rearward of the vehicle is modeled similarly and rotates at 600 RPM to draw air out of the engine compartment. All other elements of the point cloud are treated as stationary no-slip walls. These no-slip velocity boundary conditions are weakly enforced on the vehicle's point cloud.

In this incompressible airflow simulation, we use standard properties of air at atmospheric pressure and temperature $T = 20\,^{\circ}C$. The density is $1.204\,kg/m^3$, the dynamic viscosity is $1.825\,kg/(m\,s)$, the specific heat is $1007\,J/(kg\,K)$, and the thermal conductivity is $0.02514\,W/(m\,K)$. Points belonging to the engine, transmission, and axles are weakly enforced to a temperature of $T = 100\,^{\circ}C$. For all other elements of the vehicle point clouds, zero-heat flux thermal conditions are assumed. The domain is initialized with an ambient temperature of $20\,^{\circ}C$, which is also applied strongly at the inlet as a reference. Zero-heat flux conditions are prescribed on all other surfaces of the fluid domain. The time step size for the simulation is $\Delta t = 1.0 \times 10^5$.

5.3. Immersed mesh generation

Only in rare cases is an open-source meshing tool such as Gmsh [118] suitable for boundary-fitted volume meshing of production-ready product assemblies. However, owing to the topological simplicity of generating a parallelepiped volume mesh, notable examples of open-source mesh generation software typically provide more than enough features to set up an IMGA simulation case such as this example. Here, the functionalities accessed through Gmsh allow us to create the simulation domain in a streamlined manner. 2D and 3D element size parameters help control mesh density in the flow regions of interest. For targeted refinement of tetrahedra near the John Deere 544K point cloud, Gmsh's "Attractor" field trivializes the import of point coordinates to define smooth refinement regions surrounding every point cloud element. Continuing the theme of open-source workflow, sampling discrete points from CAD surfaces is similarly straightforward with Open CASCADE [119] and freely available Python packages libigl [120] and trimesh [121].

Visible in Fig. 23, the meshing strategy includes two rectangular prism refinement regions with the "outer" region completely enclosing the "inner" region, and elements adjacent to the floor have element size equal to that of the "inner" region's elements. A spherical refinement zone originates from each point cloud element of the coarsest case (⍰4 million points), prescribing a tetrahedral element size of 0.025 m to comfortably retain an average of at least one point cloud element per volume boundary element. This heuristic ratio is chosen to negate leakage of physical domain flow into the quiescent fictitious domain, based on the study conducted in Section 4.2. The size of the elements that contain (cut cells) or are near the point cloud is 0.025 m. The element size in the inner refinement zone and close to the floor is 0.1 m. The element sizes in the outer refinement zone and in the far field are 0.4 m and 1.0 m, respectively. Including both the physical and fictitious domains, the immersed mesh consists of 19,803,968 tetrahedral elements. A large majority of elements within the interior compartment must be refined, which increases the overall element count.

As mentioned earlier, the average distance between the points sampled on the B-rep model for the coarsest case is 0.025 m. Point spacing is decreased to produce three more point clouds with denser point distributions. For IMGA flow simulations, we observe that the estimated normals on the two coarsest point clouds are not sufficiently accurate to produce stable flow solutions, even though they satisfy the one-point-per-element criterion defined earlier. In the evaluation of weak-boundary-condition formulation, oscillatory normals can lead to significant instability. We find that the point cloud with 12 million points, which has an average spacing between points to be around 0.00625 m, has enough accuracy in the normal estimation and can produce a stable flow solution for our demonstration purposes. It should be noted again that requiring higher point cloud density here is not due to the point-cloud IMGA formulation but because of the need to obtain better normals estimation.

The parallelization strategy proposed by Hsu et al. [122] is employed for the IMGA simulations presented in this paper. In this strategy, the problem mesh is partitioned into subdomains by balancing the number of elements in each partition; each subdomain is then assigned to a processing core. However, this approach can sometimes create a highly unbalanced distribution of quadrature points in each partition since quadrature points aggregate in the cut elements due to the use of adaptive quadrature. In this work, we use the strategy proposed by Xu et al. [16] and weigh each element by the number of quadrature points it contains. This is then used as the metric in the graph/mesh partitioning package METIS [123] for determining mesh partitions by balancing the summation of user-defined weights. This approach produces a more balanced computational load on each processing core.

5.4. Simulation results

We solve for the airflow around the construction vehicle with the aforementioned boundary conditions and mesh. In this section, we observe an instantaneous snapshot of the flow solution. We focus on the velocity field, confirming

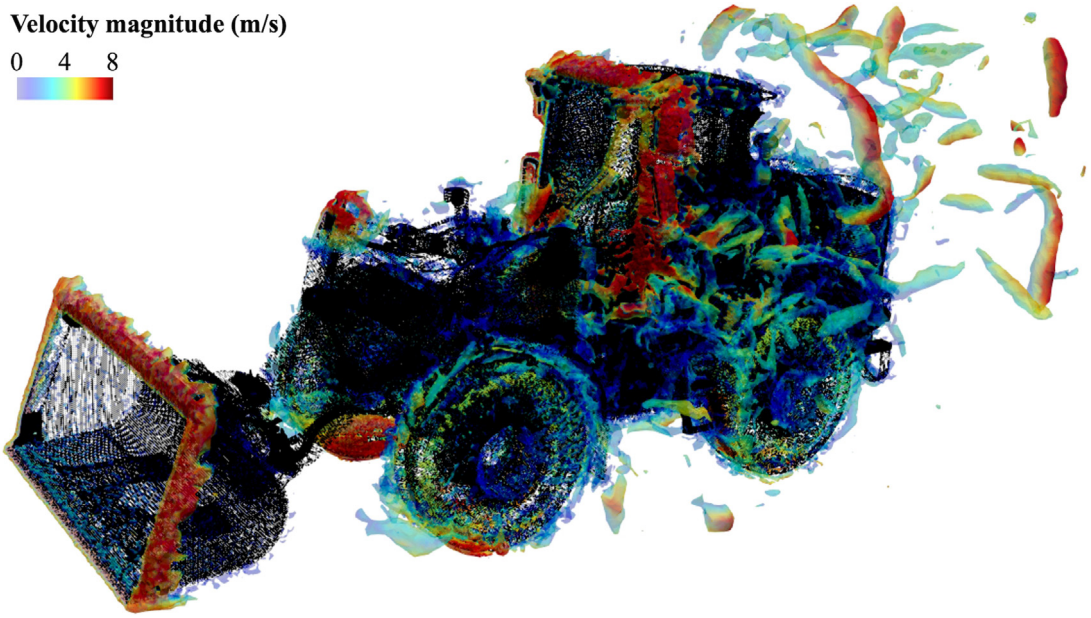


Fig. 24. Visualization of the instantaneous vortical structures of turbulent flow around the John Deere 544K colored by the velocity magnitude. The point cloud representation of the vehicle is used directly to perform IMGA. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

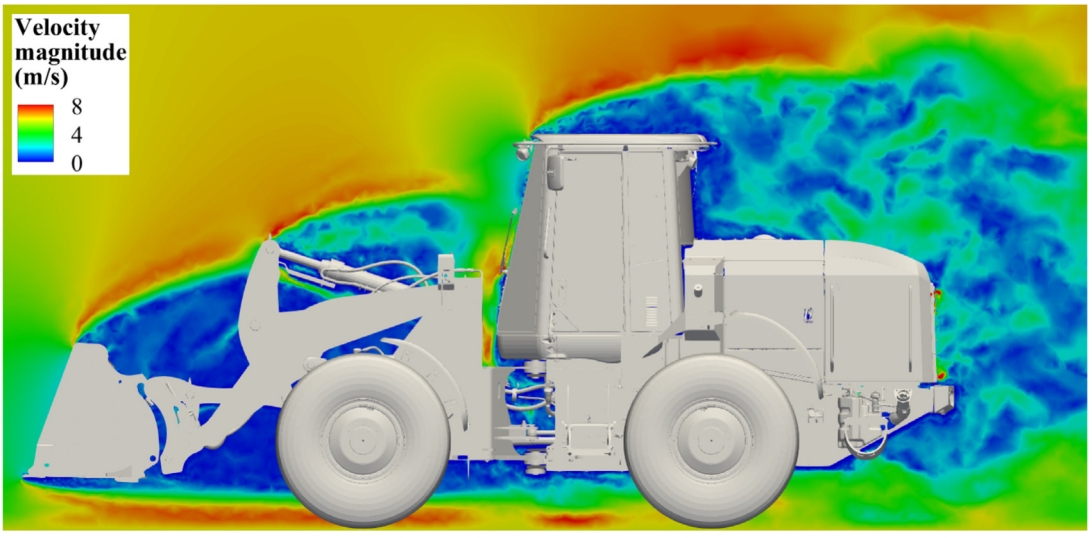


Fig. 25. Planar slice of the velocity field down the vehicle's center line.

that external airflow conforms to the immersed boundaries constructing the outer shell of the vehicle. We also look further inside the vehicle, especially in the engine compartment, to reveal the treatment of complex thin shell boundaries by inside–outside testing based on winding number. The latter portion of this section examines the vehicle interior, focusing on heat conduction and convection in the flow temperature field.

As is expected at $\mathrm{Re} = 3 \times 10^6$, external flow is fully turbulent across the vehicle. Fig. 24 shows the visualization of the instantaneous vortical structures colored by the velocity magnitude of turbulent flow around the John Deere 544K. The construction vehicle is visualized using a point cloud that was used to directly perform immersogeometric fluid flow and heat transfer analysis. To distinguish internal airflow from external airflow in subsequent visualizations, we substitute the surface CAD model in place of the point cloud utilized for simulation. Fig. 25 shows that the front bucket attachment forces the brunt of flow movement, simulating a recirculation

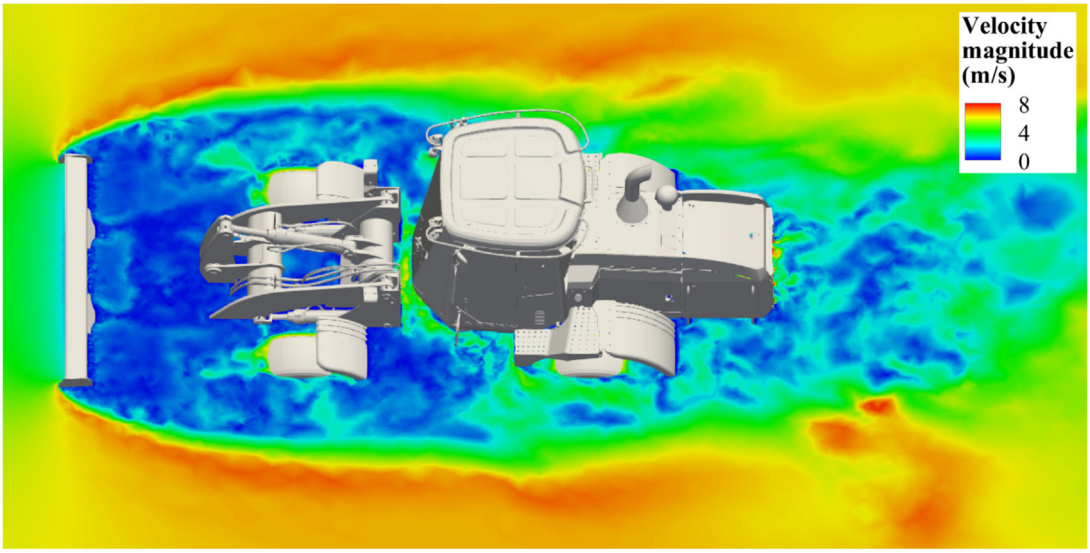


Fig. 26. Slice of the velocity field on a plane parallel to the ground.

bubble above and downstream of the bucket. The driver cabin appears to direct some flow into the lower engine compartment, whereas flow over the top of the cabin immediately separates and heightens the large wake region behind the 544K. Thicker boundary layers are apparent at the rear of the vehicle due to the adverse pressure gradients present there. Finally, the fan extracts air from the engine compartment and adds axial flow to the turbulent wake.

Looking at another sample of the same velocity field snapshot in Fig. 26, the rotating wheel boundary conditions are evident. The surface velocity of each wheel's contact patch is equal to the freestream velocity, applied as a weak boundary condition to elements containing pertinent elements of the point cloud. The tires create minor recirculation bubbles downstream, but vortex shedding appears to be deterred by the presence of mudguards. We see another dimension of flow movement by the front bucket attachment to the degree that the vehicle's entire width is engulfed by the bucket's wake region. From this angle of the velocity field slice, the fan shows a greater level of flow extraction and interacts with the weak vortex street trailing the vehicle. Moving onto the interior compartment in Fig. 27, we feature the same velocity field sample as in Fig. 25 but with 2D slicing of the 544K and visualization of individual mesh elements. The bulk of airflow enters the interior section through the gap between the cabin and the hydraulic assembly. We observe moving air in the inner region, desirable for convecting heat away from critical components into the colder ambient air.

We conclude this solution analysis by looking at the temperature field in Fig. 28, which is purely demonstrative of point-cloud IMGA's heat transfer capabilities and unrepresentative of the real-life vehicle's actual performance. A temperature boundary condition of value $T = 100\,^{\circ}C$ is weakly imposed on points belonging to the front axle hub, electronics housing, transmission case, rear axle hub, and engine. The long diagonal plate above the front axle hub impedes heat convection in that direction. As expected, the group of components in the primary engine compartment emits a greater heat load into the vehicle's wake region. The fan extracts high-temperature air that mixes with cooler air around the extremities of the compartment, pushing this air into the vehicle's wake. Again, the plates above the engine demonstrate heat conduction without air penetration across the boundaries.

6. Conclusions

We have presented a new method for immersogeometric fluid flow and heat transfer analysis that directly uses point cloud representation of objects. Analogous to previous work using analytic surface equations to generate surface Gaussian quadrature points, our method here repurposes point cloud elements into quadrature points. We employed computationally cheaper methods of normal vector and area generation using spatially local neighbors of points, proving that the results are appreciably accurate compared to full surface reconstruction. Prominent difficulties associated with geometry cleanup are avoided entirely, and the point cloud format trivializes immersed

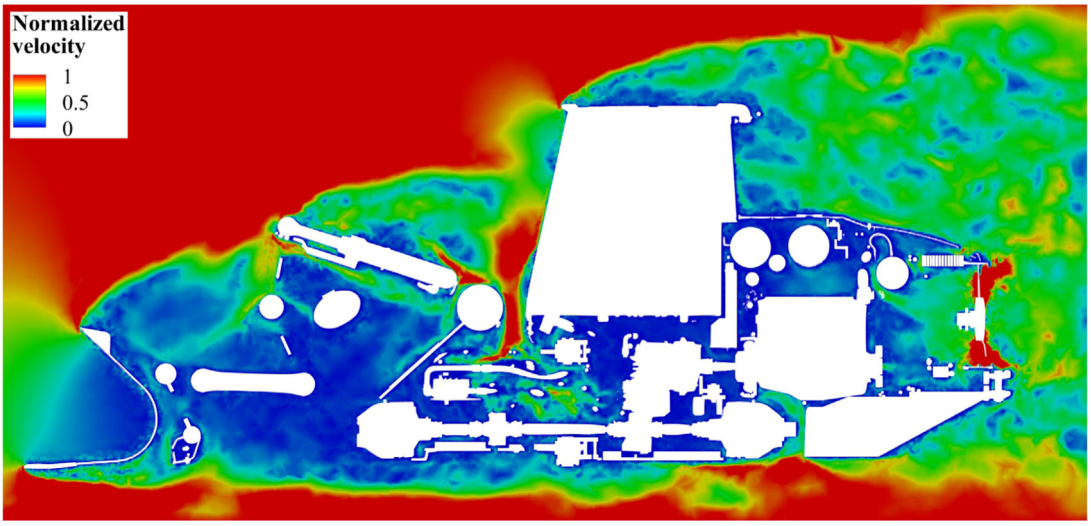


Fig. 27. Planar slice of both the normalized velocity field and John Deere 544K along the vehicle's center line. The velocity magnitude is normalized by the freestream velocity.

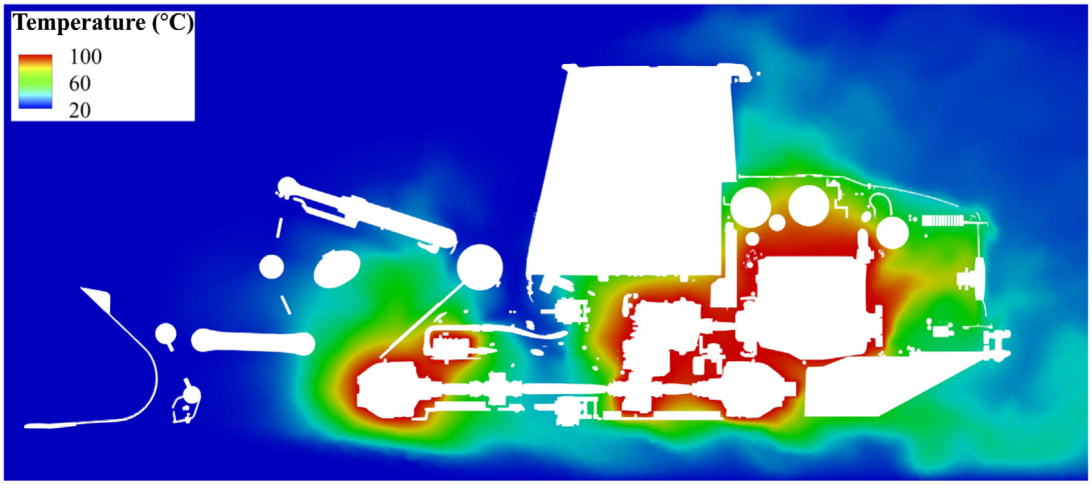


Fig. 28. Planar slice of both the temperature field (T, in °C) and John Deere 544K along the vehicle's center line.

mesh generation using even basic meshing codes. Simulation results obtained using immersogeometric point clouds are in excellent agreement with reference solutions. Our method is painlessly applied to perform flow analysis of an incredibly complex industrial geometry, a large construction vehicle, incorporating moving point clouds and thermal fluid boundary conditions to demonstrate the utility of point-cloud IMGA in commercial and industrial applications.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

J. Khristy was partly supported by Deere & Company, USA as a part-time employee. M.-C. Hsu was partially supported by the National Heart, Lung, and Blood Institute (NHLBI) of the National Institutes of Health (NIH), USA under Award No. R01HL129077. A. Krishnamurthy was partially supported by the National Science Foundation (NSF), USA Grant Nos. LEAP-HI-2053760 and OAC-1750865. This support is gratefully acknowledged. We also thank The Texas Advanced Computing Center (TACC) at the University of Texas at Austin for providing high-performance computing resources that contributed to the results presented in this paper.

Appendix. Definitions of Euler Jacobian and diffusivity matrices

In this appendix, we use the strong form of governing equations of compressible flows in 3D space to illustrate the definitions of Euler Jacobian and diffusivity matrices. The solution variables of compressible flows can be written using conservation variables U or pressure-primitive variables Y, defined as

$$U = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho e \end{bmatrix}, \text{ and } Y = \begin{bmatrix} p \\ u_1 \\ u_2 \\ u_3 \\ T \end{bmatrix} \tag{A.1}$$

where ρ is the density, u_i is the ith velocity component, $i = 1, \ldots, d$ with $d = 3$ here being the space dimension, e is the specific internal energy, p is the pressure, and T is the temperature. The pressure, density, and temperature are related through the ideal gas equation of state, $p = \rho R T$, where R is the ideal gas constant. Furthermore, we assume a calorically perfect gas in this work and define the specific internal energy as $e = c_v T$, where $c_v = R/(\gamma - 1)$ is the specific heat at constant volume, and γ is the heat capacity ratio. The governing equations of compressible flows can then be written as

$$U_{,t} + F^{adv}_{i,i} + F^{sp} - F^{diff}_{i,i} - S = 0, \tag{A.2}$$

where F^{adv}_i and F^{diff}_i are the vectors of convective and diffusive fluxes, respectively, defined as

$$F^{adv}_i = \begin{bmatrix} \rho u_i \\ \rho u_i u_1 \\ \rho u_i u_2 \\ \rho u_i u_3 \\ \rho u_i e \end{bmatrix} + \begin{bmatrix} 0 \\ p\delta_{1i} \\ p\delta_{2i} \\ p\delta_{3i} \\ 0 \end{bmatrix}, \text{ and } F^{diff}_i = \begin{bmatrix} 0 \\ \tau_{1i} \\ \tau_{2i} \\ \tau_{3i} \\ -q_i \end{bmatrix} \tag{A.3}$$

F^{sp} is the contribution of stress–power in the energy equation, defined as

$$F^{sp} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ p u_{i,i} - \tau_{ij} u_{j,i} \end{bmatrix}, \tag{A.4}$$

and S is the source term. In Eqs. (A.3)–(A.4), δ_{ij} is the Kronecker delta, and τ_{ij} and q_i are the viscous stress and heat flux, respectively, given by

$$\tau_{ij} = \lambda u_{k,k} \delta_{ij} + \mu \left(u_{i,j} + u_{j,i} \right), \tag{A.5}$$

$$q_i = -\kappa T_{,i}, \tag{A.6}$$

where μ is the dynamic viscosity, λ is the second coefficient of viscosity ($\lambda = -2\mu/3$ based on Stokes' hypothesis), and κ is the thermal conductivity. We further split the convective flux into $F^{adv}_i = F^{adv\backslash p}_i + F^{p}_i$, where $F^{adv\backslash p}_i$ and F^{p}_i are the first and second terms, respectively, on the right-hand side of F^{adv}_i's definition in Eq. (A.3).

With these preliminaries being defined, the Euler Jacobian and diffusivity matrices can be defined as follows: $\hat{A}_i = \frac{\partial F^{adv}_i}{\partial U}$, $\hat{A}^{sp}_i$ is such that $\hat{A}^{sp}_i U_{,i} = F^{sp}$, $\hat{K}_{ij}$ is such that $\hat{K}_{ij} U_{,j} = F^{diff}_i$, $A_0 = \frac{\partial U}{\partial Y}$, $A_i = \frac{\partial F^{adv}_i}{\partial Y} = \frac{\partial F^{adv}_i}{\partial U} \frac{\partial U}{\partial Y} =$

$\hat{A}_i A_0$, A^{sp}_i is such that $A^{sp}_i Y_{,i} = F^{sp}$, and K_{ij} is such that $K_{ij} Y_{,j} = F^{diff}_i$. Based on the splitting of F^{adv}_i into $F^{adv\backslash p}_i$ and F^p_i, we can further split A_i as $A_i = A^{adv\backslash p}_i + A^p_i$ to separate the pressure term from the convective flux. Detailed expressions for the matrices appearing in the quasi-linear forms can be found in Appendix A of Xu et al. [40].

References

[1] D. Kamensky, M.-C. Hsu, D. Schillinger, J.A. Evans, A. Aggarwal, Y. Bazilevs, M.S. Sacks, T.J.R. Hughes, An immersogeometric variational framework for fluid–structure interaction: Application to bioprosthetic heart valves, Comput. Methods Appl. Mech. Engrg. 284 (2015) 1005–1053.

[2] F. Xu, D. Schillinger, D. Kamensky, V. Varduhn, C. Wang, M.-C. Hsu, The tetrahedral finite cell method for fluids: Immersogeometric analysis of turbulent flow around complex geometries, Comput. & Fluids 141 (2016) 135–154.

[3] D.L. Marcum, J.A. Gaither, Unstructured grid generation for aerospace applications, in: M.D. Salas, W.K. Anderson (Eds.), Computational Aerosciences in the 21st Century, Vol. 8, Springer Netherlands, 2000, pp. 189–209.

[4] Z.J. Wang, K. Srinivasan, An adaptive Cartesian grid generation method for 'Dirty' geometry, Internat. J. Numer. Methods Fluids 39 (2002) 703–717.

[5] M.W. Beall, J. Walsh, M.S. Shephard, A comparison of techniques for geometry access related to mesh generation, Eng. Comput. 20 (2004) 210–221.

[6] Y.K. Lee, C.K. Lim, H. Ghazialam, H. Vardhan, E. Eklund, Surface mesh generation for dirty geometries by the Cartesian shrink-wrapping technique, Eng. Comput. 26 (2010) 377–390.

[7] M.-C. Hsu, C. Wang, F. Xu, A.J. Herrema, A. Krishnamurthy, Direct immersogeometric fluid flow analysis using B-rep CAD models, Comput. Aided Geom. Design 43 (2016) 143–158.

[8] C. Wang, F. Xu, M.-C. Hsu, A. Krishnamurthy, Rapid B-rep model preprocessing for immersogeometric analysis using analytic surfaces, Comput. Aided Geom. Design 52–53 (2017) 190–204.

[9] F. Xu, C. Wang, K. Hong, Y. Liu, Immersogeometric thermal analysis of flows inside buildings with reconfigurable components, J. Therm. Anal. Calorim. 143 (6) (2021) 4107–4117.

[10] V. Varduhn, M.-C. Hsu, M. Ruess, D. Schillinger, The tetrahedral finite cell method: Higher-order immersogeometric analysis on adaptive non-boundary-fitted meshes, Internat. J. Numer. Methods Engrg. 107 (2016) 1054–1079.

[11] S. Xu, F. Xu, A. Kommajosula, M.-C. Hsu, B. Ganapathysubramanian, Immersogeometric analysis of moving objects in incompressible flows, Comput. & Fluids 189 (2019) 24–33.

[12] S. Xu, B. Gao, A. Lofquist, M. Fernando, M.-C. Hsu, H. Sundar, B. Ganapathysubramanian, An octree-based immersogeometric approach for modeling inertial migration of particles in channels, Comput. & Fluids 214 (2021) 104764.

[13] Q. Zhu, F. Xu, S. Xu, M.-C. Hsu, J. Yan, An immersogeometric formulation for free-surface flows with application to marine engineering problems, Comput. Methods Appl. Mech. Engrg. 361 (2020) 112748.

[14] K. Saurabh, B. Gao, M. Fernando, S. Xu, M.A. Khanwale, B. Khara, M.-C. Hsu, A. Krishnamurthy, H. Sundar, B. Ganapathysubramanian, Industrial scale Large Eddy Simulations with adaptive octree meshes using immersogeometric analysis, Comput. Math. Appl. 97 (2021) 28–44.

[15] K. Saurabh, M. Ishii, M. Fernando, B. Gao, K. Tan, M.-C. Hsu, A. Krishnamurthy, H. Sundar, B. Ganapathysubramanian, Scalable adaptive PDE solvers in arbitrary domains, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '21, Association for Computing Machinery, New York, NY, USA, 2021, pp. 1–15.

[16] F. Xu, Y. Bazilevs, M.-C. Hsu, Immersogeometric analysis of compressible flows with application to aerodynamic simulation of rotorcraft, Math. Models Methods Appl. Sci. 29 (05) (2019) 905–938.

[17] T. Hoang, C.V. Verhoosel, C.-Z. Qin, F. Auricchio, A. Reali, E.H. van Brummelen, Skeleton-stabilized immersogeometric analysis for incompressible viscous flow problems, Comput. Methods Appl. Mech. Engrg. 344 (2019) 421–450.

[18] M.-C. Hsu, D. Kamensky, Y. Bazilevs, M.S. Sacks, T.J.R. Hughes, Fluid–structure interaction analysis of bioprosthetic heart valves: Significance of arterial wall deformation, Comput. Mech. 54 (2014) 1055–1071.

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

[20] D. Kamensky, M.-C. Hsu, Y. Yu, J.A. Evans, M.S. Sacks, T.J.R. Hughes, Immersogeometric cardiovascular fluid–structure interaction analysis with divergence-conforming B-splines, Comput. Methods Appl. Mech. Engrg. 314 (2017) 408–472.

[21] D. Kamensky, J.A. Evans, M.-C. Hsu, Y. Bazilevs, Projection-based stabilization of interface Lagrange multipliers in immersogeometric fluid–thin structure interaction analysis, with application to heart valve modeling, Comput. Math. Appl. 74 (9) (2017) 2068–2088.

[22] F. Xu, S. Morganti, R. Zakerzadeh, D. Kamensky, F. Auricchio, A. Reali, T.J.R. Hughes, M.S. Sacks, M.-C. Hsu, A framework for designing patient-specific bioprosthetic heart valves using immersogeometric fluid–structure interaction analysis, Int. J. Numer. Methods Biomed. Eng. 34 (4) (2018) e2938.

[23] M.C.H. Wu, H.M. Muchowski, E.L. Johnson, M.R. Rajanna, M.-C. Hsu, Immersogeometric fluid–structure interaction modeling and simulation of transcatheter aortic valve replacement, Comput. Methods Appl. Mech. Engrg. 357 (2019) 112556.

[24] E.L. Johnson, M.C.H. Wu, F. Xu, N.M. Wiese, M.R. Rajanna, A.J. Herrema, B. Ganapathysubramanian, T.J.R. Hughes, M.S. Sacks, M.-C. Hsu, Thinner biological tissues induce leaflet flutter in aortic heart valve replacements, Proc. Natl. Acad. Sci. 117 (2020) 19007–19016.

[25] F. Xu, E.L. Johnson, C. Wang, A. Jafari, C.-H. Yang, M.S. Sacks, A. Krishnamurthy, M.-C. Hsu, Computational investigation of left ventricular hemodynamics following bioprosthetic aortic and mitral valve replacement, Mech. Res. Commun. 112 (2021) 103604.

[26] E.L. Johnson, M.R. Rajanna, C.-H. Yang, M.-C. Hsu, Effects of membrane and flexural stiffnesses on aortic valve dynamics: Identifying the mechanics of leaflet flutter in thinner biological tissues, Forces Mech. 6 (2022) 100053.
[27] M.C.H. Wu, D. Kamensky, C. Wang, A.J. Herrema, F. Xu, M.S. Pigazzini, A. Verma, A.L. Marsden, Y. Bazilevs, M.-C. Hsu, Optimizing fluid–structure interaction systems with immersogeometric analysis and surrogate modeling: Application to a hydraulic arresting gear, Comput. Methods Appl. Mech. Engrg. 316 (2017) 668–693.
[28] D. Kamensky, Open-source immersogeometric analysis of fluid–structure interaction using FEniCS and tIGAr, Comput. Math. Appl. 81 (2021) 634–648.
[29] G.E. Neighbor, H. Zhao, M. Saraeian, M.-C. Hsu, D. Kamensky, Leveraging code generation for transparent immersogeometric fluid–structure interaction analysis on deforming domains, Eng. Comput. (2022) Accepted.
[30] M. Berger, A. Tagliasacchi, L.M. Seversky, P. Alliez, G. Guennebaud, J.A. Levine, A. Sharf, C.T. Silva, A survey of surface reconstruction from point clouds, Comput. Graph. Forum 36 (1) (2017) 301–329.
[31] A. Balu, S. Sarkar, B. Ganapathysubramanian, A. Krishnamurthy, Physics-aware machine learning surrogates for real-time manufacturing digital twin, Manuf. Lett. 34 (2022) 71–74.
[32] C. Rausch, R. Lu, S. Talebi, C. Haas, Deploying 3D scanning based geometric digital twins during fabrication and assembly in offsite manufacturing, Int. J. Comput. Math. (2021) http://dx.doi.org/10.1080/15623599.2021.1896942.
[33] A.K. Ghosh, A.S. Ullah, R. Teti, A. Kubo, Developing sensor signal-based digital twins for intelligent machine tools, J. Ind. Inf. Integr. 24 (2021) 100242.
[34] B. Jafari, A. Khaloo, D. Lattanzi, Deformation tracking in 3D point clouds via statistical sampling of direct cloud-to-cloud distances, J. Nondestruct. Eval. 36 (2017) 65.
[35] L. Kudela, S. Kollmannsberger, U. Almac, E. Rank, Direct structural analysis of domains defined by point clouds, Comput. Methods Appl. Mech. Engrg. 358 (2020) 112581.
[36] Y. Bazilevs, T.J.R. Hughes, Weak imposition of Dirichlet boundary conditions in fluid mechanics, Comput. & Fluids 36 (2007) 12–26.
[37] S. Xu, B. Gao, M.-C. Hsu, B. Ganapathysubramanian, A residual-based variational multiscale method with weak imposition of boundary conditions for buoyancy-driven flows, Comput. Methods Appl. Mech. Engrg. 352 (2019) 345–368.
[38] Y. Bazilevs, V.M. Calo, J.A. Cottrel, T.J.R. Hughes, A. Reali, G. Scovazzi, Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows, Comput. Methods Appl. Mech. Engrg. 197 (2007) 173–201.
[39] K. Takizawa, T.E. Tezduyar, T. Kuraishi, Multiscale space–time methods for thermo-fluid analysis of a ground vehicle and its tires, Math. Models Methods Appl. Sci. 25 (2015) 2227–2255.
[40] F. Xu, G. Moutsanidis, D. Kamensky, M.-C. Hsu, M. Murugan, A. Ghoshal, Y. Bazilevs, Compressible flows on moving domains: Stabilized methods, weakly enforced essential boundary conditions, sliding interfaces, and application to gas-turbine modeling, Comput. & Fluids 158 (2017) 201–220.
[41] D. Codoni, G. Moutsanidis, M.-C. Hsu, Y. Bazilevs, C. Johansen, A. Korobenko, Stabilized methods for high-speed compressible flows: toward hypersonic simulations, Comput. Mech. 67 (2021) 785–809.
[42] M.R. Rajanna, E.L. Johnson, D. Codoni, A. Korobenko, Y. Bazilevs, N. Liu, J. Lua, N. Phan, M.-C. Hsu, Finite element methodology for modeling aircraft aerodynamics: development, simulation, and validation, Comput. Mech. 70 (2022) 549–563.
[43] J. Nitsche, Über ein variationsprinzip zur lösung von Dirichlet-problemen bei verwendung von teilräumen, die keinen randbedingungen unterworfen sind, Abh. Math. Semin. Univ. Hambg. 36 (1971) 9–15.
[44] A. Düster, J. Parvizian, Z. Yang, E. Rank, The finite cell method for three-dimensional problems of solid mechanics, Comput. Methods Appl. Mech. Engrg. 197 (2008) 3768–3782.
[45] D. Schillinger, M. Ruess, The finite cell method: A review in the context of higher-order structural analysis of CAD and image-based geometric models, Arch. Comput. Methods Eng. 22 (3) (2015) 391–455.
[46] C. Johnson, Numerical Solution of Partial Differential Equations by the Finite Element Method, Cambridge University Press, Sweden, 1987.
[47] S.C. Brenner, L.R. Scott, The Mathematical Theory of Finite Element Methods, second ed., Springer, Berlin, 2002.
[48] F. Xu, S. Xu, U. Passe, B. Ganapathysubramanian, Computational study of natural ventilation in a sustainable building with complex geometry, Sustain. Energy Technol. Assess. 45 (2021) 101153.
[49] Y. Bazilevs, C. Michler, V.M. Calo, T.J.R. Hughes, Weak Dirichlet boundary conditions for wall-bounded turbulent flows, Comput. Methods Appl. Mech. Engrg. 196 (2007) 4853–4862.
[50] Y. Bazilevs, C. Michler, V.M. Calo, T.J.R. Hughes, Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes, Comput. Methods Appl. Mech. Engrg. 199 (2010) 780–790.
[51] Y. Bazilevs, A. Korobenko, J. Yan, A. Pal, S.M.I. Gohari, S. Sarkar, ALE–VMS formulation for stratified turbulent incompressible flows with applications, Math. Models Methods Appl. Sci. 25 (12) (2015) 2349–2375.
[52] K. Takizawa, T.E. Tezduyar, T. Kuraishi, S. Tabata, H. Takagi, Computational thermo-fluid analysis of a disk brake, Comput. Mech. 57 (6) (2016) 965–977.
[53] D. Schillinger, I. Harari, M.-C. Hsu, D. Kamensky, S.K.F. Stoter, Y. Yu, Y. Zhao, The non-symmetric Nitsche method for the parameter-free imposition of weak boundary and coupling conditions in immersed finite elements, Comput. Methods Appl. Mech. Engrg. 309 (2016) 625–652.
[54] M.-C. Hsu, I. Akkerman, Y. Bazilevs, Wind turbine aerodynamics using ALE–VMS: Validation and the role of weakly enforced boundary conditions, Comput. Mech. 50 (2012) 499–511.
[55] A.N. Brooks, T.J.R. Hughes, Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations, Comput. Methods Appl. Mech. Engrg. 32 (1982) 199–259.
[56] T.J.R. Hughes, T.E. Tezduyar, Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations, Comput. Methods Appl. Mech. Engrg. 45 (1984) 217–284.

[57] F. Shakib, T.J.R. Hughes, Z. Johan, A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier-Stokes equations, Comput. Methods Appl. Mech. Engrg. 89 (1991) 141–219.
[58] G.J. Le Beau, S.E. Ray, S.K. Aliabadi, T.E. Tezduyar, SUPG finite element computation of compressible flows with the entropy and conservation variables formulations, Comput. Methods Appl. Mech. Engrg. 104 (1993) 397–422.
[59] S.K. Aliabadi, T.E. Tezduyar, Space–time finite element computation of compressible flows involving moving boundaries and interfaces, Comput. Methods Appl. Mech. Engrg. 107 (1–2) (1993) 209–223.
[60] G. Hauke, T.J.R. Hughes, A unified approach to compressible and incompressible flows, Comput. Methods Appl. Mech. Engrg. 113 (1994) 389–396.
[61] T.E. Tezduyar, S.K. Aliabadi, M. Behr, S. Mittal, Massively parallel finite element simulation of compressible and incompressible flows, Comput. Methods Appl. Mech. Engrg. 119 (1994) 157–177.
[62] G.P. Wren, S.E. Ray, S.K. Aliabadi, T.E. Tezduyar, Space–time finite element computation of compressible flows between moving components, Internat. J. Numer. Methods Fluids 21 (1995) 981–991.
[63] G.P. Wren, S.E. Ray, S.K. Aliabadi, T.E. Tezduyar, Simulation of flow problems with moving mechanical components, fluid–structure interactions and two-fluid interfaces, Internat. J. Numer. Methods Fluids 24 (1997) 1433–1448.
[64] S. Mittal, T. Tezduyar, A unified finite element formulation for compressible and incompressible flows using augumented conservation variables., Comput. Methods Appl. Mech. Engrg. 161 (1998) 229–243.
[65] S.E. Ray, T.E. Tezduyar, Fluid–object interactions in interior ballistics, Comput. Methods Appl. Mech. Engrg. 190 (2000) 363–372.
[66] G. Hauke, Simple stabilizing matrices for the computation of compressible flows in primitive variables, Comput. Methods Appl. Mech. Engrg. 190 (2001) 6881–6893.
[67] T.J.R. Hughes, G. Scovazzi, T.E. Tezduyar, Stabilized methods for compressible flows, J. Sci. Comput. 43 (2010) 343–368.
[68] K. Takizawa, T.E. Tezduyar, T. Kanai, Porosity models and computational methods for compressible-flow aerodynamics of parachutes with geometric porosity, Math. Models Methods Appl. Sci. 27 (2017) 771–806.
[69] T. Kanai, K. Takizawa, T.E. Tezduyar, T. Tanaka, A. Hartmann, Compressible-flow geometric-porosity modeling and spacecraft parachute computation with isogeometric discretization, Comput. Mech. 63 (2019) 301–321.
[70] T.E. Tezduyar, Y.J. Park, Discontinuity capturing finite element formulations for nonlinear convection-diffusion-reaction equations, Comput. Methods Appl. Mech. Engrg. 59 (1986) 307–325.
[71] T.J.R. Hughes, M. Mallet, A. Mizukami, A new finite element formulation for computational fluid dynamics: II. Beyond SUPG, Comput. Methods Appl. Mech. Engrg. 54 (1986) 341–355.
[72] T.J.R. Hughes, M. Mallet, A new finite element formulation for computational fluid dynamics: IV. A discontinuity-capturing operator for multidimensional advective-diffusive systems, Comput. Methods Appl. Mech. Engrg. 58 (1986) 329–339.
[73] R.C. Almeida, A.C. Galeão, An adaptive Petrov–Galerkin formulation for the compressible Euler and Navier–Stokes equations, Comput. Methods Appl. Mech. Engrg. 129 (1) (1996) 157–176.
[74] G. Hauke, T.J.R. Hughes, A comparative study of different sets of variables for solving compressible and incompressible flows, Comput. Methods Appl. Mech. Engrg. 153 (1998) 1–44.
[75] T.E. Tezduyar, M. Senga, Stabilization and shock-capturing parameters in SUPG formulation of compressible flows, Comput. Methods Appl. Mech. Engrg. 195 (2006) 1621–1632.
[76] T.E. Tezduyar, M. Senga, D. Vicker, Computation of inviscid supersonic flows around cylinders and spheres with the SUPG formulation and YZβ shock-capturing, Comput. Mech. 38 (2006) 469–481.
[77] T.E. Tezduyar, M. Senga, SUPG finite element computation of inviscid supersonic flows with YZβ shock-capturing, Comput. & Fluids 36 (2007) 147–159.
[78] F. Rispoli, R. Saavedra, A. Corsini, T.E. Tezduyar, Computation of inviscid compressible flows with the V-SGS stabilization and YZβ shock-capturing, Internat. J. Numer. Methods Fluids 54 (2007) 695–706.
[79] F. Rispoli, R. Saavedra, F. Menichini, T.E. Tezduyar, Computation of inviscid supersonic flows around cylinders and spheres with the V-SGS stabilization and YZβ shock-capturing, J. Appl. Mech. 76 (2009) 021209.
[80] F. Rispoli, G. Delibra, P. Venturini, A. Corsini, R. Saavedra, T.E. Tezduyar, Particle tracking and particle–shock interaction in compressible-flow computations with the V-SGS stabilization and YZβ shock-capturing, Comput. Mech. 55 (2015) 1201–1209.
[81] K. Takizawa, T.E. Tezduyar, Y. Otoguro, Stabilization and discontinuity-capturing parameters for space–time flow computations with finite element and isogeometric discretizations, Comput. Mech. 62 (2018) 1169–1186.
[82] M.R. Rajanna, E.L. Johnson, N. Liu, A. Korobenko, Y. Bazilevs, M.-C. Hsu, Fluid–structure interaction modeling with nonmatching interface discretizations for compressible flow problems: computational framework and validation study, Math. Models Methods Appl. Sci. (2022) http://dx.doi.org/10.1142/S0218202522500592.
[83] G.J. Le Beau, T.E. Tezduyar, Finite element computation of compressible flows with the SUPG formulation, in: Advances in Finite Element Analysis in Fluid Dynamics, in: FED-Vol.123, ASME, New York, 1991, pp. 21–27.
[84] A. Embar, J. Dolbow, I. Harari, Imposing Dirichlet boundary conditions with Nitsche's method and spline-based finite elements, Internat. J. Numer. Methods Engrg. 83 (7) (2010) 877–898.
[85] M. Ruess, D. Schillinger, A.I. Özcan, E. Rank, Weak coupling for isogeometric analysis of non-matching and trimmed multi-patch geometries, Comput. Methods Appl. Mech. Engrg. 269 (2014) 46–731.
[86] W. Jiang, C. Annavarapu, J.E. Dolbow, I. Harari, A robust Nitsche's formulation for interface problems with spline-based finite elements, Internat. J. Numer. Methods Engrg. 104 (7) (2015) 676–696.
[87] F. de Prenter, C. Lehrenfeld, A. Massing, A note on the stability parameter in Nitsche's method for unfitted boundary value problems, Comput. Math. Appl. 75 (2018) 4322–4336.

[88] S.C. Divi, P.H. van Zuijlen, T. Hoang, F. de Prenter, F. Auricchio, A. Reali, E.H. van Brummelen, C.V. Verhoosel, Residual-based error estimation and adaptivity for stabilized immersed isogeometric analysis using truncated hierarchical B-splines, J. Mech. 38 (2022) 204–237.
[89] I. Harari, E. Grosu, A unified approach for embedded boundary conditions for fourth-order elliptic problems, Internat. J. Numer. Methods Engrg. 104 (7) (2015) 655–675.
[90] F. de Prenter, C.V. Verhoosel, G.J. van Zwieten, E.H. van Brummelen, Condition number analysis and preconditioning of the finite cell method, Comput. Methods Appl. Mech. Engrg. 316 (2017) 297–327.
[91] F. de Prenter, C.V. Verhoosel, E.H. van Brummelen, Preconditioning immersed isogeometric finite element methods with application to flow problems, Comput. Methods Appl. Mech. Engrg. 348 (2019) 604–631.
[92] F. de Prenter, C.V. Verhoosel, E.H. van Brummelen, J.A. Evans, C. Messe, J. Benzaken, K. Maute, Multigrid solvers for immersed finite element methods and immersed isogeometric analysis, Comput. Mech. 65 (2020) 807–838.
[93] J.T. Oden, I. Babuška, C.E. Baumann, A discontinuous hp finite element method for diffusion problems, J. Comput. Phys. 146 (1998) 491–519.
[94] B. Riviere, M.F. Wheeler, V. Girault, A priori error estimates for finite element methods based on discontinuous approximation spaces for elliptic problems, SIAM J. Numer. Anal. 39 (3) (2001) 902–931.
[95] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM J. Numer. Anal. 39 (2002) 1749–1779.
[96] R. Hartmann, Adjoint consistency analysis of discontinuous Galerkin discretizations, SIAM J. Numer. Anal. 45 (2007) 2671–2696.
[97] C.E. Baumann, J.T. Oden, A discontinuous hp finite element method for convection–diffusion problems, Comput. Methods Appl. Mech. Engrg. 175 (1999) 311–341.
[98] C.E. Baumann, J.T. Oden, A discontinuous hp finite element method for the Euler and Navier–Stokes equations, Internat. J. Numer. Methods Fluids 31 (1999) 79–95.
[99] E. Burman, A penalty-free nonsymmetric Nitsche-type method for the weak imposition of boundary conditions, SIAM J. Numer. Anal. 50 (2012) 1959–1981.
[100] T. Boiveau, E. Burman, A penalty-free Nitsche method for the weak imposition of boundary conditions in compressible and incompressible elasticity, IMA J. Numer. Anal. 36 (2016) 770–795.
[101] W.G. Dettmer, C. Kadapa, D. Perić, A stabilised immersed boundary method on hierarchical B-spline grids, Comput. Methods Appl. Mech. Engrg. 311 (2016) 415–437.
[102] H. Atkins, C.-W. Shu, Analysis of the discontinuous Galerkin method applied to the diffusion operator, in: 14th Computational Fluid Dynamics Conference, 1999, p. 3306, http://dx.doi.org/10.2514/6.1999-3306.
[103] R.M. Kirby, G.E. Karniadakis, Selecting the numerical flux in discontinuous Galerkin methods for diffusion problems, J. Sci. Comput. 22 (1–3) (2005) 385–411.
[104] F. Heimann, C. Engwer, O. Ippisch, P. Bastian, An unfitted interior penalty discontinuous Galerkin method for incompressible Navier–Stokes two-phase flow, Internat. J. Numer. Methods Fluids 71 (2013) 269–293.
[105] Y. Guo, M. Ruess, D. Schillinger, A parameter-free variational coupling approach for trimmed isogeometric thin shells, Comput. Mech. 59 (2017) 693–715.
[106] F. Cazals, M. Pouget, Estimating differential quantities using polynomial fitting of osculating jets, Comput. Aided Geom. Design 22 (2) (2005) 121–146.
[107] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, in: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, 1992, pp. 71–78.
[108] F. Cazals, M. Pouget, Algorithm 889: Jet_fitting_3:—A generic C++ package for estimating the differential properties on sampled surfaces via polynomial fitting, ACM Trans. Math. Software 35 (3) (2008) 24.
[109] S. Fuhrmann, M. Goesele, Floating scale surface reconstruction, ACM Trans. Graph. 33 (4) (2014) 46.
[110] M. Belkin, J. Sun, Y. Wang, Constructing Laplace operator from point clouds in R^d, in: Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '09, Society for Industrial and Applied Mathematics, USA, 2009, pp. 1031–1040.
[111] G. Barill, N.G. Dickson, R. Schmidt, D.I.W. Levin, A. Jacobson, Fast winding numbers for soups and clouds, ACM Trans. Graph. 37 (4) (2018) 43.
[112] A. Jacobson, L. Kavan, O. Sorkine-Hornung, Robust inside-outside segmentation using generalized winding numbers, ACM Trans. Graph. 32 (4) (2013) 33.
[113] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, W. Stuetzle, Piecewise smooth surface reconstruction, in: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, Association for Computing Machinery, New York, NY, USA, 1994, pp. 295–302.
[114] A. Kahler, Creating an icosphere mesh in code, 2009, http://blog.andreaskahler.com/2009/06/creating-icosphere-mesh-in-code.html.
[115] Y. Bazilevs, I. Akkerman, Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual-based variational multiscale method, J. Comput. Phys. 229 (2010) 3402–3414.
[116] I. Rodriguez, R. Borell, O. Lehmkuhl, C.D. Perez Segarra, A. Oliva, Direct numerical simulation of the flow over a sphere at Re=3700, J. Fluid Mech. 679 (2011) 263–287.
[117] H.S. Yoon, D.H. Yu, M.Y. Ha, Y.G. Park, Three-dimensional natural convection in an enclosure with a sphere at different vertical locations, Int. J. Heat Mass Transfer 53 (2010) 3143–3155.
[118] C. Geuzaine, J.-F. Remacle, Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities, Internat. J. Numer. Methods Engrg. 79 (11) (2009) 1309–1331.

[119] E.M. Datavision, Open CASCADE, 2001, http://www.opencascade.com.
[120] S. Koch, T. Schneider, F. Williams, D. Panozzo, Geometric computing with Python, in: ACM SIGGRAPH 2019 Courses (SIGGRAPH '19), Association for Computing Machinery, New York, NY, USA, 2019, pp. 1–45, Article 11.
[121] Dawson-Haggerty, et al., Trimesh, 2019, URL: https://trimsh.org.
[122] M.-C. Hsu, I. Akkerman, Y. Bazilevs, High-performance computing of wind turbine aerodynamics using isogeometric analysis, Comput. & Fluids 49 (2011) 93–100.
[123] G. Karypis, V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM J. Sci. Comput. 20 (1998) 359–392.