Technische Universität München

Ingenieurfakultät Bau Geo Umwelt

Lehrstuhl für Statik

ON THE GRID-BASED SHAPE OPTIMIZATION OF STRUCTURES WITH INTERNAL FLOW AND THE FEEDBACK OF SHAPE CHANGES INTO A CAD MODEL

Daniel Baumgärtner

Vollständiger Abdruck der von der Ingenieurfakultät Bau Geo Umwelt der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender:

Univ.-Prof. Dr.-Ing. habil. Fabian Duddeck

Prüfer der Dissertation:

1. Univ.-Prof. Dr.-Ing. Kai-Uwe Bletzinger
2. Univ.-Prof. Dr.rer.nat. Nicolas Gauger
3. Prof. Dr. Pooyan Dadvand

Die Dissertation wurde am 15.06.2020 bei der Technischen Universität München eingereicht und durch die Ingenieurfakultät Bau Geo Umwelt am 26.11.2020 angenommen.

Schriftenreihe des Lehrstuhls für Statik TU München

Band 46

Daniel Baumgärtner

ON THE GRID-BASED SHAPE OPTIMIZATION OF STRUCTURES WITH INTERNAL FLOW AND THE FEEDBACK OF SHAPE CHANGES INTO A CAD MODEL

München 2020

Veröffentlicht durch

Kai-Uwe Bletzinger
Lehrstuhl für Statik
Technische Universität München
Arcisstr. 21
80333 München

Telefon:	+49(0)89 289 22422
Telefax:	+49(0)89 289 22421
E-Mail:	kub@tum.de
Internet:	www.bgu.tum.de/st

ISBN: 978-3-943683-63-9

Abstract

When designing structures with internal flow, a typical task is to optimize specific flow characteristics, such as the pressure loss, without violating stress constraints in the structure. The task is challenging if the stresses are influenced by the fluid-structure interaction. The first part of this thesis presents a method, which solves this task by gradient-based shape optimization. The method assumes a one-way coupling and relies on high-fidelity models. An essential ingredient of the method is a grid-based parameterization of the shape using Vertex Morphing. This parameterization enables a quick set-up of the optimization problem and provides high optimization potential. So far, Vertex Morphing was primarily used in optimization problems, which involve a single physical discipline. This work extends Vertex Morphing for an application in multidisciplinary problems. Two extensions are presented: one to control several meshes simultaneously, and one to control volume meshes. The latter is used to avoid a separate adaption of the structure mesh after a shape update. The thesis shows that Vertex Morphing allows a consistent parameterization across multiple grids without extra modeling effort. In order to determine the required gradients, coupled adjoint sensitivity analysis is applied. In this context, a partitioned solution strategy is presented, which avoids the computation of cross-derivatives. The approach allows the use of already established adjoint solvers to compute the coupled gradients. The sensitivity analysis is verified using a 2D and 3D example. Finally, the overall optimization process is demonstrated with a representative part. The results show that one may significantly optimize structures with internal flow using the presented method, even with strongly conflicting response functions.

The result of a shape optimization with Vertex Morphing is a discrete free-form surface. In contrast, modern design processes often rely on computer-aided design (CAD). In CAD, however, free-form surfaces are typically described by non-uniform rational B-splines (NURBS). Therefore, it is often necessary to convert the obtained discrete free-form surface into a NURBS-based CAD model. In the second part of this thesis, a method is presented, which automatizes this step. The method is referred to as B-Rep morphing. The central idea in B-Rep morphing is to deform an existing CAD model of the initial design to match the optimization results. The deformation is achieved by mapping the optimization results onto the given

CAD model. Unlike many other approaches, B-Rep morphing considers general boundary-representation (B-Rep) models with trimmed surfaces and multiple patches. Therefore, the mapping operation is extended by constraints. Coupling constraints are introduced to preserve the continuity between surfaces as much as possible. Design constraints are suggested to consider design requirements that are independent of the optimization results. A multi-stage regularization is applied to enforce smooth surfaces and a practical layout of the control points. The method's capabilities are demonstrated at different examples ranging from a generic test case over a valve housing with internal flow to a complex sheet metal part (car door). The influence of the refinement of the CAD model is discussed. The results show: B-Rep morphing allows the seamless integration of grid-based shape optimization into a CAD process. B-Rep morphing is applicable in all cases where a CAD model must be adapted to a simulated deformation. With B-Rep morphing, a high surface quality may be achieved in the deformed CAD model.

Zusammenfassung

Bei der Konstruktion von Strukturen mit Innenströmung besteht eine typische Aufgabe darin, bestimmte Eigenschaften der Strömung zu optimieren, z.B. den Druckverlust, ohne dabei Spannungsnebenbedingungen in der Struktur zu verletzen. Die Aufgabe ist anspruchsvoll, wenn die Spannungen durch die Fluid-Struktur-Wechselwirkung beeinflusst werden. Im ersten Teil dieser Arbeit wird eine Methode vorgestellt, die diese Aufgabe durch gradientenbasierte Formoptimierung löst. Die Methode geht von einer einseitigen Kopplung aus und stützt sich auf detaillierte Simulationsmodelle. Ein wesentlicher Bestandteil der Methode ist eine gitterbasierte Parametrisierung der Form mittels Vertex Morphing. Diese Parametrisierung ermöglicht einen schnellen Aufbau des Optimierungsproblems und bietet ein hohes Optimierungspotenzial. Bislang wurde Vertex Morphing vor allem bei Optimierungsproblemen eingesetzt, die einzelne physikalische Disziplinen betreffen. Diese Arbeit erweitert Vertex-Morphing für eine Anwendung im multidisziplinären Kontext. Es werden zwei Erweiterungen vorgestellt: eine zur gleichzeitigen Steuerung mehrerer Netze, und eine zur Steuerung von Volumennetzen. Letztere wird genutzt, um eine separate Anpassung des Strukturgitters nach einer Formänderung zu vermeiden. Die Arbeit zeigt, dass Vertex Morphing eine konsistente Parametrisierung über mehrere Gitter hinweg und ohne zusätzlichen Modellierungsaufwand ermöglicht. Zur Bestimmung der erforderlichen Gradienten wird eine gekoppelte adjungierte Sensitivitätsanalyse genutzt. In diesem Zusammenhang wird eine partitionierte Lösungsstrategie vorgestellt, die die Berechnung von Ableitungen über mehrere Domänen hinweg vermeidet. Damit ist es möglich, die gekoppelten Gradienten mit bereits etablierten adjungierten Lösern zu berechnen. Die Sensitivitätsanalyse wird anhand eines 2D- und 3D-Beispiels verifiziert. Schließlich wird der gesamte Optimierungsprozess an einem repräsentativen Bauteil demonstriert. Die Ergebnisse zeigen, dass es mit der vorgestellten Methode möglich ist, Strukturen mit Innenströmung deutlich zu optimieren, auch bei stark widersprüchlicher Antwortfunktionen.

Das Ergebnis einer Formoptimierung mittels Vertex Morphing ist eine diskrete Freiformfläche. Moderne Entwicklungsprozesse bauen dagegen häufig auf eine rechnergestützte Konstruktion (CAD), worin Freiformflächen typischerweise durch „non-uniform rational B-Splines“ (NURBS) beschrieben werden. Daher muss die diskrete Freiformfläche oft in ein NURBS-basiertes CAD-Modell überführt werden. Im zweiten Teil dieser Arbeit wird eine Methode vorgestellt, die diesen Schritt automatisiert. Die Methode wird als B-Rep Morphing bezeichnet. Die zentrale Idee von B-Rep Morphing ist es, ein vorhandenes CAD-Modell des ursprünglichen Entwurfs so zu verformen, dass es mit den Ergebnissen der Optimierung übereinstimmt. Die Verformung wird dabei durch eine Abbildung des Optimierungsergebnisses auf dem gegebenen CAD-Modell erreicht. Im Gegensatz zu vielen anderen Ansätzen berücksichtigt das B-Rep Morphing allgemeine Begrenzungsflächenmodelle („B-Rep models“) mit getrimmten Oberflächen und vielen Patches. Dazu wird der Abbildungsvorgang um Nebenbedingungen erweitert. Kopplungsbedingungen werden eingeführt, um die Kontinuität zwischen Oberflächen weitestgehend zu erhalten. Konstruktionsbedingungen werden vorgeschlagen, um von der Optimierung unabhängige Anforderungen an das Design zu berücksichtigen. Eine mehrstufige Regularisierung wird angewandt, um glatte Oberflächen und eine zweckmäßige Anordnung der Kontrollpunkte zu erzwingen. Die Eigenschaften der Methode werden an verschiedenen Beispielen demonstriert. Die Beispiele reichen von einem generischen Testfall über ein Ventilgehäuse mit Innenströmung bis hin zu einem komplexen Blechbauteil (Autotür). Der Einfluss der Verfeinerung des CAD-Modells wird diskutiert. Die Ergebnisse zeigen: Mit B-Rep Morphing ist es möglich, eine gitterbasierter Formoptimierung nahtlos in einen CAD-Prozess zu integrieren. B-Rep Morphing ist in allen Fällen anwendbar, in denen ein CAD-Modell an eine simulierte Verformung angepasst werden muss. Mit B-Rep Morphing kann eine hohe Oberflächenqualität im verformten CAD-Modell erreicht werden.

Acknowledgments

First and foremost, I want to sincerely thank Prof. Dr.-Ing. Kai-Uwe Bletzinger for giving me the chance to work in his research group and the fascinating field of optimization. His insight into the topic and his always constructive advice coined this work considerably. The academic freedom he provided was the foundation for many great collaborations, and his passionate interest in the subject was both motivating and inspiring. He supported me in all phases of my research and contributed substantially to an excellent working environment. Working in this environment was an invaluable experience both from a professional and a personal perspective.

I also want to sincerely thank Prof. Dr.rer.nat. Nicolas Gauger, Prof. Dr. Pooyan Dadvand, and Prof. Dr.-Ing. habil. Fabian Duddeck for their expert advice in many situations and the always valuable and enjoyable discussions. I very much appreciate their interest in this work.

Very much appreciated is also the support of the European Commission as well as the Bayerische Forschungsstiftung. They funded this research and offered an exceptional framework for my professional and personal development, including the opportunity to work abroad.

Special thanks go to my colleagues, who ensured that the work at the chair was not only professionally appealing but also a lot of fun. Namely, I want to thank Armin Geiser, Reza Najian, and Thomas Oberbichler for the fantastic collaboration, as well as Aditya Ghantasala and Shahrokh Shayegan for the countless lively discussion in our office. Also, I would like to thank PD Dr.-Ing. habil. Roland Wüchner for the many priceless experiences I collected due to his commitment. Moreover, a wholehearted thanks to the Kratos developer community under the lead of Prof. Dr. Pooyand Dadvand and Prof. Dr. Riccardo Rossi. I am very proud to be part of such a marvelous international project.

Beyond that, I am deeply grateful to my family, who supported me unconditionally in all these years. I also want to thank Nicolas, Markus, and Jonathan for accompanying me as friends during this exciting time. Finally, I want to express my deepest gratitude to Franziska for her untiring support, encouragement, and patience. She gave me all the necessary backup to finish this thesis.

CONTENTS

LIST OF ACRONYMS

AD	Algorithmic Differentiation
B-Rep	Boundary Representation (Model)
CAD	Computer-Aided Design
CFD	Computational Fluid Dynamics
CSM	Computational Structural Mechanics
FD	Finite Difference Method
FEM	Finite Element Method
FSI	Fluid-Structure Interaction
IBRA	Isogeometric B-Rep Analysis
IGA	Isogeometric Analysis
KS	Kreisselmeier–Steinhauser
MDO	Multidisciplinary Design Optimization
NURBS	Non-Uniform Rational B-Splines
RANS	Reynolds-Averaged Navier-Stokes
RE	Reverse Engineering
TUM	Technische Universität München

CHAPTER 1

INTRODUCTION

Numerical shape optimization has become a standard technology in many fields of research and development. Crucial for a successful shape optimization is the parameterization of the underlying geometry. Depending on the parameterization and the corresponding design freedom, numerical shape optimization can be used to fine-tune a shape or to identify entirely new shapes in search of more efficient designs. Generally, one may distinguish two different kinds of parameterization:

1. one that relies on an extra parameter model,
2. one that utilizes given simulation grids.

Generating a dedicated parameterization is time-consuming, often a non-unique process, and, depending on the application, not always possible. Alternatively, one may directly use the simulation grid to describe and control the corresponding geometry, e.g., by defining the grid nodes as the design variables. The advantage of such grid-based approaches is that they do not require an extra modeling step. Also, they provide the maximum possible design freedom for a given mesh. The disadvantage is that they suffer from an ill-posed formulation, which typically causes a mesh dependency and non-smooth shape gradients. Therefore, they require a regularization, cf. Mohammadi et al. [1] and Jameson [2].

There are different grid-based approaches, which differ in how they regularize the problem, see e.g. Azegami et al. [3], Scherer et al. [4], Shimoda et al. [5], Le et al. [6], and Stück et al. [7]. This work considers the *Vertex Morphing Method* or simply *Vertex Morphing*, Bletzinger [8] and Hojjat et al. [9]. An integral part of the Vertex Morphing Method is a consistent filtering operation, which ensures that the generated shapes (or shape updates) are smooth. The effect of the filter is determined by a single quantity - the filter radius. The filter radius introduces a length scale and provides a simple handle to control shape modifications. The type of filter function decides about the geometric continuity. It has been shown by Bletzinger [10] that there is a perfect transition between Vertex Morphing and a parameterization using B-Splines. The advantage of Vertex Morphing is that it provides a simple handle to control the geometry, requires no extra modeling effort, yet it generates quality surfaces. In combination with the large design-freedom inherent to any grid-based approach, it enables a quick exploration of design potential. Vertex Morphing has proven its practical relevance in many cases, both in the field of structural and fluid optimization, Hojjat et al. [9], Baumgärtner et al. [11], Ertl et al. [12], Othmer [13], and Najian Asl et al. [14].

First goal of this thesis

So far, Vertex Morphing has primarily been used in optimization problems that involve a single physical discipline. This work considers a multidisciplinary design optimization (MDO) problem that frequently appears in the design of structures with internal flow. When designing such parts, a typical task is to find a shape, which optimizes specific characteristics of the flow, such as the pressure loss, without violating stress constraints in the structure. The task is challenging if the interaction between fluid and structure influences the stresses. The first goal of this thesis is to develop a method, which solves this task by numerical shape optimization. The method shall enable a quick design optimization at maximal design freedom without a costly setup of the optimization problem. Therefore, we utilize a combination of Vertex Morphing and gradient-based shape optimization. The method shall consider the inherent fluid-structure interaction (FSI) in order to obtain valid design suggestions. In doing so, we focus on one-way coupled systems where the fluid exerts forces on the structure, but the latter is stiff enough so that its displacement does not alter the flow. Considering, for example, valve blocks, hydraulic systems, or internal flows in turbomachinery, one-way coupled problems where an internal flow shall be optimized without violating stress constraints in the surrounding structure represent a large class of applications. The physics of the problem shall be analyzed using high-fidelity models in order to obtain reliable designs.

From a method point of view, Vertex Morphing needs to be prepared for MDO. To date, grid-based parameterization strategies are just little used for the optimization of coupled problems. A primary reason is that only with the advent of the coupled adjoint sensitivity analysis in the early 2000s, such a parameterization becomes a viable option in the first place. Another reason is that coupled problems may consist of several different numerical grids. Therefore, in the context of MDO, it is often claimed that a grid-based approach results in an inconsistent parameterization, Samareh [15]. The present thesis refutes this claim by presenting two extensions of the Vertex Morphing Method that enable consistent control of arbitrary meshes across several domains.

The intended optimization requires the computation of shape gradients by a sensitivity analysis of the underlying multidisciplinary problem. Martins et al. [16] provides a comprehensive overview over possible types of sensitivity analyses for multidisciplinary systems. Peter et al. [17] and Keulen et al. [18] also review different techniques, yet they focus on aerodynamic or structural optimization. Decisive for the type of sensitivity analysis is the ratio between the number of response functions versus the number of constraints. In the present case, the grid-based parameterization implies many design variables compared to which the number of response functions is assumed to be small. So, an adjoint sensitivity analysis is superior, cf. Peter et al. [17]. Also, since the problem includes fluid-structure interaction, *coupled* adjoint sensitivity analysis is required. In this context, we focus on partitioned solution strategies to exploit the advantages of already existing solvers. We also restrict ourselves to a discrete adjoint approach in favor of a consistent gradient computation. A continuous coupled adjoint sensitivity analysis is, e.g., described in Fazzolari et al. [19].

For a comparison of the discrete and the continuous adjoint approach, the reader is referred to Nadarajah et al. [20].

Coupled adjoint sensitivity analysis for FSI problems was initiated by Maute et al. [21] and Martins et al. [22] and is continuously developed ever since. For example, Barcelos et al. [23] suggested a solution strategy that allows for an increased robustness and convergence rate compared to conventional Gauss-Seidel schemes. Kenway et al. [24] presented a fully scalable approach based on high-fidelity models, which allows for optimizing entire aircraft configurations. Only recently, Sanchez et al. [25] showed a coupled adjoint sensitivity analysis, which completely drops an analytic formulation of the derivatives and only relies on algorithmic differentiation. In this work, a coupled adjoint sensitivity analysis is developed, which is tuned to the present one-way coupling and the selected parameterization strategy (Vertex Morphing). The resulting sensitivity analysis does not require an explicit computation of derivatives across domain boundaries (cross-derivatives). So, it can be applied in a fully partitioned manner using existing single-disciplinary adjoint solvers. Also, it avoids an extra mapping of sensitivity information between the domains. Both characteristics render the method particularly attractive for practical applications. The proposed sensitivity analysis represents a variation of the solution strategy presented in Najian Asl [26] and Najian Asl et al. [27]. In these papers, a coupled adjoint sensitivity analysis is introduced, which avoids the computation of cross-derivatives in general FSI problems. New in the present work is the rigorous derivation of such a sensitivity analysis for one-way coupled FSI problems.

From an application point of view, many studies discuss the shape optimization of internal flows. However, they often include an optimization of only the fluid domain. Xu et al. [28] and Verstraete et al. [29], for example, present such a single-disciplinary optimization, whereas both rely on a CAD parameterization and discrete adjoint sensitivity analysis. Papadimitriou et al. [30] presents a case, which too relies on a CAD parametric but uses continuous adjoint sensitivity analysis. Also grid-based shape optimization was already applied to optimize internal flows, see Hojjat et al. [9] and Alessi et al. [31].

Compared to such single-disciplinary cases, only a few studies consider problems in which the fluid interacts with the structure or where structural constraints must be satisfied. Heners et al. [32], for example, presents an adjoint shape optimization that allows considering a time-depended fluid-structure interaction. However, they focus on the coupling and do not consider any constraints or response functions from the structural domain. Another example is described in Stavropoulou [33]. Herein, the author presents the optimization of an internal flow, which interacts with a surrounding structure. In doing so, however, strong assumptions are introduced regarding the underlying coupling and sensitivity analysis. Also, the author does not consider any constraints. In the field of turbomachinery, where the shape optimization of structures with internal flow is a frequent problem, structural constraints have only recently been introduced, cf. Verstraete et al. [34] and Mueller et al. [35]. However, the mentioned papers do not include multidisciplinary coupling or coupled sensitivity analysis.

This work presents a new method for the optimization of structures with internal flow. The method enables the minimization of a specific flow characteristic, such as pressure loss, while considering stress constraints from the surrounding structure. In doing so, it considers a one-way coupling between both domains, and it relies on the Vertex Morphing Method in the parameterization of the shape. Thus the method provides high optimization potential without requiring a costly setup of the optimization problem.

Second goal of this thesis

The result of Vertex Morphing is a discrete free-form surface, e.g., given as triangular mesh. By contrast, modern design processes often rely on computer-aided design (CAD). In CAD, however, free-form surfaces are typically described by non-uniform rational B-splines (NURBS). Therefore, it is often necessary to convert the obtained discrete free-form surface into a NURBS-based CAD model. Manual conversion is time-consuming, however. So, it is of interest to automate this step. The second goal of this thesis is to develop a method that automates the conversion.

Converting a discrete surface into a NURBS model corresponds to a classical reverse engineering (RE) task in CAD. For an introduction to the topic, the reader is referred to Várady et al. [36]. More recent surveys are Buonamici et al. [37], Chang et al. [38], and Berger et al. [39]. Following the classification in Buonamici et al. [37], one can distinguish between feature-based reconstruction and free-form strategies. Feature-based reconstruction aims at generating parametric models. They often work with geometric primitives or features that are detected, fitted, and joined, see, e.g., Benkő et al. [40] and Bénière et al. [41]. By contrast, free-form strategies use free-form geometries, such as B-Splines or NURBS, to reconstruct a CAD model. Popular strategies introduce a patchwork of free-form surfaces, fit them to the input data, and enforce continuity constraints across the individual boundaries. In doing so, continuity constraints are enforced either directly using a constrained fitting technique, cf. Tsai et al. [42], or in a post-processing step using stitching, as in Milroy et al. [43]. Other strategies that create a CAD model with fee-form surfaces are, for example, described in Eck et al. [44], Rouhani et al. [45], and Becker et al. [46].

This work deals with grid-based shape optimization, which yields free-form geometries. Consequently, only free-form strategies are considered. Furthermore, the focus is set on strategies that deal with B-Splines or NURBS. Unlike many RE strategies, we assume an already*existing* CAD model, i.e., the CAD model of the initial (non-optimized) design. Given an initial CAD model, there is no need to create a new CAD topology. Instead, the existing CAD model can be deformed to match the optimization results. By that, it is possible to preserve knowledge and design intents inherent to the original model as much as possible.

Deforming an existing NURBS-based CAD model corresponds to a standard NURBS fitting problem. See, for example, Piegl et al. [47] for an introduction. However, state-of-the-art fitting techniques seem to be limited when applied to practical CAD models. One reason is that practical CAD models include, among other things, heavy trimming, a very irregular arrangement of many surfaces, or a non-matching parameterization between the individual surfaces. Maintaining the original surface continuity or defining a well-conditioned numerical problem hence becomes a challenge. Analyzing the state-of-the-art, it seems that current NURBS fitting techniques are not readily applicable for the deformation of practical CAD models because:

- They only consider a single patch or a single free-form surface, Brujic et al. [48], Fisher [49], Greco et al. [50], and Pottmann et al. [51].
- They include multiple patches, but assume small deformations so that the coupling between individual surfaces is negligible, Sarraga [52] and Weiss et al. [53].

- They consider multiple patches and continuity constraints, but impose strict requirements on the parameterization of adjacent surfaces, Kruth et al. [54].

- They do not include trimming, Ma et al. [55] and Dan et al. [56]. In general, many free-form strategies do not include trimming, cf. Tsai et al. [42], Eck et al. [44], Greiner et al. [57], and Krishnamurthy et al. [58]. The reason is that such strategies keep the individual surfaces as simple as possible when generating a new CAD topology.

- They focus on particular types of surfaces like sweeps, Bartoň et al. [59].

- They use subsequent surface fitting and stitching to enforce or maintain geometric continuity, Milroy et al. [43]. The author of this thesis considers this as sub-optimal, as it tends to produce unwanted local modifications along surface boundaries, especially with heavily trimmed surfaces. In contrast to that, incorporating continuity constraints directly into the fitting problem tends to produce better results.

A method, which was explicitly developed to deform a practical CAD model based on simulation results, is presented in Gaun et al. [60]. The method handles CAD models that include multiple patches and trimming. Also, it considers continuity requirements between individual surfaces. It relies on a separate fitting of edges and surfaces and enforces G^1-continuity by an extra continuity correction (stitching). This approach implies that all edges in the CAD model can be assigned with (enough) nodes in the simulation mesh - at least within an acceptable tolerance. This requirement, however, limits the method's applicability, as there are often cases where edges do not have corresponding nodes[1]. Also, the separate fitting and stitching is sub-optimal in the case of heavily trimmed models.

Louhichi et al. [61] presents another method to update CAD models based on a simulated displacement. The method relies on a subsequent reconstruction (not morphing) of edges and surfaces. A central characteristic of the method is that it requires the identification of model edges in the

[1] For example, when an edge does not represent a geometric feature but an internal model boundary between two adjacent surfaces.

deformed simulation mesh. The identification implies an extra feature recognition or a tight CAD/CAE integration, where the relation between CAD and mesh is known. This requirement limits the applicability of the method. Also, the authors do not specifically address the surface continuity. Indeed, the separate reconstruction of edges automatically ensures a watertight model, but they do not consider G^1-continuity.

Given the limitations above, this work proposes a new method for the intended conversion. In doing so, a particular focus is set on problems arising with practical CAD models. The method has the following properties: It maps a displacement field (shape change) from a simulation grid onto a given CAD model. As a result, one obtains a deformed CAD model that reflects the simulation results. The method can handle general boundary representation (B-Rep) models, including multiple patches, trimming, and continuity constraints. It also supports design constraints. Hence one may enforce geometric requirements, which deviate from the input but are necessary to obtain a feasible design. Since the method works with general B-Rep models, and its purpose is to morph an existing CAD geometry such that it matches a simulated deformation, we refer to the method as *B-Rep morphing*.

Contribution of this thesis

- Preparation of Vertex Morphing for an application in MDO, which includes an extension of the method to enable the handling of volume meshes as well as several meshes simultaneously.
- Development of a method for shape optimization of structures with internal flow considering one-way fluid-structure interaction.
- Derivation of a specialized coupled adjoint sensitivity analysis, which can be realized with established single-disciplinary adjoint solvers.
- Development of a method (B-Rep-Morphing) to convert the results of a grid-based shape optimization into a CAD model.

Outline

Chapter 2 starts with the introduction of the Vertex Morphing Method. It also presents two extensions, which allow an application of Vertex Morphing in MDO problems.

Chapter 3 covers the development of the intended multidisciplinary optimization approach. First, the problem is formulated, and the basic solution approach is discussed. The next section introduces the parameterization. Then the required coupled sensitivity analysis is developed. Afterwards, test cases are introduced based on which the sensitivity analysis is verified. Finally, the whole multidisciplinary process is applied to optimize a representative structure.

Chapter 4 proceeds with the development of B-Rep morphing. First, the basic idea is presented, and general assumptions are introduced. Then essential fundamentals regarding NURBS and B-Rep models are reviewed. The next section discusses special integration methods, which are required for B-Rep morphing. Afterwards, the underlying mapping operation is formulated. In the following sections, the mapping operation is more and more extended to account for practical CAD models. Those extensions include regularization, constraints, and surface smoothing. Another section discusses the necessary geometry refinement. At the end of the chapter, the method is showcased with several practical examples, including the results from chapter 3.

Chapter 5 finally summarizes the outcome of this work.

Chapter 2

The Vertex Morphing Method

This chapter presents the Vertex Morphing Method. In doing so, the original method is discussed, and two extensions are proposed: one for the simultaneous control of geometry and mesh and one for the simultaneous control of several meshes. The two extensions significantly broaden the possible field of application of Vertex Morphing. In particular, they allow the method to be applied in the context of MDO.

2.1 Basic Vertex Morphing

Vertex Morphing in its basic form is a method to control the geometry of a design surface within a shape optimization process, cf. Bletzinger [8] and Hojjat et al. [9]. The central idea is the introduction of a control field based on which the geometry of the design surface is derived. The design surface is in the following referred to as $\mathit{\Gamma}_D$, the control field and the resulting geometry as $\tilde{\mathbf{s}}(\boldsymbol{\xi}) = [\tilde{s}_x, \tilde{s}_y, \tilde{s}_z]^T \in \mathbb{R}^3$ and $\tilde{\mathbf{x}}_D(\boldsymbol{\xi}) = [\tilde{x}_{D,x}, \tilde{x}_{D,y}, \tilde{x}_{D,z}]^T \in \mathbb{R}^3$, respectively. Both $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{x}}_D$ depend on the surface coordinates of a fixed reference configuration, $\boldsymbol{\xi} = [\xi_x, \xi_y, \xi_z]^T \in \mathit{\Gamma}_D^{(0)}$. The reference configuration is defined to be the initial design surface, $\Gamma_D^{(0)}$. The space in which the geometry is defined is referred to as the *geometry space*. The space in which the control field is defined is referred to as the *control space*. Optimization problems are formulated in the control space.

In the Vertex Morphing Method, the following relation is established between the geometry of the design surface and the control field:

$$\tilde{\mathbf{x}}_D(\boldsymbol{\xi}_i) = \int_{\Gamma_D^{(0)}} A(\boldsymbol{\xi} - \boldsymbol{\xi}_i)\tilde{\mathbf{s}}(\boldsymbol{\xi})d\Gamma_D^{(0)} \tag{2.1}$$

Herein, $\boldsymbol{\xi}_i$ denotes a specific point on $\Gamma_D^{(0)}$. As one can see, the geometry results from a convolution of the control field with a kernel function A. Through the convolution, the geometry at $\boldsymbol{\xi}_i$ is linked to the geometry at points in the vicinity of $\boldsymbol{\xi}_i$. The kernel function characterizes the type of link. In this work, we assume the kernel function to be a linear hat function defined as follows:

$$A = \begin{cases} c\left(1 - \dfrac{\|\boldsymbol{\xi} - \boldsymbol{\xi}_i\|_2}{r}\right), & \text{if } \|(\boldsymbol{\xi} - \boldsymbol{\xi}_i)\|_2 < r \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

In the previous equation, c is a constant scaling factor. For normalization purposes, the scaling factor is chosen such that the kernel function satisfies the property:

$$\int_{\Gamma_D^{(0)}} A(\boldsymbol{\xi} - \boldsymbol{\xi}_i)d\Gamma_D^{(0)} = 1 \tag{2.3}$$

r represents an arbitrary radius that limits the area of influence of the kernel function. In (2.1), this radius defines which part of the control field influences a given point of the geometry. Vice versa, it defines which parts of the geometry are influenced by a given point of the control field, i.e., a given control point. The strictly local influence of the kernel function is a crucial characteristic of the Vertex Morphing Method. Given the definition of the kernel function in (2.2) and (2.3), (2.1) corresponds to a smoothing or filtering operation. A and r are therefore called the *filter function* and the *filter radius*, respectively.

The filtering operation enables the generation of globally smooth shapes, whereas the filter radius introduces a minimal or characteristic length-scale below which geometrical fluctuations are suppressed. Due to this effect, the filter radius acts as a design parameter (the only one in Vertex Morphing). The larger the filter radius, the smoother is the resulting geometry, i.e., the resulting geometry is more and more characterized by low-frequent modes. By contrast, the smaller the filter radius, the more geometrical details are possible.

For numerical analysis, the geometry and the control field must be discretized. The discretization of both quantities requires a meshing of the design surface in reference configuration, $\Gamma_D^{(0)}$. Generally, the meshing of the design surface may be different in the case of $\tilde{\mathbf{x}}_D$ and $\tilde{\mathbf{s}}$.

In grid-based shape optimization, the mesh for the discretization of the geometry is given. It corresponds to the mesh of the underlying numerical simulation[1], or more precisely, the part of it, which covers $\Gamma_D^{(0)}$. The number of nodes in this mesh is in the following represented by m, the corresponding set of nodes by $\hat{\boldsymbol{\xi}}_G = [\boldsymbol{\xi}_{G,1}^T, \boldsymbol{\xi}_{G,2}^T, \ldots, \boldsymbol{\xi}_{G,m}^T]^T$. $\boldsymbol{\xi}_G = [\xi_{G,x}, \xi_{G,y}, \xi_{G,z}]^T \in \Gamma_D^{(0)}$ denotes the coordinates of a single node in the mesh. Evaluating $\tilde{\mathbf{x}}_D$ at all m nodes, one obtains a set of points representing the discrete geometry of the design surface. In between those points, we approximate the geometry using the interpolation scheme from the underlying numerical simulation.

For the discretization of the control field, we require another mesh. The number of nodes in this mesh is represented by n, the corresponding set of nodes by $\hat{\boldsymbol{\xi}}_C = [\boldsymbol{\xi}_{C,1}^T, \boldsymbol{\xi}_{C,2}^T, \ldots, \boldsymbol{\xi}_{C,n}^T]^T$. $\boldsymbol{\xi}_C = [\xi_{C,x}, \xi_{C,y}, \xi_{C,z}]^T \in \Gamma_D^{(0)}$ denotes the coordinates of a single node in the mesh. Evaluating the control field at all n nodes yields a set of discrete *control points*. In between the control points, the control field is approximated using standard finite element basis functions. Denoting control points as $\tilde{\mathbf{s}}_j$ and basis functions as N_j, the discretization of the control field reads:

$$\tilde{\mathbf{s}} = N_j(\boldsymbol{\xi})\tilde{\mathbf{s}}_j \qquad j = 1 \ldots n \tag{2.4}$$

[1] E.g., a structural analysis based on the finite element method.

In general, one can choose an arbitrary mesh to discretize the control field. However, one needs to consider that the refinement of this mesh determines the design freedom in the controlled geometry. A smart approach is to use the same mesh as already used for the discretization of the geometry, i.e., the mesh of the underlying numerical simulation. In the context of shape optimization, this choice has two advantages:

1. No further modeling is necessary. That is, no explicit parameterization or definition of the control field is required.
2. We can exploit the maximum design freedom of the given mesh.

Alternatively, one may also use two different meshes for the discretization of the control field and the geometry. Note in this context that by choosing $n < m$, the geometry is naturally smoothed. In this case, the smoothing is a direct consequence of the reduced design freedom. However, controlling the smoothness by the discretization of the control field requires extra modeling effort. Also, it is significantly less intuitive compared to the choice of a single filter radius as in the filtering. Therefore, one typically uses the same mesh for the discretization of $\tilde{\mathbf{x}}_D$ and $\tilde{\mathbf{s}}$ and regulates the smoothness solely through the filter radius. The filter radius hence represents the only design parameter in the Vertex Morphing Method.

Introducing (2.4) in (2.1) and considering the discretization of the geometry and the control field, one obtains:

$$\tilde{\mathbf{x}}_D(\boldsymbol{\xi}_{G,i}) = \left[\int_{\Gamma_D^{(0)}} A(\boldsymbol{\xi} - \boldsymbol{\xi}_{G,i}) N_j(\boldsymbol{\xi}) d\Gamma_D^{(0)} \right] \tilde{\mathbf{s}}_j \qquad i|j = 1 \ldots m|n \tag{2.5}$$

or in abbreviated form:

$$\tilde{\mathbf{x}}_{D,i} = B_j(\boldsymbol{\xi}, \boldsymbol{\xi}_{G,i}) \tilde{\mathbf{s}}_j \qquad i|j = 1 \ldots m|n \tag{2.6}$$

In (2.5), the discrete control points are taken out of the integral. The remaining integral represents a general morphing function denoted as B_j. If A and N_j are polynomial functions of degree p and q, respectively, then B_j is a piecewise polynomial function of degree $(p+q+1)$. More precisely, Bletzinger [10] showed that (2.5) corresponds to the implicit definition of a B-Spline of degree $(p+q+1)$. As a result, the geometry follows a B-Spline curve or B-Spline surface, respectively. The role of the above-introduced control points is thus equivalent to the role of control points in B-Splines. The combination of control field and local filtering operation allows the Vertex Morphing Method to generate smooth shapes with a surface quality that is comparable to CAD geometries. For a more detailed discussion of the filtering, the corresponding smoothing properties, and the implicit definition of B-Splines in the context of Vertex Morphing, the interested reader is referred to Bletzinger [8, 10].

Having discretized $\tilde{\mathbf{x}}_D$ and $\tilde{\mathbf{s}}$, one can compute the convolution integral in (2.5) numerically. In practice, an approximation of the integral is sufficient. Therefore, the midpoint Riemann sum is applied. To formulate the Riemann sum, we evaluate the integral at all nodes of the mesh of the control field that lay within one filter radius around the geometry point $\boldsymbol{\xi}_{G,i}$. We refer to the coordinates of those nodes as $\boldsymbol{\xi}_{C,j}$. At every $\boldsymbol{\xi}_{C,j}$ it holds $N_j = 1$. Each $\boldsymbol{\xi}_{C,j}$ is assigned with a local fraction of the design surface, $d\Gamma^{(0)}_{D,j}$. We then approximate the integral by a weighted sum of the contributions from all neighboring nodes. Afterwards, we scale the weighted sum by the sum of all weights. This "post-scaling" ensures the normalization condition specified in (2.3) without a prior determination of the scaling factor c. The integral is finally computed as follows:

$$\tilde{\mathbf{x}}_{D,i} = \frac{\sum_{j=1}^{n} A(\boldsymbol{\xi}_{C,j} - \boldsymbol{\xi}_{G,i}) d\Gamma^{(0)}_{D,j} \tilde{\mathbf{s}}_j}{\sum_{j=1}^{n} A(\boldsymbol{\xi}_{C,j} - \boldsymbol{\xi}_{G,i}) d\Gamma^{(0)}_{D,j}} = A_{ij} \tilde{\mathbf{s}}_j \qquad i = 1 \dots m \tag{2.7}$$

Herein, A_{ij} denotes a mapping or filter matrix that translates the position of the control points into an actual geometry. In general, A_{ij} can be used to map any quantity from control space to geometry space. Rewriting (2.7) in matrix notation, we obtain the following equation:

$$\underset{3m\times 1}{\hat{\mathbf{x}}_D} = \underset{3m\times 3n}{\mathbf{A}} \; \underset{3n\times 1}{\hat{\mathbf{s}}} \tag{2.8}$$

In that, $\hat{\mathbf{x}}_D = [\tilde{\mathbf{x}}_{D,1}^T, \ldots, \tilde{\mathbf{x}}_{D,m}^T]^T$ represents the discrete geometry of the design surface and $\hat{\mathbf{s}} = [\tilde{\mathbf{s}}_1^T, \ldots, \tilde{\mathbf{s}}_n^T]^T$ the set of control points.

From (2.7), it is clear that the same A_{ij} applies to all coordinate directions of a single control point. That is, the geometry can be computed independently in all coordinate directions using only a reduced mapping matrix of size $m \times n$. This characteristic can be exploited in the implementation of the Vertex Morphing Method to lower the associated computational effort and memory requirements.

2.2 Control of geometry updates

Defining the geometry as in (2.7) implies a smoothing of the original design surface, $\Gamma_D^{(0)}$. I.e., a design surface which is non-smooth by construction, since it contains edges or corners, for example, will lose important geometrical features. In practice, such a change of the original design is often unwanted. Instead, it is often required to start from a fixed initial design and only introduce geometry *updates*. Hence, geometrical features may be preserved. Because of this requirement, Vertex Morphing is often used to control geometry updates instead of the geometry itself.

Adjusting the Vertex Morphing Method to control geometry updates is straightforward. Starting from the discrete definition of the geometry in (2.8) and given a reference design identified by the superscript (0), it holds:

$$\hat{\mathbf{x}}_D = \mathbf{A}\hat{\mathbf{s}} = \mathbf{A}\hat{\mathbf{s}}^{(0)} + \mathbf{A}\Delta\hat{\mathbf{s}} \tag{2.9a}$$

$$\mathbf{A}\hat{\mathbf{s}}^{(0)} = \hat{\mathbf{x}}_D^{(0)} \tag{2.9b}$$

$$\mathbf{A}\Delta\hat{\mathbf{s}} = \Delta\hat{\mathbf{x}}_D \tag{2.9c}$$

The properties of Vertex Morphing are preserved. That is, updates generated according to (2.9c) show a characteristic length-scale in the order of the selected filter radius. Moreover, $\Delta\hat{\mathbf{x}}_D$ represents discrete points of an otherwise smooth field of geometry updates, $\Delta\tilde{\mathbf{x}}_D$. In the case of polynomial basis and filter functions, $\Delta\tilde{\mathbf{x}}_D$ takes the form of a B-Spline curve or surface. So, geometry updates generated according to (2.9c) may be considered as "smooth". An important difference to the original formulation is that the actual position of the control points, $\hat{\mathbf{s}}$, is irrelevant and must

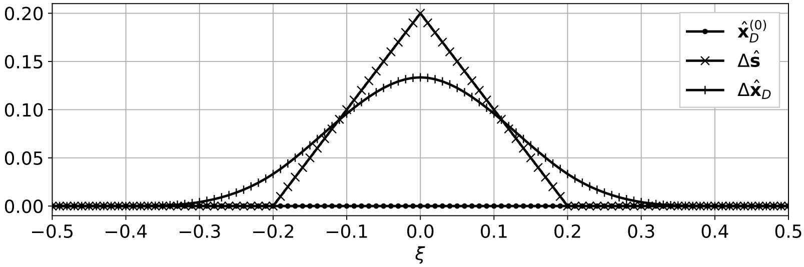


Figure 2.1: Geometry update computed by Vertex Morphing in case of a discrete line. We assume a linear displacement of the control points and a linear filter function with $r = 0.2$.

never be calculated. Only the displacement of the control points, $\Delta\hat{\mathbf{s}}$, is of importance.

The computation of a geometry update using Vertex Morphing is demonstrated in figure 2.1. The figure shows the case of a discrete line. Starting from the original geometry of the line, $\hat{\mathbf{x}}_D^{(0)}$, an update $\Delta\hat{\mathbf{x}}_D$ shall be computed using the Vertex Morphing Method. To avoid additional modeling effort, we use the given mesh for both the discretization of the geometry and the discretization of the control field, i.e. $\hat{\mathbf{x}}_D^{(0)} = \hat{\boldsymbol{\xi}}_G = \hat{\boldsymbol{\xi}}_C$. A linear displacement of the control points is assumed. For the filtering, we chose a linear hat function as described in (2.2), whereas the filter radius shall be $r = 0.2$. The geometry update is computed according to (2.9c). Figure 2.1 shows the corresponding result.

As one can see in the figure, the filtering within the Vertex Morphing Method translates the non-smooth displacement of the control points into a smooth update of the geometry. Since we assumed a linear displacement of the control points and a linear filter function, $\Delta\hat{\mathbf{x}}_D$ follows the curve of a piecewise cubic B-Spline.

In figure 2.1, one can also see that the geometry is only updated in parts of the domain. This result is a consequence of the local definition of the filtering operation, which limits the area of influence of a control point on the geometry. Control points influence the geometry only within one filter radius around the corresponding point on the design surface, $\boldsymbol{\xi}_{C,j}$.

So, in this example, the displacement of the control points in the region $-0.2 < \xi < 0.2$ updates the geometry in the region $-0.4 < \xi < 0.4$. Outside this region, the geometry remains unchanged. The local influence of the control points in Vertex Morphing is comparable to the local influence of the control points in B-Splines.

In the later shape optimization, $\varDelta\hat{\mathbf{s}}$ corresponds to the set of design variables. Accordingly, we need to compute design sensitivities of the form $dI/d\varDelta\hat{\mathbf{s}}$, whereas I denotes an arbitrary response function. Considering (2.9), the required design sensitivities can be computed as follows:

$$\frac{dI}{d\varDelta\hat{\mathbf{s}}} = \frac{dI}{d\hat{\mathbf{x}}_D}\frac{d\hat{\mathbf{x}}_D}{d\varDelta\hat{\mathbf{s}}} = \frac{dI}{d\hat{\mathbf{x}}_D}\mathbf{A} \tag{2.10}$$

Note from (2.10) and (2.9c) that the filtering operation is applied *twice* in the context of shape optimization: once to map sensitivities from the geometry space to the control space according to (2.10), and once to map design updates in the opposite direction according to (2.9c).

2.3 Simultaneous control of geometry and mesh

Originally, Vertex Morphing was introduced to control surfaces. That is, if the surface represents the boundary of a volume, then an extra simulation is necessary to adapt the volume mesh to modifications on the boundary. Consider for example the discretization presented in figure 2.2(a). If one of the boundaries is modified by Vertex Morphing, nodes in the interior have to follow to preserve a valid mesh.

From a theoretical point of view, Vertex Morphing is not limited to a specific topology. That is, with the same logic as introduced in section 2.1, one can also use Vertex Morphing to control entire volume meshes. In the shape optimization of volumetric parts, controlling the volume mesh would mean that no extra simulation of the mesh motion is necessary. While this approach seems a natural extension of the Vertex Morphing Method, there is no reported application, and it has not been investigated in the literature to date. This work shall catch up on this. So, having in mind a simultaneous control of geometry *and* mesh, the following extension of the original Vertex Morphing Method is proposed:

$$\tilde{\mathbf{x}}_D(\xi_i) = \int_{\Omega_D^{(0)}} A(\xi - \xi_i)\tilde{\mathbf{s}}(\xi)\,d\Omega_D^{(0)} \tag{2.11}$$

Comparing (2.11) to (2.1), we find that the only difference to the original formulation is the extended integration domain, $\Omega_D^{(0)}$. The latter specifies the domain in which Vertex Morphing shall control a given mesh. This domain includes the actual design surface, i.e., $\Gamma_0^{(0)} \subset \Omega_D^{(0)}$, *and* all regions of the mesh which shall adapt to a modification of the design surface. Analogously to the original Vertex Morphing, (0) indicates the reference configuration given by the initial design.

The modified integration domain implies that $\xi \in \Omega_D^{(0)}$. So, in the extended Vertex Morphing, ξ represents spatial coordinates. By contrast, in the original formulation, ξ represented surface coordinates. Also, $\tilde{\mathbf{x}}_D$ and $\tilde{\mathbf{s}}$ are defined over a volumetric domain. Accordingly, $\tilde{\mathbf{x}}_D$ describes not just the geometry (boundary) of an underlying body, as in the original formulation, but its position at all $\xi \in \Omega_D^{(0)}$.

Note that $\tilde{\mathbf{x}}_D$ in (2.11) is still generated by a convolution of a control field with a kernel function. So, the extended Vertex Morphing inherits the smoothing properties of the original method.

For numerical analysis, $\tilde{\mathbf{x}}_D$ and $\tilde{\mathbf{s}}$ must be discretized. The discretization is performed analogously to the approach explained in section 2.1. Based on the discretized quantities, the integral in (2.11) is approximated using a mid-point Riemann sum. The current design is then computed as follows:

$$\tilde{\mathbf{x}}_{D,i} = \frac{\sum_{j=1}^{n} A(\xi_{C,j} - \xi_{G,i})\,d\Omega_{D,j}^{(0)}\tilde{\mathbf{s}}_j}{\sum_{j=1}^{n} A(\xi_{C,j} - \xi_{G,i})\,d\Omega_{D,j}^{(0)}} = A_{ij}\tilde{\mathbf{s}}_j \qquad i = 1 \ldots m \tag{2.12}$$

Similarly to the original method, $\xi_{G,i}$ represents the coordinates of a node in the mesh used to discretize the geometry. $\xi_{C,j}$ represents the coordinates of a node in the mesh used to discretize the control field. Both meshes are volume meshes in this case.

Note from a comparison of (2.7) and (2.12) that the implementation of the extended formulation mostly corresponds to that of the original formulation. The only difference is that in the extended formulation, each node in the mesh of the control field must be assigned with a volume fraction, $d\Omega^{(0)}_{D,j}$, instead of a surface fraction, $d\Gamma^{(0)}_{D,j}$.

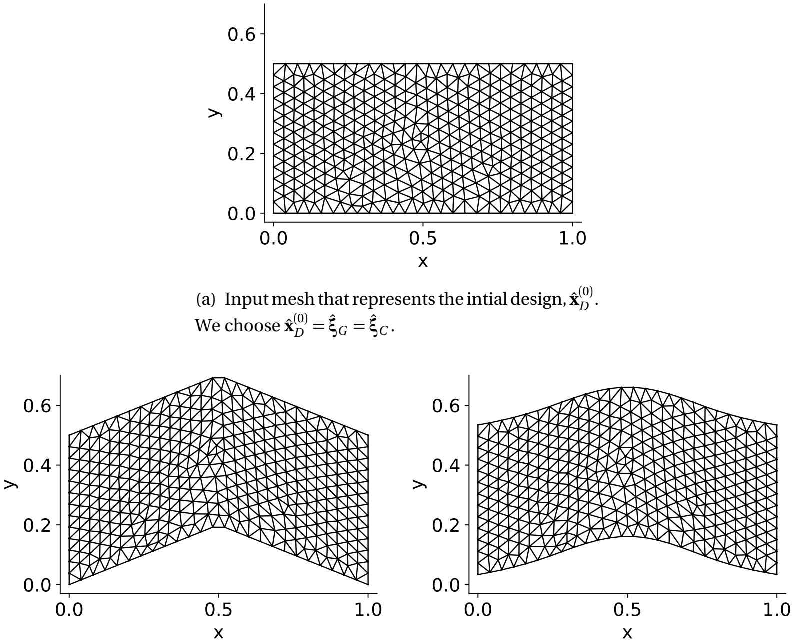
Following the matrix notation in (2.8), we define $\hat{\mathbf{x}}_D = [\tilde{\mathbf{x}}^T_{D,1},\dots,\tilde{\mathbf{x}}^T_{D,m}]^T$ and $\hat{\mathbf{s}} = [\tilde{\mathbf{s}}^T_1,\dots,\tilde{\mathbf{s}}^T_n]^T$. The latter represents the set of control points, the former the coordinates of the nodes in the mesh to be controlled.

In the following, the extended method is demonstrated with an example. Consider again the mesh in figure 2.2(a). Vertex Morphing shall be used to control all nodes of the mesh simultaneously. That is, $\Omega^{(0)}$ covers the entire structure. The original mesh shall not be modified. Instead, only mesh updates shall be introduced. Accordingly, we replace $\hat{\mathbf{x}}_D$ and $\hat{\mathbf{s}}$ by $\Delta\hat{\mathbf{x}}_D$ and $\Delta\hat{\mathbf{s}}$, respectively. The given mesh represents the initial design, $\hat{\mathbf{x}}^{(0)}_D$. It also provides a discretization of the design domain in reference configuration, $\Omega^{(0)}$. In order to avoid extra modeling effort, we chose $\hat{\mathbf{x}}^{(0)}_D = \hat{\boldsymbol{\xi}}_G = \hat{\boldsymbol{\xi}}_C$. So, there are as many control points as nodes in the mesh. For the filtering, we use the linear hat function from (2.2). In favor of low-frequent and globally smooth updates, a large filter radius of $r = 0.3$ is specified, which makes up 30% of the width of the structure.

Given this setup, a linear displacement, $\Delta\hat{\mathbf{s}}$, is assumed for all control points (see figure 2.2(b)). The resulting mesh update computed by the extended Vertex Morphing Method is visualized in figure 2.2(c).

Figure 2.2(c) shows that the displacement of the control points leads to an update of the entire mesh. As one can see, an extra adaption of the interior nodes is not necessary. The interior nodes already move consistently with the boundary nodes that define the shape. Note in this context that the linear displacement of the control points is translated into a smooth mesh update. In fact, since $\Delta\hat{\mathbf{s}}$ is linear and a linear filter function was defined, the update along any curve within the mesh corresponds to a cubic B-Spline (see, e.g., the upper and lower boundary in figure 2.2(c)). The results also show that the mesh update is low-frequent and globally smooth. This result is a consequence of the relatively large filter radius.

Looking closer at the individual elements in figure 2.2(c), it is striking that the mesh update did not destroy the element quality. Instead, we observe a smooth distribution of the motion throughout the mesh. The smooth mesh



(a) Input mesh that represents the intial design, $\hat{\mathbf{x}}_D^{(0)}$. We choose $\hat{\mathbf{x}}_D^{(0)} = \hat{\boldsymbol{\xi}}_G = \hat{\boldsymbol{\xi}}_C$.

(b) Linear $\Delta\hat{\mathbf{s}}$ visualized on $\hat{\boldsymbol{\xi}}_C$.

(c) $\Delta\hat{\mathbf{x}}_D$ visualized on $\hat{\boldsymbol{\xi}}_G$

Figure 2.2: Demonstration of simultaneous control of geometry and mesh using Vertex Morphing.

motion is again a consequence of the filtering, which in this case, ensures that mesh updates of neighboring nodes are similar. The similar motion of neighboring nodes locally reduces the deformation of the elements and globally avoids an unnecessary mesh distortion. It also causes elements that are small compared to the filter radius to move as an almost rigid body. In this regard, the resulting quality seems to be comparable to other dedicated mesh motion strategies.

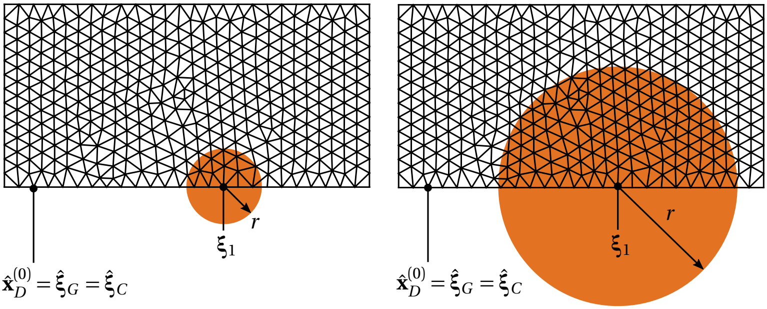
The previous results show that we can use Vertex Morphing in the extended form to unify the control of geometry and mesh. In the context of shape optimization, this unification has the following advantages:

- No extra simulation of the mesh motion is required when changing the geometry.
- As there is no extra simulation of the mesh motion, there are also no contributions from such a simulation that need to be considered in the design sensitivity analysis from equation (2.10).
- The smoothing properties of Vertex Morphing can be utilized to maintain a valid mesh throughout the optimization.

The disadvantages of such a unification are:

- The filter radius must span over a large fraction of the mesh. More precisely, the fraction must be large enough to distribute a given geometry update into the volume properly. This requirement sets a rather high lower limit on the filter radius. Figure 2.3 illustrates the problem.
- The filter radius must be large enough to span over several elements within the mesh. Otherwise, proper filtering is not possible. So, the size of the elements in the volume mesh must be considered in the choice of r.
- Control points have a local influence. That means, updating an individual control point will only cause a mesh motion within one filter radius around its corresponding surface node, see figure 2.3. This behavior limits the possible mesh adaption and is different from global mesh-motion strategies such as the pseudo-elastic approach.
- Given the local influence, mesh updates are only introduced in areas, which are relevant for the given response functions (areas with high sensitivity values). Conversely, areas with vanishing sensitivities are not considered in the mesh update. This limitation tends to generate local motion patterns.

To explain the last aspect, consider the example shown in figure 2.4. The figure shows a mesh, for which we assume an artificial sensitivity vector, $dI/d\hat{\mathbf{x}}_D$. The sensitivity vector only has non-zero values at the node on the lower right corner (see figure 2.4(a)). Additionally, we assume a linear



r
ξ_1
$\hat{\mathbf{x}}_D^{(0)} = \hat{\xi}_G = \hat{\xi}_C$
r
ξ_1
$\hat{\mathbf{x}}_D^{(0)} = \hat{\xi}_G = \hat{\xi}_C$

(a) A small radius will only cause local updates, which may quickly lead to mesh distortions.

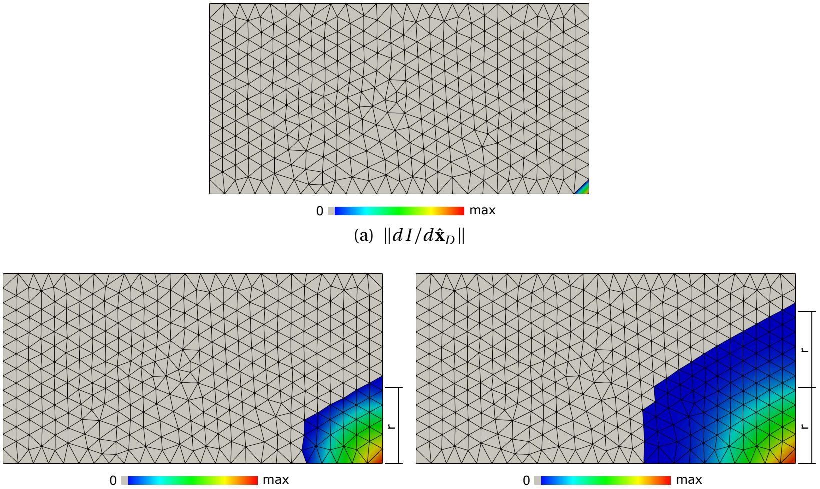
(b) With a large radius, there is enough space to relocate nodes within the volume mesh.

Figure 2.3: Dependency of the mesh motion on r in the extended Vertex Morphing. ξ_1 denotes an arbitrary node in the mesh. If the associated control point is modified, it will only cause a mesh motion within the orange region.

filter function A and $r = 0.2$. So, the filter radius is small compared to the dimensions of the structure. Filtering the given sensitivities according to (2.10), we obtain the design sensitivities in the control space. Note that the local filtering transforms the concentrated sensitivity information in the geometry space into a more distributed sensitivity information in the control space. The distribution is locally limited by the filter radius r.

Now, from the design sensitivities we define an update of the control points in direction of the steepest descent, $\Delta\hat{\mathbf{s}} = -dI/d\Delta\hat{\mathbf{s}}$. Based on the latter, we compute a mesh update according to (2.9c), which implies another filtering operation. The result is shown in figure 2.4(c). Note in the figure that the mesh update is again locally limited by the filter radius r. So, in total, the update of the mesh concentrates around the node with non-zero sensitivities. In practice, this characteristic may lead to the aforementioned local motion patterns.

All the disadvantages mentioned above limit the applicability of the extended method to cases where the filter radius is already large by design. However, there are many applications where this is the case. In such cases, the disadvantages are less pronounced or even negligible, and the advantages of the simultaneous control of geometry and mesh by Vertex



(a) $\|dI/d\hat{\mathbf{x}}_D\|$

(b) $\|dI/d\Delta\hat{\mathbf{s}}\|$

(c) $\|\Delta\hat{\mathbf{x}}_D\|$

Figure 2.4: Concentrated mesh update around an area with a high sensitivity value.

Morphing can be exploited.

2.4 Simultaneous control of several meshes

In Vertex Morphing, the geometry of a body is defined according to (2.5). Note in the previous equation that the geometry is exclusively defined by the control field and the filter function. The discretization of the geometry is irrelevant and only determines its numerical resolution. That means, the same control field and the same filter function always describe the same underlying geometry[2] independent of how it is actually discretized. The fact that the description of the geometry is independent of its discretization renders the Vertex Morphing Method mesh-independent, which is one of its central properties.

A mesh-independent description of the geometry is essential for physically meaningful shape optimization. However, in the context of Vertex Morph-

[2] Or the same geometry update, if we use Vertex Morphing to control geometry updates.

ing, it has another technical implication. I.e., if the controlled geometry is exclusively defined by the control field and the filter function, then a single set of parameters can be used to control *several* discretizations of the same geometry *simultaneously.* Controlling several discretizations at a time is useful, for example, in the context of MDO, especially in cases where the design surface is part of an interface between several coupled domains. In such a case, a modification of the design surface must be consistently translated to all domains. "Consistent" means that the position of the meshes relative to each other does not change, and the connectivity between the domains is preserved. By a straightforward extension, one can use Vertex Morphing to simultaneously control several meshes based on a consistent description of the geometry across the domains. The extension is presented in the following. Najian Asl [26] already showed an application, where Vertex Morphing was used to control two surface meshes at a time. New in this thesis is the generalization and rigorous discussion of the extension.

The consistent control of several meshes using Vertex Morphing implies that one set of control points is combined with several discretizations of the same geometry. From a theoretical point of view, the original Vertex Morphing already covers such a scenario because the discretization of the control field is arbitrary, and the discretization of the geometry is irrelevant for its description. Therefore, an extension of the theory is not necessary. Since there are no further theoretical implications, all characteristics of the Vertex Morphing Method equally apply to all controlled meshes. I.e., the generated designs are smooth and, due to the mesh-independence, they all describe the same underlying geometry.

From an implementation point of view, the extension requires two adjustments compared to the original Vertex Morphing Method: First, to avoid an extra modeling effort, one of the given meshes must be selected for the discretization of the control field, $\hat{\boldsymbol{\xi}}_C$. By contrast, in the original method, we assumed that there is just one input mesh and thus no choice if an extra modeling effort shall be avoided. Evaluating the control field at all nodes of $\hat{\boldsymbol{\xi}}_C$ yields the common set of control points, $\hat{\mathbf{s}}$.

Second, the discrete geometry, $\hat{\mathbf{x}}_D$, must be computed for each of the given meshes *separately* according to (2.7). Therefore, dedicated mapping matrices are required. For a consistent description of the geometry, the mapping matrices must be determined based on the *same* combination of filter function (A) and filter radius (r).

Considering both adjustments, we establish the following relation between the selected control points and the given meshes:

$$\underset{3m_k\times 1}{\hat{\mathbf{x}}_{Dk}} = \underset{3m_k\times 3n}{\mathbf{A}_k} \underset{3n\times 1}{\hat{\mathbf{s}}} \qquad k = 1\ldots\#\text{ meshes} \tag{2.13}$$

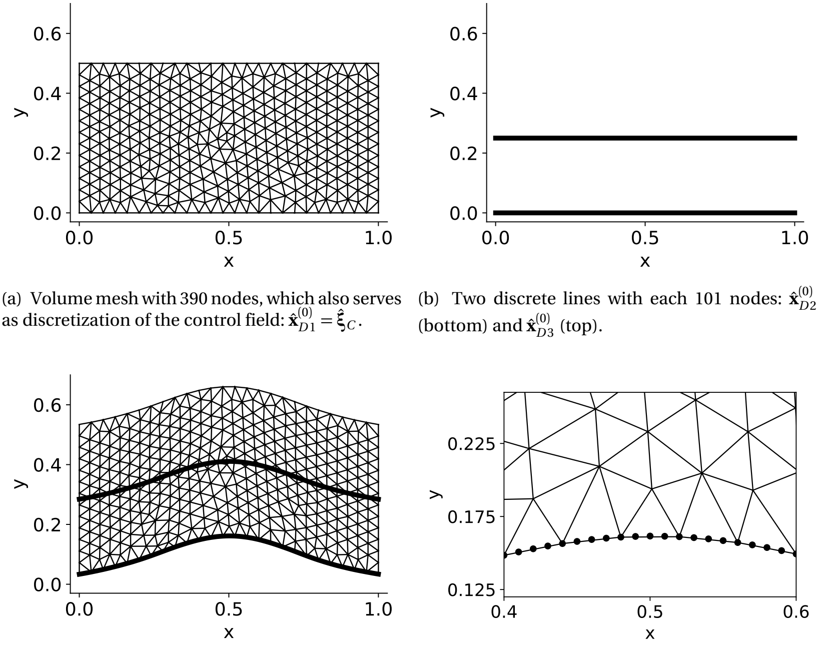
Herein, $\hat{\mathbf{s}}$ represents the common set of control points, $\hat{\mathbf{x}}_{Dk}$ the geometry described on mesh k, and $\mathbf{A}_k$ the corresponding mapping matrix. m_k denotes the number of nodes of the k-th mesh and n corresponds to the number of control points. As one can see, the size of a mapping matrix varies depending on the number of nodes in the relevant meshes. The individual terms are factorized according to (2.7). It is important to realize that the same set of control points controls the geometry in all meshes. So, an update of $\hat{\mathbf{s}}$ causes a simultaneous update of all $\hat{\mathbf{x}}_{Dk}$. If all mapping matrices build upon the same filter settings, then all $\hat{\mathbf{x}}_{Dk}$ describe the same underlying geometry. An update of the control points would then lead to a *consistent* update of every mesh.

The following example shall demonstrate the simultaneous control of several meshes by Vertex Morphing.

Consider the case in figure 2.5. In this case, there are three different meshes, $\hat{\mathbf{x}}_{D1}^{(0)}$, $\hat{\mathbf{x}}_{D2}^{(0)}$, and $\hat{\mathbf{x}}_{D3}^{(0)}$. The individual meshes describe three different parts of the same initial geometry, i.e., a rectangle. The first mesh comprises the entire rectangle (figure 2.5(a)), whereas the remaining two meshes describe two different line segments of it (figure 2.5(b)).

All meshes shall be parameterized such that they can be modified simultaneously. In doing so, a change in the control points shall cause a consistent modification of all meshes. So, the meshes shall not change their relative position to each other. The initial geometry shall not be modified. Instead, the parameterization shall only introduce geometry updates. The three given meshes represent the initial geometry.

To establish a parameterization that simultaneously controls all the given grids, we follow the logic in (2.13). To restrict the parameterization to geometry updates, we replace $\hat{\mathbf{x}}_{Dk}$ and $\hat{\mathbf{s}}$ in (2.13) by $\Delta\hat{\mathbf{x}}_{Dk}$ and $\Delta\hat{\mathbf{s}}$, respectively. The discretization of the geometry is given by $\hat{\mathbf{x}}_{D1}^{(0)} = \hat{\boldsymbol{\xi}}_{G1}$, $\hat{\mathbf{x}}_{D2}^{(0)} = \hat{\boldsymbol{\xi}}_{G2}$, and $\hat{\mathbf{x}}_{D3}^{(0)} = \hat{\boldsymbol{\xi}}_{G3}$. For the discretization of the control field, we choose $\hat{\mathbf{x}}_{D1}^{(0)} = \hat{\boldsymbol{\xi}}_{C}$ as



(a) Volume mesh with 390 nodes, which also serves as discretization of the control field: $\hat{\mathbf{x}}_{D1}^{(0)} = \hat{\boldsymbol{\xi}}_C$.

(b) Two discrete lines with each 101 nodes: $\hat{\mathbf{x}}_{D2}^{(0)}$ (bottom) and $\hat{\mathbf{x}}_{D3}^{(0)}$ (top).

(c) $\varDelta\hat{\mathbf{x}}_{D1}$, $\varDelta\hat{\mathbf{x}}_{D2}$ & $\varDelta\hat{\mathbf{x}}_{D3}$ after a linear displacement of the control points, $\varDelta\hat{\mathbf{s}}$ (cf. figure 2.2(b)).

(d) Close-up that shows a consistent update of $\hat{\mathbf{x}}_{D1}$ (mesh) and $\hat{\mathbf{x}}_{D2}$ (points).

Figure 2.5: Demonstration of simultaneous control of several meshes using Vertex Morphing.

it covers the entire rectangle and hence also the remaining two meshes. For the filtering, we use a linear hat filter as describe in (2.2). Aiming for low-frequent and globally smooth updates, we specify a large filter radius of $r = 0.3$, which corresponds to 30% of the width of the rectangle. In order to obtain a consistent parameterization, we use the same filter function and the same filter radius for every combination of $\varDelta\hat{\mathbf{x}}_{Dk}$ and $\varDelta\hat{\mathbf{s}}$. Eventually, we obtain the following parameterization:

$$\underset{1170\times1}{\varDelta\hat{\mathbf{x}}_{D1}} = \underset{1170\times1170}{\mathbf{A}_1} \quad \underset{1170\times1}{\varDelta\hat{\mathbf{s}}} \tag{2.14a}$$

$$\underset{303\times1}{\Delta\hat{\mathbf{x}}_{D2}} = \underset{303\times1170}{\mathbf{A}_2} \quad \underset{1170\times1}{\Delta\hat{\mathbf{s}}} \tag{2.14b}$$

$$\underset{303\times1}{\Delta\hat{\mathbf{x}}_{D3}} = \underset{303\times1170}{\mathbf{A}_3} \quad \underset{1170\times1}{\Delta\hat{\mathbf{s}}} \tag{2.14c}$$

According to previous relations, a displacement of the control points will cause a simultaneous update of all meshes. Since we use the same control field and the same filter settings in the construction of $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3$, the updates will be consistent as they all describe the same underlying geometry update. Note that $\hat{\mathbf{x}}_{D1}$ represents a volume mesh in this 2D case. Choosing the latter for the discretization of the control field, one can control the entire volume mesh at once. Accordingly, the computation of (2.14a) implies an extended integration as presented in section 2.3. Note that with the chosen discretization of the control field, $\mathbf{A}_1$ is square, but $\mathbf{A}_2$ and $\mathbf{A}_3$ are rectangular matrices.

Depending on the displacement of the control points, (2.14) will lead to different updates of the geometry. The figures 2.5(c) and 2.5(d) present the result for a linear displacement of the control points. The latter corresponds to the linear displacement shown in figure 2.2(b).

From the results, we first observe that the specified displacement of the control points translates into a simultaneous update of *all* meshes. So, a separate mapping of the update from one domain to another is not necessary. Moreover, the updates in all meshes are *smooth*. In fact, since $\hat{\mathbf{s}}$ is linear and a linear filter function was defined, the update along any curve within the rectangle corresponds to a cubic B-Spline. Note in this context that the B-Spline is visible in all three meshes. The observation of the same B-Spline in all meshes indicates the fact that the smoothing in Vertex Morphing equally applies to all controlled meshes.

The second striking observation is that all three meshes essentially show the same geometric update, just for different parts of the domain. This observation visualizes the fact that the specified parameterization does not depend on the actual discretization of the geometry. From the result, one can also observe that this mesh-independence yields a *consistent* update of all meshes, i.e., the position of the meshes relative to each other does not change and initially overlapping sections do not drift apart. Figure 2.5(d) provides a close-up that highlights this effect. Note herein that $\hat{\mathbf{x}}_{D2}$ perfectly follows $\hat{\mathbf{x}}_{D1}$. Moreover, looking at the locations where $\hat{\mathbf{x}}_{D1}$ and $\hat{\mathbf{x}}_{D2}$ coincide, one can see that nodes with the same coordinates are subjected to the same geometric update. Again this reflects the mesh-independent parameterization. In between the overlapping nodes, $\hat{\mathbf{x}}_{D2}$ further resolves the underlying geometry due to its finer discretization.

The last striking observation from the result in figure 2.5(c) is that, in combination with the extension from section 2.3, the presented parameterization can control various types of meshes at the same time. In this case, for example, it controls a volume mesh together with two discrete lines. Note in this context that all meshes are linked together: $\hat{\mathbf{x}}_{D2}$ is attached to the boundary of $\hat{\mathbf{x}}_{D1}$ and $\hat{\mathbf{x}}_{D3}$ is embedded into $\hat{\mathbf{x}}_{D1}$ along the center-line of the rectangle. The consistent modification of all meshes preserves these geometric links.

2.5 Damping

Vertex Morphing allows for a high degree of freedom in the optimization of design surfaces. In practice, this freedom is often limited by interfaces to adjacent non-design areas, which must not be modified. Establishing a valid transition between design and non-design domain requires the definition of relevant geometric constraints. Generally, such geometric constraints must be formulated explicitly and considered by the optimization algorithm. However, in Vertex Morphing, there is a simple alternative, i.e., *damping*. This section introduces the relevant approach.

Damping means weighting the local shape updates such that: 1) They are zero directly at the interface to the non-design domain. 2) They are continuously increasing within a transition zone. 3) They are not modified after some distance away from the non-design domain. The implementation of damping is straightforward and only affects the parameterization. One only needs to multiply the local shape or geometry update by a damping factor $d \in [0,1]$:

$$\Delta \hat{x}_{D,i} = d_i A_{ij} \Delta \hat{s}_j \tag{2.15}$$

The damping factor is computed based on the distance of a given node in $\hat{\boldsymbol{\xi}}_G$ to the non-design domain. The corresponding function is called *damping function*. Generally, the damping function is arbitrary, and its only purpose is to establish a smooth transition between the extreme values of

$d = 0$ (fully damped geometry update) and $d = 1$ (no active damping). A reasonable choice is a function, which shows vanishing tangents at $d = 0$ and $d = 1$. By that, the local impact on the continuity of the geometry update is reduced. A possible function, which we also use in this work, is the following:

$$d_i = \begin{cases} 0.5 - 0.5\cos\left(\pi \dfrac{\hat{\xi}_{G,i} - \hat{\xi}_{\partial\Gamma,i}}{r_d}\right), & \text{if } |\hat{\xi}_{G,i} - \hat{\xi}_{\partial\Gamma,i}| < r_d \\ 1, & \text{otherwise} \end{cases} \tag{2.16}$$

Herein, $\hat{\xi}_{G,i}$ represents the x-, y-, or z-coordinate of a node in $\hat{\boldsymbol{\xi}}_G$. The part of $\hat{\boldsymbol{\xi}}_G$ which is connected to the non-design domain is referred to as $\hat{\boldsymbol{\xi}}_{\partial\Gamma} \subset \hat{\boldsymbol{\xi}}_G$. For each node in $\hat{\boldsymbol{\xi}}_G$, one determines the closest node in $\hat{\boldsymbol{\xi}}_{\partial\Gamma}$. The x-, y-, or z-coordinate of the closest node is represented by $\hat{\xi}_{\partial\Gamma,i}$. Hence, $\left(\hat{\xi}_{G,i} - \hat{\xi}_{\partial\Gamma,i}\right)$ describes the distance of a given node to the non-design domain in one of the three coordinate directions.

r_d specifies a *damping radius*. The latter defines the transition zone with active damping. Generally, the damping radius is arbitrary and depends on the individual application. It is recommended, though, to chose d in the order of the filter radius r so that the modified geometry update has a similar characteristic length-scale as the original one.

The central characteristic of damping is that it is simple to implement but still effective. Damping corresponds to an implicit formulation of geometric constraints by which certain design areas may be fixed. "Implicit" means that the relevant constraint is directly incorporated in the parameterization instead of explicitly formulated as a separate function. Consequently, damping has no impact on the choice of the optimization algorithm. Another important characteristic of damping is that it can be applied in all coordinate directions independently. That is, one may use damping to suppress updates of the geometry in one direction but still allow modifications in all the others.

The effect of damping shall be demonstrated in the following. Therefore, consider again the example introduced in the previous section (figure 2.5). In this example, we used Vertex Morphing to update three different meshes simultaneously. In doing so, we assumed a linear displacement of the control field. The latter led to the mesh (geometry) updates presented in figure 2.5(c). In this figure, one could see that the assumed displacement of the control points causes a significant update in all meshes. Note that such an update is problematic if some boundaries are not allowed to move.

Given the same linear displacement of the control points, we now assume that the left and right boundary of the rectangular structure must not be modified, since they interface to a surrounding part, for example. Damping shall be applied to establish a transition between the fixed boundaries and the actual mesh update computed based on the Vertex Morphing Method. For this purpose, the relation in (2.14) is adjusted as follows:

$$\Delta \hat{x}_{D1,i} = d_i A_{1,ij} \Delta \hat{s}_j \tag{2.17a}$$

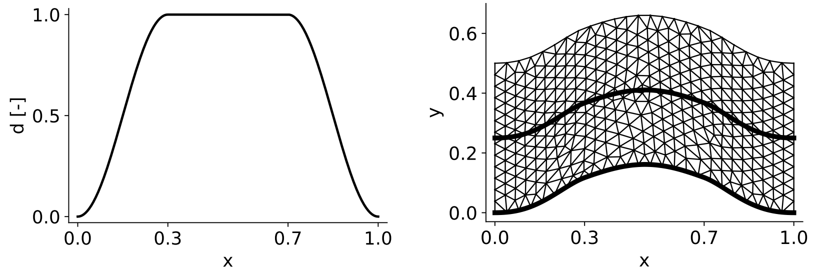
$$\Delta \hat{x}_{D2,i} = d_i A_{2,ij} \Delta \hat{s}_j \tag{2.17b}$$

$$\Delta \hat{x}_{D3,i} = d_i A_{3,ij} \Delta \hat{s}_j \tag{2.17c}$$

In all of the previous equations, the damping factors, d_i, are computed using the damping function from (2.16) with a damping radius of $r_d = r = 0.3$. So, the damping radius is defined to be equal to the filter radius. The resulting damping function is visualized in figure 2.6(a). Note in the figure that only in the range of one damping radius around the left and the right edge ($x = 0$ and $x = 1$) $d \neq 1$. This range corresponds to the zone of active damping. Within that zone, the damping factor continuously decreases towards the non-design domain until directly at the fixed edges the mesh updates are completely suppressed, i.e., $d = 0$.

Using the adjusted parameterization from (2.17), the linear displacement of the control points from before leads to the new mesh updates presented in figure 2.6(b). In the figure, one can observe the following:

- The damping successfully avoids a modification of the fixed boundaries (left and right edge). Therefore, compare the result to the one in figure 2.5(c), which, by contrast, shows a vertical displacement everywhere in the mesh.
- Within the damping zone, there is a continuous transition between the fixed boundary and the modified meshes.



(a) Damping function based on (2.16).

(b) Same case as in figure 2.5, but with active damping. The figure shows $\Delta\hat{\mathbf{x}}_{D1}$, $\Delta\hat{\mathbf{x}}_{D2}$ & $\Delta\hat{\mathbf{x}}_{D3}$ after a linear displacement of the control points.

Figure 2.6: Example of Vertex Morphing with active damping.

- Beyond the damping zone, the mesh updates correspond to the ones from figure 2.5(c). This result is expected, as the parameterization does not differ from the previous example in this region.
- The damping equally applies to all meshes, so that the updates remain consistent. Note from this result that damping may be well combined with all the above-presented extensions of Vertex Morphing.
- There is a slight hump in all meshes at $x = 0.3$ and $x = 0.7$.

The last characteristic is a consequence of the fact that we compute the damped geometry by multiplication of two functions with different continuity properties, i.e., the damping function and the function that describes the geometry update. This difference leads to a jump in the continuity of the damped update at the transition from $d < 1$ to $d = 1$. Generally, it holds that damping can cause visible changes in the continuity of the geometry update at the borders of the damping zone.

Also important to mention is that the simple scaling may cause the damping function to become visible within the damping zone. Note, e.g., the apparent cosine curve around the fixed boundaries in figure 2.6(b). Moreover, if the damping radius is chosen to be different from the filter radius, then the damped geometry update will show different characteristic length scales in the damping zone and the rest of the design domain.

The continuity jumps, the visibility of the damping function, and the possible influence on the characteristic length scale are the essential disadvantages of the presented damping. In practice, though, these disadvantages are often acceptable in favor of a feasible design. To minimize the negative influence of damping, one may restrict the damping to specific coordinate directions or apply different damping radii in different regions of the mesh.

If damping is applied within a gradient-based shape optimization process, it must also be considered in the sensitivity analysis. The damping factors are typically determined only once at the beginning of the optimization based on the initial design. That is, the damping factors are not depending on the updates of the control points. Therefore, its integration into a given sensitivity analysis is straightforward. Considering (2.15) and given that $\hat{x}_i = \hat{x}_i^{(0)} + \Delta \hat{x}_i$, the adjusted sensitivity analysis for an arbitrary response function I reads:

$$\frac{dI}{d\Delta\hat{s}_i} = \frac{dI}{d\hat{x}_{D,i}} \frac{d\hat{x}_{D,i}}{d\Delta\hat{s}_j} = \frac{dI}{d\hat{x}_{D,i}} d_i A_{ij} \tag{2.18}$$

CHAPTER 3

OPTIMIZING INTERNAL FLOWS UNDER STRESS CONSTRAINTS

In this chapter, we develop a method for the shape optimization of structures with internal flow[1]. The method shall allow optimizing internal flows[2] while considering stress constraints from the surrounding structure. Moreover, it shall enable a quick exploration of design potential based on input from high-fidelity simulations.

3.1 Problem formulation and assumptions

Optimizing an internal flow while considering stress constraints from the surrounding structure represents a multidisciplinary optimization problem, which includes an analysis of the present fluid-structure interaction. In terms of the latter, we assume a stationary one-way coupling. "Stationary" means, steady-state conditions are applicable, and time derivatives

[1] like, for example, valve blocks or tubomachinery

[2] for example, in terms of pressure loss or similar objective functions

can be neglected. "One-way coupling" means the fluid exerts forces on the structure, and the structure responds with a displacement that causes internal stresses. However, a coupling in the opposite direction is neglected. That is, displacements are assumed to be small enough not to alter the fluid flow. The assumption of a stationary one-way coupling is reasonable in many practical applications, especially when dealing with structures made from high-strength materials.

The multidisciplinary shape optimization problem is formulated based on a two-field approach, such as in Fazzolari et al. [19] or Kenway et al. [24]. In the following, we assume that the relevant equations are already discretized. In discrete form, the optimization problem reads:

$$\begin{aligned}
&\underset{\mathbf{s}}{\text{minimize}} && J(\mathbf{x}_F(\mathbf{s}),\mathbf{w}) && && \text{(3.1a)}\\
&\text{subject to} && \mathbf{R}_F(\mathbf{x}_F(\mathbf{x}_{D,F}(\mathbf{s})),\mathbf{w})=\mathbf{0} && && \text{(3.1b)}\\
& && \mathbf{R}_S(\mathbf{x}_S(\mathbf{x}_{D,S}(\mathbf{s})),\mathbf{u},\mathbf{w})=\mathbf{0} && && \text{(3.1c)}\\
& && g_i \le 0 && && \text{(3.1d)}\\
&\text{whereas} && g_i = \sigma_i(\mathbf{x}_S(\mathbf{x}_{D,S}(\mathbf{s})),\mathbf{u},\mathbf{w})-\sigma_{\max} && i=1,\dots,n_\sigma && \text{(3.1e)}
\end{aligned}$$

Herein, J represents the objective function, i.e., a function that describes a characteristic of the internal flow. $\mathbf{R}_F$ and $\mathbf{R}_S$ denote the governing equations of the fluid and the structure in residual form. $\mathbf{w}$ and $\mathbf{u}$ are the state vectors of the fluid and the structure, respectively. g_i refers to inequality constraints that limit the allowed stresses inside the structure. σ_i represents the relevant internal stress, which we herein measure as von Mises stresses. n_σ denotes the total number of stress constraints and $\sigma_{\max}$ specifies the upper threshold.

$\mathbf{x}_F$ and $\mathbf{x}_S$ are the coordinates of the nodes of the fluid and the structure mesh, respectively. The "wet" interface between both domains is referred to as $\mathit{\Gamma}_I$. On $\mathit{\Gamma}_I$, the meshes are assumed to be non-matching.

The design surface is represented by $\mathit{\Gamma}_D$ and the geometry of the design surface by $\mathbf{x}_D$. To avoid clutter in the later derivations, we drop the $\hat{\bullet}$ in the description of discrete geometries:

$$\hat{\mathbf{x}} \rightarrow \mathbf{x} \tag{3.2}$$

We assume that Γ_D includes Γ_I either in parts or entirely. So, given two non-matching meshes, there are two different discretizations of the design surface. Depending on the domain, we refer to the discrete geometry of the design surface as $\mathbf{x}_{D,F}$ or $\mathbf{x}_{D,S}$. Along the common boundary, a modification of the geometry on one side needs to be consistently mapped to the other.

$\mathbf{x}_{D,F}$ and $\mathbf{x}_{D,S}$ are controlled by a set of design variables, $\mathbf{s}$. Generally, the design variables are determined by the parameterization strategy. In this work, we use the Vertex Morphing Method to parameterize the shape. Details on the parameterization and the actual definition of the design variables follow in section 3.3.

During the optimization, $\mathbf{x}_{D,F}$ and $\mathbf{x}_{D,S}$ are subject to changes. In both domains, the changes on the boundary must be translated into the interior to avoid distorting meshes. Dedicated mesh motion strategies are used to determine the corresponding mesh motion. Given the fluid and the structure domain, we require two strategies. In each strategy, we compute an update of the volume mesh, $\Delta\mathbf{x}_F$ or $\Delta\mathbf{x}_S$, based on an update of the geometry on the design surface, $\Delta\mathbf{x}_{D,F}$ or $\Delta\mathbf{x}_{D,S}$.

Details on the fluid, the structure and the coupling

The fluid is assumed to be incompressible, viscous, fully turbulent, and governed by the Reynolds-averaged Navier–Stokes (RANS) equations combined with the Menter SST turbulence model. We consider the finite volume method for the numerical solution of the fluid problem. The application of RANS is essential for the herein presented optimization approach. By contrast, the incompressibility, the turbulence model, and the finite volume method are just specific assumptions resulting from the targeted application. The proposed optimization approach is expected to work equally well for compressible flows, other turbulence models, or a finite element approach. Given the assumption of a one-way coupling, the fluid flow is independent of the state of the structure.

For the structure, an elastic behavior without material or geometric nonlinearity is assumed. We consider the finite element method for the numerical analysis of the structure. Accordingly, it holds:

$$\mathbf{R}_S = \mathbf{K}(\mathbf{x}_S(\mathbf{x}_{D,S}(\mathbf{s})))\mathbf{u} - \mathbf{f}_S(\mathbf{x}_{D,S}(\mathbf{s}), \mathbf{w}) = \mathbf{0} \tag{3.3}$$

Herein, $\mathbf{K}$ is the global stiffness matrix, which explicitly depends on the grid coordinates and, because of the mesh motion, implicitly depends on the design surface and so the design variables. $\mathbf{f}_S$ represents the external force vector acting on the structure. Due to the excitation of the structure by the fluid, $\mathbf{f}_S$ depends on the state variables of the fluid, $\mathbf{w}$. Furthermore, $\mathbf{f}_S$ is depending on $\mathbf{x}_{D,S}$ since the design surface extends over $\mathit{\Gamma}_I$.

The relation between $\mathbf{f}_S$ and $\mathbf{w}$ arises from the coupling condition. Due to the assumed one-way coupling, there is only one coupling condition, i.e., the requirement of dynamic equilibrium[3]. In continuous form, the dynamic equilibrium requires:

$$\tilde{\boldsymbol{\sigma}}_S \tilde{\mathbf{n}}_S = -\tilde{\boldsymbol{\sigma}}_F \tilde{\mathbf{n}}_F \quad \text{on } \mathit{\Gamma}_I \tag{3.4}$$

Herein, $\tilde{\mathbf{n}}_F$ and $\tilde{\mathbf{n}}_S$ refer to the local unit surface normals in both domains. We assume that $\tilde{\mathbf{n}}_F = -\tilde{\mathbf{n}}_S$. $\tilde{\boldsymbol{\sigma}}_F$ and $\tilde{\boldsymbol{\sigma}}_S$ are the local stress tensors. Since we assume only small displacements, there is no need to distinguish between the deformed and the undeformed state in the description of the stresses. Defining local traction vectors $\tilde{\mathbf{t}}_F$ and $\tilde{\mathbf{t}}_S$, one can abbreviate (3.4) to:

$$\tilde{\mathbf{t}}_S = \tilde{\mathbf{t}}_F \quad \text{on } \mathit{\Gamma}_I \tag{3.5}$$

Since the structure is excited by the fluid flow, $\tilde{\mathbf{t}}_S$ results from $\tilde{\mathbf{t}}_F$. Given the local surface normal $\tilde{\mathbf{n}}_F$, the local pressure value $\tilde{p}$, and the local viscous stress tensor $\tilde{\boldsymbol{\tau}}$, the tractions generated by the fluid flow can be computed according to:

$$\tilde{\mathbf{t}}_F = -\tilde{p}\tilde{\mathbf{n}}_F + \tilde{\boldsymbol{\tau}}\tilde{\mathbf{n}}_F \tag{3.6}$$

In discrete form with non-matching meshes on the interface, a point-wise fulfillment of (3.5) is impossible. Instead, the tractions have to be mapped from the fluid to the structure domain and the dynamic equilibrium must be approximated using, e.g., an energy conservative approach.

[3] Kinematic equilibrium is not required as we neglect the effect of the displacement on the fluid.

This work adopts a conservative mapping as it is proposed by Farhat et al. [62]. In the mentioned approach, tractions are mapped such that the energy is conserved across the interface. Therefore, consider the nodal forces defined as follows:

$$\mathbf{f}_F = \mathbf{M}_F \mathbf{t}_F \tag{3.7}$$

$$\mathbf{f}_S = \mathbf{M}_S \mathbf{t}_S \tag{3.8}$$

Herein, $\mathbf{t}_F$ and $\mathbf{t}_S$ represent the discrete global traction vectors. $\mathbf{M}_F$ and $\mathbf{M}_S$ denote the mass matrices associated with both surface meshes on $\mathit{\Gamma}_I$. Given the displacement of the fluid and the structure domain on the interface, $\mathbf{u}_F$ and $\mathbf{u}_S$, the conservation of energy requires:

$$\mathbf{u}_F^T \mathbf{f}_F = \mathbf{u}_S^T \mathbf{f}_S \quad \text{on } \mathit{\Gamma}_I \tag{3.9}$$

From the latter equation, one can derive a relation between the generated forces in the fluid and the resulting forces on the structure. Therefore, assume a generic linear map, with which we map displacements from the structure to the fluid domain. Introducing the constant mapping matrix $\mathbf{H}_{FS}$, it holds:

$$\mathbf{u}_F = \mathbf{H}_{FS} \mathbf{u}_S \tag{3.10}$$

We obtain the external forces acting on the interface of the structure by plugging (3.10) into (3.9) and rearranging the resulting equation:

$$\mathbf{f}_S = \mathbf{H}_{FS}^T \mathbf{f}_F \quad \text{on } \mathit{\Gamma}_I \tag{3.11}$$

There exist various techniques in terms of how to construct $\mathbf{H}_{FS}$. Among those, we, in this thesis, use the mortar method and the nearest element interpolation. Refer to Wang et al. [63] for a summary of both techniques.

3.2 Multidisciplinary solution approach

The optimization problem in (3.1) requires an objective J to be minimal, some constraints g_i to be feasible, and the underlying governing equations

$\mathbf{R}_F$ and $\mathbf{R}_S$ to be satisfied. The problem is multidisciplinary in terms of the analysis, as it requires to analyze a fluid-structure interaction. Also, it represents a multidisciplinary design optimization (MDO) problem, as it contains response functions with different physics involved. So, coupled solution strategies have to be applied to simulate the physical problem, and techniques from MDO are required to find a solution for the overall optimization problem.

Regarding the fluid-structure interaction, we assumed a one-way coupling. To establish the corresponding interaction, we apply a Neumann-coupling of forces, i.e., forces are transferred from the fluid to the structure, and there applied as Neumann boundary conditions. Since we neglect the reverse coupling of the displacements, no iterations are required to satisfy the coupling condition. We apply a partitioned solution strategy to exploit specialized disciplinary solvers. Time derivatives can be neglected due to the initially assumed stationary conditions. We compute the fluid-structure interaction in every optimization step, so that (3.1b) and (3.1c) are always satisfied. That is, (3.1b) and (3.1c) are no explicit constraints to be controlled by the optimization algorithm. This approach corresponds to a multidisciplinary feasible optimization strategy, cf. Martins et al. [64].

We assume high-fidelity models for the single disciplines meaning their solution is costly. A large number of function evaluations, like in zero-order optimization algorithms, is therefore impossible. Instead, we apply gradient-based optimization. For this purpose, gradient or sensitivity analysis is necessary. More precisely, sensitivity analysis is necessary in order to determine the objective gradient $dJ/d\mathbf{s}$ and the constraint gradients $dg_i/d\mathbf{s}$.

In the present case, there is only one objective function, but the number of stress constraints n_σ may be large (in the order of the number of elements in the structure model). Also, since Vertex Morphing is applied, the number of design variables is large (say up to several million design variables). An optimization with many response functions *and* many design variables is prohibitive. Therefore, we assume that stress constraints are either restricted to very few dedicated evaluation points or aggregated to very few compromise functions using relevant aggregation techniques. See, e.g., Lambe et al. [65] for an overview of such techniques. Under the previous assumptions, we need to compute gradients of only a few response functions with respect to many design variables. Therefore, we use the *adjoint* method to determine $dJ/d\mathbf{s}$ and $dg_i/d\mathbf{s}$.

In the determination of the gradients, we must consider the coupling between the fluid and the structure. Thus, we require a *coupled* adjoint sensitivity analysis. Like in the primal solution, we in the adjoint solution also opt for a partitioned solution strategy. The goal is to utilize specialized adjoint solvers in both physical domains and to avoid the computation of cross-derivatives so that we can use the solvers in a black-box manner. A coupled adjoint sensitivity analysis specialized for this purpose is developed in section 3.4. The sensitivity analysis follows a discrete adjoint approach, i.e., equations are first discretized and then derived.

3.3 Parameterization

The optimization problem specified in section 3.1 requires the definition of some design parameters, **s**, with which the shape of the structure is controlled. Given the multidisciplinary setup, the parameterization is a non-trivial task since not only the type of parameters must be specified, but one must also consider that the design surface is linked to different meshes. If the parameterization only acts on one mesh, shape updates need to be consistently mapped to the other.

Herein, we use the Vertex Morphing Method to parameterize the shape. The goal is to realize an optimization approach, which allows for a quick[4] exploration of design potential at maximum design freedom [5]. In doing so, we assume that the two given meshes represent a fixed initial geometry. Accordingly, we use Vertex Morphing to control geometry updates, cf. section 2.2. So, the design parameters in our case are the displacement of the control points, $\Delta\hat{\mathbf{s}}$. In order to avoid clutter in the later derivations, we keep referring to the set of design variables as **s**, knowing that:

$$\mathbf{s} = \Delta\hat{\mathbf{s}} \tag{3.12}$$

Given two different discretizations of the design surface, one that corresponds to the fluid domain, $\mathbf{x}_{D,F}$, and one that corresponds to the structure domain, $\mathbf{x}_{D,S}$, this thesis suggests a custom variant of Vertex Morphing.

[4] I.e., an explicit definition of parameters is not necessary.

[5] I.e., there are as many parameters as degrees of freedom in the mesh.

This variant takes advantage of all extensions presented in chapter 2. The resulting parameterization is customized to the present MDO problem in terms of two aspects:

1. A common set of control points is introduced to simultaneously control the geometry of the design surface in *both* physical models.
2. Control points are used to simultaneously control the geometry of the design surface *and* the entire volume mesh of the structure.

Regarding the first aspect

Rather than specifying design variables that only act on one mesh and mapping any geometry update to the other, we use Vertex Morphing to control $\Delta\mathbf{x}_{D,F}$ and $\Delta\mathbf{x}_{D,S}$ *simultaneously*. By that, Vertex Morphing serves two purposes: 1) it controls the shape, and 2) it replaces a separate mapping strategy. To use Vertex Morphing for both purposes, we apply the extension described in section 2.4. That is, we introduce two separate mapping matrices, $\mathbf{A}_{FS}$ and $\mathbf{A}_{SS}$, together with a common set of control points $\hat{\mathbf{s}}$. The displacement of the control points, $\Delta\hat{\mathbf{s}} = \mathbf{s}$, represents the set of design variables. Then we establish the following relation between $\mathbf{s}$ and the geometry update in both domains:

$$\Delta\mathbf{x}_{D,S} = \mathbf{A}_{SS}\mathbf{s} \tag{3.13a}$$

$$\Delta\mathbf{x}_{D,F} = \mathbf{A}_{FS}\mathbf{s} \tag{3.13b}$$

Note herein that a displacement of the control points leads to an update of the geometry in both domains. From section 2.4, we know that if the mapping matrices are constructed based on the same filter settings, then the underlying definition of the geometry or geometry update is consistent. Since a consistent geometry update is essential in the shape optimization of coupled problems, we use the same filter function and the same filter radius to construct both $\mathbf{A}_{FS}$ and $\mathbf{A}_{SS}$.

$\mathbf{s}$ in (3.13) implies a common discretization of the control field. For the discretization of the control field, we require a dedicated mesh, $\hat{\boldsymbol{\xi}}_C$. In order to avoid extra modeling effort, we can reuse one of the given meshes for this purpose. In this context, it is reasonable to use the coarser of both meshes as the latter determines the design freedom. Since in FSI problems, the fluid mesh is typically finer than the structure mesh, we, in this work, use the mesh of the structure for the discretization of the control field. That is, $\mathbf{x}_{D,S}^{(0)} = \hat{\boldsymbol{\xi}}_C$. Consequently, $\mathbf{A}_{SS}$ in (3.13) represents a square and $\mathbf{A}_{FS}$ a rectangular matrix.

Regarding the second aspect

Instead of focusing only on the design surface, we use Vertex Morphing also to control the entire volume mesh of the structure, $\mathbf{x}_S$. So, it holds:

$$\Delta\mathbf{x}_{D,S} = \mathbf{A}_{SS}\mathbf{s} \quad \rightarrow \quad \Delta\mathbf{x}_S = \mathbf{A}_{SS}\mathbf{s} \tag{3.14}$$

The approach corresponds to the simultaneous control of geometry and mesh by Vertex Morphing, described in section 2.3. The advantages of the selected approach are:

- No extra solver is needed to compute the mesh motion in the structure domain.
- Therefore, no additional contributions from such a solver need to be considered in the sensitivity analysis.
- The smoothing properties of Vertex Morphing can be utilized to maintain a valid structure mesh throughout the optimization.

The disadvantages of this approach are compiled in section 2.3. From the discussions in this section, we know that the larger the filter radius, the less relevant the disadvantages. In the context of this work, we assume only large filter radii[6] in favor of globally smooth shape changes. On this basis, we may neglect the disadvantages and exploit the advantages mentioned above.

Combination of both aspects

Combining both aspects above, we obtain the following parameterization:

[6] Large compared to the dimensions of the structure domain.

$$\Delta \mathbf{x}_S = \mathbf{A}_{SS}\mathbf{s} \tag{3.15a}$$

$$\Delta \mathbf{x}_{D,F} = \mathbf{A}_{FS}\mathbf{s} \tag{3.15b}$$

Herein, $\mathbf{A}_{SS}$ is a square mapping matrix with which a displacement of the control points is translated into an update of the entire structure mesh. Contrarily, $\mathbf{A}_{FS}$ is a rectangular mapping matrix with which a displacement of the control points is translated into a geometry update on the fluid side. Since both matrices are constructed using the same filter function and filter radius, $\Delta\mathbf{x}_S$ and $\Delta\mathbf{x}_{D,F}$ are consistent. By having two separate mapping matrices, a displacement of the control points automatically leads to an update of the geometry in both domains. A separate mapping of geometry updates between the domains is omitted.

Note from (3.15a) that on structure side, $\mathbf{s}$ controls all mesh nodes ($\mathbf{x}_S$), whereas on fluid side, $\mathbf{s}$ only controls the geometry on the design surface ($\mathbf{x}_{D,F}$). So, in the fluid domain, it is still necessary to adapt the volume mesh by an extra mesh motion strategy. Technically it would be possible to use Vertex Morphing to control both volume meshes simultaneously. However, since the fluid domain is typically highly refined, the construction and storage of the corresponding mapping matrix can quickly become prohibitive. For this reason, Vertex Morphing is only used to replace the mesh motion in the structure domain.

In summary: The proposed parameterization strategy bases on a custom variant of the Vertex Morphing Method and unifies three, actually separate, aspects of the given multidisciplinary problem: control of the design surface, mapping of shape (geometry) updates between the domains, and the mesh motion on the structure side. Using this new strategy, we obtain a parameterization that allows maximum design freedom without the necessity of an additional mapping of shape updates or a dedicated mesh motion solver for the structure.

Figure 3.1 illustrates the combined approach at a generic but representative example.

Continuous update of the reference geometry

As explained in chapter 2, the geometry or geometry update in Vertex Morphing is defined w.r.t. a reference configuration. So far, we implied that

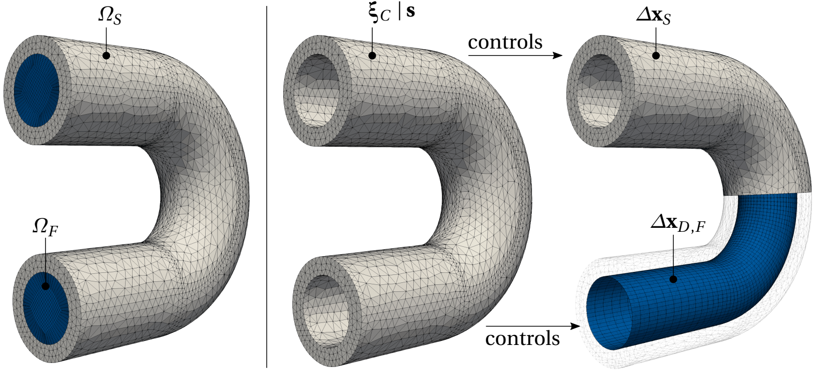


Figure 3.1: Illustration of the parameterization. Left: Discretization of both domains representing the initial configuration. Right: Control mesh ($\hat{\xi}_C$) with which a set of control points is defined whose displacement ($\mathbf{s}$) causes a shape update in the fluid domain ($\Delta\mathbf{x}_{D,F}$) and an update of the entire structure mesh ($\Delta\mathbf{x}_S$). There are as many control points as nodes in $\hat{\xi}_C$.

the reference configuration is fixed throughout the optimization (Total Lagrangian approach) and given by the initial design domain ($\Gamma_D^{(0)}$ or $\Omega_D^{(0)}$). In practice, however, the reference configuration is often updated continuously, such that it always corresponds to the geometry from the last optimization iteration (Updated Lagrangian approach). By that, one accounts for shape changes in the filtering with a fixed filter radius r. Without an update of the reference geometry, self-intruding surfaces may occur in case of big shape changes, for example. In the present work, we follow this common practice and continuously update the reference geometry in the optimization examples.

Continuously updating the reference geometry means that the filtering and the integration are done in the current configuration, whereas $\Gamma_D = \Gamma_D(\mathbf{s})$. Therefore, one has to consider the change in Γ_D when computing $d\mathbf{x}_D/d\mathbf{s}$ in the design sensitivity analysis, cf. (2.10). In the literature on Vertex Morphing, the influence of a changing reference on $d\mathbf{x}_D/d\mathbf{s}$ is neglected so far, cf. Bletzinger [8], Hojjat et al. [9], Bletzinger [10], and Najian Asl et al. [14]. Indeed, this is reasonable under the assumption of small step sizes and little changes in the geometry within one optimization iteration. We, in this work, follow the aforementioned common practice. So, despite continuously updating the reference geometry, we, in the optimization examples later, assume small steps sizes and thus:

$$\frac{d\mathbf{x}_D}{d\mathbf{s}} \approx \frac{\partial \mathbf{x}_D}{\partial \mathbf{s}} = \mathbf{A} \tag{3.16}$$

3.4 Coupled sensitivity analysis

The optimization problem in (3.1) shall be solved by gradient-based shape optimization. Therefore, shape gradients have to be computed for both the objective J and the constraints g_i, taking into account that the underlying governing equations must be satisfied. To obtain the relevant gradients, (coupled) adjoint sensitivity analysis is utilized. In this section, we derive the corresponding sensitivity analysis.

The derivations follow a discrete approach. I.e., they are based on the already discretized equations. Also, the resulting sensitivity analysis is customized such that:

- no cross-derivatives have to be computed,
- a partitioned solution approach is possible in which specialized single-disciplinary solvers can be used,
- no or just top-level adjustments have to be made to existing adjoint solvers.

For this purpose, we adopt the idea presented in Najian Asl [26] and Najian Asl et al. [27], where the authors describe a possibility to establish a partitioned adjoint sensitivity analysis for FSI problems without the need to compute cross-derivatives. The two mentioned works focus on fully coupled, fully nonlinear FSI problems. New in the present work is the rigorous derivation of such a sensitivity analysis for a one-way coupled FSI problem with a linear structure. The latter represents a variety of applications. The resulting sensitivity analysis with all the above-listed characteristics is a novelty in the field of MDO.

3.4.1 Formulation of the sensitivity analysis

To compute the derivatives of the response functions J and g_i w.r.t. the design variables $\mathbf{s}$, i.e., $dJ/d\mathbf{s}$ and $dg_i/d\mathbf{s}$, coupled adjoint sensitivity analysis is applied. In the following, we derive the corresponding formulation using the Lagrange formalism. Since the subsequent derivations are identical for J and g_i, I is used as a placeholder that represents both response functions.

Given I and the requirements that the governing equations of the fluid and the structure must be satisfied, i.e., $\mathbf{R}_F \overset{!}{=} \mathbf{0}$ and $\mathbf{R}_S \overset{!}{=} \mathbf{0}$, the following Lagrange functional can be defined:

$$L = I + \boldsymbol{\lambda}^T \mathbf{R}_F + \boldsymbol{\mu}^T \mathbf{R}_S \tag{3.17}$$

Herein, $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are arbitrary Lagrange multipliers associated with the fluid and the structure domain, respectively. Due to their meaning in the adjoint sensitivity analysis, the multipliers are referred to as the *adjoint state variables* or simply the *adjoint variables.*

By formulating a Lagrange functional, the constraints coming from the governing equations, i.e., (3.1b) and (3.1c), are eliminated from the set of explicit constraints in the optimization problem. Instead, they are implicitly included in the formulation of the response functions. Note in this context that, if the governing equations are satisfied, $L = I$. So the original problem is not altered.

To solve the optimization problem, we need to find a design, where:

$$\frac{dL}{d\mathbf{s}} \overset{!}{=} 0 \tag{3.18}$$

The relevant gradient can be computed as follows:

$$\frac{dL}{d\mathbf{s}} = \frac{dI}{d\mathbf{s}} + \boldsymbol{\lambda}^T \frac{d\mathbf{R}_F}{d\mathbf{s}} + \boldsymbol{\mu}^T \frac{d\mathbf{R}_S}{d\mathbf{s}}. \tag{3.19}$$

Expanding the previous equation by the individual total derivatives it becomes:

$$\frac{dL}{d\mathbf{s}} = \frac{\partial I}{\partial \mathbf{s}} + \frac{\partial I}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\mathbf{s}} + \frac{\partial I}{\partial \mathbf{u}}\frac{d\mathbf{u}}{d\mathbf{s}} + \boldsymbol{\lambda}^T\left(\frac{\partial \mathbf{R}_F}{\partial \mathbf{s}} + \frac{\partial \mathbf{R}_F}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\mathbf{s}} + \frac{\partial \mathbf{R}_F}{\partial \mathbf{u}}\frac{d\mathbf{u}}{d\mathbf{s}}\right)$$
$$+\boldsymbol{\mu}^T\left(\frac{\partial \mathbf{R}_S}{\partial \mathbf{s}} + \frac{\partial \mathbf{R}_S}{\partial \mathbf{w}}\frac{d\mathbf{w}}{d\mathbf{s}} + \frac{\partial \mathbf{R}_S}{\partial \mathbf{u}}\frac{d\mathbf{u}}{d\mathbf{s}}\right) \tag{3.20}$$

Reordering the equation yields:

$$\frac{dL}{d\mathbf{s}} = \frac{\partial I}{\partial \mathbf{s}} + \boldsymbol{\lambda}^T\frac{\partial \mathbf{R}_F}{\partial \mathbf{s}} + \boldsymbol{\mu}^T\frac{\partial \mathbf{R}_S}{\partial \mathbf{s}} + \left(\frac{\partial I}{\partial \mathbf{w}} + \boldsymbol{\lambda}^T\frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} + \boldsymbol{\mu}^T\frac{\partial \mathbf{R}_S}{\partial \mathbf{w}}\right)\frac{d\mathbf{w}}{d\mathbf{s}}$$
$$+\left(\frac{\partial I}{\partial \mathbf{u}} + \boldsymbol{\lambda}^T\frac{\partial \mathbf{R}_F}{\partial \mathbf{u}} + \boldsymbol{\mu}^T\frac{\partial \mathbf{R}_S}{\partial \mathbf{u}}\right)\frac{d\mathbf{u}}{d\mathbf{s}} \tag{3.21}$$

The partial derivatives in the previous equation represent the explicit dependencies of the response function and the governing equations on the state and the design variables. Given a solution of the coupled problem, i.e., a set of **w** and **u** that satisfies the governing equations, the partial derivatives can be computed explicitly.

In contrast to that, the total derivatives $d\mathbf{w}/d\mathbf{s}$ and $d\mathbf{u}/d\mathbf{s}$ refer to existing implicit dependencies. The computation of those requires a solution of the coupled governing equations for each component of **s**, which quickly becomes prohibitive. A possibility to avoid the costly computation of this total derivatives is to determine the Lagrange multipliers in (3.21) such that the multiplication terms in the brackets vanish identically. This approach gives rise to the *coupled adjoint equations* or the *coupled adjoint problem*:

$$\boldsymbol{\lambda}^T\frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} + \boldsymbol{\mu}^T\frac{\partial \mathbf{R}_S}{\partial \mathbf{w}} = -\frac{\partial I}{\partial \mathbf{w}} \tag{3.22a}$$

$$\boldsymbol{\lambda}^T\frac{\partial \mathbf{R}_F}{\partial \mathbf{u}} + \boldsymbol{\mu}^T\frac{\partial \mathbf{R}_S}{\partial \mathbf{u}} = -\frac{\partial I}{\partial \mathbf{u}} \tag{3.22b}$$

Both equations in the adjoint problem depend on the complete set of adjoint variables. So, similar to the original problem, the adjoint problem is also coupled. Moreover, the adjoint problem builds upon the solution of the original problem. Therefore, one typically refers to the original problem as the *primal problem* and **u** and **w** as the *primal variables*.

Given a solution of the coupled adjoint equations, i.e., a set of $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$that satisfies (3.22), the gradient of the Lagrangian can be computed using only, comparatively cheap, explicit operations:

$$\frac{dL}{d\mathbf{s}} = \frac{\partial I}{\partial \mathbf{s}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{s}} + \boldsymbol{\mu}^T \frac{\partial \mathbf{R}_S}{\partial \mathbf{s}} \tag{3.23}$$

In this so-called *adjoint approach*, the cost of computing $dL/d\mathbf{s}$ is independent of the number of variables. Instead, a coupled adjoint system has to be solved for every response function I. In cases of only a few response functions but many design variables, as in this work, the adjoint approach is superior to other types of sensitivity analysis.In the following, the terms in (3.23) are further specified.

Considering the governing equations of the structure in (3.3), the partial derivative of the latter w.r.t. the design variables reads:

$$\frac{\partial \mathbf{R}_S}{\partial \mathbf{s}} = \frac{\partial}{\partial \mathbf{s}}(\mathbf{Ku} - \mathbf{f}_S) = \frac{\partial \mathbf{Ku}}{\partial \mathbf{s}} - \frac{\partial \mathbf{f}_S}{\partial \mathbf{s}} \tag{3.24}$$

The external forces acting on the structure, i.e. $\mathbf{f}_S$, originate from the fluid domain and are determined by a force mapping along the interface, cf. (3.11). Plugging the latter into (3.24) and then (3.24) into (3.23), we obtain:

$$\frac{dL}{d\mathbf{s}} = \frac{\partial I}{\partial \mathbf{s}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{s}} + \boldsymbol{\mu}^T \frac{\partial \mathbf{Ku}}{\partial \mathbf{s}} - \boldsymbol{\mu}^T \frac{\partial}{\partial \mathbf{s}}(\mathbf{H}_{FS}^T \mathbf{f}_F) \tag{3.25}$$

The mapping matrix, $\mathbf{H}_{FS}^T$, is constructed based on the two non-matching meshes of the fluid and the structure alongside the common interface Γ_I. Since $\Gamma_I \subset \Gamma_D$, the mapping matrix is explicitly depending on the geometry of the design surface and implicitly depending on the control points, so:

$$\mathbf{H} = \mathbf{H}\left(\mathbf{x}_{D,F}(\mathbf{s}), \mathbf{x}_{D,S}(\mathbf{s})\right) \tag{3.26}$$

Given the dependency of the mapping matrix on the control points, we can resolve (3.25) further using the product rule:

$$\frac{dL}{d\mathbf{s}} = \frac{\partial I}{\partial \mathbf{s}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{s}} + \boldsymbol{\mu}^T \frac{\partial \mathbf{Ku}}{\partial \mathbf{s}} - \boldsymbol{\mu}^T \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{s}} \mathbf{f}_F - \boldsymbol{\mu}^T \mathbf{H}_{FS}^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{s}} \tag{3.27}$$

Note herein that:

$$\boldsymbol{\mu}^T \mathbf{H}_{FS}^T = \boldsymbol{\mu}_F^T \tag{3.28}$$

which is equivalent to:

$$\mathbf{H}_{FS} \boldsymbol{\mu} = \boldsymbol{\mu}_F \tag{3.29}$$

The previous equation represents a consistent map of the adjoint state variables of the structure onto the boundary of the fluid alongside the common interface Γ_I. Plugging (3.28) into (3.27), we obtain:

$$\frac{dL}{d\mathbf{s}} = \frac{\partial I}{\partial \mathbf{s}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{s}} + \boldsymbol{\mu}^T \frac{\partial \mathbf{Ku}}{\partial \mathbf{s}} - \boldsymbol{\mu}^T \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{s}} \mathbf{f}_F - \boldsymbol{\mu}_F^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{s}} \tag{3.30}$$

The partial derivatives in the previous equation can be further specified using the chain rule as follows:

$$\frac{\partial I}{\partial \mathbf{s}} = \frac{\partial I}{\partial \mathbf{x}_F} \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}} \frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} + \frac{\partial I}{\partial \mathbf{x}_S} \frac{\partial \mathbf{x}_S}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \tag{3.31}$$

$$\frac{\partial \mathbf{R}_F}{\partial \mathbf{s}} = \frac{\partial \mathbf{R}_F}{\partial \mathbf{x}_F} \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}} \frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} \tag{3.32}$$

$$\frac{\partial \mathbf{Ku}}{\partial \mathbf{s}} = \frac{\partial \mathbf{Ku}}{\partial \mathbf{x}_S} \frac{\partial \mathbf{x}_S}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \tag{3.33}$$

$$\frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{s}} = \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,F}} \frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} + \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \tag{3.34}$$

$$\frac{\partial \mathbf{f}_F}{\partial \mathbf{s}} = \frac{\partial \mathbf{f}_F}{\partial \mathbf{x}_F} \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}} \frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} \tag{3.35}$$

Herein, $\partial \bullet / \partial \mathbf{x}$ results from the explicit dependency of a term on the node coordinates of a mesh. $\partial \mathbf{x}/\partial \mathbf{x}_D$ results from the applied mesh-motion strategy. The latter translates geometry updates on the design surface into a motion of the entire volume mesh. $\partial \mathbf{x}_D / \partial \mathbf{s}$ results from the dependency of the geometry on the design surface on the displacement of the control points.

Plugging (3.31) till (3.35) into (3.30), we get:

$$\begin{aligned}\frac{dL}{d\mathbf{s}} = {} & \frac{\partial I}{\partial \mathbf{x}_S}\frac{\partial \mathbf{x}_S}{\partial \mathbf{x}_{D,S}}\frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} + \frac{\partial I}{\partial \mathbf{x}_F}\frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}}\frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{x}_F}\frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}}\frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} \\ & + \boldsymbol{\mu}^T \frac{\partial \mathbf{Ku}}{\partial \mathbf{x}_S}\frac{\partial \mathbf{x}_S}{\partial \mathbf{x}_{D,S}}\frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} - \boldsymbol{\mu}^T \left(\frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,F}}\frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} + \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,S}}\frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \right) \mathbf{f}_F \\ & - \boldsymbol{\mu}_F^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{x}_F}\frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}}\frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} \end{aligned} \tag{3.36}$$

Simplified and sorted into a fluid, a mapping and a structure part, the sensitivity equation finally reads:

$$\begin{aligned}\frac{dL}{d\mathbf{s}} = {} & \left(\frac{\partial I}{\partial \mathbf{x}_F} - \boldsymbol{\mu}_F^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{x}_F} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{x}_F} \right) \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}}\frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} \\ & - \boldsymbol{\mu}^T \left(\frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,F}}\frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} + \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,S}}\frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \right) \mathbf{f}_F \\ & + \left(\frac{\partial I}{\partial \mathbf{x}_S} + \boldsymbol{\mu}^T \frac{\partial \mathbf{Ku}}{\partial \mathbf{x}_S} \right) \frac{\partial \mathbf{x}_S}{\partial \mathbf{x}_{D,S}}\frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \end{aligned} \tag{3.37}$$

Given the above sensitivity equation and the corresponding coupled adjoint system, (3.22), one can determine the searched gradient by first solving (3.22) for the adjoint variables and then use the result to compute $dL/d\mathbf{s}$ according to (3.37). In the following, this procedure is further customized to the herein discussed multidisciplinary problem.

3.4.2 Customization for Vertex Morphing

In section 3.3, a parameterization based on Vertex Morphing was introduced, which unifies the control of geometry and mesh. Given this parameterization, the sensitivity equation in (3.37) can be customized as follows.

First, since Vertex Morphing is used to control the entire structure mesh, a displacement of the control points has an immediate effect on the entire structure mesh. Consequently:

$$\frac{\partial \mathbf{x}_S}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} = \frac{\partial \mathbf{x}_S}{\partial \mathbf{s}} \tag{3.38}$$

Second, considering that in general $\mathbf{x} = \mathbf{x}^{(0)} + \Delta\mathbf{x}$, one can derive from (3.15a) and (3.15b) that:

$$\frac{\partial \mathbf{x}_{D,F}}{\partial \mathbf{s}} = \mathbf{A}_{FS} \tag{3.39}$$

$$\frac{\partial \mathbf{x}_S}{\partial \mathbf{s}} = \mathbf{A}_{SS} \tag{3.40}$$

Finally, inserting (3.38), (3.39), and (3.40) in (3.37), we obtain the following sensitivity equation:

$$\begin{aligned}\frac{dL}{d\mathbf{s}} = & \left(\frac{\partial I}{\partial \mathbf{x}_F} - \boldsymbol{\mu}_F^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{x}_F} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{x}_F}\right) \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS} \\ & - \boldsymbol{\mu}^T \left(\frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS} + \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}}\right) \mathbf{f}_F \\ & + \left(\frac{\partial I}{\partial \mathbf{x}_S} + \boldsymbol{\mu}^T \frac{\partial \mathbf{K}\mathbf{u}}{\partial \mathbf{x}_S}\right) \mathbf{A}_{SS}\end{aligned} \tag{3.41}$$

Comparing the previous equation with the general sensitivity equation in (3.37), one can see that there is no need anymore to compute $\partial \mathbf{x}_S / \partial \mathbf{x}_{D,S}$, i.e., the sensitivity contribution from a separate simulation of the mesh motion. Instead, we use $\mathbf{A}_{SS}$ to map sensitivities directly from the volume mesh to the control space. This characteristic greatly simplifies the sensitivity analysis.

Note that the new parameterization strategy only affects the sensitivity equation. It has no impact on the coupled adjoint system in (3.22).

3.4.3 Customization for one-way coupling

This work considers problems with a one-way coupled fluid-structure interaction. Due to the one-way coupling, the fluid is not affected by the structure state, so:

$$\frac{\partial \mathbf{R}_F}{\partial \mathbf{u}} = \mathbf{0} \tag{3.42}$$

Also, the fluid forces are independent of the state of the structure, i.e.:

$$\frac{\partial \mathbf{f}_f}{\partial \mathbf{u}} = \mathbf{0} \tag{3.43}$$

Consequently:

$$\frac{\partial \mathbf{f}_S}{\partial \mathbf{u}} = \frac{\partial \mathbf{f}_S}{\partial \mathbf{f}_F}\frac{\partial \mathbf{f}_F}{\partial \mathbf{u}} = \mathbf{0} \tag{3.44}$$

So, no follower loads are acting on the structure. Accordingly, it holds:

$$\frac{\partial \mathbf{R}_S}{\partial \mathbf{u}} = \frac{\partial \mathbf{K}\mathbf{u}}{\partial \mathbf{u}} - \frac{\partial \mathbf{f}_S}{\partial \mathbf{u}} = \mathbf{K} \tag{3.45}$$

The previous relations allow for a simplification of the coupled adjoint system. Plugging (3.42) and (3.45) into (3.22), the coupled adjoint system reduces to:

$$\boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} + \boldsymbol{\mu}^T \frac{\partial \mathbf{R}_S}{\partial \mathbf{w}} = -\frac{\partial I}{\partial \mathbf{w}} \tag{3.46a}$$

$$\boldsymbol{\mu}^T \mathbf{K} = -\frac{\partial I}{\partial \mathbf{u}} \tag{3.46b}$$

Note in (3.46) that, due to the one-way coupling in the primal domain, also the adjoint problem is one-way coupled. I.e., (3.46a) depends on (3.46b), but not vice versa. Consequently, no iterations are necessary to solve the adjoint problem.

In this context, it is striking that the coupling direction is reversed. I.e., the adjoint state of the fluid in (3.46a) depends on the adjoint state of the structure in (3.46b). In the primal problem, by contrast, the state of the structure depends on the state of the fluid.

Important to mention is that the governing equations of the structure are self-adjoint since the stiffness matrix is symmetric ($\mathbf{K}^T = \mathbf{K}$). Therefore, (3.46b) can be solved by an analysis of the original structure for a new load case given by $-\partial I / \partial \mathbf{u}$.

Beyond the impact on the coupled adjoint system, there is no further influence of the one-way coupling on the overall sensitivity analysis. Especially, it does not affect the sensitivity equation in (3.41).

3.4.4 Gradient of the objective function

In this work, we assume the objective function represents a characteristic of the fluid, such as pressure loss, power loss, or similar quantities. In this context, we assume that the objective function does not have an explicit dependency on the structure state or mesh. Given those assumptions, the coupled adjoint system and the sensitivity equation simplifies as follows.

If the objective function does not depend on the structure state or mesh, it holds:

$$\frac{\partial I}{\partial \mathbf{x}_S} = \mathbf{0} \tag{3.47}$$

$$\frac{\partial I}{\partial \mathbf{u}} = \mathbf{0} \tag{3.48}$$

According to (3.48), the coupled adjoint system in (3.46) reduces to:

$$\boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} + \boldsymbol{\mu}^T \frac{\partial \mathbf{R}_S}{\partial \mathbf{w}} = -\frac{\partial I}{\partial \mathbf{w}} \tag{3.49a}$$

$$\boldsymbol{\mu}^T \mathbf{K} = \mathbf{0} \tag{3.49b}$$

From (3.49b), it may be readily concluded that $\boldsymbol{\mu} = 0$. So, the coupled adjoint problem reduces to an uncoupled adjoint fluid problem, which can be solved independently of the structure:

$$\boxed{\boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} = -\frac{\partial I}{\partial \mathbf{w}}} \tag{3.50}$$

The sensitivity equation also reduces accordingly. Plugging (3.47) and (3.48) in (3.41), we obtain the following sensitivity equation:

$$\boxed{\frac{dL}{d\mathbf{s}} = \left(\frac{\partial I}{\partial \mathbf{x}_F} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{x}_F} \right) \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS}} \tag{3.51}$$

The two boxes around the sensitivity equation and the adjoint system above identify the final set of equations used to compute the derivative of the objective function ($I = J$) w.r.t. to the design variables ($\mathbf{s}$) in the present application.

Solution procedure

(3.50) and (3.51) together form a regular CFD adjoint sensitivity analysis. Therein, the coupling completely vanished. Only an additional mapping of the surface sensitivities to the control space using $\mathbf{A}_{FS}$ is included.

The corresponding solution process to determine the searched sensitivity information is outlined in the following. Note in this context, that the adjoint sensitivity analysis requires the solution of the primal problem as input:

1. Solve the adjoint fluid system in (3.50) for the adjoint fluid state $\boldsymbol{\lambda}$.
2. Use $\boldsymbol{\lambda}$ in (3.51) to compute the raw surface sensitivities $dL/d\mathbf{x}_F$.
3. Determine the sensitivity contribution from the mesh motion in the fluid domain, i.e., $d\mathbf{x}_F/d\mathbf{x}_{D,F}$, and multiply it with the result from 2. to obtain the complete surface sensitivities, $dL/d\mathbf{x}_{D,F}$.
4. Map the surface sensitivities to the control space using the mapping matrix from Vertex Morphing, i.e. $\mathbf{A}_{FS}$. Finally, one obtains $dL/d\mathbf{s}$

3.4.5 Gradient of the constraint function(s)

In this work, we assume that constraints are given in the form of allowable stresses in the structure. In this case, the coupled adjoint system and the sensitivity equation can be simplified as follows.

The stresses in the structure do not explicitly depend on the fluid mesh or the fluid state, so:

$$\frac{\partial I}{\partial \mathbf{x}_F} = \mathbf{0} \tag{3.52}$$

$$\frac{\partial I}{\partial \mathbf{w}} = \mathbf{0} \tag{3.53}$$

By contrast, the governing equations of the structure depend on the state variables of the fluid through the force mapping. Considering (3.3) and (3.11), it holds:

$$\frac{\partial \mathbf{R}_S}{\partial \mathbf{w}} = -\frac{\partial \mathbf{f}_S}{\partial \mathbf{w}} = -\frac{\partial \mathbf{f}_S}{\partial \mathbf{f}_F}\frac{\partial \mathbf{f}_F}{\partial \mathbf{w}} = -\mathbf{H}_{FS}^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{w}} \tag{3.54}$$

Introducing (3.53) and (3.54) into (3.46) yields a custom variant of the coupled adjoint system:

$$\boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} - \boldsymbol{\mu}^T \mathbf{H}_{FS}^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{w}} = \mathbf{0} \tag{3.55a}$$

$$\boldsymbol{\mu}^T \mathbf{K} = -\frac{\partial I}{\partial \mathbf{u}} \tag{3.55b}$$

Rearranging the equations and using the relation in (3.28), we finally obtain the following coupled adjoint system for the stress constraints:

$$\boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} = \boldsymbol{\mu}_F^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{w}} \tag{3.56a}$$

$$\boldsymbol{\mu}^T \mathbf{K} = -\frac{\partial I}{\partial \mathbf{u}} \tag{3.56b}$$

We recall that $\boldsymbol{\mu}_F^T$ results from a consistent map of the adjoint state variables of the structure onto the boundary of the fluid alongside the common interface $\mathit{\Gamma}_I$, cf. section 3.4.1.

The simplifications mentioned in the beginning of this section also impact the sensitivity equation. Introducing (3.52) in (3.41), we obtain the following reduced sensitivity equation:

$$\begin{aligned} \frac{dL}{d\mathbf{s}} = & \left(-\boldsymbol{\mu}_F^T \frac{\partial \mathbf{f}_F}{\partial \mathbf{x}_F} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{x}_F}\right) \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS} \\ & - \boldsymbol{\mu}^T \left(\frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS} + \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \right) \mathbf{f}_F \\ & + \left(\frac{\partial I}{\partial \mathbf{x}_S} + \boldsymbol{\mu}^T \frac{\partial \mathbf{Ku}}{\partial \mathbf{x}_S} \right) \mathbf{A}_{SS} \end{aligned} \tag{3.57}$$

Note from (3.57) and (3.56) that stress constraints require a *coupled* adjoint sensitivity analysis. That is, the adjoint system represents a coupled problem, and the sensitivity equation contains contributions from both domains. Both properties reflect the fact that the stresses are a consequence of the present fluid-structure interaction. The following subsection presents a solution procedure, which considers the physical interaction in the sensitivity analysis of the stress constraints. The procedure is based on existing single-disciplinary solvers and does not require the computation of cross derivatives. Instead, it relies on an exchange of boundary conditions.

Solution procedure

Looking closer at the final sensitivity equation and the coupled adjoint system, we find that (3.56b) together with the last row of (3.57) form a regular adjoint sensitivity analysis of the structure for the given response function I:

$$\boxed{\begin{aligned} & \boldsymbol{\mu}^T \mathbf{K} = -\frac{\partial I}{\partial \mathbf{u}} && \text{(3.58a)} \\ & \frac{dL_S}{d\mathbf{s}} = \left(\frac{\partial I}{\partial \mathbf{x}_S} + \boldsymbol{\mu}^T \frac{\partial \mathbf{Ku}}{\partial \mathbf{x}_S} \right) \mathbf{A}_{SS} && \text{(3.58b)} \end{aligned}}$$

That means, a part of the final sensitivity equation in (3.57) can be computed by a separate adjoint sensitivity analysis of the structure according to (3.58). We denote this part $dL_S/d\mathbf{s}$.

Furthermore, by combining (3.56a) with the first row in (3.57), and moving the adjoint state vector of the structure inside the adjacent partial derivative, we obtain a regular adjoint sensitivity analysis of the fluid. In this sensitivity analysis, the response function is not I, but an "auxiliary" functional, I_f, which depends on the fluid forces, $\mathbf{f}_F$, along the wet interface, Γ_I:

$$\boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{w}} = -\frac{\partial I_f}{\partial \mathbf{w}} \tag{3.59a}$$

$$\frac{dL_F}{d\mathbf{s}} = \left(\frac{\partial I_f}{\partial \mathbf{x}_F} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}_F}{\partial \mathbf{x}_F} \right) \frac{\partial \mathbf{x}_F}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS} \tag{3.59b}$$

where

$$I_f = -\boldsymbol{\mu}_F^T \mathbf{f}_F \quad \text{on } \Gamma_I \tag{3.59c}$$

The previous sensitivity analysis is similar to the sensitivity analysis for a drag or a lift response. The only difference is that the fluid force is not evaluated in a constant direction,i.e., in flow-direction for drag or perpendicular to it for lift, but in a locally varying direction defined by $-\boldsymbol{\mu}_F$. That means, given $\boldsymbol{\mu}_F$ from the solution of the adjoint structure and the subsequent mapping of of it to the fluid domain, we can compute a part of the sensitivity equation in (3.57) by a separate adjoint sensitivity analysis of the fluid according to (3.59). We denote this part $dL_F/d\mathbf{s}$.

Considering (3.57) and the definition of $dL_F/d\mathbf{s}$ and $dL_S/d\mathbf{s}$, we can compute the complete sensitivity information as follows:

$$\boxed{\frac{dL}{d\mathbf{s}} = \frac{dL_F}{d\mathbf{s}} - \boldsymbol{\mu}^T \left(\frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS} + \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \right) \mathbf{f}_F + \frac{dL_S}{d\mathbf{s}}} \tag{3.60}$$

The boxes above identify the complete set of equations with which we can compute the gradient of the stress constraints, i.e., $dg_i/d\mathbf{s} = d\sigma_i/d\mathbf{s}$.

Note that, by computing $dL/d\mathbf{s}$ instead of $dg_i/d\mathbf{s}$, the resulting gradient information considers the coupling in the underlying physical problem.

Based on the previous observations, this thesis suggests the following solution procedure for the gradient computation of the constraints.

1. Run a separate adjoint senstivity analysis of the structure according to (3.58). The solution yields the adjoint state $\boldsymbol{\mu}$ and the first sensitivity contribution, $dL_S/d\mathbf{s}$.

2. Map $\boldsymbol{\mu}$ on Γ_I from the boundary of the structure to the boundary of the fluid using (3.28) in order to obtain $\boldsymbol{\mu}_F$.

3. Use $\boldsymbol{\mu}_F$ and compute the second sensitivity contribution, $dL_F/d\mathbf{s}$, by a separate adjoint sensitivity analysis of the fluid for the specified auxiliary force response, cf. (3.59). Therefore, one can reuse an existing adjoint solver that computes sensitivites for standard fluid forces (e.g., drag). Only the force direction needs to be locally adjusted to $-\boldsymbol{\mu}_F$. Depending on the implementation, or the underlying type of differentiation (analytic vs. algorithmic), this adjustment requires either no or just minimal insight into the given adjoint solver.

4. Compute the third gradient contribution from the force mapping as required in (3.60).

5. Sum up $dL_F/d\mathbf{s}$, $dL_S/d\mathbf{s}$, and the contribution from the force mapping and finally obtain $dL/d\mathbf{s}$.

Some concluding remarks about the solution procedure:

- The procedure avoids the computation of cross-derivatives. Instead, the relevant terms are computed by another adjoint fluid analysis based on an auxiliary force response. The idea of avoiding an explicit computation of cross-derivatives by running another adjoint fluid analysis with an auxiliary force response was already presented in Najian Asl [26] in the context of general FSI problems. The idea was adopted and customized to the present one-way coupling.

- The solution procedure is fully partitioned. I.e., specialized single-disciplinary solvers can be used to compute the individual sensitivity contributions.

- The information flow in the adjoint analysis is reverse to the information flow in the primal analysis. In the primal analysis, forces must be mapped from the fluid to the structure, and the fluid is independent of the structure. By contrast, in the adjoint analysis, the adjoint state of the structure, $\boldsymbol{\mu}$, must be mapped from the structure to the fluid domain, i.e., the structure acts independently of the fluid.

3.4.6 Single-disciplinary gradient terms

The derivations above showed that only single-disciplinary sensitivity analyses are necessary to compute the (coupled) gradients in the solution of the given MDO problem, cf. (3.50), (3.51), (3.58), (3.59), and (3.60). This section elaborates on the actual computation of the single-disciplinary sensitivities:

Objective gradient

We assumed that the objective function describes a characteristic of the internal flow, like pressure loss. And we saw that the corresponding gradient can be computed independently of the state of the structure. Consequently, any regular adjoint fluid solver can be used for the computation of the objective gradient. In this work, the discrete adjoint solver of SU2 is used, which relies on algorithmic differentiation (AD), cf. Albring et al. [66]. The discrete adjoint solver of SU2 is chosen because:

- AD delivers consistent[7] sensitivities and automatically incorporates all parts of the fluid model. In particular, it takes into account the turbulence modeling (i.e., there is no frozen turbulence assumption, which may induce severe errors, Dwight et al. [67])

- The implemented sensitivity analysis is independent of the actual formulation of the response function. I.e., it can be readily used with new response functions (like the auxiliary force functional as required for the sensitivity analysis of the stress constraints).

- It is publicly available, open-source, and may be freely used, see SU2 Developer Community [68].

[7] i.e., exact within the discretization

- It relies on a shifted Lagrangian formulation, which exploits the fixed-point structure of the fluid solver. By that, the adjoint solver inherits the convergence properties of the primal solver. I.e., if the primal solver converges, also the adjoint will.

SU2 computes sensitivities of the form $d\bullet/d\mathbf{x}_{D,F}$. That means:

- SU2 includes sensitivity information from the mesh motion in the fluid domain ($\partial\mathbf{x}_F/\partial\mathbf{x}_{D,F}$).
- In the context of this work, the sensitives computed by SU2 still need to be multiplied by the mapping matrix $\mathbf{A}_{FS}$ in order to obtain $d\bullet/d\mathbf{s}$.

Constraint gradient

In section 3.4.5, it was shown that the gradient of the stress constraints, (3.60), includes three parts, which can be computed successively by an adjoint sensitivity analysis of the structure, an adjoint sensitivity analysis of the fluid, and an explicit evaluation of an explicit mapping term. In the following, the computation of the three parts is further detailed. In summary, we compute the three parts as follows:

$$\frac{dL}{d\mathbf{s}} = \underbrace{\frac{dL_F}{d\mathbf{s}}}_{\substack{\text{AD}\\\text{(SU2)}}} \underbrace{-\boldsymbol{\mu}^T \left(\frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,F}} \mathbf{A}_{FS} + \frac{\partial \mathbf{H}_{FS}^T}{\partial \mathbf{x}_{D,S}} \frac{\partial \mathbf{x}_{D,S}}{\partial \mathbf{s}} \right) \mathbf{f}_F}_{\text{neglected}} + \underbrace{\frac{dL_S}{d\mathbf{s}}}_{\substack{\text{Semi-analytic}\\\text{(Kratos)}}} \tag{3.61}$$

Part 1) The adjoint sensitivity analysis of the structure described in (3.58) is realized using a semi-analytic approach. That is, the finite difference method (FD) is applied to compute $\partial\mathbf{Ku}/\partial\mathbf{x}_S$ in the otherwise analytically derived sensitivity problem. The most important advantage of this approach is that it does not depend on the actual formulation of $\mathbf{K}$. So without changes, it applies to a wide range of problems. The main disadvantage is that FD introduces numerical errors, which, in some cases, may even increase as the mesh is refined. See Bletzinger et al. [69] for a discussion of this problem and a possible solution.

Apart from $\partial \mathbf{Ku}/\partial \mathbf{x}_S$, we also compute $\partial I/\partial \mathbf{x}_S = \partial \sigma_i/\partial \mathbf{x}_S$ by FD in order to avoid a deep code intrusion. Contrarily, we determine $\partial I/\partial \mathbf{u} = \partial \sigma_i/\partial \mathbf{u}$ analytically, because in the present case, it can be obtained by a simple evaluation of the stress tensor for a unit state of the structure, cf. appendix A.1. An analytic computation of $\partial I/\partial \mathbf{u}$ has the advantage that the right-hand side of the adjoint equation is exact. So, also the solution of (3.58a), meaning the adjoint state, will be numerically exact. By that, the accuracy problem of FD is limited to the sensitivity equation, (3.58b).

The presented semi-analytic sensitivity analysis was implemented in the open-source framework Kratos Multiphysics, see Kratos Developer Community [70]. Hence, it is publicly available and freely usable. The implementation allows computing sensitivities of the form $d \bullet /d\mathbf{x}_S$. So:

- Kratos does not include sensitivity information from a separate mesh motion, meaning Kratos does not compute $\partial \mathbf{x}_S/\partial \mathbf{x}_{D,S}$.
- The gradient still needs to be multiplied by the mapping matrix $\mathbf{A}_{SS}$ in order to obtain design sensitivities in the form $d \bullet /d\mathbf{s}$.

Part 2) The adjoint sensitivity analysis of the fluid, cf. (3.59), is realized using the already mentioned discrete adjoint solver of SU2, which relies on Algorithmic Differentiation.

Section 3.4.5 showed that (3.59) corresponds to a regular fluid sensitivity analysis with only a modified scalar force functional. Therefore, a new response function was added to SU2. The new response function is similar to the given drag function but allows for a local definition of the force direction. The necessary implementation effort is minor and locally limited to the definition of the response function. There are no changes in the actual adjoint solution process. Consequently, the adjoint solver can still be used as a black box, and all its advantages can be fully exploited. Notable advantages of the discrete adjoint solver in SU2 are the consistent gradient calculation and the desirable convergence properties.

Regarding the convergence properties, it is important to mention, that the developed coupled solution procedure has no impact on the shifted Lagrangian formulation utilized in the discrete adjoint solver of SU2. The reason is that, on the level of the solver, the only relevant change is the definition of the response function, and the latter has no impact on the shifted Lagrangian formulation.

Using the new response function, SU2 computes sensitivities of the form $d\bullet/d\mathbf{x}_{D,F}$. So:

- The sensitivity analysis already contains information from the mesh motion in the fluid domain, i.e., it contains $\partial\mathbf{x}_F/\partial\mathbf{x}_{D,F}$.
- The sensitivity information from SU2 still needs to be multiplied by the mapping matrix $\mathbf{A}_{FS}$ in order to obtain the required design sensitivities, $d\bullet/d\mathbf{s}$.

Part 3) The third ingredient is the mapping term. In the scope of this work, we neglect the mapping term. The main reason for this approach is that the relevant mapping is realized using black-box tools. A derivation of the mapping operation is therefore not readily possible. Neglecting the mapping term, however, is not considered critical because of two reasons:

1. Its origin is purely numerical, and it has no physical meaning. Instead, it reflects a direct mesh dependency of the computed gradient. So, by neglecting the mapping term, one assumes a sufficiently fine interface discretization in both domains, or one intentionally excludes non-physical gradient contributions.
2. Since its origin is purely numerical, the contribution of the mapping term is expected to be small compared to the physical contributions from the adjoint sensitivity analysis of the fluid and the structure. If this was not the case, then the accuracy of the underlying simulation needs to be questioned first.

By neglecting the mapping term, we accept a slight inconsistency between the actual and the computed gradient. Later results suggest that, in the scope of Vertex Morphing and robust first-order algorithms, this inconsistency is acceptable. A detailed investigation of the influence of the mapping term is left to follow-up research.

3.4.7 Concluding remarks

In this section, we developed a sensitivity analysis for the coupled optimization problem described in (3.1). It was shown, that the objective gradient requires a single-disciplinary adjoint sensitivity analysis of only the fluid, and the constraint gradients a *coupled* adjoint sensitivity analysis involving both domains.

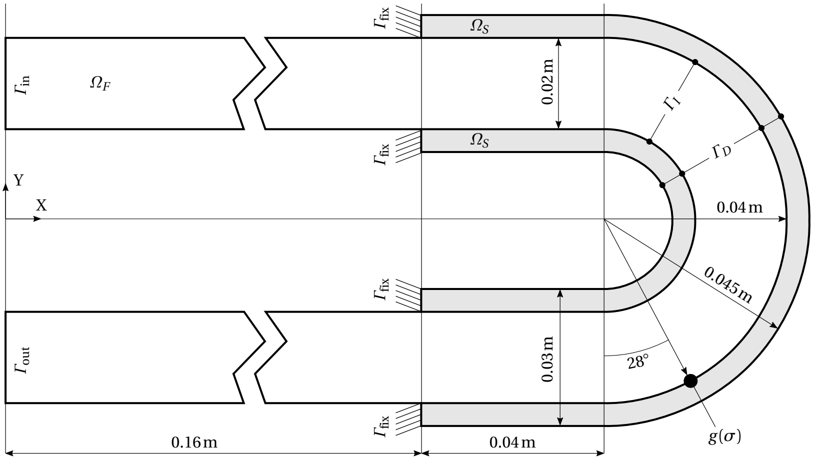
For the coupled adjoint sensitivity analysis, a strictly partitioned solution procedure was suggested, which relies on the exchange of boundary conditions, and the definition of an auxiliary force response function. By that, we avoided the explicit computation of cross-derivatives so that no intrusion into the individual adjoint solvers is required. As a result, we can exploit specialized single-disciplinary solvers to perform the necessary *coupled* sensitivity analysis.

Two different strategies were suggested for the two single-disciplinary sensitivity analyses within the coupled sensitivity analysis of the constraint. In the case of the fluid, AD is proposed. In the case of the structure, a semi-analytic strategy is suggested. Both strategies do not include any model assumptions, like a particular flow type or a specific element formulation. The combination of both strategies enables a broad field of application for the presented sensitivity analysis.

Regarding the actual definition of the constraint function(s), no specific formulations were assumed, only specific dependencies on the given state variables and meshes, see (3.52) and (3.53). So, the presented coupled sensitivity analysis is equally valid for any response function, which satisfies those properties.

3.5 Test cases

This section introduces two test cases with which the developed sensitivity analysis shall be verified, and the optimization process shall be tested. The section covers a two-dimensional test case, in which a fluid is guided through a generic bend, and a three-dimensional test case, which represents a generic valve housing with an internal flow channel. The purpose of the 2D test case is to verify the sensitivity analysis within a setup that is simple to reproduce. The 3D test case shall demonstrate the applicability of the complete optimization process in complex scenarios. Both test cases are inspired by applications from the field of fluid guidance or pressurized valve systems.



Ω_F		
$\rho_F = 1765\,\text{kg/m}^3$	$\mu_F = 0.022945\,\text{Ns/m}^2$	$Re = 10000$
$\mathbf{v}_{\text{in}} = \{6.5, 0, 0\}\text{m/s}$	$p_{\text{out}} = 0\,\text{Pa}$	

Ω_S		
$\rho_S = 7850\,\text{kg/m}^3$	$E = 1 \times 10^{13}\,\text{Pa}$	$\nu = 0.29$
$\mathbf{u}_{\text{fix}} = \{0, 0, 0\}\text{m}$	$\mathbf{f_S} = \mathbf{f_S}(\mathbf{f_F})\,[N]$ on Γ_I	

Figure 3.2: Model setup of 2D test case.

3.5.1 2D test case

The first test case represents a generic 180° bend, see figure 3.2. Fluid enters the bend at the inlet Γ_{in} with a prescribed velocity $\mathbf{v}_{\text{in}}$ and it leaves the bend at the outlet Γ_{out} under ambient pressure p_{out}. In order to avoid negative influences from the inlet or the outlet on the flow in the bend, the fluid domain is extended far beyond the actual bend. Around the bend, the fluid is bounded by a structure, whereas the structure is clamped on one side. On the interface between both domains, Γ_I, the fluid exerts forces on the structure.

The fluid is considered to be viscous and incompressible. The corresponding characteristics and relevant boundary conditions are specified in figure 3.2. We consider the pressure loss within the bend as the specific flow characteristic (objective function J) to be optimized (minimized). We determine the pressure loss in the bend by subtracting the total surface pressure at the outlet from the total surface pressure at the inlet. Defining p_{tot} as the local total pressure value, it holds:

$$J = \frac{\int_{\Gamma_{\text{in}}} p_{\text{tot}} \; d\Gamma_{\text{in}}}{\int_{\Gamma_{\text{in}}} d\Gamma_{\text{in}}} - \frac{\int_{\Gamma_{\text{out}}} p_{\text{tot}} \; d\Gamma_{\text{out}}}{\int_{\Gamma_{\text{out}}} d\Gamma_{\text{out}}} \tag{3.62}$$

For the structure, we assume a linear elastic deformation with an isotropic material behavior. Parameters that characterize the structure are compiled in figure 3.2. The figure also specifies the support conditions (Dirichlet boundary conditions). The external load vector acting on the structure (the Neumann boundary condition) results from the coupling with the fluid along Γ_I. The emerging internal stresses are measured as von Mises equivalent stresses at the Gauss points, cf. equation (A.1) and (A.2).

According to the present optimization problem, stresses are constrained to a maximum value, σ_{max}. In this 2D test case, the actual stress at a defined evaluation point shall not increase compared to its initial value. So, σ_{max} corresponds to the initial stress at the selected evaluation point and $n_\sigma = 1$, meaning there is only one constraint. The evaluation point is identified as $g(\sigma)$ in figure 3.2. The actual stress value at this evaluation point is calculated as the mean stress of all Gauss points of the surrounding element.

Despite the 2D scenario, the structure is modeled as a slightly extruded 3D body, whereas motions in extrusion direction are locked ("plane strain"). By that, we avoid the application of a plate or shell formulation so that the solution approach in this test case remains comparable to the three-dimensional test case in the subsequent section. Figure 3.3 illustrates the extrusion. The coupling between fluid and structure is only established along the edges on one side of the structure.

The analyses of the two physical domains require dedicated meshes. Two variants of the test case are introduced in this context: One variant where the two meshes are matching on the interface. In the following, this variant is referred to as "TCA." Furthermore, another variant, where the meshes on the interface are non-matching, in the following called "TCB." The latter requires a mapping of data between the individual meshes. For this purpose, a nearest element interpolation is applied, cf. Wang et al. [63]. Figure 3.4 visualizes and compares the two cases around the symmetry

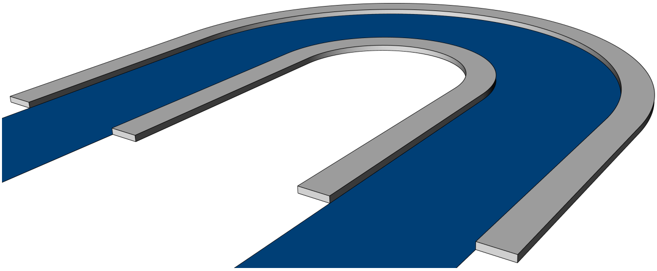


Figure 3.3: Extruded structure in 2D test case. Fluid and structure are colored in blue and grey, respectively.

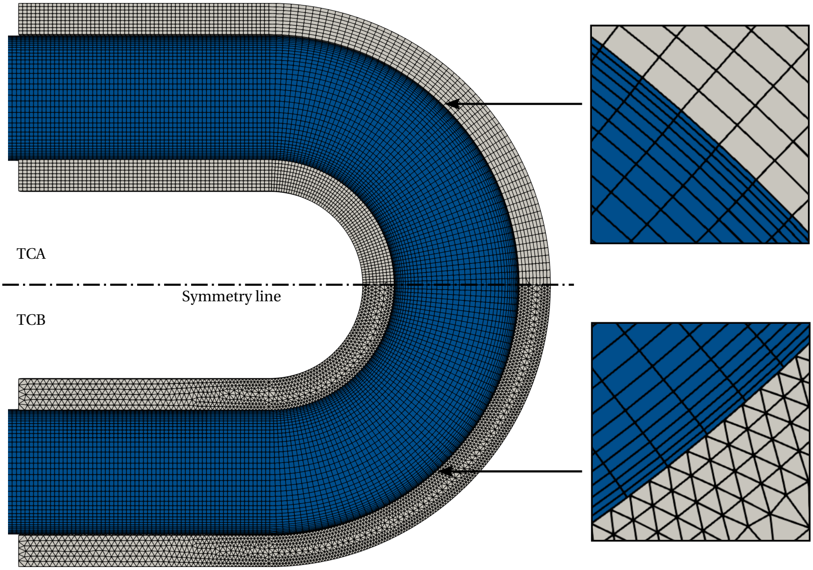
line. It also provides some specifications of the individual meshes.

The inner and outer boundary of the bend corresponds to the design surface. In the parameterization of the bend, we follow the extended Vertex Morphing described in section 3.3. That is, we introduce a control field to control the design surface in both domains and simultaneously the entire volume mesh of the structure. The latter also provides the discretization of the control field. Following this strategy, the number of design variables amounts to three times the number of nodes in the mesh of structure, i.e., 28 440 in TCA and 42 168 in TCB.

The filter radius in both cases is specified as $r = 0.03$m. Hence r is significantly larger than the diameter of the bend (0.02m). Also, it spans across a large portion of the design surface. By this choice, the two opposite walls of the bend shall get a sense of each other's sensitivities or motion. Also, the optimizer shall prefer global shape changes rather than local updates.

3.5.2 3D test case

The second test case corresponds to a generic valve housing with an internal channel that guides a hydraulic fluid. Figure 3.5 illustrates the part together with some characteristic dimensions. The figure also specifies material quantities and locates boundary conditions. Fluid enters the block at the inlet Γ_{in} with a prescribed velocity $\mathbf{v}_{\text{in}}$, and it leaves the block at the outlet Γ_{out} under ambient pressure p_{out}. To avoid negative influences from both inlet and outlet, the simulation domain of the fluid is extended far



TCA	
nodes in $\Omega_F = 17\,400$	nodes in $\Omega_S = 9\,480$
elements in $\Omega_F = 16\,926$	elements in $\Omega_S = 4\,248$
nodes on Γ_I in $\Omega_F = 474$	nodes on Γ_I in $\Omega_S = 474$

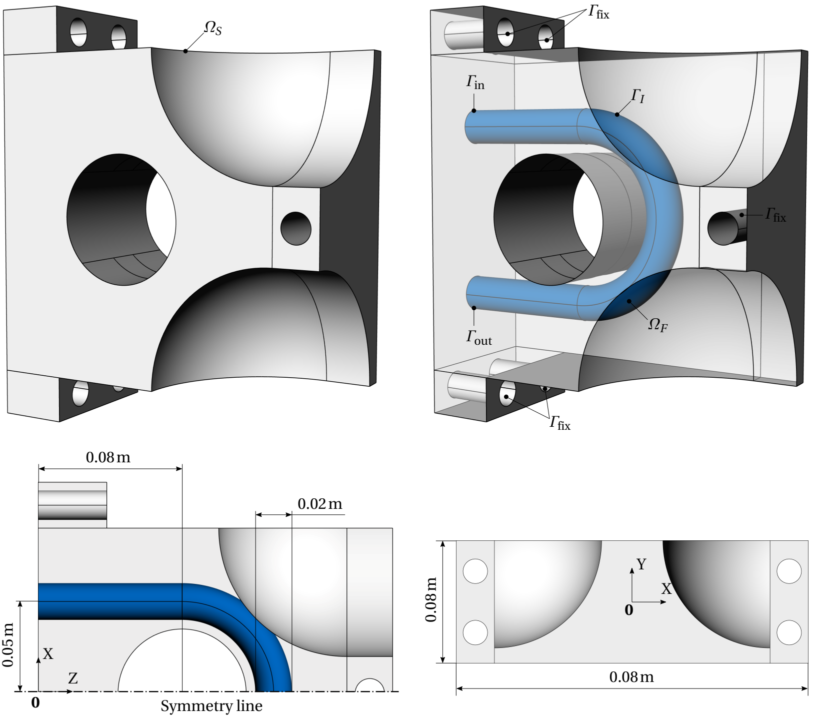
TCB	
nodes in $\Omega_F = 17\,400$	nodes in $\Omega_S = 14\,056$
elements in $\Omega_F = 16\,926$	elements in $\Omega_S = 48\,077$
nodes on Γ_I in $\Omega_F = 474$	nodes on Γ_I in $\Omega_S = 576$

Figure 3.4: Two variants of the 2D test case:1) with matching meshes (top, "TCA"), 2) with non-matching meshes (bottom, TCB).

beyond the structure (not visible in the figure). On the interface between both domains, Γ_I, the fluid exerts forces on the structure. The forces, in turn, cause internal stresses in the structure. The latter shall be controlled in the optimization.

Some central characteristics of the test case are:

- It has a complex three-dimensional geometry with regions that must not be modified, such as bolt holes or connection areas.
- It has a complex interior geometry with a narrow channel layout, i.e., geometric entities are close to each other, and thin walls form out



Ω_F

$\rho_F = 1765\,\text{kg/m}^3$	$\mu_F = 0.022945\,\text{Ns/m}^2$	$Re = 10000$
$\mathbf{v}_{\text{in}} = \{6.5, 0, 0\}\text{m/s}$	$p_{\text{out}} = 0\,\text{Pa}$	

Ω_S

$\rho_S = 7850\,\text{kg/m}^3$	$E = 1 \times 10^{11}\,\text{Pa}$	$\nu = 0.29$
$\mathbf{u}_{\text{fix}} = \{0, 0, 0\}\text{m}$	$\mathbf{f_S} = \mathbf{f_S}(\mathbf{f_F})\,[N]$ on Γ_I	

Figure 3.5: Model setup of 3D test case.

locally. Thin walls are particularly challenging in the optimization since there stresses may concentrate and easily exceed the threshold. Also, shape updates can quickly lead to overlapping surfaces.

- Stress peaks are not limited to the surface that is in contact with the fluid. They can also occur within the structure.

The fluid is considered to be viscous and incompressible. Characteristics of the fluid and relevant boundary conditions are specified in figure 3.5. As already in the 2D test case, we assume pressure loss to be the specific flow characteristic (objective function J) to be optimized (minimized). As in the 2D case, J is determined by subtracting the total surface pressure at the outlet from the total surface pressure at the inlet, cf. (3.62).

For the structure, we assume a linear elastic deformation with an isotropic material behavior. Parameters that characterize the structure are compiled in figure 3.5. The whole part is clamped at five built-in bolt connections, see $\mathit{\Gamma}_{\text{fix}}$ in the previous figure. The loading of the structure solely results from the coupling with the fluid along $\mathit{\Gamma}_I$. The arising internal stresses are measured as von Mises equivalent stresses at the Gauss points. The latter must not exceed the allowed maximum value, $\sigma_{\max}$. $\sigma_{\max}$ is defined as the maximal stress in the initial design. So, we require that the stress level in the optimal design does not increase compared to the original design.

To avoid the application of one stress constraint for each Gauss point, we average stress values within a single element, assuming a sufficiently fine discretization. Furthermore, a critical region is defined. Only stress constraints from elements inside the critical region are considered as relevant. The critical region comprises large parts of the structure around the internal bend. It is visualized in figure 3.6. Stress constraints within the critical region are aggregated using the Kreisselmeier–Steinhauser (KS) function, cf. Kreisselmeier et al. [71] and Wrenn [72]. The KS function produces a $C1$-continuous envelope around the relevant constraints and provides a conservative estimate of the included maximum value. Using the KS-function, the effective number of constraints reduces to one. The remaining constraint reads:

$$g(g_i(\mathbf{s})) = \mathrm{KS}(g_i(\mathbf{s})) < 0 \tag{3.63}$$

whereas:

$$\mathrm{KS}(g_i(\mathbf{s})) = \frac{1}{\rho} \ln \sum_{i=1}^{n_\sigma} e^{\rho g_i(\mathbf{s})/\sigma_{\max}} \tag{3.64}$$

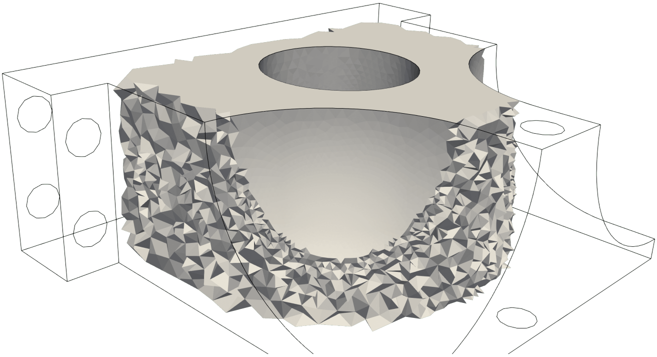
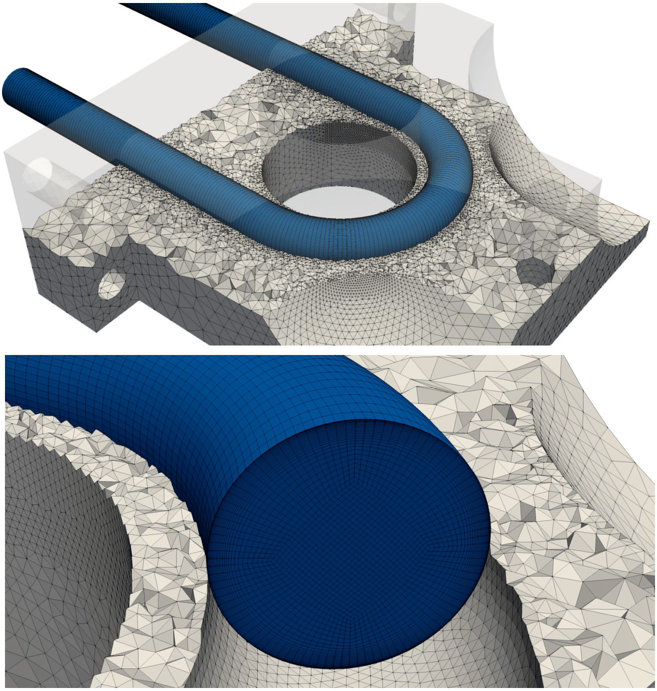


Figure 3.6: Elements of the critical region in the 3D test case.

In the previous equations, KS represents the estimated maximum constraint value (i.e., the value of the KS-function). g_i represents the local stress constraints from all elements within the critical domain. The aggregated constraints in the KS-function are scaled by the predefined threshold $\sigma_{\max}$ to avoid high exponent values. ρ is a scalar multiplier that controls the accuracy of the estimation. For an accurate estimation, ρ must be chosen as large as possible. Too large values, however, will cause numerical overflow. Given the scaled constraint values, the multiplier is chosen as $\rho = 50$, which is considered to be a reasonable choice, cf. Wrenn [72] and Martins et al. [73].

For the optimization, the gradient of the KS-function needs to be computed. Generally, the computation corresponds to the gradient computation with a single stress constraint, cf. section 3.4.5. The only difference in the present case is that one needs to consider additional contributions from the local averaging of the stresses and the nesting of constraint functions in (3.64). The additional contributions only affect the terms $\partial I/\partial \mathbf{u}$ and $\partial I/\partial \mathbf{x}_S$ and can be computed explicitly using the chain rule and simple arithmetic operations.

The meshes used to analyze the coupled problem are presented in figure 3.7. Note from the figure that both domains are discretized differently. Accordingly, the two surface grids along the interface do not match. Hence, mapping is required to transfer data between the domains. For this pur-



nodes in $\Omega_F = 1\,223\,245$	nodes in $\Omega_S = 155\,272$
elements in $\Omega_F = 1\,202\,688$ (Hex)	elements in $\Omega_S = 813\,142$ (Tet)
nodes on Γ_I in $\Omega_F = 19\,008$	nodes on Γ_I in $\Omega_S = 22\,189$

Figure 3.7: Discretization of fluid (blue) and structure domain (grey) in the 3D test case.

pose, the Mortar Mapping Method is applied, cf. Wang et al. [63].

The design surface, in this case, includes the common interface Γ_I and some faces around the structure, which are not in contact with the fluid, see figure 3.8. In the parameterization of the structure, we follow the extended Vertex Morphing described in section 3.3. That is, we introduce a control field to control the design surface in both domains and simultaneously the entire volume mesh of the structure. The latter also provides the discretization of the control field.

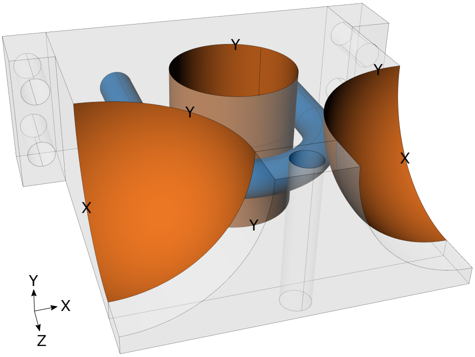


Figure 3.8: Design surface in the 3D test case including the bend (blue) and other surfaces of the structure (orange). Letters indicate damping in the specified coordinate direction.

Generally, the design surface is free to move in any direction. However, geometric constraints must be considered at the edges in order to maintain a valid connection to the surrounding non-design domain. In this context, we consider two types of edges: 1) edges that are located on the planar outside of the structure and must only be modified in the corresponding in-plane direction, and 2) edges that must not be modified at all. In figure 3.8, edges of the first type are tagged with each the prohibited coordinate direction. All other edges are of the second type.

We integrate the geometric constraints directly into the parameterization by damping shape updates around the edges. The damping follows the approach explained in section 2.5. At fixed edges, the movement of nodes is damped in all coordinate directions. At edges on the planar outside of the structure, damping is applied in only the prohibited coordinate direction. I.e., nodes may still move freely in the in-plane direction. At each edge, we use a cosine damping function with an individual damping radius in the order of the later specified filter radius, r.

As indicated before, Vertex Morphing takes control over the entire structure mesh. Given the $\mathbf{x}_S = 155\,272$ nodes, this approach gives rise to $3 \times 155\,272 = 465\,816$ design variables. By controlling the entire structure mesh, we avoid an additional simulation of the corresponding mesh motion. However, in this approach, Vertex Morphing will also cause mesh updates on the

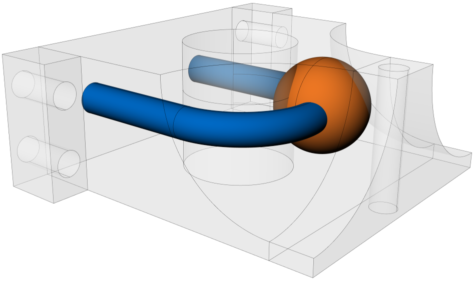


Figure 3.9: Filter radius illustrated as sphere with radius r (orange).

boundaries beyond the actual design surface. Given that those boundaries must not change its shape, the mesh update has to be restricted there. For this purpose, we apply damping. More precisely, at all planar faces that are not part of the design surface, we damp the mesh motion in the corresponding out-of-plane direction using a cosine-damping function with an individual damping radius in the order of r.

The filter radius in this case is specified as $r = 0.03$m. Hence it is larger than the diameter of the channel in the interior (0.02m), larger than the thickness of the existing thin walls, and it spans over a big portion of the entire structure, cf. figure 3.9. This choice serves several purposes:

- It allows that opposing points on the channel get a sense of each other's sensitivity values. Hence, we avoid local updates that have a negative influence on the other side of the channel.
- It causes shape updates to be low-frequent so that we can expect a globally smooth design.
- Since the filter radius is significantly larger than the smallest wall thickness of the structure, it ensures that thin walls move as a whole. That is, the large filter acts as an implicit thickness constraint. Such a constraint is especially required around the big center hole, where only a thin wall separates the fluid from the outside of the structure, cf. figure 3.5.

The possibility to establish an implicit thickness constraint through the choice of r turns out to be very useful in practical applications because the interior design of structures with internal flow often consists of many cut-outs in a complex arrangement with only a little distance in-between.

3.6 Software framework

The multidisciplinary optimization process is realized by combining and extending the capabilities of two open-source software packages: "Kratos Multiphysics", Kratos Developer Community [70] and Dadvand et al. [74], and "SU2", SU2 Developer Community [68] and Economon et al. [75].

Kratos is used to drive the optimization process, to control the FSI analysis, and to organize the interdisciplinary communication. The latter comprises the data exchange and the mapping of quantities between the different meshes. As part of the optimization process, Kratos also includes the Vertex Morphing Method. Furthermore, Kratos is used to simulate the structure and to perform the corresponding sensitivity analysis.

SU2 is used to run all fluid simulations as well as the corresponding sensitivity analyses. The fluid simulations employ the integrated solver for incompressible fluid flows, refer to Economon [76] for details. The sensitivity analyses are based on the built-in discrete adjoint solver, refer to Albring et al. [66] for details. Also, the built-in capabilities of SU2 are utilized to morph the volume mesh in the fluid domain after every design update.

Both software packages have a strong focus on multiphysics. However, the underlying numerical methods are different. SU2 mostly relies on the finite volume method with unstructured grids, whereas Kratos follows a general finite element approach.

3.7 Verification of the sensitivity analysis

In this section, the sensitivity analysis of the developed optimization process is verified. The verification is performed for the 2D and 3D test case and includes an investigation of the (single-disciplinary) objective and the (coupled) constraint sensitivities. The verification is based on a comparison of the computed analytic sensitives to relevant reference values at selected evaluation points. The verification is performed in the *con-*

trol space, so the complete sensitivity vector, as used by the optimizer, is evaluated. For the sensitivity verification, any damping is neglected.

In both test cases, the reference values are determined by a finite difference (FD) approach, which computes first-order derivatives:

$$\frac{dI}{d\mathbf{s}} \approx \frac{I(\mathbf{s}+\Delta\mathbf{s})-I(\mathbf{s})}{\Delta\mathbf{s}} = \frac{dI}{d\mathbf{s}}^{\mathrm{FD}} \tag{3.65}$$

In the FD, we vary the perturbation, $\Delta\mathbf{s}$, by several orders starting from a large value, where approximation errors have to be expected and ending at a small value, where round-off errors are dominant. Among the set of determined values, we check for convergence, and if we find a plateau of constant values, we choose the one value as reference, which is closest to the value to be verified.

The reference values are then used to asses the quality of the computed sensitivities, both in terms of magnitude and direction. Therefore, we employ the following quality indicators:

$$\text{total deviation} = \left\| \frac{dI}{d\mathbf{s}} - \frac{dI}{d\mathbf{s}}^{\mathrm{FD}} \right\| \left\| \frac{dI}{d\mathbf{s}}^{\mathrm{FD}} \right\|^{-1} \tag{3.66}$$

$$\text{local deviation} = \left\| \frac{dI}{d\mathbf{s}_i} - \frac{dI}{d\mathbf{s}_i}^{\mathrm{FD}} \right\| \left\| \frac{dI}{d\mathbf{s}_i}^{\mathrm{FD}} \right\|^{-1} \tag{3.67}$$

The first quality indicator characterizes the total deviation of the analytic to the reference gradient using the information from *all* evaluation points. The deviation is expressed in percent relative to the reference value. The second quality indicator evaluates the same deviation but *locally* at a specific evaluation point i based on the local gradient information $dI/d\mathbf{s}_i$ and $dI/d\mathbf{s}_i^{\mathrm{FD}}$.

3.7.1 2D case with matching and non-matching meshes

We first verify the proposed sensitivity analysis in 2D. For this purpose, the 2D test case from section 3.5.1 is utilized. We consider both variants of this test case, i.e., TCA with the matching mesh and TCB with the non-matching

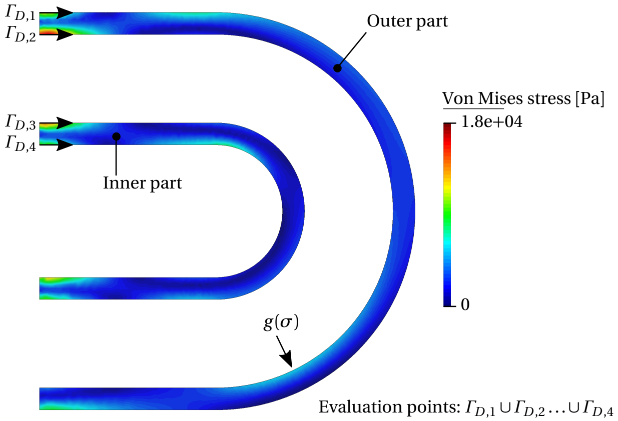
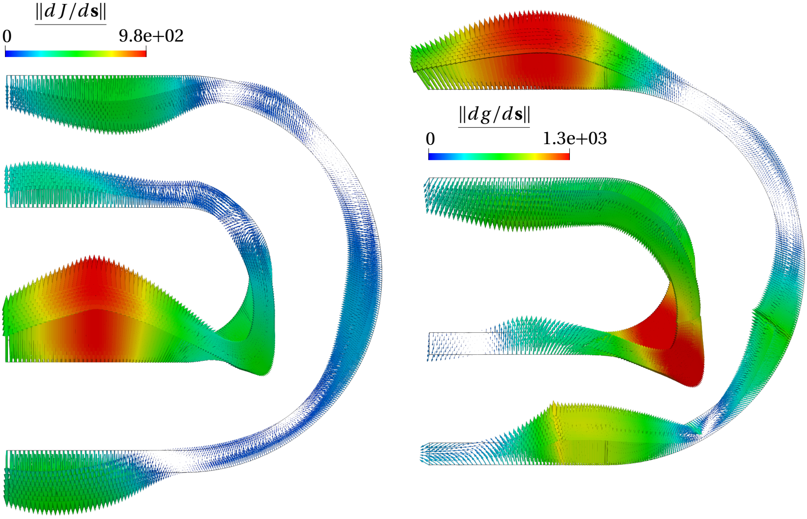


Figure 3.10: Boundaries in 2D test case at which sensitivities are verified.

mesh. In both variants, we compute reference sensitivities using FD at all nodes on the design surface Γ_D at $z = 0$. We order the corresponding set of evaluation points into four subsets corresponding to the four different boundaries of the design surface, cf. figure 3.10. The arrows in the figure indicate the order of evaluation points within a subset. The figure also shows the initial stress state of the structure and the point at which the stress constraint is evaluated, $g(\sigma)$.

The sensitivity verification is performed for both the objective and the constraint function. Figure 3.11 shows the analytic sensitivities of both response functions, computed by the herein presented sensitivity analysis. Figure 3.11(a) presents the results of the single-disciplinary sensitivity analysis of the objective (pressure loss) and figure 3.11(b) the results of the coupled sensitivity analysis of the constraint (stress value). From the figures, one can see that both sensitivities happen to have similar magnitudes, but the direction is conflicting. So, an intuitive design suggestion seems impossible. Instead, numerical optimization is required to find a good compromise.

Note that the presented sensitivities are sensitivities in the control space and *not* in the geometry space. So, the originally very discontinuous mesh sensitivities of the form $d \bullet / \mathbf{x}$ are already filtered once using Vertex Mor-



(a) Objective function (b) Constraint function

Figure 3.11: Analytic design sensitivities in 2D test case with non-matching meshes (TCB).

phing and turned into design sensitivities of the form $d\bullet/d\mathbf{s}$. Therefore, we can observe a smooth course of the sensitivities.

From a physical point of view, it is interesting to observe that the maximum constraint sensitivities are not located directly around the stress evaluation point, as could be expected, but on the opposite side of the channel. There are also other significant hotspots at a considerable distance to the actual evaluation point. Both observations indicate that the contribution from the fluid, and hence the coupling, plays an essential role in the sensitivity analysis of the structure.

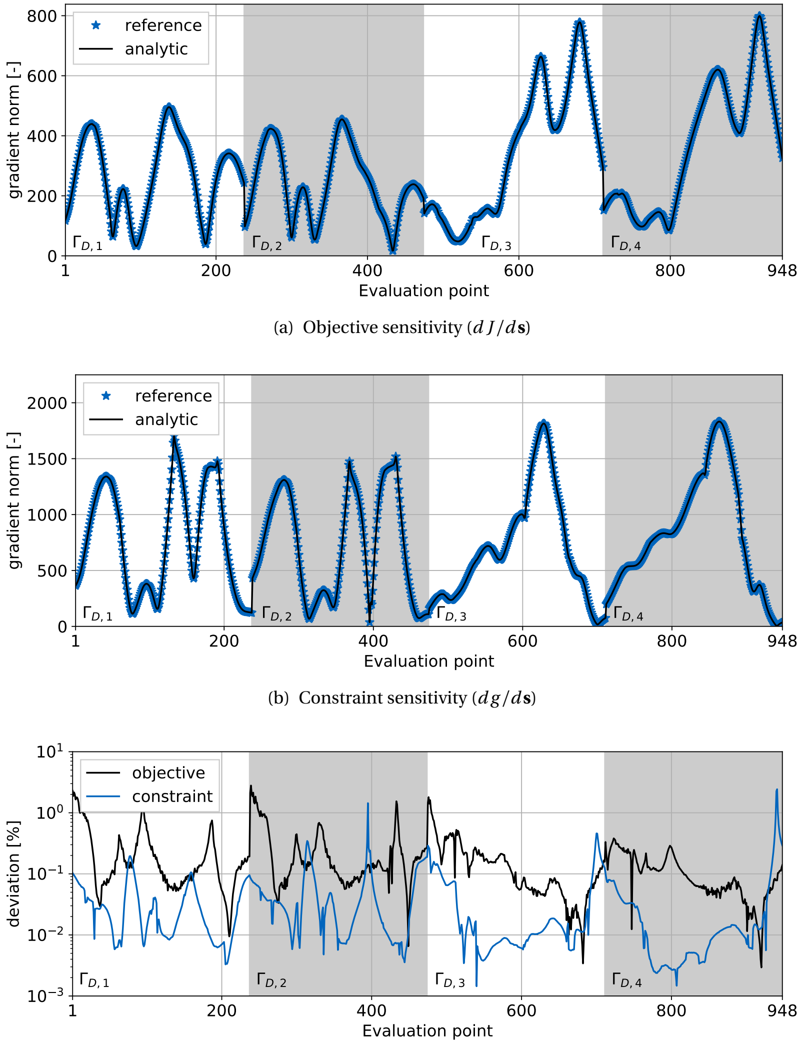
In the following, we first verify the sensitivities using TCA. The results of this sensitivity verification are compiled in figure 3.12. The figure includes three graphs. The abscissa in every graph collects all evaluation points in the order specified in figure 3.10. The first two graphs show the magnitude of the local analytic and the local reference sensitivities in case of the objective and the constraint function. We compute the magnitudes at every evaluation point i using the local norm $\|dI/d\mathbf{s}_i\|$. The third graph shows the local deviation of the analytic sensitivities from the reference values, which also includes directional deviations. The local deviations are calculated according to (3.67) at each evaluation point i.

In figures 3.12(a) and 3.12(b), we can see that the analytic sensitivities are well matching the reference values at all evaluation points - both in case of the objective and the constraint function. No significant outliers are observable, and differences in magnitude only become visible on a small scale.

Also, the computed local deviations show a good agreement between analytic and reference sensitivities, see figure 3.12(c). The deviations are mostly in the order of around $0.01-0.1\%$ and sometimes even lower. At a few locations, we observe peak deviations of around 2%. However, such peaks only appear at spots where the sensitivity value is close to zero, so that the division in (3.67) leads to relatively large percentages. Looking instead at locations with a large sensitivity value, one finds that whenever the sensitivity magnitude reaches a peak value, the deviation drops to a local minimum in the order of $0.001-0.01\%$. This behavior indicates an overall valid analytic sensitivity analysis.

The remaining deviations between reference and analytic sensitivities in figure 3.12(c) can be traced back to errors in the simple finite difference approximation. Despite trying a broad range of perturbation sizes, no constant plateau could be observed in the reference values. So, the reference values themselves deviate from the correct sensitivity values to some extend. The missing plateau in the reference values is probably a consequence of the strong nonlinearity of the governing equations of the fluid, which we analyze here by a simple first-order finite difference approach.

Having evaluated the sensitives for the test case with matching meshes, we conduct the same study for the test case with non-matching meshes (TCB). This test case includes a mapping operation between the different discretizations along Γ_I. The mapping operation is only relevant in the sensitivity analysis of the constraint but not the objective. The results of the study are compiled in figure 3.13. The figure again includes three graphs. The first two graphs compare the analytic with the reference sensitivities regarding their magnitude. The third graph presents the complete local deviations, calculated at each evaluation point i according to (3.67).

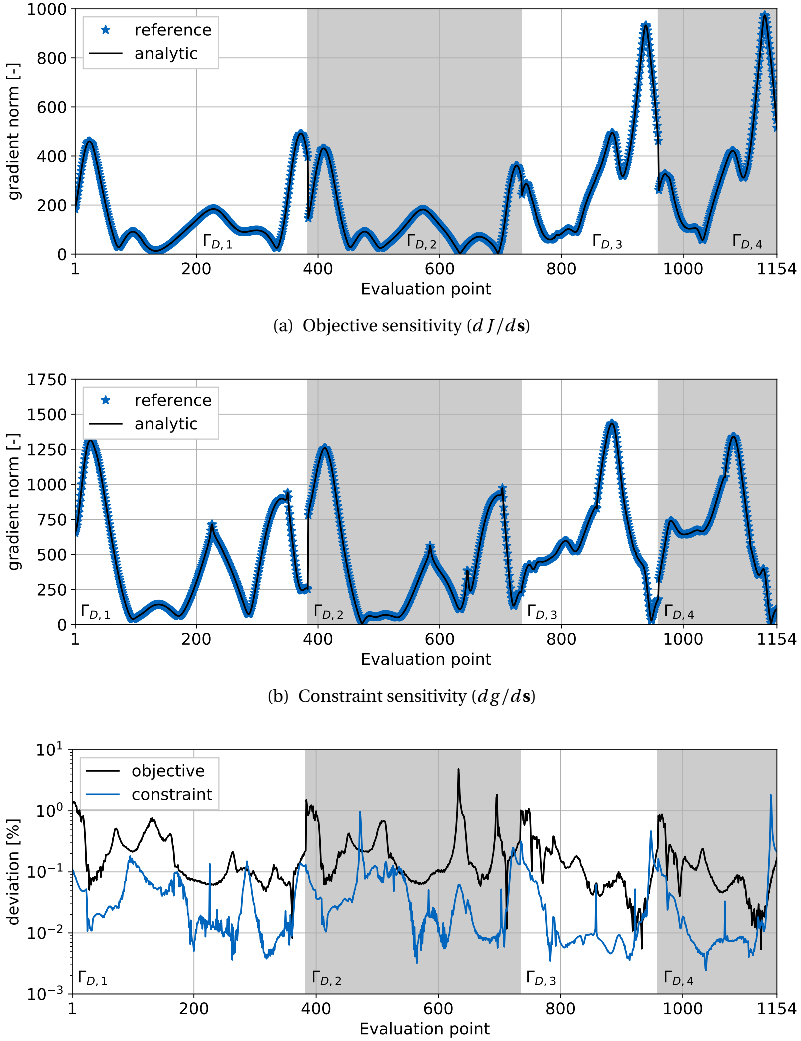


(a) Objective sensitivity ($dJ/d\mathbf{s}$)

(b) Constraint sensitivity ($dg/d\mathbf{s}$)

(c) Local deviation to reference according (3.67)

Figure 3.12: Sensitivity verification using TCA.



(a) Objective sensitivity ($dJ/d\mathbf{s}$)

(b) Constraint sensitivity ($dg/d\mathbf{s}$)

(c) Local deviation to reference according (3.67)

Figure 3.13: Sensitivity verification using TCB.

Table 3.1: Total deviation of the analytic gradient from its reference in the 2D test case according to (3.66).

	Objective	Constraint
matching meshes (TCA)	0.20%	0.02%
non-matching meshes (TCB)	0.24%	0.03%

The results in the figure essentially correspond to the results of TCA in figure 3.12. That is, the analytic sensitivities locally correspond to their reference quantities, both in terms of magnitude and direction. Especially when comparing the deviations in figure 3.12(c) with those of figure 3.13(c), no considerable differences can be observed. As far as the constraint is concerned, this observation is remarkable as we neglected the mapping contribution in the computation of the coupled gradient, cf. (3.61). So, it seems that in this case, the mapping does not have a significant contribution to the coupled sensitivity analysis. This observation fits the general expectation that mapping has only a negligible influence on the sensitivity analysis in case of properly refined mehes, cf. section 3.4.6.

As in TCA, the remaining visible deviations in figure3.13(c) can be traced back to errors in the simple finite difference approximation. Despite trying a broad range of perturbation sizes, no constant plateau could be observed for the individual reference values. So, the reference values themselves deviate from the correct sensitivity values to some extend. The general agreement of values in the presented results still suggests a valid analytic sensitivity analysis.

The impression of a valid sensitivity analysis can also be quantitatively confirmed. For this purpose, we evaluate the total deviation between the complete analytic gradient vector, $dI/d\mathbf{s}$, and the corresponding reference vector, $dI/d\mathbf{s}^{\mathrm{FD}}$, according to (3.66). Table 3.1 lists the results for TCA and TCB. From the table, we can see that the differences are generally low.

Note in the table that TCB shows a slightly larger deviation for both response functions compared to TCA. Since only the computation of the constraint gradient involves a mapping operation, we may conclude that the increased deviation in TCB is not caused by the neglected mapping term in the coupled sensitivity analysis of the constraint, cf. (3.61). Instead, it seems to be simply a consequence of the different meshes. Generally, the results from TCA and TCB suggest that the mentioned mapping term may be safely neglected in case of a sufficiently fine discretization.

From the results so far, we can conclude that:

- The sensitivities, especially those calculated by the proposed coupled sensitivity analysis, are well conforming to the reference values. So we consider them to be verified.
- Neglecting the mapping term in (3.61) seems to be a possible remedy in case the term can not be computed for any reason.

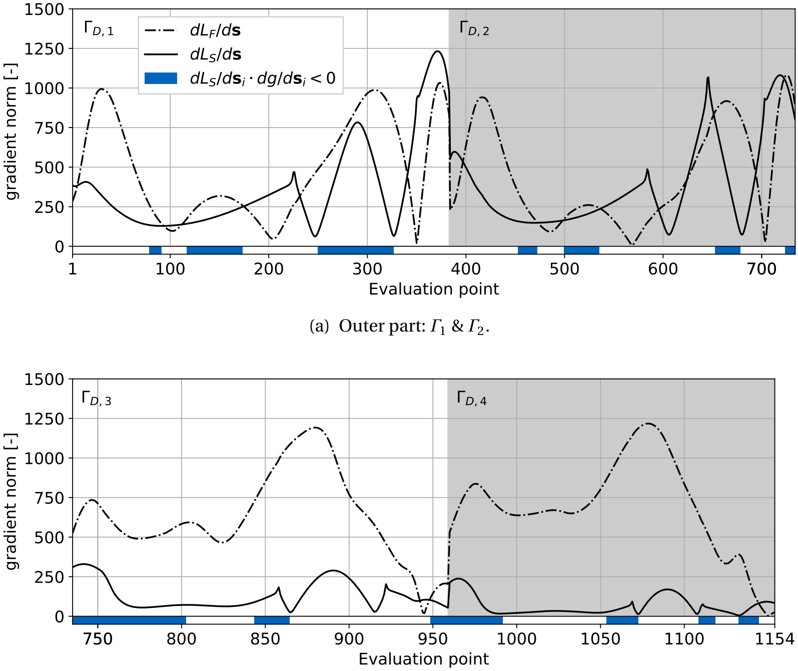
Influence of the coupling

In the following, we investigate the influence of the coupling on the sensitivity analysis of the stress constraint. In doing so, we only utilize TCB. However, the findings are equally valid for TCA.

To investigate the influence of the coupling, we distinguish the two major ingredients of the constraint gradient, i.e., the contribution from the adjoint sensitivity analysis of the structure and the contribution of the adjoint sensitivity analysis of the fluid, cf. (3.61). The complete constraint gradient was denoted $dL/d\mathbf{s} = dg/d\mathbf{s}$ and the two ingredients $dL_S/d\mathbf{s}$ and $dL_F/d\mathbf{s}$, respectively. Note that the fluid contribution is a direct consequence of the coupling. So, its impact is a good measure for the relevance of the coupling.

Figure 3.14 plots the two ingredients for all evaluation points and quantifies their contribution. In the figure, the evaluation points are grouped into two graphs: one covering the outer structure, where we evaluate the stress constraint, and one covering the inner structure, which has no physical link to the outer part. Figure 3.15 visualizes the two ingredients over the entire structure and provides a qualitative impression of their differences.

In the graphs, the first striking observation is that both the structure *and* the fluid have a significant contribution to the overall gradient of the stress constraint. Concentrating on the outer part of the structure, for example, we find that both contributions are balanced. At the inner part of the structure, which has no link to the outer part where we evaluate the stress constraint, the fluid contribution even dominates the overall gradient.

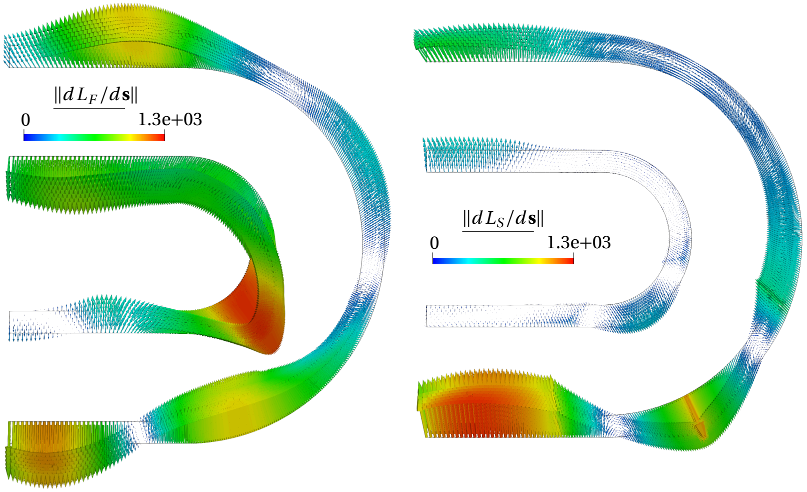


(a) Outer part: Γ_1 & Γ_2.

(b) Inner part: Γ_3 & Γ_4.

Figure 3.14: Ingredients of $dg/d\mathbf{s}$ in TCB: $dL_F/d\mathbf{s}$ represents the contribution from the adjoint fluid analysis and $dL_S/d\mathbf{s}$ from the adjoint structure analysis. $dL_S/d\mathbf{s}_i \cdot dg/d\mathbf{s}_i$ shows whether at evaluation point i both contributions point in the same direction.

Second, one can see that the contributions of the fluid and the structure have different peak values at different locations. Peak values in the contribution of the structure can only be observed in the outer part of the structure around the stress evaluation point or its direct vicinity (see evaluation point 290, 370, 645, and 716 or the relevant locations in figure 3.15(b)). In contrast to that, the fluid contribution peaks at various locations around the bend, even at locations that are remote to the actual stress evaluation point (see, e.g., evaluation point 30 in the graph or the corresponding location in figure 3.15(a)).



(a) Contribution of the fluid.

(b) Contribution of the structure.

Figure 3.15: Visualization of the ingredients of $dg/d\mathbf{s}$ in TCB.

Observing different hot-spots for the two contributions is reasonable as they track different physical phenomena: The sensitivity contribution of the structure identifies areas, which are either directly affecting the local stress value or which are indirectly influencing the overall *load-carrying capacity*. In figure 3.15(b), for example, one observes large sensitivity values around the supports and at the stress evaluation point. Those large values indicate possible reinforcements towards a reduced stress value. In doing so, however, the structural contribution neglects the variation of the fluid forces.

The variation of the fluid forces is tracked by the second sensitivity contribution, i.e., the contribution of the fluid. The latter originates from the sensitivity analysis of the auxiliary force functional. This contribution identifies areas, which are relevant for the coupling forces and so the *loading* of the structure. In this test case, there are many areas with a substantial contribution of the fluid to the coupled sensitivities (see figures 3.14 and 3.15(a)). So, there are many areas where the fluid influences the stress value in the specified evaluation point through the force coupling

A dominant sensitivity contribution of the fluid means that design modifications, which change the structure's loading through the fluid, influence the considered stress value more than design modifications, which change the structure's load-carrying capacity. Vice versa, a dominant structure contribution means that a direct reinforcement of the structure influences the considered stress value more than a modification in the fluid guidance.

In general, we find that both ingredients are crucial in the sensitivity analysis of the stress constraint. So, neglecting the contribution of the fluid[8], which corresponds to neglecting the coupling, would lead to a severe lack of sensitivity information.

This lack of sensitivity information can become critical. In the present test case, for example, the fluid contribution locally changes the direction of the coupled sensitivity vector ($dg/d\mathbf{s}$). In such a case, neglecting the fluid contribution would lead to entirely wrong sensitivity information. Figure 3.14 highlights evaluation points that are critical in this context (blue area). A point i is considered "critical" if locally $dg/d\mathbf{s}$ and $dL_S/d\mathbf{s}$ point in opposite direction, which means that the fluid contribution locally leads to a change in sign. The figure shows that the contribution of the fluid is decisive at many evaluation points. This observation again emphasizes the relevance of the coupling in the sensitivity analysis of the stress constraint.

From the previous results, we conclude:

- The coupling has a significant influence on the sensitivities of the stress constraint.
- Considering the coupling in the sensitivity analysis of the stress constraint is crucial. Neglecting the coupling would render the sensitivity information useless.

3.7.2 3D case with non-matching meshes

In the following, we verify the proposed sensitivity analysis in 3D. For this purpose, the 3D test case from section 3.5.2 is utilized. The sensitivities

[8] Neglecting the contribution of the fluid corresponds to the assumption of constant coupling forces. That means a CFD analysis still determines the loading of the structure. However, its variation through a modification of the fluid boundary is neglected in the sensitivity analysis.

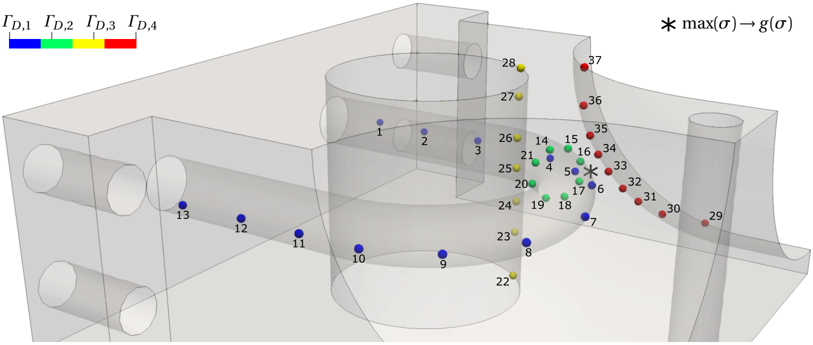
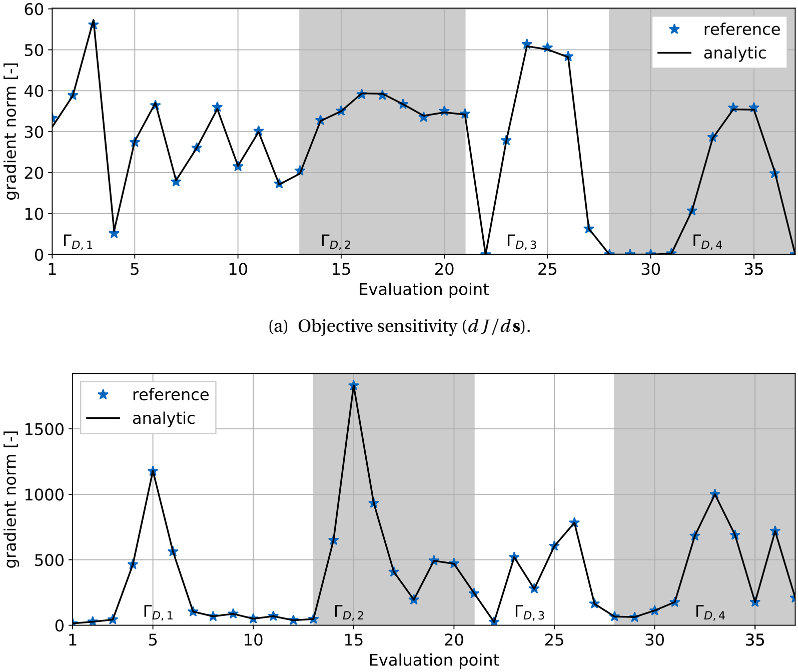

$\Gamma_{D,1}$ $\Gamma_{D,2}$ $\Gamma_{D,3}$ $\Gamma_{D,4}$
🟦 🟩 🟨 🟥
$\ast$ max(σ) → $g(\sigma)$
28
37
27
36
1
2
3
26
14
15
35
21
16
34
4
25
5
33
20
6
17
32
24
19
18
31
30
29
13
7
12
11
23
8
10
9
22

Figure 3.16: Points in the 3D test case where sensitives are verified.

in the 3D test case are verified at selected evaluation points on the design surface Γ_D. Due to the computational costs of the primal solution, the number of evaluation points is limited. So, a representative set must be chosen. The set of chosen evaluation points is visualized in figure 3.16. We order the complete set into four different subsets, $\Gamma_{D,1...4}$. The subsets represent different parts of the structure, i.e., the interface in contact with the fluid, the pipe circumference at a specific location, and two paths along on the outer boundary with no contact to the fluid. The evaluation points concentrate on the area around the maximal stress value in the initial design. Figure 3.16 highlights the location of the maximum stress value. For the sensitivity verification, we neglect the stress aggregation in this 3D test case and only consider the stress value at the specified location in the formulation of the constraint.

The results of the verification are presented in figure 3.17 and table 3.2. The figure compares the analytic sensitivity values with their reference values for both the objective and the constraint function. Therefore, the local norm $\|dI/d\mathbf{s}_i\|$ is calculated at each evaluation point i. The table presents the total deviation of the analytic sensitivity vector from its reference vector calculated according to (3.66).

In the figure, we can see that the sensitivity values are generally conforming to their reference values both in case of the objective and the constraint function. Note that the distribution is well captured, only small local



(a) Objective sensitivity ($dJ/d\mathbf{s}$).

(b) Constraint sensitivity ($dg/d\mathbf{s}$).

Figure 3.17: Sensitivity verification in the 3D test case.

Table 3.2: Total deviation of the analytic gradient from its reference in the 3D test case according to (3.66)

	Objective	Constraint
non-matching mesh	2.35%	1.20%

changes in the magnitude are visible. One also finds that the constraint sensitivity peaks at evaluation point 5 and 15. Those peaks are reasonable as the corresponding points lay in the direct vicinity of the maximum stress value. By contrast, the objective function shows a balanced level of sensitivity values at all evaluation points. Only at a few locations, the sensitivity vanishes identically.

Note in this context, that the applied Vertex Morphing links the discrete fluid boundary with the mesh of the structure within the filter radius r. That is, evaluation points that lay within one filter radius around the fluid boundary will influence the objective function and show a sensitivity value larger than zero. Conversely, evaluation points that are further away will not influence the objective function, and thus show a sensitivity value of exactly zero. The points with the number 22 and 28 are such remote evaluation points, for example. They represent the boundary points of $\Gamma_{D,3}$, cf. figure 3.16.

Note in figure 3.17 that the analytic values are generally matching the reference values, yet some small deviations are visible. The calculation of the total deviation reveals errors in the range of $1\ldots2\%$, see table 3.2. Similar to the 2D test case, those errors can be traced back to errors in the simple finite difference approximation. Despite trying a broad range of perturbation sizes, no constant plateau could be observed for the individual reference values. So, the reference values themselves deviate from the correct sensitivity values to some extend.

Nevertheless, note that the accuracy of both gradients is comparable. Given that we computed the objective gradient entirely by a verified framework based on algorithmic differentiation (SU2), from which we can expect high numerical accuracy, we may also assume a high accuracy in the constraint gradient. The latter suggests a proper coupled sensitivity analysis.

Since it was not possible to find highly accurate reference values, it was also not possible to quantify the error arising from the neglected mapping term in the computation of the constraint gradient, cf. (3.61). All we can conclude from the presented results is that the error induced by neglecting the mapping term must be lower than the 1.20% presented in table 3.2. However, this value is low enough to consider the mapping term as uncritical.

Given that the total deviations are not negligible but small, and the remaining error is inevitable due to the remaining inaccuracies in the reference values, we consider the sensitivity analysis as verified in 3D. Worthwhile to mention is that the optimization results later do not show any negative impact of the remaining deviations in the sensitivity information.

In the following, we also investigate the influence of the coupling in the sensitivity analysis of the constraint. To this end, the two relevant ingre-

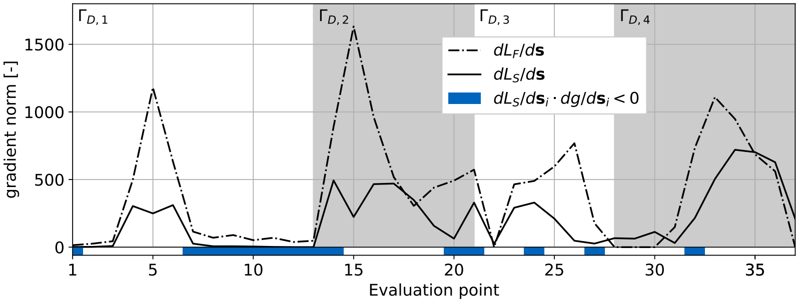


Figure 3.18: Ingredients of $dg/d\mathbf{s}$ in 3D test case: $dL_F/d\mathbf{s}$ represents the contribution from the adjoint fluid analysis and $dL_S/d\mathbf{s}$ from the adjoint structure analysis. $dL_S/d\mathbf{s}_i \cdot dg/d\mathbf{s}_i$ shows whether at evaluation point i both contributions point in the same direction.

dients of the constraint gradient are plotted in figure 3.18. Looking at the figures, one can again observe significant contributions from both domains. We especially see that the fluid contributions dominate over almost the entire region. So, in this 3D test case, design modifications, which cause a variation of the coupling forces, have a stronger influence on the stress constraint than design modifications that alter the structure's load-carrying capacity. Consequently, a plane reinforcement of the structure is less effective than an improvement in the flow guidance.

From the results in figure 3.18, we find that $dL_F/d\mathbf{s}$ makes up a significant part of the constraint sensitivity. Knowing that $dL_F/d\mathbf{s}$ is a direct consequence of the coupling, it is clear that neglecting the coupling can lead to severe errors in the constraint sensitivity. Figure 3.18 highlights evaluation points that are critical in this context (blue area). A point i is considered "critical" if locally $dg/d\mathbf{s}$ and $dL_S/d\mathbf{s}$ point in opposite direction, which means that the fluid contribution locally leads to a change in sign. As one can see in the figure, the small set of evaluation points already contains many points where such a change in sign can be observed. Consequently, the coupling in the sensitivity analysis of the stress constraint is crucial in this test case.

Influence of the parameterization

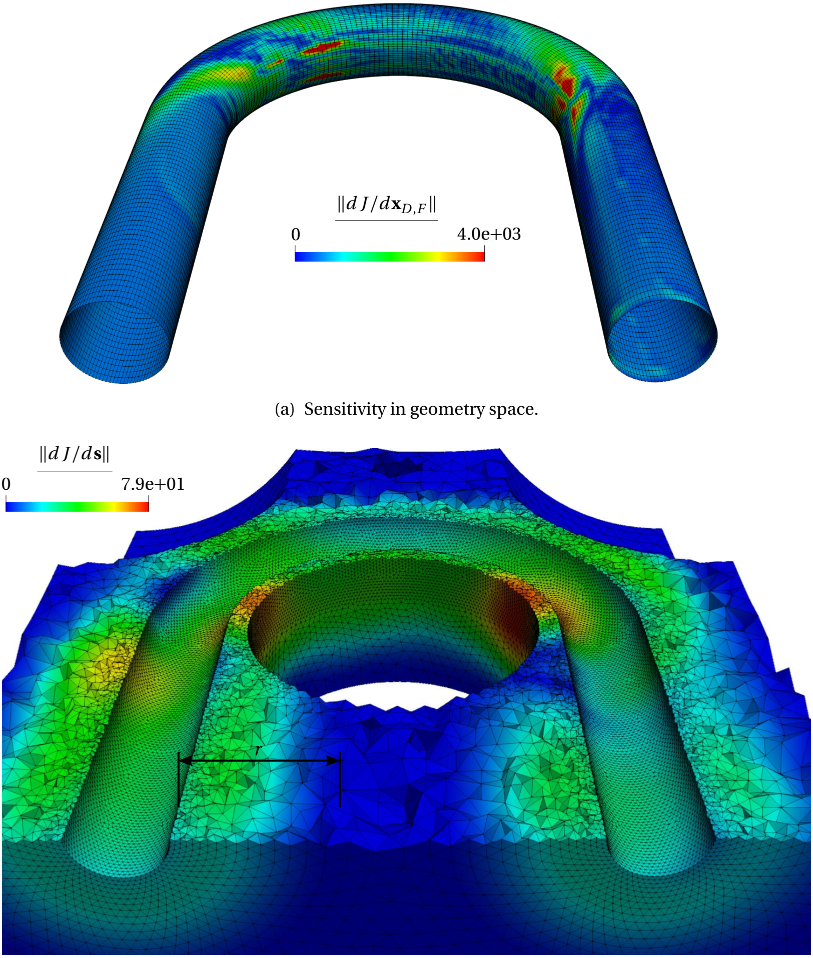
Consider the figures 3.19 and 3.20. The figures present each for the objective and the constraint function the filtered sensitivity in the control space, and the underlying mesh sensitivity in the geometry space. The mapping between design and geometry space follows the extended Vertex Morphing specified in (3.15). The figures show the following:

1) In both figures, one can observe a key feature of the underlying Vertex Morphing Method, i.e., the raw and discontinuous mesh sensitivities are filtered (regularized) such that they become smooth sensitivities in the control space. The filter radius (r) is clearly visible in this context. Note, for example, how the irregular distribution of the objective sensitivity in figure 3.19(a) is transformed to a smooth sensitivity information in the control space, see figure 3.19(b). According to our choice of design variables, the smoothing affects all nodes of the structure mesh within the filter radius. So, the design sensitivity includes not just information about the surface nodes on the boundary of the channel, but also about the adjacent volume nodes further inside the structure.

The smoothing is also visible in case of the constraint. Note, in this context, the peak values of the two sensitivity contributions, see figures 3.20(a) and 3.20(b). The peak values in each figure are due to the local nature of the stress constraint and indicate the point of maximal stress in the initial design. After filtering, a significantly smoother sensitivity information is obtained, see figure 3.20(c).

2) A key feature of the proposed variant of Vertex Morphing is, that it allows combining discontinuous mesh sensitivities from differently discretized domains to smooth sensitivity information in the common control space. That is, it already includes the necessary mapping of sensitivities between the individual domains. Consider, for example, figures 3.19(a) and 3.19(b). Note that the mesh sensitivities in case of the objective function are not just smoothed, but also mapped from the surface discretization of the fluid to the volume discretization of the structure, which represents the common control space. Both discretizations are not matching.

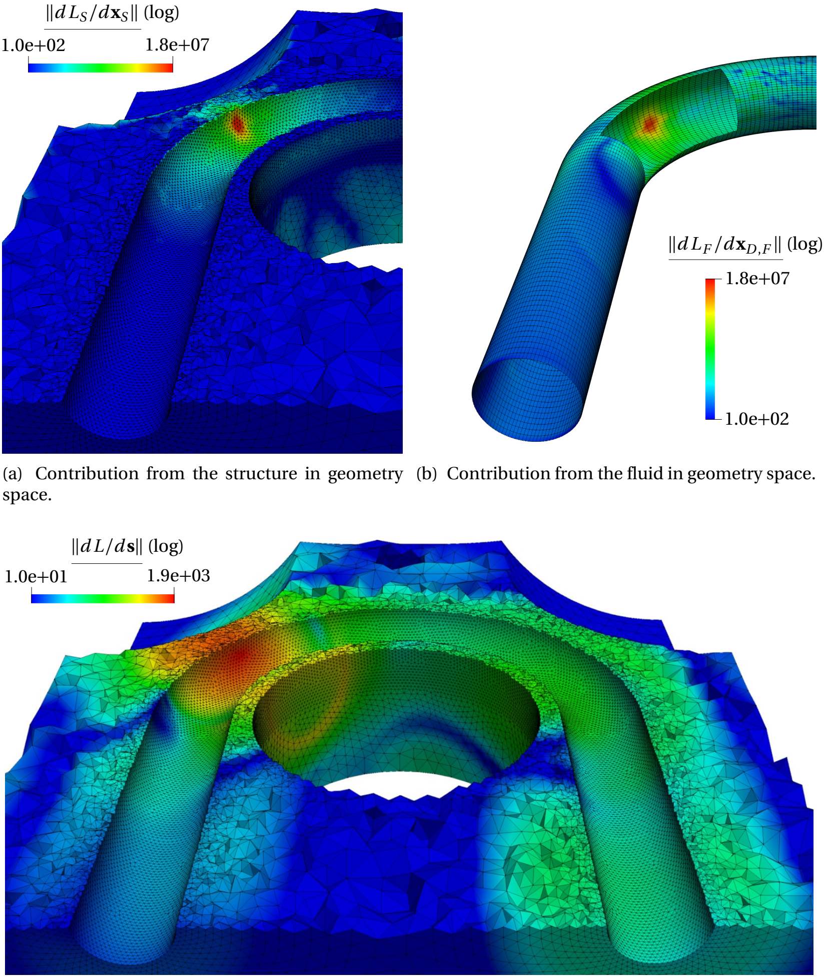
Or similarly in case of the constraint function, the two ingredients of the constraint sensitivity are originally given for different non-matching meshes, i.e. the fluid and the structure mesh, see figures 3.20(a) and 3.20(b). Due to extended Vertex Morphing, those contributions are automatically



(a) Sensitivity in geometry space.

(b) Sensitivity in control space.

Figure 3.19: Sensitivity of the objective function in geometry and control space, cf. (3.51). The structure is clipped for a view inside.


$\|dL_S/d\mathbf{x}_S\|$ (log)
1.0e+02
1.8e+07
$\|dL_F/d\mathbf{x}_{D,F}\|$ (log)
1.8e+07
1.0e+02
$\|dL/d\mathbf{s}\|$ (log)
1.0e+01
1.9e+03

(a) Contribution from the structure in geometry space.

(b) Contribution from the fluid in geometry space.

(c) Complete sensitivity in control space

Figure 3.20: Sensitivity of the constraint function in control space and its two ingredients in geometry space, cf. (3.58b), (3.59b) and (3.60). The structure and the fluid is clipped for a view inside.

mapped from the individual simulation meshes on the volume mesh of the control field in the common control space. In the control space, then, they are combined to a complete (and smooth!) coupled sensitivity information, see figure 3.20(c).

The previous observations show that the proposed variant of Vertex Morphing takes over two tasks in the sensitivity analysis - two tasks that otherwise would need to be integrated separately into the optimization process: 1) the smoothing of sensitivities and 2) the mapping of sensitivities between the individual domains. Furthermore, we saw that the custom Vertex Morphing translates the different mesh sensitivities into a (smooth) sensitivity information that covers the entire discretization of the structure, cf. figures 3.19(b) and 3.20(c). On this basis, it is possible to simultaneously control the design surface in both domains as well as the whole mesh of the structure.

3.8 Applications

In this section, we apply the developed multidisciplinary optimization process to the previously introduced 3D test case. To put the corresponding results in perspective, we first consider a single-disciplinary optimization of only the fluid in the internal channel.

3.8.1 Single-disciplinary optimization

This section only considers the interior channel from the 3D test case, see figure 3.22(a). We perform a single-disciplinary shape optimization based on regular Vertex Morphing to optimize this geometry. In doing so, we only consider pressure loss as the objective function. The filter radius is adopted from the original test case, see 3.5.2. To drive the optimization, we apply a simple steepest descent algorithm with an adaptive step size. Geometric constraints are given in the form of a fixed connection to the non-design domain at the inlet and outlet. Like in the multidisciplinary case, the fixation and the transition to the non-design domain is realized by damping geometry updates in the direct vicinity of the inlet and outlet. The damping follows the approach described in 2.5. That is, the damping is directly included in the parameterization so that no explicit geomet-

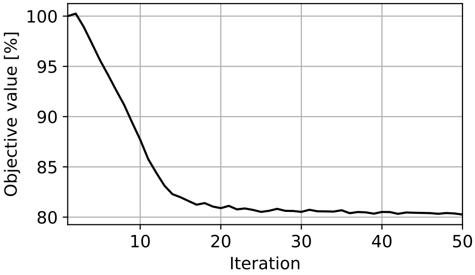
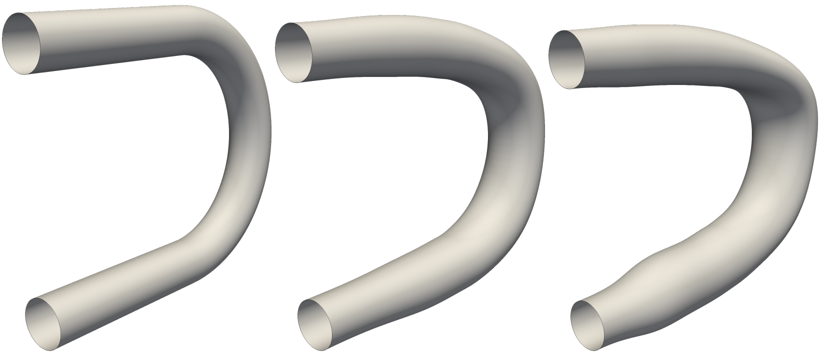


Figure 3.21: History of the objective value in the single-disciplinary optimization.



(a) Baseline design

(b) Iteration 14

(c) Iteration 50

Figure 3.22: Shape change in the single-disciplinary optimization.

ric constraints must be considered. The results of the single-disciplinary optimization of the fluid are presented in figure 3.21 and 3.22.

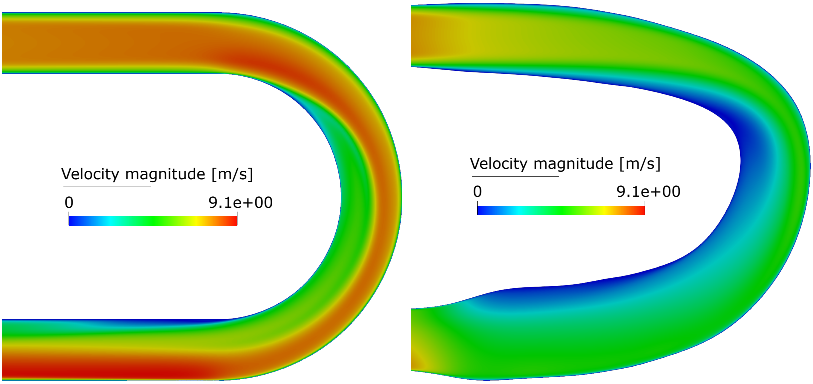
From the history of the objective value, one can see that after 50 iterations, the pressure loss in the bend is reduced by 19.7%. At this point, the optimization is not fully converged, but it reached the maximal number of iterations. Nevertheless, we can see in the graph that shape changes in the later course of the optimization only lead to slight improvements in the objective value. Note, in this context, that after 14 iterations (28% of the overall iterations), 90% of the final improvement is already reached.

Such behavior is typical for a shape optimization based on Vertex Morphing. The immense design freedom often allows for quick improvements with only a few iterations. However, convergence is difficult to reach and might take many iterations. The last aspect is also a consequence of the simple optimization algorithm, which relies only on first-order gradient information. Nevertheless, first-order algorithms are the primary choice in Vertex Morphing, because of their robustness and the collected experience from many practical applications. A detailed investigation of more advanced optimization algorithms in the context of Vertex Morphing is missing to date.

Figure 3.22 shows how the shape of the channel evolves. Note from the figure that significant shape changes are introduced, whereas an overall smooth surface is maintained throughout the optimization. Note also that the introduced geometry updates have a large characteristic length scale. We intentionally triggered the latter by choosing a large filter radius a priori. Finally, it is interesting to observe that the resulting shape is symmetric even though we did not formulate an explicit constraint in this regard.

From the shape evolution, one can see that the optimizer increases the diameter over big parts of the channel. As a consequence, the flow velocity drops significantly, see figure 3.23. The reduced velocity, in combination with the modified flow guidance, causes a significant reduction of energy losses in the fluid. Figure 3.24 quantifies the relevant losses using the total dissipation rate as described in Herwig et al. [77]. The total dissipation rate is computed as the sum of the mean dissipation rate based on the averaged flow field and the turbulent dissipation rate based on Menter's SST turbulence model. Both quantities cause a generation of entropy, responsible for the loss of energy in the fluid. We assume an isothermal flow in this context, so the influence of the temperature is neglected. Note in figure 3.24 how the optimization successfully eliminates hot spots of dissipation in the downstream direction. The overall reduced dissipation is responsible for the lower pressure loss in the optimized design.

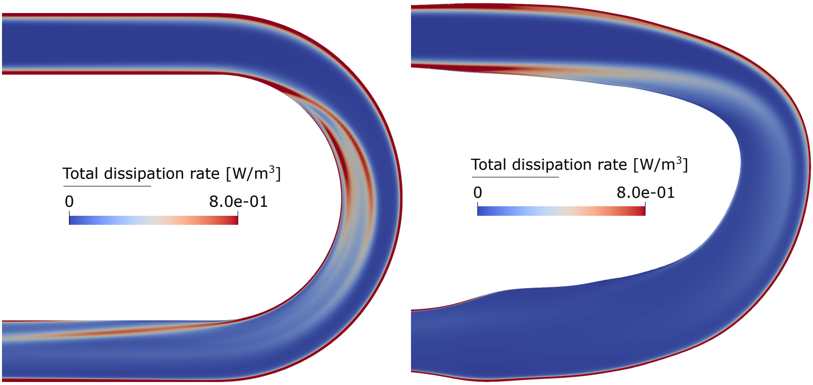
Noteworthy is that towards the outlet, the flow more and more becomes laminar up to the point that almost no turbulence can be observed directly at the outlet. So, the optimizer minimized pressure loss by a smart modification of the shape, which causes less dissipation. This result is a consequence of the high design freedom, which allows us to explore such



(a) Baseline design

(b) Iteration 50

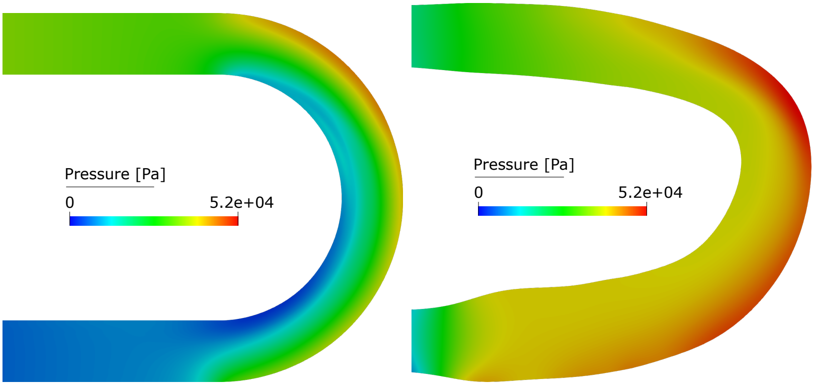
Figure 3.23: Velocity in the single-disciplinary optimization.



(a) Baseline design

(b) Iteration 50

Figure 3.24: Total dissipation rate in the single-disciplinary optimization.



(a) Baseline design

(b) Iteration 50

Figure 3.25: Static pressure in the single-disciplinary optimization.

potential in the first place.

In general, this example shows that Vertex Morphing may be successfully applied to optimize internal flows. In the present thesis, however, we not only want to consider the internal flow, but also the surrounding structure, which is subject to stress constraints. In this example, the velocity drop in the channel directly leads to an increase in the static pressure, see figure 3.25. An increasing static pressure, in turn, means a higher loading of the structure and, consequently, higher internal stresses. So we find that, in this case, a single-disciplinary optimization of only the fluid may quickly lead to problematic stress values in the surrounding structure, which eventually motivates the application of multidisciplinary optimization.

3.8.2 Multidisciplinary optimization

In this section, we apply the complete multidisciplinary optimization process proposed in this thesis to the 3D test case from section 3.5.2. The goal is to minimize the pressure loss of the interior fluid flow without increasing the maximum internal stress in the structure compared to the original design. The definition of the objective and the stress constraint is given in (3.62) and (3.63). We employ stress aggregation, as explained in section

3.5.2, so that only one constraint function needs to be considered by the optimization algorithm.

The complete optimization process is summarized in figure 3.26. The figure shows the primal and adjoint analysis each of the objective and the constraint function. The corresponding single disciplinary fluid and structure solvers are denoted as "CFD" and "CSM" or "Adj. CFD" and "Adj. CSM", respectively. As one can see in the figure, the primal fluid solver is only called once, and its result is used for both the objective and the constraint function. In contrast, the adjoint fluid solver is used two times, once for the sensitivity analysis of the objective function (pressure loss) and once to analyze the auxiliary force response in the sensitivity analysis of the constraint. The figure also shows the existing coupling in case of the constraint, including the mapping of relevant state variables. Apart from the primal and adjoint analysis, the figure identifies the Vertex Morphing Method within the overall process ("VM"). Finally, it presents the interface to the optimizer, which drives the overall process based on the specified optimization algorithm. The variables in the figure label exchange data. All variables are introduced in section 3.4. Arrows indicate the direction in which data is exchanged.

The figure shows some characteristics of the process that we already discussed earlier. For example:

- Computing the gradient of the objective (pressure loss) requires only a single-disciplinary adjoint analysis. There is no coupling involved.
- The coupling direction in the adjoint analysis of the constraint is just opposite to the coupling direction in the corresponding primal analysis.
- The coupled adjoint analysis in case of the constraint relies on the exchange of boundary conditions. The single-disciplinary adjoint solvers are used as black-boxes.
- The computation of $dg/d\mathbf{s}$ includes contributions from an adjoint fluid and an adjoint structure analysis. There is also a third contribution from the mapping between the different grids. However, for reasons specified earlier, this contribution is neglected, which is indicated by the dashed lines.

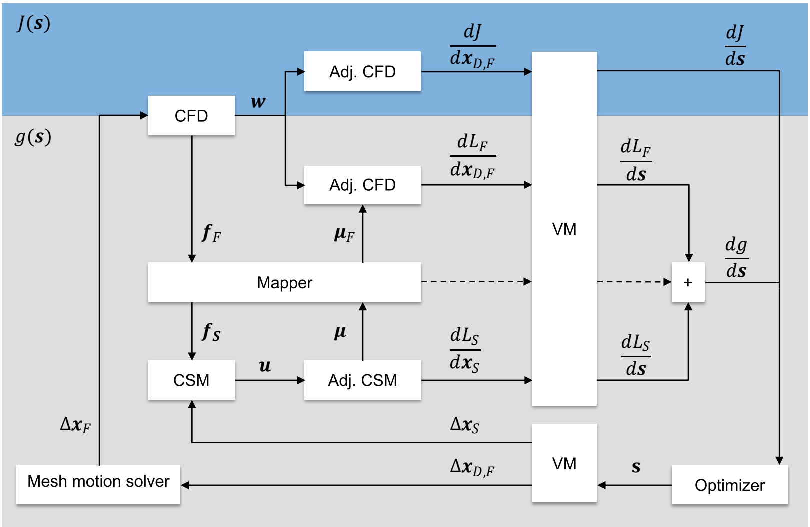


Figure 3.26: MDO process.

- In the herein presented variant of Vertex Morphing, we directly map geometry updates and sensitivities from the different simulation grids to the common discretization of the control field. So, no additional mapping of those quantities between the simulation grids is required.
- Vertex Morphing in the herein proposed form replaces a separate motion of the structure mesh. The fluid mesh must still be adapted separately using a dedicated process step (mesh motion solver).

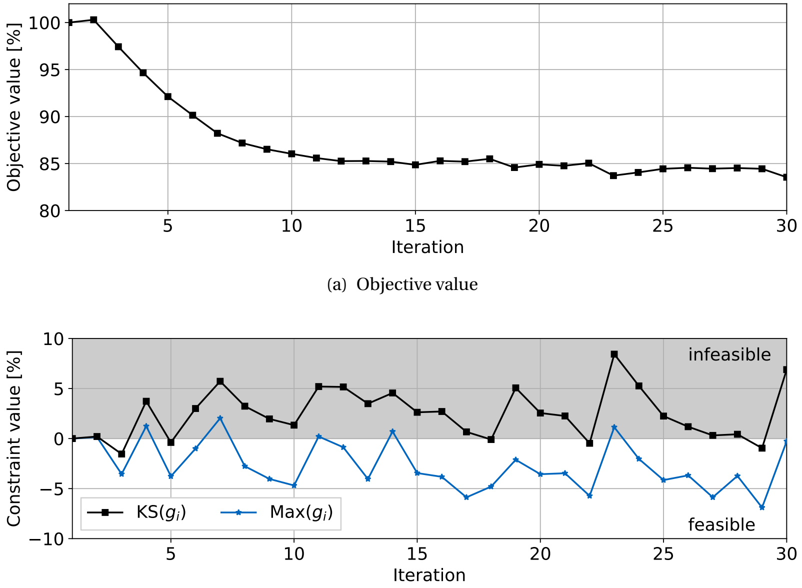
The figure also shows the interface to the optimizer. The optimizer obtains gradient information from both the objective and the constraint. According to the specified algorithm, it then proposes an update of the design variables. In this work, a variant of Rosen's gradient projection method for nonlinear constraints is applied, Rosen [78], Rosen [79], Haftka et al. [80]. So, a sequential unconstrained optimization is performed in the tangent subspace of the active constraint. No line-search is performed. Instead, we use a constant step size.

The algorithm combines gradient projection with an additional correction term. The projection avoids design updates in the direction of the infeasible domain, whereas the correction term forces the algorithm to restore feasibility if necessary. The correction relies on the penalization of infeasible response values. Also, it is applied "on the fly." I.e., no sub-iterations are performed to restore exact feasibility within every optimization iteration. Instead, we allow the algorithm to restore feasibility over several optimization iterations accepting that there may be infeasible intermediate designs. Gradient projection with constant step size is often used in conjunction with the Vertex Morphing Method due to its simplicity and robustness in practical problems, see Najian Asl et al. [14], for example.

The quantitative results of the multidisciplinary optimization are compiled in figure 3.27. The figure shows the history of the objective and the constraint value. In the case of the constraint, the figure plots: 1) the value of the KS-function, which represents the actual constraint value seen by the optimizer, i.e., an estimate of the maximal stress in the critical region, and 2) the true maximal stress for comparison. The optimization history comprises 30 iterations, which corresponds to the specified maximal number of iterations.

The figure shows that the optimization continuously improves the design in terms of pressure loss. After 30 iterations then, an improvement of ∼16% is achieved. Throughout the optimization, the objective is strongly conflicting with the constraint. The latter continuously enters the infeasible domain and must be significantly corrected by the optimizer. The reason for the observable jumps into the infeasible domain is that the given projection algorithm relies on the definition of an active set. If the given constraint is inactive, a simple steepest descent step is performed, which may cause a violation of the constraint again.

Comparing the values of the constraint, one finds that the true maximum value among all aggregated stress constraints (Max(g_i)) is significantly lower than the estimated maximum from the KS-function (KS(g_i)). The difference is due to the inherent approximation error of the KS-function with finite values of ρ. So, indeed the KS-function indicates a violation of the stress constraints in most of the optimization iterations. However, when looking at the original stress constraints (g_i), there are only a few iterations, where the current design really exceeds the predefined stress limit. Especially in iteration 30, the KS-function indicates an infeasible design.



(a) Objective value

(b) Constraint value relative to the initial design and normalized by the maximum allowed stress, so in black $(\mathrm{KS}(g_i) - \mathrm{KS}(g_i)^{(0)})/\sigma_{\max}$ and in blue $(\mathrm{Max}(g_i) - \mathrm{Max}(g_i)^{(0)})/\sigma_{\max}$.

Figure 3.27: History of the response values in the MDO.

However, the true maximum value among all the aggregated stress constraints is smaller than zero, meaning that stresses are below the threshold everywhere in the critical region. So, design 30 is practically feasible.

Regarding the herein proposed (coupled) sensitivity analysis, the most important observation in figure 3.27 is that the objective value overall decreases, and after every violation of the constraint, the optimizer is able to restore feasibility. Given the two heavily conflicting response functions, such a result implies valid gradient information. So, also the optimization results confirm the validity of the proposed (coupled) sensitivity analysis.

Striking in figure 3.27(a) is that the optimization is not yet fully converged. However, after 30 iterations, the specified maximum number of iterations is reached. The latter reflects the limited number of possible iterations due to the computational costs related to a single iteration[9]. Further converging the problem would require to restart the process at iteration 30 or to change the optimization algorithm. As changing the algorithm is the preferred approach, but requires more investigations concerning the included Vertex Morphing, further optimization is left to follow-up research. Instead, this work focuses on the given 30 design iterations, which is sufficient to demonstrate the possibilities of the proposed multidisciplinary optimization process. Among the given designs, the best feasible design, i.e., design 30, is further analyzed in the following.

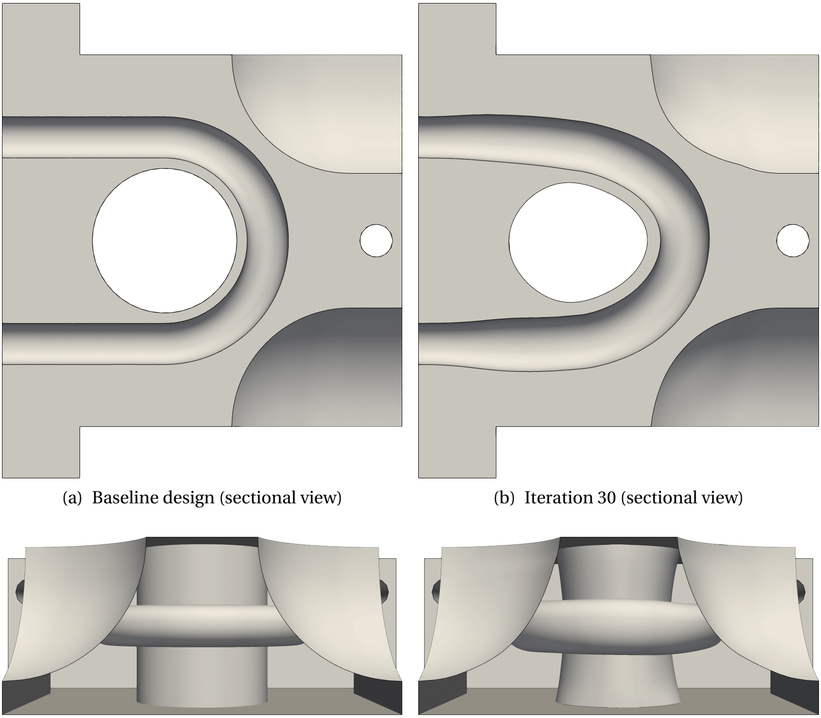
Figure 3.28 visualizes the shapes changes in design 30. In the figure, one can see that the structure is significantly modified. Striking in this context is the large characteristic length-scale of the shape modification. This characteristic is a result of the large filter radius in the proposed parameterization. Furthermore, we find that thin walls changed their shape as a whole, meaning their thickness remained almost constant, and opposite surfaces did not penetrate each other. Again, this characteristic is a result of the large filter radius.

In figure 3.28, one can also see that the optimized shape is generally smooth. Only in figure 3.28(b) at the two edges below and above the right bolt hole we observe a minor jump in the geometric continuity. This jump happens directly at the transition to the non-design domain and is due to the applied damping. As already mentioned in section 2.5, damping interferes with the smoothing in Vertex Morphing and may locally influence the geometric continuity.

In this case, damping was applied to restrict certain geometry updates on the edges of the design surface in favor of a proper transition between design and non-design domain. In figure 3.28, one can see the successful application of damping. Note in the figure, for example, the constant diameter at the inlet and outlet of the channel or the strict in-plane movement of the design surface on the structure's planar outside.

Regarding the internal fluid flow, the result of the multidisciplinary optimization is similar to the one in the single-disciplinary optimization of only the channel. That is, the internal flow in design 30 shows a signification reduced energy dissipation compared to the baseline design. As a result, the pressure loss decreases - in this case, by ∼16%. The figures 3.29(a) and

[9] Each iteration involves one primal and two adjoint RANS solutions.



(a) Baseline design (sectional view)

(b) Iteration 30 (sectional view)

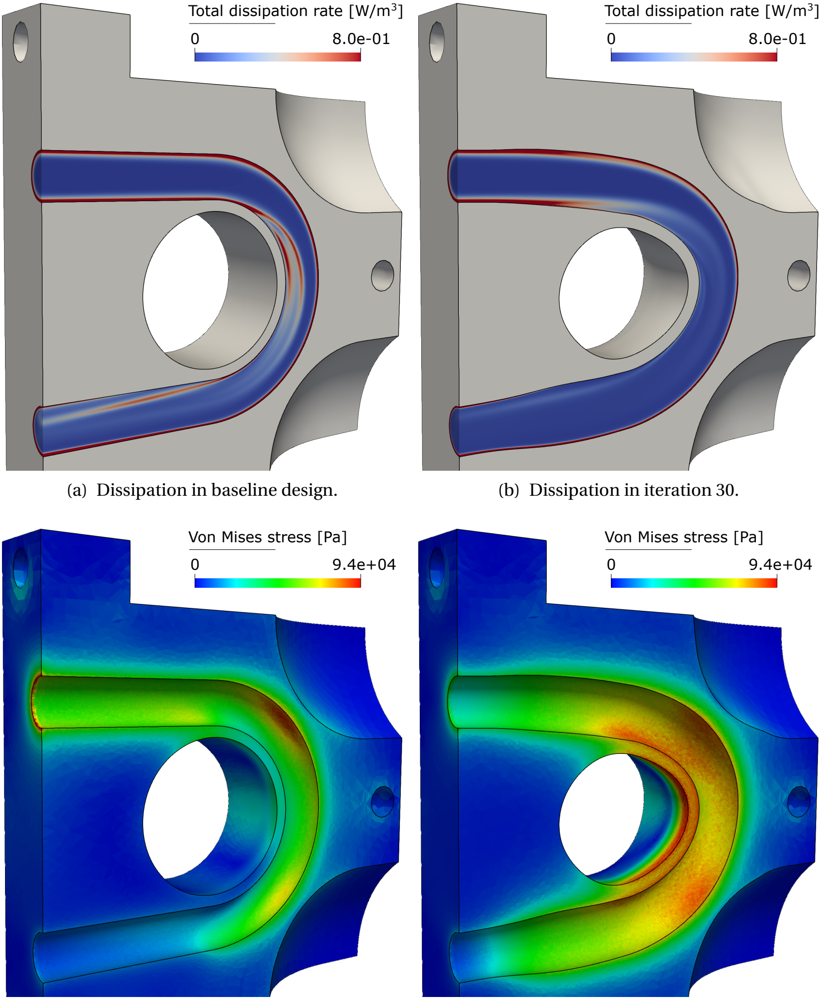
(c) Baseline design (view inside)

(d) Iteration 30 (view inside)

Figure 3.28: Shape change in the MDO.

3.29(b) visualize the effect using the total dissipation rate as described in Herwig et al. [77]. As one can see in the figure, hot spots around the areas with flow separation are almost completely eliminated.

Looking at the stresses in the structure, we can observe the conflicting nature of the two given response functions, see figures 3.29(c) and 3.29(d). Initially, there is just one spot with a critical stress value - on the boundary of the channel, where the flow gets redirected for the first time. However, during the optimization, the stress level continuously increases over big parts of the structure. The optimized design, eventually, shows many locations where the stresses reach a critical value. At the same time, the



(a) Dissipation in baseline design.

(b) Dissipation in iteration 30.

(c) Stresses in baseline design.

(d) Stresses in iteration 30.

Figure 3.29: Energy dissipation and stresses in the MDO.

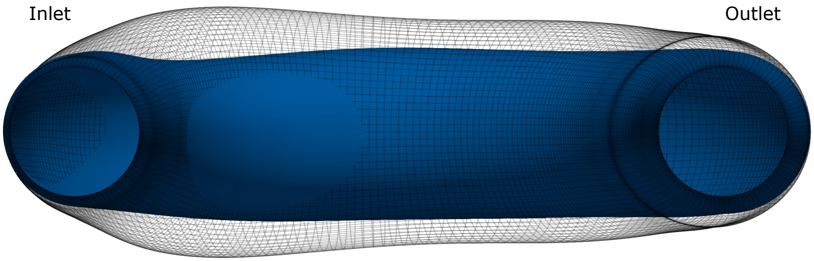


Figure 3.30: Shape of the channel after single-disciplinary (black mesh) and multi-discipinary optimization (blue surface).

region with the most critical values changed significantly. In the initial design, the most critical stress values concentrate around a small spot on the surface of the channel. After the optimization, we find them over a large area on the structure's outer face around the center hole. Despite the drastic increase in the overall stress level, the individual stress values nowhere exceed the specified threshold. So, as required by the specified constraint, the maximum stress in the optimal design is not greater than that in the initial design.

It is interesting to compare the results in this section with the results from the single-disciplinary optimization of only the channel in the previous section, cf. figure 3.30. Compared to the single-disciplinary optimization, where the optimizer considerably increased the diameter of the channel in favor of reduced flow velocities and hence an overall reduced energy dissipation, the MDO leads to a significantly smaller increase in the diameter. The reason is that an increasing diameter causes the flow to decelerate, which in turn raises the static pressure within the channel. A higher static pressure means a greater load on the structure, which quickly leads to a violation of the stress constraint. The multidisciplinary optimization recognizes this interaction between the fluid and the structure, so it prevents an arbitrary increase in the diameter. Instead, in the MDO, the pressure loss is mostly optimized by improving the flow guidance. The necessity to focus on the flow guidance in conjunction with the local nature of the stress constraint finally leads to an unsymmetric design, see especially the left half of the channel in figure 3.30.

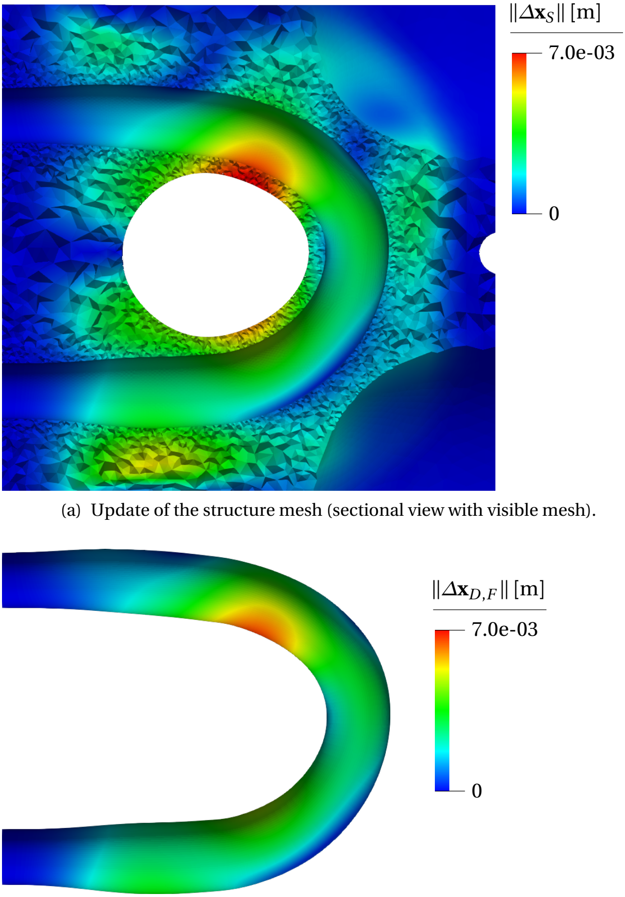
Note that the improvement in the MDO is lower than the improvement in the single-disciplinary optimization (~16% vs. ~20%). However, only in the MDO, the stress constraint is satisfied. The result is remarkable given that a manual improvement of the design seems impossible with the two heavily conflicting response functions - especially since an intuitive increase in diameter quickly causes a violation of the stress constraint.

In the following, we further investigate the role of the herein presented parameterization based on Vertex Morphing. The proposed parameterization inherits from the original Vertex Morphing the large design freedom, which in this example allowed a significant optimization without violating the constraint. However, unlike the original Vertex Morphing, the proposed parameterization includes the simultaneous control of several meshes over different domains. More precisely, with the proposed parameterization, we simultaneously control the design surface in the fluid domain as well as the entire volume mesh of the structure. So, an update of the control points causes an immediate update of the mesh in both domains. Figure 3.31 illustrates the effect. As an example, it shows the mesh updates as obtained by the displacement of the control points, **s**, in design iteration 22.

Note in the figure that **s** leads to a consistent update of both (differently discretized!) meshes. Also, the update of the structure mesh is not limited to the design surface but also includes the entire volume mesh. So, an extra mesh motion is not necessary anymore. In both meshes, the introduced damping avoids prohibited motions on the non-design boundaries. In the figure, we can see the damping effect from the vanishing mesh updates at the inlet and outlet.

In figure 3.31, one can also see that the proposed variant of Vertex Morphing leads to *smooth* mesh updates in both domains. Most striking in this context is the smooth update of the volume mesh of the structure, see figure 3.31(a). The latter is a result of the extended Vertex Morphing. Because of smooth updates across the entire volume, the structure mesh stays regular throughout the optimization.

In figure 3.31(a), one can also see that the mesh updates have a large characteristic length-scale - significantly larger than the individual elements. This characteristic length-scale is a result of the large filter radius. As a consequence of the filtering with a large filter radius, small elements in the mesh of the structure only experience little changes in their geometry and


$\|\Delta\mathbf{x}_S\|$ [m]
7.0e-03
0
$\|\Delta\mathbf{x}_{D,F}\|$ [m]
7.0e-03
0

(a) Update of the structure mesh (sectional view with visible mesh).

(b) Update of the design surface in the fluid domain (sectional view).

Figure 3.31: Update of the fluid and the structure mesh after a displacement of the control points in design iteration 22.

practically move as a whole. By contrast, the bigger the element, the greater its deformation. This observation reveals an important characteristic of Vertex Morphing when it comes to the control of volume meshes: If the filter radius is large compared to the average element size, then the filtering avoids a premature collapse of small elements. In other words, small elements, which may quickly collapse when deformed, react "stiffer" to a mesh update compared to large elements. This characteristic is desirable and similarly implemented in established mesh morphing techniques, like, for example, the pseudo-elastic approach, where the element size typically scales the stiffness of the mesh locally.

Apart from the positive characteristics presented above, figure 3.31(a) also reveals a disadvantage of Vertex Morphing when it comes to the control of volume meshes: It only introduces mesh updates in areas which are relevant for the given response functions (areas with high sensitivity values). Conversely, areas with vanishing sensitivities are not considered in the mesh update. As demonstrated with an example in section 2.4, this characteristic tends to generate local mesh updates instead of a global mesh motion. In figure 3.31(a), this effect is visible by the slightly spotted color distribution. Local mesh updates are not necessarily a problem, but a global mesh motion most likely leads to a better overall mesh quality.

Figure 3.31 showed that a displacement of the control points leads to a simultaneous mesh update in both the fluid and the structure domain. An obvious question in this regard is whether the updates are consistent, meaning the coupled boundaries do not drift apart. Due to the selected parameterization, cf. section 3.3, the updates in the present MDO are indeed consistent. The reason for this is that they are derived from a mesh-independent description of the geometry, cf. section 2.4. Figure 3.32 proves this claim qualitatively. The figure shows a superposition of the structure mesh and the discrete fluid boundary. Both meshes are from design iteration 30, i.e., the final design. As one can see from the fluctuating coloring, both meshes are still overlapping, so all the introduced mesh updates were consistent. The remaining deviations are deviations that already appear in the baseline design. They are only due to the non-matching discretization along Γ_I.

From all the results, one can conclude:

- With the herein developed optimization approach, we found a design, which causes significantly reduced pressure loss but still satisfies the stress constraint.

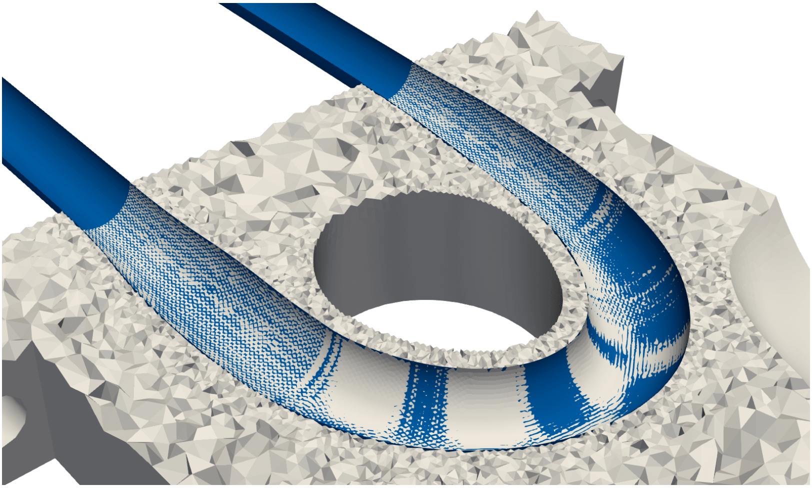


Figure 3.32: Sectional view of design 30 showing a superposition of structure mesh ($\mathbf{x}_S$, grey) and fluid boundary ($\mathbf{x}_{D,F}$, blue).

- The grid-based approach using the herein proposed variant of Vertex Morphing allowed for a design exploration beyond intuition.
- The proposed parameterization can be used to control both coupled and non-matching meshes simultaneously.
- The proposed parameterization successfully unifies the update of shape and mesh in the structure domain.

CHAPTER 4

B-REP MORPHING

The shape optimization in the previous chapter did intentionally not involve any CAD. Instead, it should allow for a quick setup of the optimization problem and provide maximal design freedom or high optimization potential, respectively. Therefore, a grid-based parameterization based on Vertex Morphing was applied. In grid-based shape optimization, the optimal design is given as discrete free-form geometry, typically in the form of a mesh with a corresponding displacement field (the overall shape change). A practical realization of the optimized design, however, often requires a CAD description. This chapter presents a method to convert the results of a grid-based shape optimization into a CAD geometry. The method is referred to as *B-Rep morphing*.

4.1 Basic idea and assumptions

The following data is assumed to be given:

1. a surface discretization in the form of an unstructured mesh of finite elements,
2. a displacement field defined on the mesh (representing a deformation from, e.g., a shape optimization process or any other simulation).
3. an initial CAD model in the form of a B-Rep model based on NURBS.

Moreover, it is assumed that the initial CAD model matches the given finite element mesh or can be sufficiently aligned otherwise. Having access to an initial CAD model that matches the finite element mesh is a reasonable assumption, as in many cases, the mesh is generated from such an initial CAD model.

The central assumption in the list above is the assumption of an existing CAD model. Given an initial CAD model, there is no need for a complete reconstruction to convert the deformed finite element mesh into a CAD geometry. Instead, the conversion may also be realized by adapting the initial CAD model such that it matches the given displacement field of the mesh. This approach has the advantage that it avoids a costly and error-prone definition of a non-unique CAD topology. Also, it allows to preserve design knowledge that is inherent to the original CAD model. So, the basic idea in this work is to obtain a CAD geometry of the deformed mesh by adapting an existing CAD model that belongs to the original mesh. The adaption includes a deformation of the initial CAD model as well as a model refinement, if necessary. The deformation of the CAD model is realized by *mapping* the displacement field from the mesh onto the CAD domain.

Practical CAD models typically include trimmed multi-patch surfaces. Deforming such a model can lead to a degrading surface smoothness or a loss of geometric continuity between the individual patches, especially in case of substantial shape changes. Maintaining the original surface quality is, however, crucial in many practical applications. Moreover, it is often necessary to consider independent design requirements in order to make the deformed CAD model usable further down the design process. In order to meet these practical requirements, the underlying mapping problem is systematically extended by additional objectives and constraints. Those extensions are an integral part of the method, as they make the method interesting for practical applications.

The prerequisite for a successful deformation of the given CAD model is that it has enough degrees of freedom to represent the intended deformation in the first place. Since this is not always the case, a refinement of the model may be necessary. Such refinement can be done based on the known displacement field. The latter can be analyzed to identify areas in the CAD model, which have an insufficient degree of freedom. The necessary refinement can then be realized either manually or using relevant algorithms. This thesis suggests a combined process consisting of an initial semi-automatic refinement of the CAD model and a subsequent mapping of the displacement field, including the extensions mentioned above. The combined process eventually defines what in this thesis is referred to as B-Rep Morphing.

4.2 Fundamentals

This section provides a brief introduction into the topic of NURBS and their topological arrangement using the concept of B-Rep modeling.

4.2.1 Curves and surfaces based on NURBS

NURBS (Non-Uniform Rational B-Splines) are parametric models used to generate and describe curves and surfaces with defined continuity properties. This section provides all definitions required to follow the later course of this thesis. For a detailed description of NURBS, as well as its differentiation to other parameter models like classical B-Splines or Bézier curves, the interested reader is referred to standard literature like Piegl et al. [81].

NURBS curves are defined by a set of control points $\mathbf{P}_i$ with each an associated weight w_i and a corresponding B-Spline basis function $M_{i,p}$ of polynomial order p. The basis functions depend on a parameter ξ. A NURBS curve is divided into individual segments. The latter are defined by a set of parametric coordinates, ξ_i, called the knots. The knots are collected in a knot vector Ξ. Let R_i describe the NURBS basis functions, the complete description of a NURBS curve reads:

$$\begin{aligned}\mathbf{C} &= \sum_{i=1}^{n} \frac{M_{i,p}(\xi) w_i \mathbf{P}_i}{\sum_{j=1}^{n} M_{j,p}(\xi) w_j} \\ &= \sum_{i=1}^{n} R_i(\xi)\, \mathbf{P}_i \end{aligned} \tag{4.1a}$$

$$\boldsymbol{\Xi} = [\xi_1, \xi_2, ..., \xi_{n_{\Xi}}] \tag{4.1b}$$

The knot vector determines where and how the control points affect the curve. Its values are arranged in ascending order, and the spacing between the individual knots can be non-uniform. Knot values appearing more than once are called multiple knots. The intervals between two consecutive, distinct knots are called nonzero knot spans dividing the curve into individual segments. The relation between the number of knots, n_{Ξ}, the number of control points, n, and the polynomial degree of the curve, p, must satisfy:

$$n_{\Xi} = n + p + 1 \tag{4.2}$$

In terms of continuity, the NURBS basis functions are C^{∞} continuous within a knot span and C^{p-1} continuous across single knots. At knots with multiplicity k, the continuity of the basis functions drops to C^{p-k}. Choosing knots with a multiplicity of $p+1$ forces the curve to interpolate the corresponding control point in geometry space. In that case, the curve may be readily split into two independent curve segments.

Analogously to a NURBS curve, a NURBS surface **S** is composed of a net of $n \times m$ weighted control points each associated with two B-Spline basis functions of the order p and q, which in turn are depending on the parameters ξ and η, respectively. A NURBS surface is divided into rectangular segments, which are defined by the knots in each parameter direction. The knots are collected in the two knot vectors $\boldsymbol{\Xi}$ and $\mathbf{H}$. The complete description of a NURBS surface (patch) reads:

$$
\begin{aligned}
\mathbf{S} &= \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{M_{i,p}(\xi)M_{j,q}(\eta)w_{ij}\mathbf{P}_{ij}}{\sum_{k=1}^{n}\sum_{l=1}^{m} M_{k,p}(\xi)M_{l,q}(\eta), w_{kl}} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m} R_{ij}(\xi,\eta)\,\mathbf{P}_{ij}
\end{aligned}
\tag{4.3a}
$$

$$\boldsymbol{\Xi} = [\xi_1, \xi_2, \ldots, \xi_{n_\Xi}] \tag{4.3b}$$

$$\mathbf{H} = [\eta_1, \eta_2, \ldots, \eta_{n_\eta}] \tag{4.3c}$$

$\mathbf{P}_{ij}$ represents the grid of control points. R_{ij} represents the corresponding NURBS basis functions. The role of the knot vectors is analogous to the case of a NURBS curve and described above. Given n control points in the first parameter direction and m control points in the second, it holds:

$$n_\Xi = n + p + 1 \tag{4.4}$$

$$n_\eta = m + q + 1 \tag{4.5}$$

Note from equation (4.1a) and (4.3a) that there are two spaces involved in a CAD geometry based on NURBS: a Eucledian (geometry) space and a parameter space. NURBS are essentially functions that map between these spaces. E.g., NURBS surfaces map a two-dimensional parameter point to a surface in three-dimensional space.

4.2.2 B-Rep models

A boundary representation (B-Rep) model is a hierarchical description of a model by its boundaries. For example, a solid may be described as a closed collection of faces, in which each face corresponds to a bounded region on a NURBS patch. The boundary of a face is composed of edges. Edges are again constructed by an ordered sequence of NURBS curves, which themselves are bounded by a dedicated start and end vertex. Following this logic, three main topological entities are identified:

1. Faces,

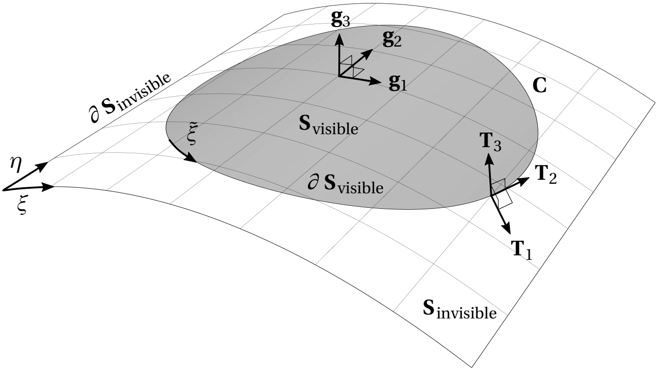

$\mathbf{g}_3$
$\mathbf{g}_2$
$\mathbf{g}_1$
$\mathbf{C}$
$\partial\,\mathbf{S}_{\text{invisible}}$
$\mathbf{S}_{\text{visible}}$
$\tilde{\xi}$
$\mathbf{T}_3$
$\mathbf{T}_2$
$\mathbf{T}_1$
η
ξ
$\partial\,\mathbf{S}_{\text{visible}}$
$\mathbf{S}_{\text{invisible}}$

Figure 4.1: Geometry description of a trimmed NURBS surface.

2. Edges,
3. Vertices.

In the context of this work, a *face* is a defined area on a rectangular NURBS patch. The area is again bounded by trimming loops, which can be classified into inner loops (holes) and outer loops (borders). These loops are constructed by an ordered network of (trimming) curves defined within the parameter space of the underlying NURBS surface. They divide the surface into a visible domain ("inside" the loops) and an invisible domain ("outside" the loops). NURBS surfaces defined by trimming curves are referred to as "trimmed" surfaces. Figure 4.1 illustrates such a surface.

Trimming curves are determined through boolean or blending operations on different geometric entities - for example, as the intersection of two surfaces. In the context of NURBS surfaces, trimming curves are typically also described with NURBS. The definition of such a trimming curve reads:

$$\tilde{\mathbf{C}}(\tilde{\xi}) = \sum_{i=1}^{n} R_i(\tilde{\xi})\,\tilde{\mathbf{P}}_i \tag{4.6}$$

$\tilde{\bullet}$ denotes the parameter space of an underlying NURBS surface. As the equation suggests, the curve is fully laying in the parameter space of a parent surface and includes an own parameter $\tilde{\xi}$. Mapping (4.6) to the geometry space through (4.3a), one obtains the actual boundary of the surface, cf. figure 4.1:

$$\mathbf{C}(\tilde{\xi}) = \mathbf{S}(\xi(\tilde{\xi}), \eta(\tilde{\xi})) \tag{4.7}$$

Tagging the boundary curve with additional information regarding its connectivity to neighboring curves and surfaces, it eventually becomes a topological *(B-Rep) edge* denoted as $\partial\mathbf{S}$. In this work, the concept of B-Rep edges is heavily exploited to define geometric constraints on surface boundaries, e.g., to control the continuity across individual faces in the deformed CAD geometry. B-Rep edges coupled to several faces are herein denoted as *coupled edges*.

A complete B-Rep model may be constructed as a collection of trimmed (visible) faces, locally connected at their edges and globally arranged as either volume or surface bodies. The latter may be again combined to complete assemblies. This hierarchical structure allows the design of most complex models following a process of divide and conquer. For more details on B-Rep models as described above, the interested reader is referred to Breitenberger et al. [82] and Teschemacher et al. [83].

4.2.3 Displacement and orientation

The displacement of a NURBS surface $\Delta\mathbf{S}$ is defined as the difference between its reference (initial) configuration $\mathbf{S}^{(0)}(\xi, \eta, \mathbf{P}^{(0)})$ and its current configuration $\mathbf{S}(\xi, \eta, \mathbf{P})$. Defining $\Delta\mathbf{P} = \mathbf{P} - \mathbf{P}^{(0)}$, it holds:

$$\Delta\mathbf{S} = \mathbf{S} - \mathbf{S}^{(0)} = \sum_{i=1}^{n} \sum_{j=1}^{m} R_{ij}(\xi, \eta)\, \Delta\mathbf{P}_{ij} \tag{4.8}$$

The orientation of a NURBS surface is described using differential geometry. In doing so, an orthogonal coordinate system is defined for each point on the surface, whereas the corresponding base vectors are aligned to the surface's parameter directions. The base vectors are derived as follows (see figure 4.1 for a visualization):

$$\mathbf{g}_1 = \frac{\partial \mathbf{S}}{\partial \xi}, \quad \mathbf{g}_2 = \frac{\partial \mathbf{S}}{\partial \eta}, \quad \mathbf{g}_3 = \frac{\mathbf{g}_1 \times \mathbf{g}_2}{\left\| \mathbf{g}_1 \times \mathbf{g}_2 \right\|_2} \tag{4.9}$$

Similarly, the orientation along an edge of a NURBS surface is characterized by an orthogonal coordinate system aligned to the parameter direction of the underlying trimming curve. The base vectors spanning this coordinate system are obtained as (cf. figure 4.1):

$$\tilde{\mathbf{T}}_2 = \frac{\partial \mathbf{C}}{\partial \tilde{\xi}} = \mathbf{g}_1 \frac{\partial \xi}{\partial \tilde{\xi}} + \mathbf{g}_2 \frac{\partial \eta}{\partial \tilde{\xi}}, \quad \tilde{\mathbf{T}}_3 = \mathbf{g}_1 \times \mathbf{g}_2, \quad \tilde{\mathbf{T}}_1 = \tilde{\mathbf{T}}_2 \times \tilde{\mathbf{T}}_3 \tag{4.10}$$

$$\mathbf{T}_i = \frac{\tilde{\mathbf{T}}_i}{\left\| \tilde{\mathbf{T}}_i \right\|_2} \tag{4.11}$$

4.2.4 Geometric continuity

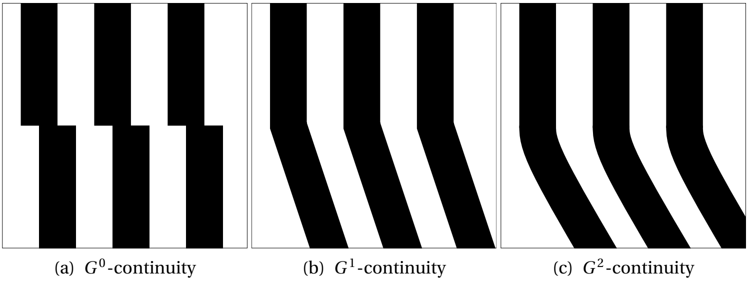
Practical CAD models often consist of multiple, up to thousands of individual patches carrying trimmed faces arranged to complex geometries, see GrabCAD Inc. [84] for impressions from a relevant online community. In such a multi-patch model, the geometric continuity between the individual faces is an important quality criterion. One distinguishes different orders of geometric continuity. In this work, we focus on:

- G^0 or positional continuity (watertightness), and
- G^1 or tangential continuity.

Positional continuity is obtained, if two edges, $\partial \mathbf{S}^M$ and $\partial \mathbf{S}^S$, from two faces, $\mathbf{S}^M$ and $\mathbf{S}^S$, coincide geometrically:

$$\partial \mathbf{S}^M = \partial \mathbf{S}^S = \partial \mathbf{S} \tag{4.12}$$

Tangential continuity is obtained when two adjacent surfaces do not only share a common edge, but also a common tangent direction $\mathbf{T}_1$, cf. figure



(a) G^0-continuity (b) G^1-continuity (c) G^2-continuity

Figure 4.2: Reflection pattern depending on the geometric continuity.

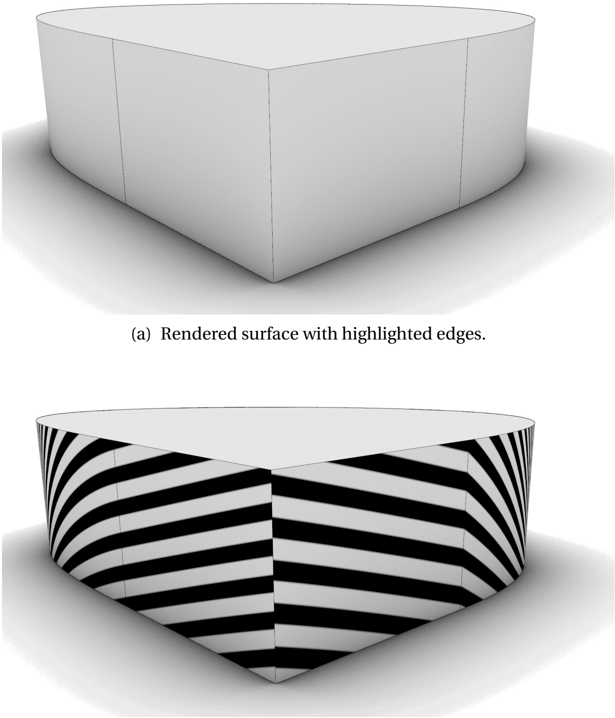
4.1. As per definition $\mathbf{T}_1$ points away from the visible surface, tangential continuity requires:

$$\mathbf{T}_1^M = -\mathbf{T}_1^S \tag{4.13}$$

A practical method for assessing the geometric continuity in a CAD model is to examine reflection lines on its surface. To this end, a pattern with straight, parallel, black and white stripes (zebra pattern) is projected onto the CAD surface, which itself is assumed to be a perfectly reflective object. The corresponding reflection pattern is then calculated and directly visualized on the surface. Depending on the continuity, the reflected stripes show different gradients, eventually revealing continuity jumps, see figure 4.2 and 4.3. This type of surface analysis is available in many commercial CAD tools. In the analog world, a similar technique is often used at the end of a car's assembly line to assess the surface quality of the exterior parts.

4.3 Numerical integration over surfaces and edges

B-Rep morphing requires the evaluation of integrals over NURBS surfaces and B-Rep edges. Given a NURBS surface $\mathbf{S}$ or a B-Rep edge $\partial\mathbf{S}$, the relevant integrals are of the form:



(a) Rendered surface with highlighted edges.

(b) Reflection pattern with highlighted edges.

Figure 4.3: Qualitative evaluation of geometric continuity. Despite looking smooth on both sides, the reflection pattern reveals each a different continuity across the three edges.

$$|\mathbf{S}| = \int_{\mathbf{S}} \mathrm{d}\mathbf{S} \tag{4.14}$$

$$|\partial\mathbf{S}| = \int_{\partial\mathbf{S}} \mathrm{d}\partial\mathbf{S} \tag{4.15}$$

Integrating over NURBS surfaces and edges is a well-known task in the field of isogeometric B-Rep analysis or immersed boundary methods in general.

Established approaches in the first case are described in Teschemacher et al. [83], Breitenberger [85], and Hughes et al. [86]. An exemplary alternative, developed in the context of the Finite Cell Method, is presented in Kudela et al. [87]. In the present work, we focus on solutions from the field of isogeometric B-Rep analysis.

4.3.1 Integration over NURBS surfaces

Consider a NURBS surface **S** with n faces (patches). The integral over such a surface can be computed by integrating over the sub-faces in each patch separately and then summing up the individual contributions:

$$|\mathbf{S}| = \int_{\mathbf{S}^1} \mathrm{d}\mathbf{S}^1 + \int_{\mathbf{S}^2} \mathrm{d}\mathbf{S}^2 + \ldots \int_{\mathbf{S}^n} \mathrm{d}\mathbf{S}^n \tag{4.16}$$

In the integration of a sub-face, one needs to distinguish two cases : 1) the face is untrimmed, 2) the face is trimmed.

Untrimmed faces

In the untrimmed case, a standard integration is applied as introduced by Hughes et al. [86] in the context of isogeometric analysis. Accordingly, the patch that belongs to sub-face $\mathbf{S}^k$ is clipped along its knot lines into m quadrilateral elements, $\mathbf{E}^k$:

$$\int_{\mathbf{S}^k} \mathrm{d}\mathbf{S}^k = \int_{\mathbf{E}^1} \mathrm{d}\mathbf{E}^1 + \int_{\mathbf{E}^2} \mathrm{d}\mathbf{E}^2 + \ldots \int_{\mathbf{E}^m} \mathrm{d}\mathbf{E}^m \tag{4.17}$$

The integral associated with each element is transformed to the Gaussian domain $\mathcal{G}$ using the following relation:

$$\int_{\mathbf{E}^k} \mathrm{d}\mathbf{E}^k = \int_{\xi} \int_{\eta} J_1 \mathrm{d}\xi \mathrm{d}\eta = \int_{\mathcal{G}} J_1 J_2 \mathrm{d}\mathcal{G} \tag{4.18}$$

Herein, J_1 denotes the Jacobian of the mapping from geometry to parameter space, and J_2 the Jacobian of the mapping from parameter space to the Gaussian domain. It holds:

$$J_1 = \left\| \mathbf{g}_1 \times \mathbf{g}_2 \right\|_2 \tag{4.19}$$

and

$$J_2 = \frac{\partial \xi}{\partial \xi_{\mathcal{G}}} \frac{\partial \eta}{\partial \eta_{\mathcal{G}}} \tag{4.20}$$

$\mathbf{g}_1$ and $\mathbf{g}_2$ are the local base vectors, cf. figure 4.1. $\xi_{\mathcal{G}}$ and $\eta_{\mathcal{G}}$ are the parameters of the Gaussian domain ($\xi_{\mathcal{G}} \in [-1,1] \times \eta_{\mathcal{G}} \in [-1,1]$).

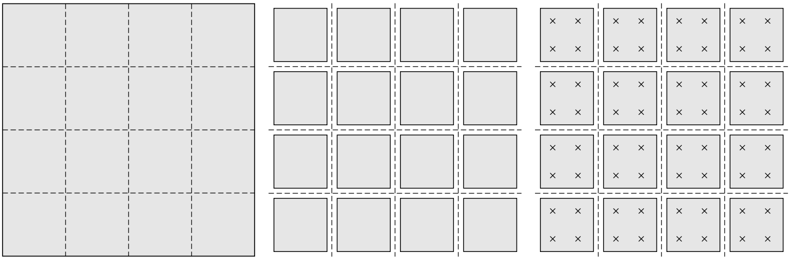
For each quadrilateral element, $p+1$ Gauss points are introduced in both parameter directions, whereas p corresponds to the maximum polynomial degree in the NURBS basis functions describing the underlying face. Each Gauss point is associated with a weight w_i. The integral within an element is eventually computed as the sum of the contributions of all $n_{\mathcal{G}}$ Gauss points:

$$\int_{\mathbf{E}^k} \mathrm{d}\mathbf{E}^k \approx \sum_{i=1}^{n_{\mathcal{G}}} J_1 J_2 w_i \tag{4.21}$$

Note that the clipping into individual elements is necessary because the Gaussian quadrature approximates an integral using polynomial Ansatz-functions. Consequently, it only delivers acceptable results within a region that can be approximated by polynomials. NURBS functions are piecewise rational polynomial functions, whereas the knot spans determine the individual pieces. An exact integration using Gaussian quadrature is hence not possible. However, assuming a polynomial form within each knot span and integrating it separately with $p+1$ Gauss points in each parameter direction is an established best-practice. Figure 4.4 conceptually visualizes the clipping and the determination of the Gauss points in case of an untrimmed face.

Trimmed faces

If a sub-face $\mathbf{S}^k$ is trimmed, then the corresponding patch is also clipped along its knot lines into individual, quadrilateral elements. Unlike in the



(a) Untrimmed face (b) Clipping of domain (c) Introduction of Gauss points

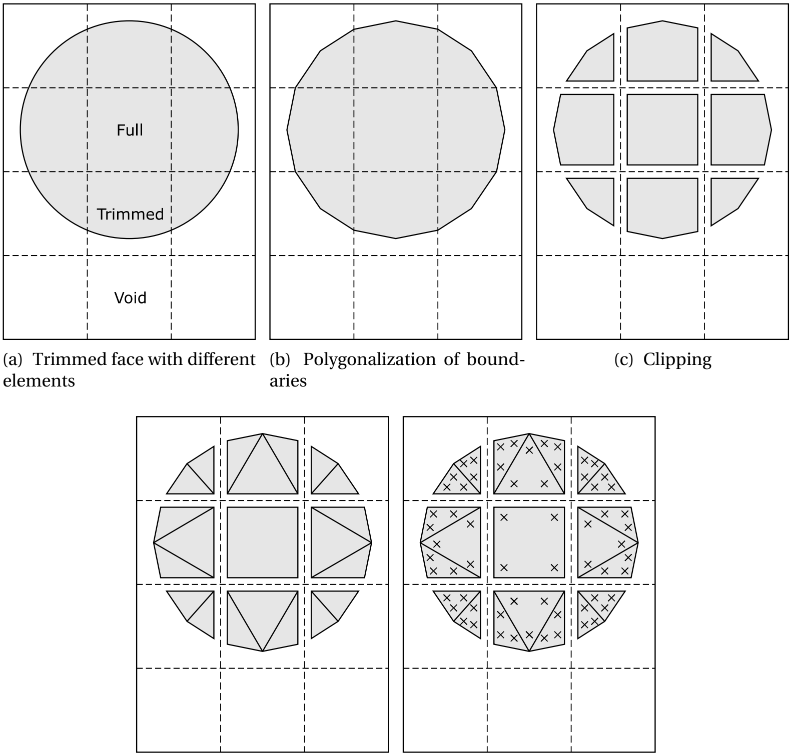
Figure 4.4: Steps to determine Gauss points in case of an untrimmed face (knot lines are drawn as dashed lines).

untrimmed case, however, the patches now have an invisible domain. The latter must be excluded from the integration. In this context, one can distinguish three different types of elements, cf. 4.5(a):

1. void,
2. full,
3. trimmed.

"Void" elements are entirely located in the invisible regime and can be simply omitted in the integration. "Full" elements are fully visible and can be integrated using the integration procedure for a single element, as described in the previous section. For the integration over "trimmed" elements, the visible domain in the trimmed element is tesselated, and each resulting triangle is integrated separately using Gaussian quadrature. The approach for the integration of trimmed surfaces is adopted from Oberbichler [88, 89]. The five necessary steps of the approach are summarized in the following, cf. figure 4.5.

1. Polygonalization of all boundaries,
2. Clipping of the parameter space,
3. Tessellation into quads & triangles,



(a) Trimmed face with different elements

(b) Polygonalization of boundaries

(c) Clipping

(d) Tessellation

(e) Introduction of Gauss points for quads and triangles

Figure 4.5: Steps to determine Gauss points for an untrimmed face (knot lines are drawn as dashed lines).

4. Determination of Gauss points & weights,
5. Numerical solution.

Given the individual elements and the cloud of Gauss points, the integral over a trimmed face is finally computed as described in (4.17) and (4.21).

4.3.2 Integration over B-Rep edges

Apart from the integration over NURBS surfaces, the present work also requires to integrate along B-Rep edges. Two different types of edges are considered in this context: coupling edges and free edges. Free edges are edges that have no link to another face. Coupling edges are edges along joined faces. In both cases, the integration is done following the Gaussian quadrature introduced by Breitenberger [85] in the context of isogeometric B-Rep analysis (IBRA). Analogous to the surface integration, the basic approach is to divide the integration domain, i.e., an edge $\partial\mathbf{S}$, into elements and integrate them separately:

$$|\partial\mathbf{S}| = \int_{\mathbf{E}^1} \mathrm{d}\mathbf{E}^1 + \int_{\mathbf{E}^2} \mathrm{d}\mathbf{E}^2 + \ldots \int_{\mathbf{E}^n} \mathrm{d}\mathbf{E}^n \tag{4.22}$$

For a detailed explanation of the integration procedure and further background information, the reader is referred to sections 4.3, 5.3, and 5.4 in Breitenberger [85] or section 5.4 in Breitenberger et al. [82]. The procedure is summarized in the following.

The integration of B-Rep edges consists of three main steps:

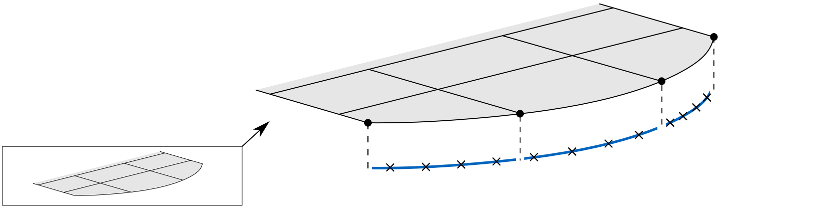
1. Clipping
2. Determination of Gauss points & weights
3. Numerical solution

The first two steps differ depending on the type of the edge. Figure 4.6 explains the procedure in both cases conceptually.

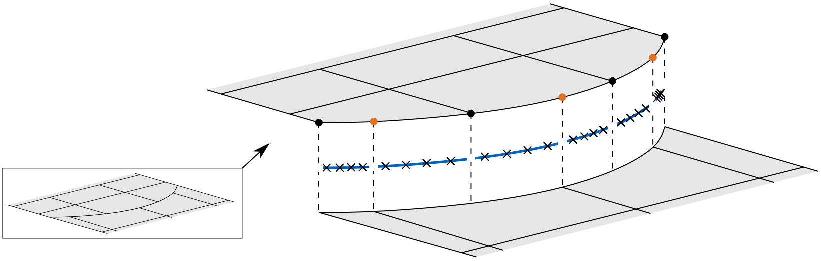
Given an edge, which was clipped into individual elements, the integral associated to each element is mapped to the Gaussian domain $\mathcal{G}$ according to the following relation:

$$\int_{\mathbf{E}^k} \mathrm{d}\mathbf{E}^k = \int_{\tilde{\xi}} \tilde{J}_1 \mathrm{d}\tilde{\xi} = \int_{\mathcal{G}} \tilde{J}_1 \tilde{J}_2 \mathrm{d}\mathcal{G} \tag{4.23}$$

Herein $\tilde{\xi}$ denotes the curve parameter, $\tilde{J}_1$ the Jacobian of the mapping from geometry to the curve's parameter space, and $\tilde{J}_2$ the Jacobian of



(a) Free edge: First, the edge is clipped at all knot lines to form individual B-Rep elements (blue). Then Gauss points are introduced separately for every element depending on the polynomial order of the underlying curve (crosses).



(b) Coupled edge: First a master curve is selected (top curve). Then all intersections with the knot lines on slave side are projected to the Master curve (orange points). These intersection together with all knot intersections on master side serve as clipping points to form individual B-Rep elements (blue). Gauss points are finally introduced for every element separately depending on the polynomial order of the underlying curve (crosses).

Figure 4.6: Clipping and determination of Gauss points at a free and a coupled B-Rep edge.

the mapping to the Gaussian domain. $\tilde{J}_1$ is deformation dependent and defined as:

$$\tilde{J}_1 = \left\| (\mathbf{g}_1 \cdot \tilde{t}_\xi + \mathbf{g}_2 \cdot \tilde{t}_\eta) \right\|_2 \tag{4.24}$$

whereas $\tilde{t}_\xi$ and $\tilde{t}_\eta$ are components of the tangent vector of the boundary curve in the two parameter directions of the underlying patch:

$$\tilde{t}_\xi = \frac{\partial \xi}{\partial \tilde{\xi}}, \quad \tilde{t}_\eta = \frac{\partial \eta}{\partial \tilde{\xi}} \tag{4.25}$$

$\tilde{J}_2$ is deformation independent. Let $\xi_{\mathcal{G}}$ be the parameter of the Gaussian domain ($\xi_{\mathcal{G}} \in [-1, 1]$), then it holds:

$$\tilde{J}_2 = \frac{\partial \tilde{\xi}}{\partial \xi_{\mathcal{G}}} \tag{4.26}$$

Once the integral over an individual element is mapped into the Gaussian domain, Gauss points are introduced and each assigned with a weight w_i. The integral over the element is then computed by summing up the contributions of all $n_{\mathcal{G}}$ Gauss points:

$$\int_{\mathbf{E}^k} \mathrm{d}\mathbf{E}^k \approx \sum_{i=1}^{n_{\mathcal{G}}} \tilde{J}_1 \tilde{J}_2 w_i \tag{4.27}$$

4.4 Underlying mapping operation

B-Rep morphing is based on a mapping operation. In this section, the underlying mapping is introduced, a solution approach is developed, and details on the implementation are discussed. The mapping is finally tested at a generic benchmark example.

4.4.1 Formulation of the mapping problem

Given the input specified in section 4.1, the goal is to map the displacement field from the finite element mesh to the NURBS surface such that the CAD model fits the deformed mesh. The displacement of a NURBS surface is represented by $\Delta\mathbf{S}$ and is defined according to (4.8). The input displacement of the finite element mesh is denoted $\Delta\mathbf{Q}$ and is defined as follows:

$$\Delta\mathbf{Q} = \sum_{i=1}^{n_Q} N_i \Delta\widehat{\mathbf{Q}}_i \tag{4.28}$$

Herein, n_Q represents the number of discrete mesh nodes. N_i are standard finite element basis functions and $\Delta\widehat{\mathbf{Q}}_i$ are the specified nodal displacements. If no finite element basis functions are prescribed, they are assumed to be (bi)linear functions.

To realize the mapping, we follow a least-squares approach. Accordingly, we seek for a displacement field $\Delta\mathbf{S}$ with minimal deviation from the prescribed displacement $\Delta\mathbf{Q}$, whereas the deviation is measured using the L_2-norm. The corresponding optimization problem is in the following referred to as the *mapping problem*:

$$\begin{aligned}
&\min_{\Delta\mathbf{P}} \; \Pi(\Delta\mathbf{P}) \\
&\Pi(\Delta\mathbf{P}) = \frac{1}{2}\int_{\mathbf{S}^{(0)}} \left[\Delta\mathbf{S}(\xi,\eta,\Delta\mathbf{P}) - \Delta\mathbf{Q}\right]\cdot\left[\Delta\mathbf{S}(\xi,\eta,\Delta\mathbf{P}) - \Delta\mathbf{Q}\right] \, \mathrm{d}\mathbf{S}^{(0)} \\
&\qquad\quad = \frac{1}{2}\int_{\mathbf{S}^{(0)}} [\Delta\mathbf{S} - \Delta\mathbf{Q}]\cdot[\Delta\mathbf{S} - \Delta\mathbf{Q}] \, \mathrm{d}\mathbf{S}^{(0)}
\end{aligned} \tag{4.29}$$

Herein, the NURBS surface is chosen as integration domain due to its exact description of the initial geometry. Important for later derivations is the fact, that the integral is always evaluated at the initial geometry $\mathbf{S}^{(0)}$ (Total Lagrange approach). By that, the integration domain does not change through a modification of the control points.

The mapping strategy described above corresponds to a mortar-based mapping strategy as, e.g., used by Apostolatos et al. [90] and Apostolatos [91] in the context of FSI. Important to note is that the mapping problem represents an inverse problem and thus may require regularization. The latter is discussed in section 4.5.

4.4.2 Solution of the mapping problem

Solving the mapping problem requires vanishing first order derivatives of the functional in (4.29), i.e.:

$$\frac{d\Pi}{d\Delta\mathbf{P}} \overset{!}{=} 0 \tag{4.30}$$

Since the problem is quadratic, this requirement yields a linear system that could be solved using any linear solution technique. Still, we use a

Newton-Raphson approach because some constraints, which we add later, render the problem nonlinear. Following the Newton-Raphson method, the solution of (4.30) is computed iteratively as:

$$\Delta \mathbf{P}_{n+1} = \Delta \mathbf{P}_n + \Delta(\Delta \mathbf{P})_{n+1} \tag{4.31}$$

$$\left.\frac{\partial^2 \Pi}{\partial \Delta \mathbf{P}^2}\right|_n \Delta(\Delta \mathbf{P})_{n+1} = -\left.\frac{\partial \Pi}{\partial \Delta \mathbf{P}}\right|_n \tag{4.32}$$

(4.32) is in the following referred to as the *mapping system*. The system or mapping matrix (left-hand side) and the residual vector (right-hand side) are abbreviated by the letters **A** and **b**, respectively:

$$\mathbf{b} = -\frac{\partial \Pi}{\partial \Delta \mathbf{P}} \tag{4.33}$$

$$\mathbf{A} = \frac{\partial^2 \Pi}{\partial \Delta \mathbf{P}^2} \tag{4.34}$$

For a better overview, an index notation including a short hand for the partial derivative is introduced:

$$b_r = -\frac{\partial \Pi}{\partial \Delta P_r} = -\partial_r(\Pi) \tag{4.35}$$

$$A_{rs} = \frac{\partial^2 \Pi}{\partial \Delta P_r \partial \Delta P_s} = \partial^2_{rs}(\Pi) \tag{4.36}$$

Given the definition of the mapping problem in (4.29), the resulting contributions to the mapping system are:

$$b_r = -\partial_r(\Pi) = \int_{\mathbf{S}^{(0)}} \partial_r(\Delta \mathbf{S}) \cdot [\Delta \mathbf{S} - \Delta \mathbf{Q}] \; \mathrm{d}\mathbf{S}^{(0)} \tag{4.37}$$

$$A_{rs} = \partial^2_{rs}(\Pi) = \int_{\mathbf{S}^{(0)}} \partial_r(\Delta \mathbf{S}) \cdot \partial_s(\Delta \mathbf{S}) \, \mathrm{d}\mathbf{S}^{(0)} \tag{4.38}$$

For the numerical solution of the mapping problem, we adopt the finite element approach from isogeometric analysis, cf. Hughes et al. [86]. Accordingly, the entire surface is discretized into individual elements, and a Gaussian quadrature is performed over each of those elements to compute the local left- and right-hand side contributions. The elemental contributions are eventually assembled to the global mapping system. The discretization of a NURBS surface into elements and the corresponding numerical integration follow the procedure described in section 4.3. Following this approach, the left- and right-hand side contributions of a single element are computed as:

$$b_r = -\sum_{i=1}^{n_{GP}} J_1 J_2 w_i \; \partial_r(\Delta \mathbf{S}_i) \cdot [\Delta \mathbf{S}_i - \Delta \mathbf{Q}_i] \tag{4.39}$$

$$A_{rs} = \sum_{i=1}^{n_{GP}} J_1 J_2 w_i \; \partial_r(\Delta \mathbf{S}_i) \cdot \partial_s(\Delta \mathbf{S}_i) \tag{4.40}$$

In the previous equation, the index i indicates a single Gauss point located at parameter position $\left(\xi_i, \eta_i\right)$. n_{GP} represents the number of all Gauss points in the element under consideration. $\Delta \mathbf{S}_i$ represents the displacement of the NURBS surface at the Gauss point. $\Delta \mathbf{Q}_i$ denotes the target displacement which has to be assigned to each Gauss point $\mathbf{S}_i$. The assignment is realized by projecting the Gauss points onto the mesh and interpolating the local displacement using the shape functions of the finite elements. The approach is further explained in the following section. Important to note is that the assignment is only done once in the initial configuration and then not altered during possible solution iterations.

Worthwhile to mention is that one could also think about solving the mapping problem using other immersed boundary or fictitious domain methods, like, e.g., presented in Rank et al. [92]. In this work, however, we will not further elaborate on this option.

4.4.3 Nearest element interpolation

To set up the mapping system, (4.32), a local displacement of the input mesh, $\Delta \mathbf{Q}_i$, has to be assigned to each introduced integration point, $\mathbf{S}_i$. The assignment of displacements can be accomplished by projecting the integration points onto the finite element mesh, then interpolating the nodal displacements at the projected points using the finite element shape functions, and finally assigning the interpolated values to the correspond-

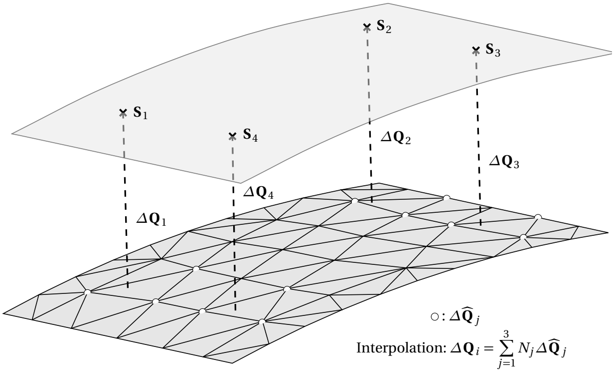


Figure 4.7: Assignment of displacements to integration points using nearest element interpolation.

ing integration points. The combination of projection, interpolation, and subsequent assignment resembles another mapping step, where the displacement field from the finite element mesh is mapped to the cloud of integration points using nearest element interpolation, cf. Wang et al. [63]. An implementation of this technique is freely available through the open-source framework "Kratos Multiphysics" [70]. The mentioned implementation is also used in the present work. The basic steps of a nearest element interpolation are summarized in algorithm 4.1 and illustrated in figure 4.7.

Algorithm 4.1: Nearest element interpolation

```
foreach Integration point $\mathbf{S}_i$ on initial CAD surface $\mathbf{S}^{(0)}$ do
    Identify subset of elements around $\mathbf{S}_i$ → set ;
    Orthogonal projection of $\mathbf{S}_i$ on all elements in
      set → distances ;
    Choose nearest element according distances → $E$;
    Interpolate $\Delta\mathbf{Q}$ at the projected point in $E$ using its shape
      functions $\mathbf{N}$ → $\Delta\mathbf{Q}_i$ ;
    Assign $\Delta\mathbf{Q}_i$ to $\mathbf{S}_i$ ;
```

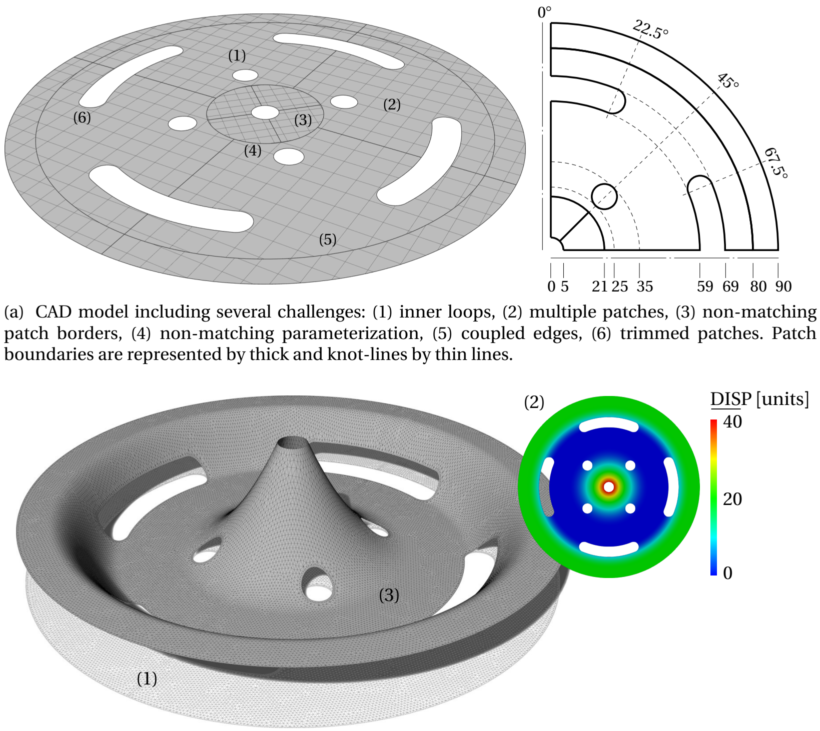
Within each iteration of the algorithm, a subset of possible nearest elements is determined first to avoid a projection of each integration point on every element. This is done using an established tree-based search technique. The subsequent orthogonal projection assumes planar elements and is computed explicitly using basic (linear) geometric operations. In the case of non-planar elements, a local approximation plane is utilized. The nearest element is determined based on the resulting distances. If the projection on the nearest element lays outside the element, e.g., because the projection relied on an approximation plane, then sub-routines are applied to determine the closest point within the element. The overall algorithm only employs efficient search techniques and basic geometric operations, making it very robust also with most complicated models.

4.4.4 Example

In this section, the basic mapping operation shall be tested. For this purpose, a test case is created, which is simple to reproduce, yet includes some of the most challenging features to be expected from practical CAD models. Figure 4.8 shows the corresponding setup. The test case is the following denoted as "test case 1".

The benchmark geometry is a flat circular plate with rotationally symmetric holes and cutouts, see figure 4.8(a). The corresponding CAD model is constructed by 12 trimmed NURBS patches of different polynomial orders ($p = q = 3...5$). Each patch is discretized by a grid of 17 × 17 knots leading to a total of 3 468 control points. The challenges with this model are the multiple patches with non-matching borders, the existing trimming, and the resulting coupling edges, which connect individual faces with different parameterizations. Also, the B-Rep description of the model contains four inner boundary loops representing the holes within the four large faces. Figure 4.8(a) labels and locates some of the challenges.

From the NURBS surface, a finite element mesh was generated, which includes 28 257 nodes and 54 064 linear triangles. For the mesh, a displacement field is assumed which turns the originally flat plate into a three-dimensional structure. The displacement is chosen not to affect the symmetry of the model, yet to cause a significant, non-uniform change of shape. The finite element mesh and the resulting deformation are illustrated in figure 4.8(b).



(a) CAD model including several challenges: (1) inner loops, (2) multiple patches, (3) non-matching patch borders, (4) non-matching parameterization, (5) coupled edges, (6) trimmed patches. Patch boundaries are represented by thick and knot-lines by thin lines.

(b) Mesh data: (1) initial mesh, (2) absolute displacement, (3) deformed mesh.

Figure 4.8: Test case 1

The goal in this test case is to deform the given CAD model such that it matches the deformed finite element mesh. To reach this goal, we use the previously introduced mapping and map the displacement field from the mesh onto the CAD model. Therefore, we factorize the mapping system from (4.32) and solve for the unknown displacement of the control points. As the system is fully linear in this case, only one solution iteration is required. The results of this test are compiled in figure 4.9.

Looking at the results and comparing them qualitatively to the input from figure 4.8(b), one can see that the displacement field was successfully mapped onto the NURBS surface. Accordingly, the deformed NURBS sur-

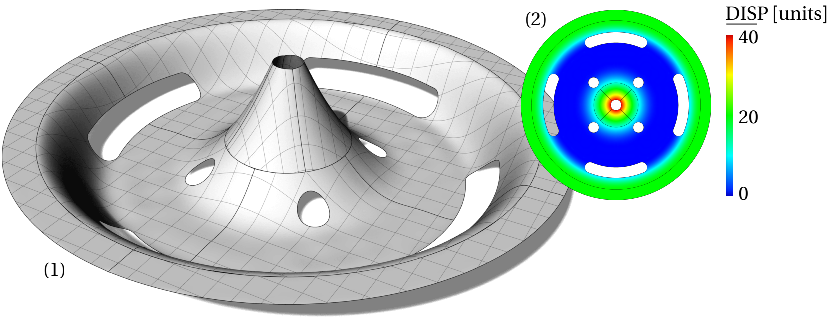


Figure 4.9: Results of the mapping in test case 1: (1) deformed NURBS surface, (2) mapped displacement field visualized on the deformed NURBS surface.

Table 4.1: Quantification of the mapping error. i denotes the index of a single Gauss point. Diameter of structure for reference: $\varnothing$180 units.

quality indicator	value
$\max\left(\lVert\Delta\mathbf{S}_i - \Delta\mathbf{Q}_i\rVert_2\right)$ [units]	2.32×10^{-1}
$\text{mean}\left(\lVert\Delta\mathbf{S}_i - \Delta\mathbf{Q}_i\rVert_2\right)$ [units]	3.47×10^{-2}

face represents the deformed finite element mesh. A quantitative evaluation of the mapping results confirms the visual impression of a successful map. Table 4.1 presents the remaining differences between the input displacement field $\Delta\mathbf{Q}$ and the mapped displacement field $\Delta\mathbf{S}$. The individual differences are evaluated at each Gauss point i after the mapping. As one can see in the table, the differences are negligible compared to the structure's dimensions. This proves that the deformed CAD model well matches the deformed mesh.

Figure 4.10 visualizes the Gauss points based on which the previous solution is computed. Note from the figure the very non-uniform distribution, especially along trimming curves. This is because, within trimmed surface elements, Gauss points are introduced based on a tessellation of the boundary (cf. figure 4.5). Since the tessellation is done using standard approaches, without considering the resulting integration points, one typically obtains an accumulation of integration points along trimmed boundaries. While

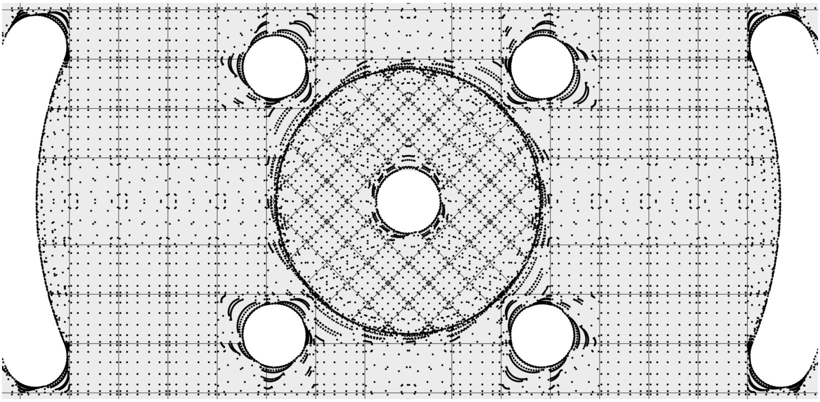


Figure 4.10: Visualization of Gauss points (black dots) around a section of the structure.

this type of integration is very robust and straightforward, it may also lead to unnecessary high computational costs in the factorization of the mapping system.

The size of the mapping system only depends on the number of unknown control point displacements. In the case of trimmed models, only those control points are relevant that affect the visible domain. All other points do not contribute to the mapping system and may be readily excluded. In the present case, for example, there are 3 468 control points, but only 1 876 of them are relevant. So the number of unknowns amounts to $1\,876 \times 3 = 5\,628$.

In the previous investigations, one could see that the mapping was successful, and the deformed NURBS surface principally matches the input. However, having a closer look at the surface and hiding all knot lines and edges, one can observe striking geometric discontinuities along initially coupled patch boundaries (see details in figure 4.11).

The geometric discontinuities originate from the discrete input data, which can cause a slightly different displacement of two adjacent faces. Moreover, the individual NURBS patches are parameterized differently without considering a later deformation. Therefore, they are by construction not able to represent the prescribed displacement with the same quality. As a consequence, small gaps and kinks appear. In general, gaps and kinks

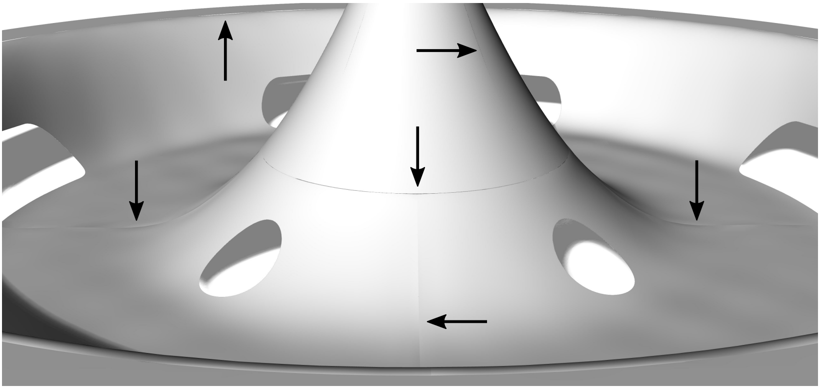


Figure 4.11: Geometric discontinuities (gaps and kinks) after deforming the NURBS model.

are unavoidable in the deformation of a multi-patch CAD model unless the parameterization of the individual patches matches in such a way that a continuous representation of the deformation is possible (either by construction or by chance).

Another critical aspect is the refinement of the NURBS model in comparison to the refinement of the mesh. In test case 1, the mesh has a high resolution and provides a smooth description of the input displacement or target geometry. On the other hand, the NURBS model has significantly fewer degrees of freedom yet enough to represent the prescribed displacement adequately. However, there are also cases where such a non-matching refinement leads to problematic results. Therefore, consider the case depicted in figure 4.12, which we in the following refer to as "test case 2".

Test case 2 contains the same structure with the same input displacement. However, it combines a very coarse mesh with an overly refined NURBS model. The NURBS model was refined through knot refinement such that each of the twelve patches is discretized by $50x50$knots leading to a total of 30 000 control points. The model topology and all polynomial degrees remain as before. The mesh was coarsened by regenerating a new mesh with only 1 417 nodes and 2 362 elements. Applying the mapping operation in this new setup, one obtains the results presented in figure 4.13.

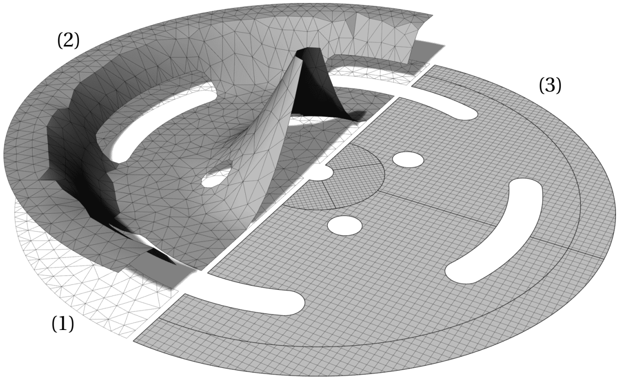


Figure 4.12: Test case 2: (1) initial finite element mesh, (2) deformed mesh, (3) initial CAD model.

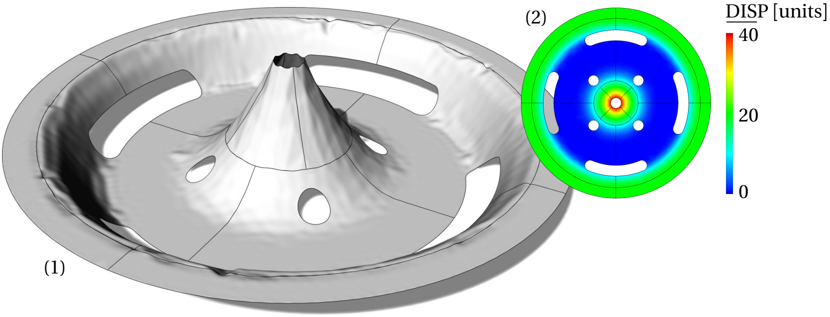


Figure 4.13: Result of the mapping in test case 2: (1) deformed NURBS surface (knot lines are hidden), (2) mapped displacement field visualized on the deformed NURBS surface.

From the figure, one can see that the overall displacement is again successfully mapped to the CAD domain. In this case, though, the coarse description of the finite element mesh in conjunction with the highly refined NURBS model causes the mesh to become visible in the deformed NURBS model (see the now occurring wrinkles on the surface). From a practical perspective, the wrinkled surface renders the result completely useless. Given the sparse input, such a result is all one can expect from a rigorous mapping. In order to still allow for a satisfying result, the mapping must be enhanced. This motivates the application of smoothing techniques, as introduced later.

From the previous investigations, one can summarize:

- With the presented mapping operation, a given CAD model can be deformed such that it matches the displacement of a corresponding finite element mesh.
- Without specific treatment, the deformed CAD model most probably contains geometric discontinuities across coupled faces.
- The quality of the deformed CAD model heavily depends on the discretization of both models, whereas discrepancies can lead to severe surface defects.
- Note also that a successful mapping implies enough degrees of freedom in the CAD model. If this is not the case, further degrees of freedom must be introduced.

In the remainder of this chapter, we more and more extend the basic mapping operation to address the problems mentioned above and prepare the method for an application with practical CAD models.

4.5 Regularization

The mapping presented in the previous section represents an inverse problem, which may suffer from ill-conditioning. The practical impacts of such an ill-conditioning are increased computation times, errors in the solution, and results containing unbounded displacements of the control points. In the best case, the corresponding quality losses are only minor, and the displacement is still well captured in the deformed CAD model. In the worst case, the deformed model is not acceptable at all.

The ill-conditioning originates from the trimming. Around trimmed edges, there are control points that are far outside the visible domain but still have a small influence on the latter, Weiss et al. [53]. Outside the visible domain, though, there is no information about the mesh deformation so that the displacement of the corresponding control points is just weakly defined.

In those cases, the system matrix in (4.32) has minimal values on the main diagonal, which leads to numerical instabilities. As a result, the mapping gives rise to an unbounded displacement of the control points around trimmed edges in the deformed CAD model. The problem is known as *flying nodes* in IBRA, cf. Breitenberger et al. [82].

In order to overcome this problem, we extend the mapping system according to the regularization proposed in Brujic et al. [93]. Herein, the authors present an effective regularization technique in the context of NURBS fitting. Due to the close relation of NURBS fitting and B-Rep morphing, we adopt the underlying idea and adjust its formulation slightly to the case of B-Rep morphing.

In Brujic et al. [93], the authors suggest an extension of the original fitting problem by two additional criteria, which they call β- and α-regularization:

1. β-regularization: minimizes the resulting displacement of the control points,
2. α-regularization: minimizes the distance between the control points and the NURBS surface.

The first regularization explicitly limits the movement of the control points. The effect of the second regularization is more subtle. According to the authors, the "control points do approximate the surface and it seems natural to keep them as close to the surface as possible." Practically, the second regularization has a smoothing effect and hence provides a useful tool to cure surface defects. Another positive side effect of the second regularization is that any further (manual) modification of the surface is significantly more straightforward since the control points have a more predictable influence on the surface if they are close to it.

Both regularization methods are implemented by adding a quadratic penalty term to the original objective function. In the context of B-Rep morphing, the resulting compromise function reads:

$$\Pi = \Pi_M + \Pi_\alpha + \Pi_\beta = \Pi_M + \alpha C_\alpha + \beta C_\beta \tag{4.41}$$

In the previous equation, Π_M denotes the objective function of the original mapping problem, cf. (4.29). C_α and C_β are two penalty functions weighted by some associated penalty factors α and β. The individual penalty terms, as well as their impact on the deformed CAD model, are further elaborated in the following.

4.5.1 Beta-regularization

Following the idea of β-regularization, we add a penalty function to the mapping problem in order to minimize the resulting control point displacement. This approach corresponds to a Tikhonov or L_2-regularization. The additional term reads:

$$\Pi_\beta = \beta\, C_\beta = \frac{\beta}{2} \sum_{i=1}^{n} \Delta\mathbf{P}_i \cdot \Delta\mathbf{P}_i \tag{4.42}$$

in which n denotes the total number of control points. The corresponding left- and right-hand side contributions to the mapping system are obtained by computing first and second order derivatives of the previous equation w.r.t. the individual control point displacements:

$$b_r = -\partial_r\left(\Pi_\beta\right) = -\beta\, \Delta P_r \tag{4.43}$$

$$A_{rs} = \partial_{rs}\left(\Pi_\beta\right) = \beta\, \delta_{rs} \tag{4.44}$$

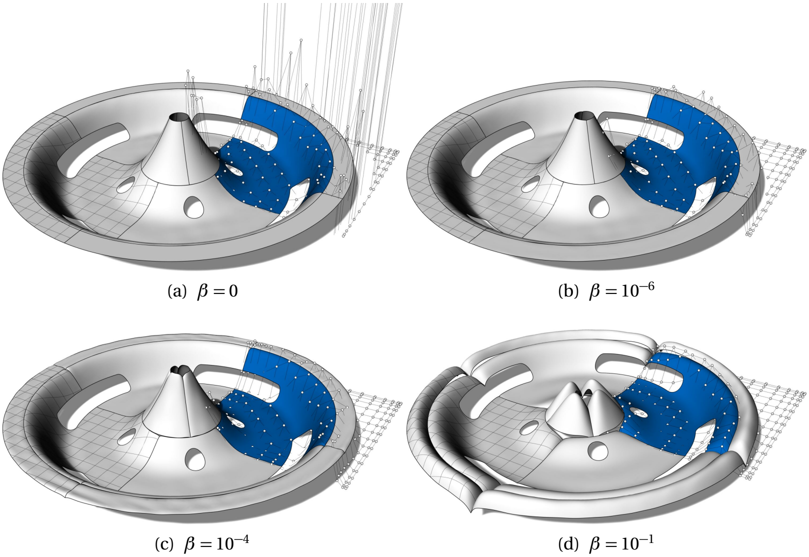
Herein, δ_{rs} denotes the Kronecker delta. So, β-regularization adds a single value to the main diagonal of **A**. Examining the definition of **A**, cf. (4.40), it can be seen that the unmodified system matrix only has positive entries. That is, any $\beta > 0$ will render the system stable.

Concerning the solution process, β-regularization may be realized by simply adding the values from (4.43) and (4.44) to the already assembled left- and right-hand side of the mapping system. The simple implementation yet effective stabilization makes β-regularization the method of choice to stabilize the mapping system.

However, the regularization also influences the original problem formulation and will, depending on the choice of β, damp the results. To demonstrate this effect, the regularization is applied to test case 1 from figure 4.8 with ascending values for β. The results are presented in table 4.2 and

Table 4.2: The effect of β on the conditioning of the mapping system.

β	0	10^{-9}	10^{-6}	10^{-5}	10^{-4}	10^{-1}
cond($\mathbf{A}$)	$> 10^{26}$	$\approx 10^{9}$	$\approx 10^{6}$	$\approx 10^{5}$	$\approx 10^{4}$	9.3
$\lVert \Delta \mathbf{P} \rVert_\infty$	$\approx 2 \times 10^{5}$	622.74	62.19	54.09	52.77	29.19
Figure	4.14(a)	-	4.14(b)	-	4.14(c)	4.14(d)


(a) $\beta = 0$

(b) $\beta = 10^{-6}$

(c) $\beta = 10^{-4}$

(d) $\beta = 10^{-1}$

Figure 4.14: B-Rep morphing applied to the test case from figure 4.8 using β-regularization and different values for β. In each case, the control points are shown for the blue patch, and the knot density of the NURBS surface is indicated in two other patches.

figure 4.14. The table collects the maximum absolute displacement of the control points in each case. It also shows each the condition number of the mapping matrix. The latter is determined as the quotient of its maximum and minimum singular value. Figure 4.14 visualizes the obtained CAD geometry for selected values of β.

From the figure, one can see that without regularization, the deformation results in an unbounded movement of the control points around the trimmed edges. From the table, one can see that this is a consequence of a severe ill-conditioning of the system. Indeed, the displacement was still successfully mapped to the NURBS surface in this case. Generally, however, the ill-conditioning may cause a complete failure of the mapping. In any case, the control points' unbounded motion is problematic because further handling of the deformed CAD model may become impossible by that. Therefore, β-regularization is inevitable.

β-regularization can successfully solve this problem. As one can see from table 4.2, with increasing value for β, the conditioning of the system continuously improves. Simultaneously, the maximum absolute displacement of the control points reduces from a value that is purely driven by numerical errors down to a value range similar to the problem dimensions. As a result, the control points do not show an unbounded movement but remain tractable (compare figure 4.14(a) and 4.14(b)). Striking is that the displacement quickly reaches acceptable values, whereas the condition number just slowly improves.

The ill-conditioning is due to control points with only little influence on the visible surface. The impact can be observed in figure 4.14(a). Herein, one can see that only control points, which are beyond the trimmed border of the highlighted surface, show an uncontrolled movement (the farther away, the larger the displacement). Note that already a small value for β seems enough to keep them under control, cf. figure 4.14(b).

From table 4.2, one also finds that β must neither be too small nor too high. In the first case, the effect of the regularization is too little, and control points may still show a critical displacement. Contrarily, a too strong penalization of the displacements causes significant damping of the geometry. The higher the values for β, the more the regularization causes such an unwanted deviation from the actual target geometry (see figure 4.14(d)).

The penalization is most influential in areas where control points obtain only marginal information from the mesh. This is the case around trimmed boundaries. Consequently, the geometric deviations resulting from the regularization are especially pronounced there. As a result, one may still observe deviations around trimming edges, despite a small value of β and an overall proper deformation of the CAD model, see figure 4.14(c). Of course, this effect reduces with decreasing β. Nevertheless, because of the nature of the L_2-regularization, it is observed to some extent for any $\beta > 0$. Note in any case that the regularization tends to generate discontinuities between adjacent surfaces, cf. figure 4.14(d).

The requirement to balance the penalty factor β naturally leads to the question of how to estimate its value a priori. To enable a qualified estimate, the authors of Brujic et al. [93] scale the penalty term before estimating β. The scaling is chosen such that the values in (4.44) are comparable to those in the original system matrix **A**. Based on the scaled values, they choose $\beta \ll 1$.

The same approach is utilized in the present work but with a slightly different scaling. Details on the applied scaling are elaborated in section 4.5.3. Based on this scaling, a good first guess for β is a value in the range of 10^{-9} to 10^{-5}.

4.5.2 Alpha-regularization

In Brujic et al. [93], the authors suggest combining β-regularization with a further regularization, which they refer to as the α-regularization. In the paper, α-regularization is introduced to smooth the NURBS surface and so remedy surface oscillations coming from insufficient input data. Herein, the α-regularization is adopted to avoid surface defects originating from the trimming or the discrete nature of the mesh.

Following the idea of α-regularization, we add a penalty term to the mapping problem. Brujic et al. [93] found that keeping the control points close to the surface has a smoothing effect when fitting NURBS geometries. Accordingly, the additional term penalizes the distance between the control points and the NURBS surface. As a result, control points tend to stay close to the surface or move towards it. The corresponding penalty term in the context of B-Rep morphing reads:

$$\Pi_\alpha = \alpha C_\alpha = \frac{\alpha}{2} \sum_{i=1}^{n} \left[\mathbf{P}_i - \mathbf{S}_i(\xi^*, \eta^*) \right] \cdot \left[\mathbf{P}_i - \mathbf{S}_i(\xi^*, \eta^*) \right] \tag{4.45}$$

The equation evaluates the sum of squared distances between n individual control points $\mathbf{P}_i$ and their associated points on the surface $\mathbf{S}_i$. To determine the latter, we use the Greville abscissae. The Greville abscissae assign each control point a parameter location on the surface, (ξ^*, η^*), "at which the control point exercises the most influence," Brujic et al. [93]. For more information about the Greville abscissae, the interested reader is referred to Farin [94].

As described in section 4.4.2, this work utilizes a finite-element-based approach in the solution of the underlying mapping problem. In order to include α-regularization into the solution process, point elements are formulated and created for each $\mathbf{S}_i$. All point elements are added to the overall list of elements and provided to the assembler to evaluate and add their local left- and right-hand side contributions to the mapping system in (4.32). The local contributions are obtained by computing first- and second-order derivatives of (4.45) w.r.t the control point displacements. Considering the definition of a surface point in (4.3a), the derivatives for each point element are defined as follows:

$$b_r = -\partial_r(\Pi_\alpha) = -\alpha[\mathbf{I} - \partial_r(\mathbf{S}_i)][\mathbf{P}_i - \mathbf{S}_i] \tag{4.46}$$

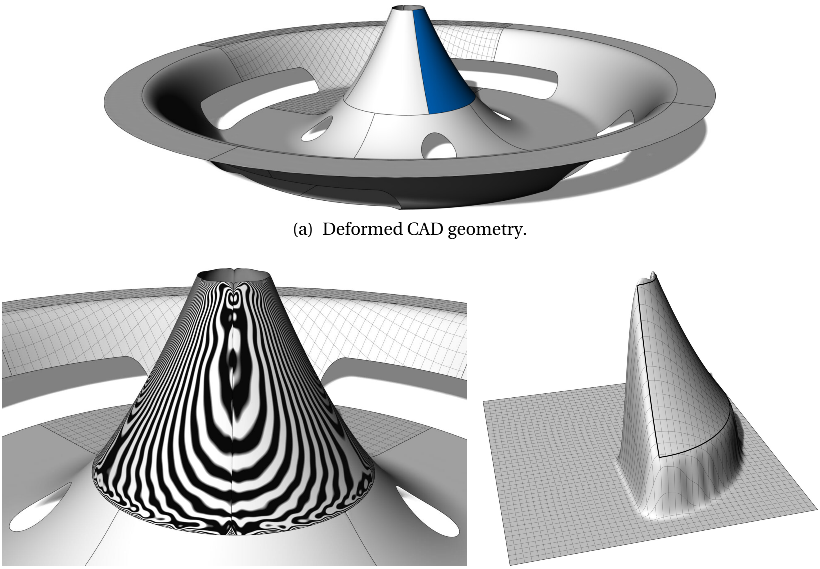
$$A_{rs} = \partial_{rs}(\Pi_\alpha) = \alpha[\mathbf{I} - \partial_r(\mathbf{S}_i)] \cdot [\mathbf{I} - \partial_s(\mathbf{S}_i)] \tag{4.47}$$

Note that (4.47) may be singular so that the α-regularization alone not necessarily stabilizes the mapping system. Therefore, we always use it in combination with the β-regularization from the previous section.

In order to allow for a qualified choice of α, the corresponding penalty term is scaled as described in section 4.5.3. α is then chosen relative to the scaled term. The effect of the α-regularization is illustrated in the following.

Consider the very fine mesh of test case 1 (see figure 4.8(b)), its displacement field , and the highly refined CAD model of test case 2 (see figure 4.12). Given this combination from both test cases, the CAD model shall be deformed such that it matches the deformed mesh. Therefore, B-Rep morphing is applied, whereas β-regularization is included to improve the system conditioning ($\beta = 10^{-6}$). First, α-regularization is deactivated. The corresponding results are presented in figure 4.15.

As can be seen from the figure, B-Rep morphing indeed leads to an overall meaningful update of the NURBS model, see figure 4.15(a). However, having a closer look at the surface, one observes a striking local formation of wrinkles most noticeably around the trimmed edges. The wrinkles



(a) Deformed CAD geometry.

(b) Detail view with partially visualized reflection pattern.

(c) Untrimmed patch of the blue surface (including knot lines and trimming curve).

Figure 4.15: Test case in which B-Rep morphing leads to wrinkles due to sharp surface gradients around the trimmed edges. The test case uses the mesh data from test case 1 (see figure 4.8(b)) and the CAD geometry from test case 2 (cf. figure 4.12).

are hardly visible at first glance, but become especially prominent when looking at the central dome's reflection pattern, see figure 4.15(b).

The wrinkles arise because of the high knot refinement of the individual NURBS patches in conjunction with the specified β-regularization. The high knot refinement causes in each patch a steep gradient between the deformed surface in the visible domain and the undeformed surface in the trimmed domain. Contrarily, the rectangular parameter grid in this transition zone only provides a limited resolution of such a gradient, which eventually induced wrinkles. Moreover, β-regularization damps the movement of the control points in this transition zone so that it tends to amplify the wrinkling. The problem becomes apparent when isolating a patch from the overall geometry and removing its trimming, as done in figure

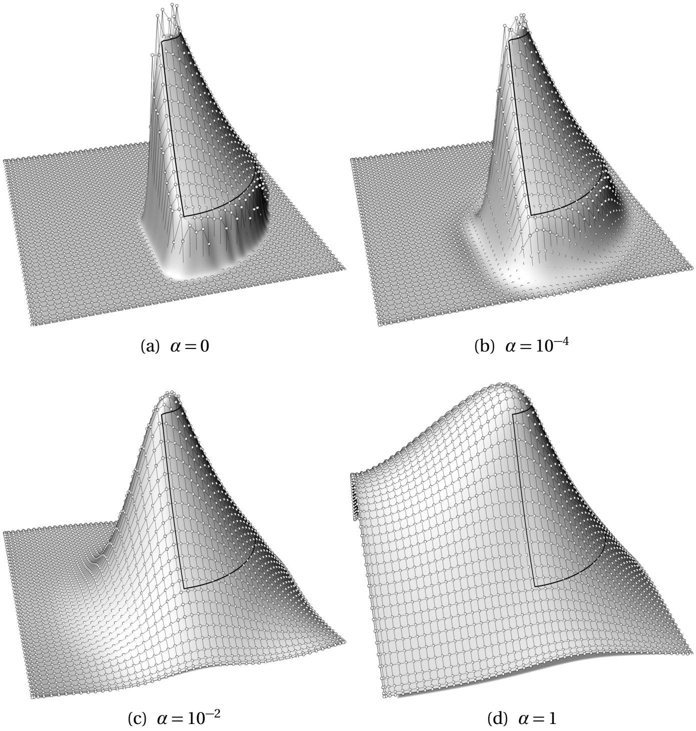
4.15(c). Herein, one can see the sharp transition between the deformed visible part of the patch and the undeformed part outside the trimming curve.

The example reveals an actual problem in the deformation of highly trimmed B-Rep models, i.e., data about the displacement is only given for the visible area. In contrast, the invisible domain suffers from insufficient data support. As a result, the deformed CAD model may show significant surface defects around trimmed boundaries, which, in the worst case, can radiate over large parts of the domain. An effective way to avoid such defects is to enforce a gentle transition between the trimmed and the visible domain. α-regularization provides a useful tool for this purpose.

To demonstrate the effect of α-regularization, we recompute the last test case with $\alpha \neq 0$. Figure 4.16 shows the effect of an increasing α on the patch from figure 4.15(c). From the results, one can observe that the higher α, the more the control points approximate the surface. This behavior improves the transition between the deformed visible surface and the unchanged trimmed domain. Interesting to see is that the adjustments coming from the α-regularization mostly apply to the trimmed region where control points tend to be further away from the surface. The general shape change is still well captured in the visible area. The authors in Brujic et al. [93] point out that α-regularization is most effective in areas with a sparse data basis. The results in this section confirm this effect also for the area around trimmed edges.

The complete CAD geometry, as it is obtained in the case of $\alpha = 10^{-2}$, is presented in figure 4.17. As one can see in the figure, the input displacement is well captured, much like before, without α-regularization. This time, however, the wrinkles are completely suppressed, improving the overall surface quality significantly. Therefore, compare the reflection patterns in figure 4.17(b) and 4.15(b). Also, the layout of the control points improved from a locally fluctuating pattern, cf. figure 4.16(a), to a globally smooth and uniform one, cf. 4.16(c).

In the previous results, we saw that α-regularization mostly affects the area around the trimmed boundaries. Whereas this feature is advantageous to cure surface defects or improve the control point layout, it also tends to amplify boundary discontinuities across coupled faces. This amplification is a characteristic which the α-regularization shares with the β-regularization. Therefore, when applying either of the regularization



(a) $\alpha = 0$

(b) $\alpha = 10^{-4}$

(c) $\alpha = 10^{-2}$

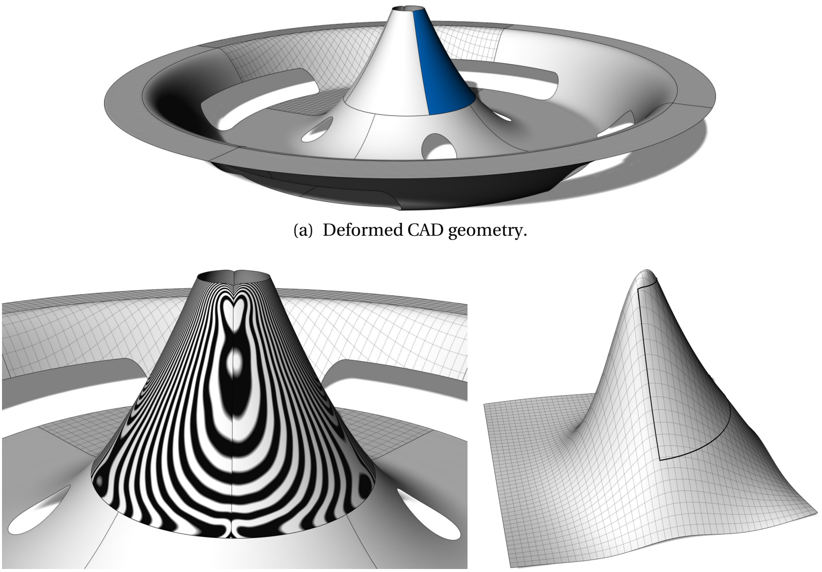
(d) $\alpha = 1$

Figure 4.16: Untrimmed patch from figure 4.15(c) when applying α-regularization with different values for α.

techniques, particular attention must be paid to the surface continuity.

Note also that the effect of the α-regularization is not limited to trimmed boundaries. It just has its most significant influence there. In fact, with increasing values for α, the smoothing effect more and more extends to the entire surface. Therefore, one can use α-regularization generally to treat possible surface defects.

To demonstrate this feature, consider again test case 2 from figure 4.12. In this case, the coarse reference mesh caused a wrinkling in the deformed CAD geometry, see figure 4.18(a). The wrinkles lead to an increased deviation of the control points from the surface. Hence α-regularization is



(a) Deformed CAD geometry.

(b) Detail view with partially visualized reflection pattern.

(c) Untrimmed patch of the blue surface (including knots lines & trimming curve).

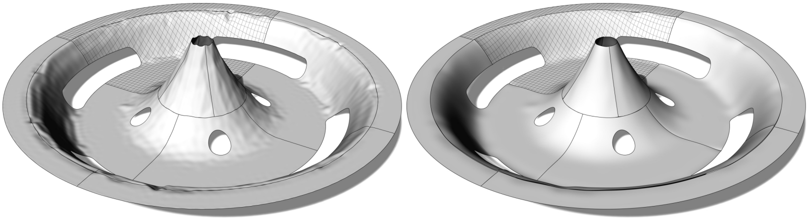
Figure 4.17: Rerun of test case from figure 4.15 with activated α-regularization ($\alpha = 10^{-2}$).

expected to improve the surface quality.

In fact, rerunning the case with active regularization, whereas $\beta = 10^{-6}$ and $\alpha = 5$, the surface quality can be significantly improved, see figure 4.18(b). Note that we assigned α with a rather large value of bigger than one. This value prioritizes the smoothing over the mapping requirements. Choosing α to be just slightly above one prevents the smoothing from being overly dominant so that the overall displacement field is still well captured in the deformed CAD geometry[1]. Choosing a too high value causes the CAD geometry to deviate from the input significantly.

At the end of this section one can conclude that α-regularization has two effects:

[1] Refer to section 4.5.3 regarding how to interpret the value of α.



(a) $\alpha = 0$ (Corresponds to fig. 4.13)

(b) $\alpha = 5$

Figure 4.18: Mapping of a displacement field from a coarse finite element mesh to a highly resolved CAD geometry with and without α-regularization.

1. It keeps the control points close to the surface, which generates a practical control point layout and positively impacts the surface quality around trimmed edges.

2. It allows for a general surface smoothing if any defects occur.

The first effect is already obtained with a relatively small value for α. The second effect needs higher values, whereby the α-regularization also impacts the mapping results. A general recommendation for its application in B-Rep morphing is: First, try a solution with $\alpha \ll 1$, say 10^{-2}. Only if a general surface smoothing is intended, raise the value to $\alpha > 1$. Both values are recommended based on the scaling introduced in section 4.5.3. In case the CAD model only includes little or no trimming, α-regularization might not be necessary at all.

4.5.3 Scaling of the regularization terms

Both α- and β- regularization introduce assumptions on the solution. Their effect has to be balanced against the mapping requirements by a proper choice of the penalty factors. To allow for a proper choice of the penalty factors, we require reference values. Therefore, the individual penalty terms, Π_α and Π_β, are scaled to the same dimension as the objective functional, Π_m. The same dimension means that, for $\alpha = \beta = 1$, the values which a regularization term adds to the mapping matrix are comparable to the values within the mapping matrix:

$$\mathit{\Pi}_\alpha = \alpha \left\| \frac{\partial^2 \mathit{\Pi}_M}{\partial \Delta \mathbf{P}^2} \right\|_\infty \left\| \frac{\partial^2 C_\alpha}{\partial \Delta \mathbf{P}^2} \right\|_\infty^{-1} C_\alpha \tag{4.48}$$

$$\mathit{\Pi}_\beta = \beta \left\| \frac{\partial^2 \mathit{\Pi}_M}{\partial \Delta \mathbf{P}^2} \right\|_\infty \left\| \frac{\partial^2 C_\beta}{\partial \Delta \mathbf{P}^2} \right\|_\infty^{-1} C_\beta \tag{4.49}$$

In the previous equations, $\left(\partial^2 \mathit{\Pi}_M / \partial \Delta \mathbf{P}^2\right)$ corresponds the initial mapping matrix, **A**. $\left(\partial^2 C_\alpha / \partial \Delta \mathbf{P}^2\right)$ and $\left(\partial^2 C_\beta / \partial \Delta \mathbf{P}^2\right)$ are the contributions of the regularization terms to the mapping matrix, without the influence of the penalty factors. Both expressions can be computed according to (4.44) and (4.47) for $\alpha = \beta = 1$. Note that for a determination of the scaling factor, one first needs to compute the contributions of the regularization terms to the mapping matrix. Only then, the values can be scaled and added to the mapping matrix.

Once the penalty terms are scaled, the penalty factors may be selected, starting from a reference value of 1. A penalty factor of 0.1 then means, for example, that the influence of the regularization on the solution is ten times lower than the influence of the mapping requirements. In this case, the latter would still be dominant. Vice versa, a penalty factor higher than 1 prioritizes the regularization compared to the mapping. Meaning, a value higher than 1 can cause the deformed CAD model to deviate from the prescribed displacement field significantly.

Note that the scaling is also necessary because the system matrix **A** involves an integral, meaning it has different value ranges for different model dimensions. In contrast, the β-regularization is invariant to the model dimensions, and the α-regularization does not involve an integration. Clearly, without any scaling, the penalty factors would be heavily case-dependent. With scaling, though, the herein specified values for α may serve as reference values.

4.6 Constraints

The previous sections introduced the underlying mapping operation in B-Rep morphing. The operation relies on a least-squares approach, requiring matching displacement fields in the CAD model and the mesh. Other than that, we did not consider any further constraints so far. However, in practical CAD models involving multiple patches and trimming, constraints may quickly become a necessity. Two reasons for this necessity are:

First: The pure mapping of the displacement field does not deliver the desired surface quality in the deformed CAD model, especially in terms of smoothness and continuity. In the case of multi-patch and trimmed NURBS geometries, the requirement of preserving smooth and continuous surfaces[2] often has a high priority since otherwise, the CAD model would be useless. Without specific quality constraints, the mapping operation does not provide any handle over such requirements.

Second: The pure mapping of the displacement field may result in a deformed CAD model, which violates specific design requirements. For example, in a CAD workflow, the designer might only use the deformed CAD geometry if certain geometrical constraints are satisfied. Such a constraint could be, e.g., a fixed model interface, which must not be altered. Using specific design constraints, we may locally sacrifice accuracy in the mapping for the sake of a usable model.

This section introduces a way to include the mentioned types of constraints into the original mapping problem. Also, some specific constraints are discussed.

4.6.1 Treatment of constraints

In this thesis, we choose a simple penalty approach to introduce constraints to the mapping problem. In the context of constrained reverse engineering, the penalty approach is an established and wide-spread technique, cf. Fisher [49] and Benko et al. [95], for example. Following this approach, the original mapping problem is augmented by penalty terms, which in the present case are formulated in an integral (weak) form:

[2] at least a watertight surface

$$\begin{aligned}\Pi &= \Pi_M + \sum_{j=1}^{n} \Pi_{\mathbf{S},j} + \sum_{k=1}^{m} \Pi_{\partial\mathbf{S},k} \\ &= \Pi_M + \sum_{j=1}^{n} p_j C_{\mathbf{S},j} + \sum_{k=1}^{m} p_k C_{\partial\mathbf{S},k}\end{aligned} \tag{4.50}$$

where:

$$C_{\mathbf{S}} = \int_{\mathbf{S}} \dots \, \mathrm{d}\mathbf{S}, \qquad C_{\partial\mathbf{S}} = \int_{\partial\mathbf{S}} \dots \, \mathrm{d}\partial\mathbf{S} \tag{4.51}$$

Herein, surface constraints $C_{\mathbf{S}}$ are distinguished from constraints over edges of the B-Rep model, $C_{\partial\mathbf{S}}$. Each penalty function is assigned with a separate penalty factor, p. It is possible to have several constraints over different surfaces and edges, indicated by the summation over the indices j and k, respectively. Π_M denotes the original mapping problem, (4.29).

The advantages of such an approach are:

1. There are no additional unknowns (no Lagrange multipliers),
2. the weak form allows a formulation of constraints over arbitrarily trimmed surfaces and edges,
3. assuming the penalty term is non-negative (e.g., a quadratic functional), it will not affect the system stability (β-regularization is still enough for stabilization),
4. the penalty factors enable a relative weighting of the individual constraint so that its influence may be controlled.

Moreover, the resulting problem formulation is very similar to the problem formulation in IBRA, where the penalty approach is already applied successfully in conjunction with multi-patch CAD models and trimmed NURBS surfaces, cf. Breitenberger et al. [82]. In the context of IBRA, alternative techniques to apply constraints are discussed in Apostolatos et al. [96].

To integrate constraints into the solution of the mapping problem, we follow the finite element approach introduced in section 4.4.2. Accordingly, constrained surfaces and edges are discretized into elements, whereas each element computes its contribution to the mapping system by numerical integration. The local contributions are then assembled to the global mapping system. The discretization of constrained surfaces and the corresponding numerical integration follow the procedure described in section 4.3.1. In the case of constrained edges, the logic is described in section 4.3.2. The local left- and right-hand side contributions of constraints are computed as follows:

$$b_r = \begin{cases} -\partial_r(\Pi_{\mathbf{S}}) \\ -\partial_r(\Pi_{\partial\mathbf{S}}) \end{cases} \qquad A_{rs} = \begin{cases} \partial^2_{rs}(\Pi_{\mathbf{S}}) \\ \partial^2_{rs}(\Pi_{\partial\mathbf{S}}) \end{cases} \tag{4.52}$$

Note that both surface and edge constraints are generally referring to the same set of control points. That is because boundary or trimming curves, which make up an edge, are embedded into the surface description, cf. (4.7). Therefore, constraints over individual edges do not introduce any additional unknowns.

Note also that the surface integral in (4.51) must not necessarily extend over the entire surface of a patch. The surface integration presented in section 4.3 allows considering any embedded area. The latter only has to be bounded by a curve on the patch. Because of this freedom in the definition, one can formulate constraints for an arbitrary part of a surface, cf. figure 4.19 for an illustration. Similarly, one can define constraints over arbitrary lines along the surface in the CAD model. In both cases, the solution process does not change due to the systematic treatment of constraints using the finite element approach. The possibility to also include constraints for embedded areas is a handy feature of B-Rep morphing in practical applications.

A disadvantage of the penalty method is that a penalty factor has to be estimated. By definition, the constraints may only be satisfied if their penalty factors tend to infinity. However, for numerical reasons, a finite but sufficiently high value has to be chosen. This calls for proper reference values. Constraint scaling is applied to allow for a qualified estimate in this context. In favor of a uniform systematic, the scaling is done analogously to the

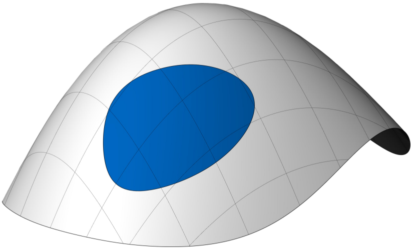


Figure 4.19: Embedded area (blue) within a trimmed NURBS surface.

scaling in the regularization, cf. section 4.5.3. That is, we scale additional penalty terms from a surface or an edge constraint to the same dimension as the objective functional, Π_m. The same dimension means that, for a penalty factor of $p = 1$, the values which a constraint adds to the mapping matrix are comparable to the values within the mapping matrix:

$$\Pi_{\mathbf{S}} = p \left\| \frac{\partial^2 \Pi_M}{\partial \Delta \mathbf{P}^2} \right\|_\infty \left\| \frac{\partial^2 C_{\mathbf{S}}}{\partial \Delta \mathbf{P}^2} \right\|_\infty^{-1} C_{\mathbf{S}} \tag{4.53}$$

$$\Pi_{\partial\mathbf{S}} = p \left\| \frac{\partial^2 \Pi_M}{\partial \Delta \mathbf{P}^2} \right\|_\infty \left\| \frac{\partial^2 C_{\partial\mathbf{S}}}{\partial \Delta \mathbf{P}^2} \right\|_\infty^{-1} C_{\partial\mathbf{S}} \tag{4.54}$$

Herein, $\left(\partial^2 \Pi_M / \partial \Delta \mathbf{P}^2\right) = \mathbf{A}$ and is computed according to (4.40). The computation of $\left\|\partial^2 C_{\mathbf{S}} / \partial \Delta \mathbf{P}^2\right\|_\infty$ and $\left\|\partial^2 C_{\partial\mathbf{S}} / \partial \Delta \mathbf{P}^2\right\|_\infty$ depends on the definition of the individual constraints. Some examples in this context are described in the following sections.

After scaling, one can assume that the original least-squares objective and the individual constraints are weighted approximately equally. On this basis, one can choose a penalty factor according to the importance of the associated constraint relative to the mapping requirements or other constraints. For example, fulfilling a local design constraint may be more important than an exact map in the same area, or patch continuity may be more important than other design constraints. Assuming comparable objective and constraint values, a good initial guess for penalty factors is $p = 10^3$.

Another aspect in the treatment of constraints concerns the computation of the necessary first- and second-order derivatives, cf. (4.52). Depending on the formulation of the constraint, an analytic computation of those derivatives can be elaborate or tedious. Therefore, we utilize algorithmic differentiation (AD) in the present work. One of the most significant advantages of AD is that one can compute derivatives without explicitly formulating them.

The AD used in this work was developed by Oberbichler et al. [97] and is based on hyper-dual numbers as described in Fike et al. [98] and Fike [99]. The implementation is taken from Oberbichler [100]. It uses operator overloading in C++ and allows the computation of numerically exact first- and second-order derivatives. "Numerically exact" refers to the fact that the method does not suffer from cancellation errors.

Oberbichler et al. [97] use AD for the element formulation based on energy functionals in the context of a finite element approach. We in this work adopt their idea of using AD on the element level. Accordingly, we do not use AD to determine the derivatives of the global least-squares (mapping) problem. Instead, we apply AD for some constraints locally within the scope of an element to compute its left- and right-hand side contribution to the mapping system. This hybrid approach allows exploiting the advantages of AD without modifying the general solution process. Also, the computational overhead coming from AD remains small.

Finally, it is worthwhile to mention that the finite-element-based solution approach used in this thesis allows for the systematic development of new constraints. To include a new constraint, one only has to define a functional and implement a new element type, which computes its local system contributions as in (4.52). The general assembly and solution process remains as is. If AD is used, the required derivatives can even be computed without the need to formulate them explicitly. We found the combination with AD to be very beneficial as it allows for rapid development and testing of new constraints without spending much time getting derivatives, which are required to include the constraint in the first place.

4.6.2 Coupling constraints

When applying B-Rep morphing to continuous trimmed multi-patch geometries, discontinuities have to be expected across patch borders since:

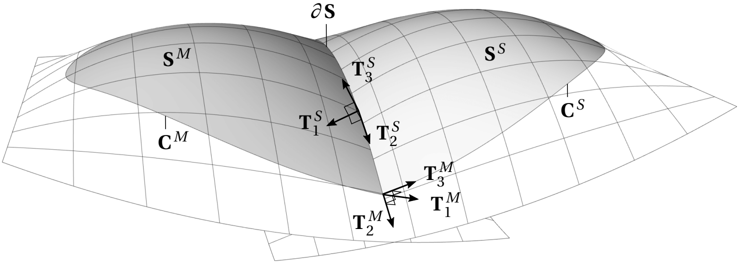


Figure 4.20: Two trimmed and coupled NURBS patches.

1. adjacent NURBS patches can be parameterized differently,
2. the input finite element mesh is most probably not uniform, especially around coupled edges,
3. the previously introduced regularization strongly affects the surface around trimmed edges.

As a possibility to maintain the continuity properties of the initial CAD model as well as possible, this thesis suggests adding coupling constraints to the mapping problem. In this context, we consider displacement and rotation coupling. The corresponding constraints are adopted from Breitenberger et al. [82], where the authors use them for the mechanical analysis on CAD models (IBRA).

Coupling constraints evaluate the continuity across a coupling edge. The latter is assumed to connect two different, possibly trimmed patches. In case several patches are attached to a single edge, we each form pairs of two patches. Among the two coupled patches, one distinguishes a master and a slave side, $\mathbf{S}^M$ and $\mathbf{S}^S$. Accordingly, the edge is represented locally by a master and a slave boundary curve, $\mathbf{C}^M$ and $\mathbf{C}^S$. Both curves can be trimming curves. Figure 4.20 illustrates two trimmed patches, which are coupled.

Displacement coupling

To maintain positional continuity over a coupled edge, the displacement of the master patch has to match the displacement of the slave patch over the entire coupling edge $\partial\mathbf{S}$, cf. figure 4.20. Following the penalty approach introduced in section 4.6.1, the corresponding penalty term reads:

$$\mathit{\Pi}_{\text{disp}} = \frac{p_{\text{disp}}}{2} \int_{\partial\mathbf{S}^{(0)}} \left[\mathit{\Delta}\mathbf{S}^M - \mathit{\Delta}\mathbf{S}^S\right] \cdot \left[\mathit{\Delta}\mathbf{S}^M - \mathit{\Delta}\mathbf{S}^S\right] \,\mathrm{d}\partial\mathbf{S}^{(0)} \tag{4.55}$$

where p_{disp} represents a dedicated penalty factor. The integration is always done over the initial configuration of the coupling edge, $\partial\mathbf{S}^{(0)}$, so that no variation of the integration domain has to be considered later. The integration domain locally corresponds to $\mathbf{C}^M$. Deriving (4.55) w.r.t. the control point displacements yields the left- and right-hand side contributions to the mapping system. Considering the definition of a displacement field over a NURBS surface, see (4.8), the contributions are:

$$\begin{aligned} b_r &= -\partial_r(\mathit{\Pi}_{\text{disp}}) \\ &= -p_{\text{disp}} \int_{\partial\mathbf{S}^{(0)}} \left[\partial_r(\mathit{\Delta}\mathbf{S}^M) - \partial_r(\mathit{\Delta}\mathbf{S}^S)\right] \cdot \left[\mathit{\Delta}\mathbf{S}^M - \mathit{\Delta}\mathbf{S}^S\right] \,\mathrm{d}\partial\mathbf{S}^{(0)} \end{aligned} \tag{4.56}$$

and

$$\begin{aligned} A_{rs} &= \partial^2_{rs}(\mathit{\Pi}_{\text{disp}}) \\ &= p_{\text{disp}} \int_{\partial\mathbf{S}^{(0)}} \left[\partial_r(\mathit{\Delta}\mathbf{S}^M) - \partial_r(\mathit{\Delta}\mathbf{S}^S)\right] \cdot \left[\partial_s(\mathit{\Delta}\mathbf{S}^M) - \partial_s(\mathit{\Delta}\mathbf{S}^S)\right] \,\mathrm{d}\partial\mathbf{S}^{(0)} \end{aligned} \tag{4.57}$$

Rotation coupling

If also the tangential continuity must be preserved, one can also couple the rotation of the master and the slave patch along the edge. The rotation of a patch at the edge is measured as the angular change of its local normal vector $\mathbf{T}_3$ around its local tangent vector $\mathbf{T}_2$ (cf. figure 4.20):

$$\omega_{\mathbf{T}_2} = \arcsin(\boldsymbol{\omega} \cdot \mathbf{T}_2) \tag{4.58}$$

Herein $\boldsymbol{\omega}$ is the rotation vector defined as follows (small letters indicate the deformed configuration):

$$\boldsymbol{\omega} = \mathbf{T}_3 \times (\mathbf{t}_3 - \mathbf{T}_3) \tag{4.59}$$

A coupling of rotations is obtained by enforcing a minimal difference between $\omega_{\mathbf{T}_2}$ on the master and the slave side. To obtain comparable rotations, $\mathbf{T}_2 = \mathbf{T}_2^M$ is chosen as common tangent vector [3]. Following the penalty approach introduced in section 4.6.1, the corresponding penalty term reads:

$$\Pi_{\text{rot}} = \frac{p_{\text{rot}}}{2} \int_{\partial \mathbf{S}^{(0)}} \left[\omega^M_{\mathbf{T}_2} - \omega^S_{\mathbf{T}_2}\right] \cdot \left[\omega^M_{\mathbf{T}_2} - \omega^S_{\mathbf{T}_2}\right] \, \mathrm{d}\partial \mathbf{S}^{(0)} \tag{4.60}$$

The integration domain is the same as for the displacement coupling. Deriving the previous equation w.r.t. the individual control point displacements, one obtains the corresponding left- and right-hand side contributions to the mapping system:

$$\begin{aligned} b_r &= -\partial_r(\Pi_{\text{rot}}) \\ &= -p_{\text{rot}} \int_{\partial \mathbf{S}^{(0)}} \left[\partial_r(\omega^M_{\mathbf{T}_2}) - \partial_r(\omega^S_{\mathbf{T}_2})\right] \cdot \left[\omega^M_{\mathbf{T}_2} - \omega^S_{\mathbf{T}_2}\right] \, \mathrm{d}\partial \mathbf{S}^{(0)} \end{aligned} \tag{4.61}$$

$$\begin{aligned} A_{rs} &= \partial^2_{rs}(\Pi_{\text{rot}}) \\ &= p_{\text{rot}} \int_{\partial \mathbf{S}^{(0)}} \left[\partial_r(\omega^M_{\mathbf{T}_2}) - \partial_r(\omega^S_{\mathbf{T}_2})\right] \cdot \left[\partial_s(\omega^M_{\mathbf{T}_2}) - \partial_s(\omega^S_{\mathbf{T}_2})\right] \\ &\quad + \left[\partial^2_{rs}(\omega^M_{\mathbf{T}_2}) - \partial^2_{rs}(\omega^S_{\mathbf{T}_2})\right] \cdot \left[\omega^M_{\mathbf{T}_2} - \omega^S_{\mathbf{T}_2}\right] \, \mathrm{d}\partial \mathbf{S}^{(0)} \end{aligned} \tag{4.62}$$

The computations in (4.61) and (4.62) requires first *and* second-order derivatives of $\omega_{\mathbf{T}_2}$. Their analytic derivation can be found in Breitenberger

[3] This is different to the formulation in Breitenberger et al. [82]. In that, the rotation is determined separately around the tangent vectors $\mathbf{T}_2^S$ and $\mathbf{T}_2^M$. However, the latter requires a case distinction regarding the orientations of both vectors

et al. [82]. Rather than an analytic computation, however, algorithmic differentiation is applied to determine the elemental system contributions because an analytic computation of $\mathbf{A}_i$ requires second-order derivatives of the local coordinate system, which were not accessible.

Important characteristics to be considered in the context of rotation coupling are:

- The rotation coupling renders the mapping problem nonlinear as $\mathbf{A}$ is not constant anymore.
- Because (4.58) includes the *arcsin*, rotations must not exceed 90°.
- The rotation coupling tends to *maintain* the relative orientation of two adjacent faces so that G^1-continuous transitions tend to stay G^1-continuous. Conversely, no higher continuity can be expected at kinks in the original geometry. Instead, the original kinks tend to be preserved. This is a very welcome feature as it allows us to keep a defined angle between adjacent faces so that, e.g., rectangular edges remain rectangular.

Effect of coupling constraints

In the following, we investigate the effect of coupling constraints. For this purpose, B-Rep morphing is applied to test case 1 from figure 4.8. Unlike in the original case, though, both α- and β-regularization are activated to keep the problem well-conditioned ($\alpha = 10^{-3}$, $\beta = 10^{-6}$).

Without any coupling constraints, B-Rep morphing results in a CAD geometry, which overall captures the displacement of the finite element mesh, but locally shows significant geometric discontinuities, cf. figures 4.9 and 4.11. These discontinuities render the deformed CAD model unusable and must be avoided. Figure 4.21 quantifies the discontinuities by evaluating $(\varDelta\mathbf{S}^M - \varDelta\mathbf{S}^S)$ and $(\omega^M_{\mathbf{T}_2} - \omega^S_{\mathbf{T}_2})$ around a representative internal coupling edge ranging 360° around the central dome (highlighted in the figure). The deviations are measured at the Gauss points of the later introduced coupling constraints so that the results are comparable.

As the figure suggests, if no coupling is applied, there are positional deviations of larger than 0.1 units. Compared to the radius of the circular geometry (90 units), this value seems small. However, 0.1 units are well

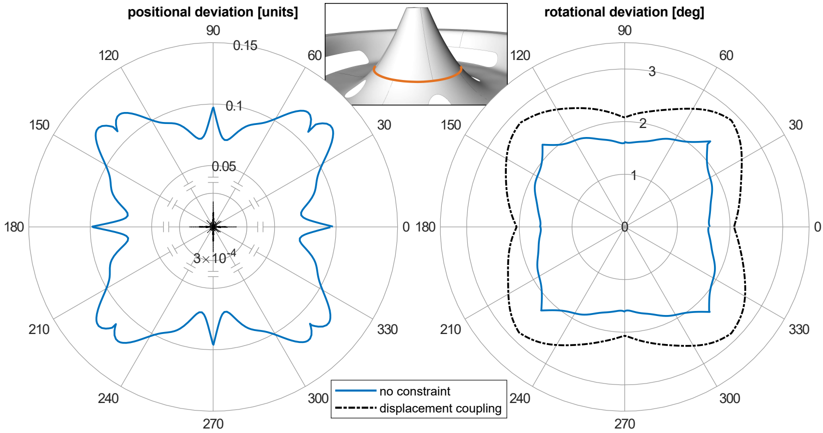


Figure 4.21: Influence of displacement coupling on positional and rotational deviation of adjacent faces around an internal edge (orange).

above the herein specified drawing tolerance (0.001 units) and hence cause a striking gap. As a consequence, the faces cannot be joined in the given CAD system without adjusting the tolerance. The rotational deviation is even more severe. Here, one observes differences of up to almost 3°, whereas peaks appear around corners of the trimmed patches. 3° is high enough to cause an apparent kink in the model, cf. figure 4.11.

Introducing displacement coupling according to (4.55) and specifying a penalty factor of $p_{\text{disp}} = 10^3$ after scaling the corresponding penalty term according (4.54), we can improve the positional deviation significantly, cf. figure 4.21. However, from the figure, it is also clear that displacement coupling alone is not enough to preserve the original G^1-continuity in the deformed CAD model. Rotational deviations remain. The graph shows that they are even amplified due to the modifications introduced by the displacement coupling. In order to preserve the original G^1-continuity, rotation coupling must be added.

Figure 4.22 compiles the results of B-Rep morphing after adding rotation coupling. Similarly as above the penalty term associated to the rotation coupling is scaled following the systematic in (4.54) and the corresponding

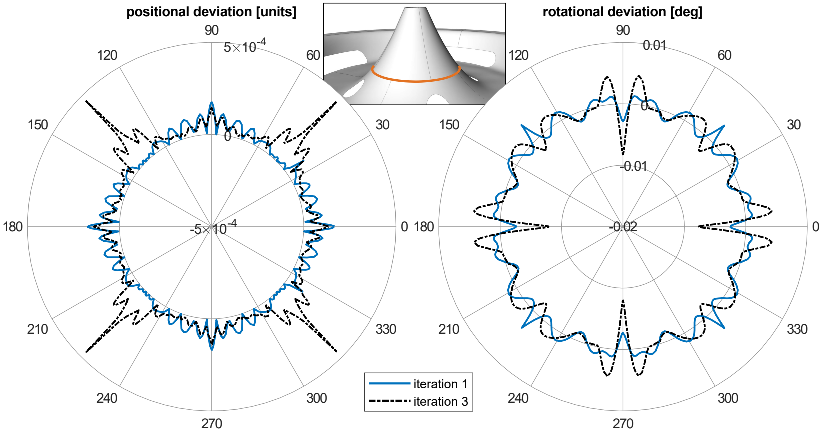


Figure 4.22: Positional and rotational deviation of adjacent faces around an internal edge (orange) with activated displacement and rotation coupling.

penalty factor is specified as $p_{\text{rot}} = 10^3$. The deviations are measured at the Gauss points of the coupling constraints. The results in the figure show that after three solution iterations, both the positional and rotational deviations drop by several orders. Especially the rotational deviation is now not more than 0.01°, so that apart from the watertight surface also G^1-continuity is maintained within practical limits.

An important insight from figure 4.22 is that rotation coupling is already effective in the first solution iteration. Meaning, no nonlinear solution iteration is necessary to maintain the original G^1-continuity within practical limits. The results from the third iteration show even higher maximum values, despite the residual in the solution continuously decrease. The reason for this behavior is two-fold and generally valid for all cases:

- The coupling is defined in an integral (weak) form over the B-Rep edge. That is, there is no control over peak values.
- The nonlinearity is due to the *difference* in the local rotation on both sides $(\omega^M_{\mathbf{T}_2} - \omega^S_{\mathbf{T}_2})$, cf. (4.62). This difference by definition is zero in the beginning and, if rotation coupling is applied, remains very small in

subsequent iterations. Consequently, a nonlinear solution iteration is often not necessary.

The quantitative improvement through the active coupling constraints can also be qualitatively observed in the deformed CAD geometry, see figure 4.23. The figure shows a flawless deformation in the CAD model. Note specifically that the surface quality is so good that one cannot identify the individual patches anymore. The corresponding reflection pattern in figure 4.23(b) confirms the high surface quality. It only shows continuous reflection stripes indicating a higher-order surface continuity [4]. Both figures emphasize the positive effect and the need for coupling constraints when fitting a multi-patch trimmed CAD model to a prescribed displacement field.

Note that we achieved a high surface quality in this example even though the individual patches have different parameterizations. In general, different parameterizations between patches may exclude a continuous surface in the deformed CAD model. In any case, coupling constraints are useful to preserve the surface continuity as much as possible.

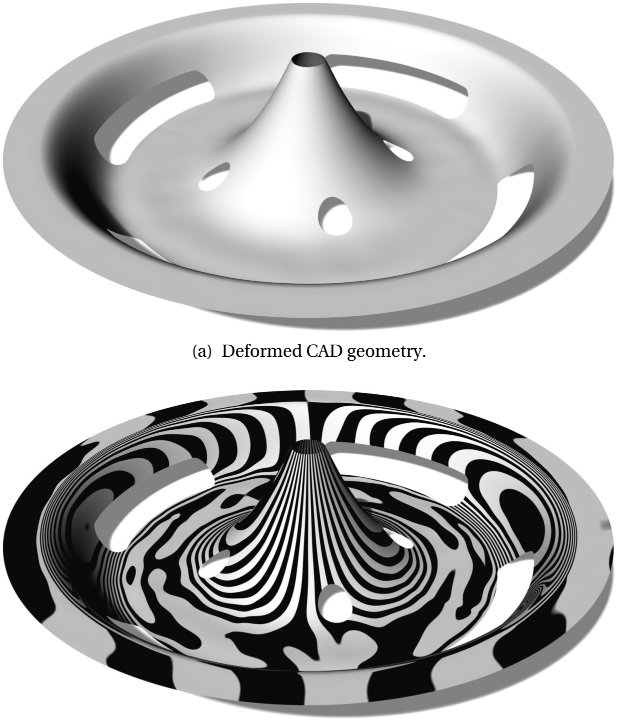
4.6.3 Fixation constraints

CAD surfaces to be fitted to a simulated displacement field are often part of a model assembly and therefore include interfaces (edges and surface transitions) that are predefined or must not change. When B-Rep morphing leads to a deviation of such specifications, we must fix the corresponding edges explicitly. This section introduces a constraint that enables the fixation of displacements or surface normals along arbitrary edges.

Following the chosen penalty approach, a specific displacement field $\widehat{\Delta\mathbf{S}}$ can be enforced at any edge $\partial\mathbf{S}$ by penalizing deviations of the actual displacement field to the target set. In weak form, the resulting penalty term reads:

$$\Pi_{\text{fix,disp}} = \frac{p_{\text{fix,disp}}}{2} \int_{\partial\mathbf{S}^{(0)}} \left[\Delta\mathbf{S} - \widehat{\Delta\mathbf{S}}\right] \cdot \left[\Delta\mathbf{S} - \widehat{\Delta\mathbf{S}}\right] \, \mathrm{d}\partial\mathbf{S}^{(0)} \tag{4.63}$$

[4] Refer to section 4.2.4 for details on the evaluation of the surface continuity.



(a) Deformed CAD geometry.

(b) Resulting surface qualty (visualized as reflection patter).

Figure 4.23: B-Rep morphing with coupling constraints.

Note that the integration is done over the edge's fixed initial configuration, $\partial\mathbf{S}^{(0)}$. Hence, a variation of the integration domain is avoided.

A certain orientation of the surface normal along the edge is reached if locally the tangent vectors of the actual surface lie within the plane spanned by the predefined normal vector. Mathematically this requires vanishing scalar products between the targeted surface normal $\hat{\mathbf{t}}_3$ and each tangent vector in the current configuration, $\mathbf{t}_1$ and $\mathbf{t}_2$ [5]. Weakly formulated, the corresponding penalty term reads:

[5] Small letters indicate the current configuration. Refer to figure 4.1 for a visualization of the individual vectors.

$$\mathit{\Pi}_{\text{fix,norm}} = \frac{p_{\text{fix,norm}}}{2} \int_{\partial \mathbf{S}^{(0)}} \left[\mathbf{t}_1 \cdot \hat{\mathbf{t}}_3\right]^2 + \left[\mathbf{t}_2 \cdot \hat{\mathbf{t}}_3\right]^2 \, \mathrm{d}\partial \mathbf{S}^{(0)} \tag{4.64}$$

The contributions of both of the above fixation constraints to the mapping system are obtained by deriving the individual penalty terms w.r.t. to the control point displacements. For the fixed displacement field, the derivation of (4.63) yields:

$$\begin{aligned} b_r &= -\partial_r(\mathit{\Pi}_{\text{fix,disp}}) \\ &= -p_{\text{fix,disp}} \int_{\partial \mathbf{S}^{(0)}} \partial_r(\mathit{\Delta}\mathbf{S}) \cdot \left[\mathit{\Delta}\mathbf{S} - \widehat{\mathit{\Delta}\mathbf{S}}\right] \, \mathrm{d}\partial \mathbf{S}^{(0)} \end{aligned} \tag{4.65}$$

$$\begin{aligned} A_{rs} &= \partial_{rs}(\mathit{\Pi}_{\text{fix,disp}}) \\ &= p_{\text{fix,disp}} \int_{\partial \mathbf{S}^{(0)}} \partial_r(\mathit{\Delta}\mathbf{S}) \cdot \partial_s(\mathit{\Delta}\mathbf{S}) \, \mathrm{d}\partial \mathbf{S}^{(0)} \end{aligned} \tag{4.66}$$

And for the fixed surface normal, deriving (4.64) w.r.t. $\mathit{\Delta}\mathbf{P}$ results in:

$$\begin{aligned} b_r &= -\partial_r(\mathit{\Pi}_{\text{fix,norm}}) \\ &= -p_{\text{fix,norm}} \int_{\partial \mathbf{S}^{(0)}} \left[\partial_r(\mathbf{t}_1) \cdot \hat{\mathbf{t}}_3\right] \cdot \left[\mathbf{t}_1 \cdot \hat{\mathbf{t}}_3\right] + \left[\partial_r(\mathbf{t}_2) \cdot \hat{\mathbf{t}}_3\right] \cdot \left[\mathbf{t}_2 \cdot \hat{\mathbf{t}}_3\right] \, \mathrm{d}\partial \mathbf{S}^{(0)} \end{aligned} \tag{4.67}$$

$$\begin{aligned} A_{rs} &= \partial_{rs}(\mathit{\Pi}_{\text{fix,norm}}) \\ &= p_{\text{fix,norm}} \int_{\partial \mathbf{S}^{(0)}} \left[\partial_r(\mathbf{t}_1) \cdot \hat{\mathbf{t}}_3\right] \cdot \left[\partial_s(\mathbf{t}_1) \cdot \hat{\mathbf{t}}_3\right] + \left[\partial_r(\mathbf{t}_2) \cdot \hat{\mathbf{t}}_3\right] \cdot \left[\partial_s(\mathbf{t}_2) \cdot \hat{\mathbf{t}}_3\right] \, \mathrm{d}\partial \mathbf{S}^{(0)} \end{aligned} \tag{4.68}$$

Note that (4.68) is constant since $\partial_r(\mathbf{t}_i) = \text{const}$. Consequently, the constraint does not render the problem nonlinear. Also, (4.68) does not contain second-order derivatives as $\partial_{rs}(\mathbf{t}_i) = 0$.

To allow for a qualified initial guess of the penalty factors, the penalty terms in (4.63) and (4.64) are scaled according to (4.54) before assembling their contributions to the global mapping system.

The following example demonstrates the two constraints. Consider again test case 1 from figure 4.8. We apply B-Rep morphing to deform the CAD model according to the prescribed displacement. Coupling constraints are specified to preserve the positional and rotational continuity across patch borders ($p_{\text{disp}} = 10^3$,$p_{\text{rot}} = 10^3$) and regularization is activated for a better system conditioning ($\alpha = 10^{-1}$, $\beta = 10^{-6}$). The value for α is intentionally chosen rather high in order to trigger deviations from the target displacement and subsequently show the effect of the fixation constraints.

Applying B-Rep morphing with the settings above, we obtain an overall good CAD description of the deformed finite element mesh, see figure 4.24 a). However, as expected, due to the rather high value for α, the outer edges of the geometry are less distinct, and a close-up on the innermost ring reveals clear deviations from the given target height (= 40 units).

Assuming that it is important to capture the deformation at the innermost ring correctly, a displacement constraint is introduced enforcing $\widehat{\Delta \mathbf{S}} = 40$ at the corresponding boundary. Indeed, rerunning the case with activated displacement constraint and a corresponding penalty factor of $p_{\text{fix,disp}} = 10^3$, we can successfully remedy the previous deviations, as figure shows 4.24 b).

To showcase a constraint on the surface normal, assume that some design criterion requires the surface normal at the innermost ring to stay vertical. To enforce such a design, we explicitly constraint the geometry to have $\hat{\mathbf{t}}_3 = [0, 0, 1]^T$ along the relevant edge. Rerunning the case with a displacement constraint and additionally an activated constraint on the surface normal ($p_{\text{fix,disp}} = 10^4$, $p_{\text{fix,norm}} = 10^4$), one obtains the geometry depicted in figure 4.24 c). As one can see qualitatively from the figure, both constraints are successfully satisfied, whereas the remaining CAD geometry still corresponds to a best-fit of the deformed finite element mesh.

It is important to realize that both constraints may not just be used to enforce a certain displacement or normal orientation, but also to prevent a modification of the latter during B-Rep morphing. Such a fixation is obtained by choosing $\widehat{\Delta \mathbf{S}} = 0$ and $\hat{\mathbf{t}}_3 = \mathbf{T}_3$. This feature is particularly relevant in bigger CAD assemblies, where it is of utmost importance not

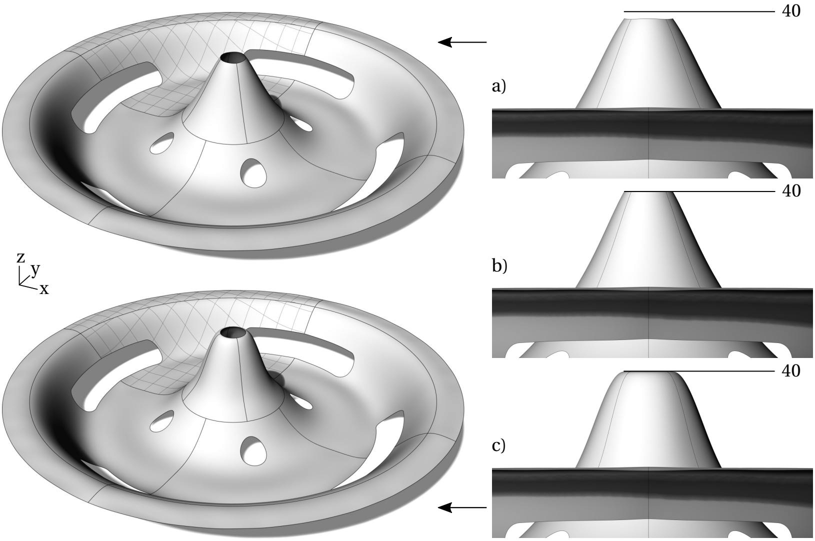


Figure 4.24: B-Rep morphing: a) without fixed boundaries, b) with fixed displacement at the innermost ring, c) with fixed displacement and surface normal at the innermost ring.

to alter interfaces to the remaining model.

In this scenario, we artificially provoked deviations on the boundary of the structure for demonstration purposes. However, practical applications show that such deviations frequently appear, as the underlying least-squares formulation does not guarantee a point-wise matching displacement field. Often the resulting deviations are negligible. Sometimes, however, they have to be controlled explicitly to maintain a defined interface. In such cases, the fixation constraints offer the possibility to reduce the deviation down to a minimum.

Finally, we note that the fixation constraint is not limited to edges. It is also possible to fix entire areas of the surface. The extension of (4.63) and (4.64) to areas of the surface is straightforward and mainly includes a change of the integration domain. In such a case, the discretization of the domain and the numerical integration follow the procedure described in section 4.3.1.

4.7 Surface smoothing

Apart from an ill-conditioning due to the involved trimming, B-Rep morphing also suffers from discrepancies in the description of the origin and the target geometry. The origin geometry is a discrete finite element mesh, whereas the target geometry is a continuous NURBS model with different degrees of freedom. This discrepancy naturally leads to an information gap, which can cause unwanted wrinkling in the deformed CAD model. The α-regularization presented in section 4.5.2 already provides a tool to smooth such surface defects. However, surface smoothing using the α-regularization is also limited, as minimizing the distances between control points and the associated NURBS faces not necessarily leads to a globally smooth surface. Control points may be locally close to the surface, while the overall surface still shows wrinkles.

An alternative way to enforce a smooth design is to add mechanical stiffness to problematic NURBS faces. Therefore, one can introduce an additional regularization (fairness) term based on the internal energy of a thin plate. Such an approach is a well-established practice in reverse engineering. See, e.g., Eck et al. [44], Greco et al. [50], Weiss et al. [53], and Greiner [101]. Herein, we adopt this approach as a means to suppress wrinkling, if necessary. We also combine it with α- and β-regularization to enforce numerical stability and achieve a practical layout of the control points.

We realize the additional smoothing by modeling the trimmed (visible) surface of the CAD model[6] as shell and penalizing the strain energy that results from the deformation of this shell during the morphing process. Therefore, the objective function of the already regularized mapping problem, (4.41), is extended by another regularization term, Π_γ. As a result, we obtain the following compromise function:

$$\begin{aligned} \Pi &= \Pi_M + \Pi_\alpha + \Pi_\beta + \Pi_\gamma \\ &= \Pi_M + \alpha C_\alpha + \beta C_\beta + \gamma W_{\text{in}} \end{aligned} \tag{4.69}$$

In the previous equation, W_{in} represents the strain or internal energy of the shell and γ the associated penalty factor. Note that minimizing (4.69) corresponds to a Pareto problem, in which α, β, and γ define the application-

[6] or just parts of it to reduce computational costs / avoid an unwanted global influence

specific weighting of the regularization terms. We compute W_{in} using isogeometric analysis (IGA) based on the Kirchhoff-Love (KL) shell formulation[7]. Accordingly it holds:

$$\Pi_\gamma = \gamma W_{\text{in}} = \gamma \int_{\mathbf{S}^{(0)}} [\mathbf{n} : \boldsymbol{\epsilon} + \mathbf{m} : \boldsymbol{\kappa}] \; \mathrm{d}\mathbf{S}^{(0)} \tag{4.70}$$

In the previous equation, **n** and **m** are the stress resultants from the forces and moments, respectively. $\boldsymbol{\epsilon}$ denotes the membrane strains and $\boldsymbol{\kappa}$ the change in curvature. $\mathbf{S}^{(0)}$ represents the surface to be modeled as a shell in its initial configuration. $\mathbf{S}^{(0)}$ can comprise the entire surface or just parts of it.

The KL theory distinguishes between a membrane action related to the strains and a bending action related to the change in curvature, cf. (4.70). In the context of B-Rep morphing, the membrane action shall avoid an uncontrolled stretching or compression of the surface. In contrast, the bending action shall keep the surface smooth by avoiding possible wrinkles (through a penalization of bending and twisting moments arising from a change in curvature).

n and **m** depend on the thickness of the shell, t, as well as its material parameters. For our purpose, a unit thickness of $t = 1$ and an isotropic material with a Poisson's ratio of $\nu = 0$ is assumed. The only remaining material parameter is the Young's modulus, E. Concerning the latter, it holds:

$$\mathbf{n} \sim E \quad \text{and} \quad \mathbf{m} \sim E \tag{4.71}$$

From (4.71) and (4.70), one can see that γ and E have the same effect on Π_γ. That is, increasing the penalty factor corresponds to a stiffening of the shell, which in turn results in a smoother surface. We set $E = 1$ so that the penalty factor is the only parameter to control the stiffness.

In order to facilitate a qualified choice of γ, (4.70) is scaled relative to the original mapping requirements. The scaling is realized analogously to all

[7] Refer to Kiendl et al. [102] for details on the isogeometric analysis of shells based on the KL shell formulation.

the other extensions so far. That is, Π_γ is scaled to the same dimension as the objective functional, Π_m. The same dimension means that, for $\gamma = 1$, the values which the regularization term adds to the mapping matrix are comparable to the values within the mapping matrix:

$$\Pi_\gamma = \gamma \left\| \frac{\partial^2 \Pi_M}{\partial \Delta \mathbf{P}^2} \right\|_\infty \left\| \frac{\partial^2 W_{\text{in}}}{\partial \Delta \mathbf{P}^2} \right\|_\infty^{-1} W_{\text{in}} \tag{4.72}$$

Given the scaled regularization term, a good initial choice of the penalty factor is $\gamma = 10^2$. The latter prioritizes a smooth shape over an accurate mapping in all areas that are covered by the shell.

To integrate the additional smoothing into the numerical solution, we follow the finite element approach introduced in section 4.4.2. Accordingly, the surface to be modeled as shell, $\mathbf{S}^{(0)}$, is discretized into elements, whereas each element computes its contribution to the mapping system separately by numerical integration. The local contributions are then assembled to the global mapping system. The discretization of $\mathbf{S}^{(0)}$ into elements and the corresponding numerical integration follow the procedure described in section 4.3.1. The local left- and right-hand side contributions are computed by deriving (4.70) w.r.t. the control point displacements. Since we are only interested in a stiffening of the surface and not a physically correct behavior, we neglect the nonlinear-terms and so assume a geometrically linear analysis of the shell. The relevant system contributions are:

$$b_r = -\partial_r(\Pi_\gamma) = -\gamma \int_{\mathbf{S}^{(0)}} [\mathbf{n} : \partial_r(\boldsymbol{\epsilon}) + \mathbf{m} : \partial_r(\boldsymbol{\kappa})] \; \mathrm{d}\mathbf{S}^{(0)} \tag{4.73}$$

$$A_{rs} = \partial_{rs}(\Pi_\gamma) = \gamma \int_{\mathbf{S}^{(0)}} [\partial_s(\mathbf{n}) : \partial_r(\boldsymbol{\epsilon}) + \partial_s(\mathbf{m}) : \partial_r(\boldsymbol{\kappa})] \; \mathrm{d}\mathbf{S}^{(0)} \tag{4.74}$$

In the scope of this work, we compute both system contributions by applying AD to (4.70) at element level. Note that the computation of both terms corresponds to the computation of the stiffness matrix and residual vector in an isogeometric shell analysis. Therefore, the interested reader is referred to Kiendl et al. [102] for a complete analytic derivation of (4.73) and (4.74).

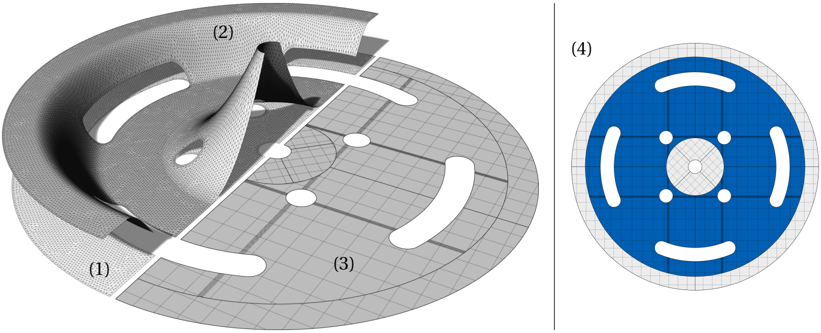


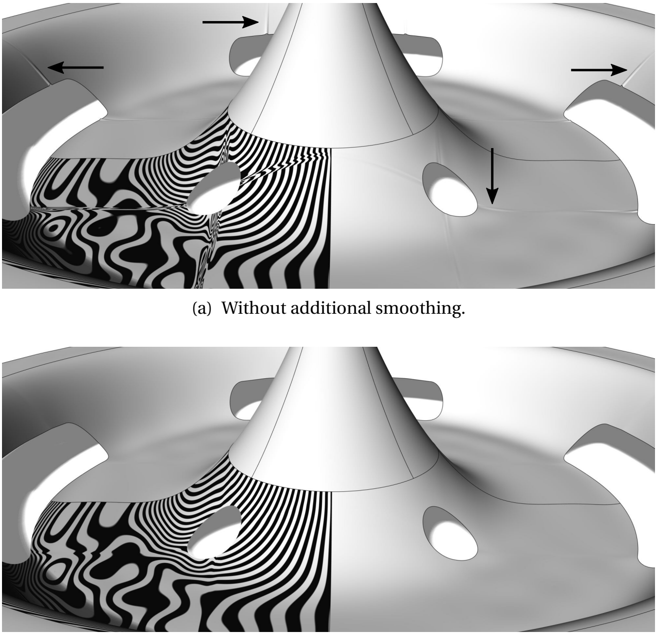
Figure 4.25: Test case 3: (1) initial mesh, (2) target displacement, (3) CAD geometry with heavy local knot refinement (stripes). Surfaces treated by shell-based smoothing are highlighted in (4).

Effect of surface smoothing

The effect of the introduced surface smoothing is demonstrated in the following. Consider "test case 3" depicted in figure 4.25. Herein, the finite element mesh, the displacement field, and the CAD geometry are adopted from test case 1, cf. figure 4.8. The only difference to the original test case is that the CAD geometry includes a heavy local knot refinement in four of the twelve patches. The goal is to obtain a quality NURBS description of the deformed finite element mesh starting from the refined CAD geometry. Therefore, B-Rep morphing is applied including a basic regularization ($\alpha = 10^{-3}, \beta = 10^{-6}$) and coupling constraints ($p_{\text{disp}} = p_{\text{rot}} = 10^4$).

Without any further treatment, B-Rep morphing is not able to deliver satisfying results. Indeed the displacement mapping leads to well-matching geometries and the specified coupling constraints to a continuous surface. Nevertheless, in the refined region, the high aspect ratio of the knot spans causes a clear wrinkling, see figure 4.26(a). Running a parameter study, one finds that this wrinkling cannot be cured by increasing α and exploiting the associated smoothing effect. Instead, forcing the control points to move closer to the surface amplifies the surface defects in this case.

In order to cure the defects, we smooth the critical area using the shell-based technique presented above. For this purpose, we model all four refined faces as shells while the rest of the domain stays untouched. Such



(a) Without additional smoothing.

(b) With shell-based smoothing.

Figure 4.26: B-Rep morphing applied to a case with heavy local knot refinement. The reflection pattern is visualized for a single face highlighting the quality difference.

a selective constraint avoids an unnecessarily global impact and may save computational resources. Both aspects are especially relevant in bigger setups with up to thousands of patches.

Figure 4.26(b) shows the result of B-Rep morphing with active surface smoothing and a penalty factor of $\gamma = 10^2$. From the figure, one can see that the stiffening successfully suppresses the wrinkles up to the point that almost no influence of the refinement is visible anymore. As a result, one obtains an overall smooth design. Still, the input displacement is well captured. We can also verify this quantitatively. Therefore compare the mean and maximal deviation between actual and target displacement in table 4.3. The deviations are measured at all Gauss points in the de-

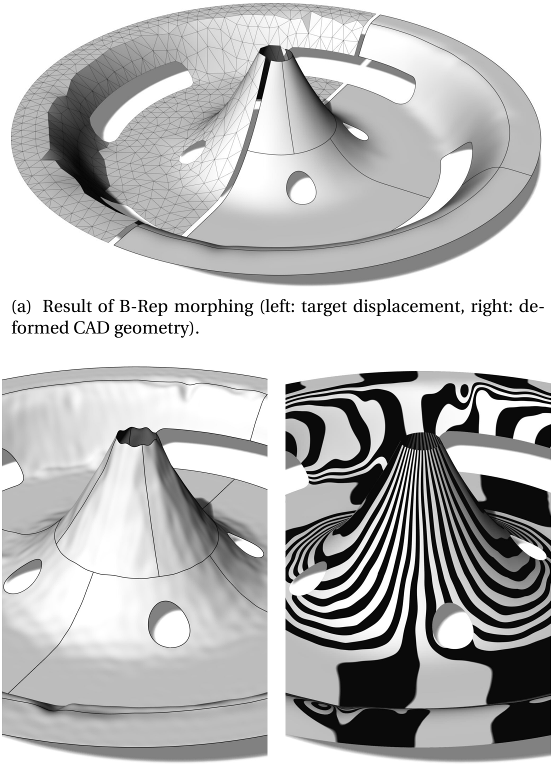
Table 4.3: Quality indicators evaluated at the deformed CAD model of test case 2 & 3. i denotes the index of a single Gauss point. Initial diameter of the shell: ⌀180 units.

quality indicator		evaluates	test case 3	test case 2
$\max\left(\lVert \mathit{\Delta}\mathbf{S}_i - \mathit{\Delta}\mathbf{Q}_i \rVert_2\right)$	[units]	mapping	0.30	1.02
$\max\left(\left\lVert \mathit{\Delta}\mathbf{S}_i^M - \mathit{\Delta}\mathbf{S}_i^S \right\rVert_2\right)$	[units]	coupling	7.80×10^{-3}	1.68×10^{-2}
$\max\left(\omega_{\mathbf{T}_2,i}^M - \omega_{\mathbf{T}_2,i}^S\right)$	[deg]	coupling	1.10×10^{-2}	1.75×10^{-2}
$\operatorname{mean}\left(\lVert \mathit{\Delta}\mathbf{S}_i - \mathit{\Delta}\mathbf{Q}_i \rVert_2\right)$	[units]	mapping	4.68×10^{-2}	8.04×10^{-2}
$\operatorname{mean}\left(\left\lVert \mathit{\Delta}\mathbf{S}_i^M - \mathit{\Delta}\mathbf{S}_i^S \right\rVert_2\right)$	[units]	coupling	1.37×10^{-4}	6.20×10^{-5}
$\operatorname{mean}\left(\omega_{\mathbf{T}_2,i}^M - \omega_{\mathbf{T}_2,i}^S\right)$	[deg]	coupling	2.40×10^{-3}	3.16×10^{-3}

formed CAD geometry. The table also shows an evaluation of the coupling constraints at all Gauss points along the coupled edges. The coupling constraints quantify positional and rotational deviations across patch borders and, by that, the surface continuity. Note from the table's values that the coupling constraints lead to a high surface continuity in this case.

Apart from models with heavy local knot refinement, shell-based smoothing is versatile in all cases where the discrete nature of the input mesh induces wrinkles in the deformed CAD model. We already discussed this scenario in test case 2, cf. figure 4.12. In this case, the discrepancy between the two geometry descriptions led to severe surface distortions, cf. figure 4.13. A possibility to still get a fair design in this test case is to apply shell-based smoothing over the entire surface. Figure 4.27 presents the corresponding results for a penalty factor of $\gamma = 50$. The factor was found by a parameter study starting from the recommendation given above. Unlike in the original test case, α- and β-regularization are also activated for a better control point layout ($\alpha = 10^{-3}, \beta = 10^{-6}$). Moreover, coupling constraints are included to obtain a continuous surface in the deformed CAD model ($p_{\text{disp}} = p_{\text{rot}} = 10^4$).

The result in figure 4.27(a) shows a smooth design with an acceptable error in the mapping. The corresponding mean and maximum deviation from the target displacement are compiled in table 4.3. The quality of the deformed CAD model is remarkable, considering the unusable design obtained without additional smoothing (see figure 4.27(b)). Thus, we find that shell-based smoothing can be used to enforce a feasible design when



(a) Result of B-Rep morphing (left: target displacement, right: deformed CAD geometry).

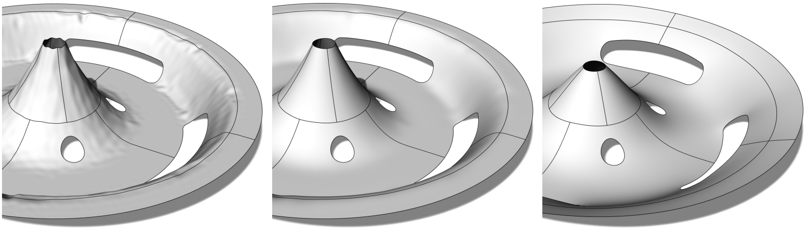
(b) Surface quality without additional smoothing.

(c) Surface quality with shell-based smoothing.

Figure 4.27: B-Rep morphing applied to test case 2 with additional shell-based smoothing.

the raw input would cause wrinkling.

Noteworthy in this test case is also the resulting surface quality, which we can assess using the reflection pattern in figure 4.27(c). Striking in this figure is the smooth gradient of the individual stripes and the invisible patch borders. These characteristics indicate a smooth design with a continuous surface. Due to the coupling constraints, the maximal positional and rotational deviations across all the patch borders are small (see table 4.3). The



(a) $\gamma = 10^{-1}$ (b) $\gamma = 50$ (c) $\gamma = 10^{4}$

Figure 4.28: Effect of γ in shell-based smoothing.

mean deviations are still much smaller. This result and the result from test case 3 show that we can readily combine the shell-based smoothing with the earlier introduced coupling constraints.

In general, a successful shell-based smoothing depends on the choice of the penalty factor γ. Figure 4.28 demonstrates the effect of the penalty factor. Herein, we compute again test case 2 with different values for γ. In the figure, one can see that a too high value for the penalty factor results in clear deviations from the target displacement as the model behaves overly stiff. Contrarily, a too low value does not generate the intended smoothing effect. Only a balanced value results in a design, which is smooth *and* adequately captures the target displacement.

Finally, it is worth noting that B-Rep morphing with shell-based smoothing may be interpreted as a forming process, where a shell with a dedicated stiffness is pressed into a mold, which is defined by the deformed finite element mesh. The stiffer the shell, the higher the mechanical resistance against the formation of details or wrinkles.

4.8 Geometry refinement

Mapping a displacement field onto a NURBS surface implies that the NURBS geometry has enough degrees of freedom to represent the target displacement in the first place. If not, it has to be enriched accordingly. Therefore, refinement strategies are necessary.

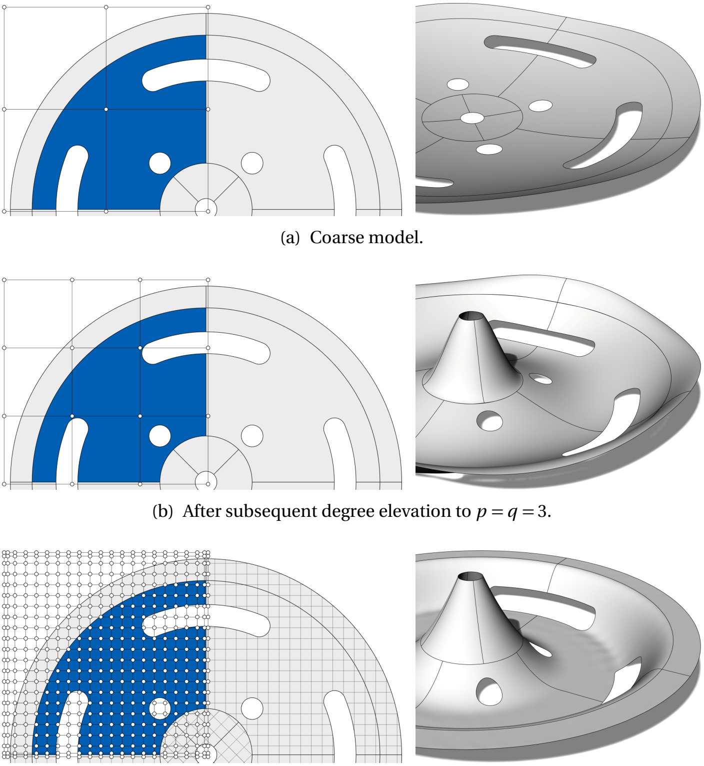
One distinguishes two basic strategies to refine NURBS: degree elevation and knot insertion Piegl et al. [81]. Both strategies increase the number of control points and thus the geometric freedom. Still, they are not changing the initial shape of the geometry. Both strategies can also be combined.

In IBRA, an established strategy is to combine an initial degree elevation with a subsequent knot-refinement. The sequence of both steps is important and does not commute. It guarantees a defined continuity of the basis functions across individual knots (required for structural analysis with IBRA). The entire process is referred to as k-refinement, cf. Hughes et al. [86].

Due to the close relation of the herein proposed B-Rep morphing to IBRA, we adopt k-refinement to refine NURBS geometries if necessary. In the included degree elevation and knot insertion, we follow the relevant algorithms from Piegl et al. [81]. Regarding the knot refinement, we implement it such that the user defines a maximum element size $h_{\max}$ in geometry space. The algorithm then inserts knots *uniformly* so that the resulting grid of nonzero knot spans only features rectangles with a maximum width of $h_{\max}$ measured at their center.

Figure 4.29 illustrates the adopted k-refinement for the test case from section 4.4.4. Model setup and target displacement are as specified in figure 4.8. The CAD model is assumed to be significantly coarser than the original one. The coarse model includes 12 patches with each 3×3 control points, a grid of 1×1 nonzero knot spans and it holds $p = q = 2$. We use coupling constraints to preserve the geometric continuity ($p_{\text{disp}} = p_{\text{rot}} = 10^3$) and apply a basic regularization for a better conditioning of the system ($\alpha = \gamma = 0, \beta = 10^{-6}$).

As one can see in the figure, the coarse CAD model does not properly represent the target displacement after deformation. Instead, the low resolution in the grid of nonzero knots and the low polynomial order limit the deformation to a parabolic shape. After an elevation of the polynomial degrees to $p = q = 3$, we can already capture the global trend, but details are still not resolved. Only a subsequent knot refinement with $h_{\max} = 5$ eventually yields a satisfying deformation, see figure. The process demonstrates a best-practice, which the author suggests for geometric refinement in the context of B-Rep morphing: First, one increases the polynomial order to a necessary but preferably low value (say $p = q = 3$). Then one applies knot refinement until the model has enough degrees of freedom to represent



(a) Coarse model.

(b) After subsequent degree elevation to $p = q = 3$.

(c) After subsequent knot refinement of case (b), $h_{\text{max}} = 5$.

Figure 4.29: Geometry refinement and its effect on the morphing result. Left: initial CAD geometry with highlighted edges, knot-lines and control points (the latter only for the blue domain), right: deformed CAD geometry.

the target displacement.

Following this approach, the necessary refinement, more precisely h_{max}, has to be estimated by the user. A qualified guess is possible from an observation of the displacement in the mesh. In the special case, where the displacement corresponds to a shape update obtained by Vertex Morphing, it is possible to quantify a good estimate of the element size. In Vertex

Morphing, the smallest geometric details are determined by the filter radius r. Moreover, for typical filter functions, the shape updates are taking the form of B-Spline surfaces, e.g., bi-cubic B-Spline surfaces for linear filter functions. So, assuming for the given NURBS patches a polynomial degree of at least $p = q = 3$, a good initial selection for the maximal element size is $r \leq h_{\max} \leq 2r$.

In this work, the geometric refinement is performed solely according to the semi-automatic procedure described above. So, the refinement still requires manual input by the user. It is, however, also possible to further automatize this step. Two ideas towards a more automatic and adaptive refinement are outlined in the following:

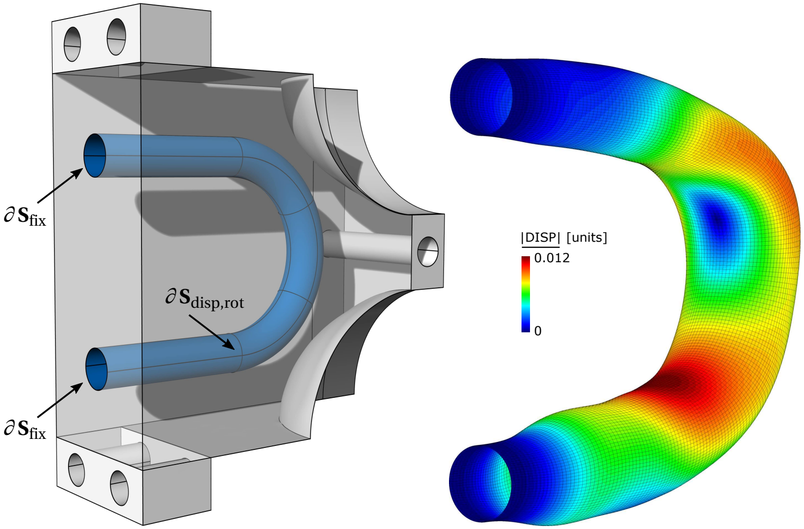
A straightforward yet rather expensive strategy could be based on an *a posteriori* comparison of the mapping results against the input. In this approach, B-Rep morphing is first applied to the unrefined CAD model. The resulting deformation is then compared to the input from the mesh, and a local refinement is triggered if the errors are too high. Then the deformation is computed again. The process is repeated until the deformed CAD model is acceptable in terms of specified error measures. An advantage of this strategy is that the user directly specifies error measures rather than estimating the resulting deviations implicitly by some refinement parameters.

A more efficient strategy would be to quantify the shape change in the given mesh *a priori* and draw conclusions on the necessary refinement before applying B-Rep morphing. Such an approach requires a criterion to decide locally whether or not a refinement is necessary. Once hot-spots are identified, an adaptive refinement of the NURBS surface may be triggered.

Both strategies may be the subject of further research.

4.9 Case studies

In the following, B-Rep morphing is tested in different application examples, showcasing its capabilities. Another example, which involves a racing car, is briefly presented in the appendix (A.2).


$\partial \mathbf{S}_{\text{fix}}$
$\partial \mathbf{S}_{\text{disp,rot}}$
$\partial \mathbf{S}_{\text{fix}}$
|DISP| [units]
0.012
0

(a) Initial CAD model of the valve housing with internal channel.

(b) Design of the channel as proposed by the shape optimization in section 3.8.1.

Figure 4.30: Discrete geometry of the shape-optimized channel to be converted into a CAD geometry.

4.9.1 Shape-optimized channel of a valve housing

First, B-Rep morphing is used to obtain a CAD description of the shape-optimized channel from section 3.8.1. The focus here is on the quality of the resulting CAD geometry as well as the practical application of B-Rep morphing. The optimized discrete geometry, as obtained from the grid-based shape optimization, are shown in figure 4.30(b). The channel is part of a valve housing. The initial CAD model of the housing is presented in figure 4.30(a)[8]. Since the optimization only affected the channel, we first focus on this part of the CAD model. Some general data for both the surface mesh and the CAD model of the channel are compiled in table 4.4.

Given this setup, the goal is now to deform the CAD model of the channel such that it matches the deformation of the surface mesh in the optimiza-

[8] For demonstration purposes, the large hole in the center was removed compared to the original model introduced in figure 3.5.

Table 4.4: Details on the initial CAD model (not refined) and the surface mesh of the channel.

CAD model (NURBS-based)				mesh	
# patches	degrees	trimming	# control points	# nodes	# elements
20	$1-3$	yes	1236	19008	18912

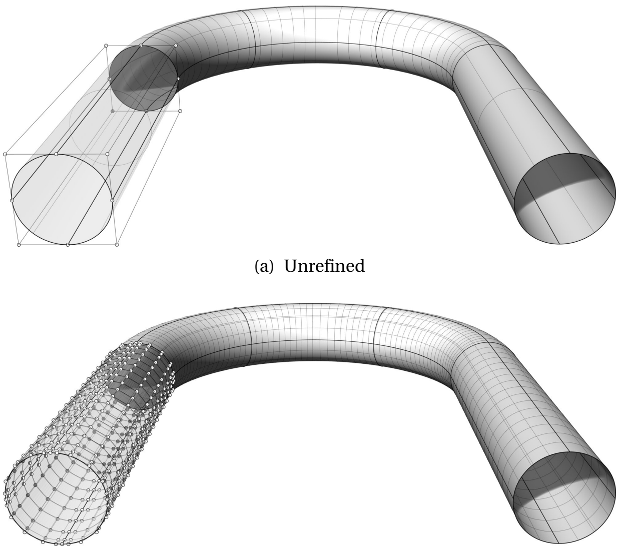
Table 4.5: Settings for B-Rep morphing in case of the channel.

α	β	p_{disp}	p_{rot}	$p_{\text{fix,disp}}$	$p_{\text{fix,norm}}$	γ
10^{-5}	10^{-8}	10^3	10^2	10^3	10^3	0

tion. Therefore, we extract the channel geometry from the CAD model, then apply B-Rep morphing, and eventually insert the new geometry back into the overall CAD model again.

Given that we treat the channel separately, we must take special care not to alter the interface to the surrounding model, $\partial \mathbf{S}_{\text{fix}}$. Therefore, fixation constraints are introduced at the interface, cf. figure 4.30(a). More precisely, at $\partial \mathbf{S}_{\text{fix}}$, we force the displacement to zero and fix the surface normal to its initial orientation. Both constraints prioritize a matching interface over any other shape modifications. They ensure that we can readily merge the modified channel geometry with the overall model afterwards. Coupling constraints are included along internal edges of the channel to preserve the initial G^0- and G^1- continuity as much as possible. Having a defined surface orientation at the inlet and outlet as well as a continuous surface in between is of particular importance in this case as it directly affects the flow guidance. We activate regularization to improve the problem conditioning and layout of the control points. The choice of the relevant penalty factors is summarized in table 4.5. The individual penalty factors were either determined based on experience with other applications or based on a systematic parameter study starting from the recommended initial values.

From an observation of the initial CAD model, it is clear that the initial model is too coarse to represent the intended shape modification accurately. Hence refinement as introduced in section 4.8 is applied. Accordingly, we first raise all polynomial degrees to $p = q = 3$ and then perform a knot-refinement. The latter requires the definition of the maximum el-



(a) Unrefined

(b) Refined

Figure 4.31: Comparison of the CAD model of the channel before and after refinement.

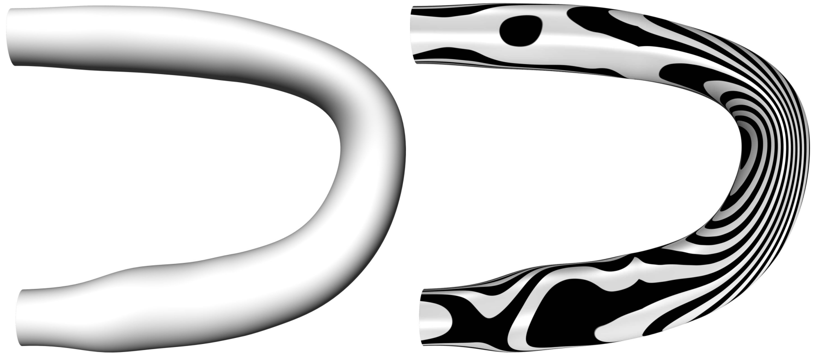
Table 4.6: Problem size after refinement in case of the channel.

# control points	# unknowns	# elements
5 264	15 792	7 532

ement size $h_{\max}$. Following the ideas discussed in section 4.8, we choose $h_{\max} = 0.05$. Figure 4.31(b) illustrates the refined CAD geometry.

The resulting size of the underlying mapping problem is summarized in table 4.6. The latter includes the number of control points of the refined CAD model, the number of unknown control point displacements, and the number of all finite elements[9] that arise from the mapping, the regularization, and the constraints.

[9] Not to be mistaken with the finite elements from the input mesh.



(a) Deformed surface (knots & edges are hidden).

(b) Corresponding reflection pattern.

Figure 4.32: Deformed CAD model of the channel obtained by B-Rep morphing.

Applying B-Rep morphing with two solution iterations using the setup introduced above, we obtain a quality CAD description of the optimized channel (see the rendering in figure 4.32(a) and compare it to the input in figure 4.30(b)). From a qualitative perspective, all apparent shape changes are well captured, and a continuous and smooth surface is obtained despite the complicated free-form modifications. An inspection of the reflection pattern qualitatively confirms this impression, cf. figure 4.32(b). Note in the figure that the reflection pattern shows a surface continuity of at least G^1. A G^1-continuous transition can be, e.g., observed from the kink in the stripes where the bend starts and ends. Apart from that, the individual patches are not visible in this view.

The qualitative impression can also be quantitatively confirmed. Table 4.7 presents a set of quality indicators, which evaluate the result in terms of the mapping, the coupling, and the fixation constraints. To evaluate the mapping, we measure the locally remaining deviation between the prescribed displacement of the mesh and the actual displacement of the deformed CAD surface. To evaluate the coupling, we measure the difference of the displacement and rotation of the master and the slave surface along all coupling edges. To evaluate the fixation, we locally measure the displacement of the relevant boundary and the rotational deviation of the

Table 4.7: Quality indicators evaluated at the deformed channel geometry - with and without active constraints. i denotes the index of a single Gauss point. Initial diameter of the channel: $\varnothing 2 \times 10^{-2}$ units.

quality indicator	evaluates	with constraints	no constraints
$\max\left(\Vert \Delta \mathbf{S}_i - \Delta \mathbf{Q}_i \Vert_2\right)$ [units]	mapping	1.23×10^{-5}	1.19×10^{-5}
$\max\left(\left\Vert \Delta \mathbf{S}_i^M - \Delta \mathbf{S}_i^S \right\Vert_2\right)$ [units]	coupling	4.58×10^{-6}	3.34×10^{-5}
$\max\left(\omega^M_{\mathbf{T}_2,i} - \omega^S_{\mathbf{T}_2,i}\right)$ [deg]	coupling	1.14×10^{-2}	1.57
$\max\left(\Vert \Delta \mathbf{S}_i - \mathbf{0} \Vert_2\right)$ [units]	fixation	4.83×10^{-10}	5.39×10^{-6}
$\max\left(\angle(\mathbf{t}_{3,i}, \mathbf{T}_{3,i})\right)$ [deg]	fixation	2.16×10^{-5}	0.16
$\mathrm{mean}\left(\Vert \Delta \mathbf{S}_i - \Delta \mathbf{Q}_i \Vert_2\right)$ [units]	mapping	1.86×10^{-6}	1.73×10^{-6}
$\mathrm{mean}\left(\left\Vert \Delta \mathbf{S}_i^M - \Delta \mathbf{S}_i^S \right\Vert_2\right)$ [units]	coupling	3.75×10^{-8}	2.58×10^{-6}
$\mathrm{mean}\left(\omega^M_{\mathbf{T}_2,i} - \omega^S_{\mathbf{T}_2,i}\right)$ [deg]	coupling	6.95×10^{-4}	0.25
$\mathrm{mean}\left(\Vert \Delta \mathbf{S}_i - \mathbf{0} \Vert_2\right)$ [units]	fixation	1.40×10^{-10}	1.74×10^{-6}
$\mathrm{mean}\left(\angle(\mathbf{t}_{3,i}, \mathbf{T}_{3,i})\right)$ [deg]	fixation	6.26×10^{-6}	0.12

new surface normal relative to the original surface normal. We compute all values in the deformed CAD geometry at the relevant Gauss points denoted by the index i. To highlight the effect of the constraints, we likewise consider the case where all constraints are deactivated. The corresponding values are also summarized in table 4.7. Looking at the values in the table, one can observe the following:

1. The deformed CAD model is very close to the target geometry (The maximum deviation resulting from the mapping is three orders below the channel's initial diameter, the mean deviation even four).

2. Activating the constraints improves the surface quality at the coupled and fixed boundaries significantly. The maximal values for all deviations drop at least by two orders, the mean values even more. Especially important in conjunction with a fluid is that the rotational deviations at coupling edges are minimal, which gives rise to an overall continuous surface. Also, we preserved the surface normal at the inlet and outlet (fixed boundary), which is essential for the flow guidance.

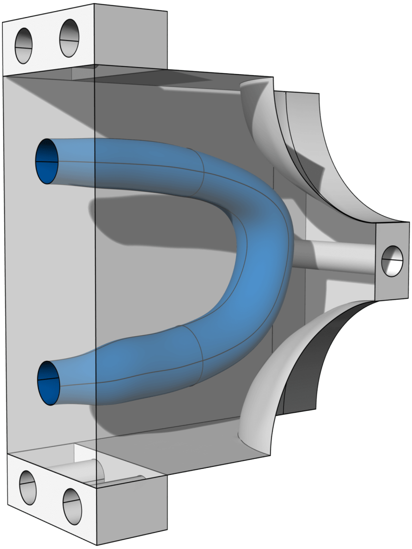


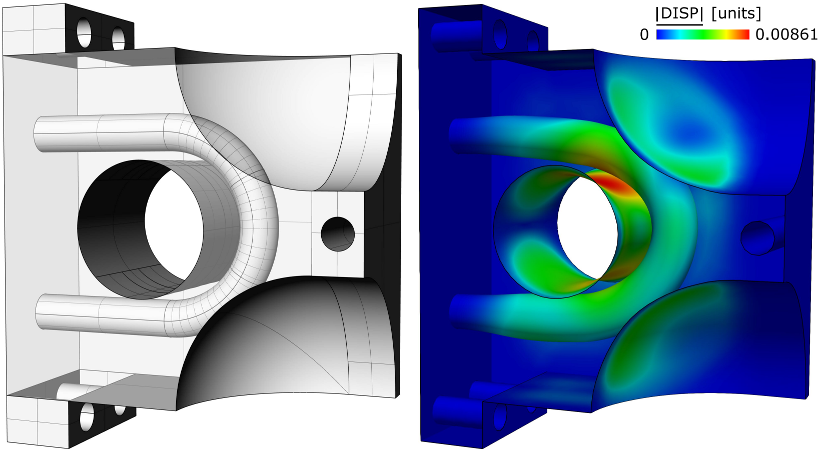
Figure 4.33: CAD description of the optimized channel inserted back into the overall model.

3. From the quality measures, it seems that one obtains an already acceptable result without constraints. However, considering the purpose of the given part, which is to guide a fluid flow, any artificial kink must be avoided, and a smooth surface is of utmost importance. In this context, the rotational deviations in the unconstrained case are considerable.

4. With or without constraints, the resulting mapping error is comparably low. So, the constraints do not have a negative influence on the overall deformation.

Having obtained a CAD description of the optimized channel, we can insert the result back into the overall model (see figure 4.33). Due to the fixed interface, the part perfectly fits into the model without any post-processing.

In the example above, we only considered the channel, which is characterized by a single open surface. B-Rep morphing is, however, equally



(a) Initial CAD model.

(b) Mesh with optimized shape and corresponding displacement field (edges are hidden).

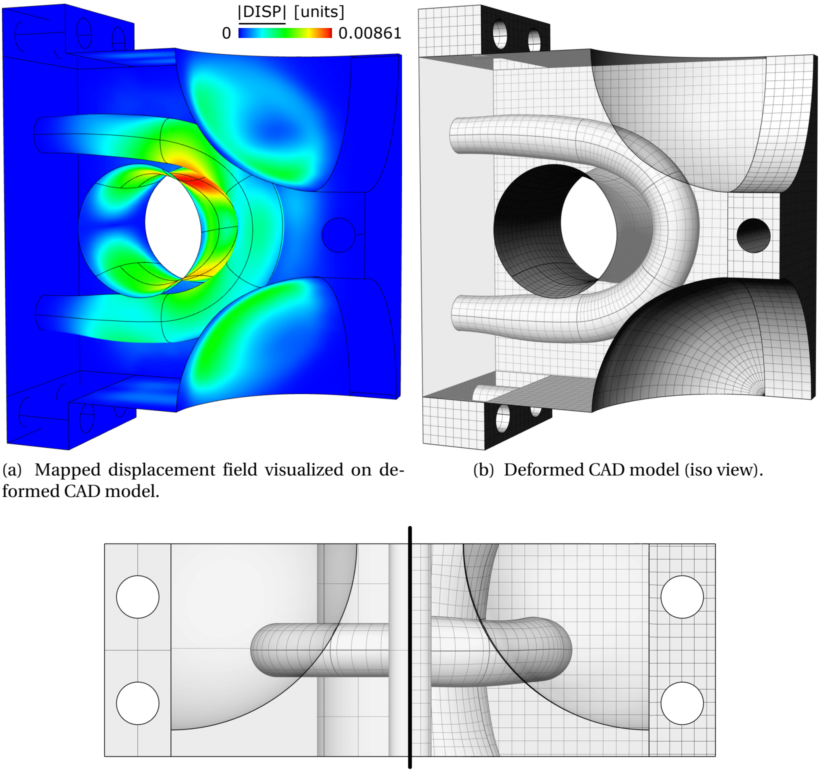
Figure 4.34: Discrete geometry of the shape-optimized valve housing to be converted into a CAD model.

Table 4.8: Details on the initial CAD model (not refined) and the mesh in case of the valve housing.

CAD model (NURBS-based)				mesh	
# patches	degrees	trimming	# control points	# elements	# nodes
66	1 – 3	yes	2 160	155 272	813 142

applicable to general B-Rep models, which also includes surfaces that define a volume. To demonstrate this capability, consider the results of the coupled optimization of the entire valve housing as presented in section 3.8.2. The optimized design, given as mesh with displacement field, is depicted in figure 4.34(b). The CAD model of the original design is shown in figure 4.34(a). Details on both the CAD model and the mesh are compiled in table 4.8.

We can obtain a CAD description of the optimized housing design by refining the initial CAD model and applying B-Rep morphing with active regularization and constraints. The mapped displacement is visualized



(a) Mapped displacement field visualized on deformed CAD model.

(b) Deformed CAD model (iso view).

(c) Side view of CAD geometry before (left) and after the deformation (right)

Figure 4.35: Results of B-Rep morphing in case of the valve housing (The knot lines in figure 4.35(b) and 4.35(c) visualize the refinement).

on the deformed CAD model in figure 4.35(a). Comparing the mapped displacements to the prescribed displacements in the mesh, we can see that the results are well-matching. The corresponding deformation of the CAD model is presented in figure 4.35(b) and 4.35(c). Note herein the significant shape changes which resemble the optimized discrete geometry in figure 3.28.

4.9.2 Sheet metal part with optimized bead pattern

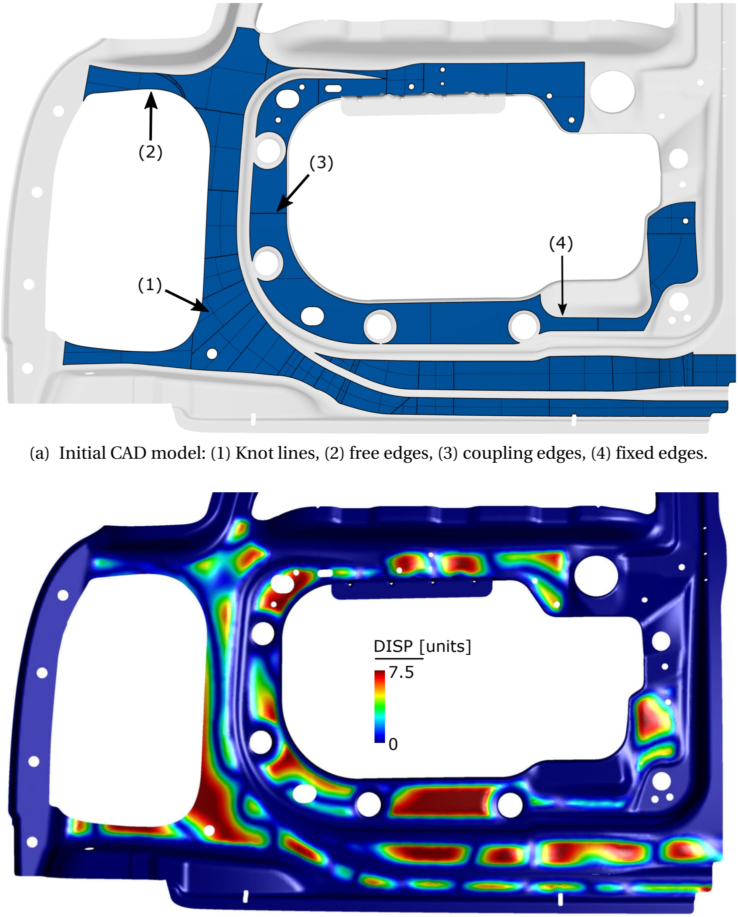
In this section, a NURBS-based CAD model of a sheet metal part shall be adapted to match the result of a bead optimization. In doing so, we set a particular focus on the combination of constraints and surface smoothing. The structure of interest represents a part of a car door and is shown in figure 4.36(a) (blue area).

For the highlighted part, a bead optimization was performed. The bead optimization was based on Vertex Morphing and operated directly on the finite element mesh used for the numerical analysis of the car door. The objective was to maximize the first three eigenfrequencies for a given set of support conditions. To this end, the three eigenfrequencies were combined to a single compromise function. A maximum bead height of 7.5 units was specified. The modified finite element mesh together with the resulting shape update is depicted in figure 4.36(b). The new shape yields an approximately 30% increased objective value and performs significantly better than the original design.

The bead pattern arises from a complex shape update with detailed geometric features and beads forming out on both sides of the structure. See figure 4.37 for a close-up on the bead pattern. Note from the close-up that some boundaries do not show any displacement. Those boundaries represent the interface to the surrounding model of the car door. Accordingly, they were fixed in the optimization. The goal is to reproduce a CAD description of the new bead pattern while ensuring a matching interface to the surrounding assembly.

The departure point is the displacement field from the finite element mesh, which shall be mapped to the initial CAD model using B-Rep morphing with relevant constraints. A particular challenge in this example is that the CAD model consists of multiple patches with heavily trimmed and sometimes minimal surfaces. If we do not want to change the CAD model's topology, we have to use this patchwork of surfaces to reproduce the detailed and continuous bead pattern.

Remember that for B-Rep morphing, integration points have to be projected from the CAD domain onto the finite element mesh to evaluate the local displacement field. So, a finite element mesh is required for the entire CAD surface to be modified. In turn, the finite element mesh may span a much larger domain without consequences on the B-Rep morphing.



(a) Initial CAD model: (1) Knot lines, (2) free edges, (3) coupling edges, (4) fixed edges.

(b) Finite element mesh with visualized shape update (displacement field).

Figure 4.36: CAD model of a part of a car door and corresponding results from a bead optimization using Vertex Morphing.

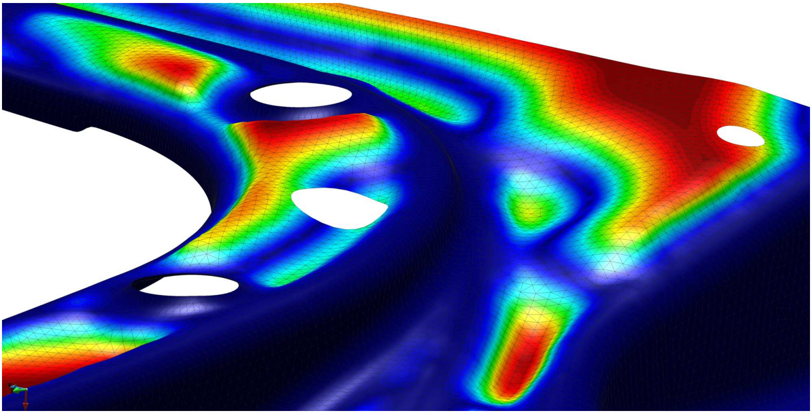


Figure 4.37: Close up of result from bead optimization.

Table 4.9: Details on the initial CAD model (not refined) and the finite element mesh in case of the sheet metal part.

CAD model (NURBS-based)				mesh	
# patches	degrees	trimming	# control points	# elements	# nodes
40	$1-3$	yes	1 488	158 136	81 061

Consequently, we do not need to cut the car door's finite element mesh to only the optimized area, but we can use the full mesh as is. This feature facilitates the practical application of B-Rep morphing significantly. Details on the finite element mesh and the initial CAD model (only the domain subjected to optimization) are summarized in table 4.9.

Before applying B-Rep morphing, the original CAD model must be refined first. The refinement is done according to the method described in section 4.8. We choose a maximal element size of $h_{\max} = 4$ units and a target degree of $p = q = 3$. The refined CAD model is presented in figure 4.38.

We introduce coupling constraints (rotation and displacement) at all internal edges to preserve the surface continuity as much as possible. Moreover, we include fixation constraints (only displacement) to suppress any deformation of the edges that interface to the surrounding assembly. The

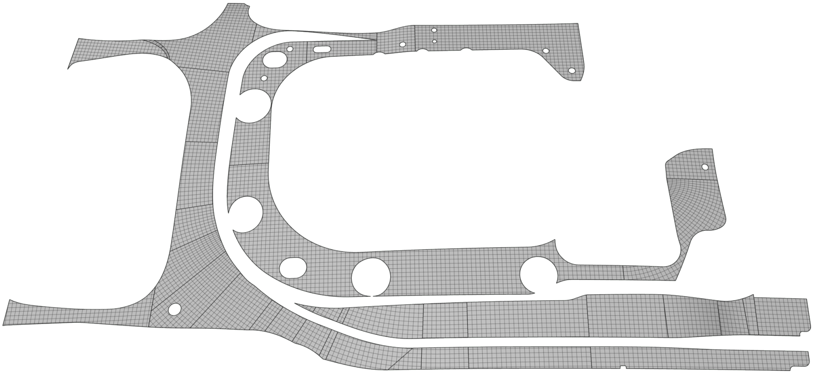


Figure 4.38: Refined CAD model of the sheet metal part.

fixation ensures that we can readily reinsert the modified CAD model into the car door's overall model. Indeed, the finite element mesh already indicates zero displacements exactly on these edges, as this condition was already included in the optimization. However, due to the nature of the present least-squares method, deviations may still appear depending on the CAD model's refinement. In any case, the fixation constraints prioritize the fixed boundary over the mapping requirements. Edges that are not connected to the remaining car door are left free. Figure 4.36(a) indicates the different types of edges.

Additional to the previous constraints, we apply β-regularization to control the problem conditioning. As shown later, this case suffers from wrinkles in the resulting CAD geometry and therefore requires surface smoothing. In order to prevent this wrinkling, we make use of α-regularization or stiffness-based smoothing. The problem size resulting from the refinement and the introduction of the previous constraints is summarized in table 4.10.

The relevant penalty factors and weights are specified in table 4.11. The values were obtained through a systematic parameter study starting from the recommendations given in the relevant sections above. Note in the table that there are two factors for α and γ. Both refer to different test cases later. Also worth emphasizing is that in this section, almost all of the previously introduced element types come together, highlighting the flexibility of the proposed B-Rep morphing.

Table 4.10: B-Rep morphing applied to the sheet metal part: Problem size after refinement in case of (1) no α-regularization or stiffness-based smoothing, (2) active α-regularization, (3) active stiffness-based smoothing.

# control points	18379
# unknowns case(1)	31599
# unknowns case(2)	55137
# unknowns case(3)	31599
# elements case (1)	9685
# elements case (2)	28064
# elements case (3)	16066

Table 4.11: Settings for B-Rep morphing in case of the sheet metal part.

α	β	p_{disp}	p_{rot}	$p_{\text{fix,disp}}$	$p_{\text{fix,norm}}$	γ
0 or 5	10^{-8}	10^3	10^4	10^3	-	0 or 500

Figure 4.39 visualizes the different elements, which result from the formulation of the mapping problem. The latter gives an impression of the finite element approach used in this thesis. Note from the figure that all elements, except the point elements from the α-regularization, lay inside the trimmed domain. In contrast to that, α-regularization affects the entire patches, leading to point elements outside the trimmed domain. In the figure, those elements reveal the trimming within the single patches. Note in this context that α-regularization always includes all control points in the problem formulation. If it is not applied, only those control points are relevant that affect the visible surface. The problem size differs accordingly.

Based on the settings specified above, we apply B-Rep morphing to map the displacement field from the finite element mesh to the CAD model. Figure 4.40(b) shows the results for the case of $\alpha = \gamma = 0$. When comparing the mapped displacement field to the input in figure 4.40(a), we observe a good qualitative agreement. However, looking at the deformed CAD model, a close-up is presented in figure 4.41(a), one finds that clear wrinkles form out. The wrinkles destroy the initially high surface quality.

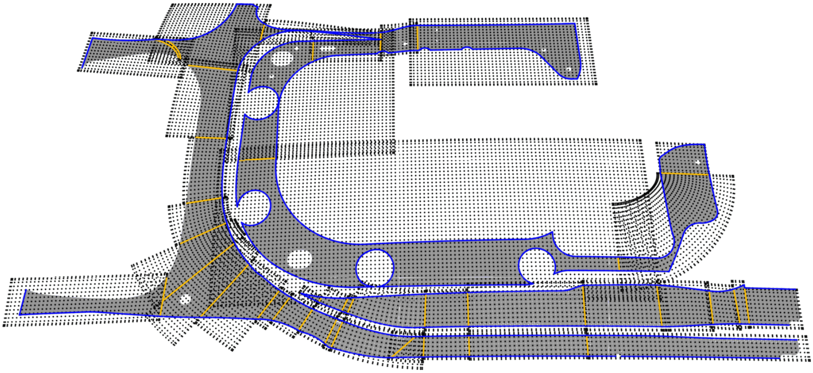
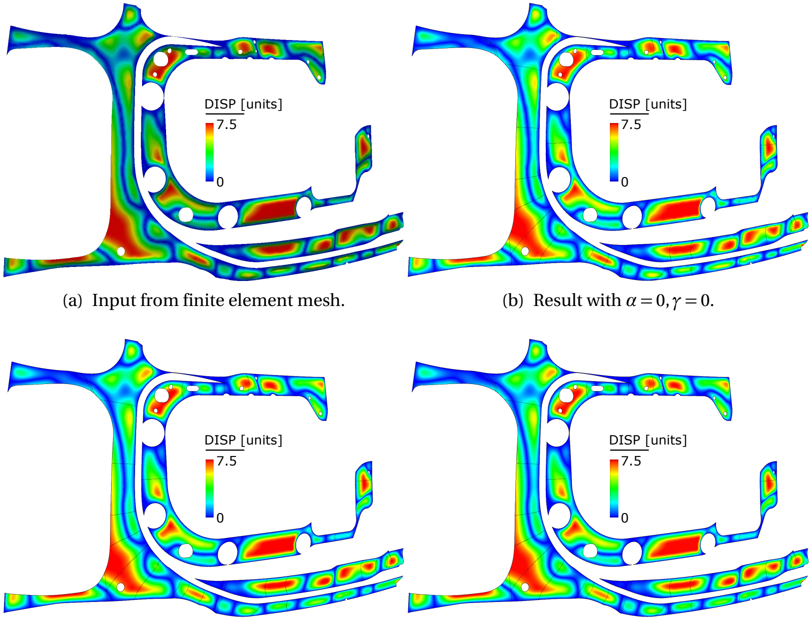


Figure 4.39: Different types of elements in case of the sheet metal part: from the mapping requirements (grey), the coupling constraints (orange), the fixation constraints (blue), and the α-regularization (black). Elements from the stiffness-based smoothing coincide with the mapping elements (grey).

The reason for these wrinkles is twofold. First, the refinement of the CAD model, which is necessary to reproduce the local beads in the first place, causes the discretization of the input mesh to become visible in the result. Second, there are control points at the trimmed boundaries that only have little effect on the visible surface and hence show an uncontrolled movement. The latter leads to abrupt surface changes in the transition from the visible to the invisible domain. As a result, the rectangular parameter grids of the NURBS surfaces induce wrinkles. The effect was already investigated in section 4.5.2. Figure 4.41(a) visualizes the control points for selected patches which clearly shows the uncontrolled movement. Earlier in this thesis, we introduced α-regularization and the stiffness-based smoothing to deal with such wrinkling. Their impact, in this case, is presented in the following.

Rerunning the case with $\alpha = 5$, the mapped displacement field hardly changes compared to the case without α-regularization, cf. figures 4.40(c) and 4.40(b). Only a closer look reveals minor local differences (see, e.g., the peak values at the lowermost beads in the figure). In contrast to that, the surface quality of the deformed CAD model improved substantially, cf. figure 4.41(b). The improved quality is reached by moving the control



(a) Input from finite element mesh.

(b) Result with $\alpha = 0, \gamma = 0$.

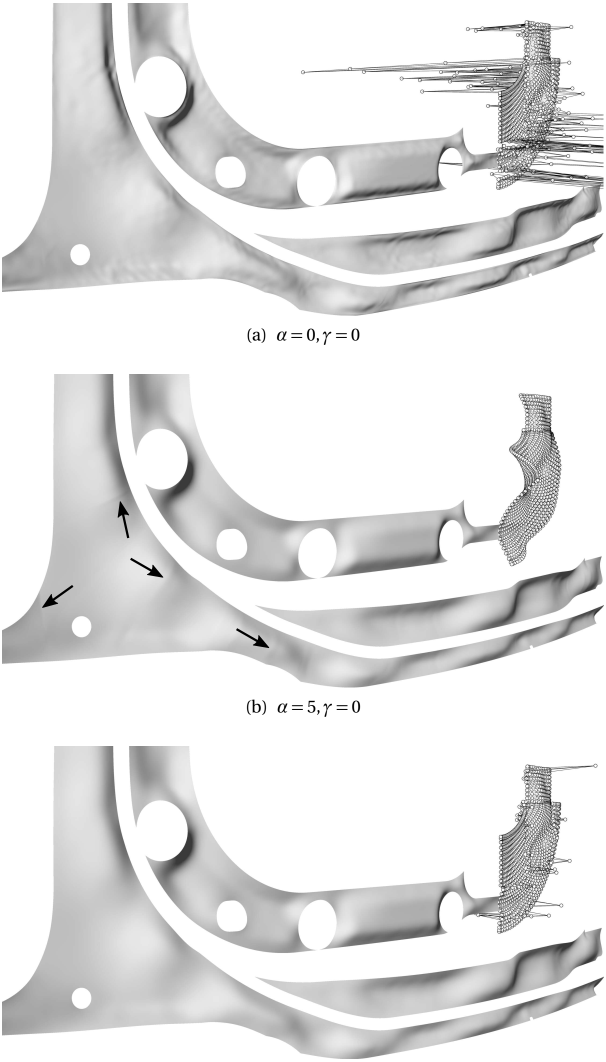
(c) Result with $\alpha = 5, \gamma = 0$.

(d) Result with $\alpha = 0, \gamma = 500$.

Figure 4.40: Mapped displacement field in case of the sheet metal part. The result is shown for different settings on the deformed CAD geometry.

points closer to the deformed CAD surface, which smooths the latter. Also, a much more practical layout of the control points is obtained (see figure). Note that the influence of the regularization on the solution of the mapping system was chosen to be significantly lower than the influence of the coupling constraints ($p_{\text{disp}}, p_{\text{rot}} >> \alpha$). This choice ensures that, despite the modified control point movement, G^1-continuity is preserved as much as possible along the internal edges.

As an alternative to the α-regularization, we, in the following, rerun the case with active stiffness-based smoothing ($\gamma = 500$). In doing so, the whole surface is modeled as a shell since the entire structure suffers from wrinkling. Rerunning the case with these settings, we find that the mapped displace-



(a) $\alpha = 0, \gamma = 0$

(b) $\alpha = 5, \gamma = 0$

(c) $\alpha = 0, \gamma = 500$

Figure 4.41: Section of the deformed CAD model in case of the sheet metal part. The result is shown for different settings. Control points are visible for selected patches.

ment field hardly differs from the one obtained without any smoothing, cf. figures 4.40(d) and 4.40(b). However, the surface quality improved substantially, see figure 4.41(c). In this case, the smoothing originates from the underlying shell formulation, which tends to minimize shape modifications that lead to a high bending moment (like wrinkles). Note that stiffness-based smoothing does not directly affect the control points. So unlike with the α-regularization, there might still be control points with an unwanted movement in the vicinity of trimmed boundaries, cf. figure 4.41(c).

In both cases, the smoothing causes a slight local deviation of the deformed CAD geometry from the input. The deviation is visualized for an isolated section of the structure in figure 4.42. The figure shows an overlay of the deformed finite element mesh (input) and the resulting CAD geometry after B-Rep morphing using different settings. Comparing the individual cases, one can see that smoothing always leads to a less accurate map, especially in regions with big curvature changes. In those regions, both techniques prioritize a smooth transition over the actual mapping requirements. However, given that an unconstrained mapping of the displacement field causes wrinkles, it is reasonable to allow deviations in favor of a practical design. Generally, though, the smoothing effect has to be balanced against the required accuracy of the map.

The previous investigations showed that both α-regularization and the stiffness-based smoothing successfully suppress the wrinkles in this case. As a result, one obtains a quality CAD model of the bead-optimized structure. Differences are observed in the quality of the surface and the resulting layout of the control points. Striking in figure 4.41(b) is that with active α-regularization, some of the coupling edges are visible in the deformed CAD model (arrows in the figure). This observation makes sense as the included rotation coupling at best preserves tangential continuity across coupling edges. Compared to that, it is striking that in the case of stiffness-based smoothing, the internal edges are entirely invisible, indicating a much better surface quality. The reason for this is not yet thoroughly understood. However, it is assumed that this behavior is a consequence of the chosen shell formulation in the stiffness-based smoothing.

An overall best design is obtained when activating both α-regularization with a small penalty factor of $\alpha = 10^{-3}$ and stiffness-based smoothing with $\gamma = 500$. The result, for the most part, corresponds to the case with only

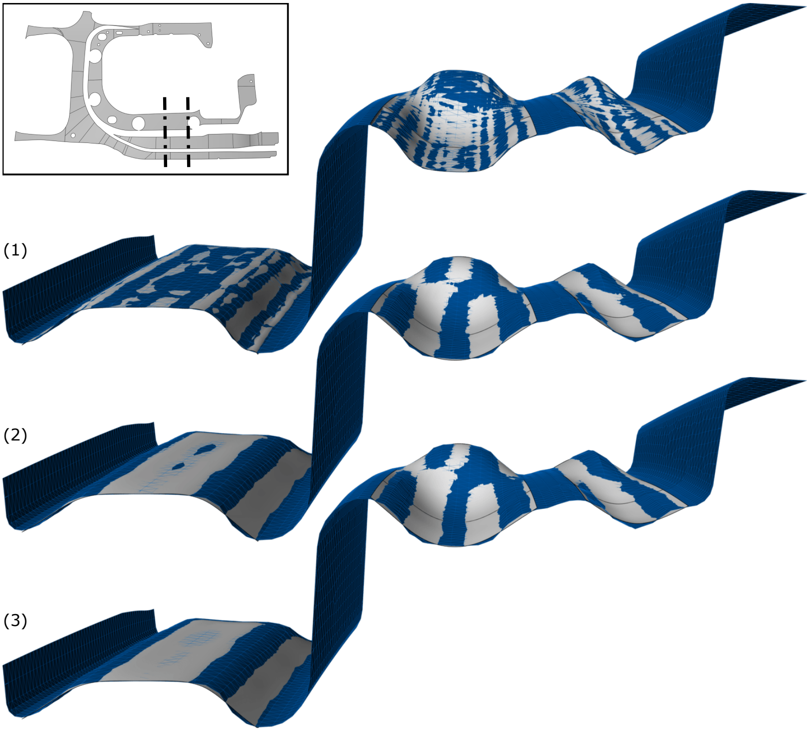


Figure 4.42: Deviations of the deformed CAD geometry (grey) from the deformed finite element mesh (blue) due to smoothing: (1) $\alpha = 0, \gamma = 0$ (no smoothing), (2) $\alpha = 5, \gamma = 0$ (3) $\alpha = 0, \gamma = 500$.

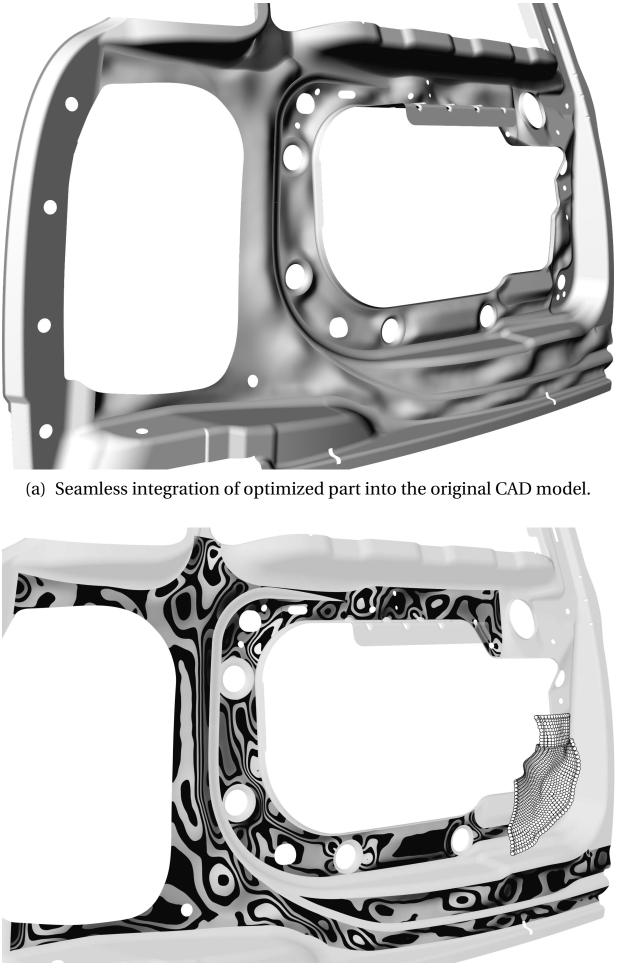
stiffness-based smoothing. Unlike in the latter case, however, the small value for α ensures a practical layout of the control points with smooth transitions between the visible and trimmed domain and without uncontrolled outliers. Hence, one obtains a design, which has a smooth surface, reproduces the deformed finite element mesh sufficiently *and* contains a practical layout of the control points.

The overall best design can be finally inserted back into the complete model assembly in place of the old part (see figure 4.43). Since we fixed the interfaces by constraints, a gapless replacement is possible without any post-processing. As a result, we obtain a *quality* CAD model of the entire sheet metal part. Figure 4.43(b) highlights the quality of the new design. The figure shows the new design's reflection pattern, together with the control points of some selected patches. Note that no kinks or internal edges are visible along the surface, which indicates a continuous geometry and results from the coupling constraints. Note also that the control points follow a regular distribution close to the underlying surface, which is advantageous for further steps in the design process.

From the investigations in this section, one can conclude:

- B-Rep morphing allows us to seamlessly integrate the result of any CAD-free shape optimization into a computer-aided design process. This feature is handy if one wants to exploit the advantages of grid-based shape optimization but needs to deliver a CAD description of the final design. It also provides the possibility for a CAD model if a CAD-free shape optimization is the only option because the definition of an adequate CAD parametric is challenging or even impossible. In this context, note that it would be troublesome to come up with a CAD parametric for the sheet metal part, allowing similar optimization results - especially a priori, so not knowing the presented results.
- With B-Rep morphing, it is possible to obtain quality CAD models even with highly trimmed multi-patch surfaces onto which an intricate free-form pattern shall be mapped.
- All regularization terms and constraints may be successfully combined as needed by merely choosing the corresponding weights and penalty factors.
- The choice of the weights and penalty factors typically requires a parameter study and, depending on the case, may become a tedious balancing act. At the same time, there is no guarantee that an acceptable solution is possible for any selection.

At the end of this section, two more remarks:



(a) Seamless integration of optimized part into the original CAD model.

(b) Resulting surface quality (overall reflection pattern and control points for selected patches).

Figure 4.43: Result of B-Rep morphing in case of the sheet metal part ($\alpha = 10^{-3}$, $\gamma = 500$).

Remark 1

In this section, we used α-regularization and stiffness-based smoothing to smooth the deformed CAD geometry. Note, however, that we may realize a similar smoothing by playing with the degrees of freedom (the refinement) of the CAD model, e.g., by choosing a coarser knot refinement or lower polynomial degrees. Mapping from a rather fine finite element mesh onto a CAD geometry with only reduced degrees of freedom would naturally smooth the displacement field. Such an approach is problematic, though, because:

- Varying the knot refinement and the polynomial degrees to obtain proper smoothing is much more complicated than just choosing a factor α or γ.
- Reducing the degrees of freedom in the CAD model is limited because enough freedom must be left to represent the input deformation in the first place.
- The input CAD model might already be refined such that a coarsening would change the underlying geometry.

Surface smoothing by variation of the refinement is hence not further considered in this work.

Remark 2

When performing a smoothing and accepting a deviation from the input, it remains the question of how the deviations influence the optimization results. In the presented case, for $\alpha = 10^{-3}$ and $\gamma = 500$, the impact on the objective value is negligible as the overall layout of the bead pattern did not change. In general, though, the impact of smoothing on the optimization results strongly depends on the robustness of the optimal design, meaning how sensitive the objective function reacts to modifications in the geometry.

4.9.3 Deformed car door

In this section, we consider a CAD model representing the interior sheet metal structure of a car door (see figure 4.44). In addition to the CAD

Table 4.12: Details on the initial CAD model (not refined) and the finite element mesh in case of the car door.

CAD model (NURBS-based)				mesh	
# patches	degrees	trimming	# control points	# elements	# nodes
851	1 − 5	yes	17 308	158 136	81 061

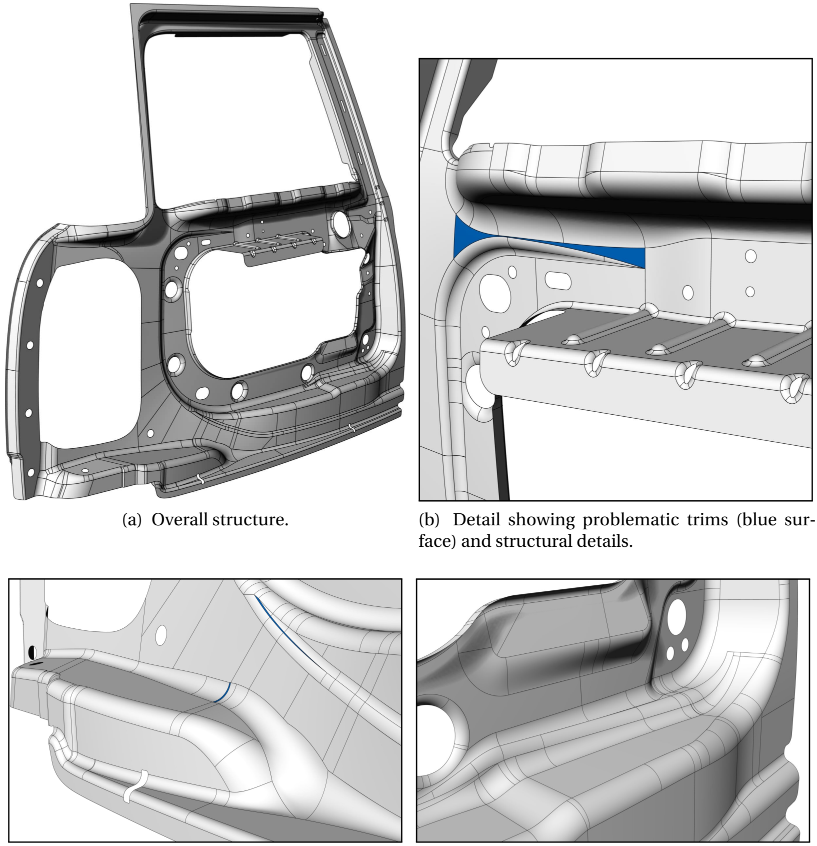
model, we are also given the corresponding finite element mesh used for structural analysis. On the finite element mesh, a displacement field is specified, see figure 4.45(a). The displacement itself represents a simple stretching and bending of the structure starting from the bottom left and becoming more and more pronounced up to the top right. The goal is to map this displacement field from the finite element mesh to the CAD model using B-Rep morphing so that a CAD description of the deformed structure is available.

Unlike the previous cases, the displacement field does not stem from an optimization but is chosen artificially. Its pattern is inspired by patterns that appear in deep-drawing simulation, where CAD models often need to be updated based on the analyzed spring-back. The case shall demonstrate that B-Rep morphing may be useful in various application cases.

A particular challenge with this model is its complexity. It includes almost a thousand fully trimmed patches that comprise both local details and global free-form areas. Other problematic properties are: the included trimming sometimes leads to overly small surface segments, geometric features often involve complex surface joins, the parameterization between individual patches is not matching, and there are chamfers as well as various holes and cutouts (inner loops). Refer to figure 4.44 for impressions.

Table 4.12 provides some details on the CAD model and the finite element mesh.

In the following, we apply B-Rep morphing to update the original CAD model such that it matches the deformed finite element mesh. As the displacement causes a comparatively global shape update, no knot refinement of the CAD model is necessary. Instead, the polynomial degrees of all patches are consistently elevated to $p = q = 5$. Coupling constraints are introduced at all internal edges to maintain the original surface continuity as much as possible. Regularization is activated to control the problem



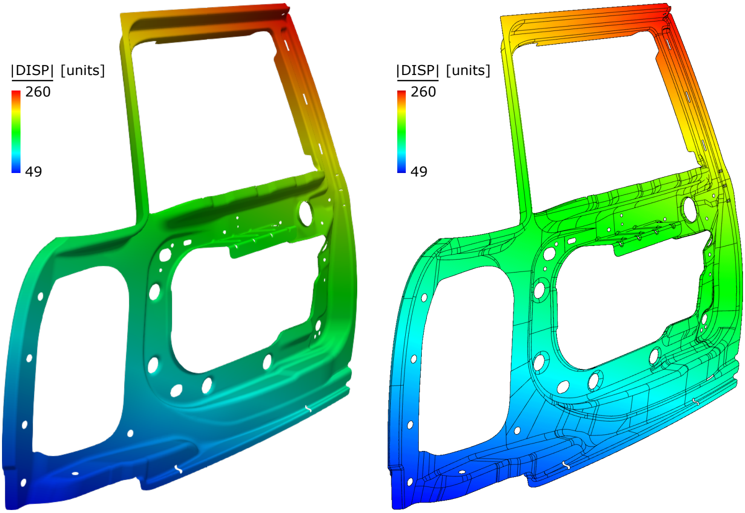
(a) Overall structure.

(b) Detail showing problematic trims (blue surface) and structural details.

(c) Detail showing overly small surfaces (blue) and free-form areas.

(d) Detail showing complex surface joins.

Figure 4.44: CAD geometry of interior sheet metal structure of a car door. Edges are visible, knot lines are hidden.



(a) Input from the finite element mesh (the mesh is hidden to avoid clutter).

(b) Mapped displacement field visualized on the deformed CAD model (surfaces edges are visible, knot lines are hidden).

Figure 4.45: Displacement field as given by the finite element mesh and as obtained after B-Rep morphing. As a reference, the overall height of the structure is ≈ 730 units.

Table 4.13: Settings for B-Rep morphing in case of the car door.

α	β	p_{disp}	p_{rot}	$p_{\text{fix,disp}}$	$p_{\text{fix,norm}}$	γ
10^{-6}	10^{-10}	10^3	10^4	-	-	-

conditioning and to obtain a beneficial layout of the control points. All relevant penalty factors are found by a parameter study starting from the recommendations given in this thesis. The final choice is summarized in table 4.13.

Table 4.14 specifies the resulting problem size. The table states the overall number of control points, the resulting number of unknowns, and all elements generated through the mapping requirements, the regularization, and the coupling constraints.

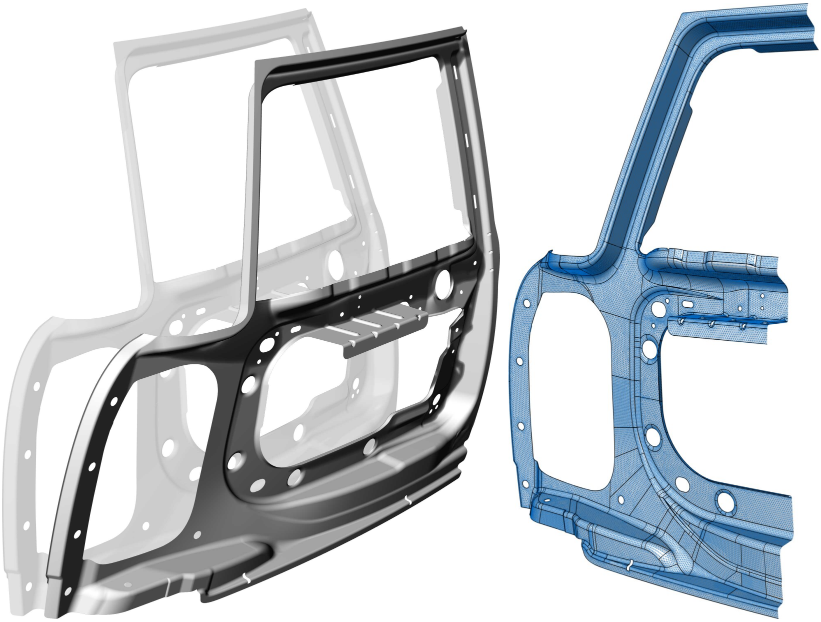
Table 4.14: Problem size after degree elevation in case of the car door.

# control points	# unknowns	# elements
37 425	112 275	45 507

Based on the setup described above, we apply B-Rep morphing to obtain a CAD model that matches the input displacement. Figure 4.45(b) presents the result of the displacement mapping. Comparing this result to the input from the finite element mesh in figure 4.45(a), one can observe an excellent agreement, which qualitatively proves a successful map. The resulting deformation of the CAD model is presented in figure 4.46(a). Herein, one can observe that the model undergoes a significant deformation consisting of a global stretching and bending. This deformation corresponds to the input from the finite element mesh. When overlaying the deformed CAD geometry with the deformed finite element mesh, it becomes obvious that they perfectly match (see the clipped view in figure 4.46(b)).

Despite the large deformation, both pictures do not show any visible defects at first glance, which indicates that the original surface quality is preserved. Investigating further on some geometric details, see figure 4.47, one finds that the resulting geometry practically adopts the quality from the initial design. Figure 4.47(a), for example, presents a watertight model with smooth surfaces. Moreover, figure 4.47(b) shows that structural features, such as chamfers, holes, or other design elements, are well preserved. Also, the complex surface joins are not generating problems so that they are invisible when not highlighting the coupling edges.

Neglecting the deformation and only concentrating on the geometry itself, the deformed CAD model can hardly be distinguished from the original one, highlighting the quality of the result. Only at a few spots one observes wrinkling induced by a locally overestimated penalty factor in the coupling constraints. The by far most pronounced wrinkles are shown in figure 4.47(c). Such a visible defect is typically not acceptable and would require further parameter studies or a locally varying definition of the penalty factor. Both might become difficult, if not impossible, in such a complex model. The search for a finely tuned set of penalty factors is a clear disadvantage of B-Rep morphing in its current form. However, this disadvantage is not necessarily a problem because instead of a tedious



(a) Deformed CAD model vs. original model (transparent).

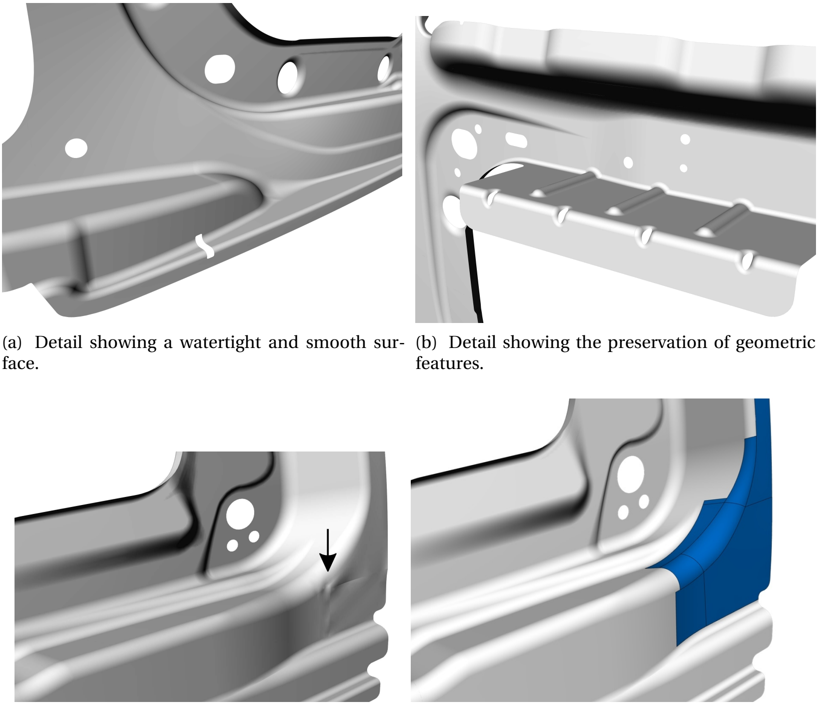
(b) Overlay of deformed mesh (blue) & the deformed CAD model (grey / black) clipped in the middle of the structure.

Figure 4.46: Resulting CAD model after applying B-Rep morphing in case of the car door.

parameter study, one could also fix local defects manually. Given the generally well-matching result, such a local fix is simple and quickly done using standard features in any CAD software. Figure 4.47(d) shows the result of such a quick fix in the case of the car door. In the figure, blue indicates the area where affected surfaces were deleted and recreated based on the surrounding edges.

From all the results discussed above, one can conclude:

- B-Rep morphing may be successfully applied to most complex CAD models.



(a) Detail showing a watertight and smooth surface.

(b) Detail showing the preservation of geometric features.

(c) Detail showing a visible defect.

(d) Manual fix of the visible defect.

Figure 4.47: Details of deformed CAD geometry after B-Rep morphing in case of the car door.

- B-Rep morphing is not limited to an optimization setting but is also useful in other applications.

Remark

The application of B-Rep morphing to many more practical CAD models like in this section revealed the following versatile characteristics:

B-Rep morphing paired with coupling constraints preserves the original surface quality as much as possible. Meaning, e.g., if the original surface is watertight, the deformed CAD model also tends to be watertight. Contrarily, it is irrelevant for the method if the original model is a "dirty" model with badly closed surfaces or gaps. If there are gaps, the corresponding edges appear as regular edges in the CAD model instead of coupling edges. That is, they are ignored in the formulation of coupling constraints. As a result, the deformed CAD model will indeed contain gaps in the same places. Nevertheless, an overall successful deformation of the CAD model is possible.

Also, B-Rep morphing is not sensitive w.r.t. overlapping surfaces. That is, if single faces in the CAD model are partly or entirely overlapping, they will be all updated consistently. A consistent update is possible because integration points are generated for every face in the model individually and then assigned with the corresponding displacement in the finite element mesh. If two faces share a common domain over the finite element mesh, their integration points receive the same update information. As a result, there will still be an overlap in the deformed CAD model, but the overlapping surfaces will likewise approximate the displacement.

Both of the above characteristics greatly add to the robustness of B-Rep morphing when dealing with practical or even "dirty" CAD models.

CHAPTER 5

Conclusions

This thesis presented a method for the grid-based multidisciplinary shape optimization of structures with internal flow and a method to translate optimization results from a grid-based shape optimization back into a CAD model. The thesis included two main parts. The first part covered the optimization, the second part the integration into CAD.

In the first part, we first prepared Vertex Morphing for an application in MDO. Therefore, we extended the formulation of Vertex Morphing to enable the control of volume meshes and the simultaneous control of multiple meshes. A central characteristic of the extended formulation is that it enables a consistent parameterization across multiple non-matching grids. Consistent means that the individual grids follow a consistent description of the underlying geometry and coupled boundaries do not drift apart. This characteristic refutes a widespread assumption that grid-based parameterization was not suited for MDO because of the inherent inconsistency with multiple grids.

In the presented MDO, Vertex Morphing took over three typically separate tasks: the control of the shape, the adaption of the volume mesh of the structure, and the mapping of shape updates or sensitivities between the physical domains. In doing so, the parameterization required no explicit modeling. Instead, it was entirely based on existing meshes and solely relied on a filter radius definition. Generally, Vertex Morphing in the extended form enables a rapid setup of multidisciplinary optimization problems.

Using Vertex Morphing to control volume meshes represents a novelty in literature. In the discussed MDO, we used Vertex Morphing to avoid an extra morphing of the volume mesh of the structure as well as the corresponding sensitivity analysis. The results showed that Vertex Morphing can successfully replace a separate mesh motion strategy. However, more thorough testing is required to evaluate the resulting mesh quality compared to dedicated mesh motion strategies. Nevertheless, we saw that Vertex Morphing is suitable for the control of volume meshes.

Based on the extended Vertex Morphing, we developed a method for grid-based shape optimization of structures with internal flow. The multidisciplinary method allows an optimization of the flow, for example, in terms of pressure loss, while considering stress constraints from the surrounding structure. In doing so, the method considers a one-way fluid-structure interaction. Moreover, it exploits the advantages of Vertex Morphing, i.e., quick setup of the optimization problem and high optimization potential.

As an integral part of the method, this thesis introduced a custom sensitivity analysis. The sensitivity analysis relies on a coupled adjoint approach and follows a partitioned solution strategy. We customized it to the present one-way coupling such that it does not need a computation of cross-derivatives. Instead, it requires the adjoint analysis of an auxiliary force functional. As a result, established single-disciplinary adjoint solvers can be used to realize the *coupled* sensitivity analysis. The latter is a central characteristic of the suggested sensitivity analysis and greatly simplifies its practical application. Another central characteristic is that it was derived independently of the actual formulation of the objective or the constraint. We only assumed specific dependencies on the given state variables and meshes, see (3.47), (3.48), (3.52) and (3.53). Therefore, the presented optimization method applies equally to any other set of response functions with the same properties.

We verified the sensitivity analysis with a 2D and a 3D test case. The values matched the reference values calculated by finite differences. In some cases, we observed minor deviations. They were traced back to the inherent approximation errors in the reference values and an intentionally neglected contribution from the force mapping.

Regarding the impact of the physical coupling on the sensitivity analysis, we saw that neglecting the coupling may considerably alter the sensitivity information. There is a high probability that the resulting deviation will render the sensitivity information useless. The actual influence is case-dependent and determined by the pressure level and the material characteristics of the structure. Therefore, one must carefully examine in the individual case whether the coupling is negligible in order to avoid critical errors.

We tested the overall optimization process on a valve housing with internal flow. The loading of the housing only originated from the interaction with the internal flow. The objective was to minimize the pressure loss in the flow without increasing the stress level in the surrounding structure. With the proposed method, we could find a feasible design with some 16% reduced pressure loss. The improvement is remarkable, considering that the two response functions were strongly conflicting and considering that we did not spend any effort on developing a specific parameterization. The results show that the overall method may be useful for the rapid design optimization of structures with internal flow.

The second part of this thesis covered the development of B-Rep morphing, i.e., a method to deform an existing CAD model to match the results of a simulated deformation (shape change). The method relies on a mapping operation that maps a displacement field from a mesh onto a corresponding CAD model. We extended this mapping operation using several constraints and regularization techniques to account for problems in conjunction with practical CAD models.

Two types of constraints were introduced, coupling constraints and design constraints. Coupling constraints were used to avoid a deterioration of the geometric continuity when deforming the CAD model. They showed to be essential in case of significant shape changes or if the model contains multipatch surfaces or trimming. Design constraints were introduced to enable the enforcement of independent design requirements. The consideration of both types of constraints is a crucial feature of B-Rep morphing.

Regularization was included to improve the problem conditioning as well as the solution quality. A combination of two different techniques was applied. From a practical perspective, the combined approach avoids an unbounded motion of the control points and keeps them as close to the design surface as possible. We could see that these characteristics improve the surface smoothness and facilitate the handling of the CAD model downstream the design process, particularly in trimmed multi-patch models.

Besides the already mentioned regularization, we extended B-Rep morphing by an optional surface smoothing based on thin shell theory. This stiffness-based smoothing represents an effective means to suppress wrinkling. Wrinkling may occur if individual NURBS patches are highly refined or due to the inherent discrepancy between the geometry description in a mesh and a CAD model. The stiffness-based smoothing is especially useful whenever the other regularization techniques do not deliver an acceptable result. Examples showed that the combination of stiffness-based smoothing and coupling constraints enables a smoothing of the surface without scarifying the geometric continuity between individual patches.

Independent of the constraints, we identified a proper refinement of the CAD model as an essential prerequisite to match a simulated displacement. To obtain a proper refinement, we herein used a semi-automatic refinement strategy, which leads to a uniform distribution of knots. The approach showed to be easily applicable and handy. Nevertheless, it requires manual input and tends to generate an overrefined model. Established techniques from CAD may be used to automatize this step further or to make it more adaptive.

The thesis demonstrated the capabilities of B-Rep morphing in various examples. The results in all cases showed a good agreement between the deformed CAD model and the deformed mesh. One could see that the constrained mapping in conjunction with a proper refinement of the original CAD model leads to quality NURBS surfaces even with substantial shape changes. Moreover, we found that smoothing, regularization, and different constraints might be combined as required to obtain a usable CAD model.

The examples also revealed a still existing limitation of B-Rep morphing. Currently, the application of constraints and regularization requires a manual selection of penalty or weighting factors. Indeed, recommendations are given in either case, and the results are convincing once proper factors are identified. However, the necessary parameter study may become troublesome. Follow-up research should investigate strategies to avoid manual parameter selection. Nevertheless, the application of B-Rep morphing in the example of a car door demonstrated that the presented method is mature enough to be used with most complex CAD models.

With B-Rep morphing, it is possible to convert the result of a grid-based shape optimization into a CAD model. Thus, the method allows the seamless integration of Vertex Morphing, or any other grid-based shape optimization, into a CAD workflow. That also means, with B-Rep morphing, it is no longer necessary to use a CAD parametric for shape optimization just because one requires a CAD description of the final design. Instead, one can now test alternative parameterization strategies and still get a CAD model of the final design.

Finally, it is worth noting that, even though B-Rep morphing was developed with optimization in mind, it does not include any specific assumption in this regard. One can apply the method in any case where a simulated deformation must be transferred to an existing CAD model. This characteristic opens the door for a broad field of applications.

APPENDIX A

Further derivations and examples

A.1 Partial derivative of the von Mises stress w.r.t. the state variables of the structure

Let us assume a structure with the state variables **u**. This section presents an approach to analytically compute $\partial\sigma_v/\partial\mathbf{u}$, whereas σ_v represents the equivalent von Mises stress corresponding to a general stress state $\boldsymbol{\sigma}$.

In the finite element method, the general stress state of a structure is computed in a post-processing step using the material matrix $\mathbf{D}$, the shape function derivatives $\mathbf{B}$ and the state variables **u**. We assume that the material matrix does not depend on **u**. In Voigt notation, the relation reads:

$$\boldsymbol{\sigma} = \left[\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{xz}, \tau_{yz}\right]^T = \mathbf{D}\,\mathbf{B}\mathbf{u} \tag{A.1}$$

Based on $\boldsymbol{\sigma}$, the equivalent von Mises stress is computed as follows:

$$
\begin{aligned}
\sigma_v &= \sqrt{\sigma_x^2+\sigma_y^2+\sigma_z^2-\sigma_x\sigma_y-\sigma_x\sigma_z-\sigma_y\sigma_z+3(\tau_{xy}^2+\tau_{xz}^2+\tau_{yz}^2)}\\
\sigma_v &= \sqrt{f(\boldsymbol{\sigma})}
\end{aligned}
\tag{A.2}
$$

Deriving (A.2) w.r.t. a state variable u_i, we obtain:

$$
\begin{aligned}
\frac{\partial\sigma_v}{\partial u_i} = \Bigg[\frac{\partial\sigma_x}{\partial u_i}2\sigma_x+\ldots-\left(\frac{\partial\sigma_x}{\partial u_i}\sigma_y+\frac{\partial\sigma_y}{\partial u_i}\sigma_x\right)-\ldots\\
+\frac{\partial\tau_{xy}}{\partial u_i}6\tau_{xy}+\ldots\Bigg]\frac{1}{2\sqrt{f(\boldsymbol{\sigma})}}
\end{aligned}
\tag{A.3}
$$

From a derivation of (A.1) w.r.t. u_i, we find that all partial derivatives in (A.3) can be computed analytically and in a single operation by evaluating (A.1) for a unit state, in which **u** contains a 1 at the position of u_i and zeros elsewhere. Assuming this unit state, an evaluation of (A.1) will not deliver the actual stresses, but the searched partial derivatives, $\partial\boldsymbol{\sigma}/\partial u_i$. In doing so, no extra implementation is necessary to compute $\partial\boldsymbol{\sigma}/\partial u_i$. Instead, one only needs to call the stress evaluation routine with a modified state variable as input.

Having computed $\partial\boldsymbol{\sigma}/\partial u_i$ analytically, we can compute $\partial\sigma_v/\partial u_i$ according to (A.3). In doing so, we again use the *actual* stress state. Computing (A.3) for all u_i in **u**, one finally obtains $\partial\sigma_v/\partial\mathbf{u}$.

Note, that the derivation presented above holds for all materials that obey Hook's law and where $\mathbf{D}\neq\mathbf{D}(\mathbf{u})$.

A.2 B-Rep morphing applied to a racing car

Without going into detail, this section shows another application example of B-Rep morphing. The example considers a racing car whose chassis was shape optimized using Vertex Morphing. The CAD model of the racing car is visualized in figure A.1. In the figure, the part of the chassis, which was optimized, is highlighted in orange. The optimized part belongs to the car's air guidance.

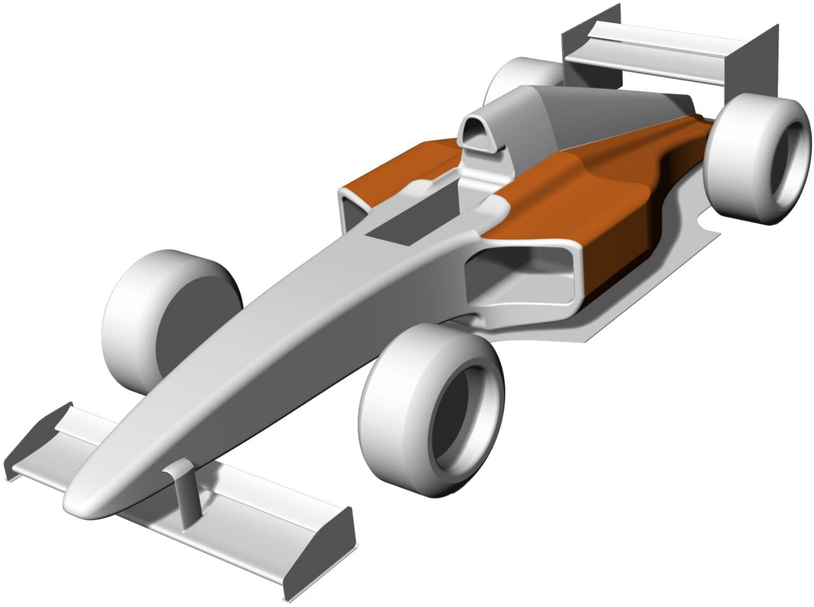


Figure A.1: CAD model of a racing car.

The optimized part of the chassis is presented in figure A.2. The corresponding section of the CAD model consists of 66 trimmed surfaces with each a different parameterization (visible by the non-matching knot-lines in the figure). The individual surfaces are joined along the internal edges so that the model is watertight. Moreover, there are many internal edges where the surface must satisfy at least G^1-continuity.

Based on the CAD model, a mesh was derived for the numerical simulation. The grid-based shape optimization caused this mesh to deform. Figure A.2 shows the surface mesh after the shape optimization. The original mesh matches the original geometry of the chassis. The deformation (shape change) of the mesh is given as a displacement field defined over the original mesh. The shown surface mesh consists of around 32 284 nodes and 63 307 triangles. As one can see in the figure, the optimization introduces significant shape changes.

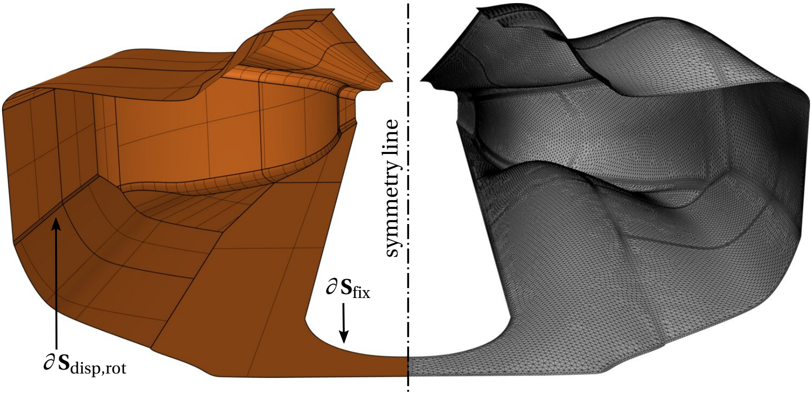

symmetry line
$\partial \mathbf{S}_{\text{fix}}$
$\partial \mathbf{S}_{\text{disp,rot}}$

Figure A.2: Shape optimized part of the chassis. Left: CAD model of the relevant part (thick lines indicate edges, thin lines the knots). Right: Corresponding surface mesh with simulated shape change.

The shape optimization was based on Vertex Morphing. So, the optimization results are given as a surface mesh with an associated displacement field. The goal is now to transfer this optimization result back into the given CAD model so that, eventually, we obtain a CAD model of the optimized design. Therefore we use B-Rep morphing.

In the morphing, we only consider the chassis part, which is subject to shape changes (see figure A.2). Accordingly, we extract the relevant faces of interest from the overall CAD model, map the optimization results from the mesh onto the extracted section, and finally, reinsert the adapted section into the overall model.

To preserve the surface quality as much as possible, we add coupling constraints at all internal edges. Moreover, to ensure that the result seamlessly fits into the overall model, we use fixation constraints and suppress any deformation at all outer edges interfacing with the surrounding car. Both constraints are indicated in figure A.2. We apply α- and β-regularization to improve the problem conditioning and obtain a practical layout of the control points. To properly represent the new shape in the first place, we refine the given CAD model using the herein presented semi-automatic refinement strategy. The result of B-Rep morphing with the specified settings

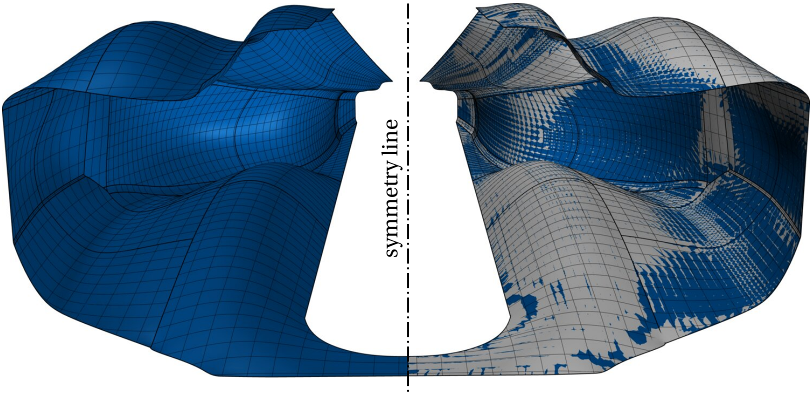


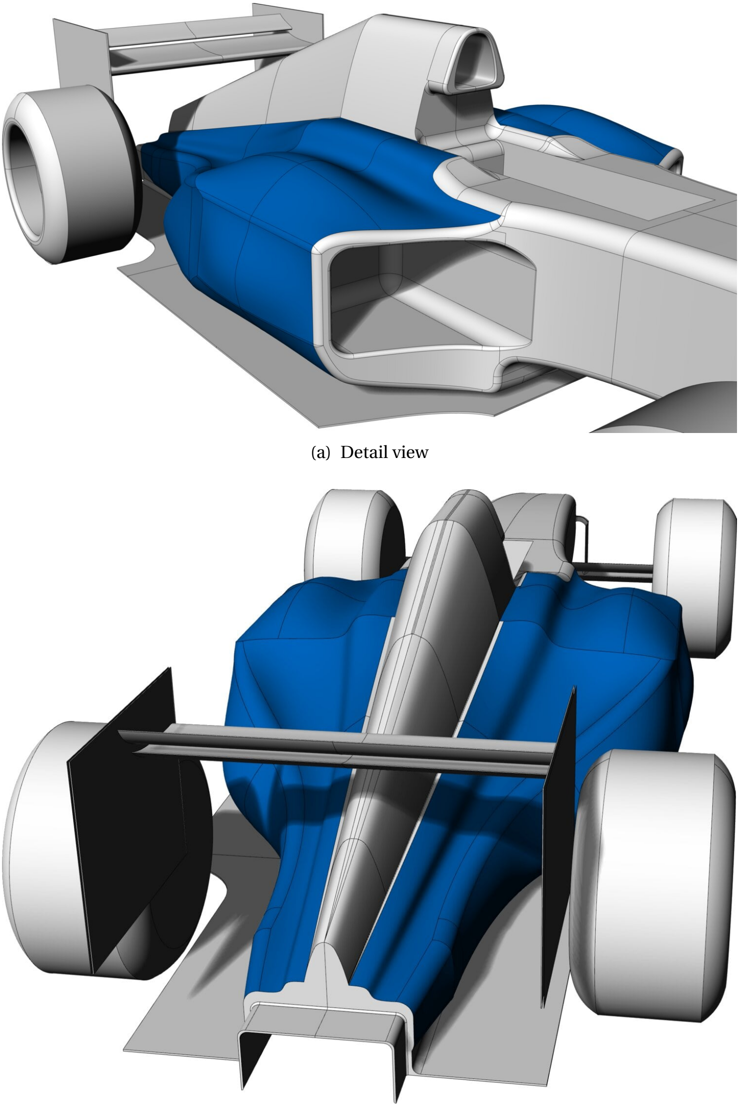
Figure A.3: Results of B-Rep morphing. Left: refined CAD model deformed by B-Rep morphing. Right: Superposition of the deformed CAD model (blue) and the input mesh from the grid-based shape optimization (grey).

is presented in figure A.3.

Note from the result that the CAD model was significantly deformed. As can be seen from the right side of figure A.3, this deformation well matches the input from the grid-based shape optimization. Looking closer at the results, one can also see that there are no gaps, and the surface remains smooth in areas that were already smooth before. This result is a consequence of the specified coupling constraints. Overall, the figure shows that B-Rep morphing allowed a successful conversion of the optimization results to a NURBS-based CAD model.

Given a CAD description of the optimized part, we can reinsert it in the overall racing car model. The result is shown in the figures A.4 and A.5. Since we used fixation constraints, the new design integrates seamlessly into the overall model. As a result, we can readily join all faces describing the car's chassis to a single surface.

The results in this section show that B-Rep morphing allows the seamless integration of Vertex Morphing (or any grid-based shape optimization) into a CAD process. Note that Vertex Morphing allows setting up a shape optimization problem quickly, and B-Rep morphing replaces a



(a) Detail view

(b) Back view

Figure A.4: Reinserted model section with optimized geometry.

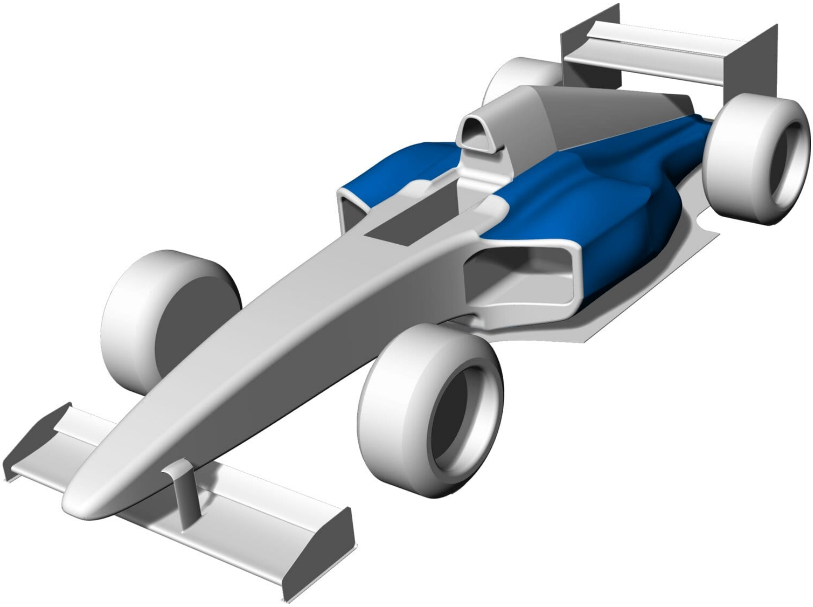


Figure A.5: CAD model of a racing car with shape-optimized chassis.

costly manual conversion of the optimization results into a CAD model. So, combining both methods, one can realize a rapid optimization-driven and CAD-integrated design process.

Bibliography

[1] B. Mohammadi and O. Pironneau. *Applied Shape Optimization for Fluids*. Ed. by A. M. Stuart and E Süli. 2nd ed. Oxford University Press, 2009. ISBN: 9780199546909. DOI: 10.1093/acprof:oso/9780199546909.001.0001.

[2] A. Jameson. "Optimum aerodynamic design using control theory." In: *Computational Fluid Dynamics Review* 3 (1995), pp. 495–528.

[3] H. Azegami and K. Takeuchi. "A smoothing method for shape optimization: Traction method using the Robin condition." In: *International Journal of Computational Methods* 03.01 (2006), pp. 21–33. DOI: 10.1142/S0219876206000709.

[4] M. Scherer, R. Denzer, and P. Steinmann. "A fictitious energy approach for shape optimization." In: *International Journal for Numerical Methods in Engineering* 82.February (2009), pp. 269–302. DOI: 10.1002/nme.2764.

[5] M. Shimoda and Y. Liu. "A non-parametric free-form optimization method for shell structures." In: *Structural and Multidisciplinary Optimization* 50.3 (2014), pp. 409–423. DOI: 10.1007/s00158-014-1059-1.

[6] C. Le, T. Bruns, and D. Tortorelli. "A gradient-based, parameter-free approach to shape optimization." In: *Computer Methods in Applied Mechanics and Engineering* 200.9-12 (2011), pp. 985–996. DOI: 10.1016/j.cma.2010.10.004.

[7] A. Stück and T. Rung. "Adjoint RANS with filtered shape derivatives for hydrodynamic optimisation." In: *Computers and Fluids* 47.1 (2011), pp. 22–32. DOI: 10.1016/j.compfluid.2011.01.041.

[8] K.-U. Bletzinger. “Shape Optimization.” In: *Encyclopedia of Computational Mechanics Second Edition*. Ed. by R. B. E. Stein and T. J. Hughes. Chichester, UK: John Wiley & Sons, Ltd, 2017, pp. 1–42. ISBN: 9781119176817. DOI: `10.1002/9781119176817.ecm2109`.

[9] M Hojjat, E Stavropoulou, and K.-U. Bletzinger. “The Vertex Morphing method for node-based shape optimization.” In: *Computer Methods in Applied Mechanics and Engineering* 268 (2014), pp. 494–513. DOI: `10.1016/j.cma.2013.10.015`.

[10] K.-U. Bletzinger. “A consistent frame for sensitivity filtering and the vertex assigned morphing of optimal shape.” In: *Structural and Multidisciplinary Optimization* 49.6 (2014), pp. 873–895. DOI: `10.1007/s00158-013-1031-5`.

[11] D. Baumgärtner, A. Viti, A. Dumont, G. Carrier, and K.-U. Bletzinger. “Comparison and combination of experience-based parametrization with Vertex Morphing in aerodynamic shape optimization of a forward-swept wing aircraft.” In: *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. Washington, D.C.: American Institute of Aeronautics and Astronautics, 2016, AIAA–2016–3368. ISBN: 978-1-62410-439-8. DOI: `10.2514/6.2016-3368`.

[12] F.-J. Ertl, G. Dhondt, and K.-U. Bletzinger. “Vertex assigned morphing for parameter free shape optimization of 3-dimensional solid structures.” In: *Computer Methods in Applied Mechanics and Engineering* 353 (2019), pp. 86–106. DOI: `10.1016/j.cma.2019.05.004`.

[13] C. Othmer. “Adjoint methods for car aerodynamics.” In: *Journal of Mathematics in Industry* 4.1 (2014), p. 6. DOI: `10.1186/2190-5983-4-6`.

[14] R. Najian Asl, S. Shayegan, A. Geiser, M. Hojjat, and K.-U. Bletzinger. “A consistent formulation for imposing packaging constraints in shape optimization using Vertex Morphing parametrization.” In: *Structural and Multidisciplinary Optimization* 56.6 (2017), pp. 1507–1519. DOI: `10.1007/s00158-017-1819-9`.

[15] J. A. Samareh. “Survey of Shape Parameterization Techniques for High-Fidelity Multidisciplinary Shape Optimization.” In: *AIAA Journal* 39.5 (2001), pp. 877–884. DOI: 10.2514/2.1391.

[16] J. R. R. a. Martins and J. T. Hwang. “Review and Unification of Methods for Computing Derivatives of Multidisciplinary Computational Models.” In: *AIAA Journal* 51.11 (2013), pp. 2582–2599. DOI: 10.2514/1.J052184.

[17] J. E. V. Peter and R. P. Dwight. “Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches.” In: *Computers and Fluids* 39.3 (2010), pp. 373–391. DOI: 10.1016/j.compfluid.2009.09.013.

[18] F. van Keulen, R. T. Haftka, and N. H. Kim. “Review of options for structural design sensitivity analysis. Part 1: Linear systems.” In: *Computer Methods in Applied Mechanics and Engineering* 194.30-33 SPEC. ISS. (2005), pp. 3213–3243. DOI: 10.1016/j.cma.2005.02.002.

[19] A. Fazzolari, N. R. Gauger, and J. Brezillon. “Efficient aerodynamic shape optimization in MDO context.” In: *Journal of Computational and Applied Mathematics* 203 (2007), pp. 548–560. DOI: 10.1016/j.cam.2006.04.013.

[20] S. Nadarajah and A. Jameson. “A Comparison of the Continuous and Discrete Adjoint Approach to Automatic Aerodynamic Optimization.” In: *Proceedings of the 38th AIAA Aerospace Sciences Meeting and Exhibit.* Reno, 2000.

[21] K Maute, M Nikbay, and C Farhat. “Sensitivity analysis and design optimization of three-dimensional non-linear aeroelastic systems by the adjoint method.” In: *International Journal for Numerical Methods in Engineering* 56.6 (2003), pp. 911–933.

[22] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. “High-Fidelity Aerostructural Design Optimization of a Supersonic Business Jet.” In: *Journal of Aircraft* 41.3 (2004), pp. 523–530. DOI: 10.2514/1.11478.

[23] M. Barcelos, H. Bavestrello, and K. Maute. “A Schur-Newton-Krylov solver for steady-state aeroelastic analysis and design sensitivity analysis.” In: *Computer Methods in Applied Mechanics and Engineering* 195.17-18 (2006), pp. 2050–2069. DOI: 10.1016/j.cma.2004.09.013.

[24] G. K. Kenway, G. J. Kennedy, and J. R. Martins. “Scalable parallel approach for high-fidelity steady-state aeroelastic analysis and adjoint derivative computations.” In: *AIAA Journal* 52.5 (2014), pp. 935–951. DOI: 10.2514/1.J052255.

[25] R. Sanchez, T. Albring, R. Palacios, N. R. Gauger, T. D. Economon, and J. J. Alonso. “Coupled adjoint-based sensitivities in large-displacement fluid-structure interaction using algorithmic differentiation.” In: *International Journal for Numerical Methods in Engineering* 113.7 (2018), pp. 1081–1107. DOI: 10.1002/nme.5700. arXiv: 1201.4903.

[26] R. Najian Asl. “Shape optimization and sensitivity analysis of fluids, structures, and their interaction using VertexMorphing parametrization.” Dissertation. Technische Universität München, 2019.

[27] R. Najian Asl, I. Antonau, A. Ghantasala, W. G. Dettmer, R. Wüchner, and K.-U. Bletzinger. “A partitioned scheme for adjoint shape sensitivity analysis of fluid–structure interactions involving non-matching meshes.” In: *Optimization Methods and Software* (2020), pp. 1–31. DOI: 10.1080/10556788.2020.1806275.

[28] S. Xu, W. Jahn, and J.-D. Müller. “CAD-based shape optimisation with CFD using a discrete adjoint.” In: *International Journal for Numerical Methods in Fluids* 74.3 (2014), pp. 153–168. DOI: 10.1002/fld.3844.

[29] T. Verstraete, L. Müller, and J.-D. Müller. “Adjoint-Based Design Optimisation of an Internal Cooling Channel U-Bend for Minimised Pressure Losses.” In: *International Journal of Turbomachinery, Propulsion and Power* 2.2 (2017), p. 10. DOI: 10.3390/ijtpp2020010.

[30] D. Papadimitriou and K. Giannakoglou. “A continuous adjoint method with objective function derivatives based on boundary integrals, for inviscid and viscous flows.” In: *Computers & Fluids* 36.2 (2007), pp. 325–341. DOI: 10.1016/j.compfluid.2005.11.006.

[31] G. Alessi, L. Koloszar, T. Verstraete, and J. P. A. J. van Beeck. “Node-Based Adjoint Surface Optimization of U-Bend Duct for Pressure Loss Reduction.” In: *Computational Methods in Applied Sciences*. Vol. 49. 2019, pp. 61–75. DOI: 10.1007/978-3-319-89890-2_5.

[32] J. P. Heners, L. Radtke, M. Hinze, and A. Düster. “Adjoint shape optimization for fluid–structure interaction of ducted flows.” In: *Computational Mechanics* 61.3 (2018), pp. 259–276. DOI: 10.1007/s00466-017-1465-5.

[33] E. Stavropoulou. “Sensitivity analysis and regularization for shape optimization of coupled problems.” Dissertation. Technische Universität München, 2014.

[34] T. Verstraete, L. Mueller, and J.-D. Mueller. “Multidisciplinary Adjoint Optimization of Trubomachinery Components Including Aerodynamic and Stress Performance.” In: *35th AIAA Applied Aerodynamics Conference*. Denver, Colorado: American Institute of Aeronautics and Astronautics, 2017, pp. 1–12. ISBN: 978-1-62410-501-2. DOI: 10.2514/6.2017-4083.

[35] L. Mueller, T. Verstraete, and M. Schwalbach. “Adjoint-Based Multidisciplinary, Multipoint Optimization of a Radial Turbine Considering Aerodynamic and Structural Performances.” In: *Volume 2D: Turbomachinery*. American Society of Mechanical Engineers, 2019, pp. 1–12. ISBN: 978-0-7918-5858-5. DOI: 10.1115/GT2019-91823.

[36] T. Várady, R. R. Martin, and J. Cox. “Reverse engineering of geometric models - An introduction.” In: *CAD Computer Aided Design* 29.4 (1997), pp. 255–268. DOI: 10.1016/s0010-4485(96)00054-1.

[37] F. Buonamici, M. Carfagni, R. Furferi, L. Governi, A. Lapini, and Y. Volpe. “Reverse engineering modeling methods and tools: a survey.” In: *Computer-Aided Design and Applications* 15.3 (2018), pp. 443–464. DOI: 10.1080/16864360.2017.1397894.

[38] K. H. Chang and C. Chen. “3D shape engineering and design parameterization.” In: *Computer-Aided Design and Applications* 8.5 (2011), pp. 681–692. DOI: 10.3722/cadaps.2011.681-692.

[39] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva. “A Survey of Surface Reconstruction from Point Clouds.” In: *Computer Graphics Forum* 36.1 (2017), pp. 301–329. DOI: 10.1111/cgf.12802.

[40] P. Benkő, R. R. Martin, and T. Várady. “Algorithms for reverse engineering boundary representation models.” In: *Computer-Aided Design* 33.11 (2001), pp. 839–851. DOI: 10.1016/S0010-4485(01)00100-2.

[41] R. Bénière, G. Subsol, G. Gesquière, F. Le Breton, and W. Puech. “A comprehensive process of reverse engineering from 3D meshes to CAD models.” In: *Computer-Aided Design* 45.11 (2013), pp. 1382–1393. DOI: 10.1016/j.cad.2013.06.004.

[42] Y. C. Tsai, C. Y. Huang, K. Y. Lin, J. Y. Lai, and W. D. Ueng. “Development of automatic surface reconstruction technique in reverse engineering.” In: *International Journal of Advanced Manufacturing Technology* 42.1-2 (2009), pp. 152–167. DOI: 10.1007/s00170-008-1586-2.

[43] M. J. Milroy, C. Bradley, G. W. Vickers, and D. J. Weir. “G1 continuity of B-spline surface patches in reverse engineering.” In: *Computer-Aided Design* 27.6 (1995), pp. 471–478. DOI: 10.1016/0010-4485(95)00020-R.

[44] M. Eck and H. Hoppe. “Automatic reconstruction of B-spline surfaces of arbitrary topological type.” In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, SIGGRAPH* (1996), pp. 325–334. DOI: 10.1145/237170.237271.

[45] M. Rouhani, A. D. Sappa, and E. Boyer. "Implicit B-Spline Surface Reconstruction." In: *IEEE Transactions on Image Processing* 24.1 (2015), pp. 22–32. DOI: `10.1109/TIP.2014.2366374`.

[46] G. Becker, M. Schäfer, and A. Jameson. "An advanced NURBS fitting procedure for post-processing of grid-based shape optimizations." In: *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition.* January. Orlando, Florida: American Institute of Aeronautics and Astronautics, 2011. ISBN: 978-1-60086-950-1. DOI: `10.2514/6.2011-891`.

[47] L. Piegl and W. Tiller. "Curve and Surface Fitting." In: *The NURBS Book.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 361–453. ISBN: 978-3-642-97385-7. DOI: `10.1007/978-3-642-97385-7_9`.

[48] D. Brujic, I. Ainsworth, and M. Ristic. "Fast and accurate NURBS fitting for reverse engineering." In: *International Journal of Advanced Manufacturing Technology* 54.5-8 (2011), pp. 691–700. DOI: `10.1007/s00170-010-2947-1`.

[49] R. B. Fisher. "Applying knowledge to reverse engineering problems." In: *Computer-Aided Design* 36.6 (2004), pp. 501–510. DOI: `10.1016/S0010-4485(03)00158-1`.

[50] F. Greco, L. Coox, F. Maurin, V. Balla, E. Deckers, and W. Desmet. "Reverse engineering of deep drawn components with an isogeometric framework." In: *Computers & Structures* 201 (2018), pp. 15–25. DOI: `10.1016/j.compstruc.2018.02.010`.

[51] H. Pottmann and S. Leopoldseder. "A concept for parametric surface fitting which avoids the parametrization problem." In: *Computer Aided Geometric Design* 20.6 (2003), pp. 343–362. DOI: `10.1016/S0167-8396(03)00078-5`.

[52] R. F. Sarraga. "Modifying CAD/CAM surfaces according to displacements prescribed at a finite set of points." In: *CAD Computer Aided Design* 36.4 (2004), pp. 343–349. DOI: `10.1016/S0010-4485(03)00101-5`.

[53] V. Weiss, L. Andor, G. Renner, and T. Várady. "Advanced surface fitting techniques." In: *Computer Aided Geometric Design* 19.1 (2002), pp. 19–42. DOI: `10.1016/S0167-8396(01)00086-3`.

[54] J.-P. Kruth and A Kerstens. “Reverse engineering modelling of free-form surfaces from point clouds subject to boundary conditions.” In: *Journal of Materials Processing Technology* 76.1 (1998), pp. 120–127. DOI: https://doi.org/10.1016/S0924-0136(97)00341-5.

[55] W. Ma and J. P. Kruth. “NURBS curve and surface fitting for reverse engineering.” In: *International Journal of Advanced Manufacturing Technology* 14.12 (1998), pp. 918–927. DOI: 10.1007/BF01179082.

[56] J. Dan and W. Lancheng. “An algorithm of NURBS surface fitting for reverse engineering.” In: *The International Journal of Advanced Manufacturing Technology* 31.1-2 (2006), pp. 92–97. DOI: 10.1007/s00170-005-0161-3.

[57] G. Greiner and K. Hormann. “Interpolating and approximating scattered 3D-data with hierarchical tensor product B-splines.” In: *Surface Fitting and Multiresolution Methods*. Ed. by A. Le Méhauté, C. Rabut, and L. L. Schumaker. Vol. 3. Nashville: Vanderbilt University Press, 1997, pp. 163–172. ISBN: 0826512941.

[58] V Krishnamurthy and M Levoy. “Fitting smooth surfaces to dense polygon meshes.” In: *Computer Graphics and Interactive Techniques, SIGGRAPH, Proceedings of the 23rd annual conference on* (1996), pp. 313–324. DOI: 10.1145/237170.237270.

[59] M. Bartoň, H. Pottmann, and J. Wallner. “Detection and reconstruction of freeform sweeps.” In: *Computer Graphics Forum* 33.2 (2014), pp. 23–32. DOI: 10.1111/cgf.12287.

[60] L. Gaun, D. Bestle, and A. Huppertz. “Hot-to-Cold CAD Geometry Transformation of Aero Engine Parts Based on B-Spline Morphing.” In: *Volume 2B: Turbomachinery*. American Society of Mechanical Engineers, 2014, pp. 1–12. ISBN: 978-0-7918-4561-5. DOI: 10.1115/GT2014-26683.

[61] B. Louhichi, G. N. Abenhaim, and A. S. Tahan. “CAD/CAE integration: updating the CAD model after a FEM analysis.” In: *The International Journal of Advanced Manufacturing Technology* 76.1-4 (2015), pp. 391–400. DOI: 10.1007/s00170-014-6248-y.

[62] C. Farhat, M. Lesoinne, and P. LeTallec. “Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: Momentum and energy conservation, optimal discretization and application to aeroelasticity.” In: *Computer Methods in Applied Mechanics and Engineering* 157.1-2 (1998), pp. 95–114. DOI: 10.1016/S0045-7825(97)00216-8.

[63] T. Wang, R. Wüchner, S. Sicklinger, and K. U. Bletzinger. “Assessment and improvement of mapping algorithms for non-matching meshes and geometries in computational FSI.” In: *Computational Mechanics* 57.5 (2016), pp. 793–816. DOI: 10.1007/s00466-016-1262-6.

[64] J. R. R. A. Martins and A. B. Lambe. “Multidisciplinary Design Optimization: A Survey of Architectures.” In: *AIAA Journal* 51.9 (2013), pp. 2049–2075. DOI: 10.2514/1.j051895.

[65] A. B. Lambe, G. J. Kennedy, and J. R. Martins. “An evaluation of constraint aggregation strategies for wing box mass minimization.” In: *Structural and Multidisciplinary Optimization* 55.1 (2017), pp. 257–277. DOI: 10.1007/s00158-016-1495-1.

[66] T. A. Albring, M. Sagebaum, and N. R. Gauger. “Efficient Aerodynamic Design using the Discrete Adjoint Method in SU2.” In: *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. Washington, D.C.: American Institute of Aeronautics and Astronautics, 2016. ISBN: 978-1-62410-439-8. DOI: 10.2514/6.2016-3518.

[67] R. P. Dwight and J. Brezillon. “Effect of Approximations of the Discrete Adjoint on Gradient-Based Optimization.” In: *AIAA Journal* 44.12 (2006), pp. 3022–3031. DOI: 10.2514/1.21744.

[68] SU2 Developer Community. *SU2: The Open-Source CFD Code*. URL: https://su2code.github.io/ (visited on 10/21/2019).

[69] K.-U. Bletzinger, M. Firl, and F. Daoud. “Approximation of derivatives in semi-analytical structural optimization.” In: *Computers & Structures* 86.13-14 (2008), pp. 1404–1416. DOI: 10.1016/j.compstruc.2007.04.014.

[70] Kratos Developer Community. *Kratos Multiphysics - a framework for building parallel multi-disciplinary simulation software.* 2019. URL: https://github.com/KratosMultiphysics (visited on 06/19/2019).

[71] G. Kreisselmeier and R. Steinhauser. "Systematic Control Design by Optimizing a Vector Performance Index." In: *IFAC Proceedings Volumes* 12.7 (1979), pp. 113–117. DOI: 10.1016/S1474-6670(17)65584-8.

[72] G. A. Wrenn. *An Indirect Method for Numerical Optimization Using the Kreisselmeier-Steinhauser Function.* NASA Contractor Report 4220. Hampton, Virgina: NASA, Office of Management, Scientific and Technical Information Division, 1989.

[73] J. R. R. A. Martins and N. M. K. Poon. "On Structural Optimization Using Constraint Aggregation." In: *Proceedings of the 6th World Congress on Structural and Multidisciplinary Optimization.* May. Rio de Janeiro, Brazil, 2005, pp. 1–10.

[74] P. Dadvand, R. Rossi, and E. Oñate. "An Object-oriented Environment for Developing Finite Element Codes for Multi-disciplinary Applications." In: *Archives of Computational Methods in Engineering* 17.3 (2010), pp. 253–297. DOI: 10.1007/s11831-010-9045-2.

[75] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. "SU2: An Open-Source Suite for Multiphysics Simulation and Design." In: *AIAA Journal* 54.3 (2016), pp. 1–19. DOI: 10.2514/1.J053813.

[76] T. D. Economon. "Simulation and Adjoint-Based Design for Variable Density Incompressible Flows with Heat Transfer." In: *AIAA Journal* 58.2 (2020), pp. 757–769. DOI: 10.2514/1.J058222.

[77] H. Herwig and B. Schmandt. "How to determine losses in a flow field: A paradigm shift towards the second law analysis." In: *Entropy* 16.6 (2014), pp. 2959–2989. DOI: 10.3390/e16062959.

[78] J. B. Rosen. "The Gradient Projection Method for Nonlinear Programming. Part I. Linear Constraints." In: *Journal of the Society for Industrial and Applied Mathematics* 8.1 (1960), pp. 181–217. DOI: 10.1137/0108011.

[79] J. B. Rosen. "The Gradient Projection Method for Nonlinear Programming. Part II. Nonlinear Constraints." In: *Journal of the Society for Industrial and Applied Mathematics* 9.4 (1961), pp. 514–532.

[80] R. T. Haftka and Z. Gürdal. "Elements of Structural Optimization." In: vol. 11. Solid Mechanics And Its Applications. Dordrecht: Springer Netherlands, 1992. Chap. 5. ISBN: 978-0-7923-1505-6. DOI: 10.1007/978-94-011-2550-5.

[81] L. Piegl and W. Tiller. *The NURBS Book*. 2nd ed. Monographs in Visual Communications. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995. ISBN: 978-3-642-97387-1. DOI: 10.1007/978-3-642-97385-7.

[82] M. Breitenberger, A. Apostolatos, B. Philipp, R. Wüchner, and K. U. Bletzinger. "Analysis in computer aided design: Nonlinear isogeometric B-Rep analysis of shell structures." In: *Computer Methods in Applied Mechanics and Engineering* 284 (2015), pp. 401–457. DOI: 10.1016/j.cma.2014.09.033.

[83] T. Teschemacher, A. M. Bauer, T. Oberbichler, M. Breitenberger, R. Rossi, R. Wüchner, and K. U. Bletzinger. *Realization of CAD-integrated shell simulation based on isogeometric B-Rep analysis*. Vol. 5. 1. Springer International Publishing, 2018. ISBN: 0123456789. DOI: 10.1186/s40323-018-0109-4.

[84] GrabCAD Inc. *GrabCAD: Design Community, CAD Library, 3D Printing Software*. URL: https://grabcad.com/ (visited on 05/31/2019).

[85] M. Breitenberger. "CAD-Integrated Design and Analysis of Shell Structures." Dissertation. Technische Universität München, 2016.

[86] T. J. Hughes, J. A. Cottrell, and Y. Bazilevs. "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement." In: *Computer Methods in Applied Mechanics and Engineering* 194.39-41 (2005), pp. 4135–4195. DOI: 10.1016/j.cma.2004.10.008.

[87] L. Kudela, N. Zander, T. Bog, S. Kollmannsberger, and E. Rank. "Efficient and accurate numerical quadrature for immersed boundary methods." In: *Advanced Modeling and Simulation in Engineering Sciences* 2.1 (2015), p. 10. DOI: 10.1186/s40323-015-0031-y.

[88] T. Oberbichler. "Entwicklung eines hierarchischen Plattenelements." Masters's thesis. Technische Universität München, 2017.

[89] T. Oberbichler. *ANurbs.* URL: https://github.com/oberbichler/ANurbs (visited on 01/23/2020).

[90] A. Apostolatos, G. De Nayer, K. U. Bletzinger, M. Breuer, and R. Wüchner. "Systematic evaluation of the interface description for fluid–structure interaction simulations using the isogeometric mortar-based mapping." In: *Journal of Fluids and Structures* 86 (2019), pp. 368–399. DOI: 10.1016/j.jfluidstructs.2019.02.012.

[91] A. Apostolatos. "Isogeometric Analysis of Thin-Walled Structures on Multipatch Surfaces in Fluid-Structure Interaction." Dissertation. Technische Universität München, 2019.

[92] E. Rank, S. Kollmannsberger, C. Sorger, and A. Düster. "Shell Finite Cell Method: A high order fictitious domain approach for thin-walled structures." In: *Computer Methods in Applied Mechanics and Engineering* 200.45-46 (2011), pp. 3200–3209. DOI: 10.1016/j.cma.2011.06.005.

[93] D. Brujic, M. Ristic, and I. Ainsworth. "Measurement-based modification of NURBS surfaces." In: *CAD Computer Aided Design* 34.3 (2002), pp. 173–183. DOI: 10.1016/S0010-4485(01)00060-4.

[94] G. Farin. "8 - B-Spline Curves." In: *Curves and Surfaces for CAGD (Fifth Edition).* Ed. by G. Farin. Fifth Edit. The Morgan Kaufmann Series in Computer Graphics. San Francisco: Morgan Kaufmann, 2002, pp. 119–146. ISBN: 978-1-55860-737-8. DOI: https://doi.org/10.1016/B978-155860737-8/50008-9.

[95] P. Benko, G. Kós, T. Várady, L. Andor, and R. Martin. "Constrained fitting in reverse engineering." In: *Computer Aided Geometric Design* 19.3 (2002), pp. 173–205. DOI: 10.1016/S0167-8396(01)00085-1.

[96] A. Apostolatos, K.-U. Bletzinger, and R. Wüchner. "Weak imposition of constraints for structural membranes in transient geometrically nonlinear isogeometric analysis on multipatch surfaces." In: *Computer Methods in Applied Mechanics and Engineering* 350 (2019), pp. 938–994. DOI: https://doi.org/10.1016/j.cma.2019.01.023.

[97] T. Oberbichler, R. Wüchner, and K.-U. Bletzinger. "Benefits of Automatic differentiation in Finite Element Analysis." In: *5th ECCOMAS Young Investigators Conference.* Krakow (Poland), 2019.

[98] J. Fike and J. Alonso. "The Development of Hyper-Dual Numbers for Exact Second-Derivative Calculations." In: *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition.* January. Orlando, Florida: American Institute of Aeronautics and Astronautics, 2011. ISBN: 978-1-60086-950-1. DOI: 10.2514/6.2011-886.

[99] J. Fike. "Multi-objective optimization using hyper-dual numbers." Dissertation. Stanford University, 2013.

[100] T. Oberbichler. *Hyperjet.* URL: https://github.com/oberbichler/HyperJet (visited on 12/10/2019).

[101] G. Greiner. "Variational Design and Fairing of Spline Surfaces." In: *Computer Graphics Forum* 13.3 (1994), pp. 143–154. DOI: 10.1111/1467-8659.1330143.

[102] J. Kiendl, K. U. Bletzinger, J. Linhard, and R. Wüchner. "Isogeometric shell analysis with Kirchhoff-Love elements." In: *Computer Methods in Applied Mechanics and Engineering* 198.49-52 (2009), pp. 3902–3914. DOI: 10.1016/j.cma.2009.08.013.

Bisherige Titel der Schriftenreihe

Band	Titel
1	Frank Koschnick, *Geometrische Lockingeffekte bei Finiten Elementen und ein allgemeines Konzept zu ihrer Vermeidung*, 2004.
2	Natalia Camprubi, *Design and Analysis in Shape Optimization of Shells*, 2004.
3	Bernhard Thomée, *Physikalisch nichtlineare Berechnung von Stahlfaserbetonkonstruktionen*, 2005.
4	Fernaß Daoud, *Formoptimierung von Freiformschalen - Mathematische Algorithmen und Filtertechniken*, 2005.
5	Manfred Bischoff, *Models and Finite Elements for Thin-walled Structures*, 2005.
6	Alexander Hörmann, *Ermittlung optimierter Stabwerkmodelle auf Basis des Kraftflusses als Anwendung plattformunabhängiger Prozesskopplung*, 2006.
7	Roland Wüchner, *Mechanik und Numerik der Formfindung und Fluid-Struktur-Interaktion von Membrantragwerken*, 2006.
8	Florian Jurecka, *Robust Design Optimization Based on Metamodeling Techniques*, 2007.
9	Johannes Linhard, *Numerisch-mechanische Betrachtung des Entwurfsprozesses von Membrantragwerken*, 2009.
10	Alexander Kupzok, *Modeling the Interaction of Wind and Membrane Structures by Numerical Simulation*, 2009.
11	Bin Yang, *Modified Particle Swarm Optimizers and their Application to Robust Design and Structural Optimization*, 2009.

Band	**Titel**
12	Michael Fleischer, *Absicherung der virtuellen Prozesskette für Folgeoperationen in der Umformtechnik*, 2009.
13	Amphon Jrusjrungkiat, *Nonlinear Analysis of Pneumatic Membranes - From Subgrid to Interface*, 2009.
14	Alexander Michalski, *Simulation leichter Flächentragwerke in einer numerisch generierten atmosphärischen Grenzschicht*, 2010.
15	Matthias Firl, *Optimal Shape Design of Shell Structures*, 2010.
16	Thomas Gallinger, *Effiziente Algorithmen zur partitionierten Lösung stark gekoppelter Probleme der Fluid-Struktur-Wechselwirkung*, 2011.
17	Josef Kiendl, *Isogeometric Analysis and Shape Optimal Design of Shell Structures*, 2011.
18	Joseph Jordan, *Effiziente Simulation großer Mauerwerksstrukturen mit diskreten Rissmodellen*, 2011.
19	Albrecht von Boetticher, *Flexible Hangmurenbarrieren: Eine numerische Modellierung des Tragwerks, der Hangmure und der Fluid-Struktur-Interaktion*, 2012.
20	Robert Schmidt, *Trimming, Mapping, and Optimization in Isogeometric Analysis of Shell Structures*, 2013.
21	Michael Fischer, *Finite Element Based Simulation, Design and Control of Piezoelectric and Lightweight Smart Structures*, 2013.
22	Falko Hartmut Dieringer, *Numerical Methods for the Design and Analysis for Tensile Structures*, 2014.
23	Rupert Fisch, *Code Verification of Partitioned FSI Environments for Lightweight Structures*, 2014.
24	Stefan Sicklinger, *Stabilized Co-Simulation of Coupled Problems Including Fields and Signals*, 2014.

Band	Titel
25	Majid Hojjat, *Node-based parametrization for shape optimal design*, 2015.
26	Ute Israel, *Optimierung in der Fluid-Struktur-Interaktion - Sensitivitätsanalyse für die Formoptimierung auf Grundlage des partitionierten Verfahrens*, 2015.
27	Electra Stavropoulou, *Sensitivity analysis and regularization for shape optimization of coupled problems*, 2015.
28	Daniel Markus, *Numerical and Experimental Modeling for Shape Optimization of Offshore Structures*, 2015.
29	Pablo Suárez, *Design Process for the Shape Optimization of Pressurized Bulkheads as Components of Aircraft Structures*, 2015.
30	Armin Widhammer, *Variation of Reference Strategy - Generation of Optimized Cutting Patterns for Textile Fabrics*, 2015.
31	Helmut Masching, *Parameter Free Optimization of Shape Adaptive Shell Structures*, 2016.
32	Hao Zhang, *A General Approach for Solving Inverse Problems in Geophysical Systems by Applying Finite Element Method and Metamodel Techniques*, 2016.
33	Tianyang Wang, *Development of Co-Simulation Environment and Mapping Algorithms*, 2016.
34	Michael Breitenberger, *CAD-integrated Design and Analysis of Shell Structures*, 2016.
35	Önay Can, *Functional Adaptation with Hyperkinematics using Natural Element Method: Application for Articular Cartilage*, 2016.
36	Benedikt Philipp, *Methodological Treatment of Non-linear Structural Behavior in the Design, Analysis and Verification of Lightweight Structures*, 2017.
37	Michael Andre, *Aeroelastic Modeling and Simulation for the Assessment of Wind Effects on a Parabolic Trough Solar Collector*, 2018.

Band	Titel
38	Andreas Apostolatos, *Isogeometric Analysis of Thin-Walled Structures on Multipatch Surfaces in Fluid-Structure Interaction*, 2018.
39	Altuğ Emiroğlu, *Multiphysics Simulation and CAD-Integrated Shape Optimization in Fluid-Structure Interaction*, 2019.
40	Mehran Saeedi, *Multi-Fidelity Aeroelastic Analysis of Flexible Membrane Wind Turbine Blades*, 2017.
41	Reza Najian Asl, *Shape Optimization and Sensitivity Analysis of Fluids, Structures, and their Interaction Using Vertex Morphing Parametrization*, 2019.
42	Ahmed Abodonya, *Verification Methodology for Computational Wind Engineering Prediction of Wind Loads on Structures*, 2020.
43	Anna Maria Bauer, *CAD-Integrated Isogeometric Analysis and Design of Lightweight Structures*, 2020.
44	Andreas Winterstein, *Modeling and Simulation of Wind-Structure Interaction of Slender Civil Engineering Structures Including Vibration Mitigation Systems*, 2020.
45	Franz-Josef Ertl, *Vertex Morphing for constrained shape optimization of three-dimensional solid structures*, 2020.