2017

A geometric framework for immersogeometric analysis

Chenglong Wang
Iowa State University

A geometric framework for immersogeometric analysis

by

Chenglong Wang

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Mechanical Engineering

Program of Study Committee:
Ming-Chen Hsu, Co-major Professor
Adarsh Krishnamurthy, Co-major Professor
Baskar Ganapathysubramanian
Wei Hong
James Oliver

Iowa State University

Ames, Iowa

2017

TABLE OF CONTENTS

ABSTRACT

The purpose of this dissertation is to develop a geometric framework for immersogeometric analysis that directly uses the boundary representations (B-reps) of a complex computer-aided design (CAD) model and immerses it into a locally refined, non-boundary-fitted discretization of the fluid domain. Using the non-boundary-fitted mesh which does not need to conform to the shape of the object can alleviate the challenge of mesh generation for complex geometries. This also reduces the labor-intensive and time-consuming work of geometry cleanup for the purpose of obtaining watertight CAD models in order to perform boundary-fitted mesh generation. The Dirichlet boundary conditions in the fluid domain are enforced weakly over the immersed object surface in the intersected elements. The surface quadrature points for the immersed object are generated on the parametric and analytic surfaces of the B-rep models. In the case of trimmed surfaces, adaptive quadrature rule is considered to improve the accuracy of the surface integral. For the non-boundary-fitted mesh, a sub-cell-based adaptive quadrature rule based on the recursive splitting of quadrature elements is used to faithfully capture the geometry in intersected elements. The point membership classification for identifying quadrature points in the fluid domain is based on a voxel-based approach implemented on GPUs. A variety of computational fluid dynamics (CFD) simulations are performed using the proposed method to assess its accuracy and efficiency. Finally, a fluid–structure interaction (FSI) simulation of a deforming left ventricle coupled with the heart valves shows the potential advantages of the developed geometric framework for the immersogeomtric analysis with complex moving domains.

CHAPTER 1. INTRODUCTION

In this dissertation, we introduce a novel geometric framework for immersogeometric analysis. This work is inspired by the integration between computer-aided design (CAD) and computer aided engineering (CAE). The goal is to efficiently handle boundary representation (B-rep) model preprocessing of complex objects for computational fluid dynamics (CFD) and fluid–structure interaction (FSI) analyses.

1.1 CAD and CAE

As reviewed by Hughes et al. [51], the market size of CAE is in the \$1–\$2 billion range and the market size of CAD is only in the \$5–\$10 billion range. However, in 2016, the market size of CAE grows to around \$3 billion [139], while the market size of CAD is still around \$9 billion [121]. It was shown in a technical report [139] that the compound annual growth rate (CAGR) of CAE engineering market is expected to be 11.1% between 2016 and 2021, which is much faster than the CAGR of CAD. The CAGR of CAD market is only close to 7% during the forecast period 2017 to 2021 [121]. With the demand to reduce the time-to-market of a product, there is a consistently increasing need for engineering practice to design, analyze and optimize a concept design before converging to a physical testing in the industry. However, there is still a noticeable gap between CAD and CAE. A typical engineering practice for CAE needs a computational mesh generated from a "clean" CAD model, which is typically an approximate geometric description of the original design. Hughes et al. [51] illustrated that in certain industries, 80% of overall analysis time can be devoted in mesh generation. To avoid this time-consuming procedure and make use of the CAD model directly, in this work, we propose to apply the immersogeometric methods for flow analysis and isogeometric methods for structural analysis.

1.2 Isogeometric and Immersogeometric Analysis

In recent years, the development of isogeometric analysis (IGA) [27, 51] has paved a path towards a tighter integration of engineering design and computational analysis. The core idea of IGA is to use the same basis functions for the representation of geometry in CAD and the approximation of solution fields in finite element analysis (FEA). Aside from its potential to eliminate unnecessary labor from the design-through-analysis pipeline [21, 98], IGA has attracted a great deal of attention due to the improvements in solution quality that follow from incorporation of smooth basis functions into engineering analysis [1, 28].

Over the last decade, IGA has been successfully employed in many areas of engineering and sciences, such as fluid mechanics and turbulence [2, 7, 8, 36, 78], structural and contact mechanics [17, 29, 30, 67, 122], fluid–structure interactions [10, 11], phase-field modeling [19, 39], collocation [3, 87, 99], efficient quadrature rules [4, 54, 100], boundary element methods [107, 112], shape and topology optimization [31, 68, 70], finite cell methods [86, 101, 102], trimmed geometries and patch coupling [42, 95, 103], analysis-suitable trivariate models [79, 118, 137], T-splines [9, 77, 106], and standardized file formats for data exchange between CAD and FEA packages [16, 20, 104].

Immersogeometric analysis was first introduced by Kamensky et al. [62] as a geometrically flexible technique for solving computational FSI problems involving large, complex structural deformations. This method focuses on accurately analyzing the physics of a complex object by immersing its surface representation into a non-boundary-fitted discretization of the background fluid domain. The method was first successfully applied to the FSI simulation of bioprosthetic heart valves (BHVs) [49, 50, 62]. The method was further investigated by Xu et al. [136] in the context of a tetrahedral finite cell approach [131] for the simulation of incompressible flow (both laminar and turbulent) around geometrically complex objects.

The main motivation behind the immersogeometric method is to alleviate the difficulties associated with CFD mesh generation around complex geometries. Creating a boundary-fitted fluid-domain mesh that accurately captures all the features of the design geometry is often time-consuming and labor intensive. Very often, small and thin geometric features are hard

to discretize and, as a result, require extensive geometry cleanup, defeaturing, and mesh manipulation [15, 76, 81, 134]. The immersogeometric method was proposed to eliminate these labor-intensive mesh generation procedures from the CFD simulation pipeline while still maintaining high accuracy of the simulation results. In addition, since the fluid domain is meshed independently and it is no longer necessary to defeature or remove small geometric features from the immersed object, the original design can be accurately preserved. The flexibility of the immersogeometric approach also allows it to be automated and placed in an optimization loop that searches for an optimal design [135].

1.3 Design-through-Analysis Workflow

Despite the progress achieved in the last decade, several challenges remain in effectively using IGA to improve the engineering process. Perhaps the biggest challenge is the rapid, (semi-)automatic construction of geometric models suitable for analysis. However, the difficulties of constructing designs and the corresponding geometric and analysis models are often overlooked in the engineering literature. It is often a time-consuming and challenging process to construct a baseline IGA model and to ensure the model has the desired features such as good parameterization, sufficient mesh density in the regions of interest, and, most importantly, analysis suitability. In many cases, intimate familiarity with CAD technology and advanced programming skills are necessary to successfully build such models. Design engineers, while professional in their application areas, may not have such skills. Furthermore, in many cases, engineers are only interested in a handful of design parameters and how they affect the product performance. As a result, to help design engineers and analysts make more effective use of IGA, in this dissertation, we develop an interactive IGA design-through-analysis platform based on the idea of parametric design and geometry modeling.

The interactive geometry modeling and parametric design platform can streamline the engineering design process by hiding the complex CAD functions in the background through generative algorithms, and let the user control the design through key design parameters. Since the design concept is integrated with analysis, the design parameters can include not only the geometry parameters, but also quantities such as material properties, loads, and boundary

conditions. In this dissertation, the concept of parametric design and geometry modeling is realized through a visual programming interface Grasshopper 3D [41], which is widely used by designers focusing on exploring new shapes using generative algorithms in Rhinoceros (Rhino) 3D. The advantage of using Grasshopper 3D for parametric design and geometry modeling is that, during the CAD model generation, one can ensure that the resulting IGA model is analysis-suitable.

The concept of parametric modeling is central to design in many fields of engineering and beyond (e.g., architecture [116]). Currently, parametric modeling is used in conjunction with solid geometry modeling that employs geometric primitives and Boolean operations (e.g., SolidWorks [113]). The use of parametric modeling with modern Spline technology like non-uniform rational B-splines (NURBS) or T-splines is not common and presents a novel research direction in IGA. This work builds on the concept of parametric modeling and provides a fairly general and convenient approach for creating parametric designs, which make use of NURBS and T-spline geometry description, using the visual programming concept. The approach is applicable to a large class of geometries, including surface and volumetric descriptions. The concept of parametric modeling plays an important role in solving three-dimensional FSI problem. As a result, one of my goals is to integrate a novel parametric modeling tool into the geometric framework to help design engineers and analysts to make more effective use of IGA and immersogeometric analysis. The FSI simulations of a complex model that includes a left ventricle (LV) and BHVs are performed.

Another novel and unique aspect of this work is the development of the IGA visualization tool directly within Rhino 3D CAD software. Good-quality visualization of the IGA simulation results is not a trivial matter. In many cases, this is done by interpolating the IGA solution with low-order finite-element functions and outputting the results using a standard finite-element data structure for visualization using existing software. However, in this dissertation, we develop a Rhino 3D plug-in that can be used to visualize NURBS and T-spline analysis results directly in Rhino.

1.4 Boundary Representations

The typical industry standard for the representation of geometry in mechanical CAD systems is B-reps. Although B-reps are ubiquitous in the CAD industry, they are not commonly used in the CAE analysis due to the challenges associated with directly using the B-rep information to perform geometric operations such as surface integration. Hence, the common practice in a mechanical analysis is to preprocess the B-reps by tessellating them into triangles and then using the triangular surface mesh to perform simulations. Generating the surface tessellations of complex CAD models is time-consuming and labor intensive, since the geometry needs to be manually checked to avoid creating any intersecting or non-manifold features (such as hanging nodes) during tessellation. In addition, the tessellation of curved surfaces represented using spline surfaces introduces a tessellation error depending on the size of the triangles used to approximate them.

A pioneering work using B-rep models directly in nonlinear isogeometric shell analysis was presented by Breitenberger et al. [21]. However, directly using B-rep models in flow analysis is still limited because for flow simulations, the meshing of the surrounding fluid domain needs to be considered in addition to having the object surface discretization. Generating a high-quality boundary-fitted fluid mesh requires intense manipulation of the surface mesh. Although there have been advances in using analysis-suitable trivariate T-splines [79, 133, 138] for volumetric discretization, using T-splines for CAD and CFD meshing continues to be limited by the geometric problems associated with the surface. To overcome these challenges, we present a novel method for immersogeometric fluid flow analysis that directly uses the CAD B-reps of a complex object and immersing it into a non-boundary-fitted discretization of the surrounding fluid domain. This work is inspired by Rank et al. [86], who proposed to extend the finite cell method [32, 84] to use CSG-tree and B-rep information for point membership classification, such that geometric models can be directly used in the finite cell analysis.

To directly use the B-reps of the CAD model and perform fluid flow simulations, the immersogeometric method needs to handle two types of surfaces. The standard surfaces for the B-rep CAD model are analytic and NURBS surfaces. CAD models of aerodynamic and hydrodynamic

structures have a large proportion of NURBS surfaces. However, CAD models of synthetic objects (such as road vehicles) usually have many flat features with rounded corners. These are represented internally in the CAD system using analytic surfaces rather than NURBS, since they can be easily manipulated by the solid modeling kernel and can be stored in a compact manner. Hanniel and Haller [44] performed a detailed survey of over $3,000$ SolidWorks models and concluded that more than 90% of the B-rep surfaces were analytic surfaces. Converting these surfaces to NURBS in order to perform analysis usually leads to poorly parameterized NURBS surfaces and can lead to poorly trimmed or missing surface features. In addition, converting simple geometries such as cylinders to NURBS imposes a performance penalty since these geometries have to be dealt with as rational splines. As a result, the geometry has to be inspected again after conversion to ensure analysis compatibility and can increase the computational cost.

1.5 Immersogeometric Analysis using Boundary Representations

The immersogeometric method for CFD is comprised of the following main components. A variational multiscale (VMS) formulation of incompressible flow [8, 52, 53, 55] is used, which provides accuracy and robustness in both laminar and turbulent flow conditions. The Dirichlet boundary conditions on the surface of the immersed objects are enforced weakly in the sense of Nitsche's method [12, 83]. Adaptively refined quadrature rules are used to faithfully capture the flow domain geometry in the discrete problem without modifying the non-boundary-fitted background mesh. Xu et al. [136] found that the faithful representation of the geometry in intersected elements is critical for accurate immersogeometric fluid flow analysis.

There are two main preprocessing steps required to perform immersogeometric analysis on complex CAD models. First, Gaussian quadrature information needs to be generated on the surface of the immersed model for the purpose of evaluating the weak Dirichlet boundary conditions [12]. Second, the fluid-domain mesh with adequate refinement along the surfaces of the immersed object needs to be generated. The point membership classification is performed on the points of the fluid-domain mesh to identify points outside or inside the solid CAD model. In this dissertation, we develop new methods to perform these operations directly using

the B-reps of the CAD model. The adaptive surface quadrature rules are applied in the both parametric and analytic surfaces used in CAD models. We also develop a mesh refinement technique that uses a hierarchical voxelization of the immersed object to generate an analysis-suitable fluid mesh.

The boundary of solid models created using CAD systems is usually represented using multiple trimmed surfaces. Previous implementations of immersogeometric analysis using complex CAD models rely on the triangular tessellation of surfaces for analysis. Tessellating the surfaces individually can lead to the generation of small gaps along the edges of adjacent surface patches. Tessellating these surfaces to create surface triangulations requires the use of adjacency information usually stored in the B-rep of the solid model to prevent gaps. This makes the tessellation process more tedious and requires specialized algorithms. In addition, since these surfaces are usually trimmed, additional surface quadrature points need to be evaluated close to the trim curves. In the proposed method, we directly make use of the trimmed surfaces without tessellating them. The surfaces are evaluated uniformly to the required level of precision and the parametric locations close to the trim curves are adaptively refined to include additional surface quadrature points as required.

In the case of triangles, the Gaussian quadrature points were directly generated on the planar triangular surface. However, in the case of parametric surfaces (NURBS), the parameterization is used for storing the trim curves. In addition, the parametric space can be easily mapped to a normalized $[0, 1] \times [0, 1]$ domain, which makes the development of Gaussian quadrature rules straight forward. The normalized parametric domain can be easily subdivided using quad-tree techniques to generate adaptive surface quadrature rules for trimmed surfaces. In the case of analytic surfaces, the surface parameterization is not easily mapped to a normalized parametric domain, which complicates the development of adaptive surface quadrature rules. In this dissertation, we develop new method making use of 2D bounding boxes in the parametric space of the different analytic surfaces to generate adaptive surface quadrature rules.

In boundary-conforming fluid-flow analysis, the surface mesh of the immersed object can be directly used to generate the fluid-domain mesh. This surface mesh along with the growth parameter of an advancing-front volume mesh generation algorithm [80] can be used together to

maintain a smaller mesh size closer to the surface of the immersed object. The purpose of the smaller mesh size is to better resolve boundary layers. However, in immersogeometric analysis, the lack of a surface mesh complicates the generation of an adaptive fluid-domain mesh, which can lead to a mesh with a large number of elements that significantly increase the computational time. In this dissertation, we create a hierarchical voxelization of the immersed CAD model using GPU rendering techniques [71, 72]. We make use of this hierarchical voxelization to set the size parameters of the fluid-domain mesh generation algorithm. This method produces a mesh that has sufficiently small elements close to the immersed object boundary while producing an overall fluid-domain mesh with fewer elements.

Point membership classification of the vertices of the background mesh is traditionally performed using a ray-tracing approach [88, 94]. Any point inside the solid model intersects the surface of the model an odd number of times. Performing this operation directly on B-reps consisting of trimmed surfaces is a compute-intensive operation. In this method, we create a high-resolution voxelization of the CAD model using GPU rendering of trimmed surfaces [71, 73]. This voxelization is then used to perform point membership classification on vertices of the background mesh. The same operation is repeated to perform point membership classification on the volume quadrature points in the background mesh.

1.6 Proposed Framework

To allow researchers to make effective use of the aforementioned methods, the focus of this research is to develop a geometric framework for immersogeometric analysis that directly uses the B-reps of complex CAD models. The object, which is based on the idea of parametric design modeling, is immersed into a non-boundary-fitted discretization of the fluid domain. The framework includes all important functionalities for the entire design process and shows rapid B-rep model preprocessing for immersogeometric analysis using both analytic and parametric surfaces of immersed objects. The concept of the proposed geometric framework for immersogeometric analysis is illustrated in Figure 1.1. The first step is the parametric geometry modeling that is typically used in CAD softwares [90, 113]. The Gaussian quadrature information of the B-rep surfaces is then extracted for surface integrals in immersogeometric

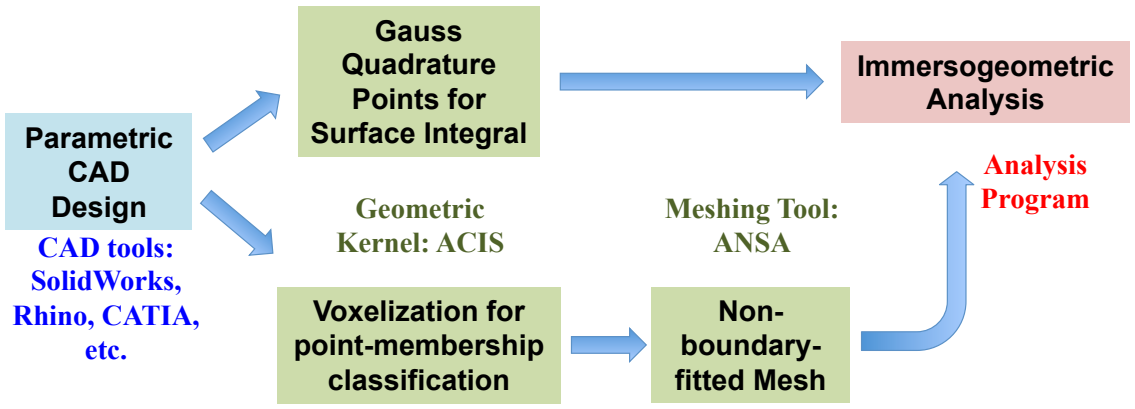


Figure 1.1: A geometric framework for immersogeometric analysis.

analysis. The voxelization of the CAD model is used for the local refinement and for the points membership classification of the non-boundary-fitted mesh. After the model preprocessing, the immersogeometric fluid flow analysis is performed to provide feedback to a designer or analyst for the evaluation of the geometry model.

1.7 Dissertation Structure

This dissertation is organized as follows. In Chapter 2, we present the development of a parametric design and geometry modeling platform for IGA. In Chapter 3, we present the development of a geometric framework for immersogeometric analysis and apply it to the simulation of flow around analytic and NURBS-based B-rep CAD models. In Chapter 4, we present the FSI simulation of a left ventricular model coupled with aortic and mitral valves using the proposed methods. In Chapter 5, we draw conclusions and discuss future possibilities.

CHAPTER 2. PARAMETRIC DESIGN PLATFORM FOR ISOGEOMETRIC ANALYSIS

In this chapter, an interactive parametric design-through-analysis platform is proposed to help design engineers and analysts make more effective use of IGA to improve their product design and performance. We develop several Rhino plug-ins to take input design parameters through a user-friendly interface, generate appropriate surface models, perform mechanical analysis, and visualize the solution fields, all within the same CAD program. As part of this effort, we develop graphical generative algorithms for IGA model creation and visualization based on Grasshopper, a visual programming interface to Rhino. The developed platform is demonstrated on a structural mechanics example of wind turbine blade.

2.1 IGA Design-through-Analysis Platform

In this section, we develop an IGA design-through-analysis platform that offers a user-friendly interface for the design process, including geometry modeling, assignment of model material parameters, loads, and boundary conditions, computational analysis, and post-processing. The concept of parametric design [6, 75, 130] is integrated into the platform in order to enable repetitive design, quality improvement of geometric models [26, 132], and rapid analysis of designs linked by model parameters.

2.1.1 Platform structure

The platform provides a closed-loop design method for engineering applications as depicted in Figure 2.1. Once a basic model has been designed through CAD software and after distributing material parameters and specifying boundary and load conditions on model surfaces, a simulation may be performed directly using this model. After inspecting the solution, the designer can then make a judgment about where to improve the current design. Because the

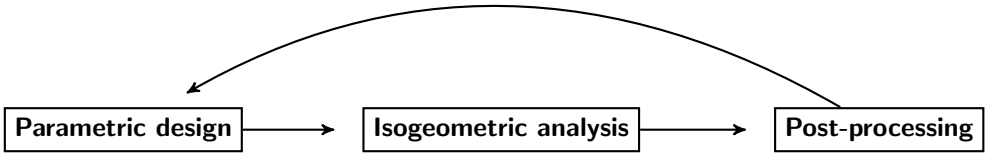


Figure 2.1: Isogeometric design-through-analysis platform structure.

original geometry is modeled parametrically, changing the design according to the analysis results would consist of simply adjusting input parameters. As a result, within this closed-loop design process, the user could conceivably create and optimize designs within a shorter timeframe using a single platform.

In order to achieve the goals, we choose Rhino 3D CAD software [90] for the development of the platform. Rhino 3D gives designers a variety of tools that are required to build complex, multi-patch NURBS surfaces [85]. Recently, additional functionality was added in Rhino 3D to create and manipulate T-spline surfaces [5, 108], which is an important enhancement allowing one to move away from a fairly restrictive NURBS-patch-based geometry design to a completely unstructured, watertight surface definition while respecting most of the constraints imposed by analysis [35, 106]. Rhino 3D also features an enhanced graphic programming tool called Grasshopper 3D [41] for designing generative algorithms, and utilizes free and open-source software development kits (SDK) [89] for plug-in development. Furthermore, Rhino 3D is relatively transparent as compared to other CAD software in that it provides the user with the ability to interact with the system through the so-called "plug-in" commands. All of these features are well aligned with my goals, and we make use of them in the design of this IGA-based design-through-analysis platform.

Figure 2.2 shows a snapshot of the Rhino 3D CAD modeling software interface, with the proposed platform plug-ins integrated. The figure shows a full wind-turbine model represented using T-spline surfaces. Figure 2.3 shows the developed plug-in commands (or "buttons"), including parametric geometry design, assignment of material parameters, boundary and load conditions imposition, interface to an IGA shell solver, post-processing, and visualization of the analysis result. The details of the plug-in commands and their use are illuminated in the applications sections of this dissertation.

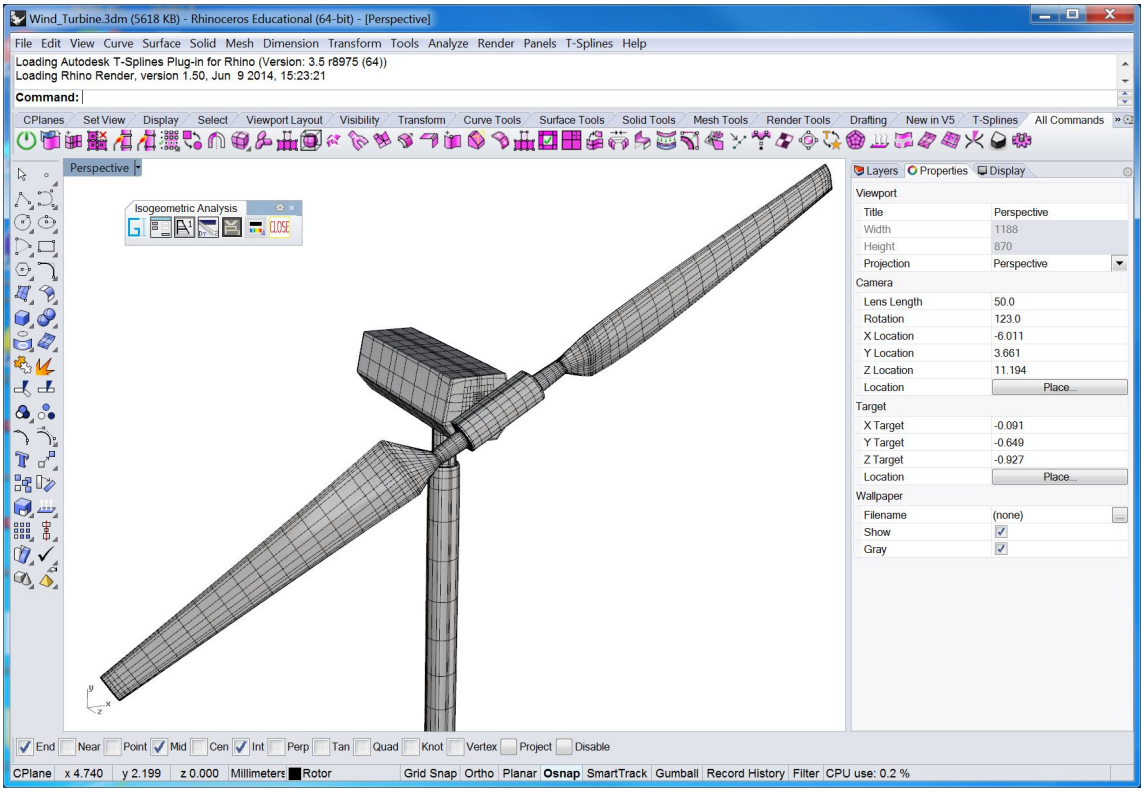


Figure 2.2: Rhino 3D CAD modeling software with the proposed plug-ins integrated.

(a) Geometry setup

(b) TsSelSet

(c) Material setup

(d) IGA shell solver

(e) Deformed shape

(f) Post-processing

(g) Visualization

(h) Close project

Figure 2.3: Platform plug-in options.

2.1.2 Visual programming for IGA modeling

As a visual programming interface, Grasshopper 3D moves away from the traditional paradigm of writing a text file with program instructions and feeding it to an off-the-shelf compiler to produce an executable file. Using Grasshopper 3D, the program is written in terms of "components" with pre-defined functionality, and "wire connections" between the components that serve as conduits of input and output data. By creatively arranging components and connections, one can rapidly generate an analysis model, establish parametric control, and link the model to the desired solver and visualization modalities. In the case when new functionality is needed, a traditional programming approach may be employed to create new components, which are then added to the library of the existing ones, and may be flexibly used by the designer.

2.1.3 Parametric design and geometry modeling

During the design cycle, geometric models are often constructed through similar design algorithms. We develop an interactive parametric geometry modeling plug-in that enables rapid construction of analysis-suitable, multi-patch NURBS models using Grasshopper 3D [41]. The plug-in streamlines the engineering design process by hiding the complex CAD functions in the background through generative algorithms and letting the user control the design through key design parameters. An added benefit of using this approach for parametric design and geometry modeling is that, during the CAD model generation, one can ensure that the resulting IGA model is analysis-suitable.

Figure 2.4 shows an example of using the Grasshopper 3D generative algorithm for parametric geometry design of a wind turbine blade. The leftmost group of components called "Input Parameters" represents operations on input data such as surface continuity, airfoil type, radial position, twist angle and axis, and chord length. The group includes the in-house developed VBScript (Visual Basic Scripting Edition) component for reading these input parameters from a user interface. The user interface is shown in Figure 2.5(a) and is developed using C# [91]. The group of components called "Airfoil Curve Construction" imports unit-chord-length airfoil data given by the users and constructs smooth NURBS curves interpolating through each set

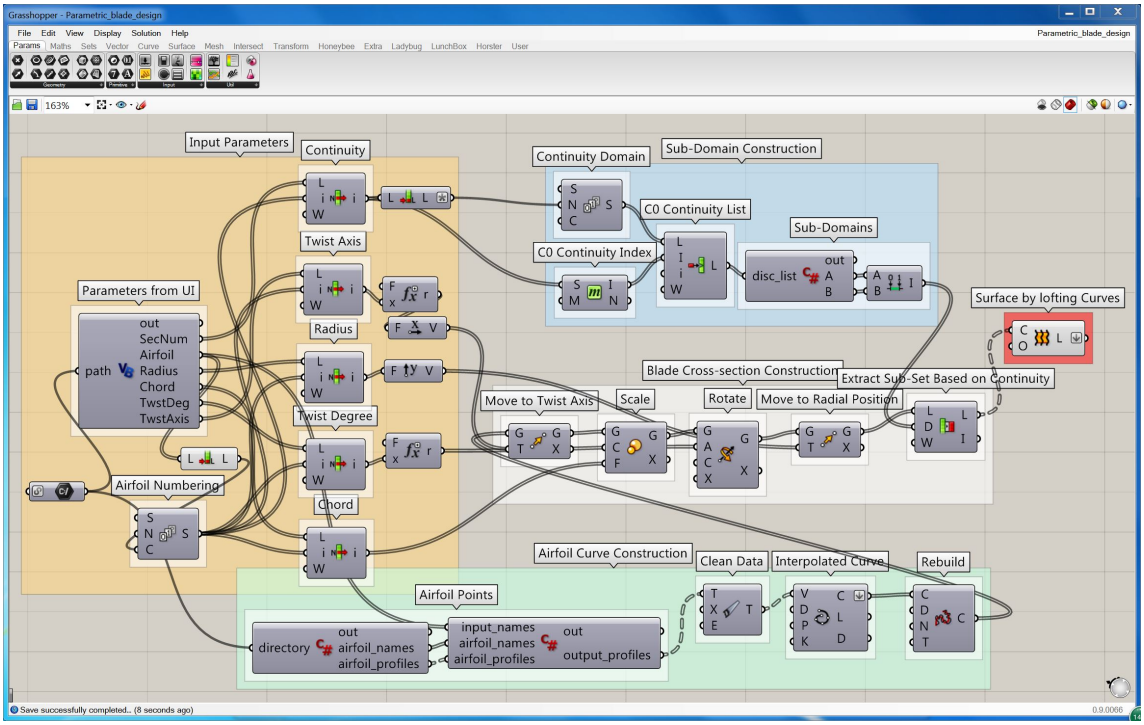
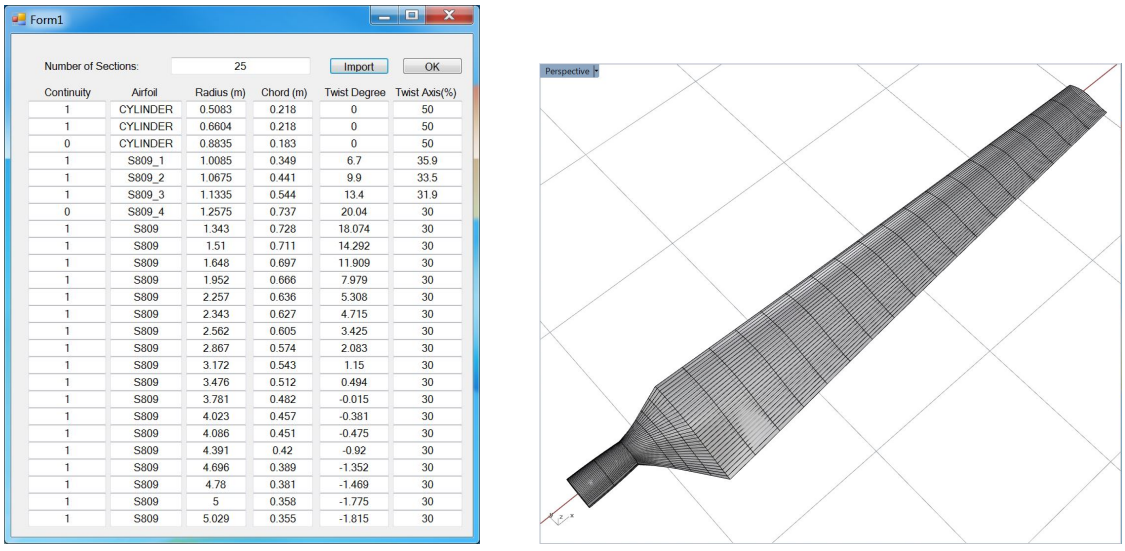


Figure 2.4: Grasshopper 3D generative algorithm for parametric geometry design of wind turbine blade.



Continuity	Airfoil	Radius (m)	Chord (m)	Twist Degree	Twist Axis(%)
1	CYLINDER	0.5083	0.218	0	50
1	CYLINDER	0.6604	0.218	0	50
0	CYLINDER	0.8835	0.183	0	50
1	S809_1	1.0085	0.349	6.7	35.9
1	S809_2	1.0675	0.441	9.9	33.5
1	S809_3	1.1335	0.544	13.4	31.9
0	S809_4	1.2575	0.737	20.04	30
1	S809	1.343	0.728	18.074	30
1	S809	1.51	0.711	14.292	30
1	S809	1.648	0.697	11.909	30
1	S809	1.952	0.666	7.979	30
1	S809	2.257	0.636	5.308	30
1	S809	2.343	0.627	4.715	30
1	S809	2.562	0.605	3.425	30
1	S809	2.867	0.574	2.083	30
1	S809	3.172	0.543	1.15	30
1	S809	3.476	0.512	0.494	30
1	S809	3.781	0.482	-0.015	30
1	S809	4.023	0.457	-0.381	30
1	S809	4.086	0.451	-0.475	30
1	S809	4.391	0.42	-0.92	30
1	S809	4.696	0.389	-1.352	30
1	S809	4.78	0.381	-1.469	30
1	S809	5	0.358	-1.775	30
1	S809	5.029	0.355	-1.815	30

(a) User interface for parametric design.

(b) NREL Phase VI wind turbine blade.

Figure 2.5: NREL Phase VI wind turbine blade parametric design and geometry modeling.

of airfoil data points. The "rebuild" function is then used on these curves to make sure that all the NURBS airfoil profiles have the same number of control points and knot vectors. This operation ensures that the NURBS surface generated by skinning (or lofting) along this series of profile curves will have the desired parameterization.

Based on the input parameters corresponding to blade cross sections, each airfoil profile is relocated such that the twist axis is aligned to the origin, scaled by the chord length, and rotated according to the twist degree. The modified airfoil profiles are then moved to their corresponding radial positions along the twist axis, which is also the blade-pitched axis. The group of components named "Blade Cross-section Construction" in Figure 2.4 perform this procedure. Due to the inherent discontinuity of sharp transition between different blade design zones, the input data are separated into different subdomains using the group of components named "Sub-Domain Construction." After all subsets of blade cross section curves are prepared, individual NURBS surfaces are generated by skinning (or lofting) along the curves within each subset. This is done by the rightmost component shown in Figure 2.4. The multi-patch NURBS surface generated through this procedure is conforming between different patches.

The proposed concept is applied to the parametric geometry design of an NREL Phase VI wind turbine blade [43, 47] that requires a considerable number of parametric inputs, including the geometric continuity of each cross section, airfoil type, radial airfoil location, and chord length. This NREL wind turbine blade has 25 airfoil cross sections. It gradually changes from a cylindrical cross section at the hub center to an S809 airfoil [114] cross section along the blade to the blade tip. The main input parameters are shown in Figure 2.5(a). The S809 airfoil data points shown in Figure 2.6 can be stored in a text file and imported via the interface. After entering all the information, a multi-patch NURBS surface of the wind turbine blade is generated as shown in Figure 2.5(b).

2.1.4 Visualization of NURBS and T-spline analysis results

After solving the IGA problem, the control variables (or degrees of freedom) for the solution fields (e.g., displacement, velocity, temperature, etc.) are obtained. These control variables are defined on the control points, which are typically not located on the physical geometry. When

S809.dat			
x/c	*y/c*	*x/c*	*y/c*
1.000	0.000	0.000	-0.002
0.985	0.002	0.001	-0.004
0.945	0.011	0.009	-0.011
0.885	0.023	0.042	-0.030
0.807	0.038	0.093	-0.052
0.716	0.055	0.158	-0.074
0.617	0.074	0.232	-0.092
0.520	0.093	0.312	-0.105
0.428	0.102	0.395	-0.108
0.337	0.100	0.482	-0.097
0.250	0.092	0.576	-0.074
0.171	0.079	0.677	-0.047
0.104	0.062	0.776	-0.024
0.052	0.042	0.867	-0.008
0.017	0.022	0.938	0.000
0.001	0.004	0.984	0.001
0.000	0.000	1.000	0.000

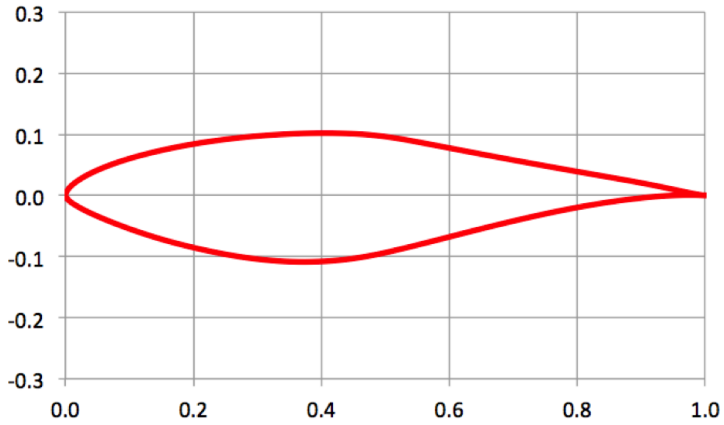


Figure 2.6: S809 airfoil data and profile [114].

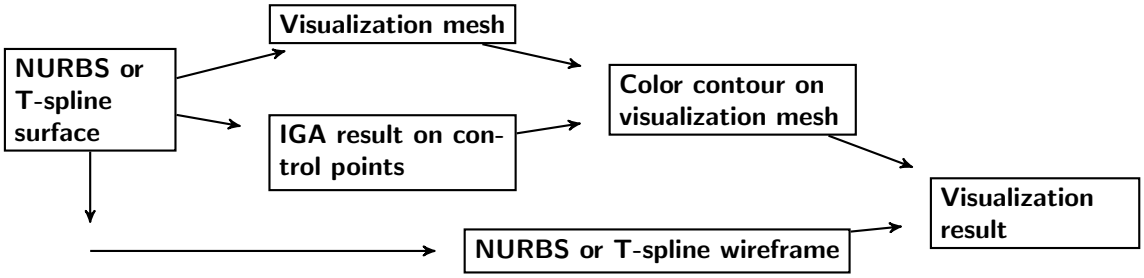


Figure 2.7: Visualization procedure for NURBS or T-spline IGA result.

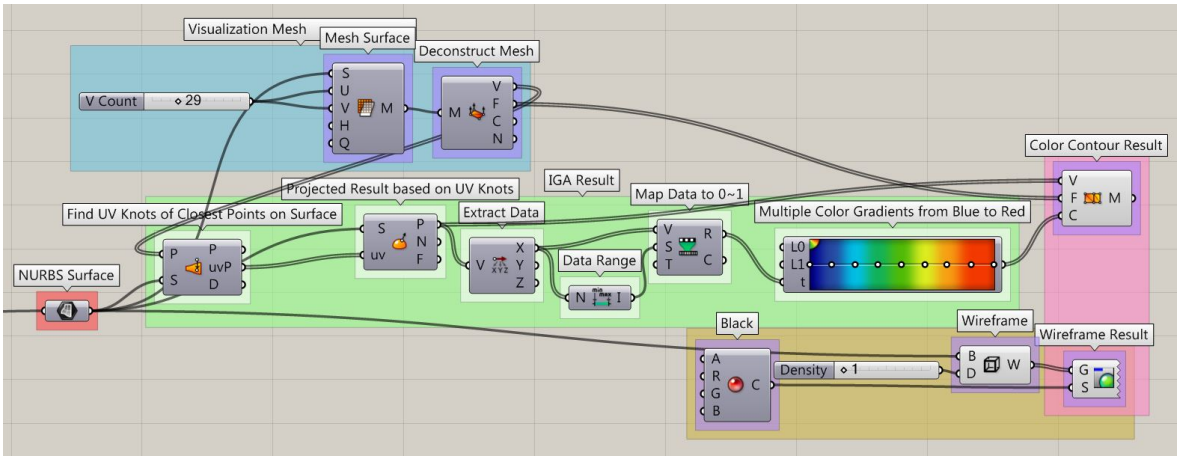


Figure 2.8: Grasshopper 3D components for visualization of NURBS analysis result.

coupled with basis functions, these give continuous solution fields on the geometry. Visualization of the IGA results is an integral component of the design-through-analysis framework and presents some challenges that we address in this section. Here we focus on visualization of the solution fields defined in terms of NURBS and T-spline functions performed within the same IGA design-through-analysis platform.

Figure 2.7 shows a conceptual diagram of the visualization procedure for NURBS and T-spline IGA results. A visualization mesh is constructed directly from the NURBS or T-spline surface in order to visualize color contours of the solution fields that are defined on the control points. The visualization mesh points require the solution values, which can be evaluated at their closest points on the NURBS or T-spline surface. The color contours of the visualization mesh are then overlapped with the wireframe extracted from the NURBS or T-spline surface. The combination of these two inputs provides a novel way of visualizing the IGA results directly within the CAD software.

An implementation of this idea as a Rhino 3D plug-in is shown in Figure 2.8, which is a Grasshopper 3D generative algorithm for visualizing IGA results directly in Rhino 3D. More details about the plug-in are given in what follows.

2.1.4.1 NURBS and T-spline surface construction

Figure 2.9 shows an example of constructing a NURBS surface in Grasshopper 3D using parametric inputs including control point information, degree, and knot vector. The example mesh has 16 control points, four in each parametric direction. The polynomial degrees are cubic in both directions, leading to a one-element mesh. The VBScript component includes all the necessary functions to build a NURBS surface based on the user inputs. The constructed NURBS surface is shown in Figure 2.10, where the control points and control polygons are also visualized. This surface is used for testing and demonstrating the color contour visualization concept and procedure. A T-spline surface can be created from the NURBS surface, as shown in Figure 2.11.

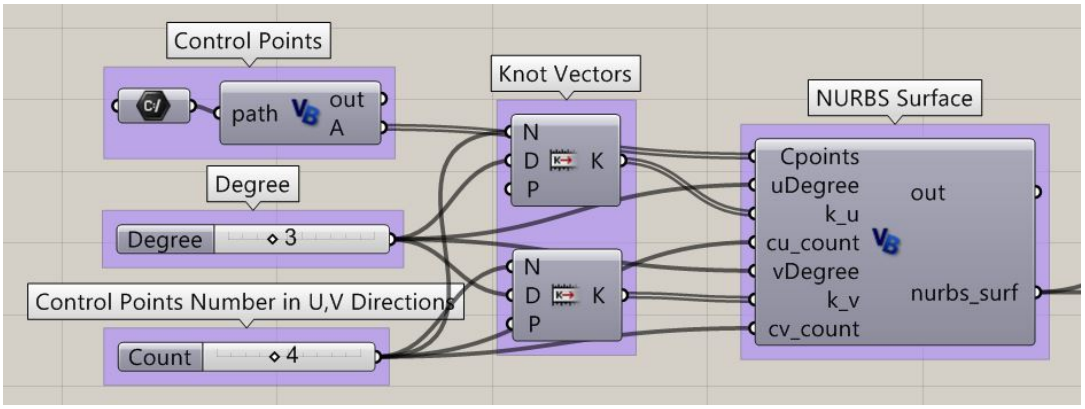


Figure 2.9: NURBS surface construction in Grasshopper 3D using parametric inputs.

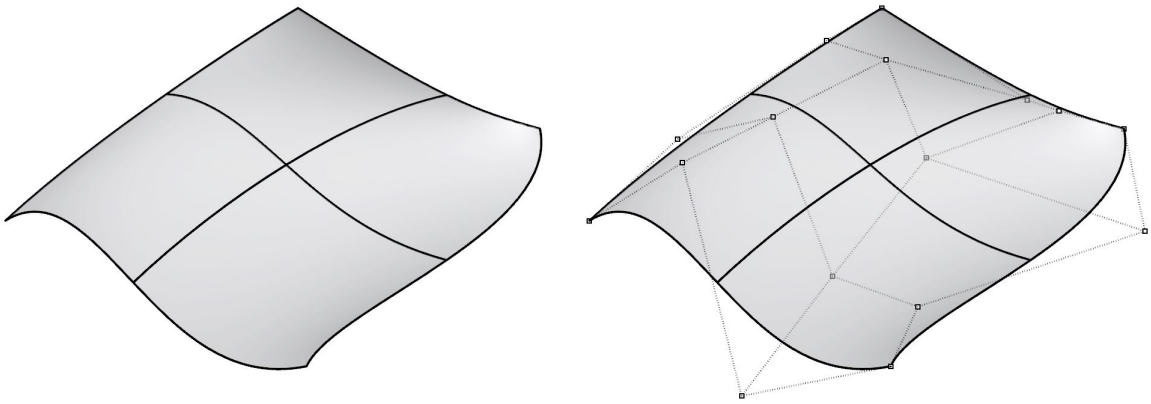


Figure 2.10: NURBS surface and control points.

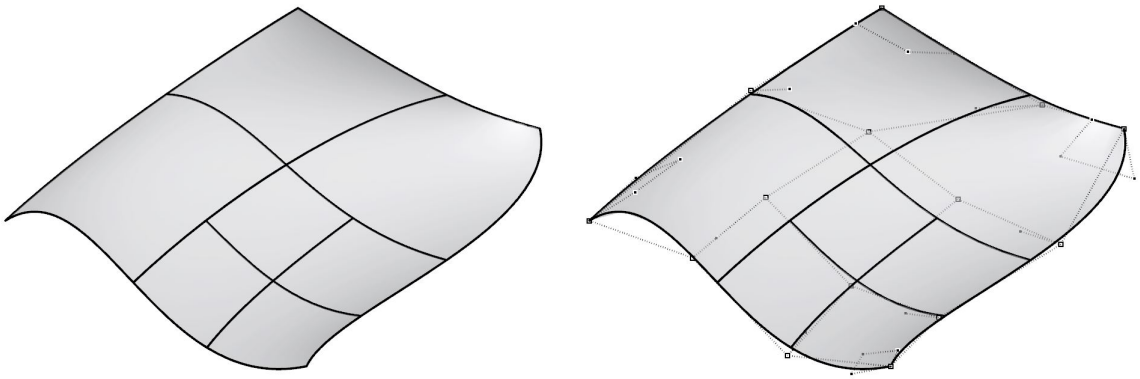
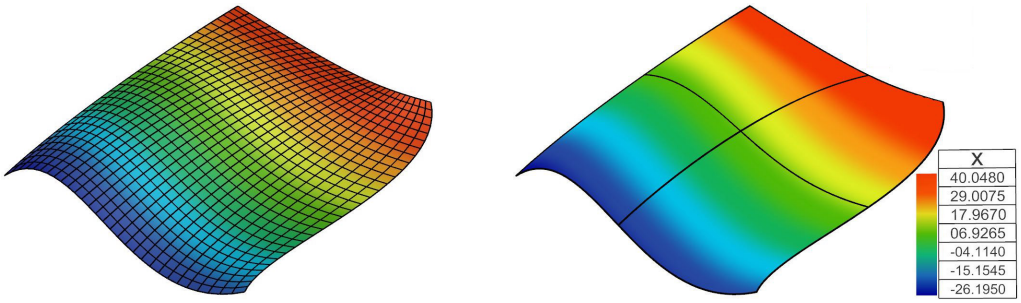


Figure 2.11: T-spline surface and control points.



(a) Visualization mesh. (b) Color contour for NURBS.

Figure 2.12: Visualization of color contour and wireframe for NURBS.

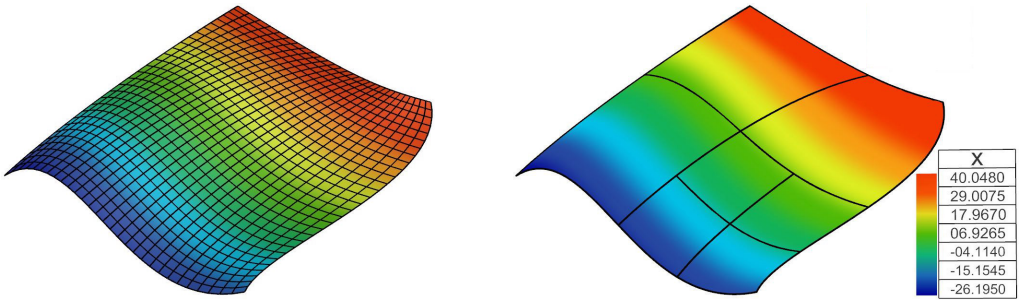
2.1.4.2 Visualization mesh and IGA results

Currently, Rhino 3D does not provide color contours directly visualized on NURBS or T-spline surface. Therefore, a visualization mesh from the NURBS or T-spline surface is generated for the purpose of visualizing color contours. This is shown as the top left group of components named "Visualization Mesh" in Figure 2.8. NURBS surface is evenly divided into several segments in both parametric coordinates. A denser mesh could be generated by increasing the number of the segments if needed. Examples of the visualization mesh are shown in Figures 2.12(a) and 2.13(a).

The solution fields of the IGA results are on the control points. These results need to be transferred to the visualization mesh points. This can be done by feeding the coordinates of the mesh points to a component or function that finds the closest points and their parametric coordinates. Once the parametric coordinates are located, the solution values can be evaluated at these locations, which correspond to the visualization mesh points. This data are extracted and mapped to a linear color gradient which is based on a range between 0 and 1. The group of components named "IGA Result" in the middle of Figure 2.8 is used for this procedure.

2.1.4.3 Wireframe and color contour surface

In Figure 2.8, the group of components named "Wireframe" is for extracting the wireframe from NURBS or T-spline surface. The wireframe density is set to 1.0, which will display one wireframe curve on each knot. It should be noted that for cases without interior knot values (e.g., only one element), one wireframe curve is displayed in the interior by default. (This



(a) Visualization mesh. (b) Color contour for T-splines.

Figure 2.13: Visualization of color contour and wireframe for T-splines.

explains the extra mesh lines in Figure 2.10.) The final visualization result is constructed in the rightmost group of components in Figure 2.8. The color contour surface from the visualization mesh is overlapped with the wireframe extracted from the original NURBS or T-spline surface as shown in Figures 2.12(b) and 2.13(b), respectively.

2.2 Example: Wind Turbine Blade

In this section, we apply the proposed IGA design-through-analysis platform to the modeling of wind turbine blades. In what follows, we make use of the rotation-free Kirchhoff–Love thin shell formulation from Kiendl et al. [65–67] to model the blade structural mechanics.

2.2.1 T-spline model

The parametric design and geometry modeling of the NREL Phase VI wind turbine blade has been discussed in detailed in Section 2.1.3 and a multi-patch NURBS surface was generated. To have better modeling features, such as local refinement and coarsening, the NURBS surface is converted to a single T-spline surface using the Autodesk T-Splines Plug-in for Rhino [5, 105]. Figure 2.14 shows the T-spline surface of the wind turbine blade to which local refinement has already been added, and from which unwanted knots have been removed.

2.2.2 Setting material properties, loads, and boundary conditions

The next logical step towards analysis is to define the material properties, loads, and boundary conditions. The user interface for setting these properties and conditions depends on the selection and assignment of T-spline surface elements. By using T-spline "TsSelSet" [105] com-

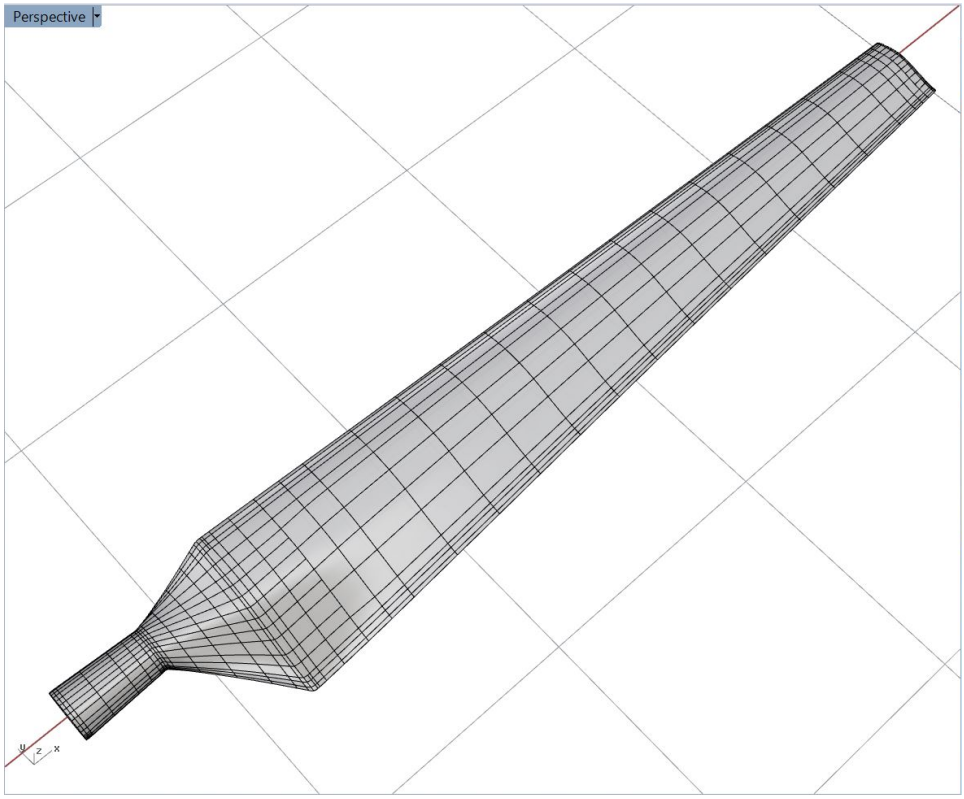


Figure 2.14: T-spline surface of the NREL Phase VI wind turbine blade.

mand in Figure 2.3(b), one can select groups of elements and define a different set for each group, as shown in Figure 2.15. After setting up these element zones, the material properties and blade thickness can be entered and assigned to each zone by using the "Material setup" command in Figure 2.3(c). The material we use for demonstration purposes is aluminum, which is isotropic and has Young's modulus of 70 GPa and Poisson's ratio of 0.35. The blade is assumed to have eight regions of constant thickness, which decreases from root to tip. Finally, we apply the clamped boundary condition at the root by selecting two rows of control points, and select four different pressure load areas on the pressure side of the blade surface, as shown in Figure 2.16.

2.2.3 Simulation results

The pressure load of 45 kPa is applied on the selected zones of the blade surface as shown in Figure 2.16. The resultant force due to the pressure load is 2.315 kN. The blade is clamped at the root and, in addition, loaded by gravity. Dynamic simulation is employed with a time-step

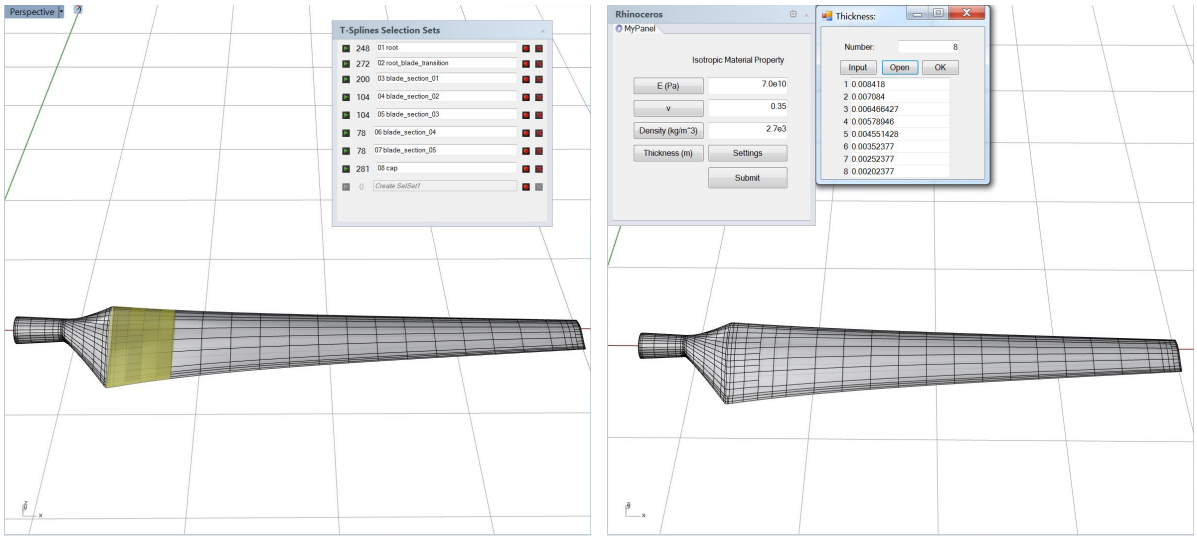


Figure 2.15: Left: Select and assign elements to different sets using "TsSelSet" command. Right: Each set can be assigned different material property using the in-house developed user interface.

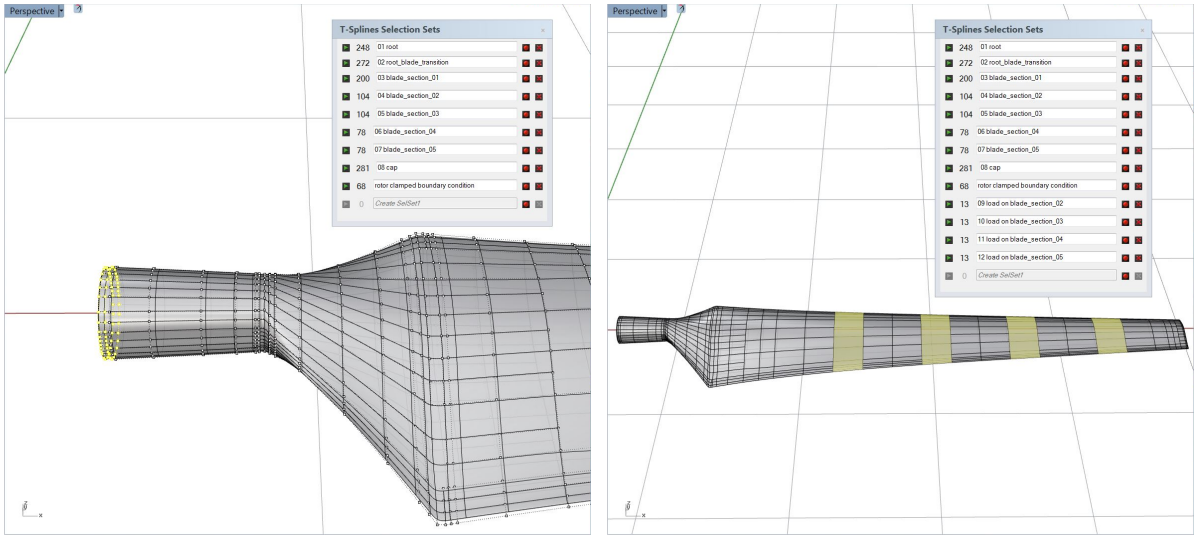


Figure 2.16: Left: Select control points to set clamped boundary condition. Right: Select elements to assign pressure loads on several upper-surface areas.

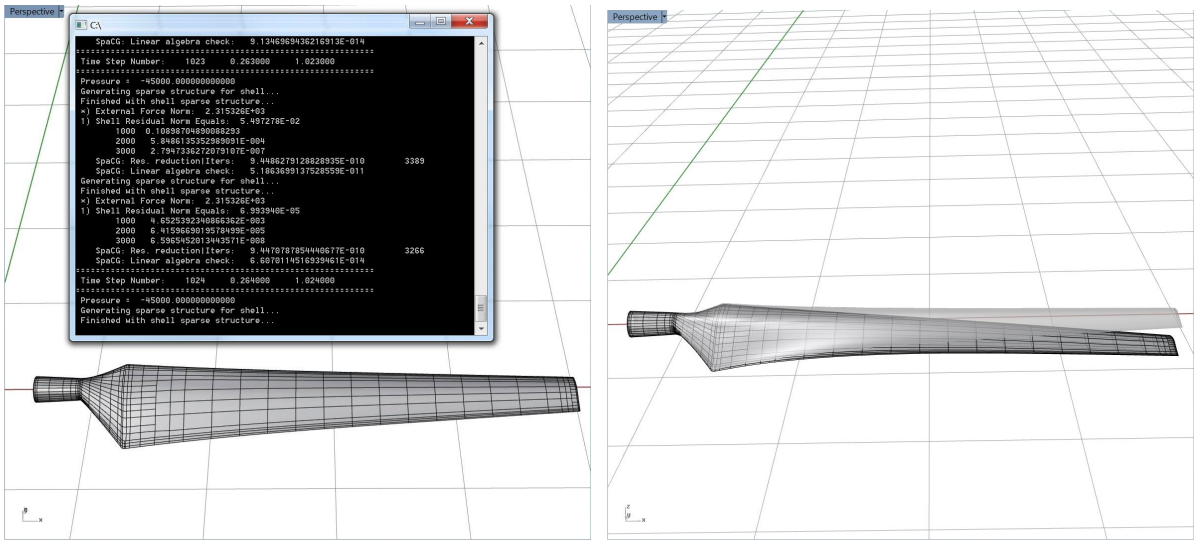


Figure 2.17: Left: Isogeometric shell analysis. Right: Deformed T-spline surface.

size of 0.001 s. The isogeometric Kirchhoff–Love shell solver is called by the "IGA Shell Solver" command shown in Figure 2.3(d). The analysis results are shown in Figure 2.17. The deformed T-spline surface is visualized by adding the displacement field to the control point coordinates. The command used to perform this function is "Deformed shape" shown in Figure 2.3(e).

2.2.4 Visualization of IGA results

To have a deeper understanding of the analysis results, one may perform post-processing of quantities of interest such as the maximum in-plane principal Green–Lagrange strain (MIPE) from shell displacement by using the "Post-processing" command shown in Figure 2.3(f). The "Visualization" command shown in Figure 2.3(g) can then be executed to visualize the color contours of either displacement magnitude or MIPE on the blade surface for a chosen time step. The steady-state results are shown in Figure 2.18. The higher MIPE area is concentrated around the sharp transition from the cylindrical root to the airfoil cross sections. This analysis result could provide guidance for potential design improvement. Finally, the "Close project" command shown in Figure 2.3(h) is used to close all the plug-ins.

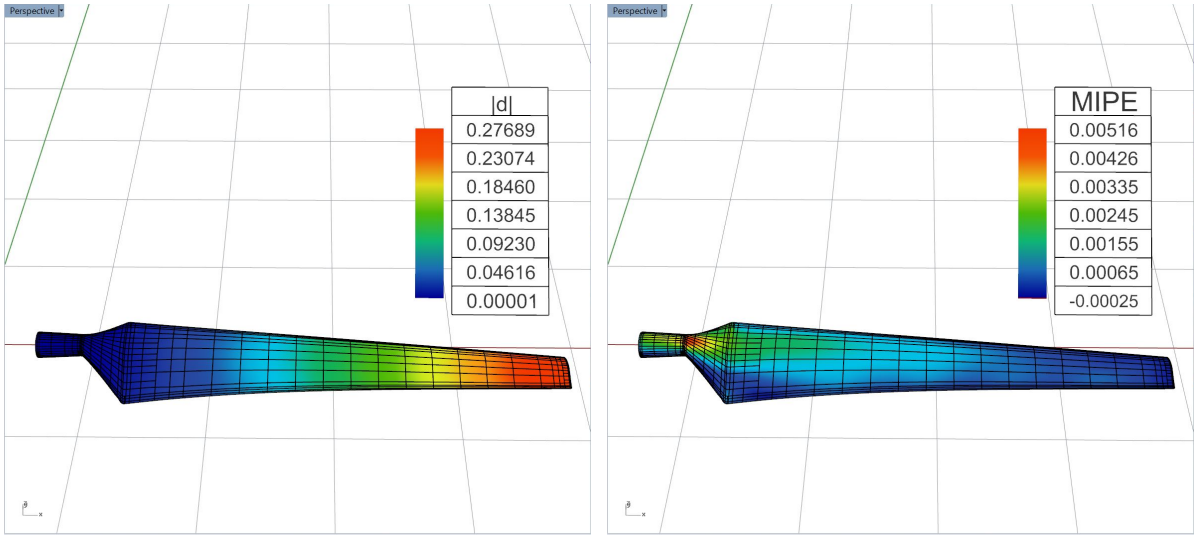
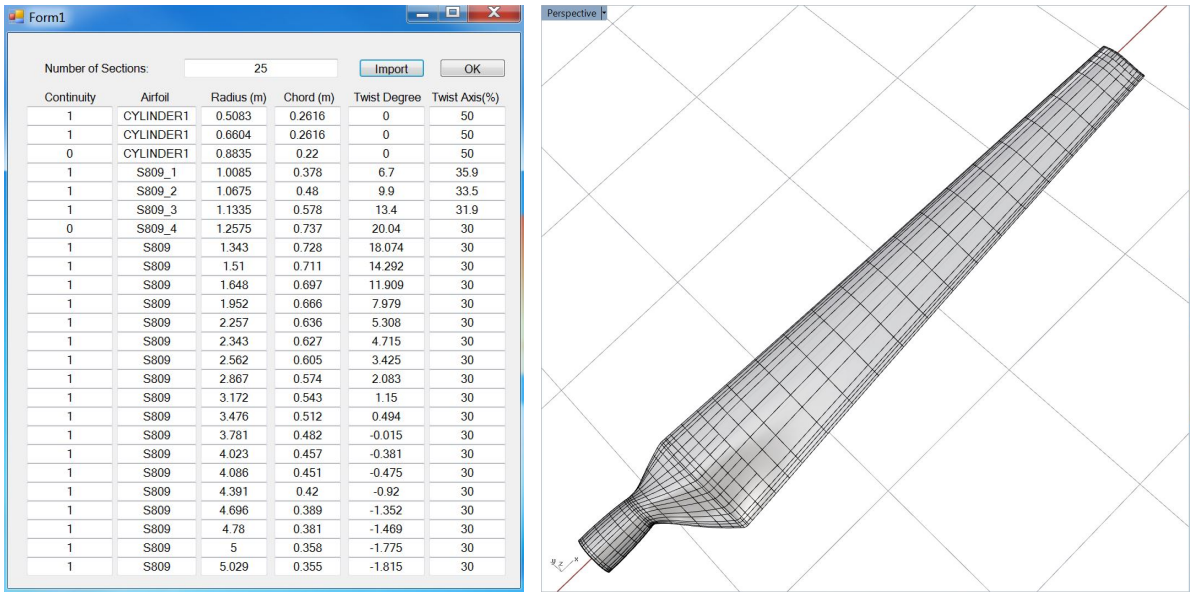


Figure 2.18: Displacement (left) and MIPE (right) contours of the isogeometric shell analysis result.



Continuity	Airfoil	Radius (m)	Chord (m)	Twist Degree	Twist Axis(%)
1	CYLINDER1	0.5083	0.2616	0	50
1	CYLINDER1	0.6604	0.2616	0	50
0	CYLINDER1	0.8835	0.22	0	50
1	S809_1	1.0085	0.378	6.7	35.9
1	S809_2	1.0675	0.48	9.9	33.5
1	S809_3	1.1335	0.578	13.4	31.9
0	S809_4	1.2575	0.737	20.04	30
1	S809	1.343	0.728	18.074	30
1	S809	1.51	0.711	14.292	30
1	S809	1.648	0.697	11.909	30
1	S809	1.952	0.666	7.979	30
1	S809	2.257	0.636	5.308	30
1	S809	2.343	0.627	4.715	30
1	S809	2.562	0.605	3.425	30
1	S809	2.867	0.574	2.083	30
1	S809	3.172	0.543	1.15	30
1	S809	3.476	0.512	0.494	30
1	S809	3.781	0.482	-0.015	30
1	S809	4.023	0.457	-0.381	30
1	S809	4.086	0.451	-0.475	30
1	S809	4.391	0.42	-0.92	30
1	S809	4.696	0.389	-1.352	30
1	S809	4.78	0.381	-1.469	30
1	S809	5	0.358	-1.775	30
1	S809	5.029	0.355	-1.815	30

Figure 2.19: New design parameters and the corresponding modified wind turbine blade geometry.

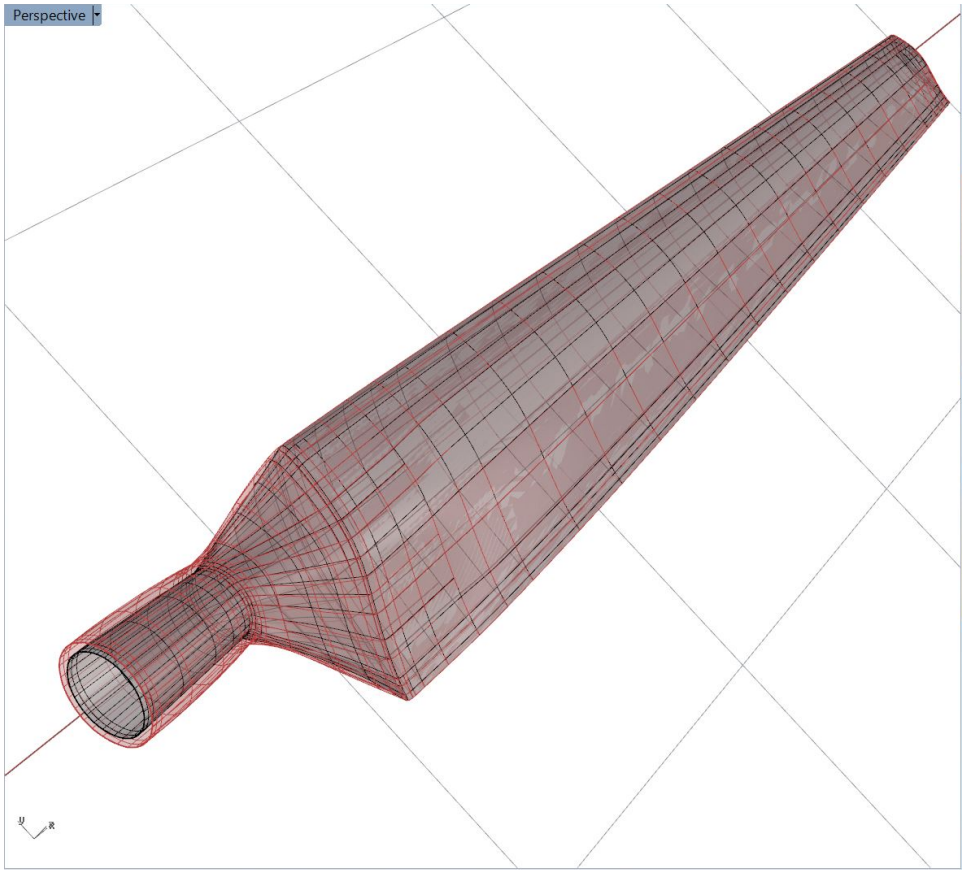


Figure 2.20: The comparison between the original (black wireframe) and modified (red wireframe) geometry.

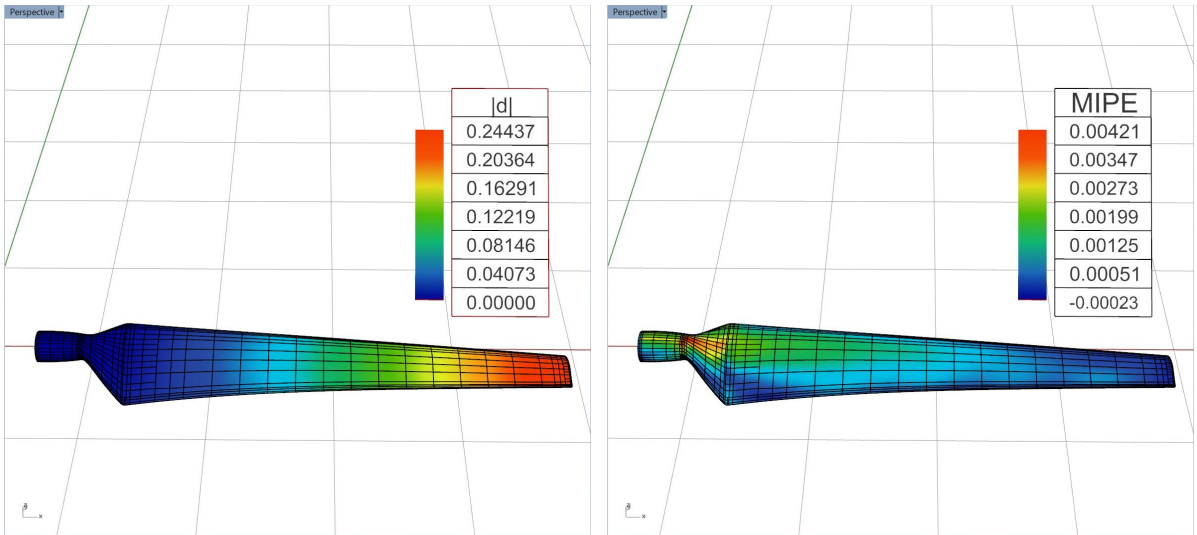


Figure 2.21: Displacement (left) and MIPE (right) contours of the isogeometric shell analysis results of the modified wind turbine blade.

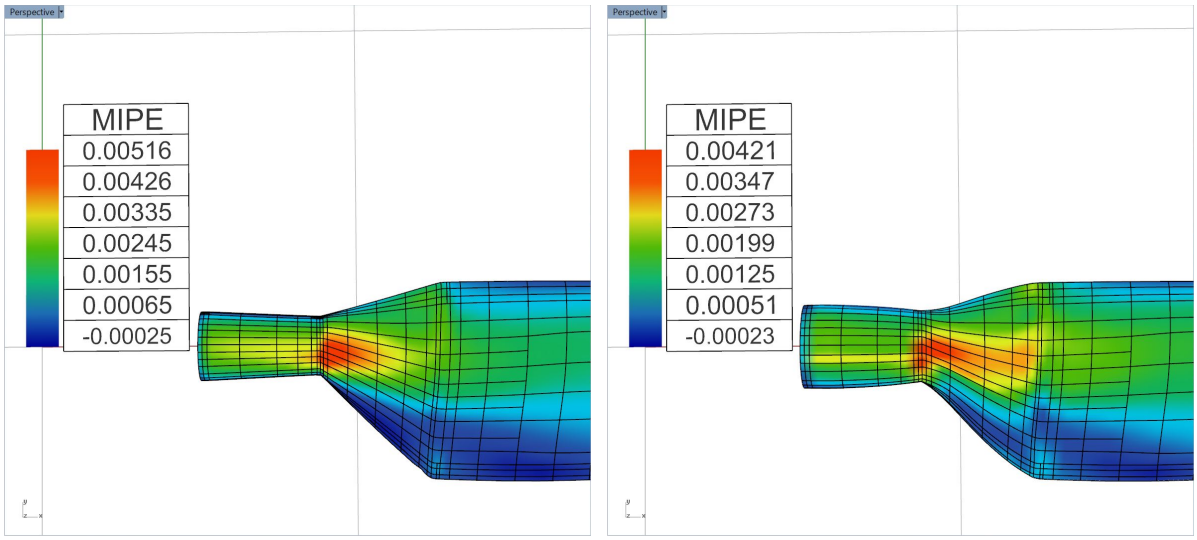


Figure 2.22: The comparison of the MIPE contour distributions of the original (left) and modified (right) designs.

2.2.5 Parametric design modification

The design of the NREL Phase VI wind turbine blade can be easily modified to have a larger root by using the parametric design user interface as shown in Figures 2.19. Figure 2.20 shows an overlapping of the original and modified designs, where the red wireframe represents the new geometry, and the black wireframe represents the original geometry. After following the same platform steps, the new deformation and MIPE results are shown in Figure 2.21. Figure 2.22 shows the comparison of MIPE between the redesigned and original cases. The maximum value of MIPE of the whole blade drops by 18.4% compared to the original design, and the maximum displacement decreases by 11.7%. This illustrates how the structural design improvements may be achieved within the same IGA design-through-analysis platform.

2.3 Acknowledgments

Chapter 2, in part, is a reprint of the material as it appears in: "An interactive geometry modeling and parametric design platform for isogeometric analysis," (with M.-C. Hsu, A.J. Herrema, D. Schillinger and Y. Bazilevs), *Computers & Mathematics with Applications*, 70:1481–1500, 2015. The dissertation author was the primary investigator of this paper.

CHAPTER 3. DIRECT IMMERSOGEOMETRIC FLUID FLOW ANALYSIS USING B-REP MODELS

The idea of immersogeometric analysis [62, 136] is to immerse a design object, such as a B-rep CAD model, directly into a locally refined, non-boundary-fitted background fluid mesh to avoid the challenges associated with geometry cleanup, mesh generation and mesh manipulation. In B-reps, a CAD solid model is represented using the set of faces that make up its boundary surfaces. The geometric descriptions of these faces can be categorized into two types: parametric and analytic surfaces. Generally, parametric surfaces are represented by NURBS surfaces. Analytic surfaces—which include planes, cones, spheres, and tori—are described using algebraic equations. In this chapter, we present the development of a geometric preprocessing framework for immersogeometric analysis and apply it to the simulation of flow around analytic and NURBS-based B-rep CAD models.

3.1 Immersogeometric Analysis

The key features of immersogeometric fluid flow analysis include a variational multiscale (VMS) formulation of incompressible flow [8, 52, 53, 55], the weakly enforced Dirichlet boundary conditions on the surface of the immersed objects [12, 136], and the adaptively refined quadrature rules for faithfully capturing the geometry in intersected background elements. The last is critical for accurate immersogeometric fluid flow analysis, as shown in Xu et al. [136].

3.1.1 Variational multiscale formulation

Let Ω (subsets of $\mathbb{R}^d, d \in \{2, 3\}$) denote the spatial domain and Γ be its boundary. Consider a collection of disjoint elements $\{\Omega^e\}$, $\cup_e \Omega^e \subset \mathbb{R}^d$, with closures covering the fluid domain: $\Omega \subset \cup_e \overline{\Omega^e}$. Note that Ω^e is not necessarily a subset of Ω. Let $\mathcal{V}_u^h$ and $\mathcal{V}_p^h$ be the discrete velocity and pressure spaces of functions supported on these elements. The VMS discretization

of the Navier–Stokes equations of incompressible flows is stated as: Find fluid velocity $\mathbf{u}^h \in \mathcal{V}_u^h$ and pressure $p^h \in \mathcal{V}_p^h$ such that for all test functions $\mathbf{w}^h \in \mathcal{V}_u^h$ and $q^h \in \mathcal{V}_p^h$:

$$B^{\mathrm{VMS}}\left(\{\mathbf{w}^h, q^h\}, \{\mathbf{u}^h, p^h\}\right) - F^{\mathrm{VMS}}\left(\{\mathbf{w}^h, q^h\}\right) = 0 \ , \tag{3.1}$$

where

$$\begin{aligned} B^{\mathrm{VMS}}\left(\{\mathbf{w}^h, q^h\}, \{\mathbf{u}^h, p^h\}\right) = & \int_\Omega \mathbf{w}^h \cdot \rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h\right) d\Omega \\ & + \int_\Omega \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}\left(\mathbf{u}^h, p^h\right) d\Omega + \int_\Omega q^h \, \boldsymbol{\nabla} \cdot \mathbf{u}^h d\Omega \\ & - \sum_e \int_{\Omega^e \cap \Omega} \left(\mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h + \frac{\boldsymbol{\nabla} q^h}{\rho}\right) \cdot \mathbf{u}' d\Omega \\ & - \sum_e \int_{\Omega^e \cap \Omega} p' \, \boldsymbol{\nabla} \cdot \mathbf{w}^h d\Omega \\ & + \sum_e \int_{\Omega^e \cap \Omega} \mathbf{w}^h \cdot (\mathbf{u}' \cdot \boldsymbol{\nabla} \mathbf{u}^h) d\Omega \\ & - \sum_e \int_{\Omega^e \cap \Omega} \frac{\boldsymbol{\nabla} \mathbf{w}^h}{\rho} : \left(\mathbf{u}' \otimes \mathbf{u}'\right) d\Omega \\ & + \sum_e \int_{\Omega^e \cap \Omega} \left(\mathbf{u}' \cdot \boldsymbol{\nabla} \mathbf{w}^h\right) \overline{\tau} \cdot \left(\mathbf{u}' \cdot \boldsymbol{\nabla} \mathbf{u}^h\right) d\Omega \ , \end{aligned} \tag{3.2}$$

and

$$F^{\mathrm{VMS}}\left(\{\mathbf{w}^h, q^h\}\right) = \int_\Omega \mathbf{w}^h \cdot \rho \, \mathbf{f} d\Omega + \int_{\Gamma^N} \mathbf{w}^h \cdot \mathbf{h} d\Gamma \ . \tag{3.3}$$

In (3.2), $\mathbf{u}'$ is defined as

$$\mathbf{u}' = -\tau_{\mathrm{M}} \left(\rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h - \mathbf{f}\right) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}\left(\mathbf{u}^h, p^h\right)\right) \tag{3.4}$$

and p' is given by

$$p' = -\rho \, \tau_{\mathrm{C}} \, \boldsymbol{\nabla} \cdot \mathbf{u}^h \ . \tag{3.5}$$

In the above equations, ρ is the density of the fluid, $\mathbf{f}$ is the external force per unit mass, $\boldsymbol{\sigma}$ and $\boldsymbol{\varepsilon}$ are the stress and strain-rate tensors, respectively, and $\mathbf{h}$ is the traction vector at the Neumann boundary Γ^N. The terms integrated over element interiors may be interpreted both as stabilization and as a turbulence model [8, 23, 48, 53, 124, 127]. $\tau_{\mathrm{M}}, \tau_{\mathrm{C}}$ and $\overline{\tau}$ are the stabilization parameters. Their detailed expression used in this work are

$$\tau_{\mathrm{M}} = \left(\frac{C_t}{\Delta t^2} + \mathbf{u} \cdot \mathbf{G} \, \mathbf{u} + C_I \, \nu^2 \, \mathbf{G} : \mathbf{G}\right)^{-1/2} \ , \tag{3.6}$$

$$\tau_{\mathrm{C}} = \left(\tau_{\mathrm{M}} \operatorname{tr} \mathbf{G}\right)^{-1} \ , \tag{3.7}$$

$$\overline{\tau} = \left(\mathbf{u}' \cdot \mathbf{G}\, \mathbf{u}'\right)^{-1/2} \ , \tag{3.8}$$

where Δt is the time-step size, C_I is a positive constant derived from an appropriate element-wise inverse estimate [22, 33, 60], $\nu = \mu/\rho$ is the kinematic viscosity, $\mathbf{G}$ generalizes the notion of element size to physical elements mapped from a parametric parent element by $\mathbf{x}(\xi)$:

$$G_{ij} = \sum_{k=1}^{d} \frac{\partial \xi_k}{\partial x_i} \frac{\partial \xi_k}{\partial x_j} \ , \tag{3.9}$$

$\operatorname{tr} \mathbf{G}$ is the trace of $\mathbf{G}$, and the parameter C_t is typically equal to 4 [8, 127].

3.1.2 Variationally consistent weak boundary conditions

The standard way of imposing Dirichlet boundary conditions in Eq. (3.1) is to enforce them strongly by ensuring that they are satisfied by all trial solution functions. This is not feasible in immersed methods. Instead, the strong enforcement is replaced by weakly enforced Dirichlet boundary conditions proposed by Bazilevs et al. [12–14]. The semi-discrete problem becomes

$$\begin{aligned}
&B^{\mathrm{VMS}}\left(\{\mathbf{w}^h, q^h\}, \{\mathbf{u}^h, p^h\}\right) - F^{\mathrm{VMS}}\left(\{\mathbf{w}^h, q^h\}\right) \\
&- \int_{\Gamma^D} \mathbf{w}^h \cdot \left(-p^h\, \mathbf{n} + 2\mu\, \boldsymbol{\varepsilon}(\mathbf{u}^h)\, \mathbf{n}\right) d\Gamma \\
&- \int_{\Gamma^D} \left(2\mu\, \boldsymbol{\varepsilon}(\mathbf{w}^h)\, \mathbf{n} + q^h\, \mathbf{n}\right) \cdot \left(\mathbf{u}^h - \mathbf{g}\right) d\Gamma \\
&- \int_{\Gamma^{D,-}} \mathbf{w}^h \cdot \rho \left(\mathbf{u}^h \cdot \mathbf{n}\right) \left(\mathbf{u}^h - \mathbf{g}\right) d\Gamma \\
&+ \int_{\Gamma^D} \tau^B_{\mathrm{TAN}} \left(\mathbf{w}^h - \left(\mathbf{w}^h \cdot \mathbf{n}\right) \mathbf{n}\right) \cdot \left(\left(\mathbf{u}^h - \mathbf{g}\right) - \left(\left(\mathbf{u}^h - \mathbf{g}\right) \cdot \mathbf{n}\right) \mathbf{n}\right) d\Gamma \\
&+ \int_{\Gamma^D} \tau^B_{\mathrm{NOR}} \left(\mathbf{w}^h \cdot \mathbf{n}\right) \left(\left(\mathbf{u}^h - \mathbf{g}\right) \cdot \mathbf{n}\right) d\Gamma \ = \ 0 \ ,
\end{aligned} \tag{3.10}$$

where Γ^D is the Dirichlet boundary that may cut through element interiors, $\Gamma^{D,-}$ is the *inflow* part of Γ^D, on which $\mathbf{u}^h \cdot \mathbf{n} < 0$, $\mathbf{g}$ is the prescribed velocity on Γ^D, τ^B_{TAN} and τ^B_{NOR} are stabilization parameters that need to be chosen element-wise as a compromise between the conditioning of the stiffness matrix, variational consistency, and the stability of the formulation.

For immersogeometric methods, weakly enforced boundary conditions are particularly attractive as the additional Nitsche terms (the third to last terms on the left-hand side of

Eq. (3.10)) are formulated independently of the mesh. In contrast to strong enforcement, which relies on boundary-fitted meshes to impose Dirichlet boundary conditions on the discrete solution space, the Nitsche terms also hold for intersected elements, where the domain boundary does not coincide with element boundaries. All that is needed is a separate discretization of the domain boundary with quadrature rules whose positions of the quadrature points in intersected elements is known or can be determined. In Xu et al. [136], the geometry of the object was described by the stereolithography (STL) format, which uses polygons (mostly triangles) to discretize (tessellate) the object surface. However, modern CAD models are typically described using B-reps, and the conversion from B-reps to STL is not trivial, especially when the geometry is not "watertight". In this work, we tackle this issue by performing the surface integration of the weak boundary conditions directly using B-rep model information. This novel approach eliminates the need for a different discretization of the object surface and allows use of the actual CAD model directly for the purpose of immersogeometric analysis. This approach also shares the same philosophy with isogeometric analysis [27, 51]—bridging the gap between design and analysis.

Another advantage of weakly enforced Dirichlet boundary conditions is the release of the point-wise no-slip condition at the boundary of the fluid domain. Although maybe counterintuitive at first sight, some violation of the no-slip boundary condition is in fact desirable, as it allows the flow to slip 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. [14] and Hsu et al. [46] that weak boundary conditions allow for an accurate overall flow solution even if the mesh size in the wall-normal direction is relatively large. Weak enforcement of Dirichlet boundary conditions also provides special benefits in turbulent flow simulation [13, 14]. 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. [136] that accurate laminar and turbulent 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.

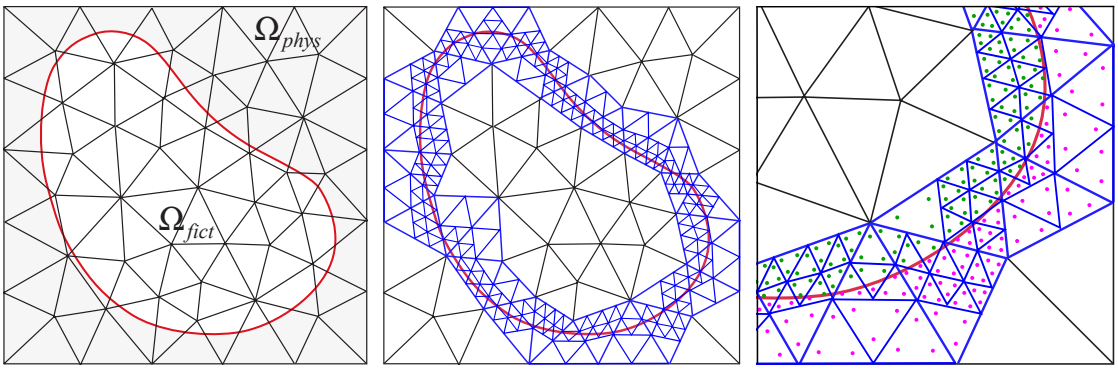


Figure 3.1: The concept of physical and fictitious domains and quadrature scheme based on adaptive sub-cells (blue lines). Quadrature points within the fluid domain (marked in pink) are used in the numerical integration. Quadrature points outside (marked in green) are discarded.

3.1.3 Sub-cell-based adaptive quadrature

The immersogeometric method introduces elements that are intersected by the geometric boundary, which creates complex, discontinuous integration domains in intersected elements. To ensure geometrically accurate evaluation of volume integrals in intersected elements, we use a sub-cell-based adaptive quadrature scheme [32, 136]. The basic concept is to increase the number of quadrature points around immersed geometric boundaries so that arbitrary integration domains resulting from the intersecting boundary can be taken into account accurately. This is achieved by recursively splitting intersected cells into sub-elements. At each level, only those sub-elements intersected by the boundary are further split. For clarity, we illustrate the quadrature scheme based on adaptive sub-cells for triangles in 2D in Figure 3.1. We emphasize that splitting is performed on the quadrature level only and does not affect the basis functions, which are still defined on the original tetrahedral element.

3.1.4 Time integration and solution strategies

We complete the discretization of Eq. (3.10) by a time integration scheme from the family of generalized-α methods [10, 25, 57], which is a fully-implicit, second-order accurate method with control over the dissipation of high-frequency modes. At each time step, the solution of a nonlinear algebraic problem is required. We directly apply Newton–Raphson iterations (with an approximate tangent) to converge the residual of this algebraic problem. For each Newton–Raphson iteration, the linear system is solved using a block-diagonal preconditioned

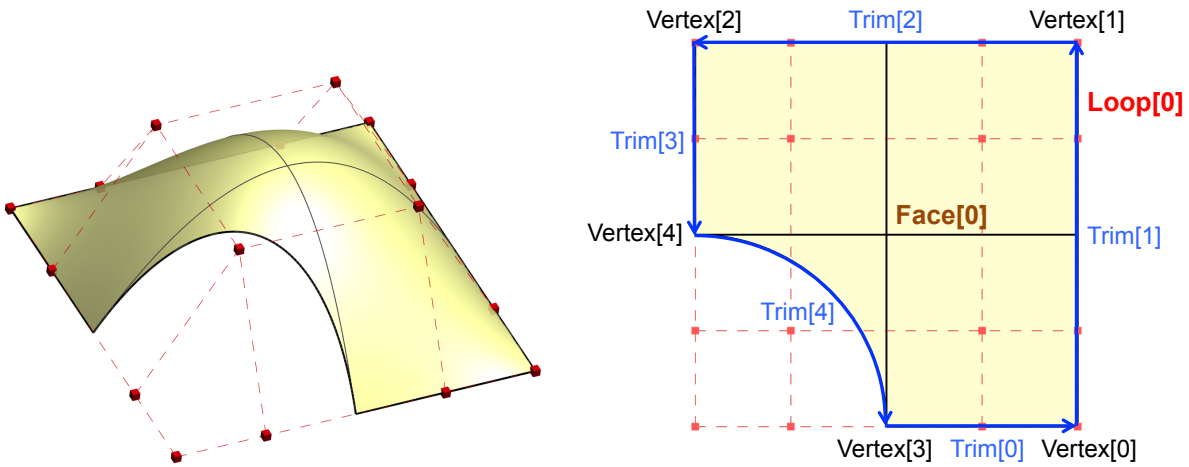
GMRES method [96, 110]. The computations reported in this work are carried out in a parallel computing environment on Linux clusters. A description of our parallelization strategy can be found in Hsu et al. [45] and a strong linear scaling of the immersogeometric method was shown in Xu et al. [136]. The mesh is partitioned into subdomains using METIS [63], and each subdomain is assigned to a processor core.

3.2 Rapid B-rep Model Preprocessing for Immersogeometric Analysis

As mentioned earlier, a B-rep solid model is represented using the set of faces that make up its boundary surfaces. The geometric descriptions of these faces can be categorized into parametric and analytic surfaces. Generally, parametric surfaces are represented by NURBS and analytic surfaces, which include planes, cones, spheres, and tori, are described using algebraic equations. Note that a cylindrical surface is considered a special case of a conical surface. In addition, the B-rep data also includes information regarding the topology of the CAD model: the connectivity between the faces. This information is used to create watertight, 2-manifold, tessellations of the CAD model.

Using traditional NURBS surfaces to represent B-rep faces restricts them to topologically rectangular sheets; they are not very flexible, especially when it comes to representing surfaces that are not rectangular or those with holes or complex local geometries that arise due to Boolean operations. As a result, the NURBS patches are typically trimmed, discarding a portion of the surface defined in the parametric domain. An example of a trimmed NURBS surface in a CAD model is shown in Figure 3.2(a). The trimming information is defined in the 2D parametric domain of the surface (Figure 3.2(b)). Typically, trim curves are represented as directed closed loops; the direction of the loop determines which side of the trim curve to cut away. There can also be multiple loops per surface, one defining the boundary and others defining interior holes, or even holes within holes. In conventional CAD B-reps, there is at least one trim curve that bounds the valid surface region for every surface (even if it is the trivial outer boundary) in order to have a consistent representation.

In reality, CAD models of synthetic objects usually have many flat features with rounded corners. These are represented internally in the CAD system using analytic surfaces rather than



(a) Trimmed NURBS surface.

(b) Parametric trim curves.

Figure 3.2: Example surface represented using trimmed NURBS. The trim curves are defined in the parametric space and, depending on the orientation of the curves, part of the NURBS surface is trimmed

NURBS surfaces, since they can be easily manipulated by the solid modeling kernel and can be stored in a compact manner. Converting these features to NURBS usually leads to poorly parametrized NURBS surfaces and can lead to poorly trimmed or missing surface features. In addition, converting simple geometries such as cylinders to NURBS imposes a performance penalty since these geometries have to be dealt with as rational splines. As a result, the geometry has to be inspected again after conversion to ensure analysis compatibility and can increase the computational cost.

In this dissertation, we extend the immersogeometric method from requiring tessellation of the object surface [136] to generating surface quadrature information directly using parametric and analytic surfaces. We develop quadrature rules for parametric surface and all four kinds of analytic surfaces: planes, cones, spheres, and tori. We also develop methods for performing adaptive quadrature on trimmed parametric and analytic surfaces.

3.2.1 Surface quadrature in trimmed NURBS patches

The Dirichlet boundary conditions on the surfaces are imposed weakly in the immersogeometric method. This method requires the evaluation of the surface integrals on the object surfaces. Practically, these weak boundary conditions are evaluated at the surface Gaussian

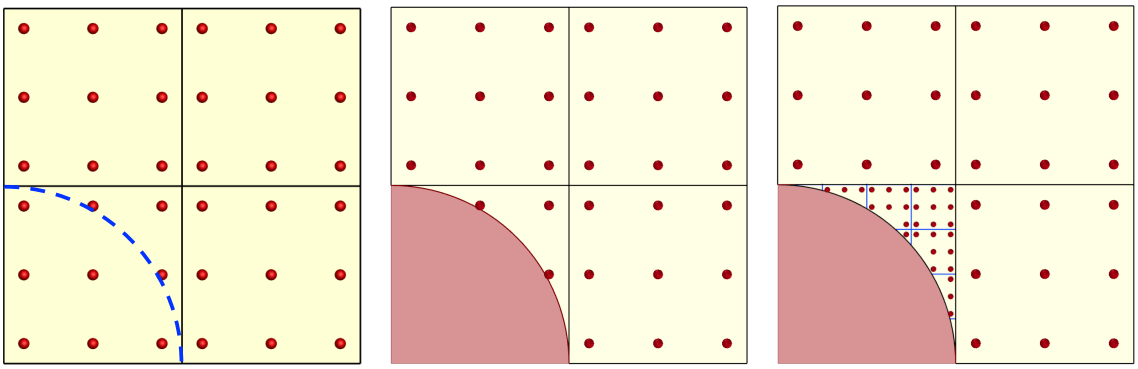
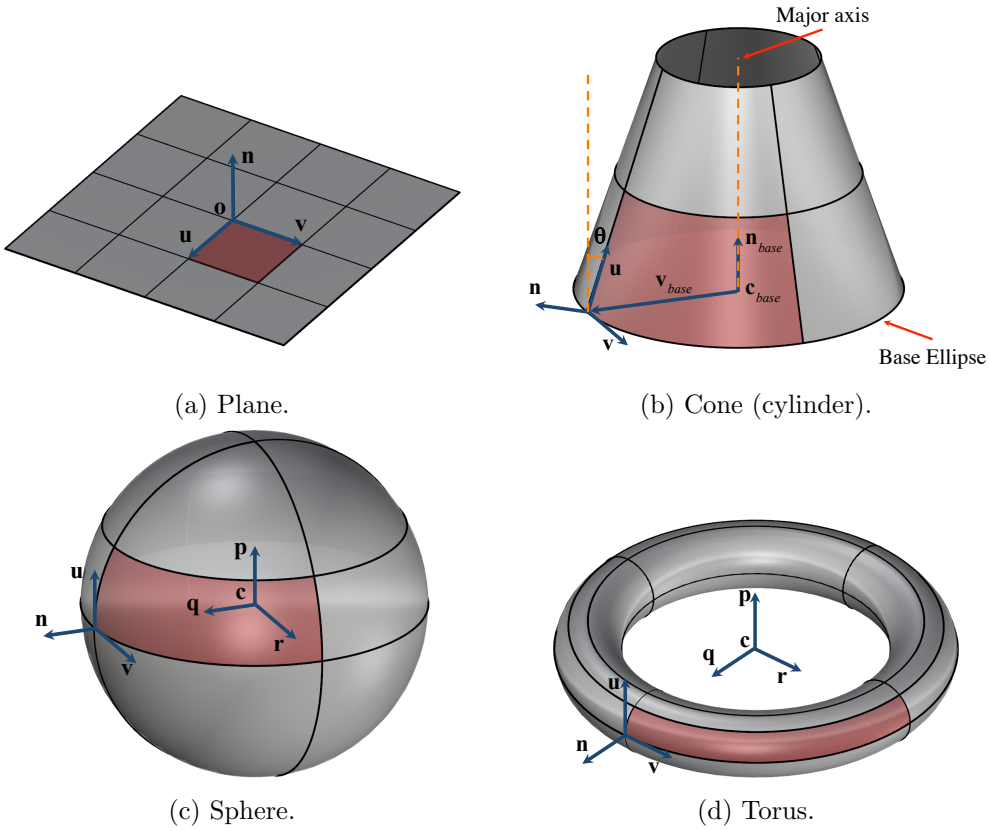


Figure 3.3: Sub-cell-based adaptive quadrature is used to more accurately evaluate the integrals near the trim curves. In each level, the Gaussian quadrature points (shown in red) that lie outside the trim curves are discarded. The Gaussian quadrature points after two levels of adaptive quadrature sub-cell refinement are shown on the right.

quadrature points. For untrimmed NURBS surfaces, the parameterization of an untrimmed NURBS surface is based on the tensor product of the knot locations along the u and v parametric directions. The tensor product parametric domain is usually divided at the knot locations to generate separate elements for the surface quadrature for each knot span. However, in the presence of trimmed NURBS surfaces, the surface quadrature points need to be checked if they lie inside the trim curves of the surface. In addition, for accurate evaluation of the function near the trim curve, the surface needs to be adaptively sampled around the trim curve.

The adaptive quadrature for trimmed NURBS is implemented by firstly dividing the base untrimmed NURBS surface patches into quadrature elements for each knot span. The Gaussian quadrature points for each quadrature element are generated in the parametric space. The parametric location of each Gauss point is then tested using the trim curves to classify them as inside or outside the trimmed-out section of the quadrature element. Those quadrature elements that have some Gauss points outside the trimmed-out section (inside the trimmed surface) are recursively refined using the sub-cell approach, and the process is repeated with new Gauss quadrature points generated inside each refined quadrature element. This approach is illustrated in Figure 3.3.



(a) Plane.

(b) Cone (cylinder).

(c) Sphere.

(d) Torus.

Figure 3.4: Analytic surfaces: plane, cone (cylinder), sphere, and torus. A sub-cell is red shaded for each analytic surface.

3.2.2 Surface quadrature in trimmed analytic surfaces

The algebraic equations for different types of analytic surfaces represent the mappings of a region of the uv parametric plane into the Euclidean three-dimensional space. An arbitrary 3D point of an analytic surface can be represented by a combination of the two parameters u and v in the algebraic equation. The parametric domain is divided into quadrature elements for surface integrals.

3.2.2.1 Planar surface

For an untrimmed planar surface, the algebraic equation is

$$\mathbf{S}(u, v) = \mathbf{o} + u\ \mathbf{u} + v\ \mathbf{v}\ , \tag{3.11}$$

where $\mathbf{o}$ is the origin of the plane and $\mathbf{u}$ and $\mathbf{v}$ are the basis vectors in a right-handed coordinate system $(\mathbf{u}, \mathbf{v}, \mathbf{n})$. $\mathbf{n}$ is the normal vector of this planar surface and is defined as the cross product

of the two basis vectors, $\mathbf{u} \times \mathbf{v}$. For surface integrals, a sub-cell of a planar surface is based on the subsets of the real number domains along both u and v directions. A typical sub-cell of the planar surface is shown in Figure 3.4(a).

3.2.2.2 Conical and cylindrical surfaces

For an untrimmed conical surface, the algebraic equation is

$$\mathbf{S}(u, v) = \mathbf{c}_{\text{base}} + (1 + u \ \sin\theta/R) \ \mathbf{v}_{\text{base}}(v) + u \ \cos\theta \ \mathbf{n}_{\text{base}} \ , \tag{3.12}$$

where $\mathbf{c}_{\text{base}}$ is the base ellipse center, $\mathbf{n}_{\text{base}}$ is the unit vector in the normal direction of the base ellipse, R is the half length of the major axis of the base ellipse, and $\mathbf{v}_{\text{base}}(v)$ is the vector from the base ellipse center to the base ellipse point at a given value of v. Let $\mathbf{u}$ be the unit vector along the generator of the cone with parameter u increasing in the direction of $\mathbf{n}_{\text{base}}$. The latitude metric u has a singular value if there is an apex. Let $\mathbf{v}$ be the unit vector along the base ellipse according to the right-hand rule around $\mathbf{n}_{\text{base}}$. The longitude metric v has a domain $[-\pi, \pi)$, which is periodic. The normal vector $\mathbf{n}$ of this conical surface is defined as $-\mathbf{u} \times \mathbf{v}$. θ has the magnitude of the half angle of the cone. $\theta > 0$ if $\mathbf{u} \cdot \mathbf{v}_{\text{base}} > 0$ and $\theta < 0$ if $\mathbf{u} \cdot \mathbf{v}_{\text{base}} < 0$. θ has a domain between $[-\pi/2, \pi/2)$. Note that the conical surface becomes a cylinder when $\theta = 0$ and a planar surface when $\theta = -\pi/2$. A typical sub-cell of a conical surface is shown in Figure 3.4(b).

3.2.2.3 Spherical surface

For an untrimmed spherical surface, the algebraic equation is

$$\mathbf{S}(u, v) = \mathbf{c} + r \sin u \ \mathbf{p} + r \cos u(\cos v \ \mathbf{q} + \sin v \ \mathbf{r}) \ , \tag{3.13}$$

where $\mathbf{c}$ is the sphere center, r is the sphere radius, $\mathbf{p}$ is the unit vector in the direction from the south pole to the north pole, $\mathbf{q}$ is the unit vector in the direction from the center to the origin of the parametric space, and $\mathbf{r} = \mathbf{p} \times \mathbf{q}$. Let $\mathbf{u}$ be the unit vector along the longitude line of the sphere with parameter u defined as the angle between the vector from the center to the test point and the equatorial plane. The latitude metric u increases from the south pole to the north pole and has a domain between $[-\pi/2, \pi/2)$. Let $\mathbf{v}$ be the unit vector along the

latitude line of the sphere according to the right-hand rule around $\mathbf{p}$. The longitude metric v has a domain between $[-\pi, \pi)$, which is periodic. The normal vector $\mathbf{n}$ of this spherical surface is defined as $-\mathbf{u} \times \mathbf{v}$. A typical sub-cell of a spherical surface is shown in Figure 3.4(c).

3.2.2.4 Toroidal surface

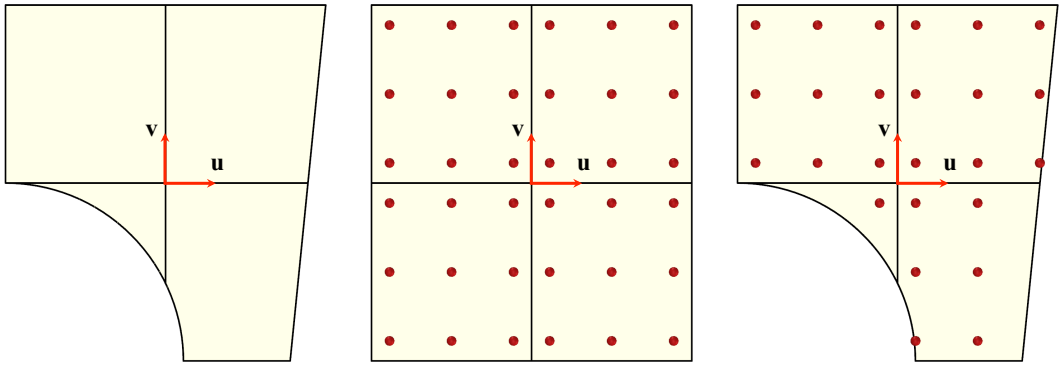
For an untrimmed toroidal surface, the algebraic equation is

$$\mathbf{S}(u, v) = \mathbf{c} + r \sin u \ \mathbf{p} + (R + r \cos u)(\cos v \ \mathbf{q} + \sin v \ \mathbf{r}) \ , \tag{3.14}$$

where $\mathbf{c}$ is the torus center, r is the minor radius, R is the major radius, $\mathbf{p}$ is the unit vector along the torus axis, $\mathbf{q}$ is the unit vector in the direction from the center to the origin of the parametric space, and $\mathbf{r} = \mathbf{p} \times \mathbf{q}$. Let $\mathbf{u}$ be the unit vector along the poloidal direction of the torus with parameter u increasing in the direction of the axis $\mathbf{p}$. $u = 0$ is the circle with the largest radius $(R + r)$ relative to $\mathbf{c}$. Let $\mathbf{v}$ be the unit vector along the toroidal direction of the torus according to the right-hand rule around $\mathbf{p}$. Both the latitude metric u and the longitude metric v have the same domain between $[-\pi, \pi)$, which are periodic. The normal vector $\mathbf{n}$ of this toroidal surface is defined as $-\mathbf{u} \times \mathbf{v}$. A typical sub-cell of a toroidal surface is shown in Figure 3.4(d).

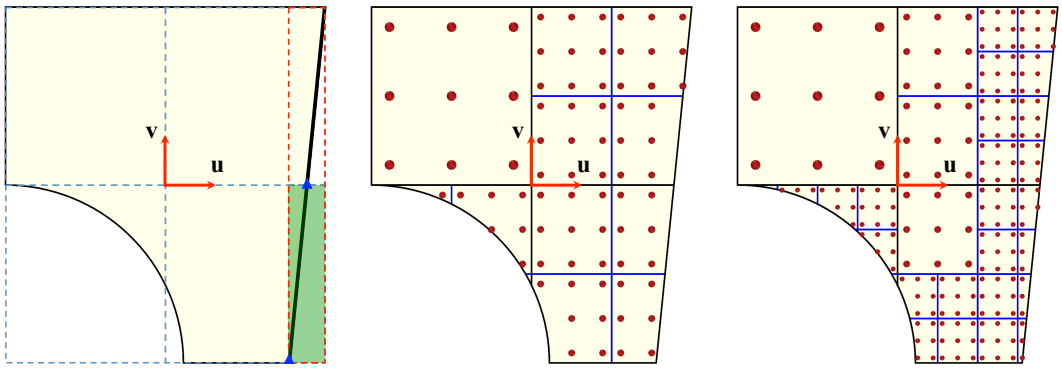
3.2.2.5 Sub-cell-based adaptive quadrature for trimmed analytic surfaces

To evaluate the quadrature points for a trimmed analytic surface, the untrimmed surface patch is evenly divided into quadrature elements along both parametric directions. The number of the quadrature elements is based on the physical Euclidean edge length along each parametric direction. Hence, the largest sub-cell will not exceed a user-defined element size. After the untrimmed domain is divided into quadrature elements, a three-point Gauss–Legendre quadrature rule in each parametric direction is used for the generation of the surface quadrature points in the parametric space for all of these surface quadrature elements. Note that the determinants of Jacobian for the quadrature points are evaluated directly using the ratio of the areas of the sub-cells in the physical and parametric spaces. In addition, patches that include degenerate points (such as the pole of a sphere or a cone vertex) need not be specifically handled, since the quadrature points are generated only on the interior of the analytic surface patches.



(a) Trimmed parametric region of a surface. (b) Untrimmed quadrature elements. (c) Quadrature points inside trimmed region.

Figure 3.5: Standard Gaussian quadrature for a trimmed surface.



(a) Intersected element. (b) Level 1 quadrature. (c) Level 2 quadrature.

Figure 3.6: Sub-cell-based adaptive quadrature is used for a better evaluation of the surface integral of a trimmed surface.

The quadrature points are tested whether they lie inside the trimmed region of the surface. The inside quadrature points are used for the evaluation of the surface integral, while the outside quadrature points are discarded. This implementation of standard three-point Gaussian quadrature rules on a trimmed analytic surface is shown in Figure 3.5.

In order to accurately evaluate the surface integral of a trimmed analytic surface, an adaptive quadrature method is implemented. The quadrature elements that are intersected by a trim curve are identified and recursively refined. We make use of a 2D bounding box in the parametric domain to perform this refinement. First, the bounding box of a trim curve is checked whether it is fully inside or outside of the domain of a sub-cell, which is then used to exclude curve segments that are completely outside. The sub-cell domain that completely

contains the bounding box of a trim curve segment is further refined. However, if the bounding box of the trim curve is not fully inside or outside of a sub-cell, a further test is required to ensure the trim curve is correctly intersected with the boundary of the sub-cell. We check the total number of the intersection points between all the trim curves and sub-cell boundary. If there exists at least two intersection points, the sub-cell is recursively refined.

Figure 3.6(a) shows an example case for finding an intersected quadrature element (marked in green) that needs to be subdivided. First, the bounding-box of the trim-curve (marked in red) is found to be intersecting with the bottom-right sub-cell quadrature element. Next, the number of intersection points between the sub-cell boundary and the trim curves are calculated. In this example, there are two intersection points (marked in blue triangles). Hence, this sub-cell is subdivided into 4 sub-cells. This process is recursively performed until a user-defined subdivision level is reached. Figure 3.6(b) and 3.6(c) show the results of two different levels of recursive subdivision for a trimmed patch.

3.2.3 Implementation on Rhino and SolidWorks

We firstly use Rhinoceros [90] (Rhino) and Grasshopper [41] to efficiently generate the surface quadrature rules directly from B-rep models. As discussed in the previous chapter, a generative algorithm designed in Grasshopper is written in terms of "components" with pre-defined or user-defined functionality and "wire connections" between the components that serve as conduits of input and output data. The large selection of pre-defined components gives users access to complex geometric modeling functionality. A Grasshopper implementation for generating Gaussian quadrature rules on untrimmed and trimmed patches is shown in Figure 3.7. The visual program executes the following steps:

- Component 1 reads in a B-rep model. Note that the orientation of the surfaces should be facing towards the interior of the object (so that the normal vector is pointing outward of the fluid domain).

- Component 2 takes the B-rep surfaces from 1 (`Srf`), builds quadrature elements on each untrimmed patch, adaptively refines quadrature elements near trim curves, and applies

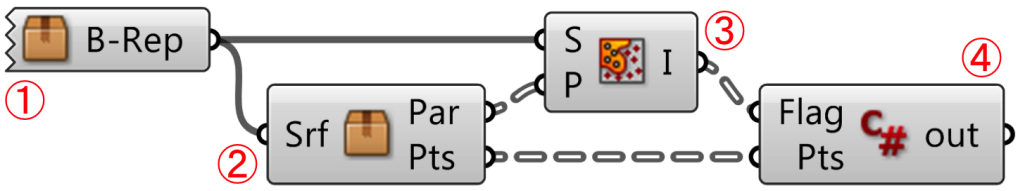


Figure 3.7: The implementation for generating surface quadrature rules directly from B-rep models.

standard Gaussian quadrature rules on each quadrature element. The component then calculates the physical location, Gauss weight, Jacobian determinant of the parameter-to-physical-space mapping, and the normal vector of each Gauss point (`Pts`), and finds the location of each Gaussian quadrature point in the parametric space (`Par`).

- Component 3, which is a Grasshopper built-in function called *Point in Trim*, takes the B-rep data from 1 (`S`) and parametric location of the Gaussian quadrature points from 2 (`P`), and determines whether the points are inside or outside the trim curves using the input B-rep data. A flag is assigned to each point as a result of the inside-outside test. Using this pre-defined component, the users do not need to know the details of the B-rep data structure.

- Component 4 takes the Gaussian quadrature rules from 2 (`Pts`) and the corresponding inside-outside flags from 3, then only outputs the surface quadrature points *inside* the trim curves with the required information for immersogeometric analysis. The output data on each Gauss point includes its physical location (used to locate the point in the background mesh), Gauss weight, Jacobian determinant of the mapping, and the normal vector (pointing towards the interior of the object). This information allows the user to integrate over the trimmed surfaces of the B-rep CAD model.

Another example is demonstrated using SolidWorks [113] and ACIS solid modeling kernel [115] to efficiently generate the surface quadrature points for both NURBS and analytic surfaces directly from B-rep models. SolidWorks is a widely used CAD software and ACIS is a commercial CAD modeling kernel. The ACIS kernel additionally includes methods that can be used to read and interpret the equations of analytic surfaces. The software implementation

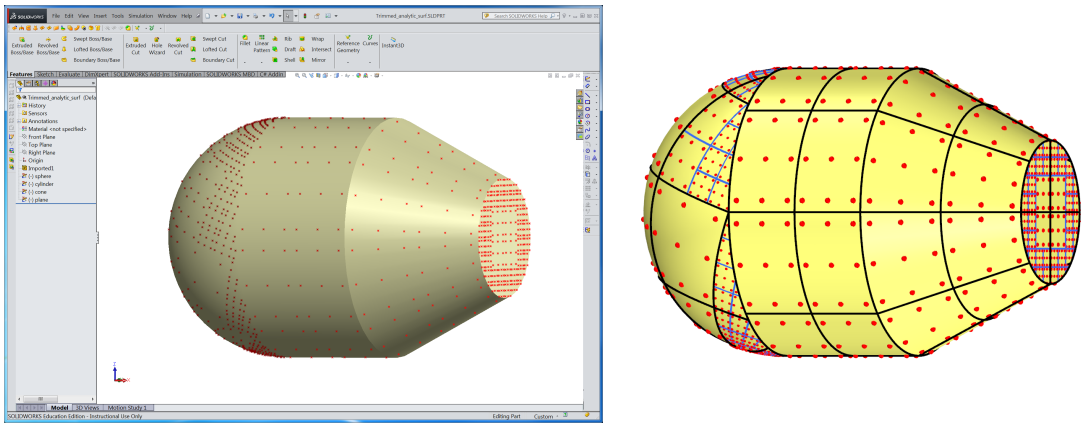


Figure 3.8: SolidWorks plug-in for generating surface quadrature rules directly from a B-rep model using ACIS kernel. The surface parameterization of the object is shown on the right.

of the generation of sub-cell-based adaptive quadrature points on analytic surfaces is shown in Figure 3.8. The implementation of the parametric surface using ACIS kernel is similar to the Grasshopper implementation.

3.2.4 Voxel-based non-boundary-fitted mesh generation

In the immersogeometric method, the mesh for the fluid domain needs to be suitably refined near the surfaces of the immersed object in order to accurately capture the flow in the boundary layer surrounding the object. For traditional boundary-fitted mesh generation, the surface tessellation can be used to control the size of the mesh in the region close to the surface. However, in the immersogeometric analysis, since the immersed object is not tessellated, the size of the immersed mesh needs to be controlled using other methods.

In this work, we propose a voxel-based approach for controlling the size of the fluid-domain mesh near the surfaces of the immersed object. For the generation of the non-boundary-fitted tetrahedral mesh, the boundary voxels of a coarse voxelization of the CAD model are used for the local refinement. The voxels are considered as individual axis-aligned bounding-boxes (AABBs) that can be used to set the maximum mesh sizes for the tetrahedral fluid-domain mesh. The boundary voxels are identified based on whether there are surface Gauss points inside them. The boundary voxels specify the regions that are near the immersed boundary where sufficient local refinement is needed to accurately capture the fluid boundary layer. To implement this method, we use the functionality of "`SizeBox`" in ANSA [18], which is a

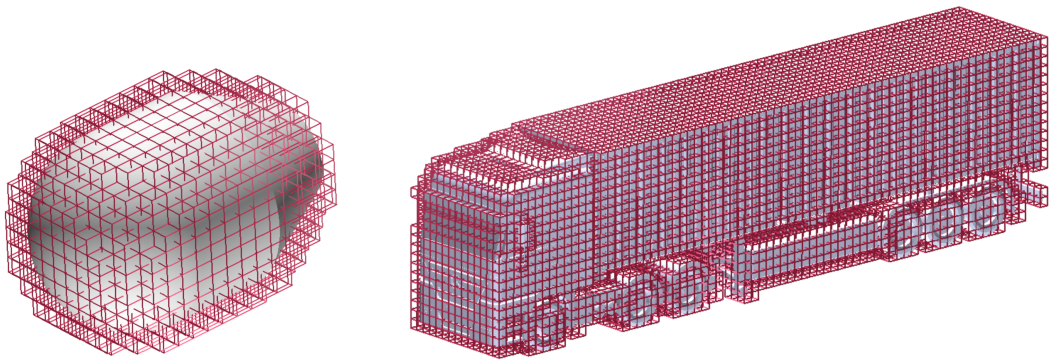


Figure 3.9: Visualization of a coarse voxelization of the CAD models used for local refinement of the immersed mesh. A fine voxelization is used for point membership classification.

commercial mesh generator. In ANSA, this function constrains the maximum length of the tetrahedral elements within each `SizeBox`. The 3D tetrahedral mesh is generated using the "`TetraCFD`" mesh generation algorithm, which is based on the advancing-front mesh-generation method [80].

A visualization of the boundary voxels used in one of our B-rep test models (torpedo) is shown on the left in Figure 3.9; this particular torpedo model has 734 boundary voxels. Since the `SizeBox` will only constrain the maximum length of the tetrahedral elements that are fully inside the AABBs of the boundary voxel, the dimension of `SizeBox` should be larger than the element size near the immersed object. The non-boundary-fitted tetrahedral mesh using these boundary voxels is shown later in Figure 3.18(b). The element size near the object surfaces is 0.04, while the voxel dimension is 0.1, which is 2.5 times larger. The tetrahedral elements, which are arbitrarily intersected with the immersed body, are sufficiently refined within each voxel. This locally refined non-boundary-fitted mesh is suitable for immersogeometric fluid-flow analysis.

3.2.5 GPU-accelerated point membership classification

To perform sub-cell-based adaptive quadrature for immersogeometric analysis, the mesh vertices of the background mesh as well as the 3D quadrature points need to be classified as being inside or outside the immersed object. We perform this operation by creating a finely sampled voxelization of the CAD object as a preprocessing step. We then consider these voxels

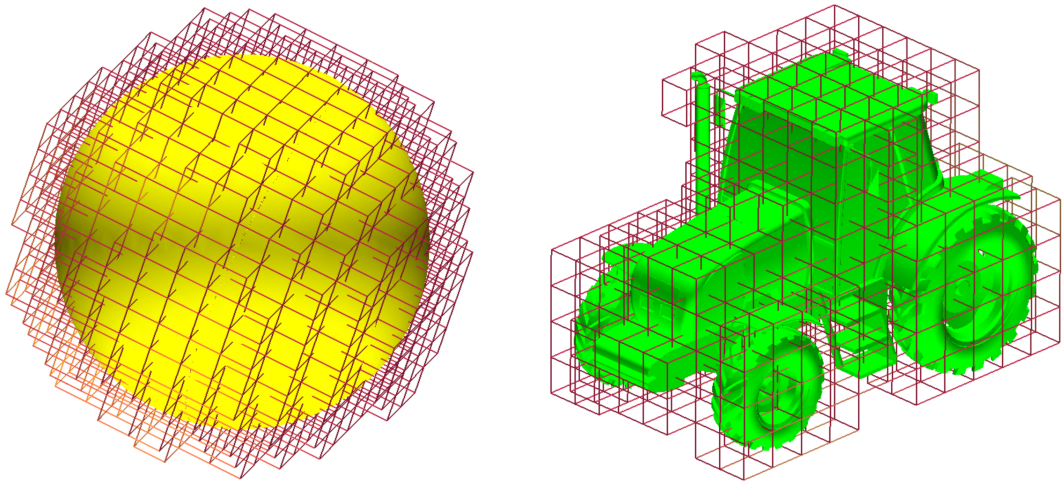


Figure 3.10: Visualization of coarse voxelization of CAD models. A fine voxelization is used for point membership classification.

as individual AABBs, which can then be used to perform the point membership classification during immersogeometric analysis.

To create the voxelized representation of the CAD model, we construct a grid of voxels in the region occupied by the object. We then make use of a rendering-based approach to classify the voxel centers as being inside or outside the object. Using this method on the GPU, a fine voxelization of the model (up to 1 billion voxels) with a relative voxel size of 0.001, can be generated (Figure 3.10). This resolution is fine enough to perform immersogeometric point membership classification for standard CAD models.

A 2D example of the method is shown in Figure 3.11; the method directly extends to 3D. The CAD model is rendered slice-by-slice by clipping it. Each pixel of this clipped model is then used to classify the voxel corresponding to the slice as being inside or outside the CAD model. This is performed by counting the number of fragments that were rendered in each pixel using the stencil buffer on the GPU. After the clipped model has been rendered, an odd value in the stencil buffer indicates that the voxel on the particular slice is inside the CAD model, and vice versa. The process is then repeated by clipping the model with a plane that is offset by the voxel size. Once all the slices have been classified, we can get the complete voxelized representation of the CAD model.

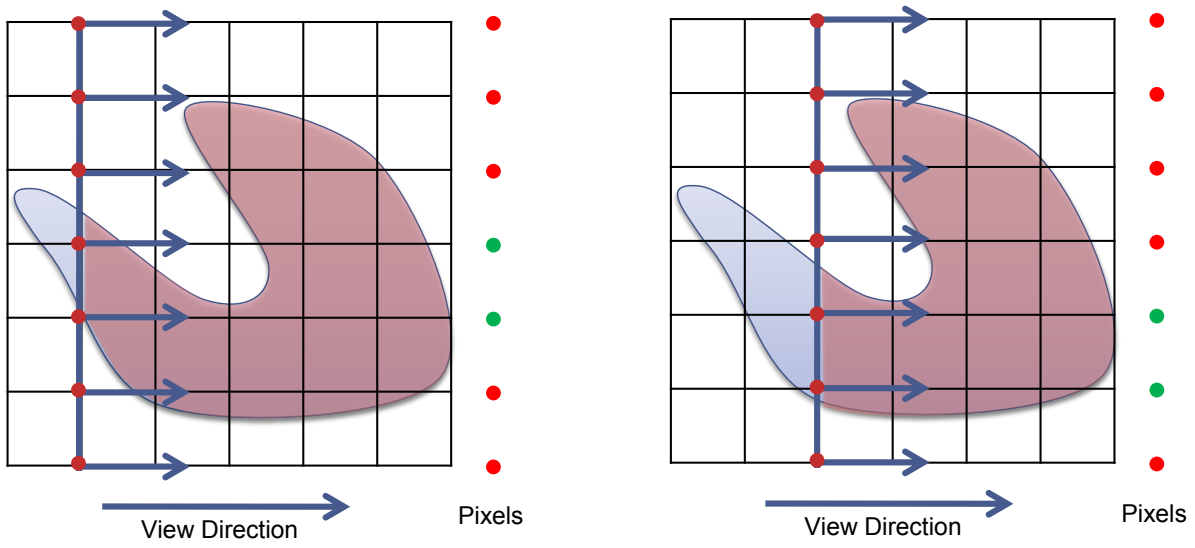


Figure 3.11: Performing point membership classification in 2D using GPU rendering. A clipped CAD model is rendered slice-by-slice and the the number of rendered pixels is counted. The pixels that are rendered an odd number of times in each slice are inside the object.

We make use of the method developed by Krishnamurthy et al. [71] to directly evaluate and render the NURBS surfaces in the model using the GPU. The B-rep model is first decomposed into its component surfaces. If the surface is a flat or a cylindrical surface, it is converted into triangles with a very fine resolution that is less than one-tenth of the voxelization resolution. All other surfaces are converted into NURBS and are evaluated using the GPU while rendering. Each surface is rendered successively to an off-screen framebuffer and the stencil buffer is used to classify the voxels corresponding to the slice as being inside or outside as explained above. Hence the model is rendered once for each slice to classify the voxelization.

The time taken to perform the classification is the sum of the time taken to evaluate the NURBS surfaces in the model once and the total time taken to render each slice. As an example, the total time taken to evaluate the NURBS surfaces in the tractor model in Section 3.3.3 considered is 3.80 seconds, while the time taken to classify 558,458,880 ($1120 \times 784 \times 636$) voxels is 22.93 seconds. At the same time, the analytic surfaces of the immersed model are converted into triangles with a very fine resolution that is less than one-tenth of the resolution of the immersed mesh. The time taken to perform the point membership classification is the sum of the time taken to evaluate the analytic surfaces in the model and the total time taken to render each slice. For example, the total time taken to evaluate the analytic surfaces in

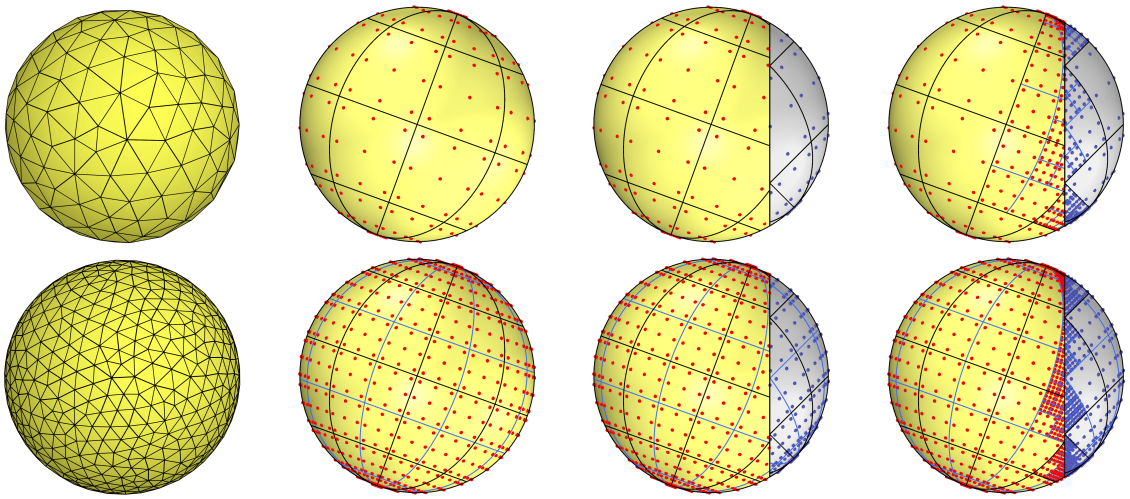
the finest model of the torpedo shape in Section 3.3.2 is 0.06 second, while the time taken to classify the 778,284,864 ($1204 \times 804 \times 804$) voxels is 28.83 seconds. These timings are obtained by running our point membership classification algorithm on a MacBook Pro with a 2.7 GHz processor, 16 GB RAM, and a NVIDIA GeForce GT 650M GPU.

3.3 Direct Immersogeometric Fluid Flow Analysis using B-rep Models

In this section, we apply our B-rep-based immersogeometric method to the simulations of flow around bluff bodies. We first consider simple B-rep models of a sphere as a benchmark study to investigate and compare the influence of trimmed parametric and analytic surfaces on the immersogeometric fluid flow analysis. We then perform simulations of a benchmark simulation of flow over a torpedo shape made of analytic surfaces and compare it to the same model made up of NURBS surfaces. Quantities of interest such as drag coefficient are in good agreement with the boundary-fitted mesh of the same geometry. Finally, we use a native B-rep NURBS-based CAD model of a full-scale agriculture tractor to demonstrate the accuracy and efficiency of our immersogeometric approach for the analysis of industrial-scale ground vehicle aerodynamics. We also perform an aerodynamic analysis of a full-scale commercial truck that has a large percentage of analytic surfaces. Using analytic surfaces over NURBS avoids unnecessary surface type conversion and significantly reduces model-preprocessing time, while providing the same accuracy for the aerodynamic quantities of interest.

3.3.1 Benchmark: Flow around a sphere

To assess the accuracy of the proposed method, we first perform computations of the benchmark problem of flow over a sphere represented using NURBS-based B-reps. Figure 3.12(a)–(c) shows the three different B-rep models of a sphere considered in this study. These models are directly used to generate the surface quadrature rules with using a same fluid domain discretization. In the first column, the surface of the sphere is tessellated with linear triangles and the geometry accuracy depends on the surface element size. The second column shows the sphere modeled using one untrimmed quadratic NURBS patch. Note that the sphere geometry is exact at the coarsest parametric level. The third column shows the sphere modeled using

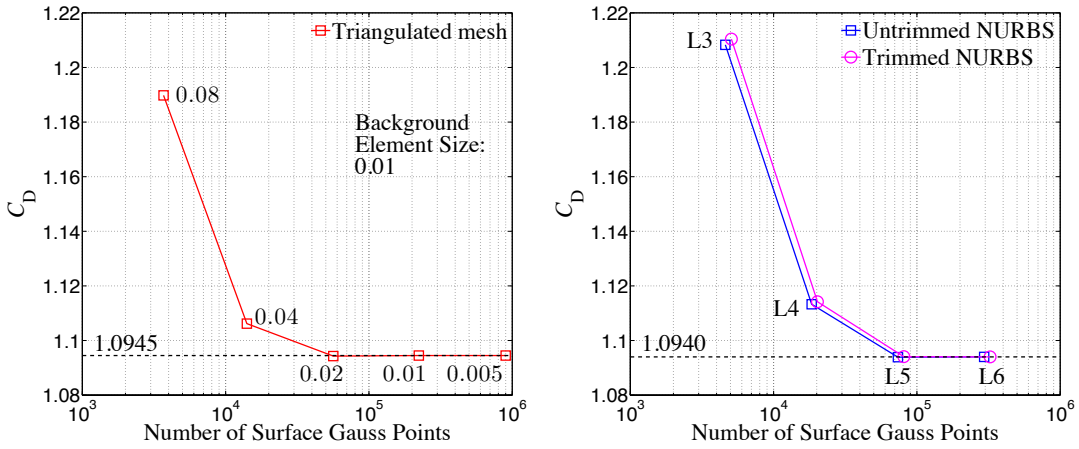


(a) Triangulated mesh. (b) Untrimmed patch. (c) Trimmed patch. (d) Case (c) with AQ.

Figure 3.12: (a)–(c) The three different B-rep models of a sphere considered in this study. The first row is the Level 1 quadrature element level and the second row is Level 2 quadrature element refinement. The triangulated mesh sizes are 0.16 and 0.08 and three quadrature points are used for each linear triangle. Nine quadrature points (three in each parametric direction) are used for each untrimmed quadrature element on the NURBS surface. (d) Addition of two levels of adaptive quadrature (AQ) sub-cells to the quadrature elements near the trim curves.

two trimmed quadratic NURBS surfaces. This allows us to study the effect of using trimmed patches in immersogeometric flow analysis.

To investigate the required surface quadrature density for the flow around a sphere problem, we consider different levels of quadrature element refinement. The first row in Figure 3.12(a)–(c) shows the coarsest quadrature level (Level 1) considered for each B-rep model. The cases shown in the second row are after one level of quadrature element refinement (Level 2). For the tessellated surface, the quadrature element refinement requires remeshing the surface using finer triangles to improve the geometry resolution. However, for the NURBS models, since the geometry is exact at the coarsest level, we refine the quadrature elements using a sub-cell approach without further discretization. At each new level, a quadrature element is split into four sub-cells, each assigned as the new quadrature elements. Standard Gaussian quadrature rule is applied to the quadrature elements. This approach is consistent and can be combined with the adaptive quadrature. We first uniformly refine the quadrature elements on the entire patch, i.e., we apply the same quadrature element refinement level to all of the NURBS elements (defined by knot spans). This is due to the necessity of having sufficient surface quadrature



(a) Triangulated surface mesh.

(b) NURBS surfaces.

Figure 3.13: Drag coefficient convergence associated with surface quadrature element densities. The triangulated surfaces cover mesh sizes from 0.08 to 0.005. The drag coefficients converge to 1.0945. The untrimmed and trimmed NURBS surface cases considered include Level 3 to Level 6 quadrature element refinements. The drag coefficients converge to 1.0940.

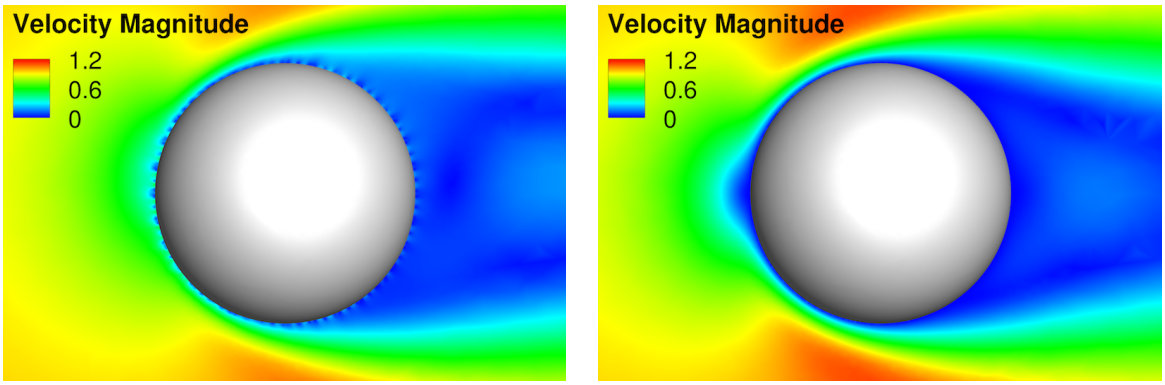
point density for immersogeometric analysis. One can then adaptively refine the quadrature elements near the trim curves for improving the accuracy of the surface integration lost due to trimming. Figure 3.12(d) shows the Level 1 and Level 2 quadrature elements of the trimmed NURBS sphere with two levels of adaptive quadrature sub-cells applied near the trim curves.

Figure 3.13 shows the drag coefficient C_D for flow around a sphere at $Re = 100$, computed with our immersogeometric method using different B-rep models of the sphere, each with several levels of surface quadrature refinement. The drag force was evaluated using the variationally consistent conservative definition of traction [7]. The triangulated surfaces cover mesh sizes from 0.08 to 0.005. The results show that taking more levels of surface quadrature element refinement converges the C_D to 1.0945, which is consistent with the value reported in Xu et al. [136]. Figure 3.13(a) also shows that choosing a similar element size between the immersed surface and the background mesh may be sufficient when a triangulated surface mesh and tetrahedral background elements are used. For the untrimmed and trimmed NURBS spheres, Level 3 to Level 6 quadrature element refinements without adaptivity near the trim curves are considered. Figure 3.13(b) shows that even though the drag coefficients between untrimmed and trimmed NURBS are slightly different at Level 3 and Level 4, this discrepancy disappears at Level 5 and Level 6. The drag coefficients for both spheres converge to 1.0940.

Comparing Figure 3.13(a) and (b), the C_D convergence curves show a very similar trend when comparable numbers of surface quadrature points are used. In fact, the triangulated surface converges faster than the NURBS surfaces. We believe this is because triangulated sphere has more evenly distributed surface elements while, due to the degenerate points and surface curvature, NURBS spheres have larger size variation between elements. The results show that the immersogeometric method is sensitive to the density and distribution of the surface quadrature points in the background mesh. It may be more efficient to refine the quadrature elements based on their physical element size. Comparing Figure 3.13(a) and (b) also reveals that when sufficiently dense surface quadrature points are used, the converged drag coefficients are slightly different when triangulated or NURBS surfaces are used. Since the background meshes are identical in all cases, this difference is likely caused by the geometric error when triangular tessellation of the curved surfaces is used (note that the NURBS geometry is exact).

Using denser and better distributed surface quadrature points increases the accuracy of the surface integration in the background mesh, which directly links to the accurate evaluation of the weak enforcement of Dirichlet boundary conditions. This can be seen in Figure 3.14, which shows the velocity magnitude contour of the immersogeometric results computed using untrimmed NURBS B-rep model with three and five levels of surface quadrature element refinement. Figure 3.13(b) shows that Level 5 produces a correct prediction of C_D while Level 3 presents a large error. It can be clearly seen from Figure 3.14(a) that due to the lack of surface quadrature point resolution in the background fluid mesh, the (weakly enforced) Dirichlet boundary conditions are satisfied poorly and the flow solutions near the boundary layer are unstable. Figure 3.14(b) shows the correct flow field obtained when sufficiently fine surface quadrature points are used. We conclude that an increased surface quadrature density is crucial to achieving accurate flow solutions in the immersogeometric analysis.

The comparison between untrimmed and trimmed NURBS results in Figure 3.13(b) suggests that with the sufficiently fine surface quadrature point density to obtain accurate flow solution, the quadrature error introduced by trimming may be insignificant. To verify this, we add 1, 2 and 4 levels of adaptive quadrature sub-cells near the trim curves to improve the surface



(a) Level 3. (b) Level 5.

Figure 3.14: Velocity magnitude contours of the immersogeometric result computed using untrimmed NURBS B-rep model with three and five levels of surface quadrature element refinement.

integration accuracy lost due to trimming. The results are shown in Figure 3.15. At Level 4, it can be clearly seen that using adaptive quadrature does influence the results. In fact, the drag coefficient at Level 4 converges under the refinement of adaptive quadrature sub-cells, but towards a value worse than the unrefined case. This may seem counterintuitive at first, but the result implies that the unrefined case was "inaccurate" due to the surface quadrature error caused by the trimming. The "correct" drag coefficient at this particular level for the trimmed NURBS sphere is the one obtained using more accurate surface integration, improved by the adaptive quadrature sub-cells.

At Level 5, when the drag coefficients are converged, Figure 3.15 shows that adding surface adaptive quadrature sub-cells near the trim curves does not change the results. This confirms that with the necessary density and distribution of surface quadrature points for accurate weak enforcement of Dirichlet boundary conditions, the quadrature error near the trim curves is negligible. As a result, it may not be necessary to use the adaptive quadrature to improve the surface integration accuracy lost due to trimming, in the context of immersogeometric fluid flow analysis. Using adaptive quadrature near the trim curves does improve the accuracy of the surface integration. However, it also increases the evaluation time. Since using the required level of surface quadrature element refinement already generates an accurate prediction of the drag coefficient, we therefore do not use adaptive quadrature on the surface in favor of the better efficiency.

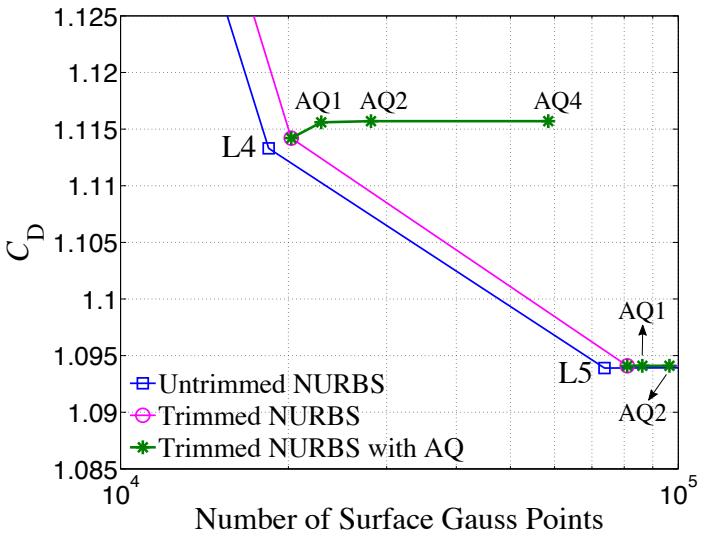


Figure 3.15: Drag coefficient predicted with our immersogeometric method using Level 4 and Level 5 quadrature elements with 1, 2 and 4 levels of adaptive quadrature (AQ) sub-cells added near the trim curves.

To investigate the required surface quadrature density for the flow around an analytic spherical surface, we also consider different levels of surface quadrature refinement. A three-point Gauss–Legendre quadrature rule in each parametric direction is applied to the quadrature elements. However, the subdivision of the analytic surface into quadrature elements is different from that of the NURBS parameterization used for the sphere. The subdivision of the analytic surface is based on the uniform length scale in physical space, while the division of the NURBS sphere is based on uniform knot spans, which in practice do not yield a uniform length in physical space [51, Appendix B]. Figure 3.16 shows three levels of surface quadrature element refinement on both analytic (red lines) and NURBS (blue lines) surfaces.

The results of the drag coefficient C_{D} for the analytic spherical surface and the NURBS surface for different levels of surface quadrature refinement (Level 3 to Level 6) are shown in Figure 3.16. The results demonstrate that the density and distribution of the surface quadrature points are crucial to apply the weak enforcement of Dirichlet boundary conditions and for obtaining accurate flow solutions.

Finally, the study presented in this section shows that B-rep CAD models, with surface quadrature rules generated using the method proposed in Section 3.2.3, are successfully and directly employed in our immersogeometric fluid flow analysis.

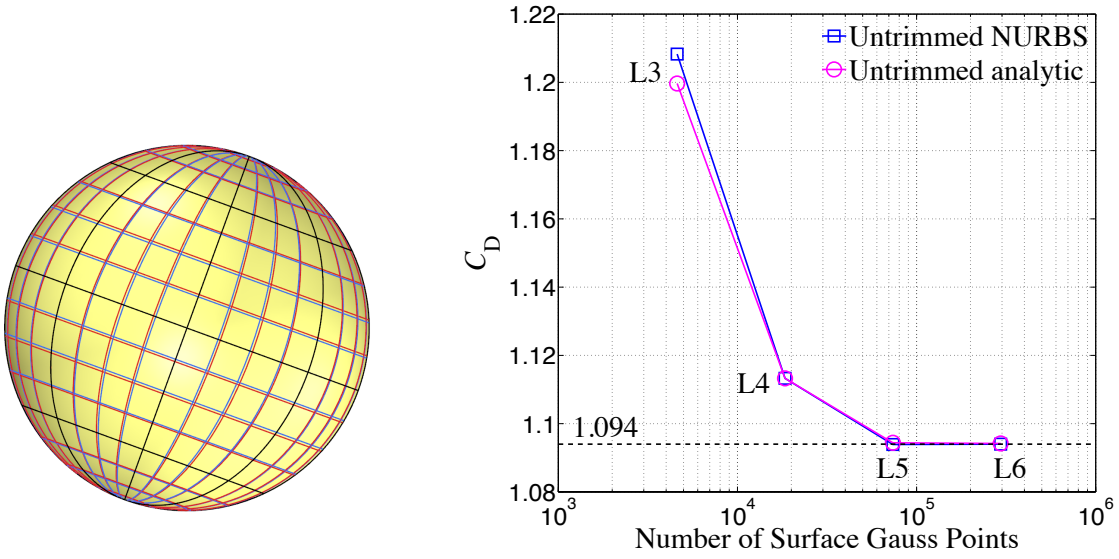
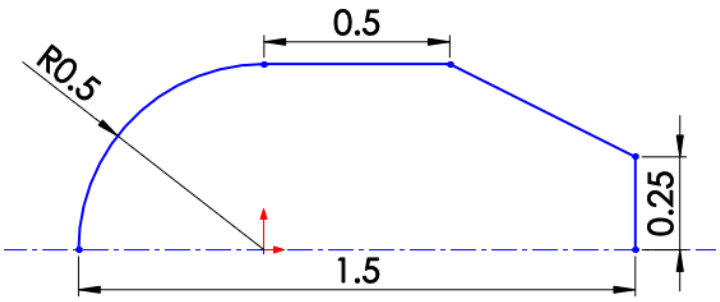


Figure 3.16: Comparison between analytic (red lines) and NURBS (blue lines) surfaces with three levels of surface quadrature element refinement. Convergence of the Drag coefficient with surface quadrature refinement. The drag coefficients converge to 1.094.

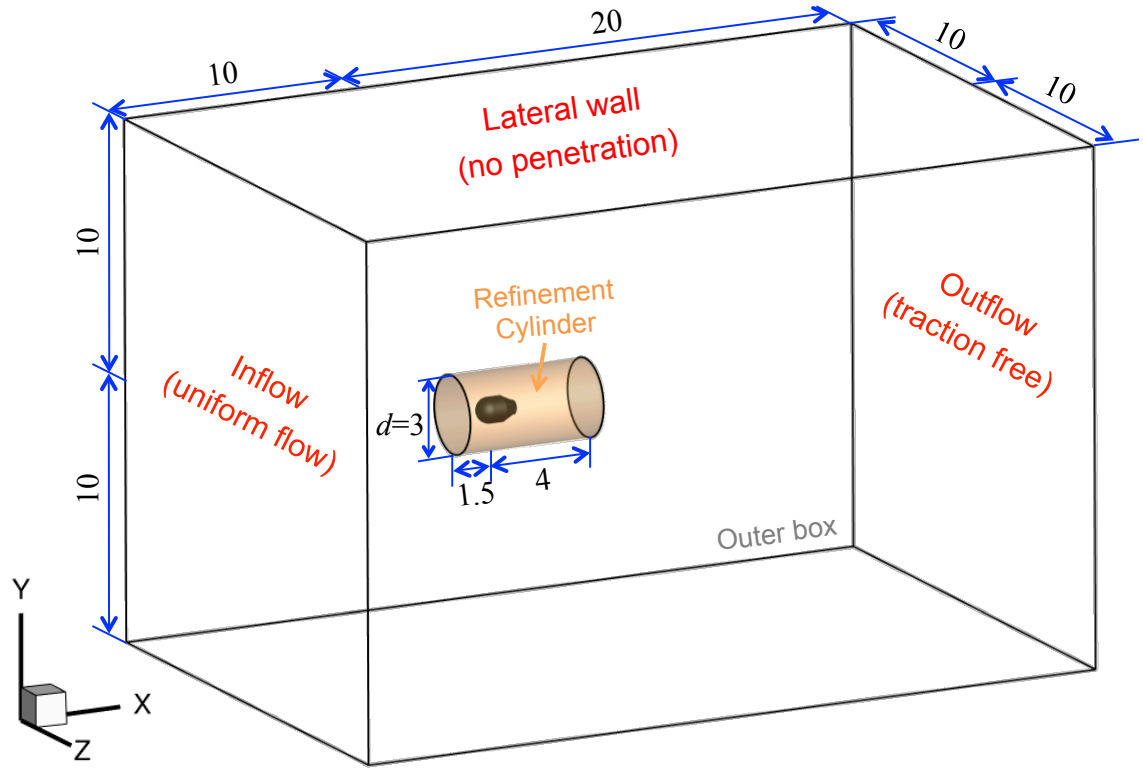
3.3.2 Benchmark: Flow around an object with trimmed analytic surfaces

In this section, we simulate the flow around a torpedo shaped object with a combination of trimmed and untrimmed analytic surfaces to assess the accuracy of our analytic-surface-based immersogeometric method. The torpedo shaped object consists of one trimmed spherical surface, one untrimmed cylindrical surface, one untrimmed conical surface, and one trimmed planar surface (Figure 3.8). The dimensions of the object are shown in Figure 3.17(a). In order to compare these two immersogeometric approaches, the same object is then converted to NURBS and immersed in the same background meshes. In addition, a boundary-fitted tetrahedral-mesh-based CFD is simulated as a reference.

The simulation setup including the dimensions of the computational domain, the refinement cylinder, and the boundary conditions is shown in Figure 3.17(b). Similar to the previous case, all sizes are non-dimensional. The inflow velocity and the fluid density are set to be unity. The characteristic length is chosen as the diameter of the front hemisphere and thus the Reynolds number can be defined as the inverse of the viscosity, that is, $Re = \mu^{-1}$. In our simulations, the viscosity is set to 0.01, so the flow around this object is at $Re = 100$. The no-slip boundary condition is enforced weakly at the surfaces of the object.

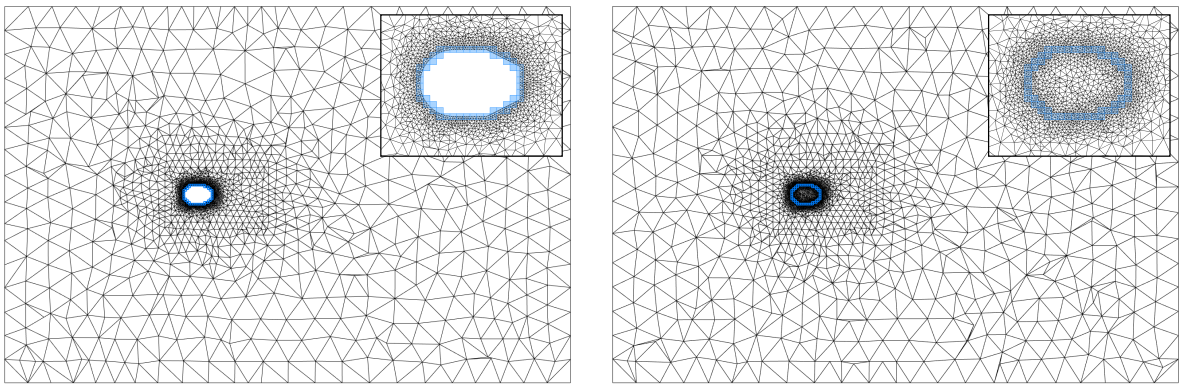


(a) Cross-section sketch.



(b) Simulation setup.

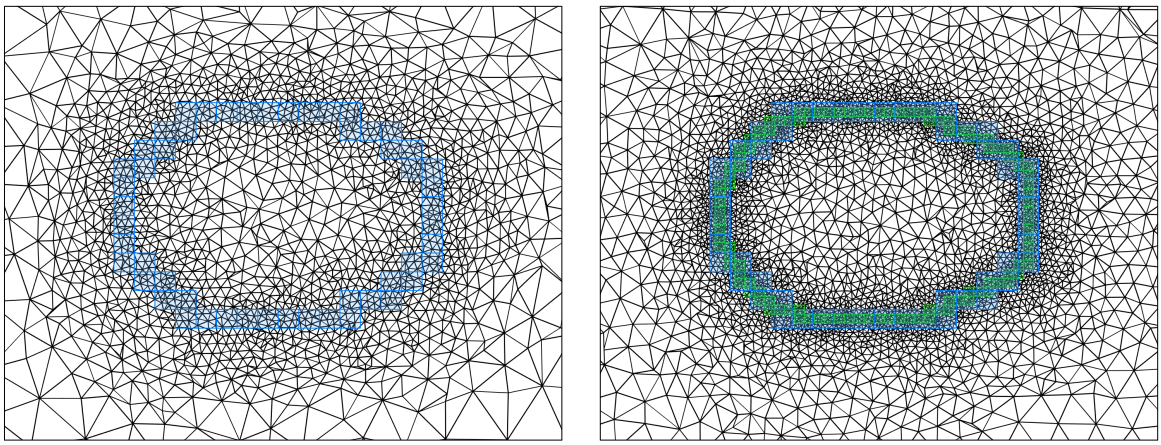
Figure 3.17: Top: Dimensions of the torpedo shaped object. Bottom: Computational domain, refinement cylinder, boundary conditions, and the immersed object.



(a) BM0 (b) IM0

Figure 3.18: Central cross-section of the coarsest boundary-fitted mesh (BM0) and the coarsest immersogeometric mesh (IM0).

The mesh resolution near the surfaces of the object is critical to resolving the boundary layers accurately. For this object, we discretize the fluid domain using linear tetrahedral elements, due to their capability of efficiently generating a locally refined non-boundary-fitted mesh around a complex object. Using the mesh generation method proposed in Section 3.2.4, the boundary voxels are used to define the locations for local refinement. To have a fair comparison, the same refinement strategy is also applied to the boundary-fitted mesh. The detailed mesh statistics and the characteristic element sizes used in the four sets of boundary-fitted meshes (BM0, BM1, BM2, and BM3) and immersogeometric meshes (IM0, IM1, IM2, and IM3) are listed in Tables 3.1 and 3.2, respectively. The central cross-section of the two coarsest meshes (BM0 and IM0) are shown in Figure 3.18. The difference in the number of elements in the respective boundary-fitted and immersed meshes is less than 4%, making them comparable for the simulations. In the BM0 and IM0 setup, the boundary voxels for Level 0 (blue) are used to set the mesh size near the surfaces of the object (Figure 3.19(a)). In the BM1 and IM1 setup, the boundary voxels for both Level 0 (blue) and Level 1 (green) are used to set the mesh size near the surfaces of the object (Figure 3.19(b)). In general, for a Level n setup, the boundary voxels of all levels until n are used to set the mesh size near the surfaces of the object. A detailed listing of the different voxel sizes used for the local refinement for the four cases is given in Table 3.3.



(a) IM0 (b) IM1

Figure 3.19: Central cross-section of the immersogeometric mesh IM0 and IM1.

Table 3.1: Element sizes and C_{D} in the boundary-fitted mesh around an object.

Mesh	Total number of elements	Near object element size	Refinement cylinder element size	Outer box element size	C_{D}
BM0	277,996	0.04	0.4	1.4	1.197
BM1	911,371	0.02	0.2	1.2	1.190
BM2	3,453,690	0.01	0.1	1.0	1.186
BM3	15,731,430	0.005	0.05	0.8	1.185

Table 3.2: Element sizes and drag coefficients in the immersogeometric mesh around an object.

Mesh	Effective number of elements	Near object element size	Refinement cylinder element size	Outer box element size	C_{D} Analytic	C_{D} NURBS
IM0	277,493	0.04	0.4	1.4	1.211	1.210
IM1	883,216	0.02	0.2	1.2	1.194	1.194
IM2	3,317,197	0.01	0.1	1.0	1.187	1.187
IM3	15,174,511	0.005	0.05	0.8	1.185	1.185

Table 3.3: Voxel-based `SizeBox` used in the mesh refinement around the object.

Mesh	Voxel size (element size) Level 0	Voxel size (element size) Level 1	Voxel size (element size) Level 2	Voxel size (element size) Level 3
IM0/BM0	0.1 (0.04)	–	–	–
IM1/BM1	0.1 (0.04)	0.05 (0.02)	–	–
IM2/BM2	0.1 (0.04)	0.05 (0.02)	0.025 (0.01)	–
IM3/BM3	0.1 (0.04)	0.05 (0.02)	0.025 (0.01)	0.0125 (0.005)

In this example, we again use sub-cell-based adaptive quadrature to handle the intersected tetrahedral elements for the immersogeometric meshes. As shown in Xu et al. [136], two levels of adaptive quadrature provide a reasonable balance between the accuracy and computational cost for the immersogeometric analysis. Meanwhile, as shown in 3.3.1, setting the surface quadrature element size the same as the volume element size near the object can ensure sufficient surface quadrature point density for the immersogeometric flow analysis of trimmed patches. The maximum quadrature element sizes used in all the cases are the same as their corresponding tetrahedral element sizes near the object.

All simulations are performed with a time-step size of 0.01. In each case, the simulation is continued until the steady state is reached. Figure 3.20 shows that the drag coefficients C_{D} for both the boundary-fitted and the analytic-surface-based immersogeometric analysis converge to 1.185 using the finest meshes. The numerical values of the drag coefficients for the two cases are also listed in Tables 3.1 and 3.2, respectively. The velocity magnitude contours for these two cases are compared in Figure 3.21 to demonstrate the quality of the flow solution.

Note that for the immersogeometric cases, the difference in the drag coefficients using the analytic surfaces and the NURBS surfaces is less than 0.001 for all levels of mesh refinement. However, as shown in Figure 3.22, the time taken to evaluate the surface Gauss points for the analytic surfaces is approximately 3 times faster than that of the NURBS surfaces. These results demonstrate that immersogeometric fluid-flow analysis using trimmed analytic surfaces, with surface quadrature rules generated using the method proposed in Section 3.2.3, produces an accurate prediction of flow quantities of interest and is more efficient than using only NURBS surfaces.

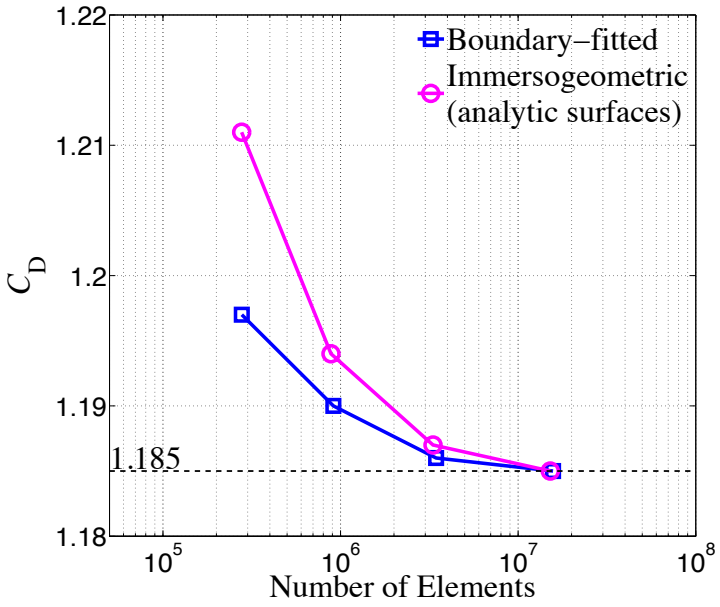
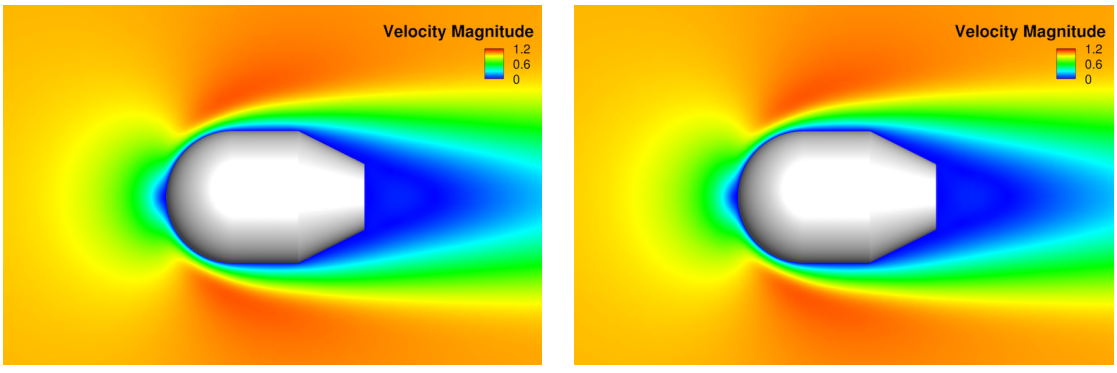


Figure 3.20: Drag coefficient convergence associated with number of effective fluid elements for boundary-fitted and immersogeometric (analytic surfaces) cases. The drag coefficients converge to 1.185.



(a) BM3. (b) IM3.

Figure 3.21: Velocity magnitude contours of the boundary-fitted and the immersogeometric simulation results.

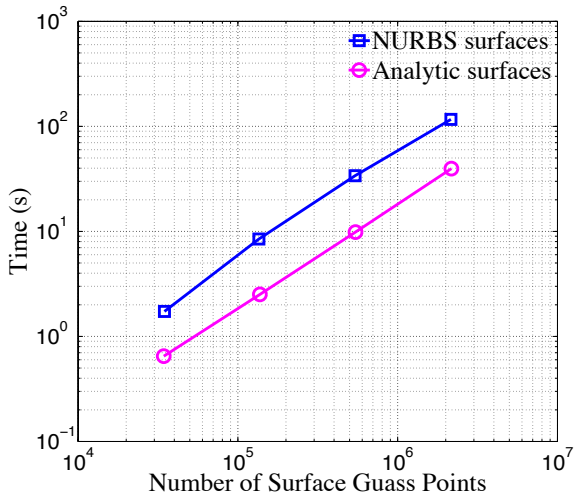
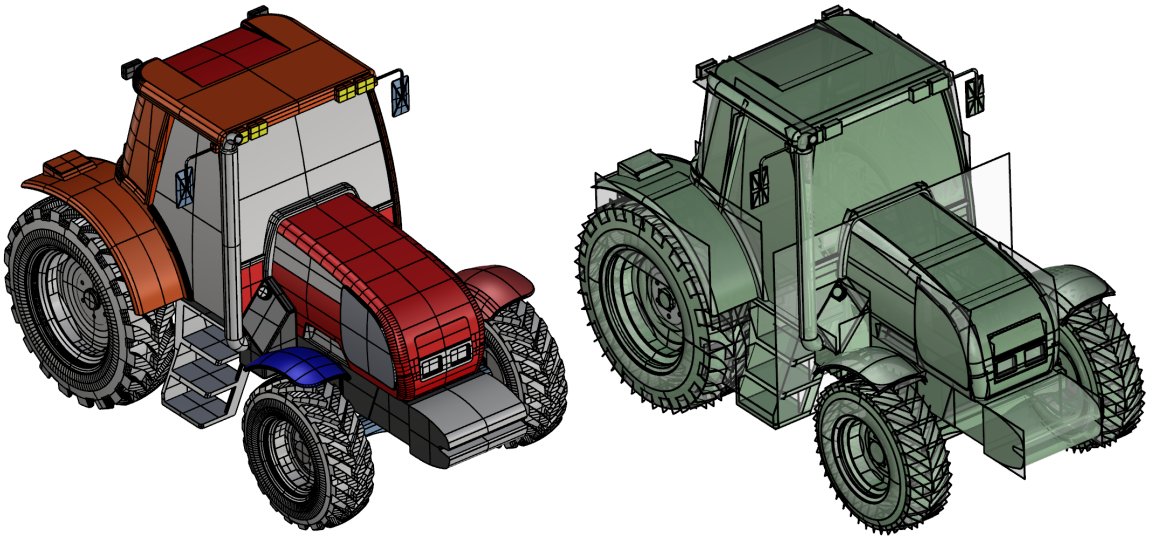


Figure 3.22: Time costs of the surface Gauss points evaluation for the NURBS surfaces and the analytic surfaces.

3.3.3 Example: Airflow around a tractor

Typical vehicle designs lead to very complex fluid domain boundaries due to the detailed geometric features of the models. This complexity constitutes a major barrier to the transfer of fluid domains into boundary-fitted computational meshes. An example is the B-rep model of the tractor shown in Figure 3.23(a), which incorporates many geometrically complex details (e.g., tires). A common practice is to preprocess the B-rep by tessellating it into triangles, and then using the triangular surface mesh for CFD mesh generation. However, generating the surface tessellations of complex CAD models is time-consuming and labor intensive, since the geometry needs to be manually checked to avoid creating any intersecting or non-manifold features during tessellation. Furthermore, the CFD mesh quality heavily depends on the surface mesh design, and, as a result, human analysts are required to perform intermediate steps such as defeaturing, geometry cleanup, and mesh manipulation [15, 76, 81, 134].

The immersogeometric method was proposed to eliminate these labor-intensive procedures from the CFD simulation pipeline while still maintaining high accuracy of the simulation results. In this section, we demonstrate how the tractor design based on B-rep can be directly immersed into a non-boundary-fitted discretization of the surrounding fluid domain and perform aerodynamic analysis in the context of large-scale industrial applications.



(a) Actual B-rep CAD. (b) Untrimmed NURBS patches.

Figure 3.23: B-rep CAD of an agricultural tractor model. The complexity of the design can be seen.

In Xu et al. [136], the tractor geometry was tessellated and described by STL format. However, conversion from B-rep to STL is not trivial, especially when the geometry is not "watertight". Here we perform the surface integration of the weak boundary conditions directly using B-rep model information. In Section 3.2.1, we discussed how B-rep involves trimmed NURBS surfaces. Figure 3.23(b) shows the untrimmed NURBS patches used to model the tractor. We use our approach proposed in Section 3.2.3 to directly access the B-rep data and trimming information and generate a proper surface quadrature rule for the actual tractor design (Figure 3.23(a)).

The problem setup, computational domain, boundary conditions, and background fluid mesh are identical to those of used in Xu et al. [136]. A zoom of the immersogeometric mesh is shown in Figure 3.24. The B-rep model of the tractor is directly immersed into the tetrahedral background mesh. Two levels of adaptive quadrature sub-cells are added in all intersected background elements to accurately integrate the volume integrals. Sufficiently accurate integration in intersected elements is essential to faithfully capture the geometry of the tractor. The inside-outside classification of the volume quadrature points is carried out using the GPU-accelerated point membership classification technique presented in Section 3.2.5.

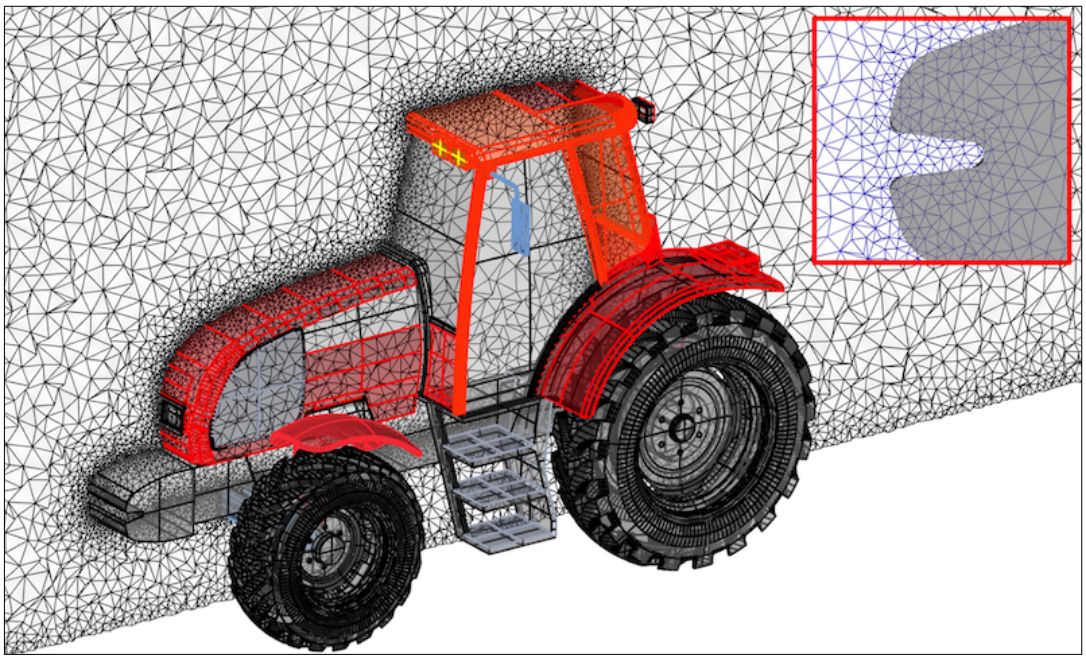


Figure 3.24: Locally refined tetrahedral meshes of the fluid domain for aerodynamic analysis of the tractor. We show the mesh cut along a plane in flow direction. The tractor model is represented directly using B-rep.

A significant advantage of the immersogeometric method is its geometric flexibility. For example, it enables us to impose a uniform mesh size along the immersed tractor surface regardless of fine-scale geometric features. It is not necessary to "defeature" those geometrically complex objects. Taking the tractor model as an example, the tires are typically smoothed due to the challenges associated with boundary-layer mesh generation. In the immersogeometric approach, this geometry manipulation is not necessary and the fluid mesh can be generated directly.

Figure 3.25 shows the immersogeometric result of the instantaneous vortical structures of the highly turbulent flow around the tractor. The vortical structure is visualized using the isosurfaces of λ_2, which is the second largest eigenvalue of the tensor $\mathbf{S}^2 + \mathbf{\Omega}^2$, where $\mathbf{S}$ and $\mathbf{\Omega}$ are the symmetric and antisymmetric components of $\boldsymbol{\nabla}\mathbf{u}$, respectively. The vortex core is defined as the region where $\lambda_2 < 0$ (see [58] for details). To assess the accuracy of the B-rep-based immersogeometric results, we first compute the time-averaged drag coefficient $\overline{C}_D = 2\overline{F}_D/\rho U^2 A$, where U is the inflow velocity, $\overline{F}_D$ is the time-averaged drag force, and A is the area of the frontal tractor surface projected onto a plane perpendicular to the main

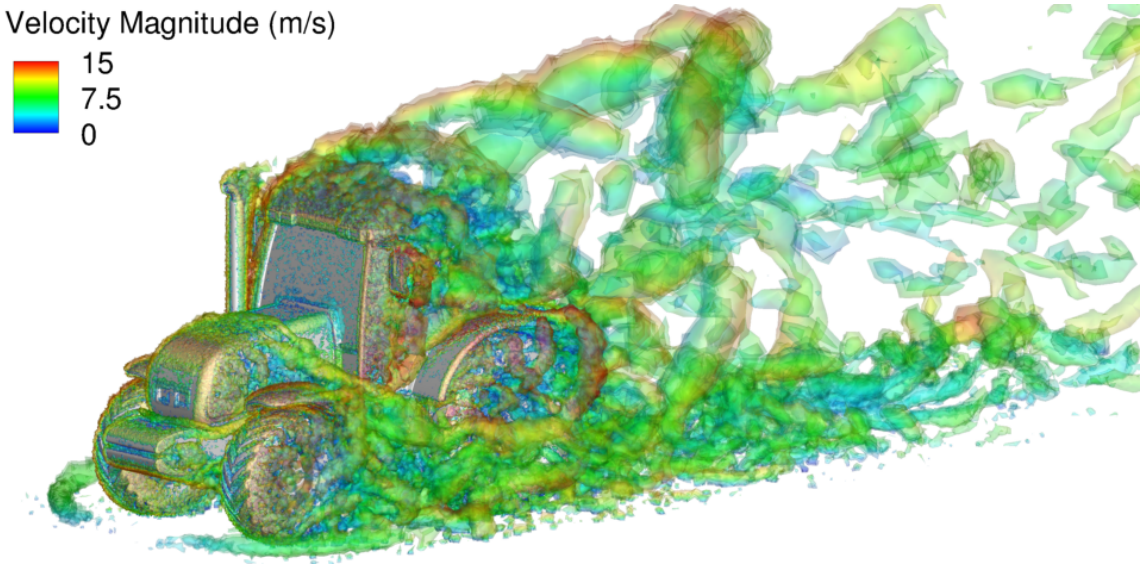
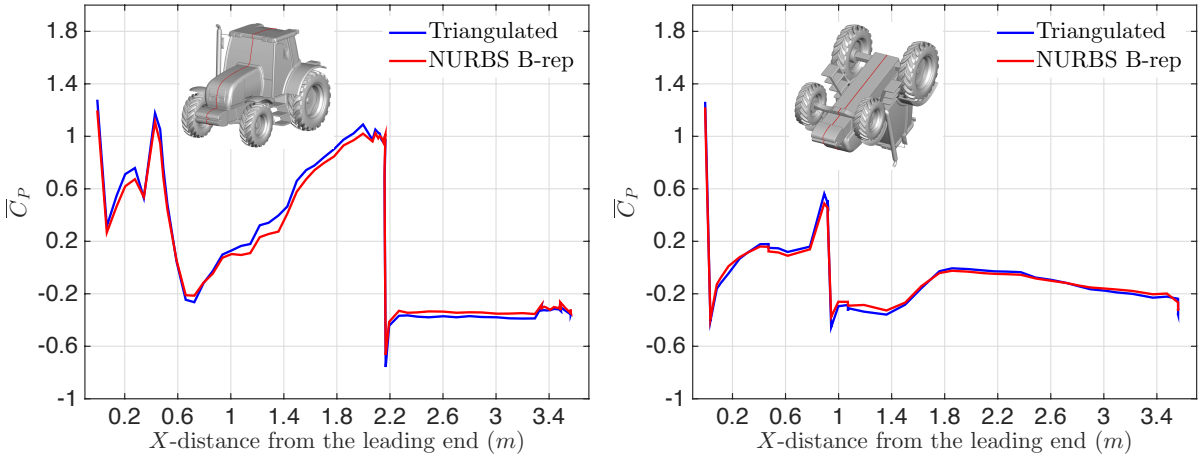


Figure 3.25: Visualization of the instantaneous vortical structures of turbulent flow around a tractor colored by velocity magnitude. The B-rep of the tractor consisting of trimmed NURBS surfaces is used directly to perform immersogeometric flow analysis.



(a) $\overline{C}_P$ distribution over the top surface.

(b) $\overline{C}_P$ distribution over the bottom surface.

Figure 3.26: Time-averaged pressure coefficient $\overline{C}_P$ plotted along center curves over the tractor surface.

flow direction. We compare our simulation result to the reference values reported by Xu et al. [136], where a triangulated tractor surface was used for the computation. The values of $\overline{C}_D$ are 0.851 and 0.864 for the triangulated-surface and NURBS B-rep computations, respectively. The results are in very good agreement between the two models. We also plot the distribution of the time-averaged pressure coefficient $\overline{C}_P$ along curves over the tractor top and bottom surfaces in Figure 3.26. An overall good agreement is again observed, which shows the effectiveness and accuracy of our direct immersogeometric fluid flow analysis using B-rep CAD models for complex-geometry problems.

Finally, the tractor analysis presented in this section indicates that our B-rep-based immersogeometric method can greatly simplify the mesh generation process for industrial turbulent flow problems without sacrificing solution accuracy.

3.3.4 Example: Airflow around a semi-trailer truck

In order to demonstrate the applicability of our analytic-surface-based immersogeometric method to industrial scale problems, we simulate the flow past a full-scale semi-trailer truck. To perform the CFD analysis of flow around a complex model, a common practice is to preprocess the B-rep by tessellating it into a triangular mesh [136] or convert it to NURBS. However, both these methods are not ideal and can lead to time-consuming mesh generation or unnecessary conversion to parametric surfaces (NURBS). The conversion to NURBS can result in poorly parametrized NURBS surfaces and often lead to poorly trimmed or missing surface features. In addition, the time cost for generating the surface quadrature for NURBS surfaces is generally higher than that for the analytic surfaces. Therefore, directly using the analytic surfaces from the B-rep model is an ideal solution to integrate design and analysis.

In a solid CAD model, analytic surfaces have frequently been used for B-rep model construction. A typical B-rep model of a road vehicle like the semi-trailer truck shown in Figure 3.27 has a larger number of analytic surfaces than parametric surfaces. This particular model has only 8 parametric surfaces but 2,480 analytic surfaces, which include 1,769 planar surfaces, 689 conical and cylindrical surfaces, 8 spherical surfaces, and 14 toroidal surfaces. Specifically, the tire mainly consists of cylindrical surfaces, the boundary of the trailer includes many planar

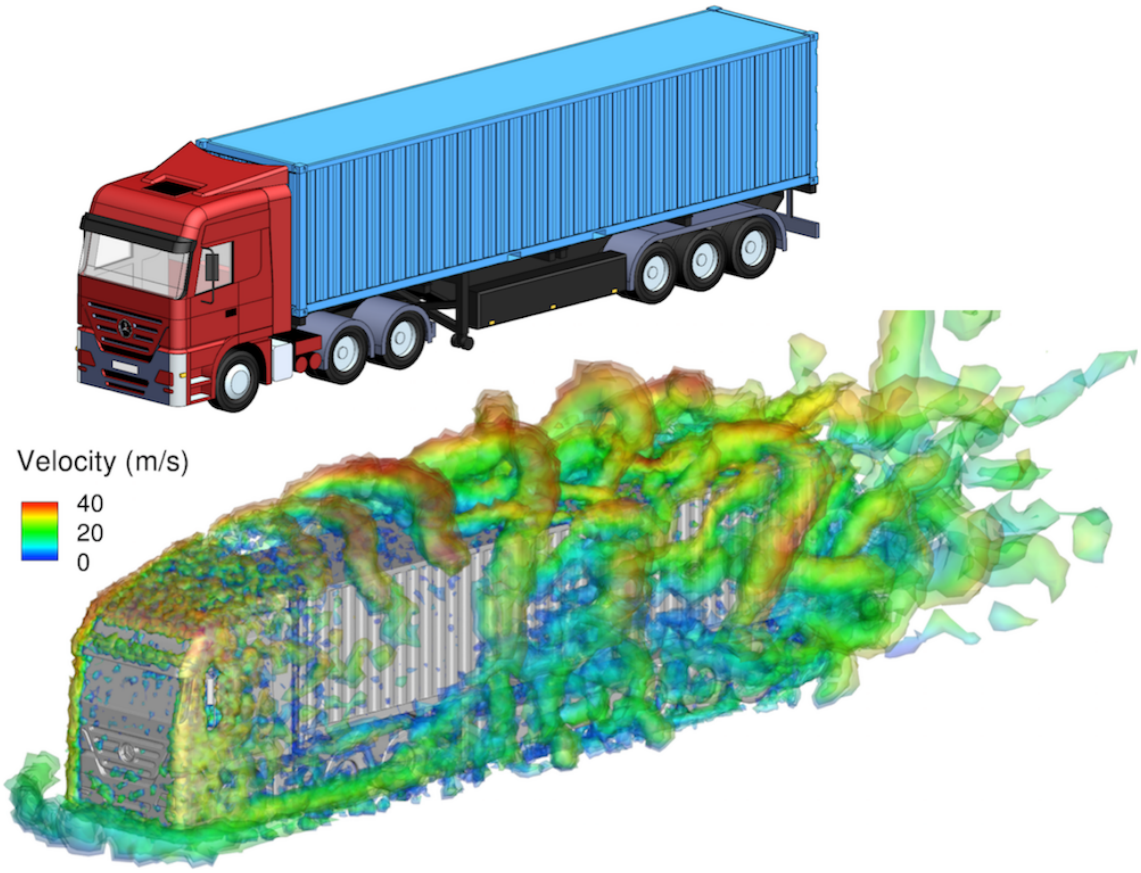


Figure 3.27: Top: The B-rep CAD model of a semi-trailer truck. The B-rep truck model consisting of analytic and NURBS surfaces is used directly to perform immersogeometric flow analysis. Bottom: Visualization of the instantaneous vortical structures of turbulent flow around the semi-trailer truck colored by velocity magnitude.

surfaces, and the corner of the mirror bracket is a toroidal surface.

There is a significant advantage of using analytic surfaces over NURBS surfaces for generating the surface quadrature points. For this semi-trailer truck, it takes only 15.54 s for generating 497,552 Gauss points using analytic surfaces, while it takes 44.12 s for generating 496,096 Gauss points when all surfaces are converted to NURBS. Using analytic surfaces, generating the surface Gaussian quadrature points is nearly 3 times faster than using NURBS. In addition, for the NURBS case, there is an additional step of converting the analytic surfaces to the NURBS surfaces. Therefore, using the analytic surfaces directly for generating the Gaussian quadrature points enables us to more efficiently preprocess the model.

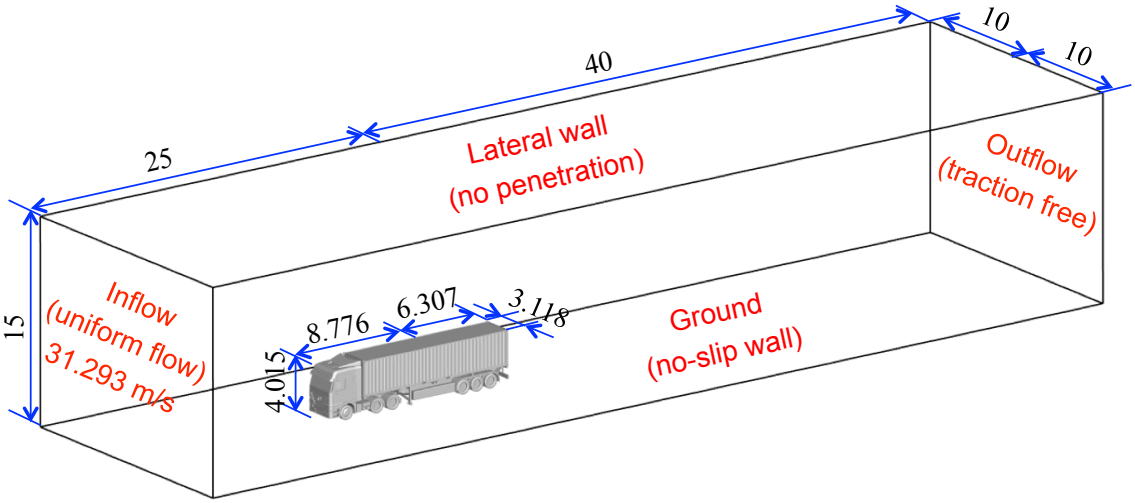


Figure 3.28: Dimensions of the immersed semi-trailer truck and the boundary conditions of the flow domain.

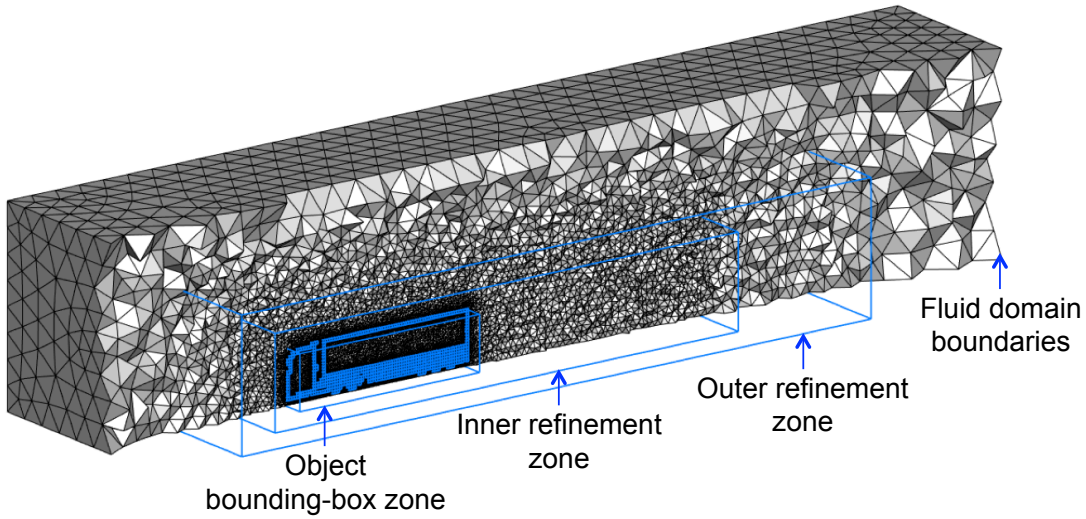


Figure 3.29: Locally refined tetrahedral meshes of the fluid domain for aerodynamic analysis of the semi-trailer truck. We show the mesh cut along a plane in flow direction. The B-rep model is used directly for generating the boundary voxels for the local mesh refinement.

The dimensions of the truck and its simulation setup including the computational domain and boundary conditions are shown in Figure 3.28. A uniform inflow with a streamwise velocity of 31.293 m/s (70 mph), which corresponds to a typical driving speed of the semi-trailer truck on highways, is applied. A no-slip boundary condition is applied to the ground for simplicity. The density and the dynamic viscosity of the air are 1.177 kg/m^3 and 1.846×10^{-5} kg/(m·s), respectively. As described in Englar [56], the characteristic length of the truck is defined as the length of the truck along the inflow direction, which is 15.083 m in our case. The Reynolds number is around 30 million, which yields a highly turbulent flow.

The B-rep model of this semi-trailer truck is directly immersed into the tetrahedral background mesh. The boundary voxels for the local mesh refinement of this semi-trailer truck are shown in Figure 3.9. The voxel-based `SizeBox`, three refinement zones and the immersogeometric mesh are shown in Figure 3.29. The element sizes set for the fluid domain boundaries, outer refinement zone, inner refinement zone, object bounding-box zone and the voxels are 2.0 m, 1.0 m, 0.5 m, 0.2 m and 0.1 m, respectively. This immersogeometric mesh consists of 2,007,309 linear tetrahedral elements (1,556,810 effective elements in the fluid domain). Two levels of adaptive quadrature are used in the intersected background elements to accurately integrate the volume integrals and faithfully capture the geometry of the semi-trailer truck. The inside-outside classification of the volume quadrature points is carried out using the finest level of the voxelization. In this case, the finest voxelization consists of 12,369,920 ($128 \times 160 \times 604$) voxels; the size of each voxel (0.025 m) is 4 times smaller than the near object element size (0.1 m).

The instantaneous vortical structures of the highly turbulent flow around the semi-trailer truck (Figure 3.27) is visualized using the isosurfaces of $\lambda_2 = -100$, -200, and -400. We also compute the time-averaged drag coefficient $\overline{C}_{\mathrm{D}} = 2\overline{F}_{\mathrm{D}}/\rho U^2 A$, where U is the inflow velocity, $\overline{F}_{\mathrm{D}}$ is the time-averaged drag force, and $A = 9.703$ m^2 is the area of the frontal truck surface projected onto a plane perpendicular to the main flow direction. The values of $\overline{C}_{\mathrm{D}}$ are 0.735 and 0.728 for the B-rep case with analytic surfaces and the case with only NURBS surfaces, respectively. These values are computed by simulating the flow for 18 s with a time-step size of 0.001 s. The drag coefficients are in good agreement with the reference drag coefficients of

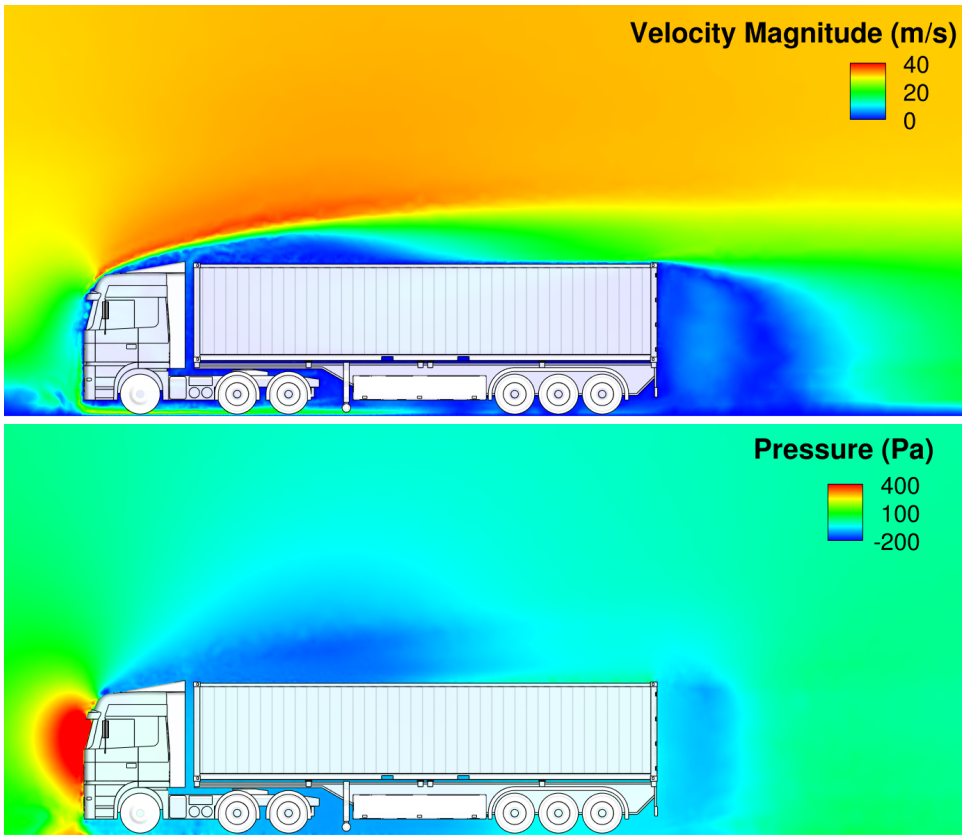


Figure 3.30: Time-averaged velocity (top) and pressure (bottom) fields on a planar cut for the case with analytic surfaces.

heavy vehicles, which are in the range of 0.6–0.9 [24, 40, 56]. The time-averaged velocity and pressure fields in a planar cross-section are shown in Figure 3.30.

The study of the semi-trailer truck presented in this section demonstrates an effective way to perform industrial-scale turbulent flow simulations using our B-rep-based immersogeometric flow analysis. The proposed method is an efficient way to overcome the complicated mesh generation process and maintain a high solution accuracy for the flow around a complex real-world geometry.

3.4 Acknowledgments

Chapter 3, in part, is a reprint of the material as it appears in: "Direct immersogeometric fluid flow analysis using B-rep CAD models," (with M.-C. Hsu, F. Xu, A.J. Herrema and A. Krishnamurthy), *Computer Aided Geometric Design*, 43:143–158, 2016; "Rapid B-rep model

preprocessing for immersogeometric analysis using analytic surfaces," (with F. Xu, M.-C. Hsu and A. Krishnamurthy), *Computer Aided Geometric Design*, 52–53:190–204, 2017. The dissertation author was the primary investigator of these papers.

CHAPTER 4. CARDIAC FSI SIMULATION WITH IMMERSED BIOPROSTHETIC HEART VALVES

In the previous chapters, a geometric framework for rigid immersed objects was developed. However, the proposed framework can be extended to perform immersogeometric analysis with deformable models, especially with models involving large and complex structural deformations such as heart valves. In addition, the parametric modeling framework can be used to streamline the modeling process to efficiently handle the deformable boundary of a moving fluid domain. The geometric framework for immersogeometric analysis can efficiently integrate the moving domain with the deformable model. Similar to the FSI simulation, the immersed object can move independently of the background fluid mesh, which can avoid time-consuming mesh re-generation process. For example, a moving fluid domain, such as a left ventricle can be coupled with the deforming heart valves using our geometric framework.

To demonstrate the capability of the framework, we perform blood flow simulations in a human left ventricle (LV) with aortic and mitral valves coupled using FSI. The LV motion is used to prescribe the boundary of the moving fluid domain. The bioprosthetic aortic and mitral valves are immersed in the moving fluid domain using FSI. In this chapter, we provide the details of the framework for the parametric design of LV with the aorta. This is based on our previous parametric design-through-analysis platform in Section 2.1.3 for geometry modeling. The utility of this immersogeometric framework for complex simulations is demonstrated using flow simulations in a moving LV with FSI of two BHVs.

4.1 Parametric Modeling of the Left Ventricle

The interactive parametric modeling platform is used to automate the geometric modeling of the left ventricle. We generate the reference configurations of the immersed object (biopros-

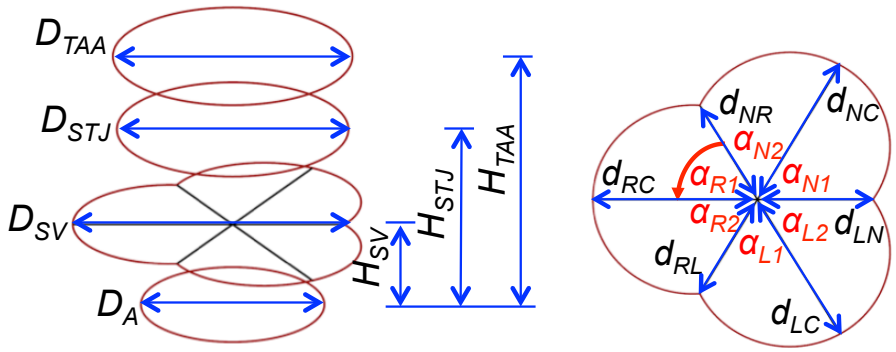


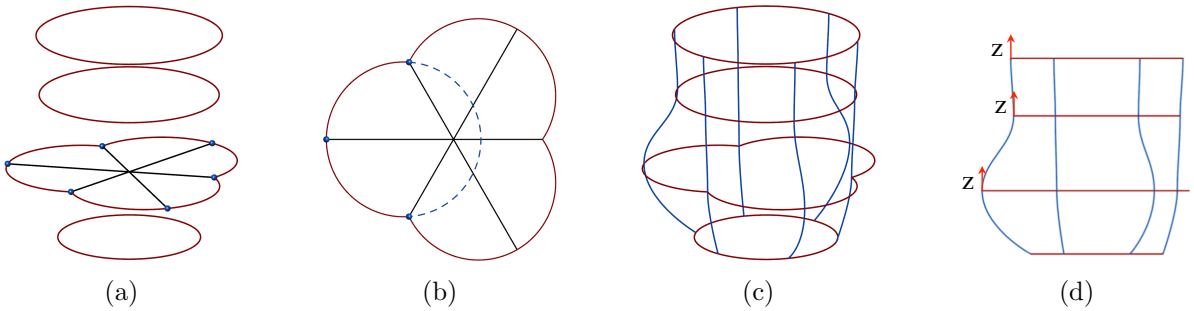
Figure 4.1: The key geometric parameters of the aortic root. The dimensions are given in Table 4.1. D_A: Ventriculo–aortic junction diameter. D_{SV}: Valsalva sinuses diameter. D_{STJ}: Sinotubular junction diameter. D_{TAA}: Ascending tubular aorta diameter. H_{SV}: Distance between the plane of maximal expansion of the Valsalva sinuses and the ventriculo–aortic junction. H_{STJ}: Distance between the sinotubular junction and the ventriculo-aortic junction. H_{TAA}: Distance between the arbitrary cross section of the ascending aorta and the ventriculo–aortic junction. d_{RC}, d_{LC}, d_{NC}: Distance between the central coaptation point and the maximal expansion of right, left and non–coronary Valsalva sinus. d_{RL}, d_{LN}, d_{NR}: Distance between the central coaptation point and each commissural line. $\alpha_{R1}, \alpha_{R2}, \alpha_{L1}, \alpha_{L2}, \alpha_{N1}, \alpha_{N2}$: Characteristic angles of the three sinuses, which are equal to 60° if this aortic root is symmetric.

thetic heart valves) and the moving fluid domain including sinuses, ascending aorta, and LV. In addition, the prescribed motion of the LV is transferred to the aorta. Finally, the moving mesh for the complete fluid domain is generated.

4.1.1 Parametric modeling of aortic root

For the geometric modeling of the aortic root, the 3D modeling algorithm developed by Morganti et al. [82] is implemented. This algorithm is based on dimensions extracted from the 2D echocardiography at four different cross-sections: the ventriculo–aortic junction, the valsalva sinuses, the sinotubular junction, and the ascending tubular aorta. These dimensions and other important parameters of the aortic root are shown in Figure 4.1. The dimensions of all parameters are chosen to be the average values from Morganti et al. [82], Saura et al. [97] and Roman et al. [92]; the selected parameters are given in Table 4.1. In this work, the diameter of the ventriculo–aortic junction is set as 23–mm, which is consistent with the leaflet design by Edwards Lifesciences used in our previous study [50].

[1] $\{d_{RC}, d_{LC}, d_{NC}\} = D_{SV} - \{d_{LN}, d_{NR}, d_{RL}\}$



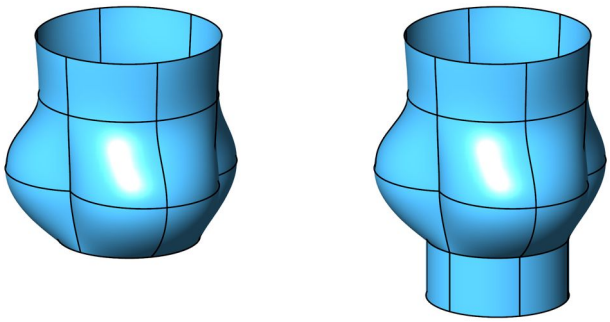
(a) (b) (c) (d)

Figure 4.2: Curves at four cross sections: (a) $d_{RC}, d_{LC}, d_{NC}, d_{RL}, d_{LN}$ and d_{NR} define blue points of cross section at the maximal sinus expansion. (b) Each arc at the maximal sinus expansion is based on three blue points. Curves along longitude direction: (c) Three commissure lines and three lines along intersected points. (d) Tangencies of the curves along longitude direction are constrained at three cross sections by unit z–vector.

Based the selected dimensions, the baseline aortic root geometric model is constructed in two steps. The first step is to construct a wireframe of the model, which will represent the shape of the aortic root with a few characteristic lines. The characteristic lines consist of two types of curves: the latitudinal curves at each cross section and the longitudinal curves along the z–axis. The latitudinal curves are assumed to be circles at the ventriculo–aortic junction, the sinotubular junction, and the ascending tubular aorta. The curves at the cross section at the maximal sinus expansion are built using three arcs. The latitudinal curves of the wireframe model are red curves shown in Figure 4.2(a) and Figure 4.2(b). The longitudinal curves are cubic NURBS curves interpolated through intersected points of the latitudinal curves which are evenly divided into six segments by these intersected points. The tangency of the interpolated cubic NURBS curves is constrained at the cross sections of the valsalva sinuses, the sinotubular

Table 4.1: The dimensions of the aortic root.

Parameters (Number of Patients)	Average Total (n = 12) [82]	Average Male (n = 68) [97]	Average Male (n = 310) [92]	Our Settings
D_A (mm)	20.5	26 ± 3	21.9 ± 2.2	23
D_{SV} (mm)	32.6	34 ± 3	33.6 ± 3.9	34
D_{STJ} (mm)	29.8	29 ± 3	28.7 ± 3.2	29
D_{TAA} (mm)	31.4	30 ± 4	29.9 ± 3.8	30
d_{RC}, d_{LC}, d_{NC} (mm)	18.8	–	–	20[1]
d_{RL}, d_{LN}, d_{NR} (mm)	13.8	–	–	14
H_{SV} (mm)	10.9	–	–	11
H_{STJ} (mm)	24.4	–	–	24
H_{TAA} (mm)	34.2	–	–	34



(a) (b)

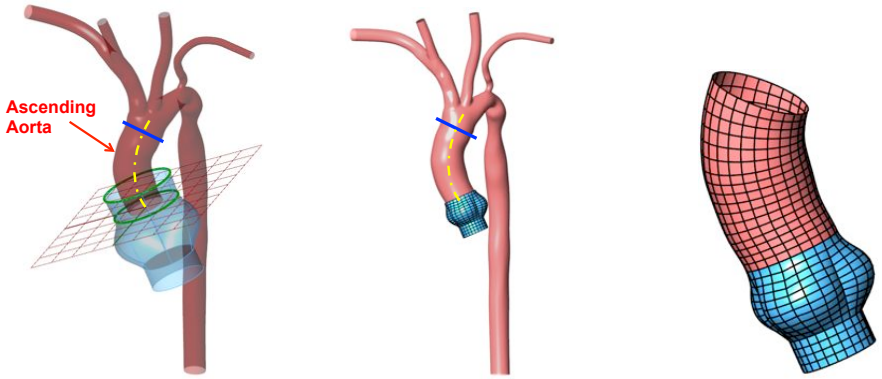
Figure 4.3: Baseline aortic root model: (a) Surface is based on the curves at each cross section and along the longitude direction. (b) Extended bottom surface using linear extrusion.

junction and the ascending tubular aorta by unit z–vector. The longitudinal curves of the wireframe model are blue curves shown in Figure 4.2(c) and Figure 4.2(d), respectively. The second step is to construct a surface based on the latitudinal and the longitudinal curves. Then the artificial part of the bottom of the aortic root can be an extended surface based on linear extrusion along the negative z-direction. The extrusion length is set as 10 mm. Therefore, the baseline aortic root model is shown in Figure 4.3(a) and Figure 4.3(b).

4.1.2 Parametric modeling of ascending aorta

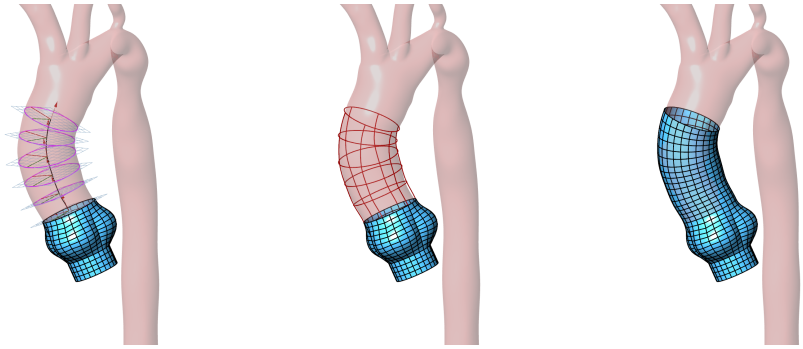
For the geometry modeling of the ascending aorta, a lofting method is used to build its vascular surface. The vascular surface is generated by lofting a circular cross-section along the arterial path. The arterial path is based on the center point of each cross section. The radius at different cross sections is then used for lofting the ascending aorta.

A STL mesh of the ascending aorta of an 8-year old female patient with Body Surface Area (BSA) of 0.94 m^2 and D_{STJ} of 16 mm [111] is used in this analysis. However, compared with the aortic root model, which has characteristic BSA of 1.9 m^2 and D_{STJ} of 29 mm, the patient-specific ascending aorta is relatively small (Figure 4.4(a)). Therefore, the patient-specific aorta is scaled up and rotated with respect to the sinotubular junction plane (Figure 4.4(b)). The new vascular surface is then smoothly connected to the parametric aortic root, which is shown in Figure 4.4(c).



(a) (b) (c)

Figure 4.4: Ascending aorta: (a) Patient-specific aorta and the baseline aortic root model. (b) Scaled aorta with respect to the sinotubular junction plane. (c) New vascular surface.



(a) (b) (c)

Figure 4.5: Ascending aorta: (a) Intersected curve at each cutting plane to get an effective radius. (b) A wireframe curves based on the new circles. (c) Conforming NURBS surfaces based on the wireframe curves.

The details of the reconstruction of the modified patient-specific ascending aorta are illustrated in Figure 4.5. In Figure 4.5(a), an iterative method is used to extract the intersected curve and the center point of the arterial path. The normal direction of the intersected curve is tangential to the intersection point along the arterial path. The effective radius of the intersected curve is used to construct a new circle at each cross section. Each circle is evenly split into 6 arcs, with latitudinal NURBS curves passing through the end points of the arcs. Wireframe of the circular arcs and NURBS curves are shown in Figure 4.5(b). Finally, Figure 4.5(c) shows the conforming NURBS surfaces generated based on the wireframe curves.

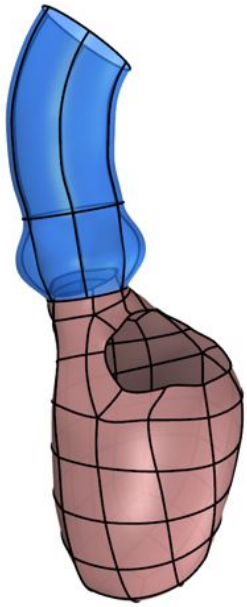


Figure 4.6: Relative location of the reference configuration of the left ventricle and aorta.

4.1.3 Ventricular model with ascending aorta and valve annuli

The parametric model of the ascending aorta is then combined with a geometric model of the left ventricle. Based on the patient-specific models of cardiac biomechanics developed by Krishnamurthy et al. [74], the LV of one patient at end-diastole is used for the construction of the reference configuration of the moving fluid domain. The relative location between the aorta and the LV is shown in Figure 4.6.

In order to couple the patient-specific left-ventricle with the parametric model of the aorta, the motion of the base of the aorta from the left-ventricular simulation needs to be transferred to the aorta (Figure 4.7). The prescribed motion of the LV model is based on the structural simulation for a full cardiac cycle. At each temporal step, the motion of the aorta is based on the quadratic interpolation with the maximum value of translation, rotation, and scaling applied at the bottom of the aorta. The maximum values are based on the ratio and the orientation between the circles from the top outlet of the LV model and the bottom inlet of the aorta. At the same time, the tangencies of the patches in the LV is also transferred to the bottom of the modified aorta. Thus, the LV and the modified aorta are smoothly connected and form the surfaces of the moving fluid domain at each temporal step.

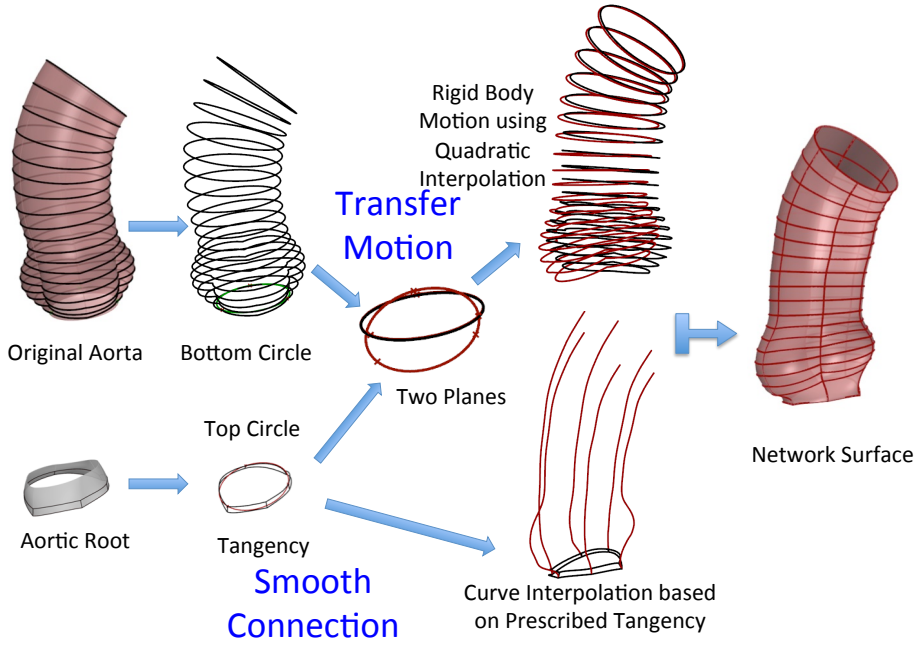


Figure 4.7: Coupling the patient-specific left ventricular geometry with the parametric model of the aorta.

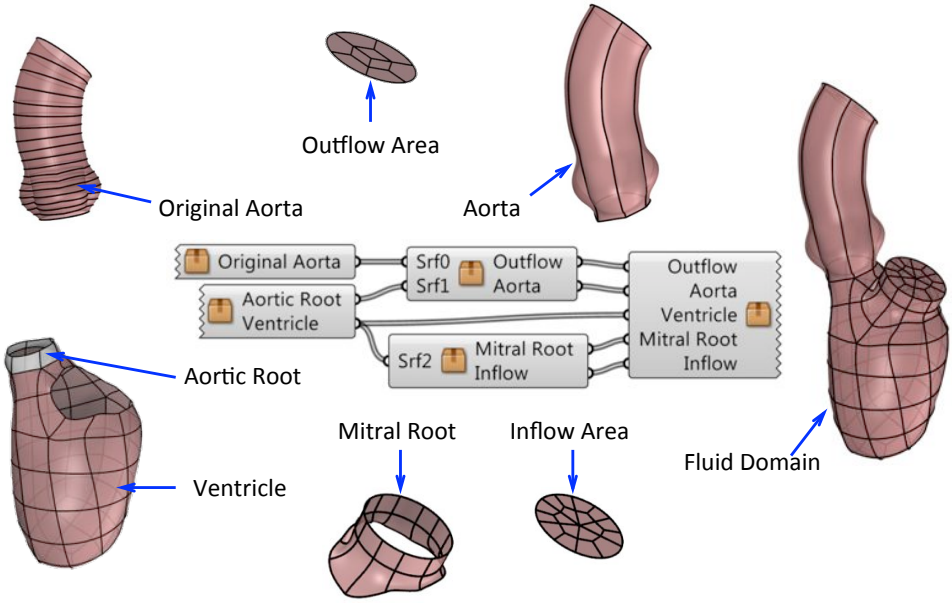


Figure 4.8: Parametric design and geometry modeling for aorta and left ventricle.

A Grasshopper implementation for generating the temporal configuration of the fluid domain boundary is shown in Figure 4.8. The left two components are the input geometries based on the aorta and patient-specific LV model. The right three components use these two models to generate a smoothly connected boundary surface with prescribed motion. The new model shown at the right side of Figure 4.8 is the current temporal configuration of the surfaces of the fluid domain.

4.1.4 Parametric modeling of bioprosthetic heart valves (BHVs)

The previous sections outlined the methods for geometric modeling the boundary surfaces of the fluid domain. In this section, the geometric algorithm used to model the bioprosthetic heart valve (BHV) as shells is presented. Figure 4.9 shows a snapshot of the Rhino CAD modeling software interface, with the T-spline BHV model that is used in performing the FSI simulations. This BHV leaflet geometry is based on a 23-mm design by Edwards Lifesciences [69, 119]. The leaflets of the BHV are modeled using three cubic T-spline surfaces, as shown in Figure 4.9. The use of unstructured T-splines enables local refinement and coarsening [109] and avoids small and degenerate NURBS elements near the commissure points (see Kamensky et al. [62] and Hsu et al. [49]). Note the presence of the metallic stent in the model, which makes BHV geometric modeling complicated. However, the design platform employed in this work can be used to model the BHV with the stent.

The Grasshopper program for the geometry design of the BHVs consists of 4 different geometry construction steps (see Figure 4.10 for a visual illustration). First, the parametric input is used to construct NURBS curves, which are the bounding curves for the NURBS surface patches that define the valve leaflet geometry. The resulting multi-patch NURBS geometry is then re-parameterized to create a single T-spline surface geometry. Finally, local refinement and/or coarsening is performed as needed to use the final design directly in FSI simulations. The complete Grasshopper program is shown in Figure 4.11.

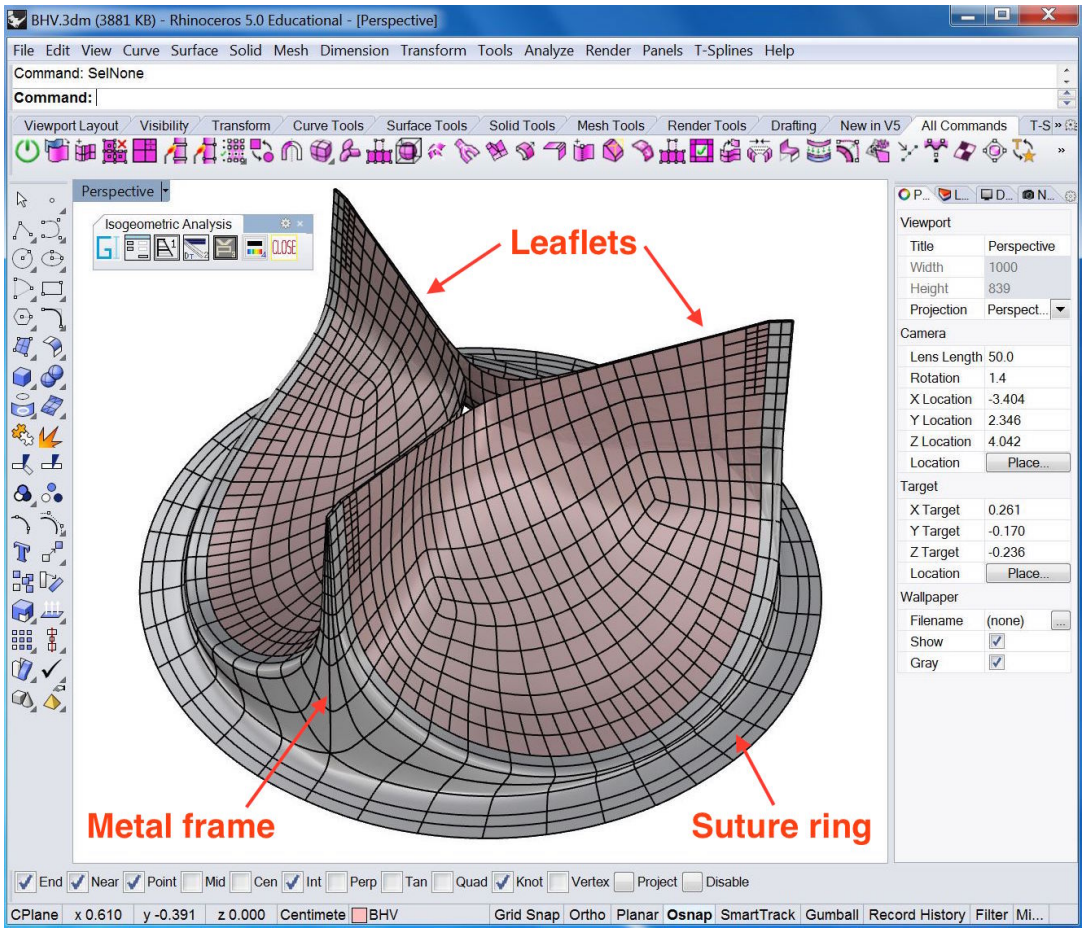


Figure 4.9: Trileaflet T-spline BHV model. The T-splines were generated using the in-house parametric modeling platform (Figure 4.11) and the Autodesk T-Splines Plug-in for Rhino [5].

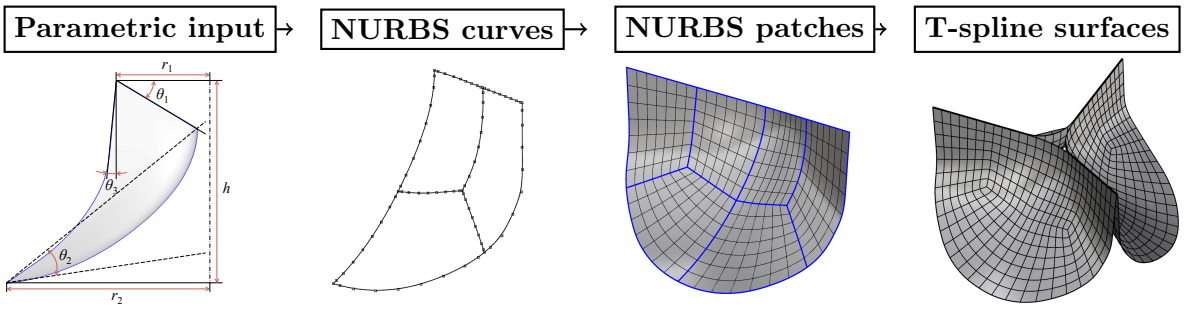


Figure 4.10: Parametric BHV geometry modeling flowchart.

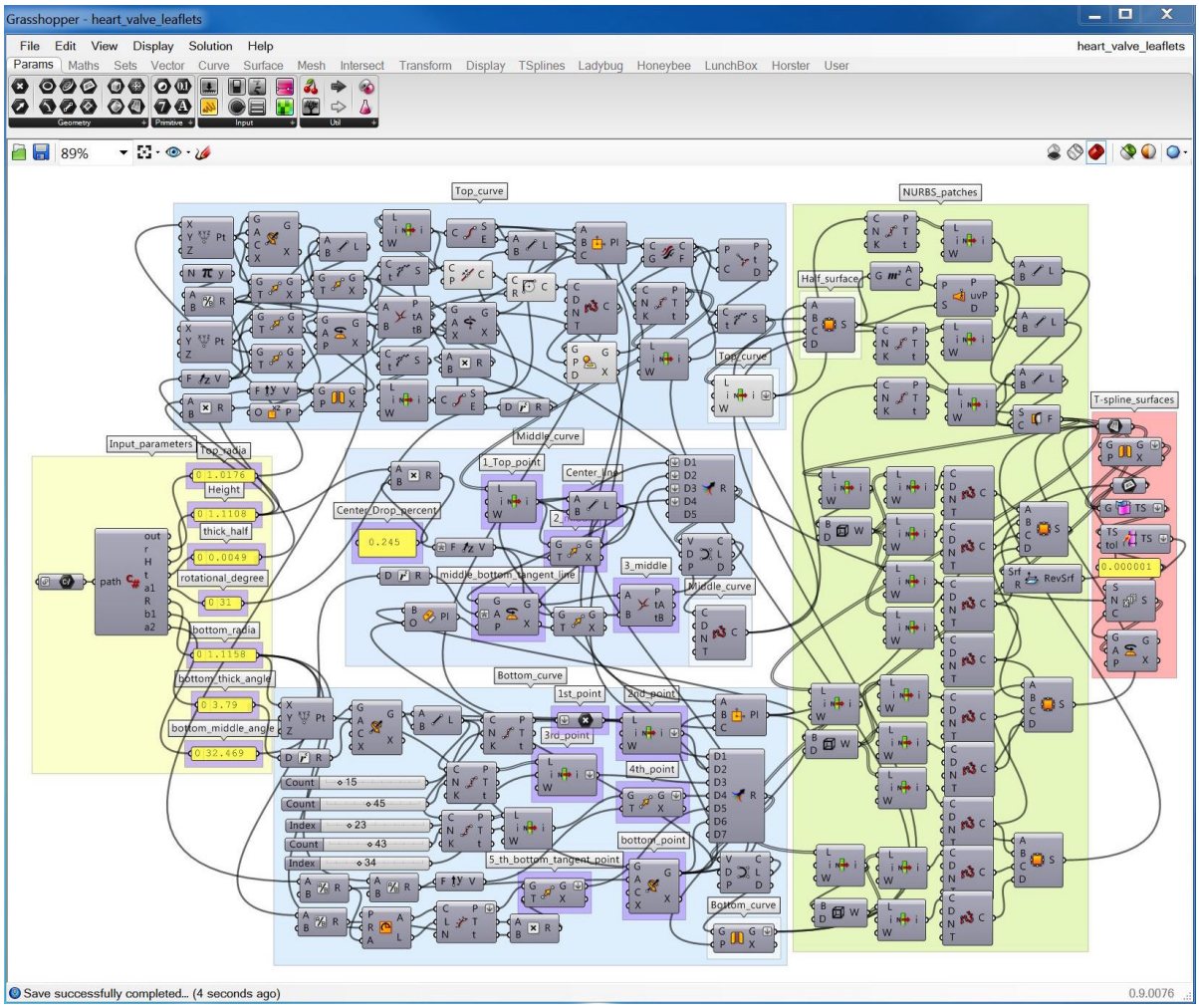


Figure 4.11: Grasshopper interface for parametric BHV geometry modeling. The geometry construction steps are shown in Figure 4.10.

4.2 FSI with Moving Boundary

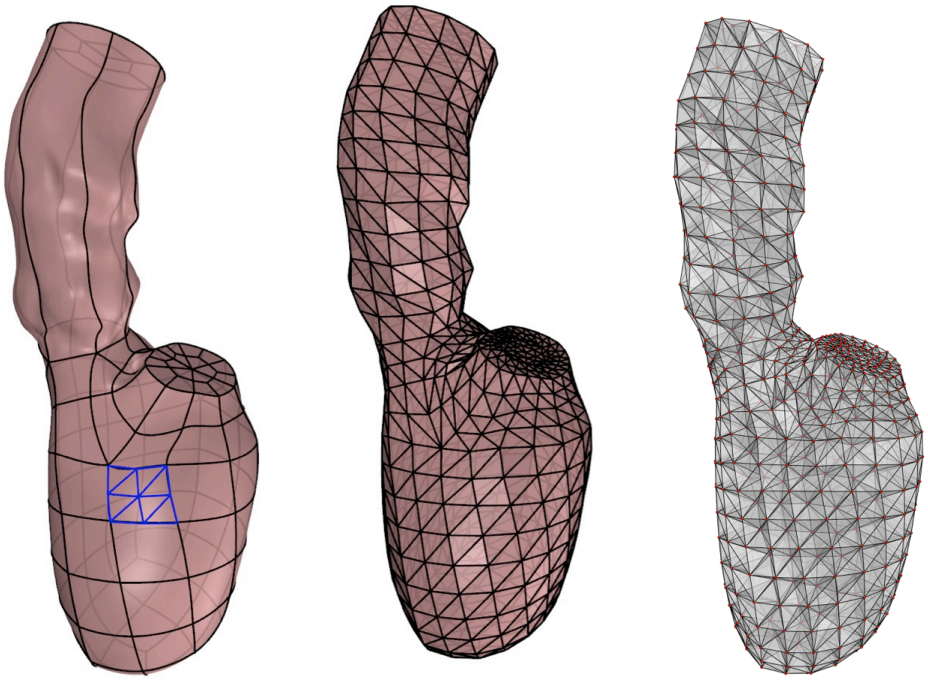
In this section, the immersogeometric method is applied to the simulations of blood flow inside a left ventricular heart model, where two BHVs are used instead of the aortic valve and the mitral valve. During diastole, the aortic valve closes while the mitral valve opens permitting injection of oxygenated blood from the left atrium into the LV. During systole, the LV contracts and the mitral valve closes to block the flow of blood back to the left atrium; the aortic valve opens to allow the ejection of blood into the aorta. The geometric framework is used to perform the FSI analysis of a full-cycle heart beat with moving left ventricular walls.

4.2.1 Moving fluid domain mesh generation

For the generation of the non-boundary-fitted tetrahedral mesh of the moving fluid domain, a closed mesh of the boundary of the fluid domain is first constructed. Since the edges of NURBS surfaces of the heart model are conforming to each other, the shared edge between the patches is divided into the same number of elements. Then a structured closed mesh is constructed by welding all the shared mesh vertices along all the shared edges. A sample mesh of single patch is shown in Figure 4.12(a), and the closed mesh of the fluid domain boundary is shown in Figure 4.12(b). The volumetric mesh is generated using the automatic FEM mesh generation software (ANSA) to generate fluid domain mesh (Figure 4.12(c)); the total number of elements in the volumetric mesh is 1,232,871. To accommodate the motion of the fluid boundary and to maintain a valid moving-mesh discretization, the fluid domain mesh is updated for each time step by solving the equations of elastostatics [59, 117, 123, 125, 126, 128]. The temporal mesh motion is interpolated using periodic cubic spline interpolation for smaller time steps.

4.2.2 BHV constitutive model and boundary conditions

Biological tissues are favored in the construction of prosthetic valves due to their unique mechanical properties. The most important of these is that they remain compliant at low strains but stiffen dramatically when stretched, allowing for ease of motion without sacrificing durability. The underlying structural mechanism is the presence of collagen fibers which are highly undulated in unloaded tissue. These fibers provide only small bending stiffnesses in unloaded



(a) (b) (c)

Figure 4.12: Reference fluid domain mesh generation: (a) A sample mesh of a single patch. (b) Closed mesh of the fluid domain boundary. (c) Volumetric mesh.

tissue, but their relatively larger tensile stiffness can be recruited when they are straightened under strain. One of the earliest and most widely used models, the Fung model, uses an exponential function of strain to describe the stiffening of tissues under tensile loading [38, 120, 129]. For smaller bending strains, such as those in an open aortic BHV during systole, the dominant contribution to material stiffness is the extracellular matrix (ECM), which supports the network of collagen fibers. Fan and Sacks [37] advocates modeling ECM as an incompressible neo-Hookean contribution to the strain-energy density functional. In this work, we combine a Fung model of collagen fiber stiffness with a neo-Hookean model of ECM stiffness to obtain the following strain-energy density functional:

$$\psi_{el} = \frac{c_0}{2}\left(I_1 - 3\right) + \frac{c_1}{2}\left(e^{c_2(I_1-3)^2} - 1\right) \ , \tag{4.1}$$

where c_0, c_1, and c_2 are material parameters. The mass density of the leaflets is set to 1.0 g/mL. The material parameters are set to $c_0 = 5.0 \times 10^6$ dyn/cm^2, $c_1 = 2.0 \times 10^5$ dyn/cm^2, and $c_2 = 100$. Further details of the Fung-type material model can be found in Hsu et al. [50].

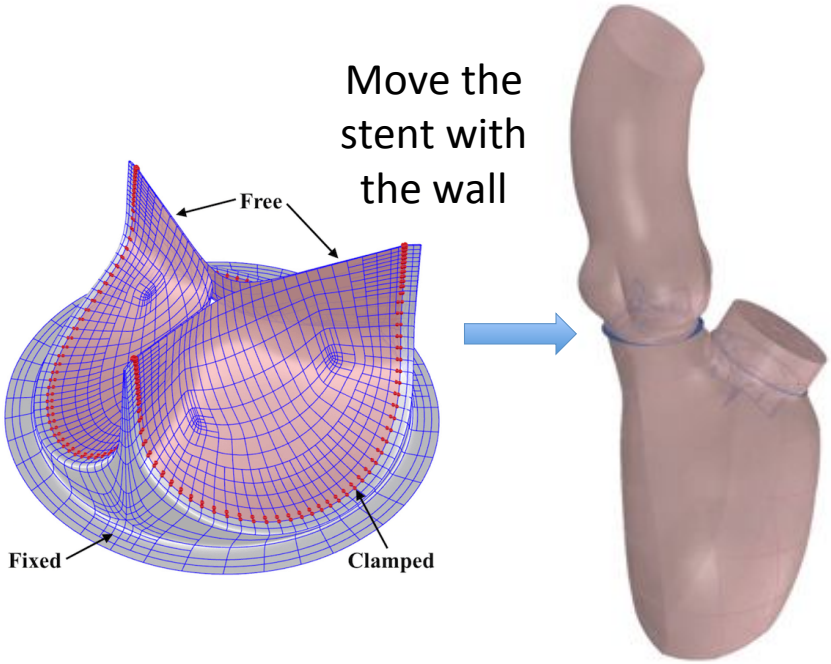
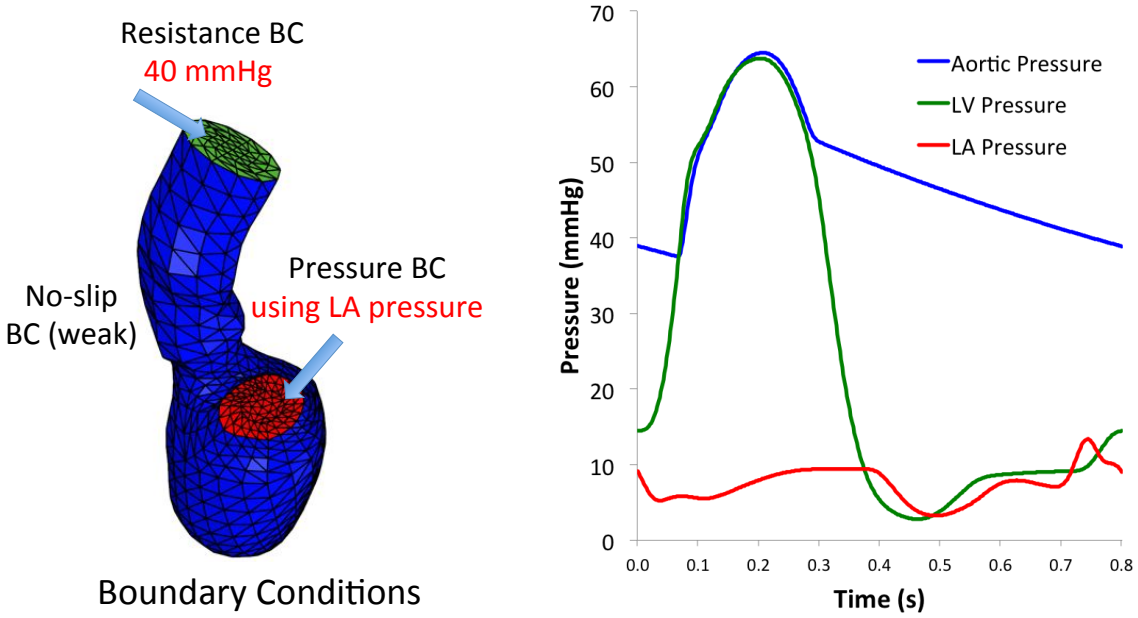


Figure 4.13: BHVs coupled with fluid domain.

The BHV model employs the T-spline geometry constructed in Section 4.1.4. The T-spline mesh for the aortic valve comprises 382 and 1,020 Bézier elements for each leaflet and the stent, respectively, and a total of 2,262 T-spline control points. The T-spline mesh for the mitral valve comprises 354 and 1,020 Bézier elements for each leaflet and the stent, respectively, and a total of 2,169 T-spline control points. The leaflets are passive in FSI simulations and their thickness is set to a uniform value of 0.0386 cm.

The leaflet control points highlighted in Figure 4.13 are restrained from moving with respect to the stent. This clamps the attached edges of the leaflets to the stent. The BHV stent is surgically sutured to the annulus at its suture ring. The size of the ring can influence the potential space for blood flow and thus is important to be included in the FSI simulation. The orientation and the size of the stent are based on the location and the diameter of the ventriculo–aortic junction at each temporal step. The stent is moved based on the movement of the boundary wall. Figure 4.13 shows the geometric intersection of the stent with the fluid domain boundary. The stent seals the gap in the fluid domain between the attached edges of the leaflets and the aortic wall.



(a) Simulation setup. (b) Pressures at Aortic, LV and LA.

Figure 4.14: Simulation setup for the CFD simulations.

4.2.3 Details of the FSI simulation

The simulation setup including the prescribed pressure at the inflow and the boundary conditions are shown in Figure 4.14. In the FSI simulation, we apply the left atrial pressure at each time step based on the simulations from Krishnamurthy et al. [74] (also plotted in Figure 4.14(b)) as a traction boundary condition at the mitral orifice. The applied pressure is periodic with a time period 0.8 s. The traction $-(p_0+RQ)\mathbf{n}$ is applied at the outflow, where p_0 is a constant physiological pressure level, $\mathbf{n}$ is the outward-facing normal of the fluid domain, $R > 0$ is a resistance constant, and Q is the volumetric flow rate through the outflow.

In the present computation, we set $p_0 = 40$ mmHg and $R = 70$ (dyn s)/cm^5. These values ensure a transvalvular pressure difference of 40 mmHg across a closed valve, when $Q = 0$, while permitting a reasonable flow rate during systole. We use backflow stabilization [34], with $\beta = 0.5$, at the outlet which is the top of the ascending aorta. The normal and tangential velocity penalization parameters used in our FSI formulation are $\tau^B_{\mathrm{TAN}} = 2.0 \times 10^3$ g/(cm^2 s) and $\tau^B_{\mathrm{NOR}} = 2.0 \times 10^2$ g/(cm^2 s). As in the earlier studies [49, 62], we set the τ_{M} scaling factor to $s^{\mathrm{shell}} = 10^6$ to obtain acceptable mass conservation near the immersed structure. The

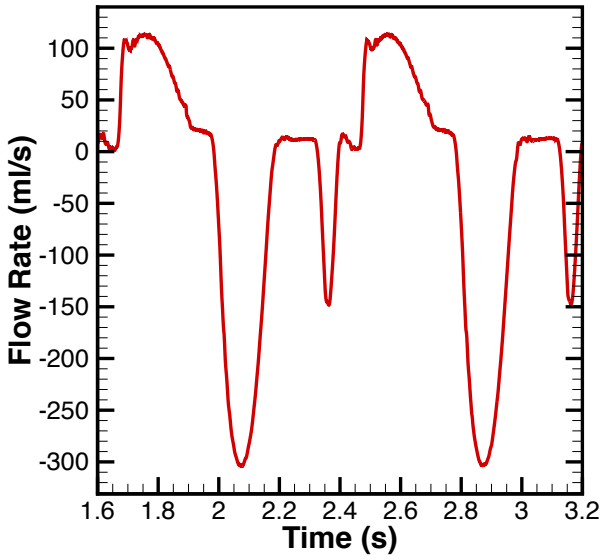


Figure 4.15: Flow rate at the outlet during both third and fourth heart beats.

time step size is $\Delta t = 1.0 \times 10^{-4}$ s. The stabilization parameter of the semi-implicit time integration scheme is $r = 10^{-5}$. This follows the recommendation by Kamensky et al. [61] to select $r \ll 1$. The fluid density and viscosity in the fluid domain are set to $\rho_1 = 1.0$ g/cm^3 and $\mu = 3.0 \times 10^{-2}$ g/(cm s), respectively, which model the physical properties of human blood [64, 93].

4.2.4 FSI simulation results

The 4th heart beat of the FSI simulation shows a relatively steady state result. The cardiac cycle starts from the end-diastole when both aortic and mitral valves are closed which follows the phase of the iso-volumetric contraction. Then during the systole, then the blood inside of the LV passes through the open aortic valve while being the blocked by the closed mitral valve. After the systolic ejection, during iso-volumetric relaxation, both valves close again. Then during diastole, the mitral valve opens which allows the blood to fill in the LV. After the diastolic filling, the the iso-volumetric contraction starts again.

The flow rate at the outlet, which is the top of the ascending aorta, is shown in Figure 4.15. The flow rate shows the systolic ejection. There are some oscillations at the beginning of the simulations, which is caused by the initialization (velocity is zero everywhere) of the fluid domain. With this geometric framework for immersogeometric analysis, we can use the same

simulation setup to test any new BHV design for efficient opening and closing. This will allow for an accelerated optimization of the leaflet design.

4.3 Acknowledgments

Certain sections of Chapter 4 is a reprint of the material as it appears in: "Dynamic and fluidstructure interaction simulations of bioprosthetic heart valves using parametric design with T-splines and Fung-type material models," (with M.-C. Hsu, D. Kamensky, F. Xu, J. Kiendl, MCH. Wu, J. Mineroff, A. Reali, Y. Bazilevs, MS. Sacks), *Computational Mechanics*, 55:1211–1225, 2015.

CHAPTER 5. CONCLUSIONS AND FUTURE WORK

In this dissertation, a geometric framework for immersogeometric analysis was developed. Immersogeometric method allows for fluid flow simulations over complex B-rep CAD models immersed in a locally refined, non-boundary-fitted background fluid mesh. This method avoids the challenges associated with geometry cleanup and mesh generation process. The geometric framework efficiently handles preprocessing of complex objects for CFD and FSI analyses.

A design-through-analysis platform was integrated with this geometric framework which allows rapid development of optimized designs. The platform is built on top of Rhino 3D CAD software, and features several plug-ins to facilitate analysis model creation for IGA, as part of a complete design-through-analysis feedback-control loop. Grasshopper 3D, a visual programming interface, was effectively employed to create parametric designs without writing tedious and "bug-prone" computer programs. In addition, we presented a method to enable direct visualization of NURBS and T-spline meshes and solutions defined on these meshes directly in Rhino 3D.

The geometric framework was developed to support fluid-flow simulations over B-reps of a complex object. B-reps of complex objects usually consist of both parametric and analytic surfaces. This geometric framework directly used the parametric and analytic surface equations to generate the surface Gaussian quadrature points, which were used for weak enforcement of Dirichlet boundary conditions. The analytic surfaces are preprocessed directly without converting to NURBS, which are computationally expensive and can lead to poorly parameterized or converted surfaces. We also performed adaptive quadrature on both trimmed parametric and analytic surfaces to maintain the accuracy of the integration around trim curves. The timing results showed that the analytic-surface-based preprocessing method is much faster in computing the Gauss point information than using only NURBS surfaces. Finally, we have developed

a method to generate a fluid domain mesh with selective refinement around the surfaces of the immersed object using hierarchical voxelization of the object.

To validate the accuracy of the proposed method, we performed simulations of benchmark problems of flow over a sphere and a torpedo shaped object represented using B-reps. Quantities of interest such as drag coefficient for the sphere were in good agreement with reference values reported in literature. In addition, the flow simulation quantities of interest obtained using immersogeometric analytic surfaces were comparable to the reference flow simulation results obtained using NURBS and boundary-fitted CFD. The results showed that the density and distribution of the surface quadrature points are crucial for accurate flow analysis. Also, with sufficient levels of surface quadrature element refinement, the quadrature error near the trim curves become insignificant.

The effectiveness of the immersogeometric method for industrial scale simulations is demonstrated by performing aerodynamic analysis of an agricultural tractor and semi-trailer truck directly represented using parametric and analytic surfaces. The tractor analysis indicated that the NURBS-based immersogeometric method can greatly simplify the mesh generation process for industrial turbulent flow problems without sacrificing solution accuracy. In addition, the analytic-surface-based immersogeometric method can be easily applied to flow analysis of industrial B-rep objects made with many analytic surfaces. In the simulation of flow past the semi-trailer truck, we directly used the CAD model generated using SolidWorks to perform immersogeometric analysis. The drag coefficient computed using the immersogeometric method was within the range of drag coefficient values of semi-trail trucks reported in the literature.

Finally, a FSI simulation of a moving left ventricle (LV) coupled with two bioprosthetic heart valves (BHVs) showed the potential utility of this geometric framework for designing BHV leaflets. We combined the parametric design platform with the immersogeometric FSI methodology proposed by Hsu et al. [49] and Kamensky et al. [62] to perform high-fidelity BHV FSI with patient-specific left ventricular geometry. The present effort represented the first step toward automated optimization of the leaflet geometry using flow driven by the deformation of the LV.

The work shown in this dissertation presents a first step in making immersogeometric analysis accessible to design engineers and analysts. Possible future research directions in the development of immersogeometric analysis include an efficient way to construct patient-specific cardiac models, a general algorithm for parametric design of the mitral valve, and efficient treatment of moving immersed objects. Finally, the proposed geometric framework needs to be versatile and robust enough to handle realistic engineering designs in all of their complexity.

We have successfully developed a geometric framework for immersogeometric analysis that integrates CAD and CAE. This geometric framework enables directly using the B-reps of CAD model for CFD and FSI simulations. We envision this geometric framework will integrate design and analysis tools that will help design engineers avoid the time-consuming boundary-fitted mesh generation process. These efficient modeling tools can be used to identify an optimal design of a product that can reduce its time-to-market. However, such framework requires a paradigm shift from the traditional linear CAD and CAE analysis workflow to a more integrated one. We hope that our geometric framework for immersogeometric analysis would fulfill this important role in this field.

Bibliography

[1] Akkerman, I., Bazilevs, Y., Calo, V. M., Hughes, T. J. R., and Hulshoff, S. (2008). The role of continuity in residual-based variational multiscale modeling of turbulence. *Computational Mechanics*, 41:371–378.

[2] Akkerman, I., Bazilevs, Y., Kees, C. E., and Farthing, M. W. (2011). Isogeometric analysis of free-surface flow. *Journal of Computational Physics*, 230:4137–4152.

[3] Auricchio, F., Beirão da Veiga, L., Hughes, T. J. R., Reali, A., and Sangalli, G. (2010). Isogeometric collocation methods. *Mathematical Models and Methods in Applied Sciences*, 20(11):2075–1077.

[4] Auricchio, F., Calabrò, F., Hughes, T. J. R., Reali, A., and Sangalli, G. (2012). A simple algorithm for obtaining nearly optimal quadrature rules for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 249-252:15–27.

[5] Autodesk T-Splines Plug-in for Rhino (2017). `http://www.tsplines.com/products/tsplines-for-rhino.html`. 2017.

[6] Azamatov, A., Lee, J.-W., and Byun, Y.-H. (2011). Comprehensive aircraft configuration design tool for integrated product and process development. *Advances in Engineering Software*, 42:35–49.

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

[8] Bazilevs, Y., Calo, V. M., Cottrel, J. A., Hughes, T. J. R., Reali, A., and Scovazzi, G. (2007a). Variational multiscale residual-based turbulence modeling for large eddy simulation

of incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 197:173–201.

[9] Bazilevs, Y., Calo, V. M., Cottrell, J. A., Evans, J. A., Hughes, T. J. R., Lipton, S., Scott, M. A., and Sederberg, T. W. (2010a). Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199:229–263.

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

[11] Bazilevs, Y., Hsu, M.-C., and Scott, M. A. (2012). Isogeometric fluid–structure interaction analysis with emphasis on non-matching discretizations, and with application to wind turbines. *Computer Methods in Applied Mechanics and Engineering*, 249–252:28–41.

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

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

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

[15] Beall, M. W., Walsh, J., and Shephard, M. S. (2004). A comparison of techniques for geometry access related to mesh generation. *Engineering with Computers*, 20:210–221.

[16] Benson, D. J., Bazilevs, Y., De Luycker, E., Hsu, M.-C., Scott, M., Hughes, T. J. R., and Belytschko, T. (2010a). A generalized finite element formulation for arbitrary basis functions: from isogeometric analysis to XFEM. *International Journal for Numerical Methods in Engineering*, 83:765–785.

[17] Benson, D. J., Bazilevs, Y., Hsu, M.-C., and Hughes, T. J. R. (2010b). Isogeometric shell analysis: The Reissner–Mindlin shell. *Computer Methods in Applied Mechanics and Engineering*, 199:276–289.

[18] BETA CAE Systems S.A. – ANSA pre-processor (2016). `http://www.beta-cae.gr/ansa.htm`. 2015.

[19] Borden, M. J., Hughes, T. J. R., Landis, C. M., and Verhoosel, C. V. (2014). A higher-order phase-field model for brittle fracture: Formulation and analysis within the isogeometric analysis framework. *Computer Methods in Applied Mechanics and Engineering*, 273:100–118.

[20] Borden, M. J., Scott, M. A., Evans, J. A., and Hughes, T. J. R. (2011). Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, 87:15–47.

[21] Breitenberger, M., Apostolatos, A., Philipp, B., Wüchner, R., and Bletzinger, K.-U. (2015). Analysis in computer aided design: Nonlinear isogeometric B-Rep analysis of shell structures. *Computer Methods in Applied Mechanics and Engineering*, 284:401–457.

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

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

[24] Chowdhury, H., Moria, H., Ali, A., Khan, I., Alam, F., and S., W. (2013). A study on aerodynamic drag of a semi-trailer truck. *Procedia Engineering*, 56:201–205.

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

[26] Cohen, E., Martin, T., Kirby, R. M., Lyche, T., and Riesenfeld, R. F. (2010). Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 199:334–356.

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

[28] Cottrell, J. A., Hughes, T. J. R., and Reali, A. (2007). Studies of refinement and continuity in isogeometric structural analysis. *Computer Methods in Applied Mechanics and Engineering*, 196:4160–4183.

[29] Cottrell, J. A., Reali, A., Bazilevs, Y., and Hughes, T. J. R. (2006). Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering*, 195:5257–5297.

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

[31] Dedè, L., Borden, M. J., and Hughes, T. J. R. (2012). Isogeometric analysis for topology optimization with a phase field model. *Archives of Computational Methods in Engineering*, 19(3):427–465.

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

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

[34] Esmaily-Moghadam, M., Bazilevs, Y., Hsia, T.-Y., Vignon-Clementel, I. E., Marsden, A. L., and of Congenital Hearts Alliance (MOCHA), M. (2011). A comparison of outlet boundary treatments for prevention of backflow divergence with relevance to blood flow simulations. *Computational Mechanics*, 48:277–291.

[35] Evans, E. J., Scott, M. A., Li, X., and Thomas, D. C. (2015). Hierarchical T-splines: Analysis-suitability, Bézier extraction, and application as an adaptive basis for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 284:1–20.

[36] Evans, J. A. and Hughes, T. J. R. (2013). Isogeometric divergence-conforming B-splines for the unsteady Navier–Stokes equations. *Journal of Computational Physics*, 241:141–167.

[37] Fan, R. and Sacks, M. S. (2014). Simulation of planar soft tissues using a structural constitutive model: Finite element implementation and validation. *Journal of Biomechanics*, 47(9):2043–2054.

[38] Fung, Y. C. (1993). *Biomechanics: Mechanical Properties of Living Tissues*. Springer-Verlag, New York, second edition.

[39] Gomez, H., Calo, V. M., Bazilevs, Y., and Hughes, T. J. R. (2008). Isogeometric analysis of the Cahn–Hilliard phase-field model. *Computer Methods in Applied Mechanics and Engineering*, 197:4333–4352.

[40] Götz, H. and Mayr, G. (1998). Chapter 9 - commercial vehicles. In Hucho, W., editor, *Aerodynamics of Road Vehicles: from fluid mechanics to vehicle engineering*, pages 415–488. Society of Automotive Engineers Inc.

[41] Grasshopper (2017). `http://www.grasshopper3d.com/`. 2017.

[42] Guo, Y. and Ruess, M. (2015). Nitsche's method for a coupling of isogeometric thin shells and blended shell structures. *Computer Methods in Applied Mechanics and Engineering*, 284:881–905.

[43] Hand, M. M., Simms, D. A., Fingersh, L. J., Jager, D. W., Cotrell, J. R., Schreck, S., and Larwood, S. M. (2001). Unsteady aerodynamics experiment phase VI: Wind tunnel test configurations and available data campaigns. Technical Report NREL/TP-500-29955, National Renewable Energy Laboratory, Golden, CO.

[44] Hanniel, I. and Haller, K. (2011). Direct rendering of solid CAD models on the GPU. In *12th International Conference on Computer-Aided Design and Computer Graphics*, pages 25–32. IEEE.

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

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

[47] Hsu, M.-C., Akkerman, I., and Bazilevs, Y. (2014a). Finite element simulation of wind turbine aerodynamics: Validation study using NREL Phase VI experiment. *Wind Energy*. doi:10.1002/we.1599.

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

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

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

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

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

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

[54] Hughes, T. J. R., Reali, A., and Sangalli, G. (2010). Efficient quadrature for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 199:301–313.

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

[56] J. Englar, R. (2001). Advanced aerodynamic devices to improve the performance, economics, handling and safety of heavy vehicles. In *SAE Technical Paper*. SAE International.

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

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

[59] Johnson, A. A. and Tezduyar, T. E. (1994). Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. *Computer Methods in Applied Mechanics and Engineering*, 119:73–94.

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

[61] Kamensky, D., Evans, J. A., and Hsu, M.-C. (2015a). Stability and conservation properties of collocated constraints in immersogeometric fluid-thin structure interaction analysis. *Communications in Computational Physics*, 18:1147–1180.

[62] Kamensky, D., Hsu, M.-C., Schillinger, D., Evans, J. A., Aggarwal, A., Bazilevs, Y., Sacks, M. S., and Hughes, T. J. R. (2015b). An immersogeometric variational framework for

fluid–structure interaction: Application to bioprosthetic heart valves. *Computer Methods in Applied Mechanics and Engineering*, 284:1005–1053.

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

[64] Kenner, T. (1989). The measurement of blood density and its meaning. *Basic Research in Cardiology*, 84(2):111–124.

[65] Kiendl, J. (2011). *Isogeometric Analysis and Shape Optimal Design of Shell Structures*. PhD thesis, Lehrstuhl für Statik, Technische Universität München.

[66] Kiendl, J., Bazilevs, Y., Hsu, M.-C., Wüchner, R., and Bletzinger, K.-U. (2010). The bending strip method for isogeometric analysis of Kirchhoff–Love shell structures comprised of multiple patches. *Computer Methods in Applied Mechanics and Engineering*, 199:2403–2416.

[67] Kiendl, J., Bletzinger, K.-U., Linhard, J., and Wüchner, R. (2009). Isogeometric shell analysis with Kirchhoff–Love elements. *Computer Methods in Applied Mechanics and Engineering*, 198:3902–3914.

[68] Kiendl, J., Schmidt, R., Wüchner, R., and Bletzinger, K.-U. (2014). Isogeometric shape optimization of shells using semi-analytical sensitivity analysis and sensitivity weighting. *Computer Methods in Applied Mechanics and Engineering*, 274:148–167.

[69] Kim, H., Lu, J., Sacks, M. S., and Chandran, K. B. (2008). Dynamic simulation of bioprosthetic heart valves using a stress resultant shell model. *Annals of Biomedical Engineering*, 36(2):262–275.

[70] Kostas, K. V., Ginnis, A. I., Politis, C. G., and Kaklis, P. D. (2015). Ship-hull shape optimization with a T-spline based BEM–isogeometric solver. *Computer Methods in Applied Mechanics and Engineering*, 284:611–622.

[71] Krishnamurthy, A., Khardekar, R., and McMains, S. (2009a). Optimized GPU evaluation of arbitrary degree NURBS curves and surfaces. *Computer Aided Design*, 41(12):971–980.

[72] Krishnamurthy, A., Khardekar, R., McMains, S., Haller, K., and Elber, G. (2009b). Performing efficient NURBS modeling operations on the GPU. *IEEE Transactions on Visualization and Computer Graphics*, 15(4):530–543.

[73] Krishnamurthy, A., McMains, S., and Haller, K. (2011). GPU-accelerated minimum distance and clearance queries. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):729–742.

[74] Krishnamurthy, A., Villongco, C. T., Chuang, J., Frank, L. R., Nigam, V., Belezzuoli, E., Stark, P., Krummen, D. E., Narayan, S., Omens, J. H., et al. (2013). Patient-specific models of cardiac biomechanics. *Journal of Computational Physics*, 244:4–21.

[75] LaBozetta, W. F. and Cole, P. E. (1985). Interactive graphics for geometry generation—A program with a contemporary design. *Journal of Aircraft*, 22:1054–1058.

[76] Lee, Y. K., Lim, C. K., Ghazialam, H., Vardhan, H., and Eklund, E. (2010). Surface mesh generation for dirty geometries by the Cartesian shrink-wrapping technique. *Engineering with Computers*, 26:377–390.

[77] Li, X. and Scott, M. A. (2014). Analysis-suitable T-splines: Characterization, refineability, and approximation. *Mathematical Models and Methods in Applied Sciences*, 24(06):1141–1164.

[78] Liu, J., Gómez, H., Evans, J. A., Hughes, T. J. R., and Landis, C. M. (2013). Functional entropy variables: A new methodology for deriving thermodynamically consistent algorithms for complex fluids, with particular reference to the isothermal Navier–Stokes–Korteweg equations. *Journal of Computational Physics*, 248:47–86.

[79] Liu, L., Zhang, Y., Hughes, T. J. R., Scott, M. A., and Sederberg, T. W. (2014). Volumetric T-spline construction using Boolean operations. *Engineering with Computers*, 30(4):425–439.

[80] Lhner, R. and Parikh, P. (1988). Generation of three-dimensional unstructured grids by the advancing-front method. *International Journal for Numerical Methods in Fluids*, 8(10):1135–1149.

[81] Marcum, D. L. and Gaither, J. A. (2000). Unstructured grid generation for aerospace applications. In Salas, M. D. and Anderson, W. K., editors, *Computational Aerosciences in the 21st Century*, volume 8, pages 189–209. Springer Netherlands.

[82] Morganti, S., Valentini, A., Favalli, V., Serio, A., Gambarin, F. I., Vella, D., Mazzocchi, L., Massetti, M., Auricchio, F., and Arbustini, E. (2013). Aortic root 3d parametric morphological model from 2d-echo images. *Computers in Biology and Medicine*, 43(12):2196–2204.

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

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

[85] Piegl, L. and Tiller, W. (1997). *The NURBS Book (Monographs in Visual Communication), 2nd ed.* Springer-Verlag, New York.

[86] Rank, E., Ruess, M., Kollmannsberger, S., Schillinger, D., and Düster, A. (2012). Geometric modeling, isogeometric analysis and the finite cell method. *Computer Methods in Applied Mechanics and Engineering*, 249-250:104–115.

[87] Reali, A. and Gómez, H. (2015). An isogeometric collocation approach for Bernoulli–Euler beams and Kirchhoff plates. *Computer Methods in Applied Mechanics and Engineering*, 284:623–636.

[88] Requicha, A. A. G. and Voelcker, H. B. (1985). Boolean operations in solid modeling: Boundary evaluation and merging algorithms. *Proceedings of the IEEE*, 73(1):30–44.

[89] Rhino Developer Tools (2017). `http://wiki.mcneel.com/developer/home`. 2017.

[90] Rhinoceros (2016). `http://www.rhino3d.com/`. 2016.

[91] RhinoCommon Plug-in SDK (2017). `http://wiki.mcneel.com/developer/rhinocommon/`. 2017.

[92] Roman, M. J., Devereux, R. B., Kramer-Fox, R., and O'Loughlin, J. (1989). Two-dimensional echocardiographic aortic root dimensions in normal children and adults. *The American Journal of Cardiology*, 64(8):507–512.

[93] Rosencranz, R. and Bogen, S. A. (2006). Clinical laboratory measurement of serum, plasma, and blood viscosity. *Am. J. Clin. Pathol.*, 125 Suppl:78–86.

[94] Rossignac, J. R. and Requicha, A. A. G. (1999). Solid modeling. Technical Report GIT-GVU-99-09, Georgia Institute of Technology.

[95] Ruess, M., Schillinger, D., Özcan, A. I., and Rank, E. (2014). Weak coupling for isogeometric analysis of non-matching and trimmed multi-patch geometries. *Computer Methods in Applied Mechanics and Engineering*, 269:46–731.

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

[97] Saura, D., Dulgheru, R., Caballero, L., Bernard, A., Kou, S., Gonjilashvili, N., Athanassopoulos, G. D., Barone, D., Baroni, M., Cardim, N., Hagendorff, A., Hristova, K., Lopez, T., de la Morena, G., Popescu, B. A., Penicka, M., Ozyigit, T., Rodrigo C., J. D., Van De Veire, N., Von B., R. S., Vinereanu, D., Zamorano, J. L., Gori, A.-S., Cosyns, B., Donal, E., Habib, G., Addetia, K., Lang, R. M., Badano, L. P., and Lancellotti, P. (2016). Two-dimensional transthoracic echocardiographic normal reference ranges for proximal aorta dimensions: results from the eacvi norre study. *European Heart Journal - Cardiovascular Imaging*.

[98] Schillinger, D., Dedè, L., Scott, M. A., Evans, J. A., Borden, M. J., Rank, E., and Hughes, T. J. R. (2012a). An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 249–252:116–150.

[99] Schillinger, D., Evans, J. A., Reali, A., Scott, M. A., and Hughes, T. J. R. (2013). Isogeometric collocation: Cost comparison with Galerkin methods and extension to adaptive

hierarchical NURBS discretizations. *Computer Methods in Applied Mechanics and Engineering*, 267:170–232.

[100] Schillinger, D., Hossain, S. J., and Hughes, T. J. R. (2014). Reduced Bézier element quadrature rules for quadratic and cubic splines in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 277:1–45.

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

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

[103] Schmidt, R., Wüchner, R., and Bletzinger, K.-U. (2012). Isogeometric analysis of trimmed NURBS geometries. *Computer Methods in Applied Mechanics and Engineering*, 241-244:93–111.

[104] Scott, M. A., Borden, M. J., Verhoosel, C. V., Sederberg, T. W., and Hughes, T. J. R. (2011). Isogeometric finite element data structures based on Bézier extraction of T-splines. *International Journal for Numerical Methods in Engineering*, 88:126–156.

[105] Scott, M. A., Hughes, T. J. R., Sederberg, T. W., and Sederberg, M. T. (2014). An integrated approach to engineering design and analysis using the Autodesk T-spline plugin for Rhino3d. ICES REPORT 14-33, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, September 2014.

[106] Scott, M. A., Li, X., Sederberg, T. W., and Hughes, T. J. R. (2012). Local refinement of analysis-suitable T-splines. *Computer Methods in Applied Mechanics and Engineering*, 213-216:206–222.

[107] Scott, M. A., Simpson, R. N., Evans, J. A., Lipton, S., Bordas, S. P. A., Hughes, T. J. R., and Sederberg, T. W. (2013). Isogeometric boundary element analysis using unstructured T-splines. *Computer Methods in Applied Mechanics and Engineering*, 254:197–221.

[108] Sederberg, T., Zheng, J., Bakenov, A., and Nasri, A. (2003). T-splines and T-NURCCS. *ACM Transactions on Graphics*, 22(3):477–484.

[109] Sederberg, T. W., Cardon, D., Finnigan, G., North, N., Zheng, J., and Lyche, T. (2004). T-spline simplification and local refinement. *ACM Transactions on Graphics*, 23(3):276–283.

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

[111] SIACOM (2012). CFD challenge: simulation of hemodynamics in a patient-specific aortic coarctation model. `http://www.vascularmodel.org/miccai2012/`.

[112] Simpson, R. N., Bordas, S. P. A., Trevelyan, J., and Rabczuk, T. (2012). A two-dimensional Isogeometric Boundary Element Method for elastostatic analysis. *Computer Methods in Applied Mechanics and Engineering*, 209-212:87–100.

[113] SolidWorks Corp. (2015). SolidWorks SDK.

[114] Somers, D. M. (1997). Design and experimental results for the S809 airfoil. Technical Report NREL/SR-440-6918, National Renewable Energy Laboratory, Golden, CO.

[115] Spatial Technology, I. (1996). *ACIS Geometric Modeler: Application Guide*, version 2.0 edition.

[116] Stavric, M. and Marina, O. (2011). Parametric modeling for advanced architecture. *International Journal of Applied Mathematics and Informatics*, 5(1):9–16.

[117] Stein, K., Tezduyar, T., and Benney, R. (2003). Mesh moving techniques for fluid–structure interactions with large displacements. *Journal of Applied Mechanics*, 70:58–63.

[118] Stein, P., Hsu, M.-C., Bazilevs, Y., and Beucke, K. (2012). Operator- and template-based modeling of solid geometry for isogeometric analysis with application to vertical axis wind turbine simulation. *Computer Methods in Applied Mechanics and Engineering*, 213–216:71–83.

[119] Sun, W., Abad, A., and Sacks, M. S. (2005). Simulated bioprosthetic heart valve deformation under quasi-static loading. *J Biomech Eng*, 127(6):905–914.

[120] Sun, W., Sacks, M. S., Sellaro, T. L., Slaughter, W. S., and Scott, M. J. (2003). Biaxial mechanical response of bioprosthetic heart valve biomaterials to high in-plane shear. *Journal of Biomechanical Engineering*, 125(3):372–380.

[121] Technavio (2017). Global computer-aided design market 2017-2021. Technical Report IRTNTR11637, Technavio.

[122] Temizer, İ., Wriggers, P., and Hughes, T. J. R. (2012). Three-dimensional mortar-based frictional contact treatment in isogeometric analysis with NURBS. *Computer Methods in Applied Mechanics and Engineering*, 209-212:115–128.

[123] Tezduyar, T., Aliabadi, S., Behr, M., Johnson, A., and Mittal, S. (1993). Parallel finite-element computation of 3D flows. *Computer*, 26(10):27–36.

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

[125] Tezduyar, T. E. (2001). Finite element methods for flow problems with moving boundaries and interfaces. *Archives of Computational Methods in Engineering*, 8:83–130.

[126] Tezduyar, T. E., Behr, M., Mittal, S., and Johnson, A. A. (1992). Computation of unsteady incompressible flows with the finite element methods – space–time formulations, iterative strategies and massively parallel implementations. In *New Methods in Transient Analysis*, PVP-Vol.246/AMD-Vol.143, pages 7–24, New York. ASME.

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

[128] Tezduyar, T. E. and Sathe, S. (2007). Modeling of fluid–structure interactions with the space–time finite elements: Solution techniques. *International Journal for Numerical Methods in Fluids*, 54:855–900.

[129] Tong, P. and Fung, Y.-C. (1976). The stress-strain relationship for the skin. *Journal of Biomechanics*, 9(10):649 – 657.

[130] Trapp, J. C. and Sobiezsky, H. (1999). Interactive parametric geometry design. In *Proceedings of 37th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA Paper 99-0829, Reno, NV.

[131] Varduhn, V., Hsu, M.-C., Ruess, M., and Schillinger, D. (2016). The tetrahedral finite cell method: Higher-order immersogeometric analysis on adaptive non-boundary-fitted meshes. *International Journal for Numerical Methods in Engineering*.

[132] Wang, W. and Zhang, Y. (2010). Wavelets-based NURBS simplification and fairing. *Computer Methods in Applied Mechanics and Engineering*, 199:290–300.

[133] Wang, W., Zhang, Y., Liu, L., and Hughes, T. J. R. (2013). Trivariate solid T-spline construction from boundary triangulations with arbitrary genus topology. *Computer-Aided Design*, 45(2):351–360.

[134] Wang, Z. J. and Srinivasan, K. (2002). An adaptive Cartesian grid generation method for 'Dirty' geometry. *International Journal for Numerical Methods in Fluids*, 39:703–717.

[135] Wu, M. C. H., Kamensky, D., Wang, C., Herrema, A. J., Xu, F., Pigazzini, M. S., Verma, A., Marsden, A. L., Bazilevs, Y., and Hsu, M.-C. (2017). Optimizing fluid–structure interaction systems with immersogeometric analysis and surrogate modeling: Application to a hydraulic arresting gear. *Computer Methods in Applied Mechanics and Engineering*, 316:668–693.

[136] Xu, F., Schillinger, D., Kamensky, D., Varduhn, V., Wang, C., and Hsu, M.-C. (2016). The tetrahedral finite cell method for fluids: Immersogeometric analysis of turbulent flow around complex geometries. *Computers & Fluids.*

[137] Zhang, Y., Bazilevs, Y., Goswami, S., Bajaj, C., and Hughes, T. J. R. (2007). Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. *Computer Methods in Applied Mechanics and Engineering*, 196:2943–2959.

[138] Zhang, Y., Wang, W., and Hughes, T. J. R. (2013). Conformal solid T-spline construction from boundary T-spline representations. *Computational Mechanics*, 51(6):1051–1059.

[139] Zion Market Research (2016). Computer aided engineering market (finite element analysis and computational fluid dynamics) for aerospace, automobile, electronic and electricals defense, industrial machineries and other applications: Global industry perspective, comprehensive analysis and forecast, 2015-2021. Technical Report ZMR-194, Zion Market Research.