Direct simulation on geometric representations with the finite cell method

Flawed B-reps, CSG models, and V-reps for functionally graded materials

Benjamin Paul Thomas Wassermann, M.Sc.

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

Dr.-Ing.

genehmigten Dissertation.

Die Dissertation wurde am 29.07.2020 bei der Technischen Universität München eingereicht und durch die Ingenieurfakultät Bau Geo Umwelt am 11.03.2021 angenommen.

To Bea, Lea and Kira

Zusammenfassung

Diese Doktorarbeit befasst sich mit der direkten numerischen Simulation von verschiedenen Computer-Aided Design (CAD) Repräsentationen mit der Finiten Zellen Methode (FCM). Die FCM ist eine Variante der Finiten Elemente Methode, die auf einer fiktiven Domäne arbeitet und Ansatzfunktionen höherer Ordnung verwendet. Die einzige geometrische Information, die die FCM dafür benötigt, ist eine eindeutige Aussage darüber, ob ein Punkt sich innerhalb oder außerhalb eines CAD-Modells befindet. Diese geometrische Flexibilität wurde genutzt, um die FCM um drei weitere CAD Repräsentationen zu erweitern. Hierfür wurden jeweils speziell zugeschnittene Methoden zur Klassifizierung der Punktezugehörigkeit (Point Membership Classification: PMC) entwickelt.

Die erste Erweiterung umfasst die direkte Simulation von volumenbasierten prozeduralen Geometriebeschreibungen, sowie von Constructive Solid Geometries (CSG). Der entwickelte PMC Test nutzt dabei die Kenntnis des Konstruktionsablaufs, indem die Frage der Punktzugehörigkeit durch den Konstruktionsbaum gereicht und schlussendlich den Grundkörpern gestellt wird. Maßgeschneiderte PMC Tests können die einzelnen Grundkörper schnell und genau auswerten. Die Ergebnisse werden anschließend durch den Baum zurückgeschickt und entsprechend der Operationen auf den jeweiligen Knoten verarbeitet, um schlussendlich eine eindeutige und genaue Aussage über die Punktzugehörigkeit zu erhalten.

Die zweite Erweiterung ermöglicht der FCM eine direkte Simulation von fehlerhaften Boundary-Representation (B-rep) Modellen. Hierfür wird die ungültige Geometrie durch eine fehlergrößenabhängige Baumstruktur approximiert. Da eine PMC auf diesem Baum nur ein grobes Ergebnis liefert, wird in der Nähe des Randes ein weiterer Test mit Raytracing ausgeführt. Dieser zweistufige Test liefert exakte Ergebnisse für fehlerfreie Bereiche und eine sehr gute Näherung in der Umgebung der Fehler.

Schließlich wird die FCM um volumetrische Modelle (V-Modelle) erweitert. Hierbei steht die Simulation verschiedenartiger functionally graded materials (FGM) im Vordergrund. Zudem wird auf Grundlage der Homogenisierung eine Methode vorgestellt, die eine Modellierung und Simulation von großskaligen, funktionell variierenden Mikrostrukturen ermöglicht.

Anhand zahlreicher Beispiele werden die verschiedenen Methoden verifiziert und deren Eignung für praktische Anwendungen verdeutlicht.

Abstract

This thesis's central objective is the direct numerical simulation of different computer-aided design (CAD) representations using the finite cell method (FCM). The FCM is a variant of the finite element method that uses a fictitious domain and high-order shape functions. The only geometric information needed by the FCM is an unambiguous statement about whether a point is inside or outside of a CAD model. Based on this geometrical flexibility, the FCM was extended to three different CAD representation schemes by developing individual point classification (PMC) methods.

The first extension involves the direct simulation of solid-based procedural and constructive solid geometry (CSG) models. The developed PMC test for these models makes full use of the knowledge about the models' construction history. For this, the question of point membership is passed through the construction tree and finally posed to the primitives. Tailored PMC tests for the individual primitives yield fast and accurate results, which are then transferred back and treated according to the respective operations.

The second extension enables the FCM to simulate on flawed boundary representation (B-rep) models directly. For this, the invalid geometry is approximated by a flaw-size dependent space-tree. Since a PMC on this tree yields only an inaccurate result, a secondary PMC based on ray-tracing is carried out close to the boundary. This two-stage PMC yields exact results in flawless regions and excellent approximations in the vicinity of the flaws.

Finally, the FCM is extended to volumetric models (V-models). Here, a particular focus is laid on the simulation of different types of functionally graded materials (FGM). Additionally, based on homogenization, a method is presented that allows to model and simulate large-scale functionally graded microstructures.

Numerous examples prove the validity of the particular methods and illustrate the applicability for practical applications.

Acknowledgments

This thesis results from the scientific research I conducted as a member of the Simulation of Applied Mechanics group during my time at the chair of Computation in Engineering at the Technical University of Munich from November 2014 to March 2020.

First of all, I would like to thank my doctoral supervisor, Prof. Dr. rer. nat. Ernst Rank for his constant support, his guidance, and the trust he set in me. Having always the bigger picture in mind, he ensured that this thesis succeeds. I also want to express my gratitude for the financial supports by the *German Research Foundation* under the grants Ra624/22-1 and Ra624/22-2.

Furthermore, I would like to thank Dr.-Ing. habil. Stefan Kollmannsberger who always had an open door and never got tired to explain and discuss all the little problems and aspects with me. Innumerable ideas and solutions arose from these fruitful discussions. Also, I thank all my colleagues. It was a real pleasure to work with all of them in the past years. I will miss the open and friendly working climate and reliability among us.

I thank Prof. Gershon Elber from the Computer Science Department at Technion - Israel Institute of Technology for the friendly and fruitful collaboration and support during the integration of the V-rep framework and the joint publication.

I also thank Prof. Dr.-Ing. Alexander Düster and Prof. Dr.-Ing. Kai-Uwe Bletzinger for reviewing my thesis and Prof. Dr.-Ing. André Borrmann for chairing my examination.

Finally, I want to thank my family, and in particular, my parents without whom I would not have found my profession and who always had faith in me. Above all, I thank my wife, Beatrice, for her never-ending support and love for me.

Contents

Introduction

Motivation

An engineer's task is the "application of science and mathematics by which the properties of matter and the sources of energy in nature are made useful to people." The result is "the design and manufacture of complex products" [1]. Nowadays, products are predominantly developed with *computer-aided engineering* (CAE), which comprises the product design with *computer-aided design* (CAD) and various numerical analyses depending on the physical phenomena of interest and relevance. Most physical phenomena, such as solid and fluid mechanics, heat transfer, acoustics, etc. are described mathematically by *partial differential equations* (PDE). As these PDEs are, in general, impossible to solve analytically, techniques were developed to solve complex problems discretely, such as the *finite difference method*, the *finite volume method*, the *boundary element method*, or the *finite element method* (FEM) of which a variant will be addressed within this thesis. Although the fundamentals were already laid in the 19th century, the FEM was not reasonably applicable until the 1950s [2], when the advances in the computer design allowed to solve large mathematical problems. Nowadays, the FEM is the method of choice to solve various engineering problems, e.g., structural and dynamic analyses, topology optimization, heat transfer, contact, fluid flow, fluid-structure interaction, etc. In its core, the FEM subdivides a complex domain into a finite number of elements. On each element, shape functions are employed, which locally approximate the exact solution of the PDE. In contrast to finite differences, the FEM merely approximates the solution in an integral, 'weak' sense. To this end, an integration over all finite elements is carried out, and the internal energy – e.g., the elastic strain energy – is set in balance to the external work – e.g., due to external forces acting on the displacements.

Independent from the numerical analysis methods, computer-aided design evolved as *the* technology for the computer-based creation and modification of two- and three-dimensional geometrical models. Modern CAD modeling systems rely basically on the two different representation techniques: *Boundary representation* (B-rep), and *solid-based procedural modeling*, where the latter follows and extends the *constructive solid geometry* (CSG) idea [3, 4, 5]. As the name suggests, B-rep describes a body by its boundary surfaces in 3D or curves in 2D. B-rep offers several advantages, first and foremost a great freedom in the creation of free-form shapes. However, it is explicitly prone to a large variety of geometrical and topological flaws. By contrast, solid-based procedural and CSG models represent a complex model with primitives that are combined with the Boolean or similar operations. Due to the closed analytical description of the primitives and operations, these models are far less sensitive towards all kinds of flaws. On the other hand, the amount of possible shapes is more limited, which restricts the fields of applications. Both geometric representation schemes, B-rep and solid-based procedural/CSG, cannot decently represent *functionally graded materials* (FGM). Yet, with the emerging of *additive manufacturing* (AM), as a novel production method, the development of representation schemes for FGM becomes more and more imperative. This issue is addressed by the *volumetric representation* (V-rep) framework that was recently proposed by Elber et al. [6]. V-models are composed of volumetric, non-singular spline primitives. The explicit volume description, in terms of splines, is thereby able to represent FGMs. Similar to CSG, these primitives are combined with the Boolean operations and preserve the inherent robustness towards flaws. Additionally, the freedom in design is augmented with additional construction rules for primitives.

The independent development of CAD and numerical analysis has led to an incompatibility between the corresponding geometric representation schemes, which causes one of the main bottlenecks in product development. At each iteration in the product design cycle, the CAD model needs to be converted into an analysis-suitable format for downstream CAE applications, such as numerical simulations, shape/topology optimizations, rapid prototyping, or automated manufacturing. Subsequently, the respective results need to be transferred back to the original or modified CAD model. In an analysis with classical FEM, this transition corresponds to a boundary conforming meshing process. The meshing requires a watertight and mathematically valid model. Hence, before meshing is even possible, many CAD models need a preceding geometry healing, which is hardly automatable. Even for an accurate geometry, the meshing of arbitrarily complex models is non-trivial and error-prone, especially in three dimensions. Additionally, automatically generated meshes are not necessarily analysis-suitable, as the mesh is often inappropriately dense or coarse, or elements are not suitable (e.g., due to bad aspect ratios, vanishing Jacobian, etc.). Such unsuitable meshes can entail an elaborate mesh healing procedure, which is again, in general, not fully automatable. A genuinely smooth transition between geometric and computational models is still an unsolved problem. A study in the U.S. car industry concluded that this transition process can be accounted for up to 80% of the overall analysis time [7].

In the last two decades, intensive research was conducted to find alternative approaches that avoid or shorten this costly transition process. Momentarily, the most prominent approach is the *isogeometric analysis*(IGA) that uses the same *B-spline* or *NURBS* shape functions for the geometric modeling, as for the numerical simulations [8, 9, 10, 11]. IGA was applied very successfully to many different geometries, especially to (trimmed) shell, or membranes structures [12]. However, IGA is strongly dependent on the quality of the underlying CAD models. Since many real-world CAD geometries are mathematically invalid and thus not directly suitable, IGA cannot entirely evade a transition process.

In contrast to IGA, the *finite cell method* (FCM) is capable of carrying out a straightforward numerical simulation on all kinds of (potentially) flawed geometric models. The versatility of the FCM is because the FEM discretization is independent of the geometric model. The FCM recovers the exact shape of the geometry during the integration of the internal energy over the domain. For this, it is sufficient to determine whether a point is inside or outside of the geometric model. Depending on the respective geometry representation, this *point membership classification* (PMC) needs to be carried out differently. The PMC can be modified to provide material properties, too. Thus, the FCM easily extends to functionally graded materials, making it ideally suited for additive manufacturing simulations.

Scope

The objective of this thesis is the extension of the finite cell method to three different geometric CAD representation schemes: (i) flawed B-reps, (ii) solid-based procedural/CSG models, and (iii) functionally graded materials based on V-rep models. Each geometrical representation scheme requires its own problem-specific point inclusion test. For this, the respective underlying geometric representations are analyzed and described. Special emphasis is put on the classification and quantification of the different geometrical and topological flaws, and their potential to obstruct a subsequent numerical simulation. This categorization allows the formulation and development of PMC methods for flawed B-rep and solid-based procedural/CSG models. Additionally, the FCM is connected to the V-rep framework and, thus, extends to different kinds of functionally graded materials. Based on homogenization, an approach is developed that allows to model and simulate large-scale functionally graded microstructures.

Outline

The thesis is structured as follows: Part I provides all necessary concepts for numerical simulations with the FCM. For this, Chapter 1 introduces the basic concepts of the continuum mechanics – exemplary for linear-elasticity. In Chapter 2, the finite element method is briefly recalled and, in Chapter 3, extended to higher-order hierarchical shape functions. Chapter 4 addresses the concepts and characteristics of the finite cell method, whose convergence properties are discussed in Chapter 5. Part II covers computer-based geometrical modeling, with focus on CAD representations. At first, Chapter 6 introduces *splines* as the basic building blocks for most CAD representations. Then, in Chapter 7, the individual representation schemes are presented and, in Chapter 8, investigated concerning their vulnerability towards various flaws. In Chapter 9, functionally graded materials, and possible representation schemes are discussed. The point membership classifications for different CAD representation schemes are presented in Part III. The PMCs for valid and flawed B-rep models can be found in Chapter 10, for the solid-based procedural/CSG models in Chapter 11, and for the V-models in Chapter 12, which explains also the extensions of the FCM regarding functionally graded materials. Part IV presents several examples for each of the geometrical representation schemes. The three examples in Chapter 13 illustrate the robustness of the PMC for flawed B-rep models. Chapter 14 includes four examples that show the PMC for the individual primitives and operations and highlight the flexibility of the FCM regarding various geometrical and topological changes. The first two examples in Chapter 15 treat multi-material FGMs. The remaining three examples illustrate the transition from a fully-resolved single-material FGM (i.e., a microstructure) to a simulation with a homogenized graded material.

Citations

This thesis is based primarily on the three papers

1. B. Wassermann, S. Kollmannsberger, T. Bog, and E. Rank, "From geometric design to numerical analysis: A direct approach using the Finite Cell Method on Constructive Solid Geometry" [13],

2. B. Wassermann, S. Kollmannsberger, S. Yin, L. Kudela, and E. Rank, "Integrating CAD and numerical analysis: 'Dirty geometry' handling using the Finite Cell Method" [14], and

3. B. Wassermann, N. Korshunova, S. Kollmannsberger, E. Rank, and G. Elber, "Direct simulation of functionally graded materials using V-Models and the Finite Cell Method" [15].

Several text passages and/or images are taken directly from one of these papers. Others are more or less modified, adapted, and extended. Since a highlighting of these self-citations within the thesis would severely impede the reading flow and lucidity, the corresponding chapters and sections are substitutionally listed once in the following Table 1.

Paper	Chapters/Sections
1. Solids/CSG: [13]	7.2, 11, 14.1, 14.2, 14.3
2. Flawed B-rep: [14]	7.1, 8.3, 10.2, 13
3. V-models/FGM: [15]	7.3, 9.2, 12, 15

Table 1: Self-citations.

Part I

Numerical simulation

The first part gives an overview of numerical simulation methods for engineering problems in mechanics and, in particular, provides all necessary parts to understand the concept of the *finite cell method* (FCM). FCM is a variant of the *finite element method* (FEM) and combines a fictitious domain approach with higher-order finite elements. Within the scope of this thesis, FCM is mainly applied to linear elastic problems. Therefore, all required equations and concepts are briefly derived for the linear elastic version of the FCM. Linear elasticity is a simplification of the non-linear *continuum mechanics*, assuming small deformations and stresses below the yield point [16, 17]. Continuum mechanics provide a mathematical way to describe the stresses and strains inside different materials due to external or internal loads and deformations. Based on the continuum mechanics approach, the kinematic, constitutive, and equilibrium equations are derived, which yield the strong form of the *partial differential equations* (PDE), describing linear elastic problems.

For the FEM formulation, these differential equations need to be converted into their equivalent weak form. The FEM is a method that seeks to find an approximate solution on a finite number of elements. To this end, the weak form of the PDEs and the domain are discretized with shape functions and a conforming mesh, respectively. Following this, the FEM is extended to the FCM, which involves introducing higher-order shape functions and the fictitious domain approach. Both offer numerous advantages over the low-order FEM. However, the latter implies the necessity of special quadrature rules and methods for applying boundary conditions in a weak sense. In this context it is noteworthy that also other direct approaches require adapted integration rules. For instance, the *isogeometric B-Rep analysis* (IBRA) – an extension to IGA – uses reparametrization to resolve trimmed surfaces [18, 19].

Chapter 1

Continuums Mechanics

In this section, the governing equations for the continuums approach are derived for linear elasticity. As the name indicates, continuum mechanics assumes a continuous maIn the case of Lagrange and Legendre terial. This assumption is not entirely accurate as all materials are at some level composed of substructures. These can be grains, fibers on the mesoscale, or atoms and molecules on the microscale. However, most engineering problems are situated on the macroscale, and the predictions for continuous materials agree well with experimental measurements and experiments [20]. The derivation of the respective equations and the notation in this chapter mostly follows the books of Holzapfel [17] and Bonet [16].

1.1 Motion

Figure 1.1 shows a body Ω in the undeformed *reference configuration* and in the *deformed configuration.* A material point in the deformed configuration $\boldsymbol{x}$ can be expressed in terms of the reference configuration $\boldsymbol{X}$ via the *motion* function $\boldsymbol{\chi}$

$$\boldsymbol{x} = \boldsymbol{\chi}(\boldsymbol{X})\,. \tag{1.1}$$

In both configurations the point is expressed in Cartesian coordinates $\boldsymbol{X} = X_i\boldsymbol{E}_i$, and $\boldsymbol{x} = x_i\boldsymbol{e}_i$ respectively. The displacement vector $\boldsymbol{u}$ of the material point between the deformed and reference configuration reads as follows

$$\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{X} = \boldsymbol{\chi}(\boldsymbol{X}) - \boldsymbol{X}\,. \tag{1.2}$$

From this, the *displacement gradient tensor* $\boldsymbol{H}$ can be derived

$$\boldsymbol{H} = \boldsymbol{\nabla}_{\boldsymbol{X}}\,\boldsymbol{u} = \frac{\partial u_i}{\partial X_j}\boldsymbol{e}_i \otimes \boldsymbol{e}_j\,, \tag{1.3}$$

where $\boldsymbol{\nabla}_{\boldsymbol{X}}$ denotes the partial gradient operator with respect to the reference configuration.

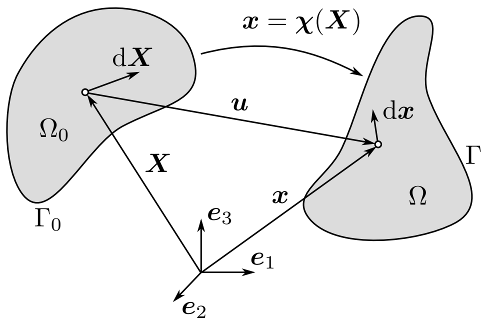


Figure 1.1: Reference and deformed configuration

1.2 Deformation

The deformations of a body are expressed with the *deformation gradient tensor* $\boldsymbol{F}$

$$\boldsymbol{F} = \boldsymbol{\nabla}_{\boldsymbol{X}} \boldsymbol{x} = \frac{\partial x_i}{\partial X_j} \boldsymbol{e}_i \otimes \boldsymbol{e}_j = \frac{\partial}{\partial \boldsymbol{X}} (\boldsymbol{X} + \boldsymbol{u}) = \boldsymbol{I} + \boldsymbol{H} \,, \tag{1.4}$$

where $\boldsymbol{I}$ is the identity matrix. $\boldsymbol{F}$ maps a differential vector d$\boldsymbol{X}$ from the reference to the deformed configuration d$\boldsymbol{x}$ (see Figure 1.1)

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{F} \mathrm{d}\boldsymbol{X} \,, \tag{1.5}$$

and vice versa

$$\mathrm{d}\boldsymbol{X} = \boldsymbol{F}^{-1} \mathrm{d}\boldsymbol{x} \,. \tag{1.6}$$

To be invertible, the determinant of the deformation gradient $\boldsymbol{F}$ must be non-singular. However, to obtain consistent results, the determinant is restricted to positive values. In a physical sense, this is the case, if the material is not vanishing, or self-intersecting

$$\det \boldsymbol{F} > 0 \,. \tag{1.7}$$

1.3 Strain

The deformation of a body consists of a translatory, a rotational and a stretching component. Translation and rotation correspond to the rigid body motions and induce no strains. Thus, they are not contributing to the internal strain energy. The deformation gradient $\boldsymbol{F}$ is already independent of translations, yet it still contains the rotational components and is consequently unsuited to measure strains. $\boldsymbol{F}$ can be decomposed into a stretching and a rotational part [16]

$$\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U} \,, \tag{1.8}$$

where $\boldsymbol{R}$ denotes the rotation tensor, and $\boldsymbol{U}$ is the stretching tensor. As shown in [16] the rotation tensor is orthogonal $\boldsymbol{R} = \boldsymbol{R}^T$. Thus, by squaring the deformation gradient, it is possible to become independent of the rotational part

$$\boldsymbol{C} = \boldsymbol{F}^T \boldsymbol{F} = \boldsymbol{U}^T \boldsymbol{U} \,, \tag{1.9}$$

where $\boldsymbol{C}$ is the *right Cauchy-Green deformation tensor*. In the case of no deformations $\boldsymbol{C}$ becomes the identity matrix $\boldsymbol{I}$. This is not desired, as intuitively no strains should yield a zero matrix. For this reason, common strain measures are formulated in terms of normalized relative deformations.

Consider for this, the squared lengths of a small line element in the reference $L = |\mathrm{d}\boldsymbol{X}|$ and in the deformed configuration $l = |\mathrm{d}\boldsymbol{x}|$, as depicted in Figure 1.1.

$$L^2 = \mathrm{d}\boldsymbol{X} \cdot \mathrm{d}\boldsymbol{X} \,. \tag{1.10}$$

The squared length of the deformed configuration can be expressed in terms of the deformation gradient $\boldsymbol{F}$

$$l^2 = \mathrm{d}\boldsymbol{x} \cdot \mathrm{d}\boldsymbol{x} = \mathrm{d}\boldsymbol{X} \cdot \boldsymbol{F}^T \boldsymbol{F} \mathrm{d}\boldsymbol{X} = \mathrm{d}\boldsymbol{X} \cdot \boldsymbol{C} \mathrm{d}\boldsymbol{X} \,, \tag{1.11}$$

Based on the difference of the squared lengths – and normalized with respect to the reference configuration – the *Green-Lagrange strain tensor* $\boldsymbol{E}$ is formulated as

$$E = \frac{1}{2}\left(\frac{l^2 - L^2}{L^2}\right) \,, \quad \boldsymbol{E} = \frac{1}{2}\left(\frac{\mathrm{d}\boldsymbol{X} \cdot \boldsymbol{C}\mathrm{d}\boldsymbol{X} - \mathrm{d}\boldsymbol{X} \cdot \mathrm{d}\boldsymbol{X}}{\mathrm{d}\boldsymbol{X} \cdot \mathrm{d}\boldsymbol{X}}\right) = \frac{1}{2}\left(\boldsymbol{C} - \boldsymbol{I}\right) \,. \tag{1.12}$$

Alternatively, the strain can also be normalized with respect to the deformed configuration, which yields the *Euler-Almansi strain tensor*

$$e = \frac{1}{2}\left(\frac{l^2 - L^2}{l^2}\right) \,, \quad \boldsymbol{e} = \frac{1}{2}\left(\boldsymbol{I} - \boldsymbol{F}^{-T}\boldsymbol{F}^{-1}\right) \,. \tag{1.13}$$

In index notation the strain tensors read[1]

$$E_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} - \frac{\partial u_k}{\partial X_i}\frac{\partial u_k}{\partial X_j}\right) \,, \tag{1.14}$$

$$e_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{\partial u_k}{\partial x_i}\frac{\partial u_k}{\partial x_j}\right) \,. \tag{1.15}$$

For small deformations the higher-order terms become negligible, which allows the linearization of the strain tensors. Moreover, the derivatives of the displacement with respect to the deformed and the reference configuration are approximately the same $\boldsymbol{u}_{,\boldsymbol{x}} \approx \boldsymbol{u}_{,\boldsymbol{X}}$. Hence, from both – the Green-Lagrange and the Euler-Almansi strain tensor – approximately the same linearized, infinitesimal strain tensor can be derived. This tensor corresponds to the *kinematic equation* of linear elasticity and reads as follows

$$\boldsymbol{E} \approx \boldsymbol{e} \approx \boldsymbol{\varepsilon} = \frac{1}{2}\left[\boldsymbol{\nabla u} + (\boldsymbol{\nabla u})^T\right] \,. \tag{1.16}$$

In index notation the infinitesimal strain reads

$$E_{ij} \approx e_{ij} \approx \varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i}\right) \,, \tag{1.17}$$

with ε_{ij} being the components of the *small strain tensor* (or *Cauchy strain tensor*) $\boldsymbol{\varepsilon}$

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{yx} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{zy} & \varepsilon_{zz} \end{bmatrix} \,. \tag{1.18}$$

[1]It should be noted that apart from the Green-Lagrange and the Euler-Almansi strain, several other strain measures exist, which primarily belong to the Seth-Hill family, such as the logarithmic, or Hencky strain [17].

Since the small strain tensor is symmetric, it can be expressed in Voigt (Nye) notation

$$\boldsymbol{\varepsilon}^V = \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ 2\varepsilon_{xy} \\ 2\varepsilon_{xz} \\ 2\varepsilon_{yz} \end{bmatrix} = \boldsymbol{L}\boldsymbol{u} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 \\ \frac{\partial}{\partial x} & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \tag{1.19}$$

with $\boldsymbol{L}$ being the kinematic differential operator, which expresses the strains in terms of the displacements.

1.4 Stress

Similar to the various strain measures, also for the stresses, different measures exist. Most prominent are the 2^{nd} *Piola-Kirchhoff stress tensor* – that is used for nonlinear material behavior and which is formulated in the reference configuration – and the *Cauchy* or *true stress tensor* [16, 17]. The Cauchy stress is typically used for linear-elasticity and will be considered in the following. Similar to the small strain tensor, also the Cauchy stress tensor is symmetric and defined in global directions. Thereby, Cauchy stress tensor measures the stresses acting on an infinitesimal area in the deformed configuration. For computational purposes typically the matrix notation is used, where $\boldsymbol{\sigma}$ denotes the Cauchy stress matrix

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz} \end{bmatrix}. \tag{1.20}$$

At any surface the equivalent traction vector $\boldsymbol{t} = [t_x, t_y, t_z]^T$ can be evaluated as

$$\boldsymbol{t} = \boldsymbol{\sigma}\boldsymbol{n}, \tag{1.21}$$

with $\boldsymbol{n} = [n_x, n_y, n_z]^T$ being the unit normal vector on the surface. In Voigt notation the true stress tensor reads

$$\boldsymbol{\sigma}^V = [\sigma_{xx},\, \sigma_{yy},\, \sigma_{zz},\, \tau_{xy},\, \tau_{xz},\, \tau_{yz}]^T. \tag{1.22}$$

1.5 Constitutive equation

The constitutive equation sets the stresses in relation to the strains. Magnitude and behavior of the resulting stresses depend on the considered material. For small deformations most engineering materials behave linear elastic, i.e., the relation between stress and strain is constant. Also, after removing the loading, the material returns into its initial state. *Hooke's law* describes such a linear elastic behavior for small deformations

$$\boldsymbol{\sigma} = \mathcal{C} : \boldsymbol{\varepsilon}, \tag{1.23}$$

where $\mathcal{C}$ is the fourth-order elasticity tensor and $\mathbf{A} : \mathbf{B} = A_{ij}B_{ij}$ denotes the dot product of two second-order tensors. The components of $\mathcal{C}$ are defined as

$$C_{ijkl} = \lambda\delta_{ij}\delta_{kl} + \mu(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}), \tag{1.24}$$

with λ and μ being the *Lamé constants* and δ_{ij} denoting the *Kronecker delta.* Expressed by the *Young's modulus* E and the *Poisson's ratio* ν, that will be used in the following, the Lamé constants read

$$\lambda = \frac{\nu}{1-2\nu} \cdot \frac{1}{1+\nu} \cdot E \tag{1.25}$$

$$\mu = \frac{1}{2} \cdot \frac{1}{1+\nu} \cdot E \tag{1.26}$$

As the strain and stress tensors are both symmetric, Hooke's elasticity tensor can be simplified into a 6×6 matrix $\boldsymbol{C}$, relating strains $\boldsymbol{\varepsilon}^V$ and stresses $\boldsymbol{\sigma}^V$ in Voigt notation

$$\boldsymbol{\sigma}^V = \boldsymbol{C}\boldsymbol{\varepsilon}^V \, , \tag{1.27}$$

with

$$\boldsymbol{C} = \begin{bmatrix} C_{1111} & C_{1122} & C_{1133} & C_{1123} & C_{1113} & C_{1112} \\ C_{2211} & C_{2222} & C_{2233} & C_{2223} & C_{2213} & C_{2212} \\ C_{3311} & C_{3322} & C_{3333} & C_{3323} & C_{3313} & C_{3312} \\ C_{2311} & C_{2322} & C_{2333} & C_{2323} & C_{2313} & C_{2312} \\ C_{1311} & C_{1322} & C_{1333} & C_{1323} & C_{1313} & C_{1312} \\ C_{1211} & C_{1222} & C_{1233} & C_{1223} & C_{1213} & C_{1212} \end{bmatrix} . \tag{1.28}$$

Exemplarily depicted for an isotropic material the elasticity matrix reads

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{xy} \\ \tau_{xz} \\ \tau_{yz} \end{bmatrix} = \begin{bmatrix} 2\mu+\lambda & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & 2\mu+\lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & 2\mu+\lambda & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix} \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ 2\varepsilon_{xy} \\ 2\varepsilon_{xz} \\ 2\varepsilon_{yz} \end{bmatrix} . \tag{1.29}$$

1.6 Equilibrium

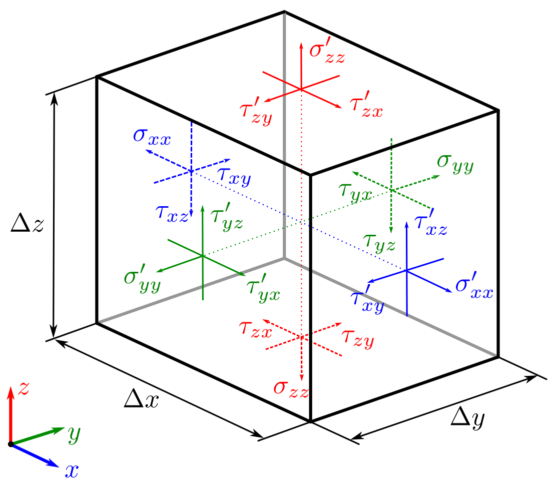
The set of partial differential equations describing three-dimensional linear elasticity can be derived with the equilibrium of forces on an infinitesimal volume inside a body Ω, as illustrated in Figure 1.2. With the sum of forces in each direction the partial differential equations can be stated as

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} = -b_x \tag{1.30}$$

$$\frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \tau_{zy}}{\partial z} = -b_y \tag{1.31}$$

$$\frac{\partial \sigma_{zz}}{\partial z} + \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} = -b_z \, , \tag{1.32}$$

where b_i denotes the body force acting on the volume $\Delta V = \Delta x \Delta y \Delta z$.



$$\sigma'_{xx} = \sigma_{xx} + \tfrac{\partial \sigma_{xx}}{\partial x}\Delta x$$
$$\tau'_{xy} = \tau_{xy} + \tfrac{\partial \tau_{xy}}{\partial x}\Delta x$$
$$\tau'_{xz} = \tau_{xz} + \tfrac{\partial \tau_{xz}}{\partial x}\Delta x$$
$$\sigma'_{yy} = \sigma_{yy} + \tfrac{\partial \sigma_{yy}}{\partial y}\Delta y$$
$$\tau'_{yx} = \tau_{yx} + \tfrac{\partial \tau_{yx}}{\partial y}\Delta y$$
$$\tau'_{yz} = \tau_{yz} + \tfrac{\partial \tau_{yz}}{\partial y}\Delta y$$
$$\sigma'_{zz} = \sigma_{zz} + \tfrac{\partial \sigma_{zz}}{\partial z}\Delta z$$
$$\tau'_{zx} = \tau_{zx} + \tfrac{\partial \tau_{zx}}{\partial z}\Delta z$$
$$\tau'_{zy} = \tau_{zy} + \tfrac{\partial \tau_{zy}}{\partial z}\Delta z$$

Figure 1.2: Stresses on an infinitesimal volume

1.7 Boundary value problem

To set up the boundary value problem, consider a domain Ω and the respective boundary Γ, as depicted in Figure 1.3. The boundary is subdivided into two distinct parts: the Dirichlet Γ_D and the Neumann Γ_N part, whereby $\Gamma = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. With the PDEs from the equilibrium of forces, the boundary value problem can be formulated in its strong form [17]

$$\operatorname{div}\boldsymbol{\sigma} + \boldsymbol{b} = \boldsymbol{0} \qquad \forall\, \boldsymbol{x} \in \Omega\,, \tag{1.33}$$
$$\boldsymbol{u} = \overline{\boldsymbol{u}} \qquad \forall\, \boldsymbol{x} \in \Gamma_D\,, \tag{1.34}$$
$$\boldsymbol{t} = \boldsymbol{n}\cdot\boldsymbol{\sigma} = \overline{\boldsymbol{t}} \qquad \forall\, \boldsymbol{x} \in \Gamma_N\,, \tag{1.35}$$

where $\overline{\boldsymbol{u}}$ and $\overline{\boldsymbol{t}}$ are the predefined displacements and tractions on the Dirichlet Γ_D, and Neumann boundary Γ_N, respectively.

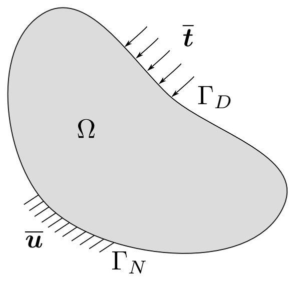


Figure 1.3: Boundary value problem

Using the kinematic (1.16) and the constitutive (1.23) equation, it is now possible to express the boundary value problem with the displacement field $\boldsymbol{u}$ as sole unknown.

$$\operatorname{div}\boldsymbol{\sigma} + \boldsymbol{b} = \operatorname{div}\left(\mathcal{C} : \frac{1}{2}\left[\boldsymbol{\nabla}\boldsymbol{u} + (\boldsymbol{\nabla}\boldsymbol{u})^T\right]\right) + \boldsymbol{b} = \boldsymbol{0}\,. \tag{1.36}$$

Chapter 2

Finite Element Method

In the context of structural analysis, the aim is now to find the correct solution for the displacement field $\boldsymbol{u}(\boldsymbol{X})$ of the boundary value problem, introduced in Section 1.7. From this, strains, and stresses can subsequently be derived using the kinematic and constitutive equations. For simple problems – e.g., a cuboid under dead load – the boundary value problem can be solved analytically by integration. The boundary conditions then define the unknown integration constants. However, in general, it is impossible to find an analytical solution for an arbitrarily complex domain with any boundary conditions. Hence, an approximate solution for the displacement field is sought. For this, various approximation techniques are available, such as *finite differences*, the *finite volume method*, and most prominently the *finite element method* (FEM), which is considered in the following.

The FEM is based on the weak form of the differential equations and discretizes the domain into a finite number of elements, i.e., a boundary conforming mesh. On each element, the solution is approximated with typically linear or quadratic shape functions. FEM most commonly uses the isoparametric concept, which uses the same shape functions to represent the solution and the elements' geometric shape. Thus, especially for low-order FEM, the accuracy highly depends on the size of the elements for two reasons: Firstly, the mesh needs to be sufficiently dense to approximate the potentially complex domain decently, and secondly, the shape functions must be able to represent the solution.

2.1 Weak form

For the finite element method, the PDEs are not solved in their *strong form* but are converted into their equivalent *weak form*. In the weak form, the solution must not fulfill the PDEs over the entire domain, but in an 'average' integral sense. Additionally, the weak form poses reduced continuity requirements on the solution.

In solid mechanics, the weak form is related to the *principle of virtual work* [16, 17], which states that for a body that is in equilibrium, the virtual work of the internal forces (or virtual work of the stresses) is identical to the virtual work of the external forces.

$$\delta W_{int} = \delta W_{ext} \, . \tag{2.1}$$

To derive the internal and external virtual work, the boundary value problem is multiplied with an arbitrary test function $\boldsymbol{v}$ and integrated over the domain

$$\int_\Omega (\mathrm{div}\boldsymbol{\sigma} + \boldsymbol{b}) \cdot \boldsymbol{v} \, \mathrm{d}\Omega = \int_\Omega \mathrm{div}\boldsymbol{\sigma} \cdot \boldsymbol{v} \, \mathrm{d}\Omega + \int_\Omega \boldsymbol{b} \cdot \boldsymbol{v} \, \mathrm{d}\Omega = 0 \, . \tag{2.2}$$

Applying the product rule

$$\operatorname{div}\boldsymbol{\sigma} \cdot \boldsymbol{v} = \operatorname{div}(\boldsymbol{\sigma v}) - \boldsymbol{\sigma} : \boldsymbol{\nabla v} \,, \tag{2.3}$$

and Gauss's theorem

$$\int_\Omega \operatorname{div}(\boldsymbol{\sigma v}) \mathrm{d}\Omega = \oint_\Gamma \boldsymbol{\sigma v} \cdot \boldsymbol{n} \, \mathrm{d}\Gamma \,, \tag{2.4}$$

yields

$$-\int_\Omega \boldsymbol{\sigma} : \boldsymbol{\nabla v} \, \mathrm{d}\Omega + \int_\Omega \boldsymbol{b} \cdot \boldsymbol{v} \, \mathrm{d}\Omega + \oint_\Gamma \boldsymbol{\sigma v} \cdot \boldsymbol{n} \, \mathrm{d}\Gamma = 0 \,. \tag{2.5}$$

Since $\boldsymbol{\sigma}$ is symmetric, the following relation holds

$$\boldsymbol{\sigma} : \boldsymbol{\nabla v} = \boldsymbol{\sigma} : \delta\boldsymbol{\varepsilon} = \boldsymbol{\sigma} : \frac{1}{2}\left[\boldsymbol{\nabla v} + (\boldsymbol{\nabla v})^T\right] \,. \tag{2.6}$$

The internal and external virtual work – or bilinear and linear form – read as follows [16, 17]

$$\delta W_{int} = \mathcal{B}(\boldsymbol{u}, \boldsymbol{v}) = \int_\Omega \boldsymbol{\sigma} : \delta\boldsymbol{\varepsilon} \, \mathrm{d}\Omega \,, \tag{2.7}$$

$$\delta W_{ext} = \mathcal{F}(\boldsymbol{v}) = \int_\Omega \boldsymbol{b} \cdot \boldsymbol{v} \, \mathrm{d}\Omega + \int_{\Gamma_N} \bar{\boldsymbol{t}} \cdot \boldsymbol{v} \, \mathrm{d}\Gamma_N \,. \tag{2.8}$$

In Voigt notation the equilibrium reads

$$\int_\Omega \delta\boldsymbol{\varepsilon}^{VT} \boldsymbol{C} \boldsymbol{\varepsilon}^V \, \mathrm{d}\Omega = \int_\Omega \boldsymbol{v}^T \boldsymbol{b} \, \mathrm{d}\Omega + \int_{\Gamma_N} \boldsymbol{v}^T \bar{\boldsymbol{t}} \, \mathrm{d}\Gamma_N \,. \tag{2.9}$$

Thereby, the solution $\boldsymbol{u}$ belongs to the function space $\boldsymbol{u} \in \mathcal{U}(\Omega)$ which is defined in such a way that $\boldsymbol{u}$ satisfies the Dirichlet boundary conditions

$$\mathcal{U}(\Omega) = \left\{\boldsymbol{u} \;\middle|\; \boldsymbol{u} \in H^1(\Omega), \boldsymbol{u} = \overline{\boldsymbol{u}} \;\; \forall\, \boldsymbol{X} \in \Gamma_\mathrm{D}\right\} \,, \tag{2.10}$$

and the test function $\boldsymbol{v}$ belongs to the function space $\boldsymbol{v} \in \mathcal{V}(\Omega)$ that contains all admissible test functions which satisfy homogeneous Dirichlet boundary conditions

$$\mathcal{V}(\Omega) = \left\{\boldsymbol{v} \;\middle|\; \boldsymbol{v} \in H^1(\Omega), \boldsymbol{v} = 0 \;\; \forall\, \boldsymbol{X} \in \Gamma_\mathrm{D}\right\} , \tag{2.11}$$

with H^1 being the first order Sobolev space [21]. In the following, the *Bubnov–Galerkin* method is considered, in which the same shape functions are used for both the approximation of the solution and the test functions.

2.2 Discretization

For solving the weak form, it is necessary to carry out integrations over the domain and the boundary. As already explained, an integration over a complex domain is, in general, not analytically feasible. Thus, the domain is subdivided into simple, integrable parts – called elements. This spatial discretization is commonly known as meshing. The resulting elements are usually either triangles and quadrilaterals in two dimensions or tetrahedrons and hexahedrons in three dimensions. For the sake of simplicity, only quadrilaterals and hexahedrons will be considered in the following. Besides the spatial discretization, also the continuous unknown displacement field $\boldsymbol{u}(\boldsymbol{X}) = [u(\boldsymbol{X}), v(\boldsymbol{X}), w(\boldsymbol{X})]^T$ needs to be discretized. Therefore, $\boldsymbol{u}$ is approximated by a linear combination of linearly independent shape functions $\boldsymbol{N}$

$$\boldsymbol{u}(\boldsymbol{X}) \approx \boldsymbol{u}^h(\boldsymbol{X}) = \boldsymbol{N}(\boldsymbol{X})\tilde{\boldsymbol{u}} = \begin{bmatrix} N_1 & 0 & 0 & ... & N_n & 0 & 0 \\ 0 & N_1 & 0 & ... & 0 & N_n & 0 \\ 0 & 0 & N_1 & ... & 0 & 0 & N_n \end{bmatrix} \begin{bmatrix} \tilde{u}_1 \\ \tilde{v}_1 \\ \tilde{w}_1 \\ \vdots \\ \tilde{u}_n \\ \tilde{v}_n \\ \tilde{w}_n \end{bmatrix} \tag{2.12}$$

where N_i is the trivariate shape function that corresponds to the unknown, scalar *degree of freedom* $\tilde{u}_i$. For classical FEM, the shape functions are defined over each element.

To be independent of the location of the respective element, shape functions are typically not formulated in global coordinates $\boldsymbol{X}$, but in parametric coordinates $\boldsymbol{\xi}^T = [\xi, \eta, \zeta]$. The most common one-dimensional shape functions are defined on the interval $\xi \in [-1, 1]$

$$N_1^{1D}(\xi) = \frac{1}{2}(1 - \xi)\,, \tag{2.13}$$

$$N_2^{1D}(\xi) = \frac{1}{2}(1 + \xi)\,. \tag{2.14}$$

Multi-dimensional shape functions for quadrilaterals and hexahedrons, are then created as the tensor product of the one dimensional shape functions $\boldsymbol{\xi} \in [-1, 1]^n$, e.g., for the three dimensional case:

$$N_{i,j,k}(\xi, \eta, \zeta) = N_i^{1D}(\xi) \cdot N_j^{1D}(\eta) \cdot N_k^{1D}(\zeta)\,. \tag{2.15}$$

For the transfer, from the parametric to the physical space, a mapping is required. Following the isoparametric concept, this mapping is usually carried out with the same shape functions

$$\boldsymbol{X}(\boldsymbol{\xi}) = \boldsymbol{N}(\boldsymbol{\xi})\tilde{\boldsymbol{X}}\,, \tag{2.16}$$

with $\tilde{\boldsymbol{X}}$ being the vector of scalar values determining the shape of the element. For classical, linear finite elements, this corresponds to the coordinates of the corner points $\tilde{\boldsymbol{X}}^T = [\tilde{x}_1, \tilde{y}_1, \tilde{z}_1, ..., \tilde{x}_n, \tilde{y}_n, \tilde{z}_n]^T$. To solve the weak form, also the derivatives of the shape functions with respect to $\boldsymbol{X}$ are required. These can be expressed with the derivatives of the shape functions with respect to $\boldsymbol{\xi}$, using the chain rule

$$\nabla_{\boldsymbol{X}} \boldsymbol{N}(\boldsymbol{\xi}) = \boldsymbol{J}_{\boldsymbol{\xi}}^{-T} \nabla_{\boldsymbol{\xi}} \boldsymbol{N}(\boldsymbol{\xi})\,, \tag{2.17}$$

where $\boldsymbol{J}$ is Jacobian tensor of the mapping

$$\boldsymbol{J}_{\boldsymbol{\xi}} = \frac{\partial X_i(\boldsymbol{\xi})}{\partial \xi_j} \boldsymbol{e}_i \otimes \boldsymbol{e}_j\,. \tag{2.18}$$

$\nabla_{\boldsymbol{X}}$ and $\nabla_{\boldsymbol{\xi}}$ are the partial gradient operators with respect to the Cartesian, and parametric coordinates, respectively.

Following the *Bubnov-Galerkin* approach [22], the same shape functions are chosen for displacement field $\boldsymbol{u}$ and the test functions $\boldsymbol{v}$

$$\boldsymbol{v}(\boldsymbol{X}) \approx \boldsymbol{v}^h(\boldsymbol{X}) = \boldsymbol{N}(\boldsymbol{X})\tilde{\boldsymbol{v}}\,. \tag{2.19}$$

2.3 Discrete weak form

Applying the discretizations for the (virtual) displacements $\boldsymbol{u}^h$ and $\boldsymbol{v}^h$ and exploiting the symmetries of the strain and stress tensor, the discretized, virtual internal and external work can be derived [23]

$$\delta W^h_{int} = \tilde{\boldsymbol{v}}^T \int_\Omega \boldsymbol{B}^T \boldsymbol{C} \boldsymbol{B} \, \mathrm{d}\Omega \, \tilde{\boldsymbol{u}}, \tag{2.20}$$

$$\delta W^h_{ext} = \tilde{\boldsymbol{v}}^T \int_\Omega \boldsymbol{N}^T \boldsymbol{b} \, \mathrm{d}\Omega + \tilde{\boldsymbol{v}}^T \int_{\Gamma_N} \boldsymbol{N}^T \, \bar{\boldsymbol{t}} \, \mathrm{d}\Gamma_N \, , \tag{2.21}$$

where $\boldsymbol{B}$ is the linear strain operator to compute the discretized strain vector (in Voigt notation)

$$\boldsymbol{\varepsilon}^{h,V} = \boldsymbol{L}\boldsymbol{N}\tilde{\boldsymbol{u}} = \boldsymbol{B}\tilde{\boldsymbol{u}} = \begin{bmatrix} \frac{\partial N_1}{\partial x} & 0 & 0 & ... & \frac{\partial N_n}{\partial x} & 0 & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & ... & 0 & \frac{\partial N_n}{\partial y} & 0 \\ 0 & 0 & \frac{\partial N_1}{\partial z} & ... & 0 & 0 & \frac{\partial N_n}{\partial z} \\ \frac{\partial N_1}{\partial x} & \frac{\partial N_1}{\partial y} & 0 & ... & \frac{\partial N_n}{\partial x} & \frac{\partial N_n}{\partial y} & 0 \\ 0 & \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial z} & ... & 0 & \frac{\partial N_n}{\partial y} & \frac{\partial N_n}{\partial z} \\ \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_1}{\partial z} & ... & \frac{\partial N_n}{\partial x} & 0 & \frac{\partial N_n}{\partial z} \end{bmatrix} \begin{bmatrix} \tilde{u}_1 \\ \tilde{v}_1 \\ \tilde{w}_1 \\ \vdots \\ \tilde{u}_n \\ \tilde{v}_n \\ \tilde{w}_n \end{bmatrix} = \begin{bmatrix} \varepsilon^h_{xx} \\ \varepsilon^h_{yy} \\ \varepsilon^h_{zz} \\ 2\varepsilon^h_{xy} \\ 2\varepsilon^h_{xz} \\ 2\varepsilon^h_{yz} \end{bmatrix} . \tag{2.22}$$

From the discretized virtual work of Equations (2.20) and (2.21), the well-known linear system of equations can be derived

$$\boldsymbol{K}\tilde{\boldsymbol{u}} = \boldsymbol{f} \, , \tag{2.23}$$

where $\boldsymbol{K}$ denotes the global stiffness matrix, and $\boldsymbol{f}$ the global load vector that is composed of body and external loads $\boldsymbol{f} = \boldsymbol{f}_b + \boldsymbol{f}_N$. The stiffness matrix and the load vector are evaluated element-wise and subsequently assembled into the global stiffness matrix and global load vector, respectively

$$\boldsymbol{K} = \overset{n_{el}}{\underset{e}{\mathsf{A}}} \boldsymbol{K}^e \, , \tag{2.24}$$

$$\boldsymbol{f} = \overset{n_{el}}{\underset{e}{\mathsf{A}}} \boldsymbol{f}^e \, , \tag{2.25}$$

where A is the assembly operator [24] that assembles the entries of the element stiffness matrices $\boldsymbol{K}^e$ and load vectors $\boldsymbol{f}^e$ to the global stiffness matrix $\boldsymbol{K}$ and global load vector $\boldsymbol{F}$ according to the relation between local and global degrees of freedom. Briefly: Element entries that correspond to the same global degree of freedom are summed up.

Since the shape functions $\boldsymbol{N}$ and the linear strain operator $\boldsymbol{B}$ are formulated, in general, with respect to parametric coordinates, the integration is performed over the parametric space of the element, typically with $\boldsymbol{\xi} \in [-1, 1]^3 \subset \mathbb{R}^3$. The boundary integration requires an additional mapping from the parametric space of the boundary $\boldsymbol{s}$ to the physical space $\boldsymbol{X}$, and subsequently to the parametric space of the element

$\boldsymbol{\xi}$: $\boldsymbol{s} \subset \mathbb{R}^2 \mapsto \boldsymbol{X} \mapsto \boldsymbol{\xi}$

$$\boldsymbol{K}^e = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \boldsymbol{B}^T(\boldsymbol{\xi})\boldsymbol{C}\boldsymbol{B}(\boldsymbol{\xi}) \,\det\boldsymbol{J}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \,\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta \tag{2.26}$$

$$\boldsymbol{f}_b^e = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \boldsymbol{N}^T(\boldsymbol{\xi})\boldsymbol{b}(\boldsymbol{X}(\boldsymbol{\xi})) \,\det\boldsymbol{J}_{\boldsymbol{\xi}}(\boldsymbol{\xi}) \,\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta \tag{2.27}$$

$$\boldsymbol{f}_N^e = \int_{-1}^{1}\int_{-1}^{1} \boldsymbol{N}^T(\boldsymbol{\xi}(\boldsymbol{X}(\boldsymbol{s})))\bar{\boldsymbol{t}}(\boldsymbol{X}(\boldsymbol{s})) \,\det\boldsymbol{J}_{\boldsymbol{s}}(\boldsymbol{s}) \,\mathrm{d}s\mathrm{d}t\,, \tag{2.28}$$

with $\boldsymbol{J}$ being the Jacobian matrix of the respective mappings, following Equation (2.18).

2.4 Numerical integration

For the integration of the stiffness matrix and the load vector, a numerical integration scheme is applied – typically Gauss quadrature or a variation

$$\int_{-1}^{1} f(t)\mathrm{d}t \approx \sum_{i=1}^{n} w_i \cdot f(t_i)\,, \tag{2.29}$$

where t_i and w_i are the locations and weights of the n Gauss points. For polynomials with a degree of $p \leq 2n-1$, the Gauss quadrature yields the exact result. Since, the Gauss quadrature is defined for $t \in [-1,1]$, a mapping is required, if the integration domain differs from the Gauss space

$$\int_{a}^{b} f(x)\mathrm{d}x \approx \sum_{i=1}^{n} w_i \cdot f(x(t_i)) \cdot \det\boldsymbol{J}_t(t_i)\,. \tag{2.30}$$

Chapter 3

High-order finite elements

One of the core aspects of the finite element method is the approximation of the typically unknown solution with an element-wise linear combination of shape functions. The standard $h-$version of the FEM, which employs low-order, or linear shape functions, yields convergence by refining the mesh. Consequently, a fine discretization is required in regions where the solution becomes complex, or even singular, e.g., at corners, support points, or concentrated loads. In contrast to $h-$FEM, the $p-$FEM achieves convergence by locally or globally increasing the shape functions' polynomial degrees. In many cases, it has better convergence than the $h-$version [25, 26]. Additionally, high-order shape functions are robust against locking effects and element distortions. The most prominent high-order shape functions are based on polynomials since they are easy to evaluate and can represent different continuities of the solution between the elements[1]. Possible choices are, for example *Lagrange polynomials*, *integrated (hierarchic) Legendre polynomials*, or *splines.* From a computational point, it is desirable to have a sparse stiffness matrix with a small conditioning number. Both criteria are fulfilled well by the hierarchic Legendre polynomials due to their limited support and orthogonal basis functions, briefly described in the following.

3.1 Hierarchical shape functions

The basic idea of hierarchical shape functions is to continuously add higher-order functions to the low-order basis, instead of replacing the low-order basis functions with higher-order polynomials. In 1D, higher-order functions can easily be chosen to be orthogonal with respect to certain scalar products. Additionally, they should vanish at the boundaries. The integrated Legendre polynomials N_i provide such hierarchical shape functions in the parametric interval $\xi \in [-1, 1]$ [27], as depicted in Figure 3.1.

$$N_1(\xi) = \frac{1}{2}(1 - \xi)\,, \tag{3.1}$$

$$N_2(\xi) = \frac{1}{2}(1 + \xi)\,, \tag{3.2}$$

$$N_i(\xi) = \phi_{i-1}(\xi), \quad i = 3, ..., p+1\,, \tag{3.3}$$

with

$$\phi_j(\xi) = \sqrt{\frac{2j-1}{2}} \int_{-1}^{\xi} L_{j-1}(x)\,\mathrm{d}x = \frac{1}{\sqrt{4j-2}}(L_j(\xi) - L_{j-2}(\xi)), \quad j = 2, 3, \dots\,, \tag{3.4}$$

[1] *Remark:* In the case of Lagrange and Legendre shape functions, only C^0-continuity is enforced at the element boundaries since here only shape functions that are composed of at least one linear shape function – i.e., corner, edge, and face modes (see Figures 3.3 and 3.4) – are coupled by the respective global degree of freedoms.

where $L_i(\xi)$ is the i^{th} Legendre polynomial, which can be computed with *Bonnet's recursion formula*

$$L_i(\xi) = \frac{1}{i}\left[(2i\xi - \xi)L_{i-1}(\xi) - (i-1)L_{i-2}(\xi)\right], \quad i = 2, 3, \ldots, \tag{3.5}$$

The first two Legendre polynomials are given as

$$L_0(\xi) = 1\,,\ L_1(\xi) = \xi\,. \tag{3.6}$$

The degrees of freedom that correspond to the linear shape functions relate to the nodes' actual displacements. The other degrees of freedom belong to higher-order functions which are zero at the nodes and (in the non-trivial case) non-zero in between. Consequently, the *partition of unity* is no longer fulfilled $\sum_i N_i(\xi) \neq 1\ \forall \xi \in]-1, 1[$, which is, however, not problematic in the context of the FEM. In contrast to the Lagrange polynomials, the degrees of freedom related to the high-order shapes do not directly correspond to displacements at specific locations.

The derivatives of the shape functions are computed similarly to the shape functions

$$N_{1,\xi}(\xi) = -0.5\,, \tag{3.7}$$

$$N_{2,\xi}(\xi) = 0.5\,, \tag{3.8}$$

$$N_{i,\xi}(\xi) = \phi_{i-1,\xi}(\xi), \quad i = 3, \ldots, p+1 \tag{3.9}$$

with

$$\phi_{j,\xi}(\xi) = \frac{1}{\sqrt{4j-2}}(L_{j,\xi}(\xi) - L_{j-2,\xi}(\xi)), \quad j = 2, 3, \ldots, \tag{3.10}$$

based on the Lagrange polynomials and their derivatives

$$L_{i,\xi}(\xi) = \left[(2i-1)L_{i-1}(\xi) + L_{i-2,\xi}(\xi)\right], \quad i = 2, 3, \ldots, \tag{3.11}$$

where

$$L_{0,\xi}(\xi) = 0\,,\ L_{1,\xi}(\xi) = 1\,. \tag{3.12}$$

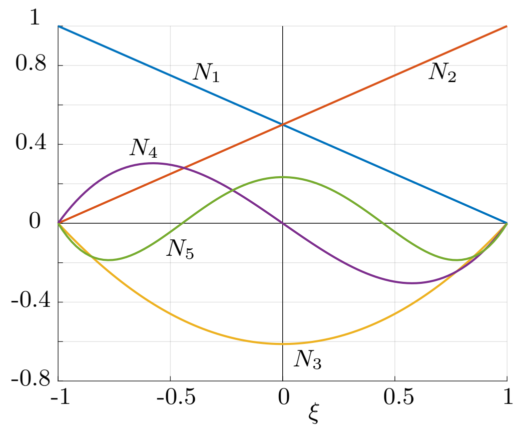


Figure 3.1: First five integrated Legendre polynomials for $p = 1, \ldots, 4$.

3.2 Shape functions for higher dimensions

Higher-dimensional shape functions are created – analogously to low-order – with the tensor product of the one-dimensional shape functions (see Equation (2.15)). The polynomial degree in each direction can be chosen independently. Depending on the complexity of the solution, it is sensible to limit the number of resulting shape functions in the function space by truncation. For this, all shape functions of a combined polynomial degree p_{res} higher than a certain limit are discarded (typically, $p_{res} = p + 1$ in 2D and $p_{res} = p + 2$ in 3D.) Depending on the shape of the elements (e.g. in the case of quadrilaterals) additional shape functions of degree $p_{res} + 1$ are added [27]. Consider for this *Pascal's 'triangle'* of the tensor product space for the two-dimensional shape functions depicted in Figure 3.2.

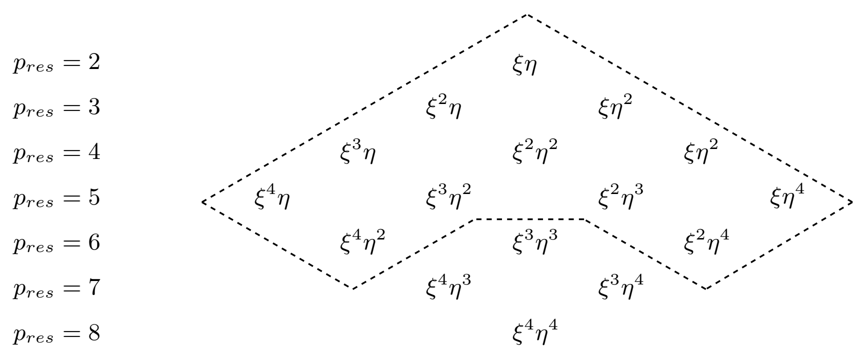
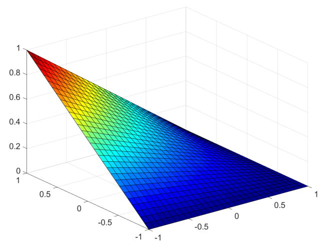


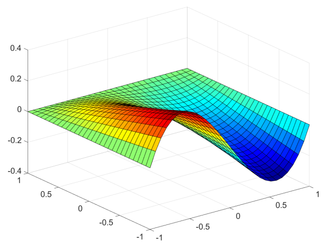
Figure 3.2: Full polynomial space for 2d quadrilateral shape functions up to a polynomial degree $p = q = 4$. The trunk space is indicated by the dashed line and contains all shape functions with $p_{res} = [2, 5]$ and supplementary shape functions of degree $p_{res} = 6$.

For the $n-$dimensional case, the number of constitutive linear shape functions allows the categorization of the resulting tensor product shape functions into

- corner modes, with exclusively linear shape functions, e.g. $N = 0.125(\xi - 1)(\eta - 1)(\zeta - 1)$,
- edge modes, with $n - 1$ linear shape functions, e.g. $N = 0.15(\xi - 1)(\eta - 1)(\zeta^2 - 1)$,
- face modes, with $n - 2$ linear shape functions, e.g. $N = 0.18(\xi - 1)(\eta^2 - 1)(\zeta^2 - 1)$, and
- volume modes, with $n - 3$ linear shape functions, e.g. $N = 0.216(\xi^2 - 1)(\eta^2 - 1)(\zeta^2 - 1)$.

Modes, which have no support on the boundaries (i.e., no constitutive linear shape functions) are also called bubble modes. In the two-dimensional case these are the face modes and in the three-dimensional case the volume modes, as illustrated in Figures 3.3 and 3.4.


(a) Corner mode


(b) Edge mode

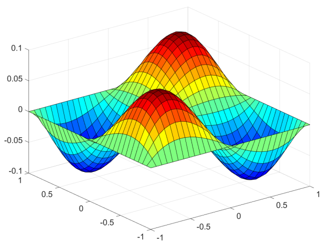
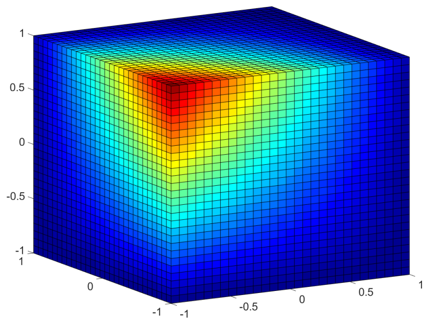
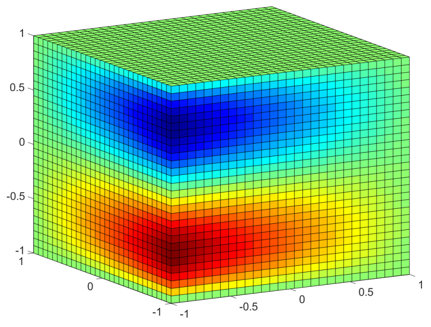
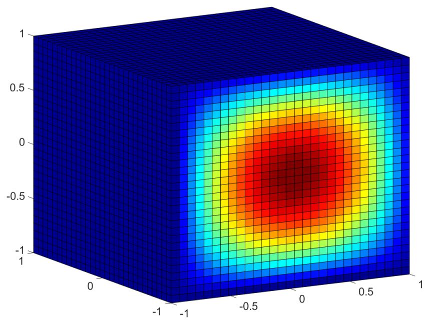

(c) Face, or bubble mode

Figure 3.3: Different modes of the two-dimensional shape functions.


(a) Corner mode


(b) Edge mode


(c) Face mode

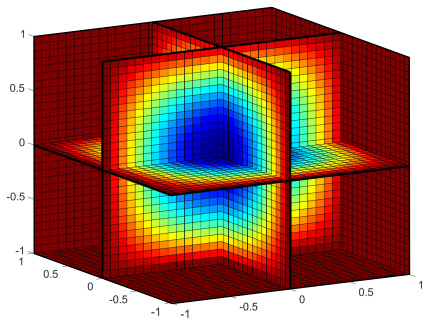

(d) Volume, or bubble mode

Figure 3.4: Different modes of the three-dimensional shape functions.

Chapter 4

Finite Cell Method

Nowadays, CAD models are the basis for almost all FEM simulations. As explained in the Introduction, the transition process between a CAD model and analysis-suitable representation is the major bottleneck in the product design process. For this reason, several 'direct' finite element methods were developed. Apart from the *isogometric analysis* (IGA) [11], *embedded*, or *fictitious domain methods* have been under intense research during the past decades. In the literature these methods are named differently, e.g., fictitious domain approach [28, 29, 30], implicit meshing [31], immersed FEM/boundary method [32, 33, 34], fixed/Cartesian grid FEM [35, 36, 37, 38], etc. Also the *finite cell method*(FCM) belongs into this category [39].

The basic idea of these methods is similar: The complex model is embedded into a fictitious domain, which has a simple shape and is therefore easy to mesh. The geometry's real shape is then – fully automatically and robustly – captured during the downstream numerical analysis. For the FCM, this happens during the integration of the system matrices. The FCM combines two concepts: the $p-$version of the finite element method, and the above mentioned fictitious domain approach. In the following, the necessary modifications of the FEM towards the FCM are explained.

4.1 Formulation of the finite cell method

The finite cell method embeds a physical domain Ω_{phy} – i.e., the geometrical model – into a fictitious domain Ω_{fict}, thus, forming an extended domain $\Omega_{\cup}$. The domain $\Omega_{\cup}$ is then subdivided into *finite cells* of simple shape, e.g., into rectangles in 2D or cuboids in 3D[1]. The finite cell mesh serves as spatial FEM discretization of $\Omega_{\cup}$. Figure 4.1 illustrates this concept for a two-dimensional problem. The adapted weak form of linear elasticity for the extended domain $\Omega_{\cup}$ reads as follows, according to the Equations (2.7) and (2.8)

$$\int_{\Omega_{\cup}} [\boldsymbol{L}\boldsymbol{v}]^T \alpha \boldsymbol{C} [\boldsymbol{L}\boldsymbol{u}] \, \mathrm{d}\Omega_{\cup} = \int_{\Omega_{\cup}} \alpha \boldsymbol{v}^T \boldsymbol{b} \, \mathrm{d}\Omega_{\cup} + \int_{\Gamma_N} \boldsymbol{v}^T \bar{\boldsymbol{t}} \, \mathrm{d}\Gamma_N \, , \tag{4.1}$$

where $\alpha(\boldsymbol{X})$ is an indicator function that weights the material matrix $\boldsymbol{C}$ and body forces $\boldsymbol{b}$

$$\alpha(\boldsymbol{X}) = \left\{ \begin{array}{ll} 1 & \forall \, \boldsymbol{X} \in \Omega_{phy} \\ 10^{-q} & \forall \, \boldsymbol{X} \in \Omega_{fict} \end{array} \right. . \tag{4.2}$$

[1] *Remark:* Generalizations to adaptive multi-level $hp-$refinements [40, 41, 42] and unstructured meshes [43, 44, 45] have been studied extensively.

In the limit, where $q = \infty$, Equation (4.1) conforms with the standard weak form of the finite element method. However, if $\alpha = 0$, a finite element-like discretization of the solution field $\boldsymbol{u}$ will lead to ill-conditioned or even singular system matrices. The reasons for this are (i) cells that are entirely outside and thus add no information to the global system matrices and (ii) shape functions on badly cut cells that appear to be nearly linear dependent [46]. The former can simply be avoided by deactivating outside cells. A remedy which resolves both cases is choosing a finite q, typically in the range of $q \in \{6, ..., 12\}$ [47, 39]. In any case it is sensible to deactivate outside cells; not only to improve the conditioning number, but also to reduce the size of the system matrices.

The conditioning number can be further reduced with a suitable preconditioning, an orthogonalization of the shape functions of badly cut cells, or the increase of the continuity between cut cells [46, 48]. Weighting the material in the fictitious domain Ω_{fict}, with $\alpha > 0$, can be considered as embedding the physical domain Ω_{phy} into a very soft material. Obviously, this introduces a modeling error. However, provided that q is chosen large enough, the FCM was proven to be convergent to a solution close to that of the original problem [49] .

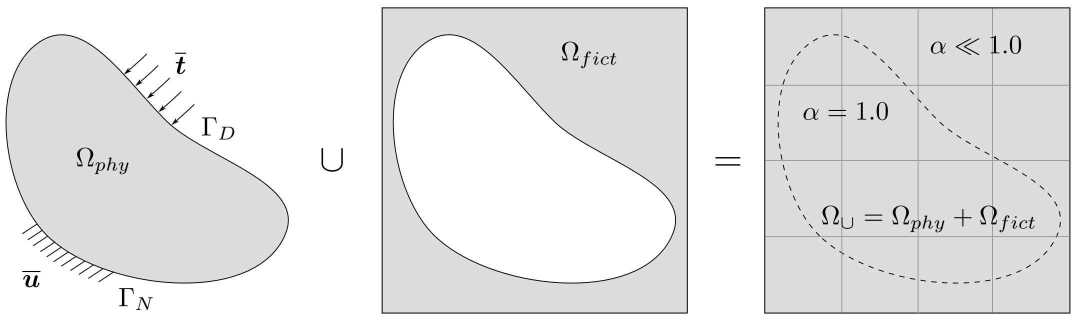


Figure 4.1: The concept of the finite cell method.

The finite cell method resolves the real shape of the physical domain with the indicator function α. Hence, handling of the complex geometry is shifted from a discretization – i.e., a boundary conforming meshing – to the integration of the system matrices. This imposes less geometrical requirements on the model, as only a robust *point membership classification* (PMC), or *point-inclusion test* is required. This means, for every (integration) point $\boldsymbol{X}$, it must be possible to decide whether it is inside or outside of Ω_{phy}, which implies that Ω_{phy} must have a mathematically valid description.

4.2 Quadrature rules for the FCM

The discontinuity of the indicator function α necessitates an adapted integration scheme for the integration of the element matrices and load vectors on the cut finite cells. The simplest and most robust – yet not necessarily the most efficient – choice is a composed integration.

4.2.1 Composed integration

The basic idea of composed integration is to approximate the boundary recursively with an adaptive space-tree T_{int}, e.g., a quadtree in two dimensions and an octree in three dimensions (see Figure 4.2). The leaves of T_{int} are called integration leaves c_{int}. The denser distribution of integration points around the boundaries allows an accurate integration over the domain, provided the space-tree is sufficiently refined.

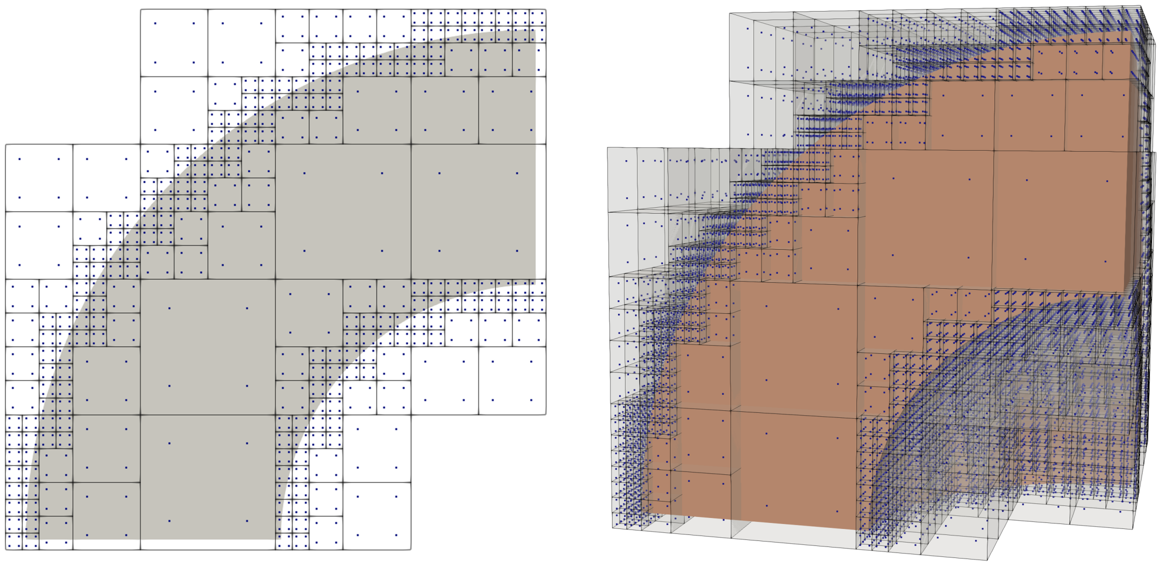

(a) (b)

Figure 4.2: Spacetree approximation with two integration points per dimensions and integration cell: (a) quadtree in two dimensions and (b) octree in three dimensions.

Using the Gauss quadrature rule of Equation (2.30) the numerical integration extends to

$$\int_a^b f(x)\mathrm{d}x \approx \sum_{j=1}^{n_{c_{int}}} \sum_{i=1}^{n_{GP}} w_i \cdot f(x(t_i)) \cdot \det \boldsymbol{J}_t(t_i) \cdot \det \boldsymbol{J}_{c_{int,j}}(t_i)\,. \tag{4.3}$$

Composed integration is not restricted to space-trees, but can also be applied to other domain decompositions. One possibility is to resolve the geometry on the finite cells by an integration mesh. This seems to be counter-intuitive, as the purpose of an embedded domain approach is to avoid meshing. However, the requirements on an integration mesh are much lower, compared to a FEM discretization – e.g., hanging nodes, or high aspect ratios are not problematic.

For the finite cell methods, an extension to an octree integration is implemented with the *smart octree* integration scheme [50, 51]. Here, the octree cells are fully-automatically subdivided at the intersections of finite cells with the domain boundary. Additionally, the actual shape of the boundary is recovered by higher-order blending functions. Consequently, the smart octree has far fewer integration cells, which moreover are almost entirely non-cut and can represent the geometry's exact shape.

4.2.2 Moment fitting

Another approach for the integration of the discontinuous indicator function is moment-fitting [52, 51]. The basic idea is to find – on each finite cell – the optimal location and weights of the integration points (see Equation (2.30)). To this end, the moment fitting equations need to be solved

$$\sum_{i=1}^{n_{GP}} N_j(\boldsymbol{x}_i) w_i = \int_{\Omega_{cell_{phy}}} N_j(\boldsymbol{x})\,\mathrm{d}\Omega\,, \tag{4.4}$$

where N_j are the m independent basis functions of the finite cell ansatz and Ω_{cell} is the domain of a cut cell. This leads to a linear system of equations

$$\begin{bmatrix} N_1(\boldsymbol{x}_i) & ... & N_1(\boldsymbol{x}_{n_{GP}}) \\ \vdots & \ddots & \vdots \\ N_m(\boldsymbol{x}_i) & ... & N_m(\boldsymbol{x}_{n_{GP}}) \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_{n_{GP}} \end{bmatrix} = \boldsymbol{A}\boldsymbol{w} = \boldsymbol{b} = \begin{bmatrix} \int_{\Omega_{cell_{phy}}} N_1(\boldsymbol{x})\,\mathrm{d}\Omega \\ \vdots \\ \int_{\Omega_{cell_{phy}}} N_m(\boldsymbol{x})\,\mathrm{d}\Omega \end{bmatrix}, \tag{4.5}$$

where the right hand-side vector $\boldsymbol{b}$ contains the individual volume integrals, also called moments. The integration of the moments is much cheaper than the evaluation of the stiffness matrices, and can be computed, for instance, with the standard octree, or smart octree. Using the divergence theorem, it is also possible to transform the volume integral into a boundary integral, which is even cheaper to solve [52]. The boundary integral reads

$$\int_{\Omega_{cell}} N_j(\boldsymbol{x})\,\mathrm{d}\Omega = \int_{\Gamma_{cell}} \boldsymbol{g}_i(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x})\,\mathrm{d}\Omega\,, \tag{4.6}$$

with $\boldsymbol{n}(\boldsymbol{x})$ being the normal vector on the surface and $\boldsymbol{g}_i$ denoting the antiderivatives [53] that can be easily computed in their closed form as follows

$$\boldsymbol{g}_i = \frac{1}{3} \begin{bmatrix} \int N_i\,\mathrm{d}x \\ \int N_i\,\mathrm{d}y \\ \int N_i\,\mathrm{d}z \end{bmatrix}. \tag{4.7}$$

4.3 Boundary conditions

Apart from the PDEs, the boundary value problem also defines boundary conditions that need to be satisfied by the solution. Two kinds of boundary conditions play a major role: (a) Dirichlet conditions, which set the solution at the boundary Γ_D and (b) Neumann conditions, which specify the normal derivative of the solution on the boundary Γ_N. Other boundary conditions, such as *Robin*, or *Cauchy*, are a combination of Dirichlet and Neumann conditions.

Neumann boundary conditions are also called natural boundary conditions, as they emerge naturally from the weak form (see Equation (4.1)). For the FCM, the Neumann boundary conditions are independent of α, and are, consequently, applied according to Equation (2.28) in an integral sense on the boundary Γ_N. Homogeneous Neumann conditions (i.e., zero traction) require no additional treatment, as they are automatically satisfied implicitly by setting $\alpha = 0$ or, in an approximate sense, to a small value in Ω_{fict}.

Dirichlet or essential boundary conditions are, unlike Neumann boundary conditions, imperative for the solution of the linear system of equations. The minimum amount of Dirichlet conditions restrict all rigid body motions and rotations. For the FCM, the Dirichlet boundary Γ_D typically does not coincide with the boundaries of the finite cell mesh. Consequently, the Dirichlet boundary conditions cannot be enforced in a strong sense, i.e., by applying a predefined solution directly on the degrees of freedom, but need to be enforced in a weak sense. For this, several methods have been adopted, such as *Lagrange Multipliers*, the *penalty method*, or *Nitsche's method* [54, 55, 56, 57], of which the first two are briefly explained in the following.

For the integration of Dirichlet and inhomogeneous Neumann boundary conditions, an explicit surface description is required, such as a surface triangulation. Since this surface description is only needed for the integration, it can be of poor quality and contain hanging nodes. Similar to the Neumann boundary conditions (see Equation (2.28)), the integration over the Dirichlet boundaries requires two additional mappings from the parametric space of the boundary $\boldsymbol{s}$ to the physical space $\boldsymbol{X}$ and further to the parametric space of the element $\boldsymbol{\xi}$.

4.3.1 Lagrange multipliers

Lagrange multipliers offer the possibility to satisfy the boundary conditions exactly. Following [58], the weak boundary conditions with Lagrange multipliers can be derived by finding the saddle point of an extended energy functional. For the discrete case this reads

$$\Pi_L^h = \frac{1}{2}\tilde{\boldsymbol{u}}^T \boldsymbol{K} \tilde{\boldsymbol{u}} - \tilde{\boldsymbol{u}}^T \boldsymbol{f} + \boldsymbol{\lambda}^T (\boldsymbol{A}\tilde{\boldsymbol{u}} - \boldsymbol{b}) \rightarrow \min_{\tilde{\boldsymbol{u}}}, \max_{\boldsymbol{\lambda}}\,, \tag{4.8}$$

which leads to a modified linear system of equations

$$\begin{bmatrix} \boldsymbol{K} & \boldsymbol{A}^T \\ \boldsymbol{A} & 0 \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{u}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{b} \end{bmatrix}\,. \tag{4.9}$$

$\boldsymbol{A}$ corresponds to the boundary condition matrix, and $\boldsymbol{b}$ carries the predefined displacements

$$A_{ij} = \int_{\Gamma_D} N_i^L N_j \,\mathrm{d}\Gamma\,, \tag{4.10}$$

$$b_i = \int_{\Gamma_D} N_i^L \tilde{\boldsymbol{u}} \,\mathrm{d}\Gamma\,. \tag{4.11}$$

N_i^L are the shape functions for the interpolation space of the Lagrange multipliers [58]. The Lagrange multipliers λ_i need only to be determined to enforce the boundary conditions and can be discarded after the computation.

Although being a very general approach, the Lagrange multiplier method has several drawbacks: (i) The number of unknowns, and so, the size of the *linear system of equations* (LSE) is increased, (ii) the resulting global matrix can have zero entries on the main diagonal and is no longer positive definite, (iii) most solvers are unsuited to find a solution for a saddle point problem and most central (iv) for too many Lagrange multipliers the linear system of equations becomes singular.

4.3.2 Penalty method

Another very popular approach is the penalty method, as it turned out to be robust and simple to implement [59, 60]. Additionally, the number of unknowns and, thus, the size of the LSE is not increased and the resulting system matrices remain positive definite, provided the penalty factor β is chosen sufficiently large. Similar to the Lagrange multipliers, the penalty method can be derived by minimizing an extended energy functional [58]

$$\Pi_P^h = \frac{1}{2}\tilde{\boldsymbol{u}}^T \boldsymbol{K} \tilde{\boldsymbol{u}} - \tilde{\boldsymbol{u}}^T \boldsymbol{f} + \frac{1}{2}\beta \parallel \boldsymbol{A}\tilde{\boldsymbol{u}} - \boldsymbol{b} \parallel^2 \rightarrow \min_{\tilde{\boldsymbol{u}}}\,, \tag{4.12}$$

which leads again to a modified linear system of equations

$$\left(\boldsymbol{K} + \beta \boldsymbol{M}^P\right) \tilde{\boldsymbol{u}} = \boldsymbol{f} + \beta \boldsymbol{f}^P\,, \tag{4.13}$$

with

$$M_{ij}^P = [\boldsymbol{A}^T \boldsymbol{A}]_{ij} = \int_{\Gamma_D} N_i N_j \,\mathrm{d}\Gamma\,, \tag{4.14}$$

$$f_i^P = [\boldsymbol{A}^T \boldsymbol{b}]_i = \int_{\Gamma_D} N_i \tilde{\boldsymbol{u}} \,\mathrm{d}\Gamma\,. \tag{4.15}$$

The penalty factor β can be considered as an artificial stiffness along the Dirichlet boundary. Only in the limit of $\beta \to \infty$, the penalty method fulfills the Dirichlet boundary conditions exactly. However, a too large choice for β leads to ill-conditioned system matrices. Consequently, a balance needs to be found between approximation accuracy and condition number. In the context of the FCM, the penalty factor β is limited even more, as the indicator function α already leads to small entries in the system matrices. Nevertheless, very often, the penalty method delivers accurate results for the FCM.

Chapter 5

Convergence and Refinement

On the way, from the mathematical description of physical phenomena to numerical simulations, certain assumptions need to be made, which introduce various errors. For a simulation with the FEM, or the FCM, typical errors are:

1. Modeling errors: The underlying differential equations describe an idealized problem with several restrictions. Also, the material parameters and behavior can not be resolved and determined to the finest scale and are, thus, approximated, averaged, or simply assumed.

2. Discretization errors: The actual object is represented by an idealized model, e.g., a CAD model. Furthermore, for classical FEM, the domain is approximated by the element-wise low-order shape functions that can neither capture the potentially curved shape, nor the boundary exactly.

3. Approximation errors: The 'exact' solution of the PDE can usually not be determined analytically and, hence, is approximated by the element-wise linear combination of shape functions.

4. Integration errors: The integration over the physical domain is typically performed with Gaussian quadrature. This integration is, however, only exact for polynomials. Even though the shape functions may be defined as polynomials, this is not necessarily the case for the external loads. Since for the FCM, the geometric shape is recovered during the integration, these errors are of special interest, as they may be a major cause for inaccurate results.

5. Numerical errors: Tolerances, precision, and rounding errors limit the accuracy.

The first two points are in the scientists' or engineers' responsibility to choose the best-suited analysis with reasonable material properties. When possible, or necessary, the results can be compared to experimental data. Experimental data can also be used to adjust the material properties. Points 3-5 relate to the numerical simulation itself. These errors are addressed by the question, whether the method is converging. The last point can limit the achievable precision of the method.

5.1 Convergence

Convergence describes the capability to reach – in the limit (infinitesimal element size $h \to 0$ and/or infinite polynomial degree $p \to \infty$) – the exact solution. In the following linear elasticity is assumed. Let

$\boldsymbol{u}_{ex}$ be the – in general unknown – exact solution and $\boldsymbol{u}_{fe}$ some finite element approximation. Then

$$\lim_{\substack{h \to 0 \\ and/or \\ p \to \infty}} \| \boldsymbol{u}_{ex} - \boldsymbol{u}_{fe} \|_{E(\Omega)} = 0\,, \tag{5.1}$$

where the energy norm is defined as

$$\| \boldsymbol{u} \|_{E(\Omega)} = \sqrt{\frac{1}{2}\mathcal{B}(\boldsymbol{u}, \boldsymbol{u})}\,. \tag{5.2}$$

For convergence, the method must be (i) *consistent* and (ii) *robust*. Consistency (i) comprises completeness and compatibility, where completeness means that the shape functions can approximate the analytical solution in the limit, which is approached by decreasing h and/or increasing p. Compatibility means that the shape functions ensure a displacement continuity between the elements. The robustness (ii) depends on rank sufficiency and Jacobian positiveness. Rank sufficiency means that apart from the rigid body motions no non-zero kinematic modes appear in the stiffness matrix. Jacobian positiveness is given if the determinant of the Jacobi matrix of the geometry is positive everywhere [23]. For the Bubnov-Galerkin approach of the FEM, the *best-approximation theorem* proofs that the finite element solution $\boldsymbol{u}_{fe}$ is always the optimal approximation of the exact solution $\boldsymbol{u}_{ex}$ with respect to the bilinear form $\mathcal{B}(\boldsymbol{u}, \boldsymbol{u})$, see Equation (2.7). This can be interpreted as least squares fit of $\boldsymbol{u}_{fe}$ in terms of the bilinear form. Consequently, for linear elasticity not the solution $\boldsymbol{u}_{fe}$, but the derivatives (i.e., stresses and strains) are optimized [24].

5.2 Refinement

Different refinement schemes can approach the limit – required for the convergence:

- $h-$refinement: The classical refinement strategy, where the element size h is decreased. For classical FEM, this is the only available refinement strategy.
- $p-$refinement: The element size is kept constant, and the polynomial degree of the shape functions is increased. Using hierarchical shape functions, this means that additional shape functions are added.
- $hp-$refinement: Elements are refined using both $h-$ and $p-$refinement. This refinement can be applied globally, or – to obtain the best-possible (exponential) convergence rate – adaptively depending on the smoothness of the solution on the respective elements [61], as described in Section 5.3.
- Hierarchic refinement: In the region of refinement, additional linear independent shape functions are added. This enrichment can entail the deactivation of some of the original shape functions. Hierarchic refinement also comprises the *extended finite element method* (XFEM), where the function space for the solution is enriched with discontinuous shape functions to resolve material interfaces, cracks, etc.

For the FCM, various hierarchic refinement strategies were developed for both, integrated Legendre and B-spline [62] shape functions – for instance, the $hp-d$ method [63, 64], or the (non-uniform) multi-level *hp*-method [65], as depicted in Figure 5.1. For higher dimensions, an adaptive hierarchical refinement leads to hanging nodes. Thus, the respective degrees of freedom need to be deactivated.

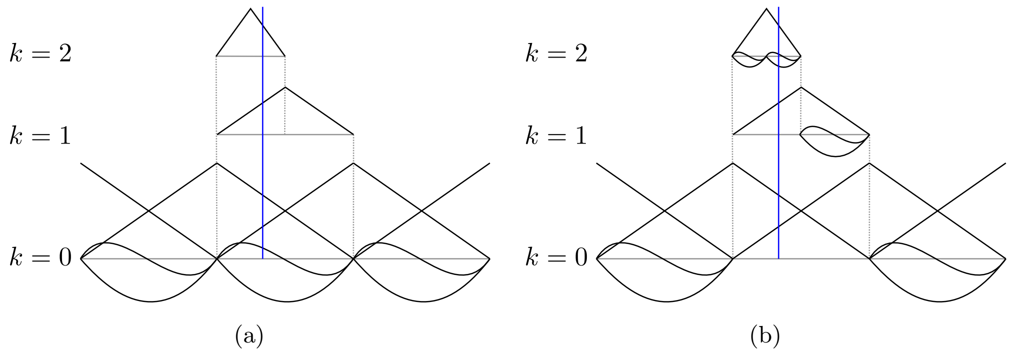


Figure 5.1: Comparison of (a) the $hp-d$ method and (b) the multi-level hp-method.

5.3 Convergence rate

The convergence rate indicates how fast the exact solution is approximated for a certain refinement. A possibility to measure the convergence, is to track certain representative points and compare their displacements to the exact solution. Most expressive is, however, to measure the relative error in the energy norm $(e_r)_{E(\Omega)}$ [26], since this value is determined by an integration over the entire domain

$$(e_r)_{E(\Omega)} = \frac{\parallel \boldsymbol{u}_{ex} - \boldsymbol{u}_{fe} \parallel_{E(\Omega)}}{\parallel \boldsymbol{u}_{ex} \parallel_{E(\Omega)}} \,, \tag{5.3}$$

The convergence rate depends on how good the shape functions can approximate the exact solution. The $h-$version shows always an algebraic convergence rate

$$\parallel \boldsymbol{u}_{ex} - \boldsymbol{u}_{fe} \parallel_{E(\Omega)} \leq \frac{k}{N^{\beta}} \,, \tag{5.4}$$

whereas the $p-$version, and so the FCM, show for smooth problems exponential convergence rates

$$\parallel \boldsymbol{u}_{ex} - \boldsymbol{u}_{fe} \parallel_{E(\Omega)} \leq \frac{k}{e^{\gamma N^{\theta}}} \,, \tag{5.5}$$

where k, β, θ and γ are all positive constants and N is the number of degrees of freedom. β depends on the polynomial degree and the smoothness of the solution and θ depends on the dimension [66].

Singularities have a significant impact on the convergence rate. For linear elasticity, these singularities can stem from geometrical features (e.g., sharp corners), boundary conditions (e.g., point bearings, or clamped surfaces), or loadings (e.g., concentrated loads). For non-smooth problems, the $p-$version of the FEM and the FCM do no longer converge exponentially but decrease to algebraic convergence[1].

A remedy to preserve exponential convergence rates – i.e., to obtain the best-achievable precision for the invested number of degrees of freedom – is offered by $hp-$refinement for which the mesh is refined with a geometric progression factor q towards singularities and the polynomial degree is increased suitably. As the location of the singularities is known in many cases, this refinement can be performed beforehand. Additionally, the smoothness of the solution must be determined to choose the appropriate polynomial degree. For smooth regions, large elements with a high polynomial degree, and for non-smooth regions, small elements with a low polynomial degree are chosen. Typically, error estimators are applied to evaluate the smoothness and indicate the elements, which need to be refined. For a detailed overview over different *a priori* and *a posteriori* (explicit and implicit) error estimators please refer to [67, 68, 69].

[1]In the case of a singularity, the convergence of $p-$FEM is still two times faster than $h-$refinement, provided the singularity is located at a node

5.4 Comparison of *h*-, *p*-FEM and the FCM

To compare the convergence behavior of the $h-$version and the $p-$version of the FEM, consider the one-dimensional bar example stated in [70] that is here extended to the FCM (see Figure 5.2). The bar has a length and stiffness of $l = 1$ and $EA = 1$, is fixed on the left boundary and loaded with two different line loads: f_1 resulting in a smooth solution and f_2 being singular at $x = 0$.

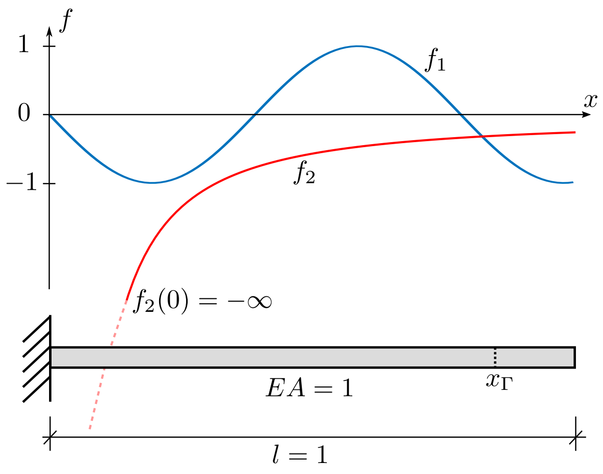


Figure 5.2: Convergence example: One-dimensional bar under different line loads.

The weak form of the one-dimensional problem reads

$$\int_0^1 u_{,x}\, \alpha EA\, v_{,x}\, \mathrm{d}x = \int_0^{x_\Gamma} f(x) v_x\, \mathrm{d}x\,. \tag{5.6}$$

For the $h-$version and the $p-$version of the FEM no fictitious domain is considered, i.e. the right boundary is at $x_\Gamma = l$ and thus $\alpha = 1$ everywhere. For the FCM, the domain is split at $x_\Gamma = 0.85$, dividing the domain into $\Omega_{phy} \in [0, x_\Gamma]$ and $\Omega_{fict} \in]x_\Gamma, 1]$.

For the FCM a boundary fitting composed integration – similar to the smart octree [50] is used. Yet, also an unfitting integration on a binary tree with a sufficiently large subdivision depth k leads to similar results. Additionally, the influence of the value for α in the fictitious domain is studied. The line loads are given as

$$f_1(x) = -sin(8x)\,, \tag{5.7}$$
$$f_2(x) = \lambda(\lambda - 1)x^{\lambda-2}\,, \tag{5.8}$$

where λ is chosen to be 0.65. The analytical solutions can be computed by integrating twice and considering 'no force' at $EA\, u_{,x} = 0\,|_{x_\Gamma}$ as additional boundary condition

$$u_{ex,1}(x) = -\frac{1}{64} sin(8x) + \frac{1}{8} cos(8x_\Gamma)x\,, \tag{5.9}$$
$$u_{ex,2}(x) = -x^\lambda + \frac{1}{(10x_\Gamma \cdot 10^\lambda)} 10\lambda(10x_\Gamma)^{\lambda x}\,. \tag{5.10}$$

The Figures 5.3 to 5.5 show the approximate solutions for $h-$, or $p-$refinements of the smooth problem f_1. As can be seen, the higher-order shape functions are better suited to capture the exact solution, which leads to an exponential convergence. The convergence of the $h-$version of the FEM is as expected algebraic (see Figure 5.6). Since the solution of the FCM is almost anti-symmetric (apart from the linear part) the symmetric shape functions (with even polynomial degree) are hardly contributing, which leads to a staircase-like convergence. The $p-$refinement reaches the numerical precision around 10^{-13} (actually 10^{-15}, since the relative error is stated in percent). The maximum accuracy for the FCM is limited by the chosen value for α in the fictitious domain.

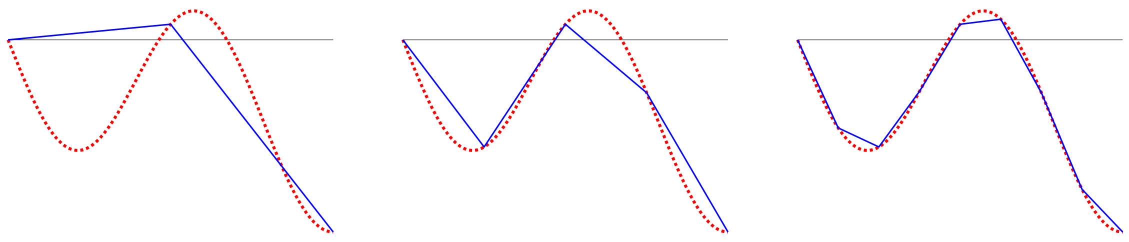

(a) $h = 0.5$ (b) $h = 0.25$ (c) $h = 0.125$

Figure 5.3: $h-$refinement of the smooth problem, where $p = 1$. The exact solution is dotted.

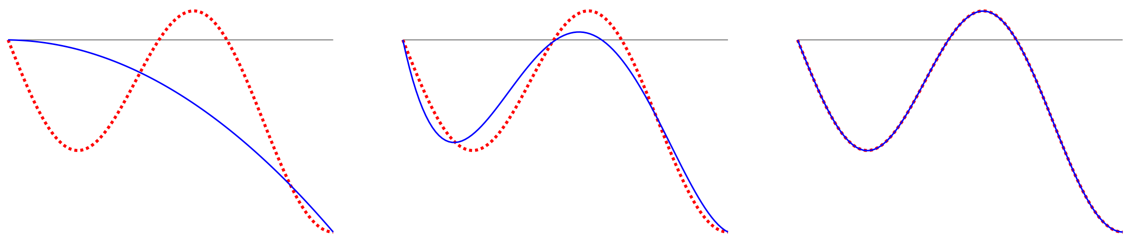

(a) $p = 2$ (b) $p = 4$ (c) $p = 8$

Figure 5.4: $p-$refinement of the smooth problem, where $h = 1$. The exact solution is dotted.

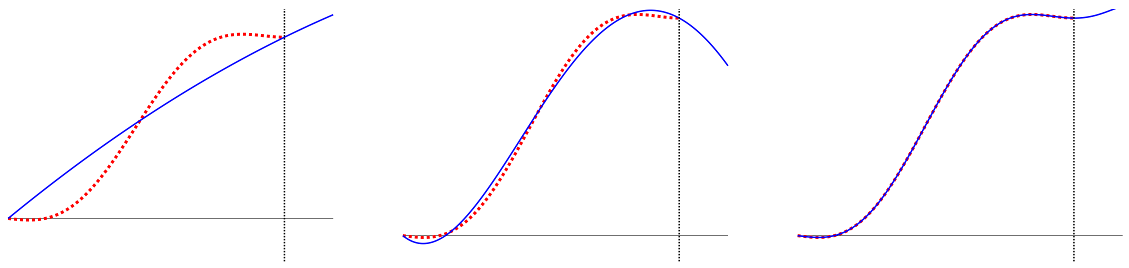

(a) $p = 2$ (b) $p = 4$ (c) $p = 8$

Figure 5.5: FCM with $p-$refinement of the smooth problem and $x_\Gamma = 0.85$, where $h = 1$. The exact solution is dotted.

The second load case introduces a singularity at $x = 0$, where $f_2(x = 0) = -\infty$. This leads to infinite strain and, consequently, a vertical tangent of the solution. The shape functions are not able to represent

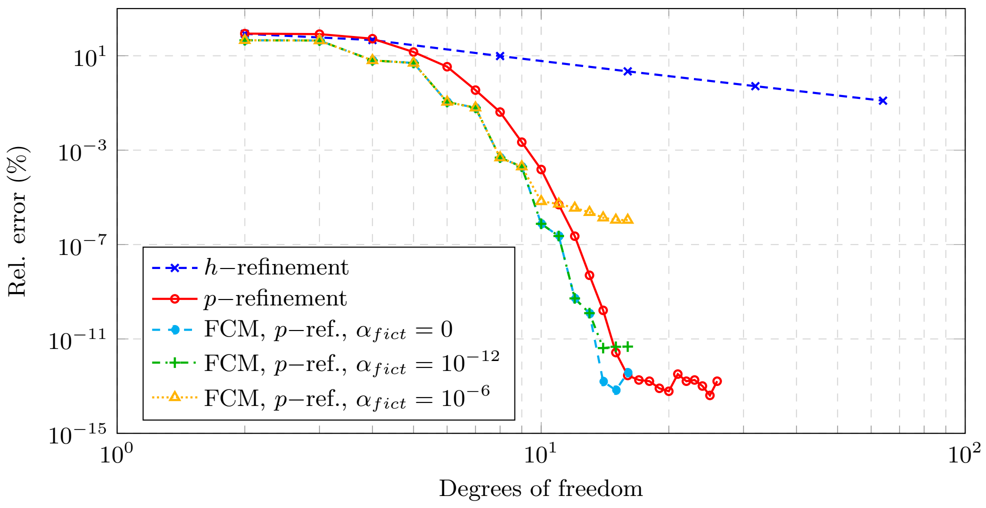


Figure 5.6: Convergence of the smooth problem.

this singularity. Consequently, the $p-$version and the FCM fall back to algebraic convergence, as depicted in Figure 5.7. However, the convergence rate is still higher compared to the $h-$version.

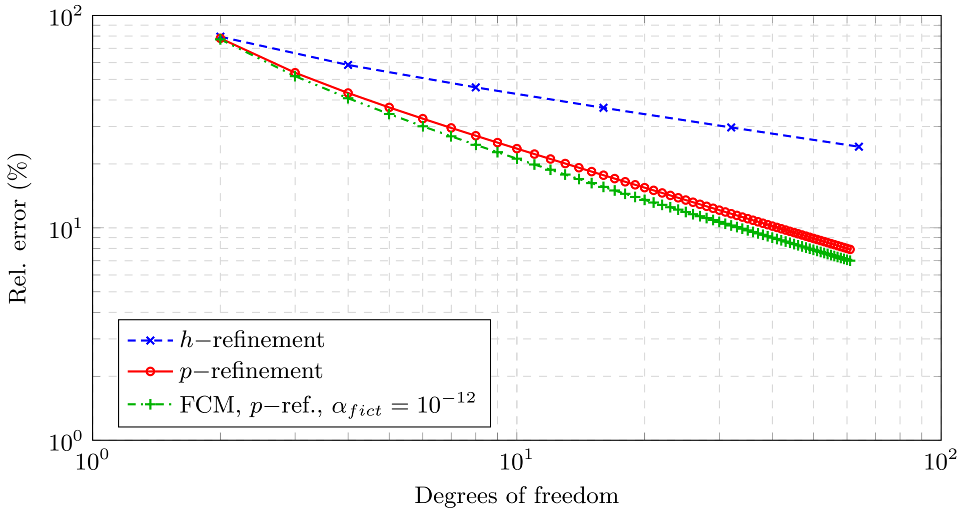


Figure 5.7: Convergence of the singular problem.

Using an adaptive h-refinement towards the singularity, with a subsequent uniform $p-$refinement, yields a pre-asymptotic, exponential convergence. However, the uniform refinement ignores the smoothness of the solution, which then leads to a leveling-off into algebraic convergence. With hp-refinement, which takes the smoothness into account and adapts the polynomial degree on each element accordingly, it is possible to regain an exponential convergence. Figures 5.8 and 5.9 show the convergence for the $p-$version and the FCM, respectively. The geometric progression factor is set to $q = 0.2$ and $\alpha_{fict} = 10^{-12}$. For the $hp-$version the polynomial degrees are increased linearly from $p_{min} = 1$ on the smallest element to $p_{max} = \max(1, \frac{n_e}{5})$ on the largest element, where n_e is the number of elements. Carrying out 35 refinements yields a maximum polynomial degree of $p_{max} = 7$.

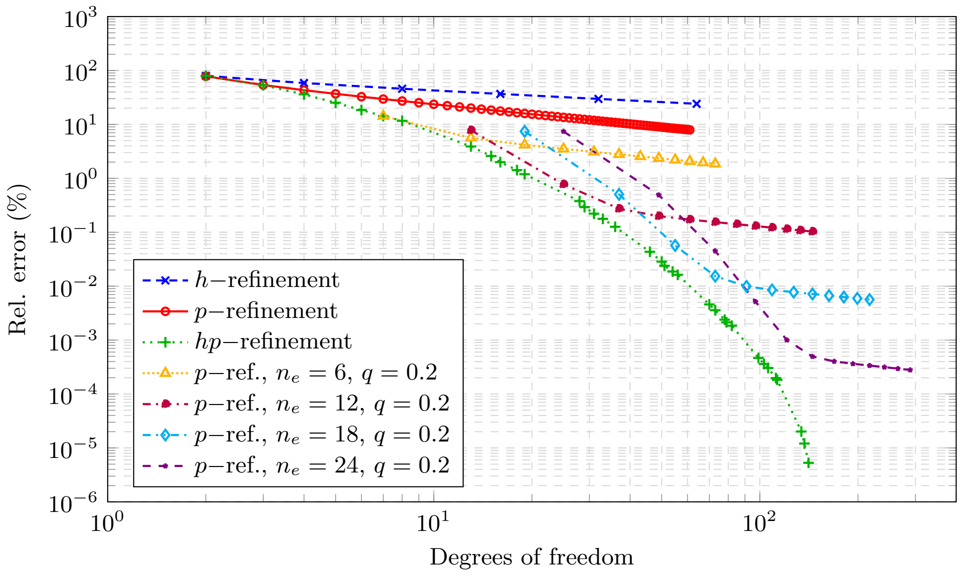


Figure 5.8: Convergence of the $p-$version of the FEM for the singular problem with adaptive refinements.

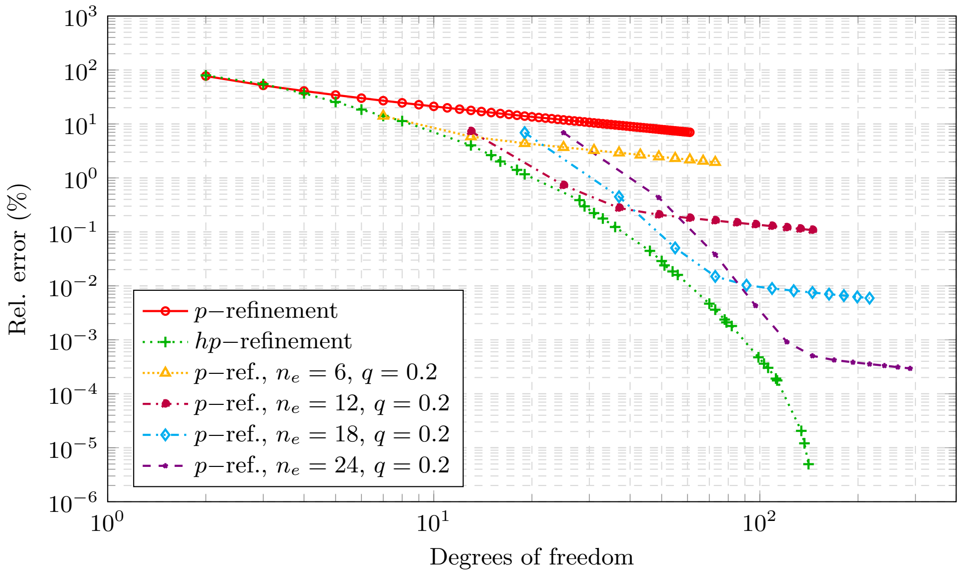


Figure 5.9: Convergence of the FCM for the singular problem with adaptive refinements and $\alpha_{fict} = 10^{-12}$.

Part II

Geometric modeling

Nowadays, CAD is *the* method of choice to generate all kinds of geometric models. Therefore, CAD models serve multiple purposes and are, in general, not solely created for numerical simulations. In the context of the FCM, the only information required from a CAD model is the point membership classification, as explained in Section 4.1. This part provides an overview of the specific geometric modeling techniques to understand the respective potentials and pitfalls regarding a simulation with the FCM. For the sake of completeness, non-CAD models, such as point-clouds and spatial-decomposition models, are briefly discussed as well. Additionally, the concept of *flawed geometrical models* and the possibilities to represent *functionally graded materials* are presented. However, first, the concept of splines is introduced, since they form the basis of modern CAD systems to describe curves, surfaces, and volumes.

Chapter 6

Splines

The term *spline* stems originally from shipbuilding, where curves were generated with slim wooden bars – called splines – that were fixated at several points. The resulting shape minimizes the potential energy of a corresponding mechanical beam model and can be described by piece-wise cubic polynomials. However, the set of possible shapes is very limited, so cubic splines are only used in applications where minimal potential energy is crucial. Only after the invention of the *Bézier* curves in the 1960s, CAD began its triumphant. It was now possible to model a large variety of free-form shapes. Nowadays, CAD systems use *B-splines*, or *NURBS*, which combine the idea of piece-wise polynomials with the parametric Bézier curves. Please note that, within this thesis, the term 'spline' is used as an umbrella name for Bézier, B-spline, and NURBS representations. Spline shape functions M_i fulfill the *partition of unity*

$$\sum_i M_i(\xi) = 1\,, \quad \forall \xi \in [\xi_{start}, \xi_{end}]\,, \tag{6.1}$$

and can thus naturally be used as basis functions for a FEM discretization, as it is the case for the isogeometric analysis[1]. The notation in this chapter is based mainly on the book of Piegl and Tiller [71].

6.1 Bézier curves

Bézier curves, developed by Pierre Bézier at Renauld cars in 1962 [72], are parametric curves that interpolate the starting and ending point, but only approximate the so-called *control points* in-between. This leads to fewer oscillations compared to interpolating curves, such as parametric Lagrange polynomials. A Bézier curve can be created geometrically by *de Casteljau*'s algorithm, or expressed as a linear combination of shape functions as follows

$$\boldsymbol{C}_{Bezier}(\xi) = \sum_{i=1}^{n} B_{i,p}(\xi)\boldsymbol{P}_i\,, \quad \xi \in [0,1]\,, \tag{6.2}$$

where $B_{i,p}$ is the *Bernstein* polynomial that corresponds to the control point $\boldsymbol{P}_i$. In the case of a mere geometrical description, the control points are defined as $\boldsymbol{P}_i^T = [x_i, y_i, z_i] \in \mathbb{R}^3$ for a three-dimensional curve, or $\boldsymbol{P}_i^T = [x_i, y_i] \in \mathbb{R}^2$ for a planar curve. However, the dimension of the control points can be arbitrarily extended to $\mathbb{R}^{d+s}$, where d denotes the 'geometrical' dimension of the curve and $s \geq 0$

[1] *Remark:* For the representation of rigid body motions, basis functions must be able to represent a constant value over the domain, which is naturally given when they fulfill the partition of unity. However, this is not a necessary condition as it was illustrated with the hierarchic Legendre polynomials in the previous part.

additional dimensions that can store further information (see Section 9.2.1). The Bernstein polynomials are computed as follows

$$B_{i,p}(\xi) = \binom{p}{i-1} \xi^{i-1} \cdot (1-t)^{p-i+1}\,, \quad i \in \{1, ..., n\}. \tag{6.3}$$

Note that the polynomial degree p depends on the number of control points $p = n - 1$ and that the Bernstein polynomials have support over the entire parameter space, as can be seen in Figure 6.1a. Two properties limit the applicability for CAD: (a) With a rising number of control points, the (average) distance between control points and curve increases, and (b) the change of any control point affects the entire shape (see Figure 6.2a).

6.2 B-spline curve

B-splines (short for basis-splines) were developed to overcome the deficits of the Bézier curves. With the introduction of a *knot vector* in the parameter space, it is possible to generate piece-wise, smooth, approximative curves, for which the polynomial degree p is independent of the number of control points n, as depicted in the Figures 6.1b and 6.2b. In contrast to Bézier curves, this leads to a limited, local support/influence of each control point. The expression for the computation of the points on the B-spline curve reads

$$\boldsymbol{C}_B(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\boldsymbol{P}_i\,, \quad \xi \in [\xi_1, \xi_m]\,. \tag{6.4}$$

The n B-spline shape functions can be computed with *Cox–de Boor*'s recursive formula

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi), \quad i \in \{1, 2, ..., n\}, \tag{6.5}$$

where ξ_i denotes the i^{th} knot of the non-descending knot vector $\xi_i \in \Xi = [\xi_1, ..., \xi_m]$ with $m = n + p + 1$ being the required number of knots. The basic, constant shape functions are defined piece-wise on the non-zero knot spans

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if} \quad \xi \in [\xi_i, \xi_{i+1}[\\ 0 & \quad\text{else} \end{cases} \,. \tag{6.6}$$

The k^{th} derivative of the curve is evaluated as

$$\boldsymbol{C}_B^{(k)}(\xi) = \frac{\mathrm{d}^k \boldsymbol{C}_B(\xi)}{\mathrm{d}\xi^k} = \sum_{i=1}^{n} \frac{\mathrm{d}^k N_{i,p}(\xi)}{\mathrm{d}\xi^k} \boldsymbol{P}_i = \sum_{i=1}^{n} N_{i,p}^{(k)} \boldsymbol{P}_i\,, \tag{6.7}$$

where the k^{th} derivates of the shape functions are computed as follows

$$N_{i,p}^{(k)} = p \left(\frac{N_{i,p-1}^{(k-1)}(\xi)}{\xi_{i+p} - \xi_i} - \frac{N_{i+1,p-1}^{(k-1)}(\xi)}{\xi_{i+p+1} - \xi_{i+1}} \right), \quad i \in \{1, 2, ..., n\}. \tag{6.8}$$

Crucial for the shape and continuity of the curve is the knot-vector $\Xi = [\xi_1, ..., \xi_i, ... \xi_m] \subset \mathbb{R}$. To interpolate the first and last control point, the *knot-multiplicity* κ of the first and last knot must be $p+1$, hence $\xi_1 = \xi_2 = ... = \xi_{p+1}$ and $\xi_{end} = \xi_{end-1} = ... = \xi_{end-p}$. This is called an *open knot-vector*. The continuity of the curve is C^∞ everywhere, except on the knots, where it is decreased to $C^{p-\kappa}$. This can be used purposefully to represent kinks or gaps on the curve.

6.3 NURBS curves

B-splines allow to create a large variety of free-form shapes. However, certain commonly used shapes, such as circles, or spheres cannot be represented exactly. For this, *NURBS* were developed. NURBS stands for *non-uniform rational basis splines*. The basic idea is to introduce *weights* $w_i \in [0, 1]$ which control the influence of each control point (see Figures 6.1c and 6.2c)

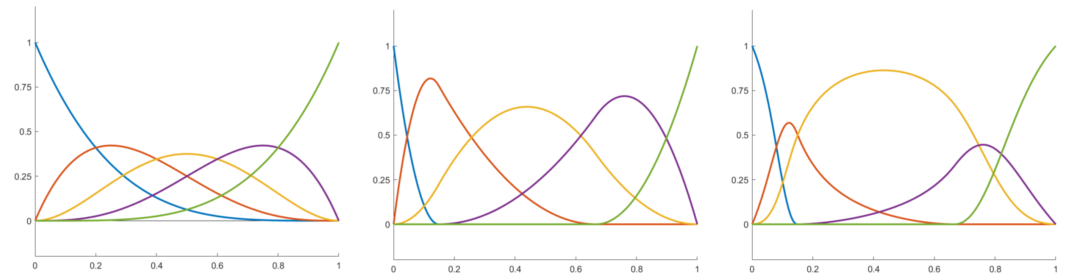
$$\boldsymbol{C}_N(\xi) = \sum_{i=1}^{n} R_{i,p}(\xi)\tilde{\boldsymbol{P}}_i\,, \quad \xi \in [\xi_1, \xi_{end}], \tag{6.9}$$

where $\tilde{\boldsymbol{P}}_i^T = [x_i, y_i, z_i, w_i]$ are the standard NURBS control points in three dimensions, and $R_{i,p}$ correspond to a rational shape function that is evaluated based on the B-spline shape functions

$$R_{i,p}(\xi) = \frac{N_{i,p}(\xi)w_i}{\sum_{i=1}^{n} N_{i,p}(\xi)w_i}\,. \tag{6.10}$$

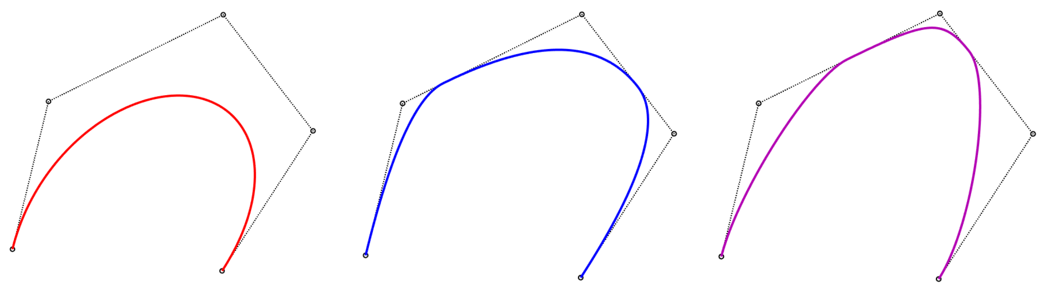
The denominator acts as normalization to preserve the partition of unity.

Although NURBS can represent more shapes exactly, they come along with increased complexity. Also, the desirable features of the polynomial base are lost. For example, a numerical integration on NURBS can, in general, only be performed approximately, as standard quadrature rules do not apply for rational functions.



(a) Bézier, $p \overset{!}{=} 4$ (b) B-spline, $p = 2$ (c) NURBS, $p = 2$

Figure 6.1: Spline shape functions for five control points, knot vector $\boldsymbol{\Xi} = [0\,0\,0\,\frac{3}{20}\,\frac{2}{3}\,1\,1\,1]$ and weights $\boldsymbol{w} = [1\,\frac{3}{10}\,1\,\frac{3}{10}\,1]$.



(a) Bézier, $p \overset{!}{=} 4$ (b) B-spline, $p = 2$ (c) NURBS, $p = 2$

Figure 6.2: Spline curves with the same curve parameters as in Figure 6.1.

6.4 Spline surfaces and volumes

Similar to the shape functions for the FEM (see Section 2.2), higher dimensional splines – most commonly surfaces or volumes – are created as tensor product of uni-variate splines. Consequently, a bi-variate surface is described as

$$\boldsymbol{S}(\boldsymbol{\xi}) = \sum_{i=1}^{l} \sum_{j=1}^{m} M_{i,p}(\xi) M_{j,q}(\eta) \boldsymbol{P}_{i,j} \,, \tag{6.11}$$

where $\boldsymbol{S}(\boldsymbol{\xi})$ is a point on the surface and $\boldsymbol{\xi} = [\xi, \eta]^T$ is the corresponding two-dimensional parameter position in the parameter space $\boldsymbol{\xi} \in \Xi \times \mathrm{H} \subset \mathbb{R}^2$. $M_{i,p}$ denotes the i^{th} uni-variate spline basis function (Bézier, B-spline, or NURBS) of polynomial degree p. $\boldsymbol{P}_{i,j}$ are the $l \cdot m$ control points of the corresponding control point mesh. Note, in the case of NURBS, the control points are $\tilde{\boldsymbol{P}}$, according to Equation (6.9). Tri-variate volumes are described analogously

$$\boldsymbol{V}(\boldsymbol{\xi}) = \sum_{i=1}^{l} \sum_{j=1}^{m} \sum_{k=1}^{n} M_{i,p}(\xi) M_{j,q}(\eta) M_{k,r}(\zeta) \boldsymbol{P}_{i,j,k} \,, \tag{6.12}$$

with the difference, that $\boldsymbol{\xi} = [\xi, \eta, \zeta]^T$ corresponds to a three-dimensional parameter space $\boldsymbol{\xi} \in \Xi \times \mathrm{H} \times \mathrm{Z} \subset \mathbb{R}^3$ and $\boldsymbol{P}_{i,j,k}$ are the $l \cdot m \cdot n$ points of the three-dimensional control point mesh.

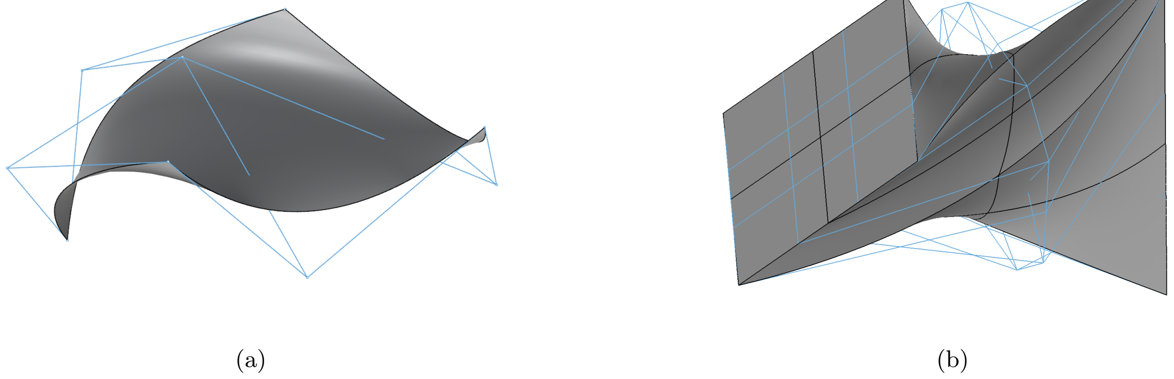

(a) (b)

Figure 6.3: Higher dimensional splines with their corresponding control point meshes: (a) surface (generated with Rhinoceros®), and (b) volume (generated with GuIrit [73]).

6.5 Inverse mapping

The *inverse mapping* denotes the transformation from the physical – or real – space $\boldsymbol{x}$ to the parameter space of the spline $\boldsymbol{\xi}$. In general, an analytic inversion of spline mappings is not available. Therefore, this mapping can only be applied for an individual point $\boldsymbol{R}$. Consequently, inverse mappings of higher-dimensional shapes, such as curves, surfaces, or volumes must be reduced to individual point mappings. For the case, that $\boldsymbol{R}$ is not on the spline, the closest point on the spline is sought $\boldsymbol{Q} \in \boldsymbol{C}(\xi)$, or $\boldsymbol{S}(\boldsymbol{\xi})$, or $\boldsymbol{V}(\boldsymbol{\xi})$. In general, this mapping cannot be solved in closed form – this is only applicable for $p \leq 4$. Using the property that the closest point is perpendicular to the spline, the inverse mapping can be formulated as a root finding problem [71]. For the one-dimensional case this reads

$$f(\xi) = \boldsymbol{C}_{,\xi}(\xi) \cdot (\boldsymbol{C}(\xi) - \boldsymbol{R}) \overset{!}{=} 0\,, \tag{6.13}$$

where $\boldsymbol{C}_{,\xi}$ denotes the tangential vector, i.e. the gradient of the curve $\boldsymbol{C}$ (see Figure 6.4). The zero(s), or roots are typically found by a *Newton iteration*

$$\xi^{j+1} = \xi^j - \frac{f(\xi^j)}{f'(\xi^j)} = \xi^j - \frac{\boldsymbol{C}_{,\xi}(\xi^j) \cdot \left(\boldsymbol{C}(\xi^j) - \boldsymbol{R}\right)}{\boldsymbol{C}_{,\xi\xi}(\xi^j) \cdot (\boldsymbol{C}(\xi^j) - \boldsymbol{R}) + \boldsymbol{C}_{,\xi}(\xi^j) \cdot \boldsymbol{C}_{,\xi}(\xi^j)}\,, \tag{6.14}$$

where $j \in \{0, ..., end\}$ is an iteration step, and $\boldsymbol{C}_{,\xi\xi}$ denotes the curvature vector, i.e. the second derivative.

For two dimensions, the inverse mapping relates to the projection onto a surface $\boldsymbol{S}(\xi, \eta)$, and the formulation extends to

$$f(\xi, \eta) = \boldsymbol{S}_{,\xi}(\xi, \eta) \cdot \boldsymbol{r}(\xi, \eta) \overset{!}{=} 0\,, \tag{6.15}$$

$$g(\xi, \eta) = \boldsymbol{S}_{,\eta}(\xi, \eta) \cdot \boldsymbol{r}(\xi, \eta) \overset{!}{=} 0\,, \tag{6.16}$$

where $\boldsymbol{S}_{,\xi}$ and $\boldsymbol{S}_{,\eta}$ are the two linear independent tangent vectors on the surface and $\boldsymbol{r}$ is the vector pointing from $\boldsymbol{R}$ to the surface

$$\boldsymbol{r}(\xi, \eta) = \boldsymbol{S}(\xi, \eta) - \boldsymbol{R}\,. \tag{6.17}$$

The roots are again sought with the Newton iteration, leading to the following linear system of equations

$$\begin{bmatrix} \boldsymbol{S}_{,\xi} \cdot \boldsymbol{S}_{,\xi} + \boldsymbol{S}_{,\xi\xi} \cdot \boldsymbol{r} & \boldsymbol{S}_{,\xi} \cdot \boldsymbol{S}_{,\eta} + \boldsymbol{S}_{,\xi\eta} \cdot \boldsymbol{r} \\ \boldsymbol{S}_{,\eta} \cdot \boldsymbol{S}_{,\xi} + \boldsymbol{S}_{,\eta\xi} \cdot \boldsymbol{r} & \boldsymbol{S}_{,\eta} \cdot \boldsymbol{S}_{,\eta} + \boldsymbol{S}_{,\eta\eta} \cdot \boldsymbol{r} \end{bmatrix}_{(\xi^j, \eta^j)} \begin{bmatrix} \xi^{j+1} - \xi^j \\ \eta^{j+1} - \eta^j \end{bmatrix} = - \begin{bmatrix} f(\xi^j, \eta^j) \\ g(\xi^j, \eta^j) \end{bmatrix}\,. \tag{6.18}$$

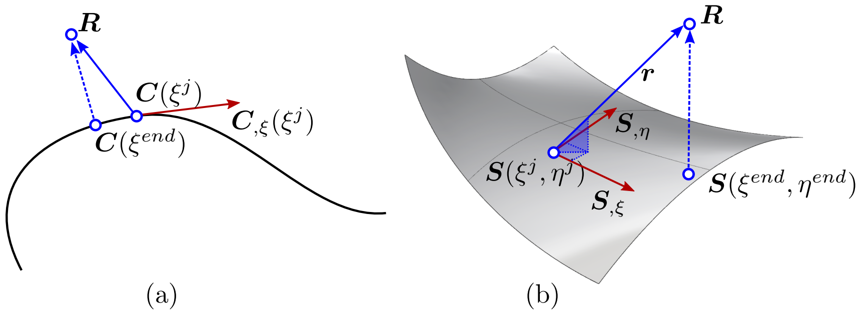

$\boldsymbol{R}$
$\boldsymbol{C}(\xi^j)$
$\boldsymbol{C}_{,\xi}(\xi^j)$
$\boldsymbol{C}(\xi^{end})$
(a)
$\boldsymbol{R}$
$\boldsymbol{r}$
$\boldsymbol{S}_{,\eta}$
$\boldsymbol{S}(\xi^j, \eta^j)$
$\boldsymbol{S}_{,\xi}$
$\boldsymbol{S}(\xi^{end}, \eta^{end})$
(b)

Figure 6.4: Inverse mapping on (a) a curve, and (b) a surface.

The extension to three dimensions is straightforward. Since the Newton iteration is highly dependent on good initial values $\boldsymbol{\xi}^0$, the spline is typically approximated with a polygonal mesh beforehand, and the parameter position of the corresponding closest mesh point is used for $\boldsymbol{\xi}^0$. As the spline is only defined inside the knot-vector, it must be ensured that $\xi^{j+1} \in \boldsymbol{\Xi}$. If more than one point is perpendicular to the spline's tangent space, Equations (6.13) and (6.16) have multiple roots. For a limiting example, consider the midpoint of a circle, where each curve point is the closest point. For the correct inverse mapping, all roots need to be found, and the respective distances must be compared. Unfortunately, with Newton iteration, it cannot be guaranteed that all roots are found, even if multiple suited initial values are given.

Nevertheless, for most applications, this inverse mapping is robust. Typically, the accuracy of the mapping – in terms of the numerical tolerances – is more critical. It is noteworthy that the computational cost for the inverse mapping of splines can be quite significant.

Remark: The herein presented inverse mapping is not limited to splines but can be used for any kind of parametric curve, surface, or volume description.

6.6 Trimming

The concept of *trimming* is crucial for the generation of free form shapes. In many real-world CAD models, the majority of surfaces are trimmed. For the sake of simplicity, only the bivariate case of a trimmed surface is considered. The concept extends naturally to higher dimensions.

A trimmed surface consists of a bi-variate surface $\boldsymbol{S}(\boldsymbol{\xi}) \in \mathbb{R}^3$ and one, or more additional uni-variate trimming curves $\boldsymbol{C}_i^{Trim}(\xi) \in \mathbb{R}^3$. The trimming curves divide the surface into an inside and an outside part. Usually, they are selected by the designer in the physical space $\boldsymbol{x} \in \mathbb{R}^3$ and need then to be mapped to the surface's parameter space $\boldsymbol{\xi} \in \mathbb{R}^2$. Even though the trimming curve on the surface might be exactly representable, in general, the required polynomial degree is unreasonably high, and the curve is approximated.

To this end, an additional mapping space is introduced (see Figure 6.5). The basic idea is to map a discrete set of physical coordinates of the trimming curve $\boldsymbol{C}_i^{Trim}(\xi)$ to the parameter space of the surface $\boldsymbol{\xi}$. These mapped points (on the parameter space) are then used as interpolation, or approximation points of another uni-variate two-dimensional trimming curve $\tilde{\boldsymbol{C}}_i^{Trim}(\xi) \in \mathbb{R}^2$. Certainly, the accuracy is highly dependent on the tolerances of the inverse mapping and the resolution of the curve fitting $\tilde{\boldsymbol{C}}_i^{Trim}(\xi)$.

Usually, a model consists of many trimmed surfaces, which are *joined* at their common boundaries, i.e., the trimming curves. Since on each corresponding surface, the mapping of the trimming curves may yield a slightly different approximation, it is, in general, not possible to gain a perfectly accurate, watertight joined surface. This inaccuracy can be a crucial pitfall regarding subsequent meshing and direct approaches, such as IGA.

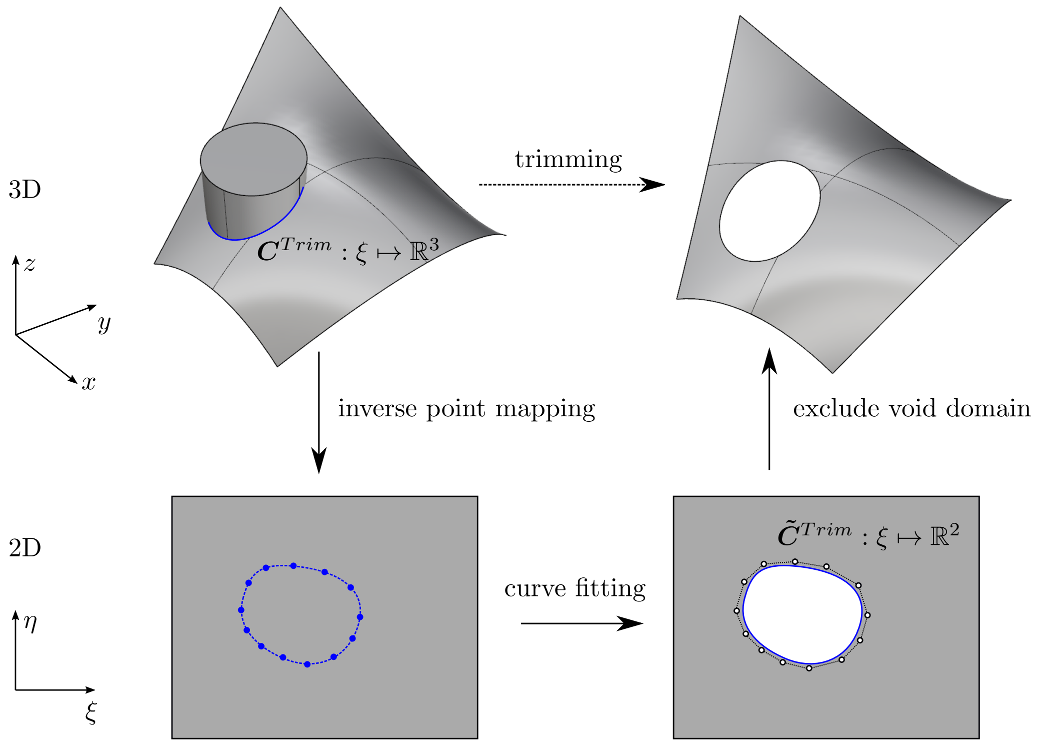

3D
trimming
$\boldsymbol{C}^{Trim} : \xi \mapsto \mathbb{R}^3$
z
y
x
inverse point mapping
exclude void domain
2D
η
ξ
curve fitting
$\tilde{\boldsymbol{C}}^{Trim} : \xi \mapsto \mathbb{R}^2$

Figure 6.5: Trimming of a surface with a trimming curve $\boldsymbol{C}^{Trim}$.

6.7 Refinements

Modeling or simulating with splines often requires modifying their parameters without changing the actual shape of the curve, surface, or volume. The form defining parameters are the knots in the knot-vector, the polynomial degree of the shape functions, and the control points' location and weights. Analogously to the FEM, two different refinement schemes are possible: (a) knot insertion, which corresponds to the $h-$refinement of the FEM, and (b) degree elevation, which relates to $p-$refinement[2]. For reasons of clarity, only the one-dimensional cases are explained.

6.7.1 Weight projection

In the case of NURBS, it is necessary to transform the control points $\tilde{\boldsymbol{P}}_i^T = [x_i, y_i, z_i, w_i]$ into *projective* control points before the refinement

$$\hat{\boldsymbol{P}}_i^T = \begin{bmatrix} \hat{x}_i & \hat{y}_i & \hat{z}_i & w_i \end{bmatrix} = \begin{bmatrix} x_i w_i & y_i w_i & z_i w_i & w_i \end{bmatrix} . \tag{6.19}$$

After the refinement the projective control points are transformed back into NURBS control points.

$$\tilde{\boldsymbol{P}}_i^T = \begin{bmatrix} \hat{x}_i/\hat{w}_i & \hat{y}_i/\hat{w}_i & \hat{z}_i/\hat{w}_i & w_i \end{bmatrix} . \tag{6.20}$$

In the following of this section, all control points are referred to as $\boldsymbol{P}_i$, yet, one should bear in mind that for NURBS, these are the projective control points $\hat{\boldsymbol{P}}_i$.

6.7.2 Knot insertion

By introducing a new knot ξ_{new} into the knot vector, the number of shape functions, and thus, the number of control points is increased by one.

$$\boldsymbol{\Xi} = [\xi_1, ..., \xi_{new}, ..., \xi_{end}]. \tag{6.21}$$

The shape of the curve is preserved only by the locations of the new control points since the polynomial degree remains unchanged. Let ξ_{new} be in the knot span $\xi_{new} \in [\xi_i, \xi_{i+1}[$. Then, p new control points $\boldsymbol{Q}_j$ need to be found on the legs of the control polygon from $\overline{\boldsymbol{P}_{i-p}\boldsymbol{P}_{i-p+1}}$ to $\overline{\boldsymbol{P}_{i-1}\boldsymbol{P}_i}$, i.e., the lines between affected control points[3] (see Figure 6.6a). The new control points are computed as follows

$$\boldsymbol{Q}_j = (1-\alpha_j)\boldsymbol{P}_{j-1} + \alpha_j \boldsymbol{P}_j \, , \tag{6.22}$$

with

$$\alpha_j = \frac{\xi_{new} - \xi_j}{\xi_{j+p} - \xi_j}, \quad j \in \{i-p+1, ..., i\} \, . \tag{6.23}$$

Note, that the first and last new control point are the same as in the original control polygon $\boldsymbol{Q}_{i-p} = \boldsymbol{P}_{i-p}$ and $\boldsymbol{Q}_{i+1} = \boldsymbol{P}_i$.

[2]It is noteworthy that the preservation of the shape can only be guaranteed for a degree elevation and a knot insertion, not the inverse operations, i.e., degree decrease or knot deletion.

[3]The p new control points therewith replace $p-1$ old control points and add one new control point.

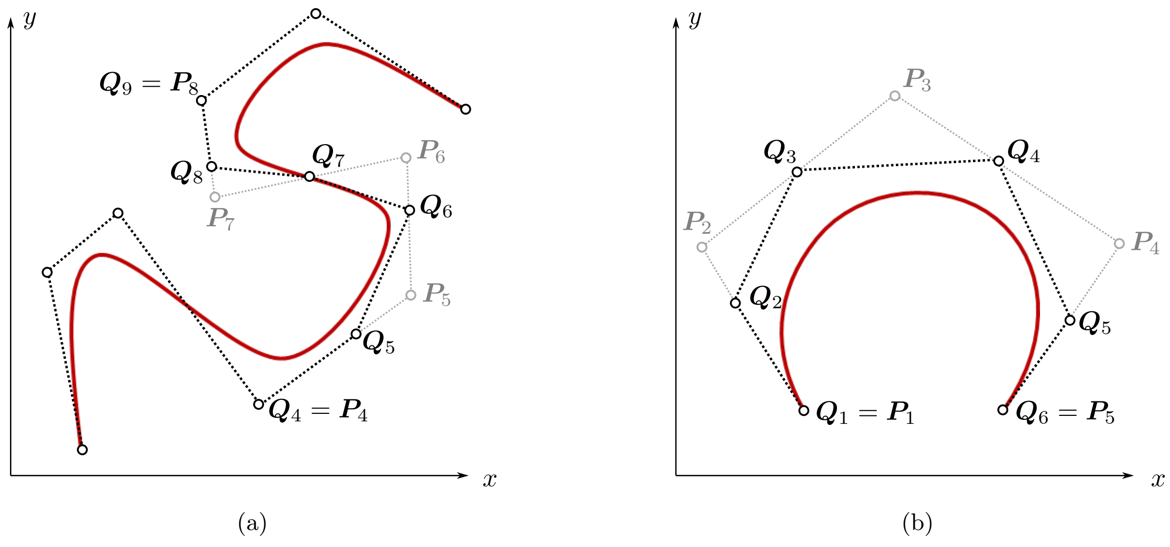


Figure 6.6: Refinements for splines: (a) knot insertion for a B-spline curve of polynomial degree of $p = 4$ with the new knot in the knot-span $\xi_{new} \in [\xi_8, \xi_9[$ and (b) degree elevation of a Bézier segement.

6.7.3 Bézier extraction

Since the continuity depends on the multiplicity of the knots $C^{p-\kappa}$, it is possible to extract a Bézier element for each patch (i.e., a non-zero knot span $\xi_{i+1} - \xi_i \neq 0$). To this end, knots are inserted until the multiplicity at each knot is $k = p + 1$. This knot insertion leads to a set of individual, mathematically independent curves, surfaces, or volumes, as the shape functions are discontinuous or C^0-continuous at the knots. However, the original geometrical continuity of the curve is preserved by the newly introduced control points. At the knots, the limiting control points of a patch equal the limiting control points of the adjacent patches $\boldsymbol{P}_{end}^{i-1} = \boldsymbol{P}_1^i$ and $\boldsymbol{P}_{end}^i = \boldsymbol{P}_1^{i+1}$.

Another variant of Bézier extraction for B-splines – often used in FEM – does not change the spline parameters but expresses the piece-wise defined shape functions on each patch with the Bernstein polynomials. Since Bernstein polynomials and B-spline shape functions represent the same function space, a straightforward mapping can be defined

$$\boldsymbol{N} = \boldsymbol{C}\boldsymbol{B}\,, \tag{6.24}$$

where $\boldsymbol{N}$ are the B-spline shape functions on a patch, $\boldsymbol{B}$ denote the Bernstein polynomials of the same polynomial degree as $\boldsymbol{N}$, and $\boldsymbol{C}$ is the (non-quadratic) mapping matrix containing the coefficients that are weighting the corresponding contribution of each Bernstein polynomial to the B-spline shape functions.

6.7.4 Degree elevation

The degree elevation involves three steps. First, the curve is subdivided at each knot into Bézier segments using knot insertion. Then, the polynomial degree on each Bézier segment is increased, and finally the curves are merged by deleting knots and control points at the contact points of the Bézier segments. Similar to the knot-insertion, for the degree elevation of the Bézier curve, the number and location of the control points is changed. Keeping the first and last control point $\boldsymbol{Q}_1 = \boldsymbol{P}_1$ and $\boldsymbol{Q}_n = \boldsymbol{P}_{n-1}$, the location of the new control points read

$$\boldsymbol{Q}_i = \frac{i}{n}\boldsymbol{P}_{i-1} + \left(1 - \frac{i}{n}\right)\boldsymbol{P}_i, \quad i \in \{2, 3, ..., n-1\}, \tag{6.25}$$

with $n = p_{new} + 1$ being the number of new control points (see Figure 6.6b).

6.8 Spline fitting

Splines are often used for curve, surface, or volume fitting. The properties to be fitted can be geometrical coordinates and more abstract data, such as material properties. For this, two different types of fittings are available: (a) *interpolation* through a finite set of points and (b) *approximation* – typically using the *least squares approach*. The first approach yields as many control points as given interpolation points. Consequently, for a large number of sample points, noisy point clouds, or continuous functions, spline approximation is chosen. In the following, only B-spline curves are considered. However, fitting extends naturally to higher dimensions and NURBS. Bézier splines are also possible, however, due to their global support less suited.

6.8.1 Interpolation

To set up a B-spline curve, the polynomial degree p, the knot-vector $\xi_i \in \boldsymbol{\Xi}$ and the control points $\boldsymbol{P}_i$ are needed. The polynomial degree can be chosen arbitrarily up to a Bézier representation $p \leq n-1$, where n is the number of control points $\boldsymbol{P}_i$ and interpolation points $\boldsymbol{Q}_i$ (see Figure 6.7). First, the knot-vector is defined. Here, different techniques are available to find an optimal position of the internal knots $m_{interal} = (n+p+1) - 2(p+1) = n - p - 1$, for instance, an *equidistant distribution*, the *chord length method*, and the *centripetal method*. The latter two provide reasonable distributions for the knots, also if the interpolation points are unevenly distributed. For this, the *Euclidean distances* between the interpolation points are computed as

$$d_k = \parallel \boldsymbol{Q}_{k+1} - \boldsymbol{Q}_k \parallel_2, \quad k \in \{1, 2, ..., n-1\}. \tag{6.26}$$

For the chord length method, the overall length is computed as

$$d = \sum_{k=1}^{n-1} d_k\,. \tag{6.27}$$

From this the parameter positions $\bar{\xi}_k$ of the interpolation points can be determined

$$\bar{\xi}_k = \bar{\xi}_{k-1} + \frac{d_k}{d} \quad \text{with } \bar{\xi}_1 = 0\,, \tag{6.28}$$

For the centripetal method this reads

$$d = \sum_{k=1}^{n-1} \sqrt{d_k}\,, \tag{6.29}$$

$$\bar{\xi}_k = \bar{\xi}_{k-1} + \frac{\sqrt{d_k}}{d} \quad \text{with } \bar{\xi}_1 = 0\,. \tag{6.30}$$

The position of the knots are then determined by averaging the parameter positions of the interpolation points with respect to the polynomial degree. Considering an open knot-vector the inner knots are computed as

$$\xi_{p+1+i} = \frac{1}{p}\sum_{j=1}^{p} \bar{\xi}_{i+j}, \quad i \in \{1, 2, ..., m_{internal}\}. \tag{6.31}$$

Finally, the locations of the control points are determined. For this, the shape functions are evaluated at the parameter positions $\bar{\xi}_k$ of the interpolation points, yielding a linear system of equations

$$\begin{bmatrix} N_{1,p}(\bar{\xi}_1) & ... & N_{n,p}(\bar{\xi}_1) \\ \vdots & \ddots & \vdots \\ N_{1,p}(\bar{\xi}_n) & ... & N_{n,p}(\bar{\xi}_n) \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_1 \\ \vdots \\ \boldsymbol{P}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{Q}_1 \\ \vdots \\ \boldsymbol{Q}_n \end{bmatrix} \tag{6.32}$$

This LSE needs to be solved for each component of $\boldsymbol{P}$, yet, one should note that only the right hand-side changes. Since the curve is interpolating and an open knot-vector is considered, the first and the last control point equal the respective interpolation points $\boldsymbol{P}_1 = \boldsymbol{Q}_1$ and $\boldsymbol{P}_n = \boldsymbol{Q}_n$.

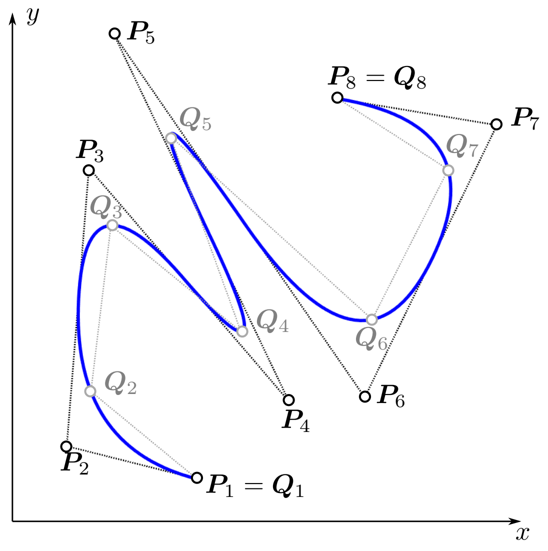


Figure 6.7: Spline interpolation with a B-spline curve of polynomial degree $p = 2$.

6.8.2 Approximation

Similar to the interpolation, the spline parameters p, $\xi_i \in \boldsymbol{\Xi}$ and $\boldsymbol{P}_i$ need to be found. For the approximation – apart from the polynomial degree p – also the number of control points n and the knot-vector $\boldsymbol{\xi}$ are independent and, thus, need to be chosen beforehand. For a uniform distribution of the n_s approximation or sampling points $\boldsymbol{Q}_j$, an open knot-vector with equidistantly distributed internal knots yields reasonable results $\xi_{i+1} - \xi_i = const$ for $i \in [p+1, m-p-1]$. However, for non-uniformly distributed sample points the chord length method, or the centripetal method can be applied. Again, a linear system of equations is found

$$\boldsymbol{A}\boldsymbol{P} = \boldsymbol{b}, \tag{6.33}$$

where $\boldsymbol{b} = \boldsymbol{Q}$ and $\boldsymbol{A}$ is a non-quadratic $n_s \times n$ matrix, containing the shape functions which need to be evaluated at n_s sampling parameters $\overline{\xi}_j \in [\xi_1, \xi_{end}]$. Thereby, each sample point $\boldsymbol{Q}_i$ corresponds to the parameter position $\overline{\xi}_i$.

$$\boldsymbol{A} = \begin{bmatrix} N_{1,p}(\overline{\xi}_1) & \dots & N_{n,p}(\overline{\xi}_1) \\ \vdots & \ddots & \vdots \\ N_{1,p}(\overline{\xi}_j) & \dots & N_{n,p}(\overline{\xi}_j) \\ \vdots & \ddots & \vdots \\ N_{1,p}(\overline{\xi}_{n_s}) & \dots & N_{n,p}(\overline{\xi}_{n_s}) \end{bmatrix} \tag{6.34}$$

Since, in general the number of sampling points is larger than the number of control points $n_s > n$, the resulting LSE is non-symmetric and over-determined. An approximative solution is found with the least squares approach, which minimizes the residuals r_j

$$\min_{\boldsymbol{P}} \sum_{j=1}^{n_s} r_j^2 = \min_{\boldsymbol{P}} \sum_{j=1}^{n_s} \left(\boldsymbol{V}\left(\overline{\xi}_j, \boldsymbol{P}\right) - \boldsymbol{Q}_j \right)^2 = \min_{\boldsymbol{P}} \|\boldsymbol{V}\left(\overline{\boldsymbol{\xi}}, \boldsymbol{P}\right) - \boldsymbol{Q}\|_2^2 \,. \tag{6.35}$$

In the matrix form, this corresponds to solving the following LSE

$$\boldsymbol{A}^T \boldsymbol{A} \boldsymbol{P} = \boldsymbol{A}^T \boldsymbol{b} \,. \tag{6.36}$$

Note, this approximation does not interpolate the first and last sampling point, which is often required. A simple replacement of these control points with the first and last interpolation points can lead to a messy approximation.

An interpolation can be better achieved by modifying the linear system of equations. For this, the terms regarding the first and last control point are brought to the right hand-side, yielding a modified reduced LSE

$$\hat{\boldsymbol{A}} \hat{\boldsymbol{P}} = \hat{\boldsymbol{b}} \,, \tag{6.37}$$

where $\hat{\boldsymbol{A}}$ is the non-symmetric, reduced $n_s \times (n-2)$ matrix

$$\hat{\boldsymbol{A}} = \begin{bmatrix} N_{2,p}(\overline{\xi}_1) & \dots & N_{n-1,p}(\overline{\xi}_1) \\ \vdots & \ddots & \vdots \\ N_{2,p}(\overline{\xi}_{n_s}) & \dots & N_{n-1,p}(\overline{\xi}_{n_s}) \end{bmatrix} , \tag{6.38}$$

$\hat{\boldsymbol{b}}$ the modified right hand-side vector

$$\hat{\boldsymbol{b}} = \begin{bmatrix} \boldsymbol{Q}_1 - N_{1,p}(\overline{\xi}_1)\boldsymbol{Q}_1 - N_{n,p}(\overline{\xi}_1)\boldsymbol{Q}_{n_s} \\ \vdots \\ \boldsymbol{Q}_{n_s} - N_{1,p}(\overline{\xi}_{n_s})\boldsymbol{Q}_1 - N_{n,p}(\overline{\xi}_{n_s})\boldsymbol{Q}_{n_s} \end{bmatrix} , \tag{6.39}$$

and $\hat{\boldsymbol{P}}$ the intermediate control points $\hat{\boldsymbol{P}}^T = [\boldsymbol{P}_2, ..., \boldsymbol{P}_{n-1}]$. An approximate solution is found again with the least squares approach (see Figure 6.8). Similar to the spline interpolation, the LSE need to be solved for each component of $\boldsymbol{Q}_i$. To avoid a singular matrix, the sample parameters need to be chosen definitely inside the knot-vector $\overline{\xi}_j \in]\xi_1, \xi_{end}[$.

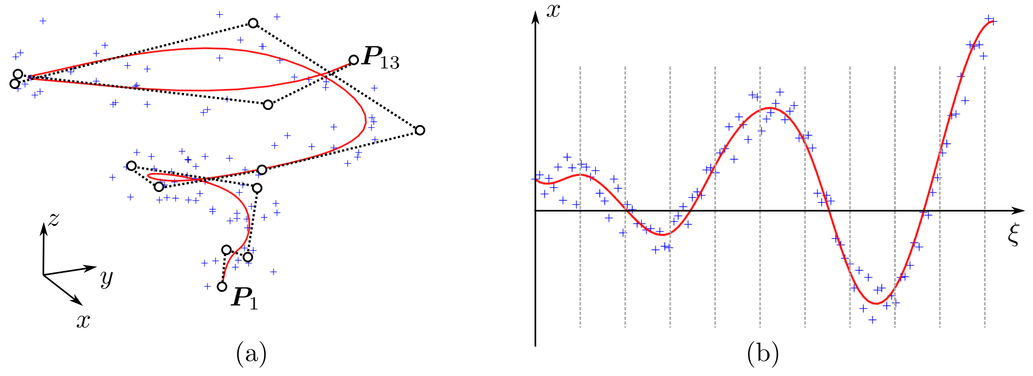


Figure 6.8: Spline approximation with 13 control points and a polynomial degree of $p = 3$. To obtain the approximation points $\boldsymbol{Q}_i$ a widening spiral was sampled at 100 points and an additional random deviation was added to each point. This produces a noisy point-cloud (indicated with blue '+'). (a) Resulting three-dimensional spline approximation curve and (b) approximation of the $x-$component. The vertical dashed lines indicate the equidistant knot positions.

Chapter 7

Geometric Representations

Computer-aided design uses several different representation schemes for the modeling and representation of three-dimensional geometric objects. The two most common representation forms are (i) *boundary representation* (B-rep), and (ii) parametric, procedural modeling with solid primitives, following the *constructive solid geometry* (CSG) idea. However, especially with the emerging of *additive manufacturing* (AM) as a novel production technique, it has become vital to develop new schemes that are better suited to model the interior of the volume, such as the *volumetric representation* (V-rep). Next, different CAD representation schemes will be outlined. For the sake of completeness, also non-CAD models are briefly introduced. The descriptions of geometric representations are based on the publications [13] for procedural and CSG models, [14] for B-reps, and [15] for V-reps.

7.1 Boundary Representation

Boundary representation (B-rep) describes a geometrical object by its boundaries [74], i.e., a volume is implicitly defined by the surfaces that form the outer hull. In fact, a valid B-rep model consists only of the outer hull. A model Ω can consist of several sub-domains, which all describe separate closed volumetric bodies B_i.

$$\Omega = \{\, B_i \;\mid\; i \in \{1, ..., n\} \,\} \tag{7.1}$$

with n being the number of volumetric bodies. For simplicity, only B-rep models with one domain $\Omega = B$ are considered. A B-rep body consists of a topology T which is explained in Section 7.1.1 and a geometry G, described in Section 7.1.2 [3]

$$B\,(\,T, G\,)\,. \tag{7.2}$$

The topology T describes all entities' relations or logical locations, whereas the geometry G provides the physical location of points and the description of the curves and surfaces. Consequently, the geometry defines the actual shape of the model.

B-rep offers several desirable properties, such as direct access to surfaces and great freedom in the design, as all kinds of free-form shapes are possible. Figure 7.2 shows exemplarily a complex B-rep model with many trimmed freeform spline surfaces[1]. On the other hand, B-rep also poses several difficulties, especially regarding a subsequent numerical simulation, since B-rep models are not mathematically,

[1]The model was submitted in the scope of a CAD challenge form General Electric® [75] for the design of a jet engine bracket.

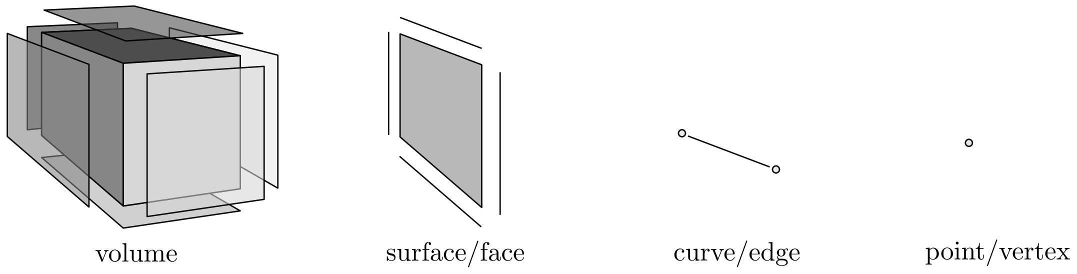


Figure 7.1: Structure of a simple, classical B-rep model. The entities can be described by their subordinated entities, i.e. a volume consists of faces, which consist of edges and finally of nodes.

inherently valid. For example, watertightness can not be guaranteed and, thus, a point membership classification may fail, as inside and outside cannot be unambiguously distinguished. In this context, various flaws in the topology or geometry, but also inaccuracies, can render a numerical simulation difficult or even impossible. Such flawed B-rep models are strictly spoken in a mathematical sense not inherently valid and are treated in detail in Section 8.3.

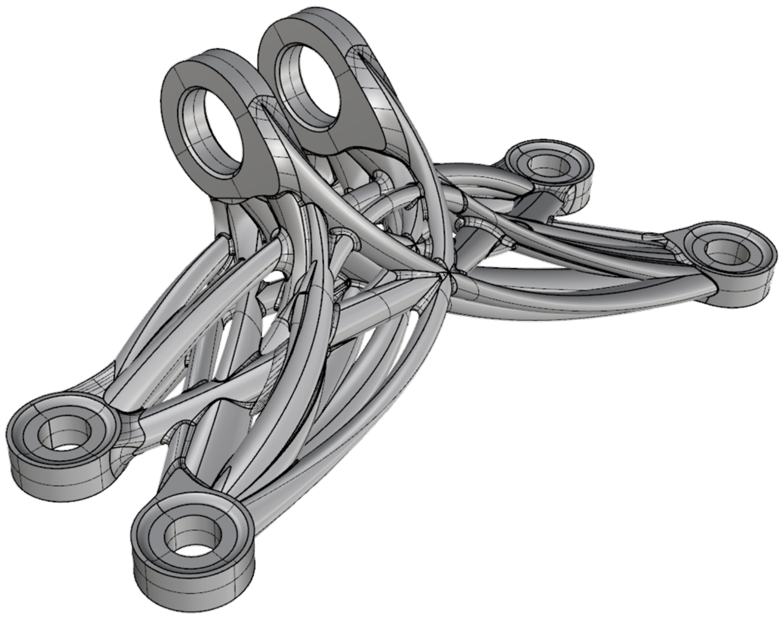


Figure 7.2: Complex free form B-rep model consisting of 631 trimmed spline surfaces and 3 506 boundary curves.

7.1.1 Topology

The topology $T(\,t, r^{int}, r^{ext}\,)$ provides the logical relations between the topological entities $t = \{t_i\}$. Topological entities are vertices v_i, edges e_j, and faces f_k. The relations can be sub-categorized into internal $r^{int} = \{r_i^{int}\}$ and external relations $r^{ext} = \{r_i^{ext}\}$. Internal relations r_i^{int} store vertical adjacency information and, thus, define how a topological entity $t = \{t_i\}$ is composed and which are the respective underlying topological entities. The topological entities are typically represented as sets, with the internal relations as side conditions

$$V = \{\, v_i \;\mid\; i \in \{1,...,n\}\,\}\,, \tag{7.3}$$

$$E = \{\, e_j \;\mid\; j \in \{1,...,m\}\,,\, e_k = (v_\alpha, v_\beta)\,,\, v_\alpha, v_\beta \in V\,\}\,, \tag{7.4}$$

$$F = \{\, f_k \;\mid\; k \in \{1,...,o\}\,,\, f_k = \big((e_\kappa)_{\kappa\in\{\alpha,...,\psi\}}\,, \boldsymbol{n}_i\big)\,,\, e_\kappa \in E\,\}\,, \tag{7.5}$$

where n, m, o are the number of the vertices v_i, edges e_j, and faces f_k, respectively. The internal relation of an edge is the ordered pair of bounding vertices (v_α, v_β). A face f_k can have – apart from the boundary edges – its orientation as an additional internal relation. This is represented by an ordered pair containing: (i) an ordered n-tuple, which contains the n boundary edges (e_κ), and (ii) the respective normal vector $\boldsymbol{n}_i$. Note, the orientation can also be provided implicitly by the order of the boundary edges (e_κ).

The external relations r^{ext} describe the global, horizontal relations between the topological entities, i.e., the adjacency information. Various representation methods are available to express the internal and external adjacency relations (please refer to [3] for an overview). Many of these representations, e.g., the *winged edge model*, or the *double connected edge list*, are combining internal and external relations.

Graphs – for instance represented by adjacency matrices – offer a well-suited possibility to represent and visualize the adjacency relations. As an example, consider a part of a triangular mesh with three faces as depicted in Figure 7.3. Different adjacency matrices are possible: in Equation (7.6) only the external relations between the faces are represented in r^{ext}_{FF}. The adjacency matrices for faces and edges r_{FE} (see Equation (7.7)) and for edges and vertices r_{VE} (see Equation (7.8)) store vertical information, which relates to the internal relations. However, the adjacency matrices are defined over the entire topology, thus, horizontal information is contained as well. Consequently, they combine internal and external relations[2].

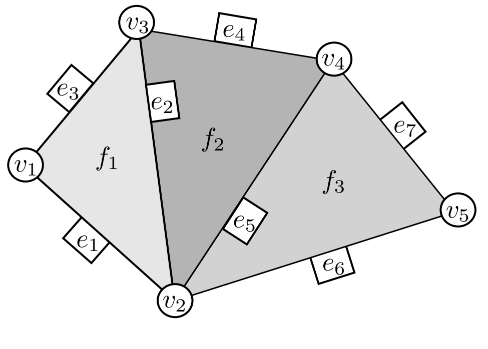


Figure 7.3: Example topology with vertices ($n = 5$), edges ($m = 7$), and faces ($o = 3$).

$$r^{ext}_{FF} \subset F \times F = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{7.6}$$

$$r_{FE} \subset F \times E = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{7.7}$$

$$r_{VE} \subset V \times E = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{7.8}$$

7.1.2 Geometry

The geometry $G(\{g_i\})$ consists of the geometric entities g_i, i.e., the points $\boldsymbol{P}_i$, curves $\boldsymbol{C}_j(\xi)$, and surfaces $\boldsymbol{S}_k(\xi, \eta)$. The geometry describes the actual physical location of the points and the shape of the boundary curves and surfaces.

In the easiest case, the geometry is tied closely to the topology, allowing only straight edges and plane triangles. Hence, the shape is determined solely by the location of the corner points and the

[2] *Note:* An entry $r_{ij} = 1$ in the adjacency matrix indicates a relation between entity i and entity j. If no relation exists $r_{ij} = 0$.

respective relations for edges and triangles. Nowadays, the geometry is overwhelmingly defined with splines, as illustrated in Chapter 6. Yet, also non-spline geometries are typically expressed in parametric representation (using shape functions), to be independent from the position and orientation of the model.

$$\boldsymbol{P}_i = [x_i,\, y_i,\, z_i]^{\mathrm{T}} \tag{7.9}$$

$$\boldsymbol{C}_i(\xi) = \begin{bmatrix} x(\,\xi\,) \\ y(\,\xi\,) \\ z(\,\xi\,) \end{bmatrix} \quad e.g., \quad \boldsymbol{C}_i(\xi) = \sum_j^{n_{Q_j}} N_j(\,\xi\,) \cdot \boldsymbol{Q}_j \tag{7.10}$$

$$\boldsymbol{S}_i(\boldsymbol{\xi}) = \begin{bmatrix} x(\,\boldsymbol{\xi}\,) \\ y(\,\boldsymbol{\xi}\,) \\ z(\,\boldsymbol{\xi}\,) \end{bmatrix} \quad e.g., \quad \boldsymbol{S}_i(\boldsymbol{\xi}) = \sum_j^{n_{Q_j}} \sum_k^{n_{Q_k}} N_j(\,\xi\,) \cdot N_k(\,\eta\,) \cdot \boldsymbol{Q}_{j,k} \tag{7.11}$$

with $\xi \in \mathbb{R}$ and $\boldsymbol{\xi} = (\xi, \eta) \in \mathbb{R}^2$. $N_i(\xi)$ are the shape functions – such as Lagrange or Legendre polynomials, B-splines, NURBS, etc. – and $\boldsymbol{Q}_i$ the respective interpolation, or control points. Depending on the curve, and surface description, these points $\boldsymbol{Q}_i$ may completely, or partially coincide with the geometrical points $\boldsymbol{P}_i$. Similar to the topology, the geometry G can be represented by sets

$$P = \{\, \boldsymbol{P}_i \;\mid\; i = \{1, ..., n\}\,\}, \tag{7.12}$$

$$C = \{\, \boldsymbol{C}_i \;\mid\; i = \{1, ..., 2m\}\,\}, \tag{7.13}$$

$$S = \{\, \boldsymbol{S}_i \;\mid\; i = \{1, ..., o\}\,\}, \tag{7.14}$$

where the number of points and surfaces must equal the number of vertices n and faces o, respectively. The number of geometrical curves m_{curves}, however, is twice the number of topological edges $m_{curves} = 2m$, since at each edge two surfaces/faces meet and each surface has its own boundary curve description (see Figure 7.4).

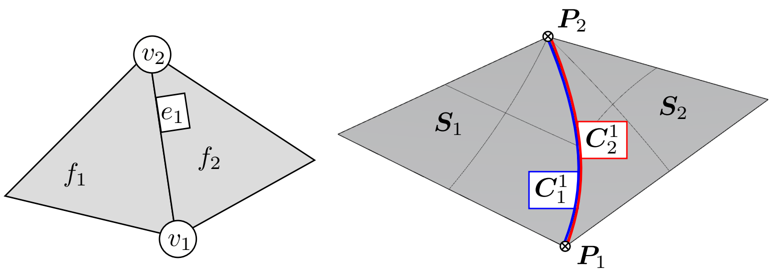


Figure 7.4: At each edge e_i two boundary curves $\boldsymbol{C}_{S_j}^{e_i}$ and $\boldsymbol{C}_{S_k}^{e_i}$ exist.

7.2 Solid-based modeling

In contrast to B-rep, solid-based procedural modeling describes an object's volume explicitly, whereas the boundary or outer hull is defined implicitly. The basic idea of procedural modeling is to represent the entire modeling process in chronological order, instead of merely storing the geometric model. The actual geometry corresponds to the resulting model at the end of the construction chain (see Figure 7.5a). Procedural modeling allows a smooth introduction of parametric dependencies and facilitates a subsequent modification of the model. The primary individual steps in the procedural chain are (i) the generation of three-dimensional solids, (ii) the adjustment of a solid, or (iii) the combination of multiple solids with specialized operators. The underlying concept is constructive solid geometry [76], which was widely extended to create a large variety of forms, especially for applications in mechanical engineering.

7.2.1 Constructive solid geometry

In CSG, a three-dimensional object consists of a set of simple primitives, such as *cubes*, *cylinders*, *cones*, *spheres*, *tori*, etc. These primitives are then combined with the three basic binary *Boolean* operations: *union* $\cup$, *intersection* $\cap$, and *difference* $\setminus$. A binary operation takes two input entities and combines them into one output entity, which can again serve as an input entity for another Boolean operation. With the *negation* $\neg$, also, a unary operation is imaginable, yet, in practice, the negation is hardly applicable, as it easily leads to infinite domains.

The resulting CSG object is stored implicitly in a CSG tree, as illustrated in Figure 7.5b. A procedural model's construction history can be regarded as a maximal unbalanced CSG tree, i.e., a CSG tree with one 'branch' that is of maximal depth. If parts are independent of each other, it is possible to rearrange, or even balance the tree. However, in general, a CSG tree is non-commutative, as the order in which the primitives are created, and the operations are applied is not interchangeable.

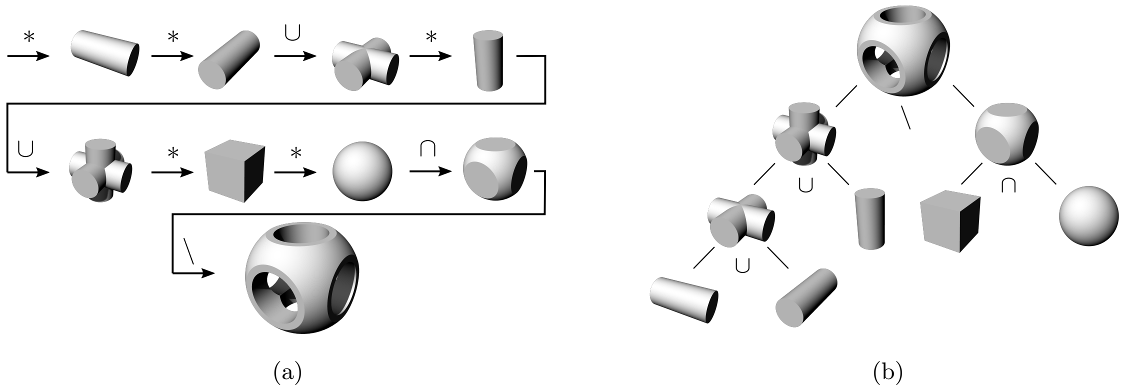


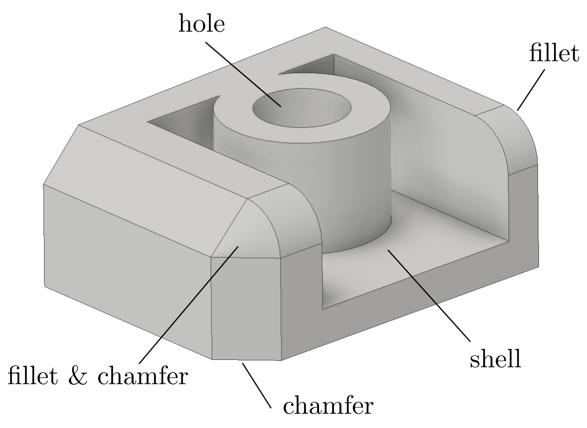
Figure 7.5: Constructive solid object creation with the simple primitives cylinder, cuboid, and sphere and the Boolean operations: union $\cup$, intersection $\cap$, and difference $\setminus$. The construction can be represented either by (a) a construction history of a procedural model (an asterisk * indicates the creation of a new object), or (b) a CSG tree, where the leaves correspond to the creation of a new primitive.

7.2.2 Extended primitives and operations

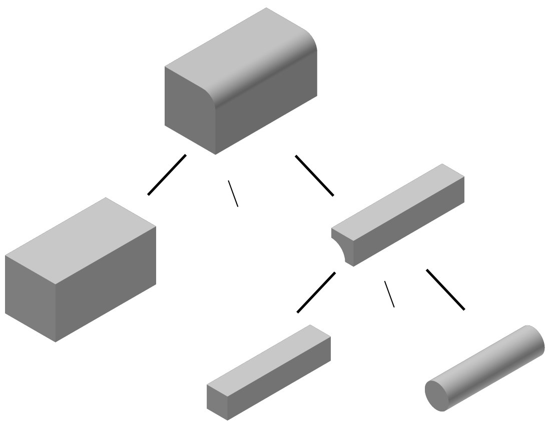
Since, with CSG, the amount of possible shape is very limited – due to the few simple primitives – procedural modeling CAD tools were extended with more complex primitives. Additionally, they provide a richer set of operations. In the following, these more elaborate primitives and operations are referred to as extended operations and extended primitives. Examples for extended operations are *chamfer*, *fillet*, *drilling a hole*, or *shell* as depicted in Figure 7.6a. A closer look reveals that most of these operations can be represented by a sequence of the original Boolean operations and additional primitives, which are summarized for convenience[3] (see Figure 7.6b). Unlike the Boolean operations, the extended operations are mostly unary.

In contrast to the extended operations, the extended primitives such as *extrusions*, *sweeps*, *lofts*, and *solids of revolution* can be regarded as a true extension to the CSG primitive set (see Figure 7.7). All extended primitives are constructed very similarly. A two-dimensional closed contour line is drawn on a construction plane (e.g., the $xy-$plane, or a plane in the normal direction of a curve). Then, the

[3]Critical in this context is the shell operation, as it is defined with the distance to the boundaries. Thus, it is not a solid-based operation. For simple models, it might be possible to express the shell operation as a sequence of primitives and operations, yet, the process is hardly automatable and not applicable for complex geometries.



(a)



(b)

Figure 7.6: Extended operations: (a) overview over some possible operations provided in Autodesk Inventor®, and (b) the *fillet* operation expressed with the classical Boolean operations.

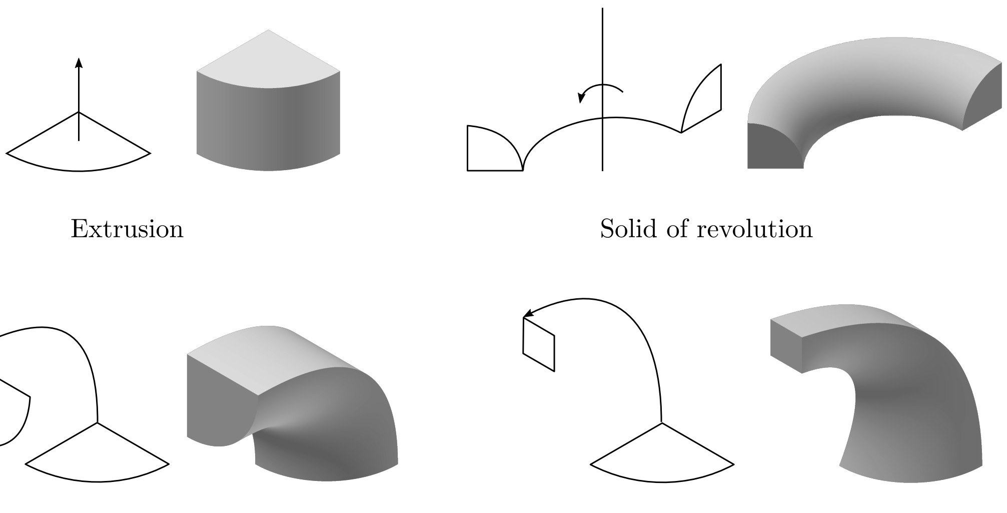


Figure 7.7: Extended primitives: extrusion, solid of revolution, sweep, and loft.

sketch is extruded along a sweep path. Depending on the path's shape, either an extrusion, a solid of revolution, or a sweep is created. Only the construction of a loft is slightly different. Here, the initial sketch is not just extruded but blended along the path into the ending sketch[4]. Note, that a loft can pass through/interpolate several intermediate sketches. However, such a loft can simply be divided at the intermediate sketches, yielding a chain of elementary lofts.

[4]The definition of the sweep and loft is inconsistent between different CAD tools. In the following, the terms are used as herein introduced.

7.3 Volumetric representation

An alternative geometrical modeling technique is the volumetric representation (V-rep), introduced by Massarwi and Elber [6]. The basic idea is to use trimmed, volumetric, trivariate B-splines that explicitly model the interior volume. The motivation for this elaborate object description lies in the necessity to represent functionally graded materials for additive manufacturing directly (see Chapter 9). Like procedural modeling, the V-rep framework follows the CSG idea, i.e., solid primitives are combined with the Boolean operations. However, in contrast to classical procedural modeling CAD tools, the V-rep framework provides a fully volumetric description for the primitives and the resulting objects from the Boolean operations.

The V-rep framework is implemented in the open-access geometry library *Irit* [77]. Irit is not limited to V-reps but provides a large amount of different geometric modeling and analysis functionalities. Apart from the direct access via the C++ interface, or the Irit scripting language, the graphical CAD environment *GuIrit* [73] is available.

7.3.1 V-rep primitives

The basic building block of the V-rep framework is a volumetric, tri-variate spline, according to Equation (6.12). Irit supports Bézier, B-splines, and NURBS. From a topological point of view, this basic trivariate corresponds to a cuboid, whose geometry depends on the control points' location. V-rep allows only regular splines, which means, no self-intersections, or vanishing or negative Jacobian occur.

Apart from the trivial case of a cuboid, the V-rep framework offers a variety of both: high-level and simple primitives. Similar to procedural modeling, constructors for extrusions, solids of revolution, sweeps, and lofts are available. Since V-rep is not limited to two-dimensional input sketches, two additional constructors are provided that use curved surfaces as input parameters: (i) the volume of a *ruled solid* is defined as a interpolation between two potentially curved surfaces, and (ii) with the *Boolean sum* a volume is created from six (curved) boundary surfaces [78]. Figure 7.8 illustrates the constructors, which all yield one single trivariate spline.

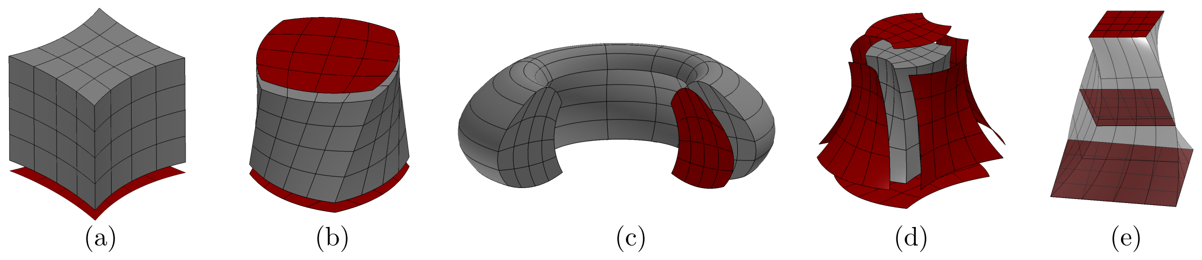


Figure 7.8: V-rep constructors: (a) extrusion, (b) ruled solid, (c) solid of revolution, (d) boolean sum, and (e) loft.

Some simple primitives – such as spheres, cylinders, tori, and cones – introduce singularities when represented by a single trivariate spline. As an example, consider the mid axis of a sphere, where the Jacobi matrix vanishes $\det(\boldsymbol{J_V}(r = 0)) = 0$. Singularities can render a numerical simulation impossible, especially if the geometry description's shape functions are the same as for the FEM formulation. The V-rep framework provides special constructors to avoid these singularities. These constructors compose singular primitives from several non-singular trivariate elements, as illustrated in Figure 7.9.

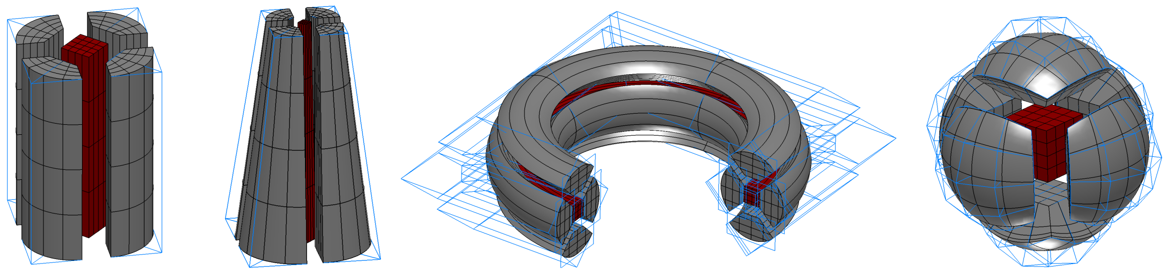

(a) (b) (c) (d)

Figure 7.9: Composed primitives to avoid singularities: (a) A cylinder is composed of five extruded solids, whereas (b) a cone is composed of five ruled solids. (c) A torus is constructed using five solids of revolution, and (d) a sphere is composed of six ruled solids and one cuboid in its center.

7.3.2 V-model construction

Since a CAD model is only in the rarest of cases representable solely with primitives, the V-rep framework provides also geometric operations, such as the Boolean operators. Applying these operations yield trimmed 3-manifold trivariate splines, called *V-cells* ν_C^i. V-cells are individual, non-intersecting entities $\nu_C^i \cap \nu_C^j = \emptyset \ \ ,\forall i \neq j$. They store additional adjacency information, which allows efficient data queries. The union of all V-cells is called *V-model* $V_m = \bigcup \nu_C^i$. In the case of a union, or intersection of two, or more V-cells, it is necessary to create new V-cells in the overlapping regions. For this, the Boolean constructor is used to create new trivariate splines from the trimming surfaces (see Figure 7.10). Due to the spline-

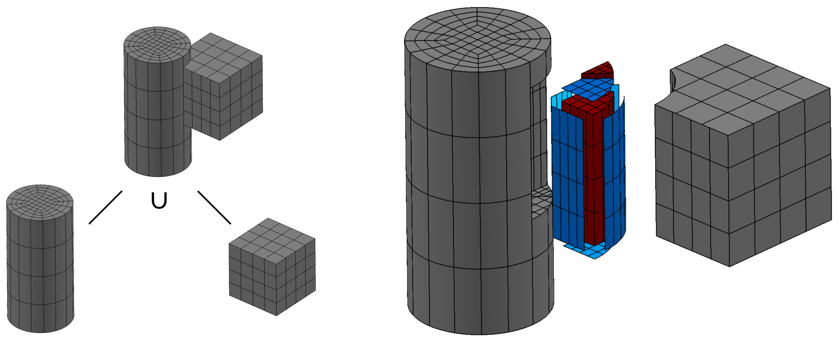


Figure 7.10: V-Model created as the union of a trivariate cuboid and a trivariate, non-singular cylinder. The intersected volume yields two V-cells (marked in red) which are constructed with the trimming surfaces (highlighted in blue).

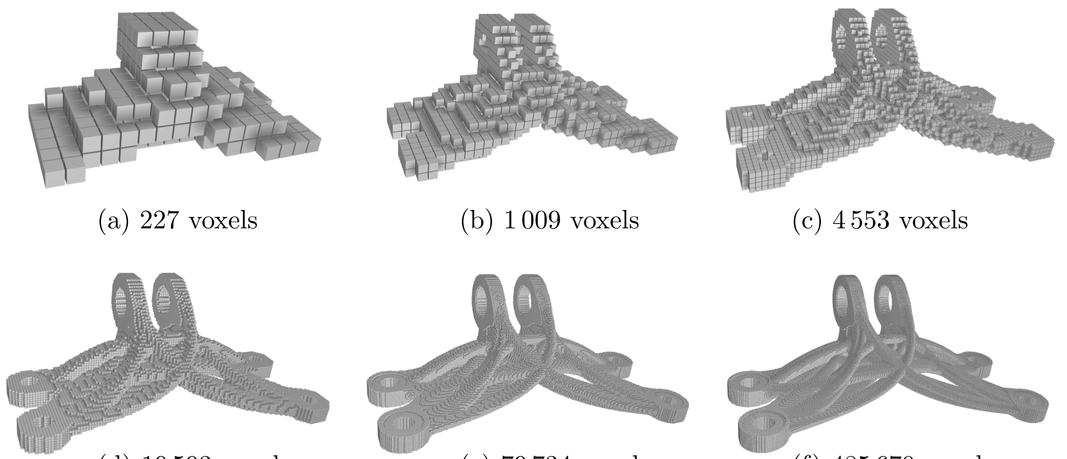
based description, the isogeometric analysis seems to be the obvious choice for numerical simulations based on V-models. However, considering the overlap of primitives, two significant obstacles render the use of the IGA complex. Firstly, intersected primitives are trimmed. Secondly, the parametrizations – i.e., knot vectors, control point meshes, and polynomial degrees – of the resulting 'intersection' V-cells do not coincide with their parent primitives. The latter requires special methods to 'glue' the primitives together, such as Mortar methods, or T-splines [79, 80]. By contrast, embedded domain methods require no special treatment of overlapping regions and pose far fewer requirements on the underlying geometric model.

7.4 Non-CAD model representations

A CAD model cannot always be generated directly. In particular, this is the case, if the respective object already exists, or does not result from a design process – for example – (ancient) constructions or natural objects. For this, mainly two different imaging techniques are available: (i) *spatial decomposition*, typically voxel domains, and (ii) *point clouds*. If needed, the resulting representations can subsequently be transferred into CAD models. As these representation forms allow a direct simulation with the FCM, the underlying concepts are briefly discussed.

7.4.1 Spatial decomposition

For small to medium-sized objects, various tomography procedures offer the possibility to gain an approximative representation of the volume with its internal structures. For this, the specimen is scanned slice by slice by any kind of penetrating wave, e.g., X-rays for CT scans, or magnetic fields and radio waves for MRI. Depending on the used wave type, different material properties can be detected, such as density or water content. The information is stored on small spatial units, so-called voxels. Of course, the quality of the approximation is highly dependent on the resolution of the imaging process. By implication, a decent approximation yields an enormous amount of data as illustrated, in Figure 7.11[5] for the B-rep model depicted in Figure 7.2.



(a) 227 voxels (b) 1 009 voxels (c) 4 553 voxels

(d) 19 593 voxels (e) 79 734 voxels (f) 485 670 voxels

Figure 7.11: Voxel reconstructions with different resolutions of a complex B-rep model. From (a) to (f) the number of voxels in each direction was doubled, starting with 16 voxels and ending with 512 voxels in $x-$direction.

Most spatial decomposition models are voxel domains, yet, they are not restricted to this. Any kind of representation subdividing a volume into non-overlapping cells is possible, such as tree structures like the octree, or $kd-$tree. Advantageous is that tree-like structures can represent the same shape as a voxel model and are, at the same time, considerably more memory efficient. However, voxels typically do not

[5]Created with the website: https://www.drububu.com/miscellaneous/voxelizer/

just represent a shape, but rather the material distribution. Thus, a transformation into space-trees is not always applicable/sensible.

7.4.2 Point clouds

Point clouds are often the only choice to represent large objects and structures that cannot be moved, and thus, need to be recorded in situ. As the name already suggests, point clouds are, in the simplest case, just (large) sets of points (see Figure 7.12). However – apart from the coordinates – usually additional data such as the outwards pointing normals and the color are stored. Several point cloud generation techniques are available, which yield results of varying precision. For instance, laser scanners provide very accurate point clouds, whereas point clouds extracted from pictures are often very noisy. Similar to

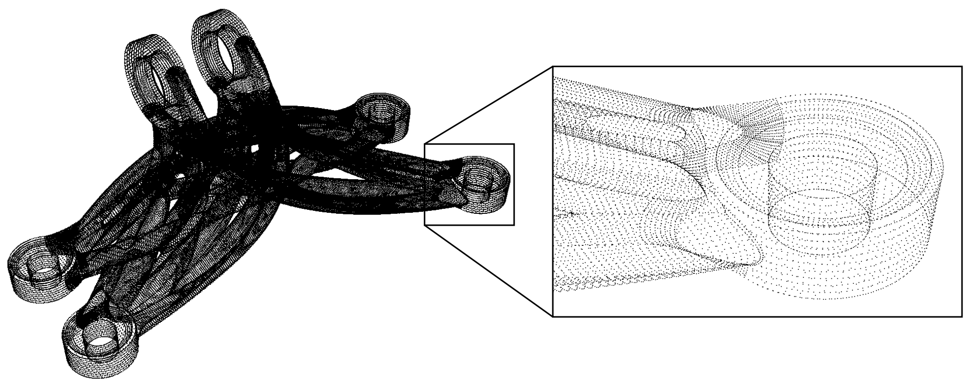


Figure 7.12: Point cloud extracted from the B-rep model depicted in Figure 7.2 with 1 012 635 points.

the voxel reconstructions, point clouds are very memory extensive and challenging to handle. However, since points close to each other often store redundant information, it is possible to reduce the number of points without losing crucial information.

7.5 Model conversion and data exchange

The development cycle of complex products typically involves many distinct designers. Different objectives, personal preferences, or affordability can lead to a large variety of choices for the respective CAD tools. Since commercial CAD systems use their proprietary formats, data exchange is a non-trivial and error-prone task. This is especially the case if features are exclusive of a particular CAD system. Even more complicated is the change between different kinds of geometric representations – e.g., from B-rep to procedural modeling or vice versa.

7.5.1 CAD exchange formats

Apart from the proprietary data formats, several neutral, open exchange formats exist. One widely implemented and commonly used format is *initial graphics exchange specification* (IGES). IGES was initiated by several industrial partners in the 1970s and proposed as an initial draft for a neutral exchange format. The format supports free-form B-rep modeling as well as solid modeling. With the launch of the ***st**andard for the **e**xchange of **p**roduct model data* (STEP – ISO 10303) format in 1994, further developments of IGES were declined. In contrast to IGES, the STEP format can describe product data throughout the entire life cycle. Also, ongoing continuous development allows the incorporation of new features, such as *T-splines*. STEP is organized in modules, depending on the respective field of application, e.g., mechanical engineering, aerospace industry, circuit design, etc. Even though, or probably because STEP provides such a tremendous amount of description possibilities, IGES is still extensively used.

Often not the complete, accurate model is needed, but an approximated surface description is sufficient. For this, a surface triangulation is convenient and widely supported, as depicted in Figure 7.13. The standard exchange format for triangular surfaces is *STL*. The name STL stems from its initial field of application ***st**ereo**l**ithography*, but stands unofficially also for the more expressive term *standard tessellation language*. STL can be interpreted as a minimal B-rep format, as it provides only the least amount of necessary information. Additionally, STL makes no explicit separation between topology and geometry. STL consists of independent triangles, which are defined by their three corner points. As geometric information in the form of point coordinates is provided only for vertices explicitly, curves and surfaces are linearly interpolated. No adjacency, or 'consistency' information is provided, which makes STL quite flexible, yet particularly prone to various potential flaws. The relation between faces and vertices reads:

$$F^{STL} = \{\, f_i \;\mid\; i \in \{1, ..., o\}\,,\; f_i = ((v_\alpha,\, v_\beta,\, v_\gamma),\, \boldsymbol{t}_i)\;,\; v_\kappa \in V\;,\; |V| = 3 \cdot o \,\}\,. \tag{7.15}$$

Note that – due to the multiple definitions of vertices – STL models are, strictly speaking, topologically not valid. Furthermore, the redundancy of point definitions and normal vectors that could be derived from the face's orientation has an imminent impact on the required memory.

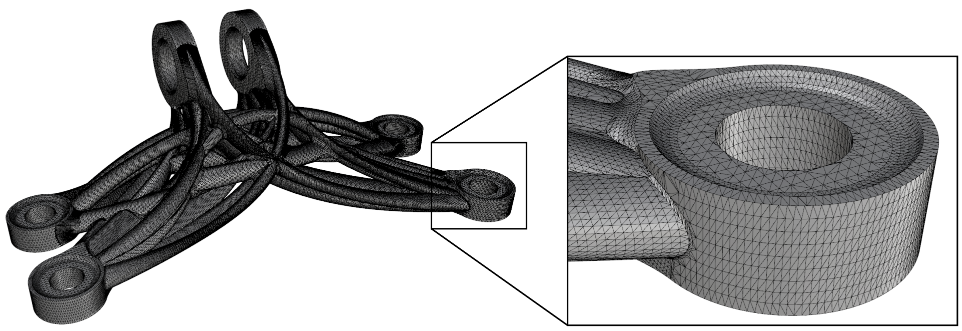


Figure 7.13: STL triangulation of the B-rep model depicted in Figure 7.2 with 337 545 individual triangles.

7.5.2 Conversion between representation forms

A conversion from a B-rep model to a procedural model is, in general, not possible since B-rep models contain no information about the construction process and only provide an explicit boundary description. Yet, even if the construction process was known, the conversion is, in general, not possible as solid-based procedural models do not cover the larger freedom in design (i.e., the number of possible shapes) of B-rep.

On the other hand, the conversion from a procedural model to B-rep is always possible. Fortunately, most procedural modeling CAD tools directly allow the export to a neutral B-rep format. In the worst case, it is still possible to derive an approximate surface description, for instance, with the *marching cubes algorithm* [81]. The marching cubes algorithm only needs a point membership classification from the model, which is easily feasible on the construction (CSG) tree.

Procedural models can easily be converted into volumetric representation since solid-based procedural modeling is a subset of V-rep. Furthermore, it is possible to migrate spline-based B-rep models to V-rep models [6]. A conversion from V-rep to B-rep is straightforward, considering only the respective volumetric splines' surfaces need to be extracted and exported.

Chapter 8

Flawed models

Flawed, or *dirty* models are models whose geometries, or topologies are mathematically invalid. The reasons for flaws can be manifold. B-rep is especially prone to a wide variety of possible flaws. Yet, also with solid based procedural modeling, it is possible to generate invalid topologies. Another aspect is the incompatibility or shortcomings of exchange formats. In the following, possible flaws for the respective representations are illustrated. Special focus is laid on flawed B-rep since, within this thesis, a method for the direct numerical simulation for these models is presented.

8.1 Flawed procedural models

Solid-based procedural models are comparably resistant towards most flaws. Nevertheless, invalid topologies can appear here as well. These flawed models are often referred to as *non-manifolds*. A non-manifold in this context can be regarded as a volumetric object that could not exist in reality. Figure 8.1 shows some possible flaws, which can be grouped roughly into two categories: (i) dangling surfaces and (ii) contact lines, or points. Dangling surfaces can occur due to an intersection, or difference operation, where two objects share one or more common boundary surfaces. As a result, only these boundary surfaces remain. Since such a (dangling) surface represents no volume, the orientation cannot be defined, and thus inside, or outside cannot be distinguished. Internal surfaces and non-closed boundaries belong to this category as well. The creation and occurrence of dangling surfaces are usually intercepted by the CAD tools, meaning, the respective operations will directly fail during the modeling process.

This is, however, not the case for flaws that belong to the second category. It is easily possible to create two solid objects that touch each other at one point, or have zero thickness along a contact line. Here, a point membership classification is ambiguous, which limits the applicability of certain CAD operations. For instance, a filleting operation can not be conducted on such a contact edge. Regarding a direct numerical simulation, contact points or lines are not critical as they can simply be assumed to be either inside or outside.

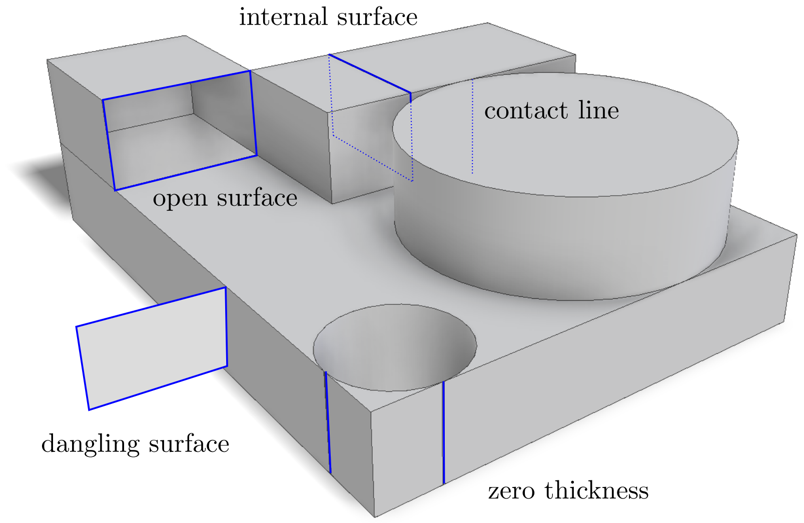


Figure 8.1: Examples of flaws on a procedural model that result in a non-manifold topology.

8.2 Data exchange issues

As described in Section 7.5.1, several neutral formats are available to facilitate the data exchange between different CAD software tools. It is, however, not guaranteed, that the model is transferred correctly. Reasons for flawed outcomes are implementation issues, or the representation possibilities of the neutral format do not reflect the proprietary data structure. It is not unlikely that even the export and re-import to IGES, or STEP with the same CAD software fails, as depicted in the Figures 8.2 for the comparably trivial example of an intersection of two spheres. A remedy might be the STL format, provided that a triangular approximation suffices the particular requirements.

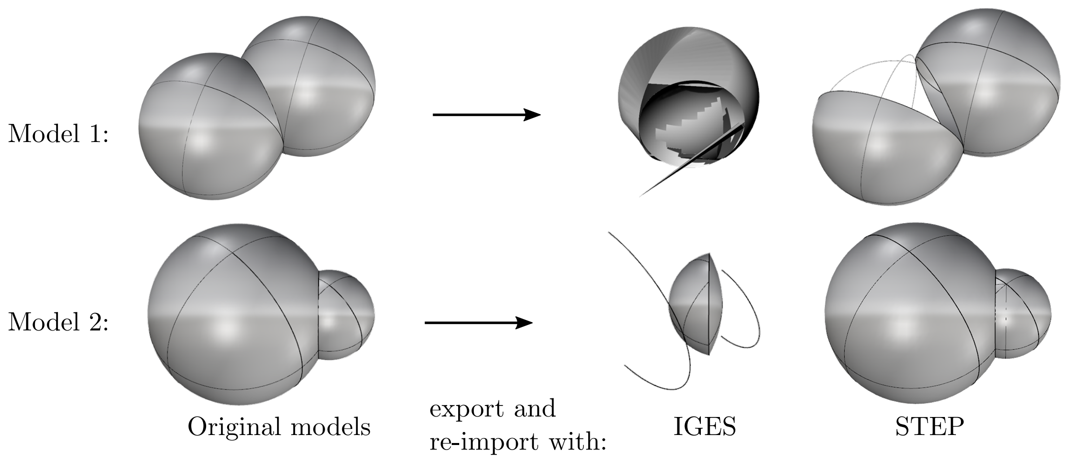


Figure 8.2: Two simple models that are each constructed by an intersection of two spheres. The export to IGES and STEP and re-import with Rhinoceros® easily yields invalid geometric models.

8.3 Flawed B-rep models

B-rep offers the greatest freedom in design. However, this freedom comes with a price. The implicit volume description and loose restrictions on the boundary connections, open the gate for all kinds of geometrical and topological flaws, as illustrated in the Figures 8.4, 8.5 and 8.6 [82, 83]. Due to this vulnerability, and since these flaws are typically tiny and do not affect the visualization, most non-trivial B-rep models are flawed to some extent. On the first glimpse, the B-rep model depicted in Figure 7.2 looks perfectly fine, yet, a magnification of some details reveals numerous severe issues. This section follows the publication [14].

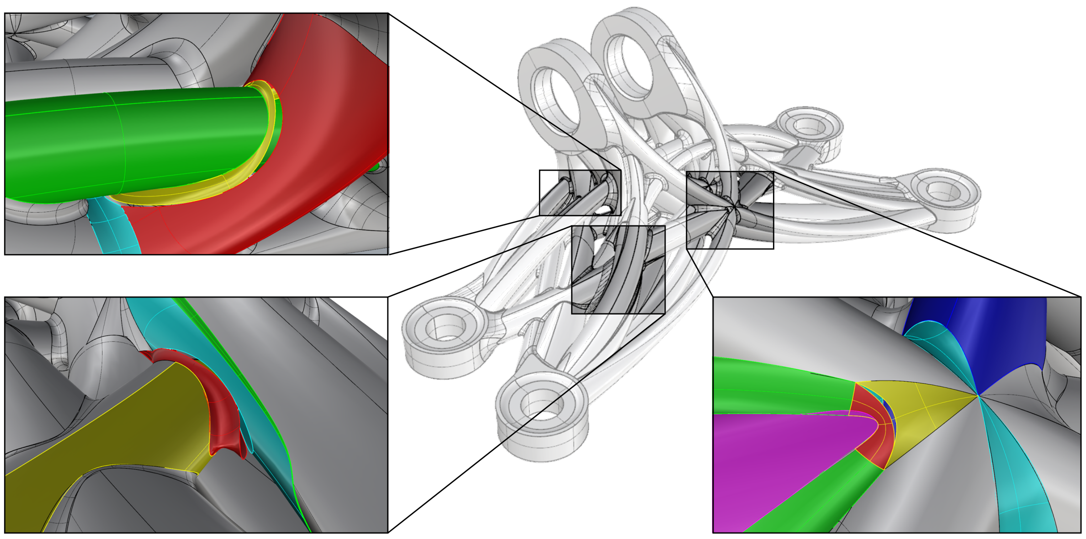


Figure 8.3: Complex B-rep model. A zoom into some details shows serious flaws.

Reasons for a flawed B-rep model can be data loss during data exchange (as explained in Section 8.2), inappropriate or inaccurate operations by the designer(s), or tolerances and precision limitations of the CAD software (see Section 6.5). As mentioned before, regarding a numerical simulation, these flaws are critical, as a mathematically invalid model can obstruct a meshing or a direct simulation. In the best case, they only generate excessively fine meshes in regions with no relevance to the succeeding analysis (e.g., at fillets, etc.) but increase the computational time unnecessarily.

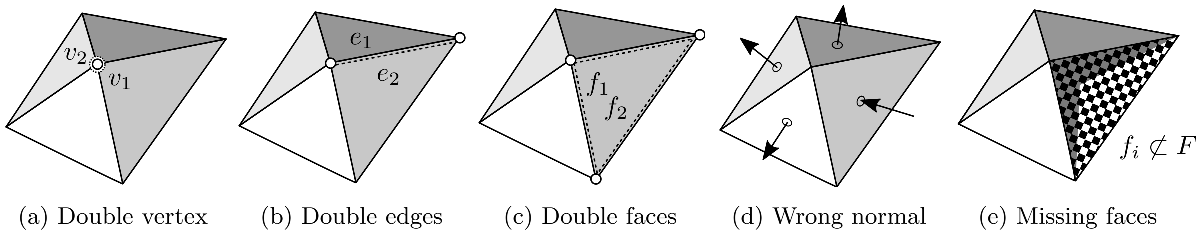
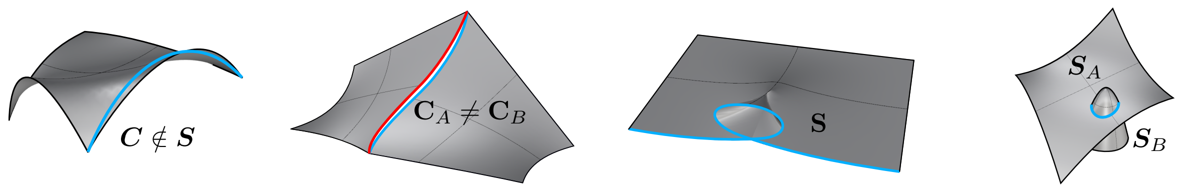


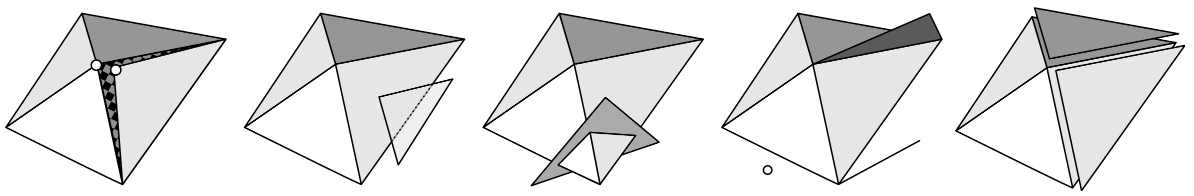
Figure 8.4: Topological flaws.

To perform a numerical simulation directly on flawed models with the FCM, it is necessary to categorize the flaws with respect to their impact on the point membership classification. To this end, mathematical operators are defined in Section 8.3.3. In Section 8.3.4 these operators are then applied to a valid B-rep


$\boldsymbol{C} \notin \boldsymbol{S}$
$\mathbf{C}_A \neq \mathbf{C}_B$
$\mathbf{S}$
$\boldsymbol{S}_A$
$\boldsymbol{S}_B$

(a) Boundary curve is not on the surface (b) Curves at common edge do not coincide (c) Surface and/or boundary curve self-intersect (d) Surfaces are intersecting

Figure 8.5: Geometrical flaws.



(a) Gaps (b) Overlaps (c) Intersections (d) Artifacts (e) Offsets

Figure 8.6: Hybrid flaws which consist of topological and geometrical components.

model. Thereby, a *valid* model is transferred into a *flawed* B-rep model. The flaw operators allow to introduce a control parameter ε that quantifies the characteristic size of the respective flaw. Consequently, ε limits the deviation from a *fictional*, flawless model. It is noteworthy that no explicit knowledge of the flawless model is required. Some flaws correspond to the application of one flaw operator, but most flaws result from a chain of subsequently executed flaw operations.

8.3.1 Conditions for validity

Although intuitively quite apparent, it is not trivial to provide a definition of a valid B-rep model. Patrikalakis et al. [84] stated:

"A B-rep model is valid, if its faces form an orientable 2-manifold without boundary."

Several requirements can be derived from this definition, which can be categorized into topological and geometrical conditions. Some of these requirements are also mentioned in [3, 85].

Topology

1. Different vertices do have different coordinates (see Figure 8.4a).

2. One edge is shared by exactly two faces (see Figures 8.6d and 8.7).

3. Faces at one vertex belong to one surface, i.e., at a vertex it is possible to cycle through all adjacent faces such that all of the vertex' edges are crossed exactly once (see Figure 8.7).

4. The orientation of faces must follow *Moebius' Rule*, i.e., inside and outside must be distinguishable from each other (see Figure 8.4d).

Geometry

5. A curve must lie on the respective surface whose partial boundary it forms (see Figure 8.5a).

6. Both boundary curves at one edge must coincide (see Figure 8.5b).

7. Surfaces must not self-intersect. From this – and from Condition 5. – follows that curves do not self-intersect either (see Figure 8.5c).

8. Surfaces must not touch or intersect with other surfaces except at common edges (see Figure 8.5d).

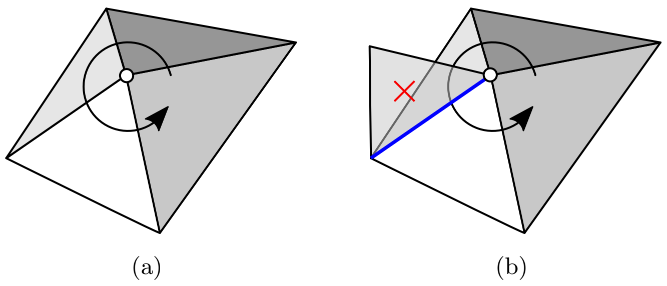


Figure 8.7: Vertex with adjoined edges and faces: (a) All faces belong to the same surface. Hence, it is possible to cycle through the faces, passing each adjoined edge once. (b) Not all faces belong to the same surface, i.e., when cycling through all faces, the highlighted edge needs to be passed twice.

8.3.2 Geometry healing

The most direct way to address CAD model flaws is to heal or repair the model before meshing, or a direct simulation. Geometry healing is a well-known and extensively researched topic. Basically, three different procedures are conceivable:

1. The identification of the location and type of the individual model errors and a subsequent correction,

2. global model reconstructions, or

3. hybrid methods that apply model reconstruction locally around the flaws.

A detailed literature study on different healing methods for the three procedures is provided in [14].

In summary, there exist numerous methods for geometry/topology healing. However, they all suffer from two main obstacles. Firstly, methods addressing local healing of individual flaws rely on a complete detection and correct classification of all modeling errors. To promote the individual approaches, often, an average percentage of detected and healed flaws is provided. By implication, this means that it can never be guaranteed that all defects are detected. Secondly, although it can be guaranteed to bring forth a valid model with global reconstruction methods, these methods can only approximate the original shape, and sharp features, such as edges, corners, or small details get easily lost. These shortcomings were the motivation for the hybrid methods, which only reconstruct locally and preserve small features. However, hybrid methods also rely on the correct and complete detection of all flaws.

Figure 8.8 shows a geometry healing for the flawed jet engine bracket depicted in Figure 8.3. As can be seen, some flaws are detected and healed reasonably. Others are identified and treated, yet, obviously

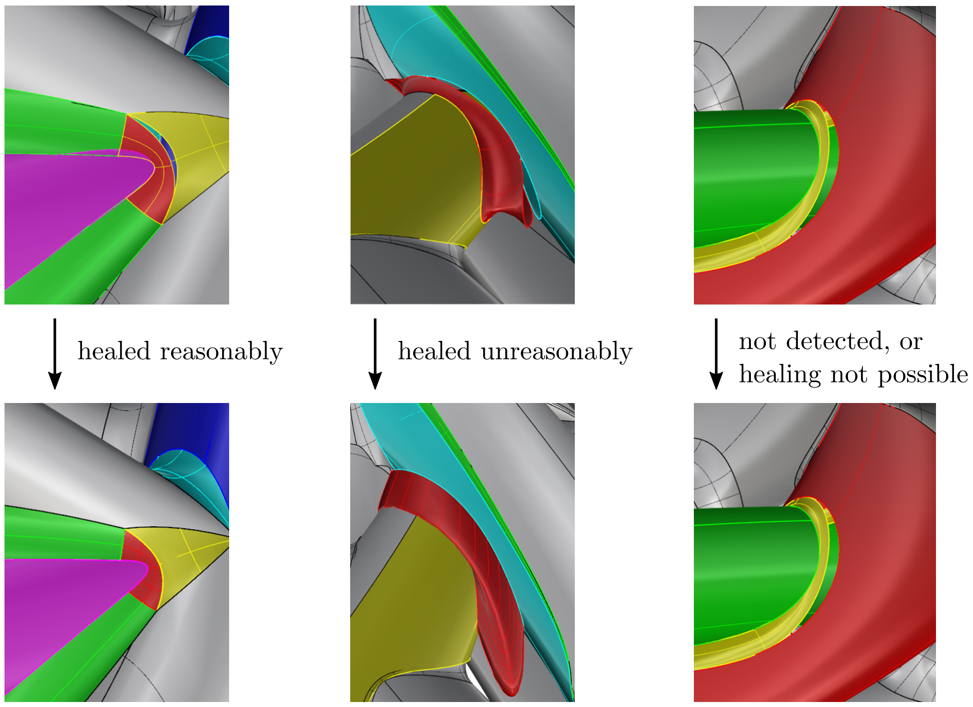


Figure 8.8: Automated healing of a flawed B-rep can have different outcomes. The flaw can be healed reasonably, unreasonably, or not at all, if it was either not detected, or healing was not applicable.

not as in the designer's intent. Finally, there exist also defects that were either not detected or could not be resolved.

Since a single flaw can already exclude the entire downstream simulation process, a considerable effort needs to be spent cleaning up a geometric model. This preparation often needs to be done by hand. An automatic model reparation can – as described – in general, not guarantee to deliver the required mathematically valid model, with all associated small features.

8.3.3 Flaw operators

In this section, several operators are introduced that perform transformations on a valid B-rep model, allowing for a controlled imposition of different flaws. For measuring the size of the defects, an error parameter ε is introduced. ε indicates the magnitude of the flaws and provides a measure for the inaccuracy of the model [14].

To provide an easily understandable formulation of the operators, consider an object-oriented B-rep data structure. Thereby, the implementation must allow a distinction between internal and external/adjacency relations. Figure 8.9 provides a *UML* diagram of a possible hierarchical implementation[1]. Here, all external adjacency relations r^{ext} are realized at the faces, where the adjacent faces are stored in the field: *adjacentFaces*. All other external adjacency relations (e.g., which edges are adjacent) can be derived from this and the respective internal relations r^{int}.

Let ω^{t_i} be the B-rep sub-part, or segment, which corresponds to a topological entity t_i, e.g., a face, an edge, or a vertex. The segment ω^{t_i} consists of all information that is needed to visualize t_i. Hence,

[1]For an introduction to the notation of the UML (Unified Modeling Language) refer, for instance, to [86].

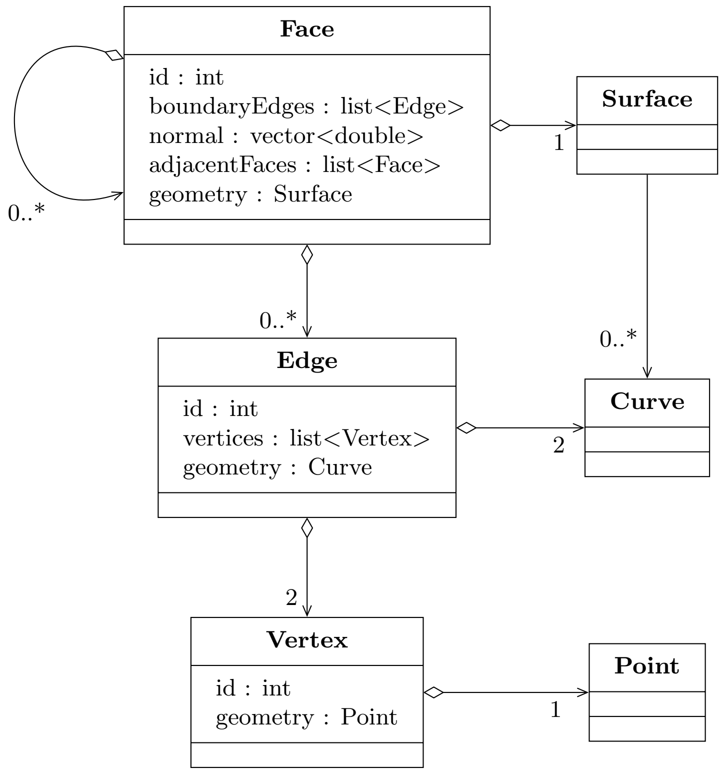


Figure 8.9: UML-diagram of a possible object-oriented B-rep implementation that clearly distinguishes between topology and and geometry.

it must contain t_i and, recursively, all underlying sub-topologies and geometries that are related by the respective internal adjacency relations r^{int} (see Figure 8.10).

$$\omega^{t_i}(T^{t_i}, G^{t_i}) \subset B\,, \tag{8.1}$$

with $T^{t_i}\left(\tau, \rho^{int}\right)$ and $G^{t_i}(\gamma)$ being the respective topology and geometry, where $\tau = \{\tau_i\}$ and $\gamma = \{\gamma_i\}$ denote the sets of those topological and geometrical entities which are recursively related by the internal relations $\rho^{int} = \{\rho_i^{int}\}$. Consequently, three different segments are possible: (a) vertex sub-parts, (b) edge sub-parts, and (c) face sub-parts. As topological entities $\{\tau_i\}$, a face segment, for example, contains the face itself and all associated edges and vertices. As geometric entities $\{\gamma_j\}$, it holds the corresponding surface with its boundary curves and corner points. Additionally, all internal relations $\{\rho_i^{int}\}$ are contained, i.e., the relations among the face, the boundary edges, and the vertices, as well as the relations to the geometric entities. Not contained are *external* adjacency relations, i.e., those to neighboring faces or edges.

In the following, $\tilde{a}$ denotes the object a after the transformation and let $\operatorname{dist}(a, b) = \inf\{\, \|a, b\|_2 \,\}$ be the minimum Euclidean distance between two objects, e.g., the distance between the closest points on two different surfaces.

Flaw operators

1. **Selection:** Let O^{select} be an extraction operator, which selects for a topological entity t_i the

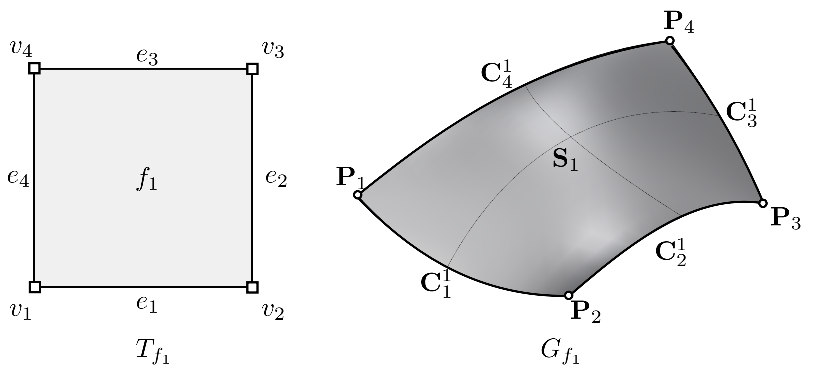


Figure 8.10: B-rep sub-part ω^{f_1}, which corresponds to face f_1 and consists of the topology T^{f_1} and the corresponding geometry G^{f_1}.

corresponding segment ω^{t_i} from the body B.

$$O^{select}\ (B\,,\, t_i) \mapsto \omega^{t_i}. \tag{8.2}$$

Note that the external relations are not extracted. Hence, the segment forgets about its *logical* location in the body.

2. **Adding:** Let O^{add} be a join operator that adds a segment ω^{t_i} to the body B.

$$O^{add}\ \left(B\,,\, \omega^{t_i}\right) \mapsto \tilde{B}\,, \ \text{ where } \ \tilde{B} = B \cup \omega^{t_i}. \tag{8.3}$$

3. **Shallow copy:** Let $O^{shallowCopy}$ be a shallow copy operator that copies an arbitrary entity a_i. Briefly, *shallow* means only the object itself, but not the underlying and related data is copied. For a more detailed description of the object-oriented concept of *shallow* and *deep* copying see, e.g., [87]. a_i can be a topological entity t_i, a geometrical entity g_i, or an internal r_i^{int} or external relation r_i^{ext}.

$$O^{shallowCopy}\ (a_i) \mapsto \tilde{a}_i\,, \ \text{ where } \ \tilde{a}_i := a_i. \tag{8.4}$$

The ':=' in (8.4) is to be understood as the shallow copy assignment, according to [87]. Note that the new object is distinguishable from the old object, e.g., by an updated id, or, in the context of object-oriented programming, by a different memory address. Yet, it still uses the same references to other objects as the original segment.

4. **Deep copy:** Let $O^{deepCopy}$ be an internal deep copy operator [87] that performs a deep copy operation on a segment ω^{t_i}. To this end, a shallow copy operation is carried out on all corresponding topological and geometrical entities as well as the internal adjacency relations.

$$\begin{aligned} O^{deepCopy}\ \left(\omega^{t_i}\right) \mapsto \tilde{\omega}^{t_i}(\tilde{T}^{t_i}, \tilde{G}^{t_i})\,, \ \text{ where } \ \tilde{T}^{t_i}\left(\tilde{\tau}, \tilde{\rho}^{int}\right) \text{ and } \tilde{G}^{t_i}\left(\tilde{\gamma}\right) \text{ consist of} \\ \tilde{\tau}_i := O^{shallowCopy}(\tau_i)\ \ \forall \tau_i \in T^{t_i}\ , \\ \tilde{\rho}_i^{int} := O^{shallowCopy}(\rho_i^{int})\ \ \forall \rho_i^{int} \in T^{t_i}\ , \\ \tilde{\gamma}_i := O^{shallowCopy}(\gamma_i)\ \ \forall \gamma_i \in G^{t_i}\ . \end{aligned} \tag{8.5}$$

Note that the deep copied segment ω^{t_i} has no information about its *logical* location in B – i.e., it has no external adjacency relations – and that all internal relations are updated to reference the new topological and geometrical entities.

5. **Deletion:** Let O^{delete} be a deletion operator that deletes a face f_i and its related geometry $g^{f_i} \subset G$ consisting of the underlying surface $\boldsymbol{S}_i$ and the corresponding boundary curves $\{\boldsymbol{C}_j^{\boldsymbol{S}_i}\}$. Thereby, the characteristic size of the resulting opening must not exceed a given, e.g., user-defined minimal accuracy ε. Let δ be the diameter of the largest possible inscribed sphere of the surface $\boldsymbol{S}_i$ to be deleted.

$$O^{delete}\left(B, f_i, g^{f_i}\right) \mapsto \tilde{B}\,(\tilde{T}, \tilde{G})\,, \quad \text{where} \;\; \tilde{F} = F \setminus f_i \subset \tilde{T}\;,\; \tilde{G} = G \setminus g^{f_i}\;,\; \delta < \varepsilon\,. \tag{8.6}$$

Since a deletion of an edge, or a vertex would lead to an uncontrollable cascade of deletions of superior entities, only a face deletion is allowed in this context. A B-rep model with a deleted edge or node without deletion of referencing faces would not be interpretable and is not considered in our investigation.

6. **Explosion:** Let $O^{explode}$ be an operator that removes all external relations r^{ext} from a body B. This can be achieved by extracting (8.2), copying (8.5), and adding (8.3) all face segments $\omega^{f_i}\ \forall f_i \in F$. The resulting body $\tilde{B}$ is then described by independent topological sup-parts/segments ω^{f_i}.

$$O^{explode}\,(B) = O^{add}\left(B^{\circ},\, O^{deepCopy}\left(O^{extract}\,(B,\, F)\right)\right) \mapsto \tilde{B}, \tag{8.7}$$

where B° is an empty body. As main consequence all information about adjacency relations is lost $\tilde{r}^{ext} = \emptyset$.

7. **Flipping:** Let O^{flip} be a topological flip operator that flips the normal $\boldsymbol{n}_i$ of a face f_i.

$$O^{flip}\left(f_i\right) \mapsto \tilde{f}_i\;, \quad \text{where} \;\; \tilde{f}_i = ((e_\kappa), \tilde{\boldsymbol{n}}_i = -1 \cdot \boldsymbol{n}_i)\;. \tag{8.8}$$

8. **Moving:** Let O^{move} be a geometric move operation that moves the point $\boldsymbol{P}_i$ (corresponding to vertex v_i) within the range ε. Additionally, all adjoined surfaces and curves are adapted consistently such that they form a 2-manifold without boundaries after the operation. This involves the following adaptions to the adjoined surfaces $S^{v_i} = \{\boldsymbol{S}_j^{v_i}\}$ and curves $C^{v_i} = \{\boldsymbol{C}_j^{v_i}\}$:

   - The resulting point $\tilde{\boldsymbol{P}}_i$ must again lie on the altered surfaces $\tilde{S}^{v_i}$ and curves $\tilde{C}^{v_i}$.
   - The altered curves $\tilde{\boldsymbol{C}}^{v_i,\tilde{\boldsymbol{S}}_j}$ must again form the boundary of the respective surfaces $\tilde{\boldsymbol{S}}_j$.
   - All pairs of the resulting adjoined surfaces $(\tilde{\boldsymbol{S}}_A^{e_k},\, \tilde{\boldsymbol{S}}_B^{e_k})$ must again meet at their common edge/curve e_k. Consequently, the two respective boundary curves $(\tilde{\boldsymbol{C}}_A^{e_k},\, \tilde{\boldsymbol{C}}_B^{e_k})$ must coincide.

   The latter condition is omitted in the case of an already broken topology, where an edge no longer has two adjoined faces/surfaces.

$$\begin{aligned} & O^{move}\,(\boldsymbol{P}_i,\, G) \mapsto \tilde{G}\;, \quad \text{where} \;\; 0 < \operatorname{dist}\left(\boldsymbol{P}_i, \tilde{\boldsymbol{P}}_i\right) < \varepsilon\,, \\ & \qquad \text{and} \\ & \qquad \exists\, \boldsymbol{\xi} : \tilde{\boldsymbol{S}}_j^{v_i}(\boldsymbol{\xi}) = \tilde{\boldsymbol{P}}_i \;\; \forall\, \tilde{\boldsymbol{S}}_j^{v_i}\;,\; \exists\, \zeta : \tilde{\boldsymbol{C}}_j(\zeta) = \tilde{\boldsymbol{P}}_i \;\; \forall\, \tilde{\boldsymbol{C}}_j^{v_i}\;, \\ & \qquad \text{and} \\ & \qquad \operatorname{dist}\left(\tilde{\boldsymbol{C}}_k^{v_i,\tilde{\boldsymbol{S}}_j}, \tilde{\boldsymbol{S}}_j^{v_i}\right) = 0 \quad \forall j, k \text{ at } v_i\,, \\ & \qquad \text{and} \\ & \qquad \operatorname{dist}\left(\tilde{\boldsymbol{C}}_A^{e_k},\, \tilde{\boldsymbol{C}}_B^{e_k}\right) = 0 \;\; \forall\, (\tilde{\boldsymbol{S}}_A^{e_k},\, \tilde{\boldsymbol{S}}_B^{e_k}) \text{ at } v_i\,. \end{aligned} \tag{8.9}$$

9. **Detaching:** Let O^{detach} be a geometrical operator that detaches two adjacent surfaces, $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$, which meet at the edge e_k (see Figure 8.5b). To this end, one surface $\boldsymbol{S}_i$ and its respective boundary curve $\boldsymbol{C}^{e_k,\boldsymbol{S}_i}$ at e_k are changed. Again, the characteristic size of the potentially resulting opening must not exceed ε.

$$\begin{aligned} &O^{detach}\left(G, e_k\right) \mapsto \tilde{G}\ , \ \text{ where } \operatorname{dist}\left(\tilde{\boldsymbol{S}}_i, \tilde{\boldsymbol{C}}^{e_k,\tilde{\boldsymbol{S}}_i}\right) = 0\ , \\ &\qquad 0 < \operatorname{dist}\left(\boldsymbol{C}^{e_k,\boldsymbol{S}_i}(\xi), \tilde{\boldsymbol{C}}^{e_k,\tilde{\boldsymbol{S}}_i}(\xi)\right) < \varepsilon \ \ \forall \xi \in [\xi_a, \xi_b]\ , \end{aligned} \tag{8.10}$$

with $[\xi_a, \xi_b]$ being the respective interval on which the boundary curve is defined.

10. **(Self-)intersection:** Let $O^{intersect}$ be a geometric operator that alters a surface $\boldsymbol{S}_i(\boldsymbol{\xi})$ such that it touches or intersects with another surface $\boldsymbol{S}_j(\boldsymbol{\eta})$ apart from common edges. Note that no intersection in the original model is assumed, according to the definition of a valid B-rep model.

$$\begin{aligned} &O^{intersect}\left(\boldsymbol{S}_i(\boldsymbol{\xi})\right) \mapsto \ \tilde{\boldsymbol{S}}_i(\boldsymbol{\xi})\ , \ \text{ where} \\ &\ \exists\,(\boldsymbol{\xi}, \boldsymbol{\eta}) \ : \ \operatorname{dist}\left(\tilde{\boldsymbol{S}}_i(\boldsymbol{\xi}), \boldsymbol{S}_j(\boldsymbol{\eta})\right) = 0 \ \wedge \ \operatorname{dist}\left(\tilde{\boldsymbol{S}}_i(\boldsymbol{\xi}), \tilde{\boldsymbol{C}}_k^{\tilde{\boldsymbol{S}}_i}\right) > 0 \\ &\ \forall\, \tilde{\boldsymbol{C}}_k^{\tilde{\boldsymbol{S}}_i} \in \Gamma^{\tilde{\boldsymbol{S}}_i}\ ,\ i \neq j\ , \end{aligned} \tag{8.11}$$

with $\Gamma^{\tilde{\boldsymbol{S}}_i}$ being the set of boundary curves of $\tilde{\boldsymbol{S}}_i$.

A special case of intersections are self-intersections:

$$O^{selfIntersect}\left(\boldsymbol{S}_i(\boldsymbol{\xi})\right) \mapsto \ \tilde{\boldsymbol{S}}_i(\boldsymbol{\xi})\ , \ \text{ where } \exists\,(\boldsymbol{\xi}, \boldsymbol{\eta}) \ : \ \operatorname{dist}\left(\tilde{\boldsymbol{S}}_i(\boldsymbol{\xi}), \boldsymbol{S}_i(\boldsymbol{\eta})\right) = 0\ ,\ \boldsymbol{\xi} \neq \boldsymbol{\eta}\ . \tag{8.12}$$

8.3.4 Application of the flaw operators

To create a flawed model, the flaw operators, defined in Section 8.3.3, are applied onto a valid B-rep model. The 'defectiveness' of the model is then defined and controlled by ε. It should be mentioned that, in reality, flaws do not necessarily originate from these operators. However, it is possible to create most flaws by a sequence of these operators.

Let $B(T, G)$ be a valid, flawless B-rep body. Note that operators acting on the body are to be understood as working on a segment ω^{t_i}, or single topological, or geometrical entity or relation.

1. **Multiple entities:** Single topological entities t_i and their corresponding segments ω^{t_i} can be copied and added to B with a combination of the extraction (8.2), the deep copying (8.5), and the joining (8.3) operator:

$$\tilde{B}\,(\tilde{T}, \tilde{G}) := O^{add}\ \left(B\,(T, G),\, O^{deepCopy}\ \left(O^{extract}\,(B\,(T, G),\, t_i)\right)\right) \tag{8.13}$$

The resulting B-rep model is invalid as it has multiple entities (refer to Figures 8.4a, 8.4b, and 8.4c), which violates Condition 1 for valid B-rep models. As an example, consider the STL format where each triangle (re-)defines its corner points. Also, multiply defined faces/surfaces appear frequently in free form CAD models, which then additionally lead to overlaps and intersections of the surfaces (see Figures 8.6c and 8.6b).

2. **Missing entities:** Application of the deletion operator (8.6) on a face f_i:

$$\tilde{B}\,(\tilde{T}, \tilde{G}) := O^{delete}\ (B, f_i, g_{f_i}) \tag{8.14}$$

The deletion of a face violates Condition 2 (see Figure 8.4e). Thereby, the size of the resulting opening is restricted to be smaller than ε.

3. **Detached topology:** Application of the explosion operator (8.7):

$$\tilde{B}\,(\tilde{T}, \tilde{G}) := O^{explode}(B(T, G)) \tag{8.15}$$

Most B-rep models are constructed from independent surfaces, which are later joined into a (hopefully) valid B-rep model. This join operation corresponds to the inverse of the explosion operation. Often a 'join'-operation is not possible (or maybe not feasible), e.g., in case of an intersection of two NURBS surfaces [80]. STL models are constructed that way as well. Such models violate the topological Conditions 1, 2, and 3. Geometrically, they can still form a closed 2-manifold without boundaries. However, these models are very prone to a variety of different flaws, as no external adjacency relations are provided explicitly.

4. **Wrong orientations:** Application of the flip operator (8.8):

$$\tilde{B}\,(\tilde{T}, G) := O^{flip}(B(T, G)) \tag{8.16}$$

The resulting B-rep model does not follow Moebius' Rule anymore (see Condition 4). This flaw usually appears, if the normal is defined implicitly by the order of the boundary edges (see Figure 8.4d). However, this error also appears quite frequently, if the normal is given explicitly, e.g., in the case of STL.

5. **Movement:** Application of the move operator (8.9):

$$\tilde{B}\,(T, \tilde{G}) := O^{move}(B(T, G)) \tag{8.17}$$

The movement of a point does not necessarily cause any flaws. Only if applied on a point $\boldsymbol{P}_i$ on surface $\boldsymbol{S}_j$, which is close to surface $\boldsymbol{S}_k$ with distance $\operatorname{dist}(\boldsymbol{P}_i,\, \boldsymbol{S}_k) < \varepsilon$ the move operation can lead to an intersection (or a self-intersection, if $\boldsymbol{S}_j = \boldsymbol{S}_k$), which violates the Conditions 8, 7 (see Figure 8.5d). Such an intersection can lead also to non-orientability and thus a violation of Condition 4.

6. **Surfaces not coinciding:** Application of the detach operator (8.10):

$$\tilde{B}\,(T, \tilde{G}) := O^{detach}(B(T, G)) \tag{8.18}$$

The resulting model violates Condition 6. For most B-rep CAD models this appears frequently at the connection of free-form surfaces. The boundary curves would require unreasonably high polynomial degrees to perfectly coincide. Possible flaws can be, e.g., openings or intersections (see Figures 8.5b and 8.5d). As an example, consider the well-known leaking Utah teapot.

7. **(Self-)intersections:** Application of the intersection operator (8.11):

$$\tilde{B}\,(T, \tilde{G}) := O^{intersect}(B(T, G)) \tag{8.19}$$

The resulting model may violate Conditions 7 or 8. Apart from gaps, intersections appear frequently at patch boundaries as well (see Figure 8.5b). Intersections can also occur if two surfaces are too close to each other. In this case, they additionally violate Condition 4 (see Figures 8.5d and 8.5c). A special case are overlaps, where two surfaces touch each other (see Figure 8.6b).

8. **Offsets & artifacts:** Application of the copy (8.5) and move operators (8.9) to a single face f_i :

$$\begin{aligned}
\tilde{\omega}^{f_i} &:= O^{deepCopy}\left(O^{extract}\,(B,\, f_i)\right) \\
\breve{\omega}^{f_i} &:= O^{move}(\boldsymbol{P}_j \in \tilde{\omega}^{f_i},\, \tilde{G}^{t_i}) \\
\tilde{B}\,(\tilde{T}, \tilde{G}) &:= O^{add}\left(B,\, \breve{\omega}^{f_i}\right)
\end{aligned} \tag{8.20}$$

This chain of operations allows to create offsets and artifacts, i.e., entities which do not belong to the outer hull and lead to a violation of the Conditions 2 and 3 (see Figures 8.6d and 8.6e).

9. **Gaps & intersections:** Application of the explosion (8.7) and move operators (8.9):

$$\begin{aligned}\tilde{B}(\tilde{T}, G) &:= O^{explode}(B(T, G)) \\ \breve{B}(\tilde{T}, \breve{G}) &:= O^{move}(\boldsymbol{P}_i \in G, G)\end{aligned} \tag{8.21}$$

   Starting from an exploded model, moving one or more points can lead to various common flaws – such as gaps, intersections, or overlaps (see Figures 8.6a, 8.6c, 8.6b). Since many B-rep modeling tools work with exploded models, these flaws appear very commonly, in particular, at surface boundaries.

Regarding a direct numerical simulation with the FCM, as described in Section 10.2, the size of all openings and gaps must be smaller than a pre-defined ε_{crit}, independent of the performed operations. This limit is required not only for each flaw operation but also for the resulting model after a sequence of flaw operations, e.g., a series of individual movements of a segment.

Chapter 9

Geometric models for functionally graded material

Functionally graded materials (FGM) are a class of advanced materials that offer the possibility to exploit various desired physical properties within one component – for instance, chemical, or thermal resistance of load-bearing parts. FGMs allow the production of high-performance and multi-functional objects that can resist environmental exposures that could not be withstood by a single material [88]. In contrast to composite materials with a step-wise material distribution, the changes in the material properties in FGMs are continuous. Consequently, no material interfaces occur, which renders FGM resistant to delamination [89]. Specific material properties are achieved in various ways. Conceivable are, among others, continuously changing micro-structures, grain sizes, crystal structures, fiber orientations, or a varying composition of different materials such as metal, ceramics, polymers, or biological tissues [90, 91], as depicted in Figure 9.1. This chapter follows mainly the publication [15].

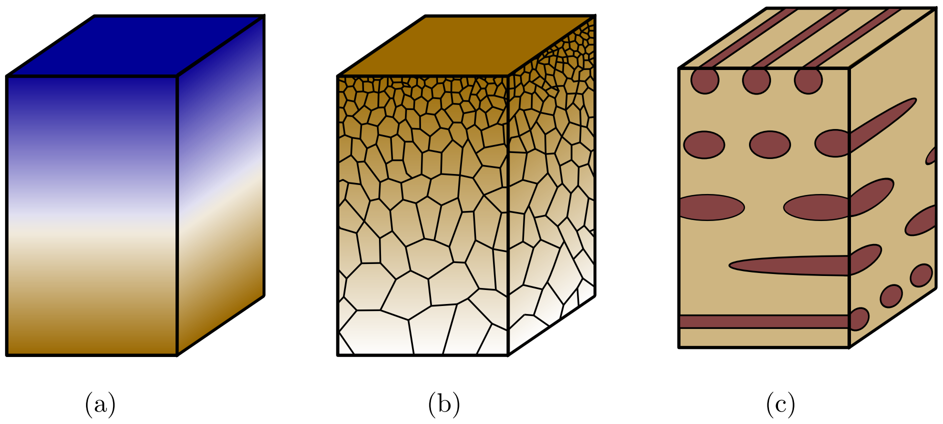


Figure 9.1: Some different types of functionally graded materials: (a) blending of two different materials, (b) changing grain size, and (c) varying orientation of fibers.

With *additive manufacturing* (AM) emerging, FGMs became better applicable, since the layer-wise material deposition and fine resolution allow the creation of microstructures that can even be enclosed, e.g., by a shell. Additionally, a continuous blending of different materials can be realized by changing the source material's composition on the respective layers. Furthermore, the size or orientation of grains can be influenced by the adjustment of AM parameters. As an example, consider selective laser melting. Different material properties are obtained, by varying, for instance, the composition and the grain size of the source material, the laser velocity, or the shape and power output of the laser beam [92, 93]. Since, AM is the method of choice for the fabrication of FGM, the field of *functionally graded additive manufacturing* (FGAM) came up, which coarsely categorizes the different FGMs into *single-* and *multi-material FGMs* [94].

Single-material FGMs consist of only one material that changes its properties due to an adaption of the microstructure, density, or grain size [95]. In nature, a prominent example is a trabecular bone, where the size and alignment of thin rods and plates of bone tissue create stiffness trajectories that follow the principal stresses for the most common load cases [96]. Such materials can easily be created with all AM techniques since the inherent capability to create free form structures allows the creation of changing microstructures. In fact, AM already uses porous infill structures to support the outer hull, which is usually the desired shape. However, this infill is typically a constant, repetitive lattice and is either not taken into account for the material behavior, or is assumed to be isotropic [97]. Regarding structural analyses, recent approaches in topology optimization try to exploit the contribution of the infill to the load transfer [98]. Problem-fitted complex 3D anisotropic microstructures can reduce the printing time and material consumption substantially and, at the same time, improve the load-carrying properties and buckling behavior.

Multi-material FGMs, on the other hand, combine two or more materials and have recently been under intensive research [99]. A particular focus was laid on metal-metal combinations, for instance steel and titanium-based combinations (see [91]). The blending of materials of different natures, such as ceramic-metal compositions [100], is significantly more involved. However, these compositions might carry the highest potential, as the underlying material properties are very distinct.

In the following, geometric modeling techniques are introduced to represent single-, as well as multi-material FGM.

9.1 FGM with geometrical representations

Material testing is the industry standard to determine the behavior of FGM components. Yet, physical test series are often elaborate and expensive. Therefore, the goal of simulation supported development is to reduce testing to only calibrating data for functionally graded materials and then numerically analyze different shapes and compositions of artifacts. For this, in general, an analysis-suitable CAD model is required. The common representation schemes – B-rep and solid procedural models – are only limited suitable to represent FGMs.

B-rep only implicitly describes the volume and thus offers no possibility to directly represent a heterogeneous material distribution. Nevertheless, various methods examine the applicability of B-rep models. The basic idea is to describe the FGM as vector functions, which can be (i) geometrically-independent, e.g., in Cartesian coordinates, (ii) distance-based, (iii) blending composition, or (iv) sweeping composition functions (for a detailed explanation refer to [101, 102]). However – except (i) – these functions only allow a smooth transition of material properties between the different surfaces, which is not suitable for all kinds of material distributions. On the other hand, geometrically-independent functions are cumbersome as they are not related to the object itself.

With solid procedural modeling, different materials can easily be assigned to the individual primitives. However, for adjacent, non-overlapping primitives, this leads to a material distribution that is constant in the primitives and then abruptly changes at the interfaces. This material jump corresponds to a composite, not an FGM. In regions with overlapping primitives, the material needs to be interpolated [92]. Again, vector functions, applied to the intersected regions, are a possible workaround.

Spatial decomposition offers another approach to represent FGM. Most models of this type are monochrome voxel domains that originate from some sort of tomography. Here, each voxel's gray value is related to the particular material property, e.g., the density, or water content. Although voxel models only provide a coarse approximation and are simultaneously memory inefficient – they have been used to resolve fine microstructures and quasi-continuous changes of the material properties [103, 104].

9.2 FGM based on V-rep

As part of the V-rep framework, V-models were developed to overcome the shortcomings of the common geometric representation forms, regarding the possibility to represent and model FGM. Within the V-rep framework, it is possible to model multi-material FGM and single-material FGM in the form of microstructures.

9.2.1 Multi-material FGM based on V-rep

Apart from the geometric shape, V-cells – as smallest building blocks of the V-models – are able to represent additional material properties. For this, simply the dimension of the control points is extended to $\mathbb{R}^{3+s}$, with $s > 0$ being the additional material parameters.

$$\boldsymbol{P}_i = \left[x_i, y_i, z_i, \mu_i^1, ..., \mu_i^h, ..., \mu_i^s\right]^T \in \mathbb{R}^{3+s}, \quad h \in \{1, ..., s\}, \tag{9.1}$$

where μ_i^h corresponds to the $i^{th}-$ material 'coordinate' of the $h^{th}-$ material property. Consequently, evaluating a V-cell according to Equation (6.12) yields, in addition to the geometric coordinates, also the respective material values

$$\boldsymbol{V}(\boldsymbol{\xi}) = \left[x, y, z, m^1, ..., m^h, ..., m^s\right]^T(\boldsymbol{\xi}) \in \mathbb{R}^{3+s}, \quad h \in \{1, ..., s\}. \tag{9.2}$$

For instance, consider a control point that carries additional material properties for the Young's modulus E, Poisson's ratio ν, and thermal conductivity κ, as needed in Example 15.2: $\boldsymbol{P}_{i,j,k}^T = [x,\ y,\ z,\ E,\ \nu,\ \kappa]_{i,j,k}$.

The material properties of a V-cell that originate from the overlap of two or more trivariate splines and carry different material information require additional handling. Either one of the initial trivariate splines can be set prevailing – thus, its properties are inherited to the V-cell –, or some sort of blending scheme interpolates the material properties. For detailed information, refer to [6].

Inside a patch, splines are typical of higher continuity, which renders them perfectly suitable for modeling smooth geometries. However, this restricts the material function to be of the same continuity. Knot insertion, as described in Section 6.7.2 allows the representation of C^0 or discontinuous material. It preserves the original geometrical continuity but reduces the continuity of the shape functions[1]. Hence, it is possible to represent material kinks or material discontinuities. Nevertheless, due to the global influence of the knots' position and multiplicity, splines are not the method of choice to represent highly discontinuous material distributions, as, e.g., underlying voxel data provided by CT-scans.

Given a sufficiently smooth material distribution f_{m^h}, the material 'coordinates' μ_i^h of the control points can be obtained using a least squares approximation, according to Section 6.8.2. The geometrical position of the control points as well as the knot positions and polynomial degree can not be changed, since this would lead to a different geometrical shape. These restrictions lead to a modified least squares problem (refer to Equation 6.35), which needs to be solved for each material property m^h

$$\min_{\boldsymbol{\mu}^h} \sum_{j=1}^{n_s} r_j^2 = \min_{\boldsymbol{\mu}^h} \sum_{j=1}^{n_s} \left(\boldsymbol{V}\left(\overline{\boldsymbol{\xi}}_j, \boldsymbol{\mu}^h\right) - f_{m^h}(\overline{\boldsymbol{\xi}}_j)\right)^2 = \min_{\boldsymbol{\mu}^h} \|\boldsymbol{V}\left(\overline{\boldsymbol{\xi}}, \boldsymbol{\mu}^h\right) - \boldsymbol{f}_{m^h}\|_2^2, \tag{9.3}$$

[1] *Remark:* This is only the case for the undeformed, initial CAD model. The deformed shape can be of $C^{p-\kappa}-$continuity, for instance, a kink in the case of C^0.

where the minimization variables are the material 'coordinates' $\boldsymbol{\mu}^h = \mu^h_{i,j,k} \in \mathbb{R}^{l \cdot n \cdot m}$ of the $l \cdot n \cdot m$ control points. Since the function describing the respective material distributions f_{m^h} is, in general, not defined in the spline's parameter space, one or more intermediate mappings are required. For instance, for a function that is defined in Cartesian coordinates this mapping reads

$$f_{m^h}(\boldsymbol{x}) = f_{m^h}(\boldsymbol{V}(\boldsymbol{\xi}))\,. \tag{9.4}$$

It is noteworthy that due to the fixed geometry, the material approximation is limited, as can be seen in Figure 9.2 for a one-dimensional approximation of a sinusoidal material function.

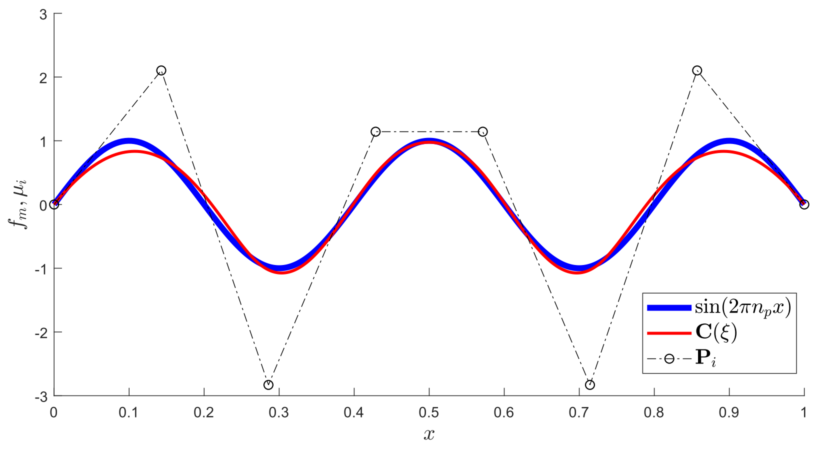
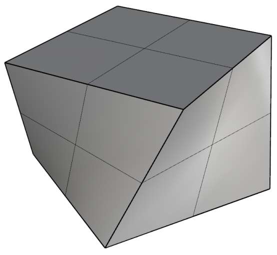


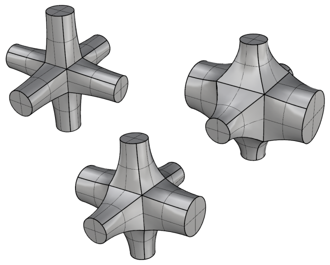
Figure 9.2: One-dimensional least squares approximation of a sinusoidal material function $f_m(x) = sin(2\pi n_p x)$, with $n_p = 2.5$ being the number of periods. The rather large deviation between the curves is due to the fact that the location (i.e., $x-$coordinates) of material control points, the polynomial degree, and the positions of the knots are fixed.

9.2.2 Single-material FGM based on V-rep

The V-rep framework offers the possibility to create single-material FGM in the form of complex anisotropic microstructures with its *tiling* operation. To this end, copies of a unit structure are consecutively created inside a base volume. Following the shape of the base volume, and by using layers of different unit cells, a complex constructive FGM can be created as depicted in Figure 9.3. Since the resulting microstructure is composed of several V-cells, it is again a V-model. Naturally, each V-cell can represent a heterogeneous material distribution within its volume.

Even for such complex tile-based structures, like the example shown in Figure 9.3, a simulation on the V-rep model is possible. Yet, in the case of single-material structures, it turns out that the conversion into an auxiliary B-rep is computationally much more efficient.


(a) Ruled base volume


(b) Unit tiles

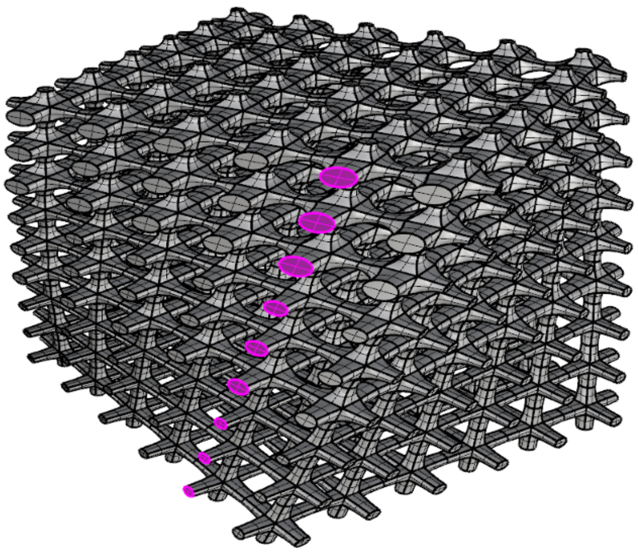

(c) Microstructure

Figure 9.3: Functionally graded microstructure: (a) A rotating ruled volume is tiled with (b) different anisotropic unit tiles that have a varying stiffer direction. (c) The resulting microstructure has a continuously changing anisotropic material behavior.

Part III

Direct simulation with the FCM

As explained in Chapter 4, direct simulation methods were developed to circumvent the error-prone and cumbersome transition process from a CAD model to an analysis-suitable geometry description. Whereas IGA was very successfully applied to shell problems, the FCM is ideal for simulations of solid objects. Unlike IGA, the FCM is not so much limited on the kind of geometric representation. For instance, a simulation can be directly carried out on point clouds or voxel domains. For a simulation with IGA, this would involve a preceding conversion into a CAD model. Besides, FCM can simulate directly on invalid, flawed CAD models. This flexibility is due to the sole requirement of a point membership classification (PMC), which is the central aspect of this part.

The point membership can be considered to be the most basic geometrical information. Consequently, it must be provided by every geometric (volumetric) representation. PMC algorithms are extensively used operations, e.g., in computer graphics, computer games, and in geoinformatics [105]. The PMC can be carried out also on non-CAD models, e.g.:

- **Spatial decomposition/voxel models:** The point membership is directly stored on the leaves of the representation tree and, thus, directly accessible. Voxel domains typically stem from monochromatic (grey-scale) tomography scans. Here the difficulty is to classify the voxels correctly into inside and outside since the images are typically not homogeneous. In general, it is not sufficient to choose one threshold on the grey value for the entire scan. Often it is necessary to adapt the threshold for the different slices or even inside one single slice. To obtain decent results from a numerical simulation, the voxel resolution must be sufficiently high so that the non-physical step-wise approximation has only a minor influence. For a detailed explanation, refer to [106].

- **Point clouds:** Provided that the points in the point cloud carry – apart from their location – also their respective normals, a PMC can easily be realized. For this, the sign of the scalar product between the vector from the point of interest to the closest point in the point cloud and the respective normal indicates the point membership (see Section 10.1). However, since point clouds are usually not created from valid CAD models – but rather from some imaging or scanning process – point clouds are typically noisy. In order to obtain reliable results, additional PMCs are carried out with several neighbors of the closest point and averaged accordingly (refer to [107]). Another problem is the application of boundary conditions, as for these, a surface description is required. As a remedy, the boundary conditions can be converted into volume conditions.

In the following, point inclusion tests for the respective CAD representations of Part II are explained. In addition, the modifications of the FCM to deal with FGMs are discussed. Regarding the simulation of single-material FGM (i.e., microstructures), homogenization is briefly explained.

Chapter 10

Point membership classification on B-rep models

Since B-rep models represent the volume via the boundaries, the point membership classification also depends solely on the boundaries. For this, various algorithms are available. Naturally, all of them require a valid surface description. Only a combination of different approaches gains a robust and accurate algorithm for flawed B-rep models.

10.1 PMC for valid B-rep models

Applicable PMC algorithms for valid B-reps are, for instance, the (a) *winding number* method [108], (b) querying the normal of closest point on the boundary, (c) a test on a point cloud approximation, or (d) *ray tracing* [109]. Figure 10.1 illustrates the concepts of the individual algorithms for a two-dimensional polygon.

(a) **Winding number:** The basic idea of the winding number is, to sum up all angles between line segments and the point of interest. The resulting angle θ_{res} is 0 or 2π, if the point is outside or inside, respectively. The winding number approach is not easy to extend to three dimensions since the term 'angle' is in 3D not directly applicable [110]. Moreover, for a curved boundary, it is not sufficient to simply sum up the 'angles' between line segments, but the winding number needs to be determined by a boundary integral.

(b) **Closest boundary point:** For this approach the closest point $\boldsymbol{Q}$ on the boundary – for a point $\boldsymbol{P}$ – is sought. At $\boldsymbol{Q}$ the normal $\boldsymbol{n_Q}$ is evaluated. Based on the sign of the scalar product $\boldsymbol{n_Q} \cdot (\boldsymbol{Q} - \boldsymbol{P})$ it is possible to distinguish inside and outside. This test is also applicable for multiple closest points, or multiple normals at the same closest point, e.g., at corners. Additionally, this test can be extended straightforwardly to three dimensions. However, B-rep models are usually composed of a set of independent boundary surfaces and, thus, do not necessarily provide the correct normal direction.

(c) **Point clouds:** B-rep surfaces can be easily approximated with point clouds, for instance, by extracting the corner points of a surface triangulation. The PMC test on point clouds is carried out similarly to the test with the closest boundary point by determining the sign of the scalar product. Admittedly, this PMC – like the test with the nearest boundary point – depends on the correct definition of the boundary normals. Additionally, point clouds are only limited suited to represent sharp features, such as corners and edges. Hence, this PMC is only sensible, if no other geometrical description is available.

(d) **Ray tracing:** The most robust and widely used PMC algorithm for B-rep models uses *ray tracing*. The basic idea of ray tracing dates back to the 16^{th} century. It corresponds to the inverse of the natural way light is emitted, reflected on objects, and finally hitting the retina. Nowadays, the largest field of application is computer graphics, where it is used to render two-dimensional images of three-dimensional objects. Modern ray-tracing involves complex light scenes, shadows, transparency, etc. For this, rays are sent out from a virtual observer – passing through the visualization plane – into the scene of interest until it reaches an outer bounding box or a light. In contrast to ray casting, where the ray stops at the first obstacle, ray tracing evaluates all objects' (potential) intersections.

In a slightly modified way, this can be used for a point inclusion test [105]. Starting from the point of interest – i.e., an integration point in the case of the FCM – a ray shoots to a position which is certainly outside. The parity of the number of intersections then determines the point membership. Meaning the particular point is inside or outside if the number of intersections is odd or even. Critical in this context are intersections at edges and corners, as these can correspond to both, a real intersection, or only a contact (depicted by the blue rays in Figure 10.1(d)). Another critical case occurs if the ray is parallel to the tangential plane at the 'intersection' or 'contact' point of a curved surface. However, ambiguous cases can easily be detected and treated by a shift of the ray's outer endpoint.

10.1.1 Ray tracing with triangles

In the simplest case, the boundary is represented by plane polygons. As all polygons can be subdivided into triangles, only an intersection test with triangles is required. An efficient implementation is given with the *Möller-Trumbore* algorithm [111]. For this the triangle is transformed into *barycentric* coordinates

$$\boldsymbol{Q} = u\boldsymbol{A} + v\boldsymbol{B} + w\boldsymbol{C}\,, \tag{10.1}$$

where $\boldsymbol{Q}$ is a point on the triangle $\boldsymbol{ABC}$, at the local, baricentric coordinates $\boldsymbol{u}^T = [u, v, w]$, with $u + v + w = 1$. Rearranging leads to

$$\boldsymbol{Q}(v, w) = (1 - v - w)\boldsymbol{A} + v\boldsymbol{B} + w\boldsymbol{C} = \boldsymbol{A} + v(\boldsymbol{B} - \boldsymbol{A}) + w(\boldsymbol{C} - \boldsymbol{A})\,. \tag{10.2}$$

The parametric representation of the ray reads

$$\boldsymbol{R}(t) = \boldsymbol{O} + t\boldsymbol{D}\,, \tag{10.3}$$

with starting point, or origin, $\boldsymbol{O}$ and direction vector $\boldsymbol{D}$. The intersection point is computed as $\boldsymbol{Q}(v, w) = \boldsymbol{R}(t)$. If an intersection point exists, the resulting 3×3 linear system of equations is not singular and can be solved for instance with *Cramer's rule*

$$\begin{bmatrix} -\boldsymbol{D} & (\boldsymbol{B} - \boldsymbol{A}) & (\boldsymbol{C} - \boldsymbol{A}) \end{bmatrix} \begin{bmatrix} t \\ v \\ w \end{bmatrix} = \boldsymbol{O} - \boldsymbol{A}\,. \tag{10.4}$$

A decent approximation of a curved CAD model usually requires an enormous amount of triangles. For a numerical simulation with the FCM, the PMC needs to be carried out on each integration point. Since a triangle intersection test with every triangle would lead to extreme computational times, the triangles are typically stored in tree-structure such as a $kd-$tree.

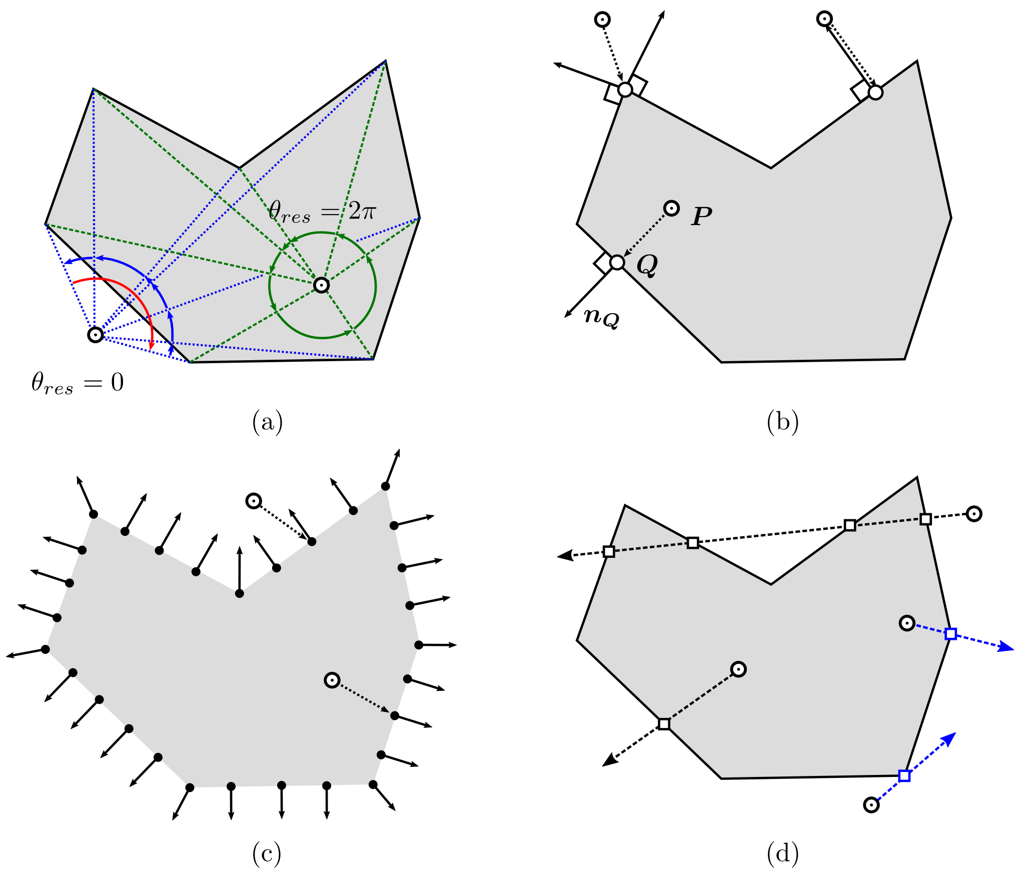


Figure 10.1: Point membership classification in two dimensions: (a) winding number, (b) normal of the closest point, (c) point cloud, and (d) ray tracing.

10.1.2 Ray tracing with curved boundaries

For lower-order spline surfaces, it is possible to extract the corresponding Bézier patches and find an analytical solution. To this end, the Bézier patches are expressed as implicit functions $f(x, y, z)$ and equated with the ray. The degree of the implicit equation is $2pq$, where p and q are the polynomial degrees of the tensor product splines [112]. Since, for a cubic spline this already leads to a polynomial of degree of $2pq = 18$ and 1330 terms, a surface-ray intersection is typically solved iteratively

$$\boldsymbol{r}(\xi, \eta, t) = \boldsymbol{S}(\xi, \eta) - \boldsymbol{R}(t) = \boldsymbol{0}\,, \tag{10.5}$$

for instance with the multivariate Newton's method

$$\begin{bmatrix} \dfrac{\partial r_1}{\partial \xi} & \dfrac{\partial r_1}{\partial \eta} & \dfrac{\partial r_1}{\partial t} \\ \dfrac{\partial r_2}{\partial \xi} & \dfrac{\partial r_2}{\partial \eta} & \dfrac{\partial r_2}{\partial t} \\ \dfrac{\partial r_3}{\partial \xi} & \dfrac{\partial r_3}{\partial \eta} & \dfrac{\partial r_3}{\partial t} \end{bmatrix}_{(\xi_i, \eta_i, t_i)} \begin{bmatrix} \xi_{i+1} - \xi_i \\ \eta_{i+1} - \eta_i \\ t_{i+1} - t_i \end{bmatrix} = -\boldsymbol{r}(\xi_i, \eta_i, t_i)\,. \tag{10.6}$$

10.2 PMC for invalid, flawed B-rep models

Considering flawed B-rep models, as presented in Section 8.3, the concept of 'inside' and 'outside' is in a strict mathematically sense not applicable. However, since most flaws are usually comparably small, a point inclusion test can be formulated, which is 'blind' up to the characteristic size of the defects ε. With a combination of spatial decomposition and ray tracing, it is possible to obtain a robust PMC algorithm that is accurate in flawless regions and delivers excellent approximations even in the vicinity of flaws. The PMC for flawed B-reps is described in [14].

This allows for a paradigmatic change in computational analysis:

"Instead of healing the geometry and topology, compute directly on the flawed models."

Certainly, the results' quality depends then on the type, size, amount, and location of the flaws. If a flaw lies directly in the region of interest, it can be fixed to achieve higher accuracy. An analysis, however, can be carried out either way.

Figure 10.2 illustrates the general approach of the PMC for a two-dimensional flawed model. The individual steps are as follows:

- The B-rep model is approximated by a space-tree T_{geo}, i.e., an octree in 3D and a quadtree in 2D. Thereby, all leaves that intersect with the boundary are marked as cut. The tree must be watertight to ensure that the subsequently applied *flood-fill algorithm* can distinguish between inside and outside.
- Starting from several seed points, a flood-fill algorithm is applied on T_{geo} to mark all connected points as inside and outside, respectively. This yields a filled space tree $\widehat{T}_{geo}$. For a detailed description of the flood-fill algorithm, refer to [5].
- Several additional ray tracings are carried out on points lying on the cut boundary leaves[1].

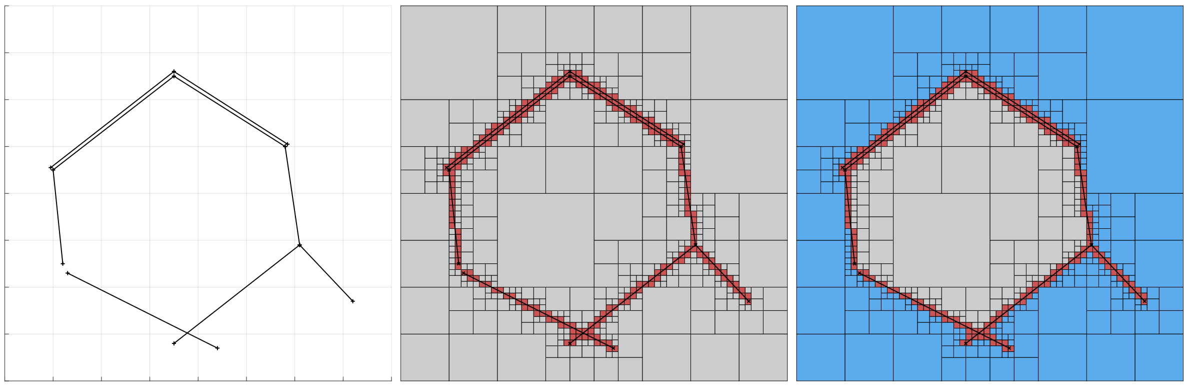

(a) Flawed geometry (b) Quadtree: $n_{max} = 6$ (c) Flood-fill: $n_{max} = 6$

Figure 10.2: The quadtree approximation of a flawed geometry refines recursively towards the boundary and marks the cut cells (red). The subsequent flood-fill algorithm marks the remaining cells as inner (gray) and outer (blue) regions accordingly.

[1]The approximation tree $\widehat{T}_{geo}$ is used for an efficient and accurate point inclusion test for all points that are not on cut leaves.

10.2.1 Watertight space tree approximation

The filled space-tree approximation $\widehat{T}_{geo}$ serves as initial, coarse stage of the PMC. To construct the unfilled tree T_{geo}, the domain is recursively subdivided at the boundaries up to certain level. Leaf cells c_{geo} on the boundary are marked as cut. Consequently, all cut cells are on the highest level (i.e., on the level most distant from the root of the tree) and are, thus, of the smallest size. To ensure that the approximation space-tree $\widehat{T}_{geo}$ is watertight, the size of the smallest leaves $d_{c_{geo}}$ must not undercut the characteristic size of the largest gap/opening ε_{gap}, as depicted in Figure 10.3

$$d_{c_{geo}} > \varepsilon_{gap}\,. \tag{10.7}$$

ε_{gap} is typically not known a priori and is determined by an iterative decrease of the cell size until the subsequent flood-fill algorithm marks the entire domain as inside or outside (of course, apart from the cut cells). Consequently, the maximal partitioning depth n_{max} is limited by the ratio between domain size d_{domain} and the size of the gaps ε_{gap}

$$n_{max} < \log_2\left(\frac{d_{domain}}{\varepsilon_{gap}}\right)\,. \tag{10.8}$$

Regarding all other types of flaws, the space-tree reconstruction $\widehat{T}_{geo}$ is robust. Depending on the gaps' size, only a very coarse, step-wise approximation of the actual geometry might be possible. However, as long as at least one inner cell can be detected, the reconstruction tree can be set up for an arbitrary flaw size ε_{gap}. The result's quality then depends only on the secondary ray tracing test (see Section 10.2.2).

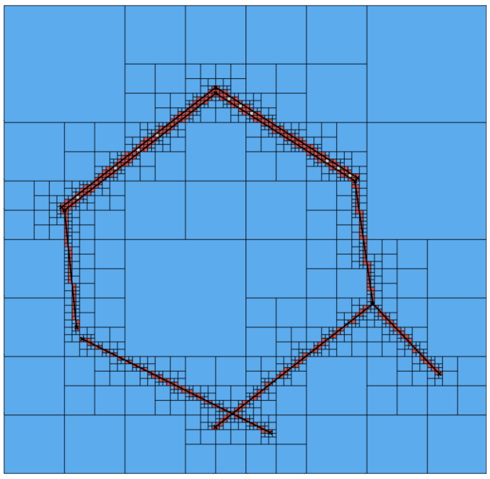


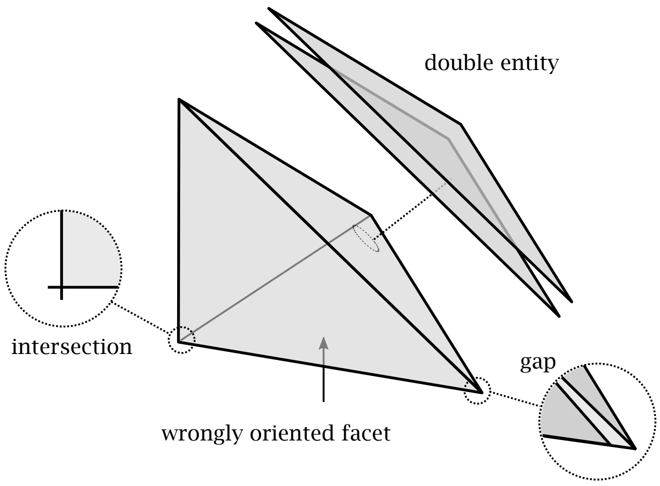
Figure 10.3: Too fine approximation tree $\widehat{T}_{geo}$ with a subdivision depth of $n_{max} = 7$ (see Figure 10.2). The flood-fill marks the entire domain as outside.

After the space-tree approximates the surface, the flood-fill algorithm is applied to mark connected regions [5]. The flood-fill algorithm requires seed cells as starting points. To this end, the computational domain is extended or surrounded by a ghost layer of outside cells, from which the algorithm starts and marks all outer cells accordingly. The remaining cells are inside. Obviously, this is only applicable if the model has no inner voids.

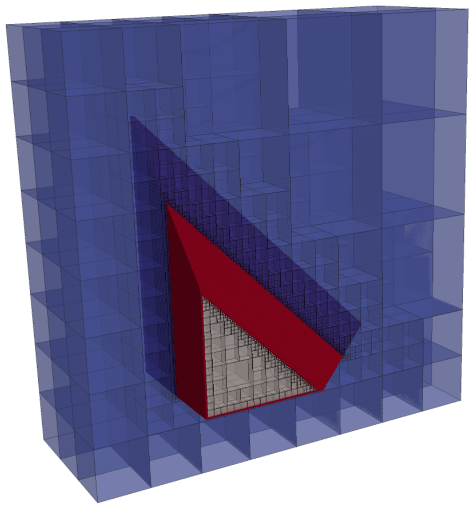
For voids, the remaining cells cannot be considered inside, but additional seed points need to be found inside the domain. An unmarked cell close to the boundary layer opposite to an outer cell is chosen as a

seed cell for the inner domain. Depending on how the inner voids divide the overall domain, it might be necessary to find multiple inner seed points.
Figure 10.4 shows the octree approximation of a simple three-dimensional example that has several common flaws. The size of the opening ε_{gap} allows a maximum subdivision depth of $n_{max} = 7$. Hence, the ratio of the largest gap to the overall size is in the range of:

$$\frac{1}{256} < \frac{\varepsilon_{gap}}{d_{domain}} < \frac{1}{128}\,. \tag{10.9}$$



(a) Flawed triangular B-rep domain.



(b) Octree approximation tree $\widehat{T}_{geo}$.

Figure 10.4: Octree approximation of a flawed embedded tetrahedral domain. The outer domain (blue) is separated by the cut leaves (red) from the inner domain (gray). The subdivision level is $n_{max} = 7$.

Remarks:

- In general, the space-tree for the integration T_{int} of the discontinuous element matrices and the space-tree for the PMC $\widehat{T}_{geo}$ are distinct.
- In our implementation, the approximation tree for the PMC $\hat{T}_{geo}$ is set up on a grid, which is typically (yet not necessarily) coinciding with the finite cell grid. Consequently, $\widehat{T}_{geo}$ is not a single tree, but rather a forest with each grid cell carrying its tree. All trees are connected via the grid. Also, the flood-fill algorithm is applied to the grid. However, one could still assume $\widehat{T}_{geo}$ to be only one tree, which carries on the first level – instead of the standard four (in 2D), or eight (in 3D) children – all the trees of the grid cell. The application on a grid has several advantages. The shape (i.e., aspect ratio) of the leaf cells is decoupled of the domain's shape. Additionally, the regions of refinement are better localized around the actual geometry. This independent grid leads to fewer leaf cells with the same approximation accuracy.

10.2.2 Secondary PMC on cut cells

The space-tree $\widehat{T}_{geo}$ represents the surface only very roughly and step-wise and cannot be used for a precise numerical analysis. In order to improve the representation of the boundary, an additional PMC with ray tracing is carried out on cut leave cells. In flawless regions, this test delivers accurate results. In the vicinity of a flaw, the ray tracing result may vary, depending on the selected direction of the ray, as depicted in Figure 10.5. In order to increase the probability of a correct result, multiple rays are sent out to points in the surrounding inner and outer cells. In the case of an ambiguous result – at least one statement differs from the others –a majority decision determines the point membership. This 'vote' can be wrong with respect to the (in general unknown) flawless model, thus, introducing a modeling error. In a mathematical sense, a 'variational crime' is committed (see, e.g., [113]). However, for small flaws, this error becomes not dominant, since it is restricted to the smallest cut cells and in these only to a part of the integration points.

In addition, a bracketing simulation can provide an upper and lower limit of this modeling error. To this end, the problem is solved twice – once under the assumption that all ambiguous integration points are inside, and once assuming them to be outside of the model (see Example 13.1). With this, it can be ensured that the approximation quality of the method is not corrupted.

Apart from ray tracing also other possibilities for the secondary PMC are conceivable. For instance, a PMC test based on point clouds, as this test is sensitive to different types of flaws, such as wrongly oriented normals or intersections. If the accuracy around flaws is not of primary interest, the number of additional ray tracings can be reduced to only a few, or just one direction, which leads to a significant speed-up but increases the probability of wrong results.

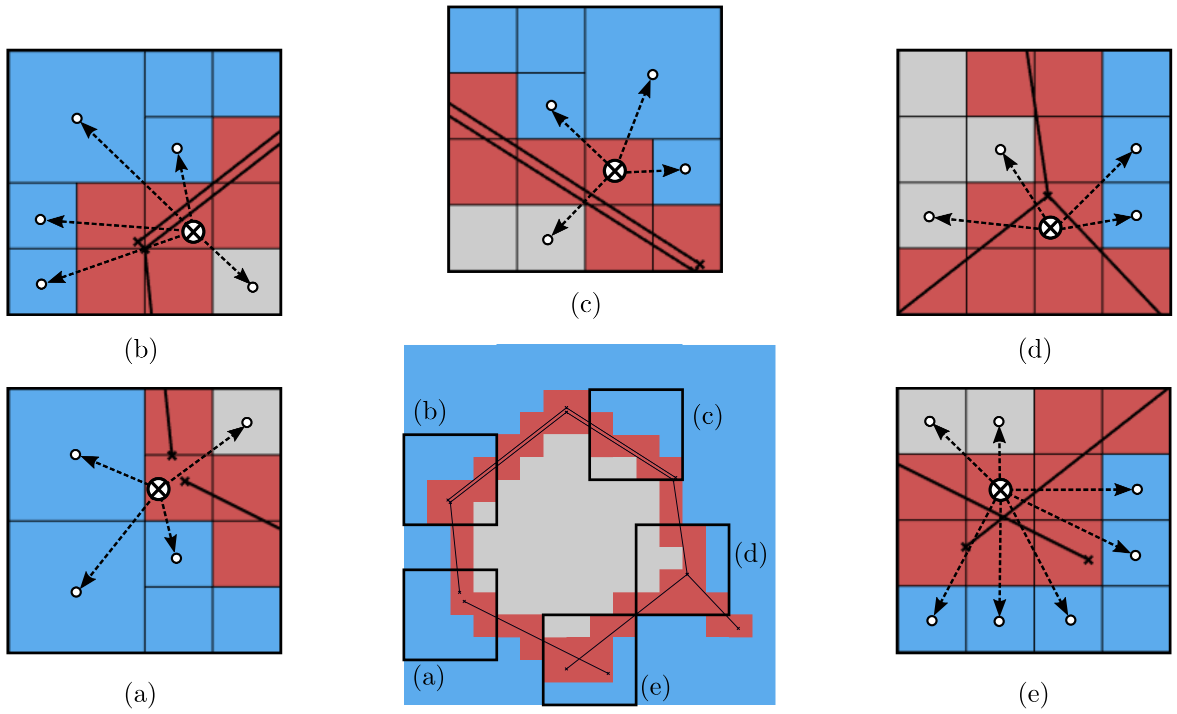


Figure 10.5: Multiple ray tracings at different flaws: (a) gaps, (b) & (c) multiple entities and offsets, (d) artifacts, and (e) intersections.

10.2.3 Boundary conditions for flawed geometries

Non-homogeneous Neumann boundary conditions require a surface discretization without multiple entities or overlaps since such a flawed surface would introduce physically modified boundary conditions, such as additional loads, heat sources, etc. The following automatable method can convert a flawed surface into a surface without multiple entities or overlaps:

1. First, the respective surface is triangulated.
2. Then, the triangles that are cut by the finite cell mesh are subdivided at the intersection points. This division ensures that each triangle can be associated with one finite cell.
3. On each finite cell, a *point cloud* is created from the corner points of the associated triangles.
4. Finally, a finite cell-wise *Delaunay triangulation* is carried out on the respective point clouds.

The resulting finite cell-wise triangular meshes are only used to integrate the Neumann boundary conditions and, consequently, can be independent of each other. The triangulation of the surface might cause an approximation error, compared to an exact curved shape. However, provided a sufficiently fine triangulation, this error is negligible compared to the error introduced by the model's flaws.

10.2.4 Parameter study on the influence of the gap size

As stated in Section 7.1, a flawed B-rep model is, in general, mathematically not valid. Therefore, it is impossible to define an 'error' of the computed approximation with respect to an exact solution since a correct solution does not exist. Yet, to give a feeling for the achievable quality, a simple example is computed, and the approximate solution's internal strain energy is compared to the reference solution.

As for the presented PMC, gaps are the critical flaws, the influence of an increasing gap size ε_{gap} on the internal strain energy – for a cube with the dimensions $1 \times 1 \times 1$ loaded under self-weight – is investigated. The cube is clamped at the bottom and is embedded in $9 \times 9 \times 9$ elements, employing integrated Legendre polynomials of degree $p = 3$. The B-rep model of the cube consists of twelve triangles. One of these triangles is not correctly connected to two of its neighbors, leading to a gap with a characteristic size ε^i_{gap} (see Figure 10.6). The size of the gap limits the maximum subdivision depth n_{max} of the reconstruction tree T_{geo}, meaning that more refinements lead to a non-watertight boundary approximation, as explained in Section 10.2.1.

The domain of the reconstruction tree has the dimension $1.6 \times 1.6 \times 1.6$. The quality of reconstruction does not necessarily only depend on the subdivision depth of the tree. Also the relative position between model and reconstruction tree can play a role, since the smallest cell might, by chance, be congruent with the gap. To study this influence, the origin $\hat{\boldsymbol{X}}_0(\beta)$ of the tree is gradually 'shifted' along a diagonal in space:

$$\hat{\boldsymbol{X}}_0(\beta) = \begin{bmatrix} -0.3 \\ -0.3 \\ -0.3 \end{bmatrix} + \beta \cdot \begin{bmatrix} 0.05 \\ 0.05 \\ 0.05 \end{bmatrix} , \tag{10.10}$$

with $\beta = 0...3$. Figure 10.7 illustrates two different reconstruction trees for different origin positions and different subdivision depths.

Figure 10.8 depicts the influence of the characteristic size of the gap ε_{gap} on the error in the internal energy. The abscissa shows the gap's characteristic size compared to the unit length of the cube in percent. The values correspond to the respective maximum subdivision depths $n^i_{max} = 9...3$ resulting from gap sizes $\varepsilon^i_{gap} = \frac{1.6}{2^{n^i_{max}}}$ from left to right. The ordinate shows the deviation of the internal strain energy U to the reference energy U_{ref} in percent. The reference energy U_{ref} is computed on a flawless

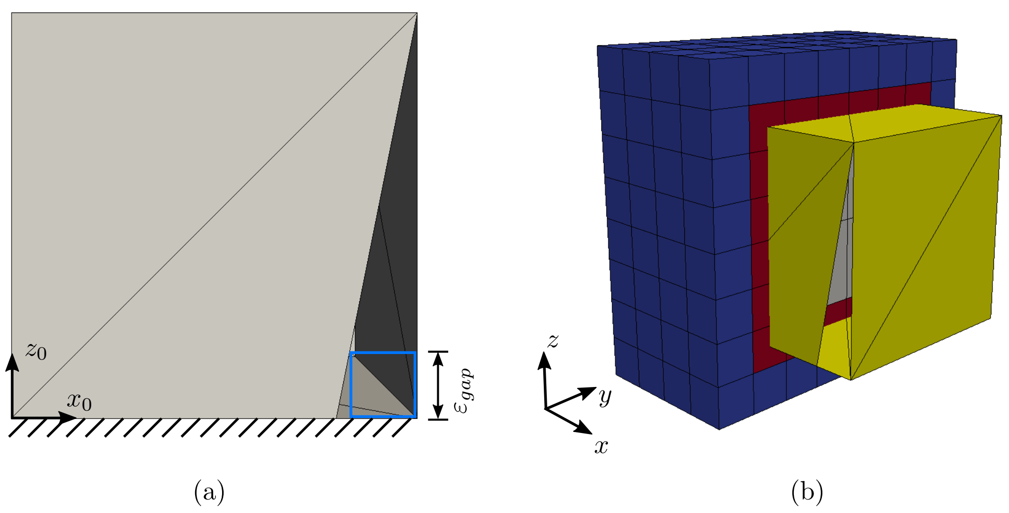


Figure 10.6: Parameter study on a unit cube: (a) Characteristic gap size ε_{gap}, and (b) reconstruction tree on the flawed geometry.

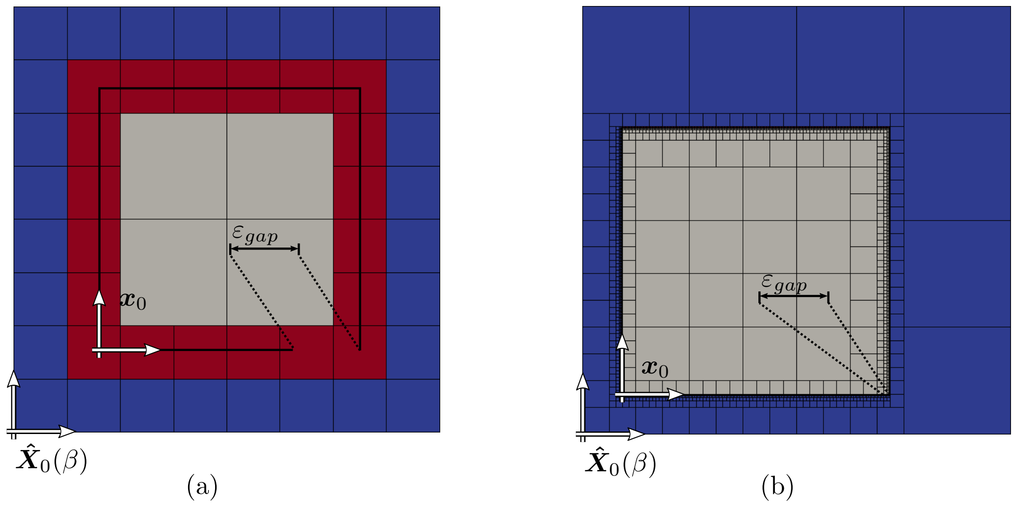


Figure 10.7: Cut through two reconstruction trees for different gap sizes – and thus maximum subdivision depths – and for different origin positions: a) $\varepsilon_{gap} = 0.2$, $n_{max} = 3$ at a shift of $\beta = 0$, and b) $\varepsilon_{gap} = 0.0031$, $n_{max} = 9$ at a shift of $\beta = 3$.

model. Accurate results in energy are obtained even for large gap sizes of up to 20% of the domain length. Figure 10.9 confirms the quality of the solution. It shows the *von Mises stresses* of the reference solutions and approximate solutions for two gap sizes.

Although this study supports the quality of the presented approach, it cannot be guaranteed that the local energy error is limited, in general. The error strongly depends on the complexity of the model and the amount and type of flaws. In addition, the location of the defects is crucial, as well. The influence will be more significant if the flaw lies in a high-stress region.

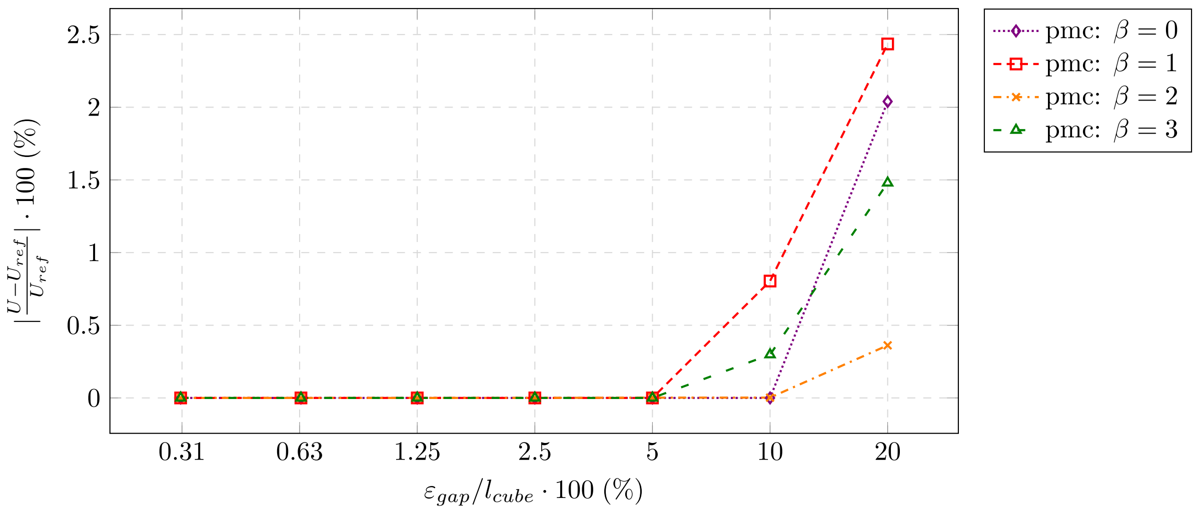
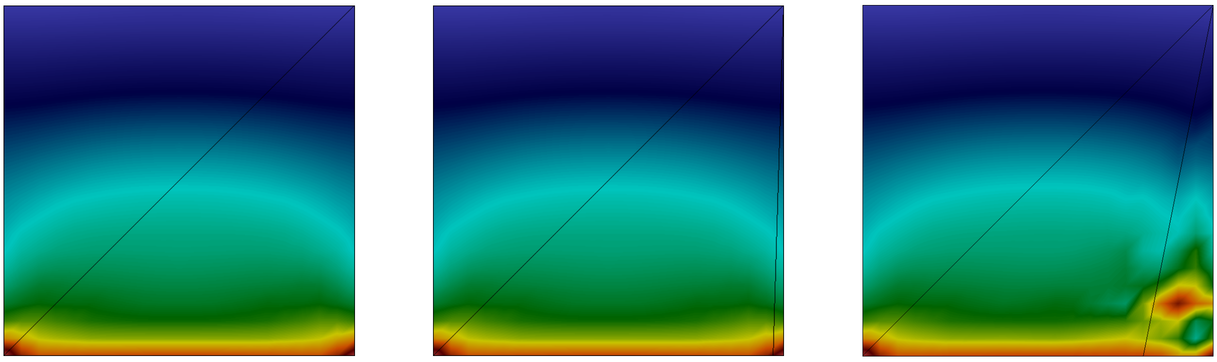


Figure 10.8: Relative deviation of energies depending on the gap size for different positions of the cube.



(a) $\varepsilon_{gap} = 0$, $n_{max} = \infty$ (b) $\varepsilon_{gap} = 0.025$, $n_{max} = 6$ (c) $\varepsilon_{gap} = 0.2$, $n_{max} = 3$

Figure 10.9: Von Mises stresses for the (a) flawless model and gap sizes of (b) $\varepsilon_{gap} = 2.5\%$, and (c) $\varepsilon_{gap} = 20\%$.

Chapter 11

Point membership classification on solid-based procedural/CSG models

In contrast to B-rep models, a point membership classification can be carried out easier, more efficiently – and, in particular, robustly – on a CSG-tree or construction history (tree) of a solid-based procedural model. The following chapter is based on the publication [13]. As explained in Section 7.2.1, the classical CSG-tree is a full binary-tree, where each node has either exactly two or zero child nodes. Thereby, the leaf cells correspond to the primitives, and the bifurcation nodes can be one of the three Boolean operations: intersection, union, difference (see Figure 11.1). In contrast to the classical CSG-tree, CAD systems implement unary operations, as well. Consequently, intermediate nodes still correspond to operations, yet, need not be bifurcations. As an example of a unary operation node, consider 'mirroring', which has the object to be mirrored as its single child.

For a point membership classification, at first, the root node – that corresponds to the final object – is queried. Apart from the trivial case, where the root is a primitive, it is a binary or unary operation node. The request is then forwarded recursively to all children until it reaches the primitives/leaf nodes of all branches. The PMC for the leaves can be very efficient since, for many simple primitives, an analytical test is available (see Section 11.3). The respective point memberships are then combined according to the operations.

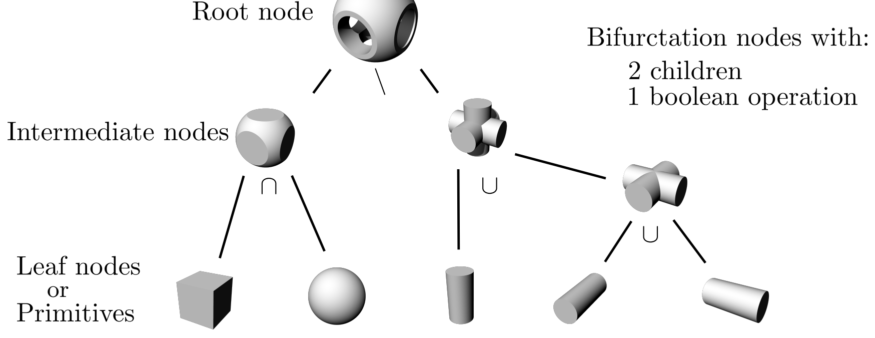


Figure 11.1: Classical CSG-tree consisting of primitives at the leaf nodes and bifurcation nodes at an intermediate or root node where two children are combined with a Boolean operation.

The PMC algorithm can be sped up by exploiting the recursive property at a bifurcation node. For this, the entire branch of the first child is evaluated before the second child is queried. In case of an intersection or a difference, the PMC on the second branch can be omitted, if the PMC on the first branch is already negative. Performing PMC tests with the primitives' bounding boxes of computationally expensive branches yield an additional speed-up.

11.1 Local coordinate systems

For many primitives, a simple PMC is available, yet, only if it is defined in Cartesian coordinates. In general, this is not the case. Consequently, the point of interest $\boldsymbol{P}$ needs to be expressed in terms of the respective local coordinates – which corresponds to a *change of basis*. Another kind of transformation is necessary for sweeps and lofts, where different path-following local basis systems are used to generate different shapes.

11.1.1 Change of basis

Change of basis means that the coordinate system in which a point is defined changes. Naturally, this yields different point coordinates in both systems. In other words, the change of basis describes the mapping between different coordinate systems. In the following, all coordinate systems are assumed to be orthonormal $\boldsymbol{a}_i \cdot \boldsymbol{a}_j = 0$, $\forall i \neq j$ and $\boldsymbol{a}_i \cdot \boldsymbol{a}_i = 1$, since only these bases are needed for the description of CAD geometries[1].

Let $A = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3\}$ and $B = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$ be two coordinate systems with their respective base vectors $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$, and origins $\boldsymbol{A}_0$ and $\boldsymbol{B}_0$ (see Figure 11.2). A global point $\boldsymbol{P}$ (in Cartesian coordinates $\boldsymbol{P} = P_x\boldsymbol{e}_1 + P_y\boldsymbol{e}_2 + P_z\boldsymbol{e}_3$) is denoted with $\boldsymbol{P}^A$ and $\boldsymbol{P}^B$, when expressed in A and B, respectively

$$\boldsymbol{P} = P_x^A\boldsymbol{a}_1 + P_y^A\boldsymbol{a}_2 + P_z^A\boldsymbol{a}_3 = P_x^B\boldsymbol{b}_1 + P_y^B\boldsymbol{b}_2 + P_z^B\boldsymbol{b}_3 \,. \tag{11.1}$$

The mapping between $\boldsymbol{P}^A$ and $\boldsymbol{P}^B$ is defined with the transformation matrices $\boldsymbol{T}^{A\mapsto B}$, where

$$T_{ij}^{A\mapsto B} = \boldsymbol{b}_i \cdot \boldsymbol{a}_j \,, \tag{11.2}$$

and $\boldsymbol{T}^{B\mapsto A}$

$$\boldsymbol{T}^{B\mapsto A} = (\boldsymbol{T}^{A\mapsto B})^T \,. \tag{11.3}$$

The transformation matrix consists of the relative rotation, distortion and scaling between the base vectors $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$. If only orthonormal coordinate systems are considered, this reduces to relative rotations. Not contained in $\boldsymbol{T}$ are the translatory relations, which need to be considered, if the origins $\boldsymbol{A}_0$ and $\boldsymbol{B}_0$ are distinct. For this, the coordinate systems are translated into the Cartesian origin $\boldsymbol{E}_0 = \boldsymbol{0}$. The mapping from A to B reads then as follows

$$\boldsymbol{P}^B = \boldsymbol{T}^{A\mapsto B}\boldsymbol{P}^A + \boldsymbol{T}^{E\mapsto B}(\boldsymbol{A}_0 - \boldsymbol{B}_0) \,, \tag{11.4}$$

with $\boldsymbol{T}^{E\mapsto B}$ being the transformation matrix for the mapping into Cartesian coordinates ($T_{ij}^{E\mapsto B} = \boldsymbol{b}_i \cdot \boldsymbol{e}_j$). In the case of coinciding origins $\boldsymbol{A}_0 = \boldsymbol{B}_0$, Equation (11.4) reduces to a mere transformation of the basis vectors. As can be seen, the translatory transformation is independent of the point $\boldsymbol{P}$ and can thus be pre-computed.

[1] *Remark:* However, the change of basis is not limited to orthonormal coordinate systems.

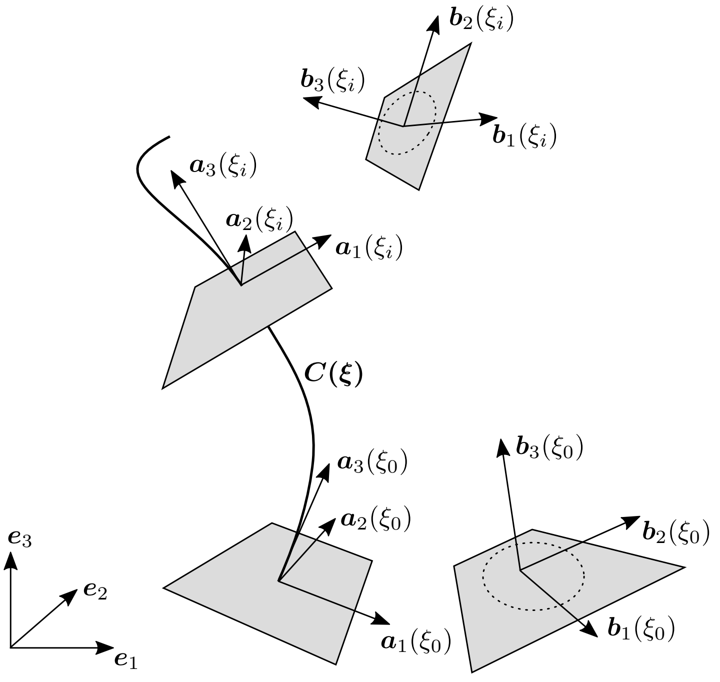

$\boldsymbol{b}_2(\xi_i)$
$\boldsymbol{b}_3(\xi_i)$
$\boldsymbol{b}_1(\xi_i)$
$\boldsymbol{a}_3(\xi_i)$
$\boldsymbol{a}_2(\xi_i)$
$\boldsymbol{a}_1(\xi_i)$
$\boldsymbol{C}(\boldsymbol{\xi})$
$\boldsymbol{a}_3(\xi_0)$
$\boldsymbol{b}_3(\xi_0)$
$\boldsymbol{a}_2(\xi_0)$
$\boldsymbol{b}_2(\xi_0)$
$\boldsymbol{e}_3$
$\boldsymbol{e}_2$
$\boldsymbol{e}_1$
$\boldsymbol{a}_1(\xi_0)$
$\boldsymbol{b}_1(\xi_0)$

Figure 11.2: Fixed relations between different coordinate systems that change along a path.

Often, only a mapping from Cartesian coordinates $E = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\}$ into a local coordinate system $A = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3\}$ is required. In this case, Equation (11.4) simplifies to

$$\boldsymbol{P}^A = \boldsymbol{T}^{E\mapsto A}(\boldsymbol{P} - \boldsymbol{A}_0)\,, \tag{11.5}$$

where $\boldsymbol{P} = \boldsymbol{P}^E$ is the point in Cartesian coordinates. The transformation matrix also simplifies to $\boldsymbol{T}^{E\mapsto A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3]^T$. The inverse operation, i.e., the mapping from the local coordinate system to Cartesian coordinates, reads as follows

$$\boldsymbol{P} = \boldsymbol{T}^{A\mapsto E}\,\boldsymbol{P}^A + \boldsymbol{A}_0\,, \tag{11.6}$$

with $\boldsymbol{T}^{A\mapsto E} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3]$.

11.1.2 Path-following local basis system

Sweeps and lofts are generated by moving a two-dimensional sketch along a sweep path. The relation between sweep path and sketch plane is described by a path-following local coordinate system. For this several possibilities are conceivable: (a) the *Frenet basis*, (b) an *angle preserving basis*, (c) a *parallel, constant basis* and (d) for lofts also an *interpolating basis* (see Figure 11.3). In the following the individual local bases are explained.

Let $A(\xi) = \{\boldsymbol{a}_1(\xi), \boldsymbol{a}_2(\xi), \boldsymbol{a}_3(\xi)\}$ denote the tangential local coordinate system of the path. A is defined in such a way that the base vector in local $z-$direction $\boldsymbol{a}_3$ equals the (normalized) tangent vector of the curve $\boldsymbol{a}_3(\xi) = \alpha\boldsymbol{C}(\xi)$, where the scalar α corresponds to the length of the tangent vector. Let $B(\xi) = \{\boldsymbol{b}_1(\xi), \boldsymbol{b}_2(\xi), \boldsymbol{b}_3(\xi)\}$ be the local coordinate system of the sketch.

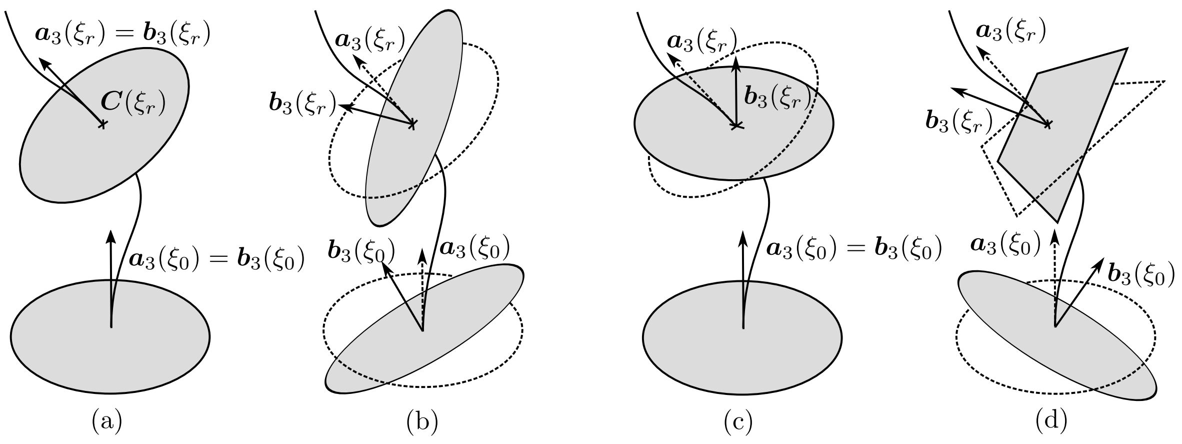


Figure 11.3: Path-following local coordinate systems: (a) Frenet basis, (b) angle preserving basis, (c) parallel, constant basis, and (d) an interpolating basis. The tangential basis A is indicated with the dashed profiles.

Frenet basis (a): The Frenet basis is directly related to the sweep path and, thus, the tangential and the sketch coordinate systems coincide $A(\xi) = B(\xi)$. To set up the basis, two vectors are required. The first vector is given by the tangent vector of the curve. To find a second base vector, the curvature is used. With $\boldsymbol{C}_{,\xi}(\xi)$ and $\boldsymbol{C}_{,\xi\xi}(\xi)$ being the tangent and curvature vectors, i.e., the first and second derivative of $\boldsymbol{C}(\xi)$, the three base vectors are defined as

$$\boldsymbol{a}_1(\xi) = \frac{\boldsymbol{a}_2(\xi) \times \boldsymbol{C}_{,\xi}(\xi)}{\| \boldsymbol{a}_2(\xi) \times \boldsymbol{C}_{,\xi}(\xi) \|_2}\,, \tag{11.7}$$

$$\boldsymbol{a}_2(\xi) = \frac{\boldsymbol{C}_{,\xi}(\xi) \times \boldsymbol{C}_{,\xi\xi}(\xi)}{\| \boldsymbol{C}_{,\xi}(\xi) \times \boldsymbol{C}_{,\xi\xi}(\xi) \|_2}\,, \tag{11.8}$$

$$\boldsymbol{a}_3(\xi) = \frac{\boldsymbol{C}_{,\xi}(\xi)}{\| \boldsymbol{C}_{,\xi}(\xi) \|_2}\,. \tag{11.9}$$

Obviously, the Frenet basis is only sensible if the sweep path is curved, as otherwise, the curvature vector is zero, and consequently, no local basis is spanned. However, in the case of a straight line, the base vector $\boldsymbol{a}_1$ is simply chosen arbitrarily on the normal plane of $\boldsymbol{a}_3$. The remaining vector then reads $\boldsymbol{a}_2 = \boldsymbol{a}_3 \times \boldsymbol{a}_1$. Regarding its application for a sweep or loft, the Frenet basis implies that the sweep path is orthogonal on the initial and ending sketch.

Angle preserving basis (b): The angle preserving coordinate system is applied if the sweep or loft's shape shall follow the curve, but the sweep path is not orthogonal on the initial sketch. For the setup of the local coordinate system $B(\xi)$, the basis vectors $\boldsymbol{b}_i(\xi_0)$ are expressed in terms of $A(\xi_0)$ at the starting point of the sweep path ξ_0. With this, the relative positions between both coordinate systems are stored. Setting up the sketch coordinate system $B(\xi)$ on any curve point $\boldsymbol{C}(\xi)$ involves the following steps:

- At the initial sketch ξ_0, the transformation matrix $\boldsymbol{T}^{B\mapsto A}(\xi_0)$ for the mapping from $B(\xi_0)$ to $A(\xi_0)$ is determined, according to Equation 11.3.
- The sketch base vectors $\boldsymbol{b}_i$ are expressed in terms of the tangent basis (e.g., the Frenet base) $A(\xi_0)$

$$\tilde{\boldsymbol{b}}_i^0 = \boldsymbol{T}^{B\mapsto A}(\xi_0)\boldsymbol{e}_i\,, \tag{11.10}$$

where $\boldsymbol{e}_i$ are the Cartesian base vectors. Actually, $\boldsymbol{e}_i$ are the local base vectors $\boldsymbol{b}_i$ expressed in B. As an orthonormal basis is assumed, they equal $\boldsymbol{e}_i$ in their own coordinate system.

- On a curve point $\boldsymbol{C}(\xi)$ the respective tangential basis $A(\xi)$ (e.g., the Frenet basis) is determined and the transformation matrix $\boldsymbol{T}^{A\mapsto E}(\xi)$ is set up, following Equation (11.6).
- The base vectors of the sketch at ξ are computed in Cartesian coordinates

$$\boldsymbol{b}_i(\xi) = \boldsymbol{T}^{A\mapsto E}(\xi)\tilde{\boldsymbol{b}}_i^0\,, \tag{11.11}$$

which then constitute the local coordinate system $B(\xi) = \{\boldsymbol{b}_1(\xi), \boldsymbol{b}_2(\xi), \boldsymbol{b}_3(\xi)\}$.

Parallel, constant basis (c): A parallel, constant basis remains parallel to the initial sketch plane. Consequently, the basis vectors of the sketch $\boldsymbol{b}_i$ are constant along the sweep path $\boldsymbol{b}_i(\xi) = \boldsymbol{a}_i(\xi_0)$. Only the origin $\boldsymbol{B}_0$ is changing. Regarding the creation of a local basis system, a coplanarity of the tangent vector and initial sketch plane is not critical. In this context, 'coplanarity' means $\exists\,\xi_r$, where $\boldsymbol{a}_3(\xi_r)\cdot\boldsymbol{b}_3(\xi_r) = 0$. However, regarding the generation of sweeps and lofts, this kind of basis system allows non-manifold geometries. For instance, consider a sweep created by moving a sketch in the $x-y-$plane along the $x-$axis. The resulting shape has no extension in $z-$direction and thus forms no volumetric body.

Interpolating basis (d): In the case of lofts, it is possible that the relative position between tangential basis and the local coordinate systems of the initial and final sketches is differently $\boldsymbol{T}^{A\mapsto B}(\xi_0) \neq \boldsymbol{T}^{A\mapsto B}(\xi_{end})$. The general approach is similar to (b). However, the transformation matrices are computed for both, the initial and the final sketch, $\boldsymbol{T}^{B\mapsto A}(\xi_0)$ and $\boldsymbol{T}^{B\mapsto A}(\xi_{end})$. Also the base vectors are expressed terms of the tangential basis at both positions $\tilde{\boldsymbol{b}}_i^0$ and $\tilde{\boldsymbol{b}}_i^{end}$, respectively. To set up a local coordinate system $B(\xi)$ the vectors $\tilde{\boldsymbol{b}}_i^0$ and $\tilde{\boldsymbol{b}}_i^{end}$ are transformed into Cartesian coordinates, yielding two distinct coordinate systems $B^0(\xi) \neq B^{end}(\xi)$. The base vectors for $B(\xi)$ are then (linearly) interpolated from $B^0(\xi)$ and $B^{end}(\xi)$ using the arc-length of the respective position on the loft path.

11.2 PMC for basic operations

In the following, the PMCs for the basic operations are described. Excluded are the extended operations since these operations correspond to individual CSG-trees with adequate primitives and the three Boolean operations (see Section 7.2.2). Let A and B be two volumetric objects, i.e., either primitives or solids of intermediate nodes and $\boldsymbol{P}$ the point which is to be classified. The PMC is defined as follows for a

- **difference**, where B is subtracted from A

$$\boldsymbol{P} \in (A \setminus B) \;\Leftrightarrow\; \boldsymbol{P} \in A \wedge \boldsymbol{P} \notin B\,, \tag{11.12}$$

- for a **union**

$$\boldsymbol{P} \in (A \cup B) \;\Leftrightarrow\; \boldsymbol{P} \in A \vee \boldsymbol{P} \in B\,, \tag{11.13}$$

- for an **intersection**

$$\boldsymbol{P} \in (A \cap B) \;\Leftrightarrow\; \boldsymbol{P} \in A \wedge \boldsymbol{P} \in B\,, \tag{11.14}$$

- for a **negation**

$$\boldsymbol{P} \in (\neg A) \;\Leftrightarrow\; \boldsymbol{P} \notin A\,, \tag{11.15}$$

- for a **mirror** operation, in which an object A mirrored at a mirror plane M, yielding an object A_{mirror} (see Figure 11.4).

  $$\boldsymbol{P} \in A_{mirror} \;\Leftrightarrow\; \boldsymbol{P} \mapsto \tilde{\boldsymbol{P}} \in A\,, \tag{11.16}$$

  with $\tilde{\boldsymbol{P}}$ being the mirrored point. Let $M : \boldsymbol{n} \cdot (\boldsymbol{x} - \boldsymbol{x}_0)$ be the mirror plane, defined by its origin $\boldsymbol{x}_0$ and normal vector $\boldsymbol{n}$ pointing to the side on which the mirrored object lies. The mirror plane spans a family of local basis systems, with $\boldsymbol{a}_3 = \boldsymbol{n}$. The other basis vectors can be chosen arbitrarily, as long as $\boldsymbol{a}_1 \times \boldsymbol{a}_2 = \alpha\boldsymbol{a}_3$, $\alpha > 0$. To determine $\tilde{\boldsymbol{P}}$ the following steps need to be carried out:

  - Check whether $\boldsymbol{P}$ is on the positive side of M: $(\boldsymbol{P} - \boldsymbol{x}_0) \cdot \boldsymbol{n} \overset{!}{>} 0$, else $\boldsymbol{P}$ is anyway not in A_{mirror}.
  - Map $\boldsymbol{P}$ into the local coordinate system of M: $\boldsymbol{P} \mapsto \boldsymbol{Q}$, where $\boldsymbol{Q} = [q_x, q_y, q_z]^T$ is the mapped point, according to Equation (11.5). q_z corresponds to the distance from $\boldsymbol{P}$ to M.
  - Flip the local $z-$coordinate, so that $\tilde{\boldsymbol{Q}}^T = [q_x, q_y, -q_z]$.
  - Map $\tilde{\boldsymbol{Q}}$ back from the local coordinate system of M into Cartesian coordinates: $\tilde{\boldsymbol{Q}} \mapsto \tilde{\boldsymbol{P}}$, according to Equation (11.6).

  The mirror operation can be implemented inclusively, or exclusively – meaning, the original object is either conserved, or deleted.

- For an **orthogonal pattern**, where the object A is repeated n, m, o times in $x-$,$y-$, and/or $z-$direction, yielding a grid of $A_{i,j,k} \in \{A_{1,1,1}, ..., A_{n,m,o}\}$ (see Figure 11.4). For the PMC, the point is translated to the unit cell U (e.g., the bounding box), containing the original/initial object $A_{1,1,1}$, on which the PMC is then carried out

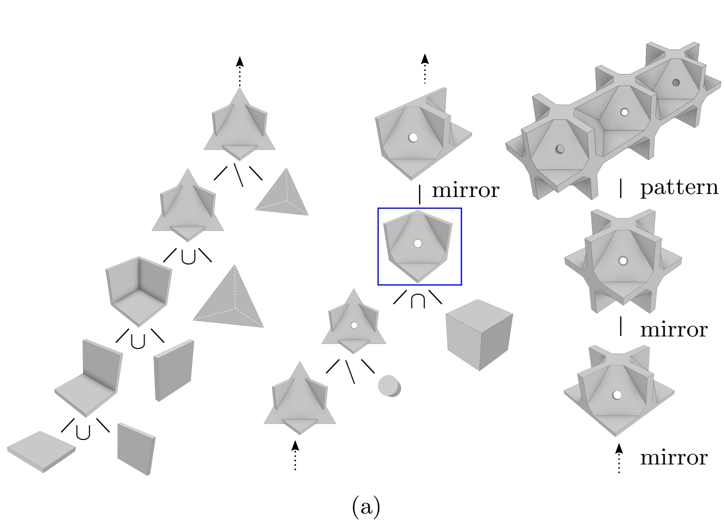
  $$\boldsymbol{P} \in A_{i,j,j} \;\Leftrightarrow\; \boldsymbol{P} \mapsto \tilde{\boldsymbol{P}} \in A_{1,1,1}\,, \tag{11.17}$$

  where $\tilde{\boldsymbol{P}}$ is the translated point. Exemplary for the $x-$direction, with a side length u_x and an origin u_0 of the unit cell, the translation reads

  $$\tilde{x} = (x - u_0) - \min\left(\left\lfloor \frac{x - u_0}{u_x} \right\rfloor, n-1\right) u_x + u_0\,,$$

  where $\lfloor ... \rfloor$ corresponds to the floor function. An offset $\mathrm{d}x$ between the individual objects can easily be realized by extending the unit cells with $\frac{\mathrm{d}x}{2}$ in both directions.

- For **other patterns**, where the pattern aligns with a circular or a free-form path. The general approach is the same as for the orthogonal pattern. At first, the point is translated into the original unit cell, according to the respective path description. Then, the PMC is carried out on the initial object.



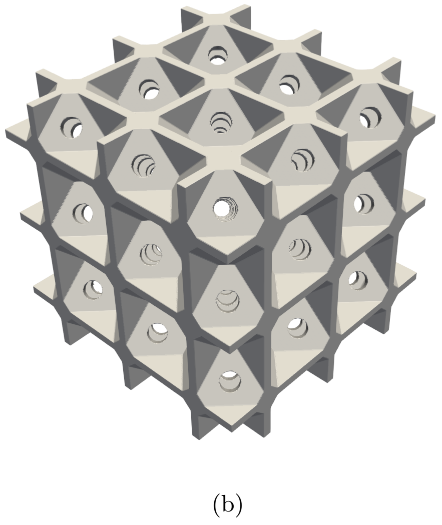


Figure 11.4: Periodic domain implementing the most common operations: (a) CSG-tree with the classical binary Boolean operations up to the marked object (denoted by the blue rectangle), and subsequently three unary, inclusive mirror operations and one unary pattern operation. (b) Surface reconstruction with the marching cubes algorithm, thus, using merely the PMC on the CSG-tree.

11.3 PMC for simple primitives

For classical primitives, a simple analytical description is available, which allows a very efficient and accurate PMC. In the following, a primitive $\boldsymbol{B}_i$ is considered to be created axis-aligned on the $x-y$ plane. Additionally, it is assumed that each primitive is a closed body, i.e., the boundary is included in the body. Let $\boldsymbol{P} = [x, y, z]^T$ be the point of interest to be determined. The PMC reads as follows for a

- **sphere** with center point $\boldsymbol{C}_{\text{sphere}}$ and radius r_0

$$\boldsymbol{P} \in \boldsymbol{B}_{\text{sphere}} \;\Leftrightarrow\; ||(\boldsymbol{P} - \boldsymbol{C}_{\text{sphere}})||_2 \leq r_0\,, \tag{11.18}$$

- for a **cuboid** defined by its diagonal two corner points $\boldsymbol{P}_{\text{start}} = [x_s, y_s, z_s]^T$ and $\boldsymbol{P}_{\text{end}} = [x_e, y_e, z_e]^T$

$$\boldsymbol{P} \in \boldsymbol{B}_{\text{cuboid}} \;\Leftrightarrow\; x \in [x_s, x_e] \wedge y \in [y_s, y_e] \wedge z \in [z_s, z_e]\,, \tag{11.19}$$

- for a **cylinder** defined by its center point $\boldsymbol{C}_{\text{cylinder}} = [x_c, y_c, z_c(\overset{!}{=} 0)]^T$, radius r_0, and height h_0

$$\boldsymbol{P} \in \boldsymbol{B}_{\text{cylinder}} \;\Leftrightarrow\; ||(\tilde{\boldsymbol{P}} - \boldsymbol{C}_{\text{cylinder}})||_2 \leq r_0 \wedge z \in [0, h_0]\,, \tag{11.20}$$

  where point $\tilde{\boldsymbol{P}} = [x, y, 0]^T$ is the projection of point $\boldsymbol{P}$ onto the $x-y$ plane,

- for a **cone frustum** with the same set up as for the cylinder, whose bottom and tip circles are concentric with radii r_0 and r_1

$$\boldsymbol{P} \in \boldsymbol{B}_{\text{cone}} \;\Leftrightarrow\; ||(\tilde{\boldsymbol{P}} - \boldsymbol{C}_{\text{cone}})||_2 \leq r(z) \wedge z \in [0, h_0]\,, \tag{11.21}$$

with

$$r(z) = \frac{r_1 - r_0}{h_0} z + r_0 \,. \tag{11.22}$$

A radius $r_1 = 0$ corresponds to a complete cone.

- For a rectangular **pyramid frustum** with a set up similar to the cone frustum. The rectangular bounding box at the bottom $[x_{s0}, x_{e0}]$, $[y_{s0}, y_{e0}]$ and on the top $[x_{s1}, x_{e1}]$, $[y_{s1}, y_{e1}]$ have the same center point.

$$\boldsymbol{P} \in \boldsymbol{B}_{\text{pyramid}} \;\Leftrightarrow\; x \in [x_s(z), x_e(z)] \wedge y \in [y_s(z), y_e(z)] \wedge z \in [z_c, z_c + h_0] \,, \tag{11.23}$$

with

$$x_s(z) = \frac{x_{s1} - x_{s0}}{h_0} z + x_{s0} \,. \tag{11.24}$$

The other intermediate corner coordinates $x_e(z), y_s(z), y_e(z)$ are computed accordingly. If $x_{s1} = x_{e1}$ and $y_{s1} = y_{e1}$ the pyramid frustum becomes a complete pyramid.

- And for a **tetrahedron** defined by its four corner points $\boldsymbol{P}_1$ to $\boldsymbol{P}_4$. The PMC can be carried out easily by mapping the point into barycentric coordinates. With $\boldsymbol{v}_{ij} = \boldsymbol{P}_i - \boldsymbol{P}_j$ being the edge vectors of the tetrahedron, the transformation matrix is defined as $\boldsymbol{T} = [\boldsymbol{v}_{14}, \boldsymbol{v}_{24}, \boldsymbol{v}_{34}]$. The mapped point in local (barycentric) coordinates $\boldsymbol{Q}^T = [q_1, q_2, q_3]$ is determined according to Equation (11.5), with $\boldsymbol{A}_0 = \boldsymbol{P_4}$. The forth barycentric coordinate is defined as $q_4 = 1 - (q_1 + q_2 + q_3)$.

$$\boldsymbol{P} \in \boldsymbol{B}_{\text{tetrahedron}} \;\Leftrightarrow\; \boldsymbol{P} \mapsto \boldsymbol{Q} \,,\; q_i \in [0,1],\; i = \{1, ..., 4\} \,. \tag{11.25}$$

Also for other primitives — such as wedges or tori – fast analytical solutions are available. In general, primitives are not constructed aligned to the $x - y$ plane. Thus, a change of basis needs to be carried out according to Section 11.1.1.

Remark: The change of basis is also valid for non-normal, skewed basis vectors. However, the herein defined PMC tests for the individual primitives are only defined for orthonormal coordinate systems.

11.4 PMC for extended primitives

In the case of sweeps or lofts (see Section 7.2.2), the point membership classification is more complex, as, in general, no analytical test is available. Nevertheless, it is possible to perform a fast and reliable test on these geometries as well. The basic idea is to reduce the dimension of the problem, taking advantage of the fact that extended primitives are constructed by moving two-dimensional sketches along a curve, i.e., the sweep/loft path. For this, the sketches are considered to be planar, which corresponds to the options available in common CSG/procedural modeling tools – such as Autodesk Inventor® or Siemens NX®[2]. Figure 11.5 illustrates the steps for the PMC for sweeps and lofts. For this, let $A(\xi) = \{\boldsymbol{a}_1(\xi), \boldsymbol{a}_2(\xi), \boldsymbol{a}_3(\xi)\}$ denote the tangential local coordinate system and $B(\xi) = \{\boldsymbol{b}_1(\xi), \boldsymbol{b}_2(\xi), \boldsymbol{b}_3(\xi)\}$ the local sketch coordinate system. The PMC consists of the following basic steps:

- For the point of interest $\boldsymbol{P}$ the *respective curve point* $\boldsymbol{C}(\xi_r)$ on the sweep path is found according to the applied path-following coordinate system (see Section 11.1.2).

[2]Primitives with non-planar surfaces are provided for instance by the V-rep framework with the *ruled body* (see Section 7.3.1).

- At $\boldsymbol{C}(\xi_r)$ the local basis of the sketch B is set up and $\boldsymbol{P}$ is mapped into B, according to Equation (11.5). This yields the mapped point $\boldsymbol{Q}$ on the sketch.
- The PMC is carried out in two dimensions on the closed sketch profile using any PMC test for two-dimensional B-reps, as e.g., presented in Section 10.1.

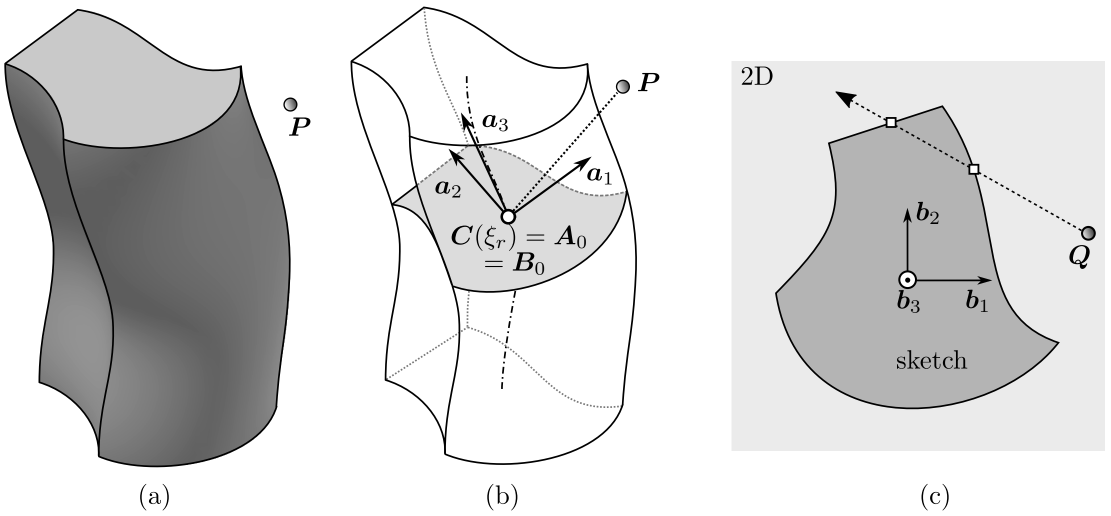


Figure 11.5: Point membership classification for (a) a sweep with Frenet basis as path-following coordinate system, hence $\boldsymbol{a}_i = \boldsymbol{b}_i$ (Note: in general $\boldsymbol{a}_i \neq \boldsymbol{b}_i$). First, the respective curve point $\boldsymbol{C}(\xi_r)$ is found. Then, at ξ_r the tangential basis system $A(\xi_r)$ and subsequently the sketch basis system $B(\xi_r)$ are set up and $\boldsymbol{P}$ is mapped onto the sketch, yielding $\boldsymbol{Q}$. Finally, the PMC is carried out in two dimension with the sketch profile. Here indicated with ray tracing.

11.4.1 Determination of the respective curve point

Depending on the definition of the path-following local basis, the *respective curve point* $\boldsymbol{C}(\xi_r)$ is determined differently. In the case of a tangential coordinate system, such as the Frenet base, the curve point $\boldsymbol{C}(\xi_r)$ corresponds to the closest point on the sweep path and, thus, can be determined with the standard iterative inverse mapping, according to Equation (6.13).

In the case of another coordinate system (see Section 11.1.2), $\boldsymbol{C}(\xi_r)$ can be found with the general iterative approach which is described in the following. For this, the condition is used that the $z-$coordinate of a mapped point $\boldsymbol{Q}$ must be zero $q_z(\xi_r) \overset{!}{=} 0$. Figure 11.6 illustrates the procedure to determine $\boldsymbol{C}(\xi_r)$ with Newton iteration. Since, in general, an analytical derivative for $q_z(\xi)$ is not available, $\dot{q}_z(\xi)$ is approximated with finite differences. For this let $\Delta\xi$ be a sufficiently small parameter increment for the finite differences. The following steps yield the respective curve point $\boldsymbol{C}(\xi_r)$ and the corresponding local coordinate systems $A(\xi_r)$ and $B(\xi_r)$:

- Find an appropriate initial value ξ_r^0, e.g., at the closest point.
- Start the Newton iteration, with $q_z(\xi) \overset{!}{=} 0$ as termination criterion.

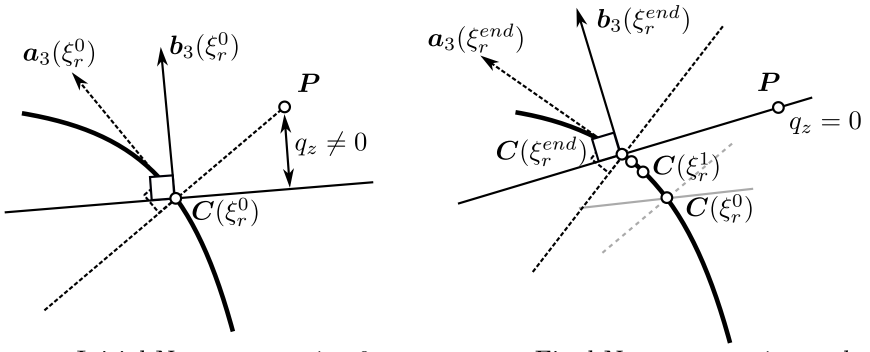
- At step j, compute for ξ_r^j the local tangential basis system $A(\xi_r^j)$ and based on this, the sketch basis system $B(\xi_r^j)$, following Section 11.1.2. Map the point $\boldsymbol{P}$ into $B(\xi_r^j)$, yielding the mapped point $\boldsymbol{Q}^j$.
- Compute for $\xi_r^j + \Delta\xi$ the local tangential basis system $A(\xi_r^j + \Delta\xi)$ and based on this, the sketch basis system $B(\xi_r^j + \Delta\xi)$. Map the point $\boldsymbol{P}$ into $B(\xi_r^j + \Delta\xi)$, yielding the mapped point $\boldsymbol{Q}^{j,\Delta\xi}$.
- Evaluate the finite difference

$$\dot{q}_z^j \approx \frac{q_z^{j,\Delta\xi} - q_z^j}{\Delta\xi}\,, \tag{11.26}$$

  and update the path parameter

$$\xi_r^{j+1} = \xi_r^j - \frac{q_z^j}{\dot{q}_z^j}\,. \tag{11.27}$$

- Continue until $q_z(\xi_r^{end}) = 0$.


$\boldsymbol{a}_3(\xi_r^0)$
$\boldsymbol{b}_3(\xi_r^0)$
$\boldsymbol{P}$
$q_z \neq 0$
$\boldsymbol{C}(\xi_r^0)$
$\boldsymbol{a}_3(\xi_r^{end})$
$\boldsymbol{b}_3(\xi_r^{end})$
$\boldsymbol{P}$
$q_z = 0$
$\boldsymbol{C}(\xi_r^{end})$
$\boldsymbol{C}(\xi_r^1)$
$\boldsymbol{C}(\xi_r^0)$

Initial Newton step $j = 0$ Final Newton step $j = end$

Figure 11.6: Iterative determination of the respective curve point $\boldsymbol{C}(\xi_r)$, exemplary for the angle preserving basis. The path parameter ξ^j is updated until the $z-$coordinate of the mapped point $\boldsymbol{Q}^j$ in the sketch basis $B(\xi_r^j)$ is zero: $q_z^j = 0$.

It is possible that $q_z(\xi)$ has multiple roots, as depicted in Figure 11.7. Among the corresponding curve points $\boldsymbol{C}(\xi_i)$, the one with the smallest distance to $\boldsymbol{P}$ is the respective curve point $\boldsymbol{C}(\xi_r)$. For the unlikely case that multiple curve points $\boldsymbol{C}(\xi_i)$ are equidistant, the two-dimensional PMC needs to be carried out on all corresponding sketches. The point is inside if at least one PMC returns 'inside'[3].

11.4.2 Two-dimensional PMC for sweeps and lofts

For sweeps, the sketch profile is not changing along the sweep path, and hence the PMC can be carried out on the initial sketch S_0, e.g., with ray tracing. For a robust PMC for two-dimensional spline contours based on ray tracing refer to [13].

$$\boldsymbol{P} \in B_{\text{Sweep}} \;\Leftrightarrow\; \boldsymbol{Q} \in S_0\,. \tag{11.28}$$

[3]Note: If not all PMCs are either inside or outside, this corresponds to a self-intersecting body. As the PMC still delivers the correct result, a self-intersection is, from a simulation point of view, not critical.

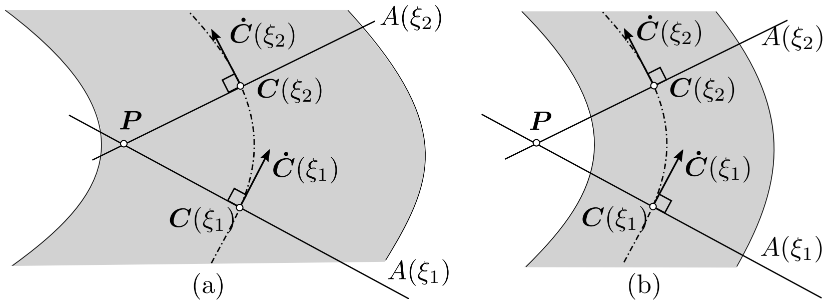


Figure 11.7: Multiple roots of $q_z(\xi)$ can yield different curve points $\boldsymbol{C}(\xi_i)$ with the same distance to $\boldsymbol{P}$.

Lofts are slightly more involved, since the sketch profile is changing. In contrast to sweeps, for lofts the PMC is carried out on the initial and final sketch. Additionally, the distances to the closest points on the initial d_0 and final sketch profiles d_{end} are required. This narrows the choice for the two-dimensional PMC to a test which also provides distances. In the end, it comes down to finding the closest points on each line segment of the sketch profile curves. For the case that the mapped point $\boldsymbol{Q}$ lies inside one sketch and outside the other, the distance to the intermediate profile line d_r is interpolated using the arclength $s(\xi_r)$ at the respective curve point $\boldsymbol{C}(\xi_r)$ (see Figure 11.8). Using a linear interpolation, the PMC for lofts reads

$$\boldsymbol{P} \in B_{\text{Loft}} \Leftrightarrow d_r \geq 0\,, \tag{11.29}$$

with

$$d_r = \frac{(d_{end} - d_0)}{s(\xi_{end})} s(\xi_r) + d_0\,. \tag{11.30}$$

The distance to the point outside is set negative. Consequently, a negative d_r indicates that the point is outside. The arc-length is determined via integration along the sweep path

$$s(\xi_i) = \int_{\xi_0}^{\xi_i} \parallel \boldsymbol{C}_{,\xi}(\xi) \parallel_2 \mathrm{d}\xi\,. \tag{11.31}$$

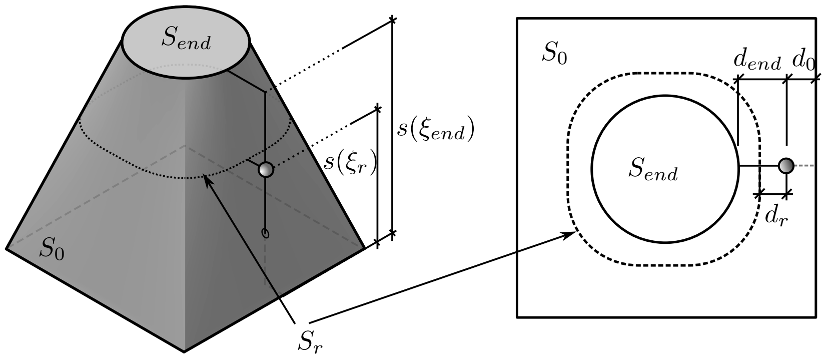


Figure 11.8: Point membership classification for lofts.

Chapter 12

Direct simulation of V-models

As already explained in Section 9.2, the V-rep framework was developed to overcome the shortcomings of the classical CAD representations regarding the modeling and description of functionally graded materials. For a numerical simulation with the FMC, V-models provide, apart from a PMC, also the relevant material properties. The following chapter follows the publication [15].

12.1 PMC for V-models

The V-rep framework already offers a fast and robust PMC, based on inverse mapping with Newton iteration, as described in Section 6.5. Since V-models consist, in general, of several non-intersecting V-cells ν_C^i, the test needs to be carried out on each V-cell. As explained in Section 7.3.1, V-rep primitives are regular by construction. Consequently, the mapping on each V-cell

$$f : \boldsymbol{\xi} \mapsto \nu_C^i(\boldsymbol{\xi})\,, \tag{12.1}$$

is bijective, which guarantees that the result of the inverse mapping

$$f^{-1} : \boldsymbol{P} = \nu_C^i(\boldsymbol{\xi}) \mapsto \boldsymbol{\xi}\,, \tag{12.2}$$

is unique, if $\boldsymbol{P} \in \nu_C^i$. Obviously, otherwise no $\boldsymbol{\xi}$ is found.

The number of required Newton-Raphson iterations for the inverse mapping can be substantially decreased, providing a good guess as an initial value. This property is efficiently exploited by the finite cell method as, due to the Cartesian grid-based data structure, consecutive integration points are often geometrically adjacent. Therefore, each V-cell's last inner point is cached and used as an initial guess for the next query. Like the PMC on CSG trees, an additional speed-up can be achieved with a preceding point inclusion test on the bounding boxes of the respective V-cells.

Since the underlying Irit library [77] offers already a robust point inclusion test, the extension of the FCM to V-models is straight-forward. It is noteworthy that – in contrast to IGA – trimmed splines and non-coinciding spline parameters at adjacent faces, require no special treatment since the adaptive quadrature rules automatically recover the actual shape of the geometry.

12.2 Multi-material FGM with the FCM

V-rep represents a multi-material FGM distribution inside a volume – and ultimately in the V-cells – with additional material-dimensions/coordinates of the corresponding control points (see Section 9.2.1).

In the context of a simulation with the finite cell method, this simply implies that the PMC yields – apart from the inside, outside statement – also the material properties at the respective location, according to Equation (9.2). Naturally, these material values are then used to set up the position-dependent material matrix of the respective constitutive equation (see Section 1.5). For instance, Example 15.2 requires the material matrix for isotropic, linear elasticity

$$\boldsymbol{C}_{lin.el.}(\boldsymbol{\xi}) = \begin{bmatrix} 2\mu(\boldsymbol{\xi}) + \lambda(\boldsymbol{\xi}) & \lambda(\boldsymbol{\xi}) & \lambda(\boldsymbol{\xi}) & 0 & 0 & 0 \\ \lambda(\boldsymbol{\xi}) & 2\mu(\boldsymbol{\xi}) + \lambda(\boldsymbol{\xi}) & \lambda(\boldsymbol{\xi}) & 0 & 0 & 0 \\ \lambda(\boldsymbol{\xi}) & \lambda(\boldsymbol{\xi}) & 2\mu(\boldsymbol{\xi}) + \lambda(\boldsymbol{\xi}) & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu(\boldsymbol{\xi}) & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu(\boldsymbol{\xi}) & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu(\boldsymbol{\xi}) \end{bmatrix}, \qquad (12.3)$$

and heat transfer

$$\boldsymbol{C}_{heat}(\boldsymbol{\xi}) = \begin{bmatrix} \kappa(\boldsymbol{\xi}) & 0 & 0 \\ 0 & \kappa(\boldsymbol{\xi}) & 0 \\ 0 & 0 & \kappa(\boldsymbol{\xi}) \end{bmatrix}. \qquad (12.4)$$

12.3 Analyzing large-scale microstructures using homogenization

The V-rep framework allows the representation of FGM also in terms of a gradually changing microstructures (see Section 9.2.2). A microstructure is again a V-model and thus consists of V-cells. Consequently, the numerical simulation is carried out as described before. However, to obtain reliable results, a fine numerical resolution of the detailed geometrical features is required. Thus, large microstructures might only be computable on high-performance computers. Here, homogenization offers a remedy.

The basic idea of homogenization is to represent the entire domain with a homogeneous material tensor. This material tensor is determined so that the overall material behavior of the homogeneous domain and the microstructure is – at least on the macro scale – equivalent. For this, the existence of a *representative volume element* (RVE) ω^{RVE} is assumed (see Figure 12.1). The RVE is a microstructural domain which is sufficiently large to represent the macroscopic behavior and – at the same time – as small as possible to limit the computational cost [114]. The material tensor is then determined for the RVE by solving several boundary value problems with varying boundary conditions. Thereby, the applied boundary conditions depend on the particular problem. In the present case of recurring microstructures, periodic boundary conditions provide the best results. Using the *Hill conditions*, the material properties are transferred from the microscopic to the macroscopic scale [115]. For a detailed explanation of homogenization with the finite cell method, refer to [116].
In the case of a random structure – in which no single general RVE can be found or for non-linear computations – the homogeneous material behavior needs to be solved in an FE^2 sense [117]. Here, on each integration point in the macroscopic scale, a homogenization simulation on the microscale is carried out.

Functionally graded microstructures – as provided by the V-rep framework – lay in-between strictly periodic and entirely random structures. Since the parameter-based construction plan of the FGM is known apriori, it is sufficient to compute the effective material tensors $\boldsymbol{C}_i^*$ for several 'representative' RVEs (see Figure 12.2). At any realization in-between, the material is then interpolated from corresponding adjacent representative tensors $\boldsymbol{C}_i^*$.

For the microstructures, considered in this thesis, the RVEs correspond to the constituting unit tiles, which can have different properties, for instance, a stiffer direction, a rotation around some axis,

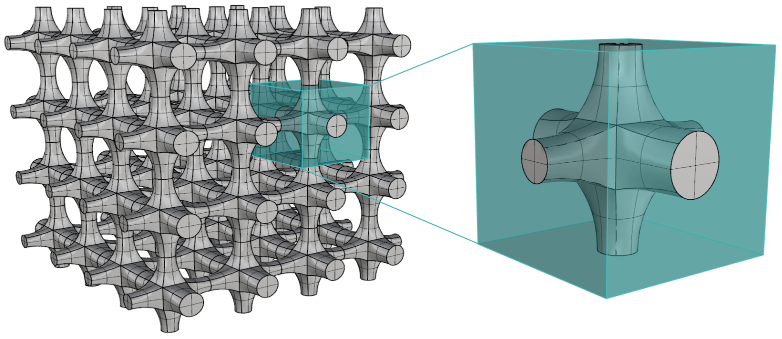


Figure 12.1: Representative volume element of a periodic microstructure. For the homogenization the effective material tensor for the corresponding volume part (indicated by the blue box) is computed.

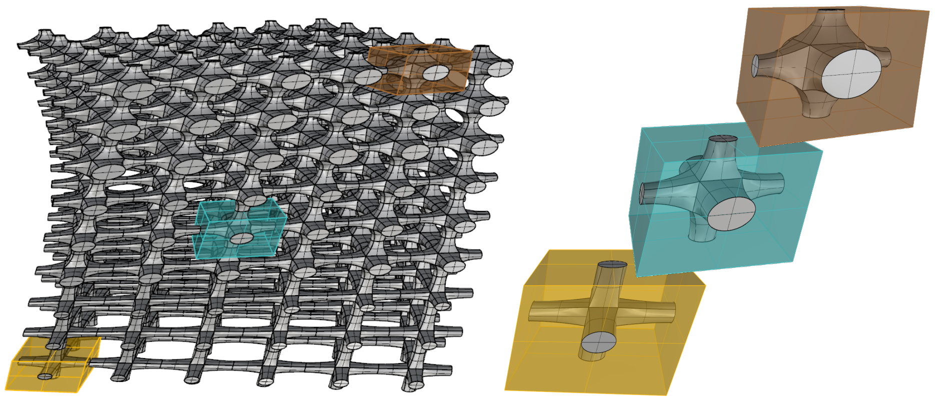


Figure 12.2: Continuously changing microstructure with different representative volume elements. The material properties in-between can be interpolated.

or a material composition. All these properties are defined using suitable construction parameters. The following approach then allows the efficient computation of large-scale functionally graded microstructures with the FCM:

- Using the microstructure's parametric description, several representative unit tiles are selected in different configurations (see Example 15.4).
- For the unit tiles, the effective material tensors $\boldsymbol{C}^*_{Ti}$ are computed with a combination of numerical homogenization and the finite cell method [116] and stored in a look-up table (see Example 15.4).
- During the simulation of a large-scale FGM microstructure, the effective material tensor at each integration point is determined by an interpolation of the values from the look-up table (see Example 15.5).

Based on the microstructure in Example 15.3, this approach is illustrated in the Examples 15.4 and 15.5.

Part IV

Applications

Chapter 13

Applications of flawed B-rep models

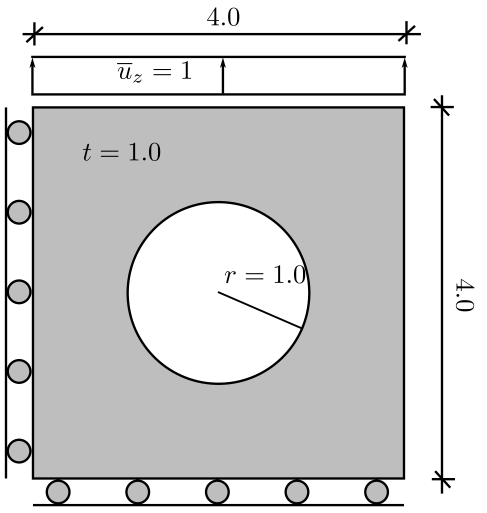
In the following, three examples illustrate the accuracy and robustness of the PMC for flawed B-rep models, which was presented in Section 10.2. The first simple example of a plate with a hole serves as verification, whereas the second and third example prove the applicability for complex defective CAD models. For the complex screw in the second example, a flawless reference model is available. The engine bracket in the last example was taken directly from engineering practice (see Figure 7.1), and thus, no flawless reference model exists. The model is a perfect example for a real-word flawed geometry, which suffers from CAD exchange shortcomings.

13.1 Flawed plate with a hole

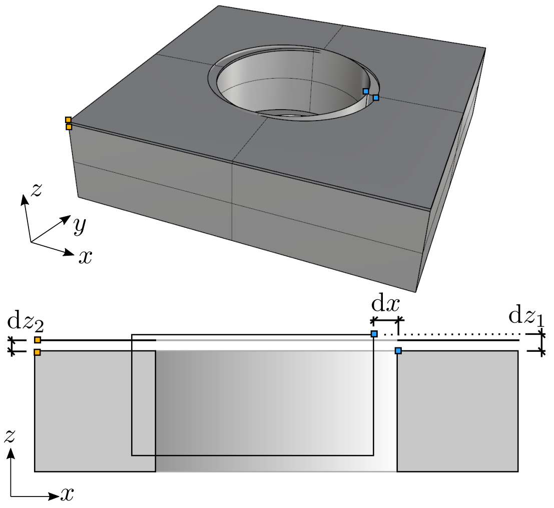
As a standard benchmark for three-dimensional problems, the thick-walled plate with four circular holes is chosen [47]. Using symmetric Dirichlet boundary conditions, only a quarter of the original domain needs to be computed (see Figure 13.1(a)). As material properties, a Young's modulus of $E = 10000$ and a Poisson's ratio of $\nu = 0.3$ are chosen. On the top surface, a prescribed deflection of $\overline{u}_z = 1.0$ is applied in $z-$direction. The dimensions of the model are $b = h = 4$ and $t = r = 1$. The computational domain is slightly larger ($\pm\, 0.2$ in each direction) and discretized with $11\times11\times3$ finite cells using integrated Legendre polynomials. The integration of the system matrices uses moment-fitting with $2\,(p+1)$ integration points in each direction. The moment-fitting weights are determined with the standard octree partitioning, using a maximum subdivision depth $k = 4$ and $p + 1$ integration points in each direction.

To show the robustness and accuracy of the proposed method, nine convergence studies on different flawed models are carried out and compared to a flawless model's reference solution. For this, four different flaws – namely intersection, gap, multiple surfaces, and offset – are applied and incrementally increased, leading to nine different flaw combinations, as illustrated in Table 13.1. More precisely, the sizes of the flaws are increased in three steps from small over medium to large. In each step, the cylinder jacket of the inner hole is shifted by $i \cdot \mathrm{d}x = i \cdot 0.03$ in (negative) $x-$direction and $i \cdot \mathrm{d}z_1 = i \cdot 0.015$ in $z-$direction, where $i \in \{1, 2, 3\}$ is the respective enlargement step.

The convergence studies on the resulting models illustrate the influence of gaps and intersections. Additionally, on each step, two further convergence studies are carried out to demonstrate the influence of multiple entities and offsets. To this end, the top surface is doubled, and subsequently shifted by $i \cdot \mathrm{d}z_2 = i \cdot 0.0075$ leading to an offset (see Figure 13.1(b)). The flaw sizes are chosen such that the maximum subdivision depth of the reconstruction tree $\widehat{T}_{geo}$ is decreasing from $n_{max} = 4$ for the small flaws, over $n_{max} = 3$ for the medium flaws to $n_{max} = 2$ for the large flaws. Note, that reconstruction trees are set up on each finite cell (see Remarks in 10.2.1).



(a) Dimensions and boundary conditions.



(b) Step-wise increase of the dimension of different flaws.

Figure 13.1: Flawed B-rep model of a thick plate with a hole.

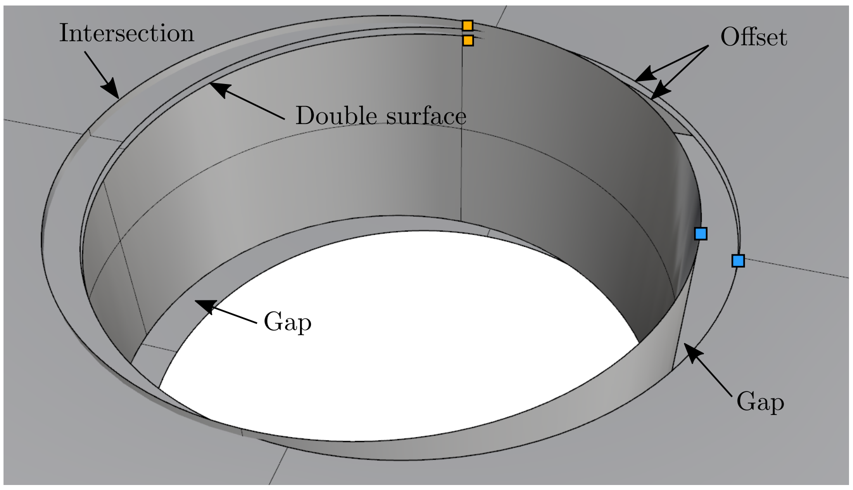


Figure 13.2: Flaws around the hole – gaps, intersection, double surface, and offset. Depicted is the largest opening. For the valid model, the blue marks would coincide. The orange marks indicate the offset.

Convergence of the flaw combinations

Figure 13.3 shows the convergence of the strain energies[1] for increasing polynomial degrees $p = 1...7$. As can be seen, the solution of the flawed and the valid model converge to different values. Naturally, this is the expected outcome, as the models have all different shapes and volumes. In the cases of gaps and intersections only, the strain energy approaches the valid model's strain energy quite well. Compared to

[1]The *relative* error in the strain energy – as the standard measure of the convergence – can only be computed for the valid model, as only here a reference solution is available.

Flaw size	dx	dz_1	dz_2	Gap	Intersection	Double surface	Offset	Abbreviation
small $n_{max} = 4$	0.03	0.015	–	✓	✓	✗	✗	small_gap
			0	✓	✓	✓	✗	small_double
			0.0075	✓	✓	✗	✓	small_offset
medium $n_{max} = 3$	0.06	0.03	–	✓	✓	✗	✗	medium_gap
			0	✓	✓	✓	✗	medium_double
			0.015	✓	✓	✗	✓	medium_offset
large $n_{max} = 2$	0.09	0.045	–	✓	✓	✗	✗	large_gap
			0	✓	✓	✓	✗	large_double
			0.0225	✓	✓	✗	✓	large_offset

Table 13.1: Flaw combinations of different sizes.

this, the deviation is rather significant for the doubled surface and the offset ($\approx 5\%$). This is because the secondary PMC is relatively robust for gaps and intersections, but sensitive for multiple entities and offsets.

Additionally, the size of the double surface and offset is much larger compared to the flawed region of the intersections and gaps. Nevertheless, also for double entities and offsets, the strain energies are still converging. The kink in some convergence curves at $p = 6$ comes from a disadvantageous location of the integration points and can be resolved with a finer integration, e.g., using $2\,(p + 3)$ integration points, as illustrated in Figure 13.4 for the large flaw without double surface, or offset (large_gap). However, for comparability reasons, all convergence studies are carried out with the same – above declared – integration settings. In this context, it is noteworthy that for the valid, the small, and the medium flawed model (without double surface, or offset), a subdivision depth of $k = 3$ is sufficient to determine the moment-fitting weights accurately.

As can be seen, the flawed models converge to higher energies compared to the valid model. The reasons for this are that the offset surface is shifted away from the model, thus, forming a larger volume. And especially because our implementation considers a stalemate in the secondary PMC as inside, i.e., when the 'majority decision' is not possible because the number of rays voting for inside and outside is equal (see Section 10.2.2). Figure 13.5 shows the reconstruction tree $\widehat{T}_{geo}$ for the PMC for the small flaws and qualitative displacements with the finite cell discretization.

Investigation of the secondary PMC

As explained in Section 10.2.2, the secondary ray tracing test is carried out with all non-cut neighbor cells to obtain a more robust membership statement, which allows a detailed investigation of the flawed geometry. Exemplary, the simulations with a polynomial degree of $p = 4$ are looked at closer, yet, it is noteworthy that the numbers and relations are very similar for other polynomial degrees. Depending on the geometry, the total number of PMC tests[2] varies between $72.4 \cdot 10^6$ and $75.4 \cdot 10^6$. Table 13.2 provides an overview of the behavior of the secondary PMC for the different flaw combinations and respective flaw sizes. As can be seen, the maximum reconstruction depth n_{max} has a direct impact on the number of secondary PMCs. For the small flaws with $n_{max} = 4$, only $\sim 30\%$ of the queries are carried out on cut cells. For the large flaws, where $n_{max} = 2$, up to $\sim 92\%$ of the evaluation points are on cut cells and,

[2] *Remark:* The number of PMC tests does not equal the number of integration points. It adds up from many different queries during the simulation, e.g., the setup of the finite cell mesh, the determination of the weights for the moment-fitting, the evaluation of the stiffness matrices, or the computation of the strain energy.

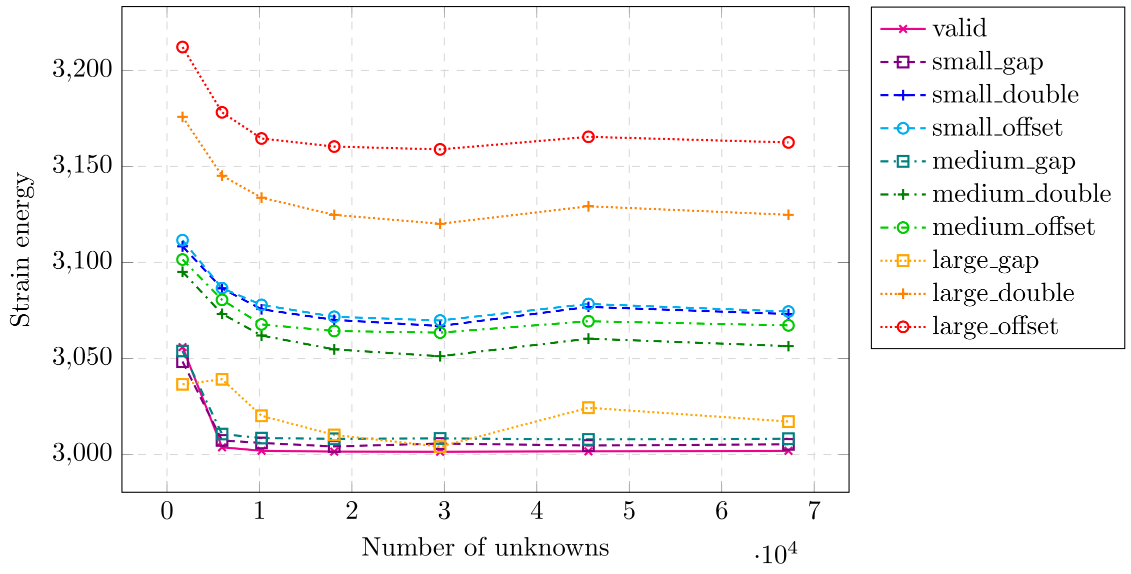


Figure 13.3: Convergence of the strain energy for different flaw combinations and sizes.

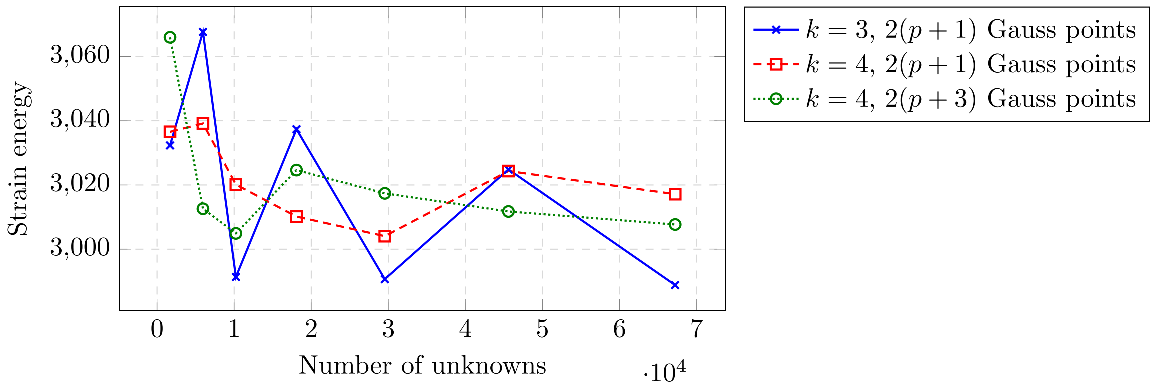
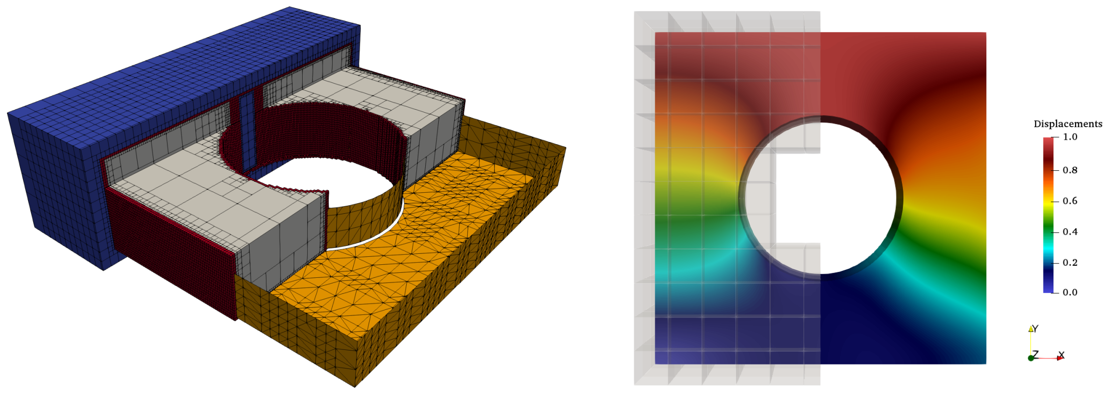


Figure 13.4: Influence of the location and number of integration points on the convergence.

thus, classified with costly multiple ray tracings. This is to be expected since the volume of the cut cells $\hat{c}_{geo}$ is 8 times (for the medium flaws) and 64 times (for the large flaws) larger compared to the cells for the small defects.

For each evaluation point in a cut cell, between one and 22 secondary ray tracings are carried out – one, if the evaluation point is directly on the boundary of the geometry. In average this leads to 13.3 to 15.5 ray tracing evaluations depending on the geometry. In the vicinity of flaws, the multiple secondary PMCs yield, in general, ambiguous results – i.e., at least one ray tracing delivers a different result compared to the others. As illustrated in Table 13.2, this is especially the case for double entities or offsets and gets significantly worse for larger flaws and thus, a coarser reconstruction tree $\widehat{T}_{geo}$. Multiple entities and offsets are also the main cause for stalemates between inside and outside statements.

Furthermore, based on the secondary PMC, it is possible to get an upper and lower bound of the strain energy. For this, two additional simulations are carried out – once with all ambiguous points counting as inside and once counting as outside. Figure 13.7 illustrates such a bracketing simulation for the medium-



(a) (b)

Figure 13.5: (a) Reconstruction tree $\hat{T}_{geo}$ with a maximum subdivision depth of $n_{max} = 4$. (b) Qualitative displacements and finite cell discretization.

Flaw size	Flaw type	Average number of ray tracings to neighbor cells	Amount of PMCs on cut cells (%)	Amount of ambiguous PMCs (%)	Amount of stalemate PMCs[a] (%)
small	gap	15.5	30.2	0.3	1.6
	double	15.5	30.2	7.3	89.7
	offset	15.5	30.2	7.3	89.7
medium	gap	14.3	76.3	1.3	4.1
	double	13.5	61.6	7.5	50.2
	offset	13.5	61.5	7.6	52.4
large	gap	13.9	91.5	3.1	1.2
	double	13.3	75.9	13.7	21.5
	offset	13.3	75.6	12.9	26.0

Table 13.2: Statistics of the secondary PMC for a polynomial degree of $p = 4$.
[a] Referred to the number of ambiguous PMCs.

sized flawed model with an offset (medium_offset). Figures 13.8 and 13.9 show the von Mises stresses from the top and the bottom (evaluated on the flawed STL model) for the reference simulation – where a vote for the majority determines the point membership – and the upper and lower bracketing simulation. The simulations are carried out exemplarily with a polynomial degree of $p = 4$. The displacements are similar for all three simulations. As can be seen, the general stress distribution is alike for all simulations. Yet, when comparing the stress fields on the top and the bottom surface, it becomes apparent that, in particular, the offset has a significant influence on the quality of the PMC. This is expected due to the ray tracing's sensitivity towards multiple entities and the large size of the flaw. By contrast, the stresses in the vicinity of the gaps and intersections around the holes are resolved much better and are similar for all three simulations and both sides.

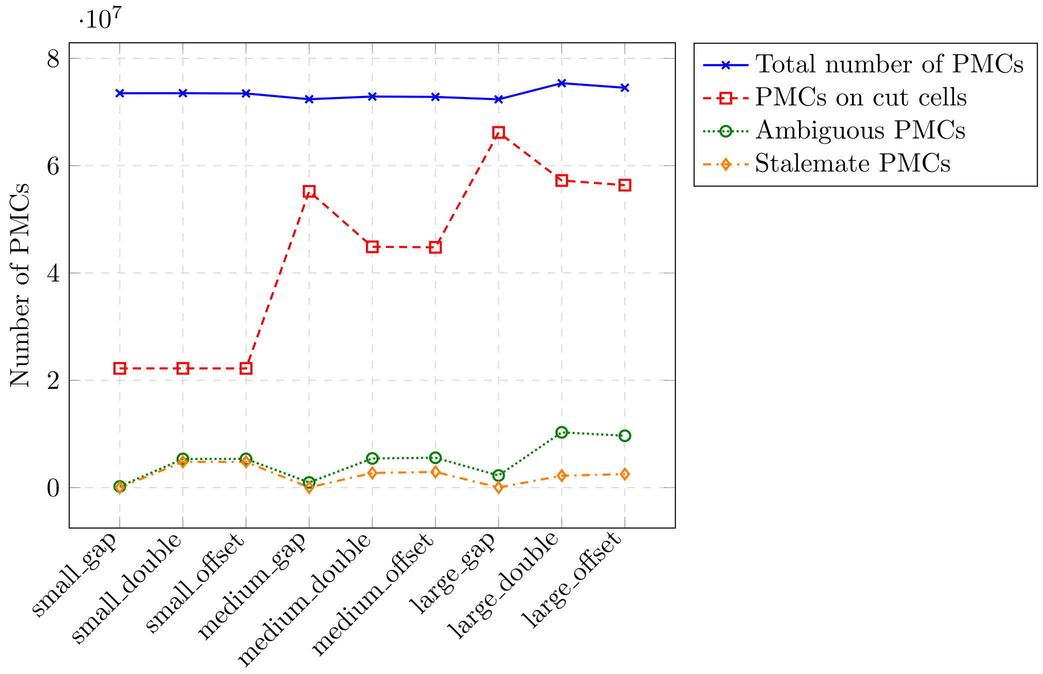


Figure 13.6: Visual representation of the statistics of the secondary PMC for a polynomial degree of $p = 4$.

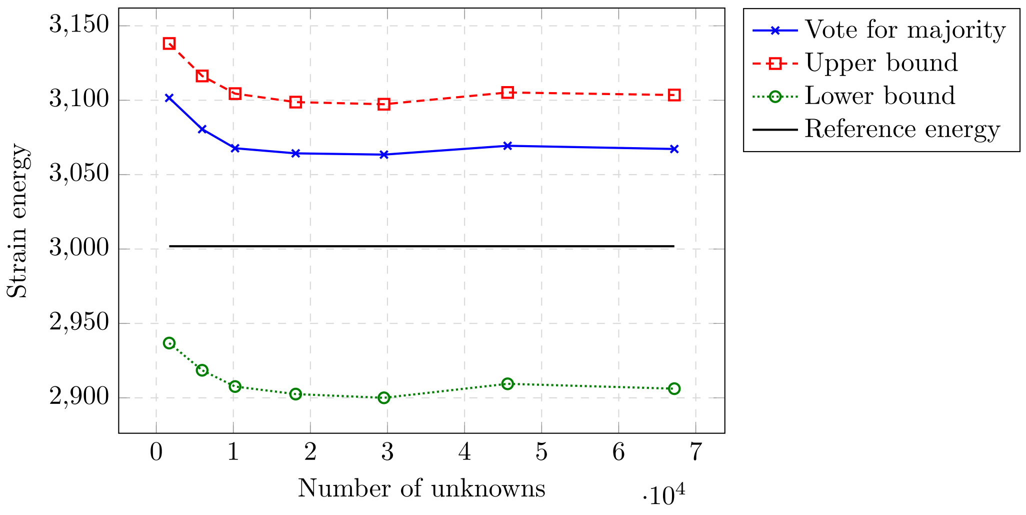


Figure 13.7: Bracketing simulation for the medium sized flaw with offset (medium_offset).

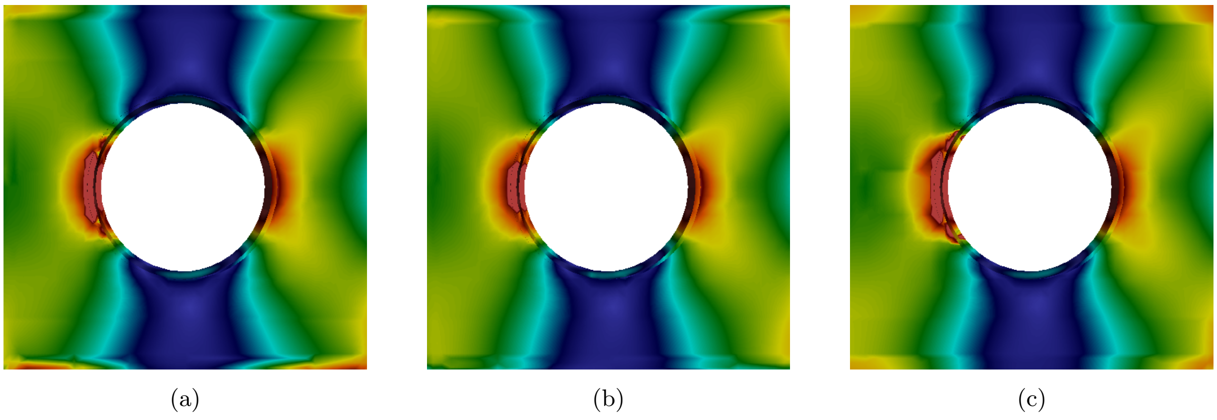


Figure 13.8: Top view of the von Mises stresses for the bracketing simulation of the medium sized flaw with offset (medium_offset): (a) Reference solution with the majority vote, (b) upper boundary, and (c) lower boundary.

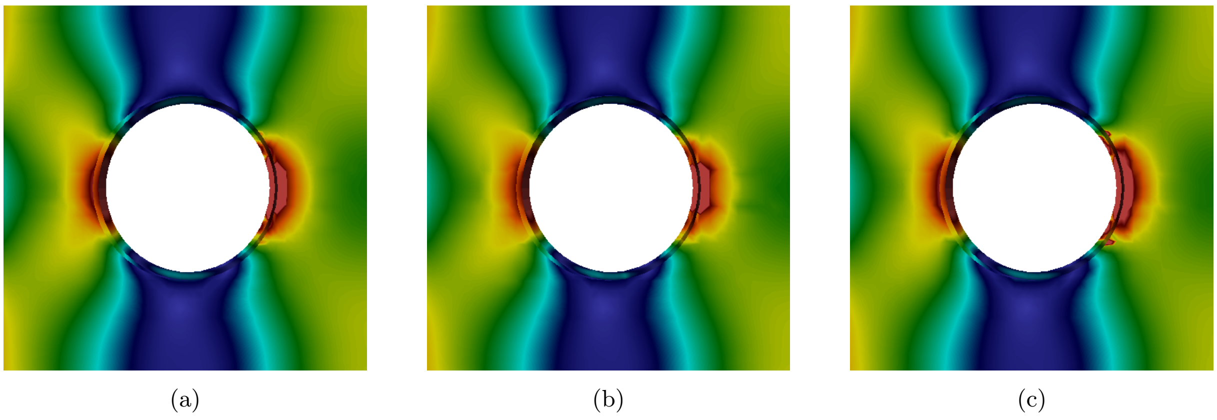


Figure 13.9: Bottom view of the von Mises stresses for the bracketing simulation of the medium sized flaw with offset (medium_offset): (a) Reference solution with the majority vote, (b) upper boundary, and (c) lower boundary.

13.2 Flawed screw

The numerical analysis of the screw, depicted in Figure 13.10, serves as illustration that also for complex models the deviations from the flawless model is not dominant, provided that the size of the flaws is small compared to the model size. For this, several flaws are introduced by changing the position of individual nodes of the STL mesh, resulting in gaps and intersections. Additionally, several triangles are multiplied, leading to double entities and overlaps. Similar to Example 13.1 two flawed models with smaller and larger characteristic flaw sizes are generated. Since a flawless model is available, it is possible to compare the solutions of the two flawed models with the valid model. The simulation is carried out on $10 \times 30 \times 10$ finite cells, employing integrated Legendre shape functions with a polynomial degree of $p = 3$. The maximum partitioning depth for integrating the cut FCM cells was set to $k = 4$. The screw is clamped on the bottom and loaded on the top with a Dirichlet boundary condition that bends the screw in $x-$direction, as depicted in Figure 13.12.

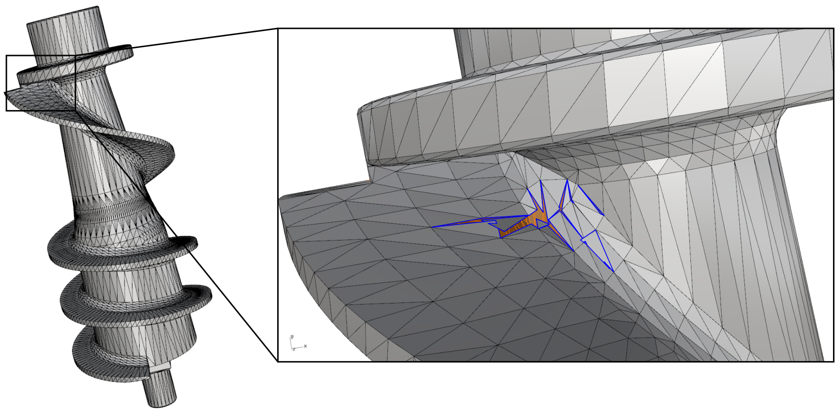


Figure 13.10: STL model of a screw with locally, artificially introduced (large) flaws. Free edges are highlighted in blue. Gaps allow to see the orange-colored backside of the screw.

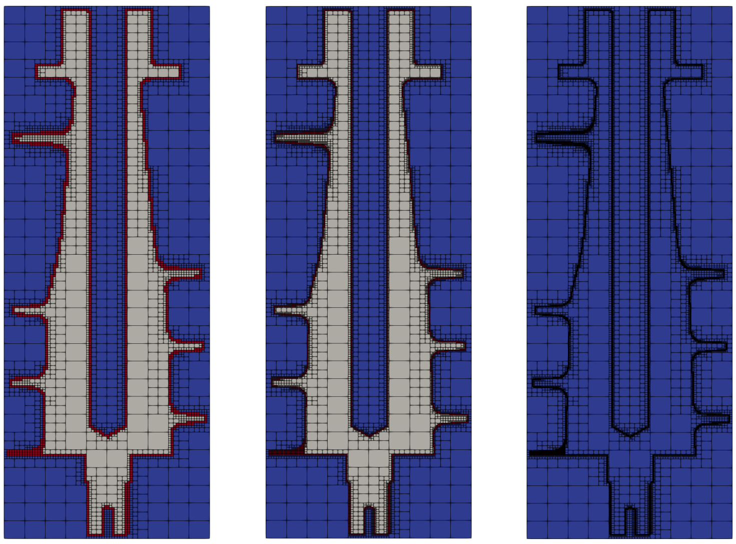
The largest gaps of both flawed models allow reconstruction trees $\widehat{T}_{geo}$ with maximum subdivision depths of $n_{max} = 4$ and $n_{max} = 3$ for the small-flawed and the large-flawed model, respectively. Figure 13.11 illustrates the case in which the resolution is chosen too fine, and the flood-fill algorithm marks all non-cut leaves as outside. Assuming a finite cell discretization with cubic cells, the ratios between screw length l_y and the characteristic sizes of the largest gaps ε^i_{gap} are in the following ranges

$$\frac{1}{240} = \frac{1}{2^3 \cdot 30} < \frac{\varepsilon^{large}_{gap}}{l_y} \leq \frac{1}{480} = \frac{1}{2^4 \cdot 30} < \frac{\varepsilon^{small}_{gap}}{l_y} \leq \frac{1}{960} = \frac{1}{2^5 \cdot 30}\,. \tag{13.1}$$

The factor 30 comes from the fact that a reconstruction tree $\widehat{T}_{geo}$ is deployed on each finite cell (see Remarks in 10.2.1).

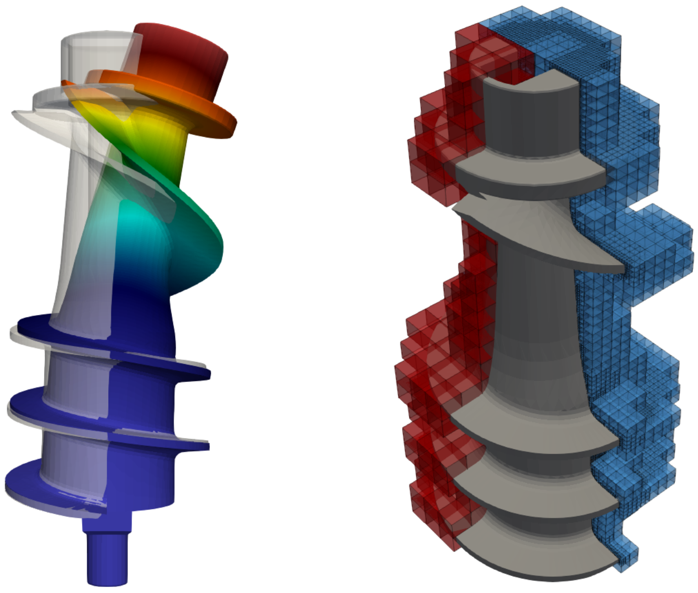
Figure 13.13 shows the qualitative von Mises stresses for the valid, and the small- and large-flawed model. For the three models on the upper row, the stresses are evaluated on the valid STL file. As can be seen, a visual comparison shows no differences. The same also holds for the less critical displacements. For the two details of the flawed region (in the lower row), the stresses are evaluated on the respective defective STL files. Here, the gaps allow the sight on the interior/backside of the model. However, the stresses on the foreground mesh are similar to the valid model.

Analogously to Example 13.1, the secondary PMC allows a more detailed investigation of the flaws (see Table 13.3). As can be seen, the coarser reconstruction of the large-flawed model leads to a significantly



(a) (b) (c)

Figure 13.11: Slice through an octree reconstruction tree $\hat{T}_{geo}$, for (a) the large-flawed model with $n_{max} = 3$, (b) the small-flawed model with $n_{max} = 4$, and (c) a too fine reconstruction with $n_{max} = 5$, where the flood fill algorithm marks the non-cut domain as outside.



(a) (b)

Figure 13.12: (a) Warped qualitative displacements. (b) Finite cell discretization (red), and integration octree (blue).

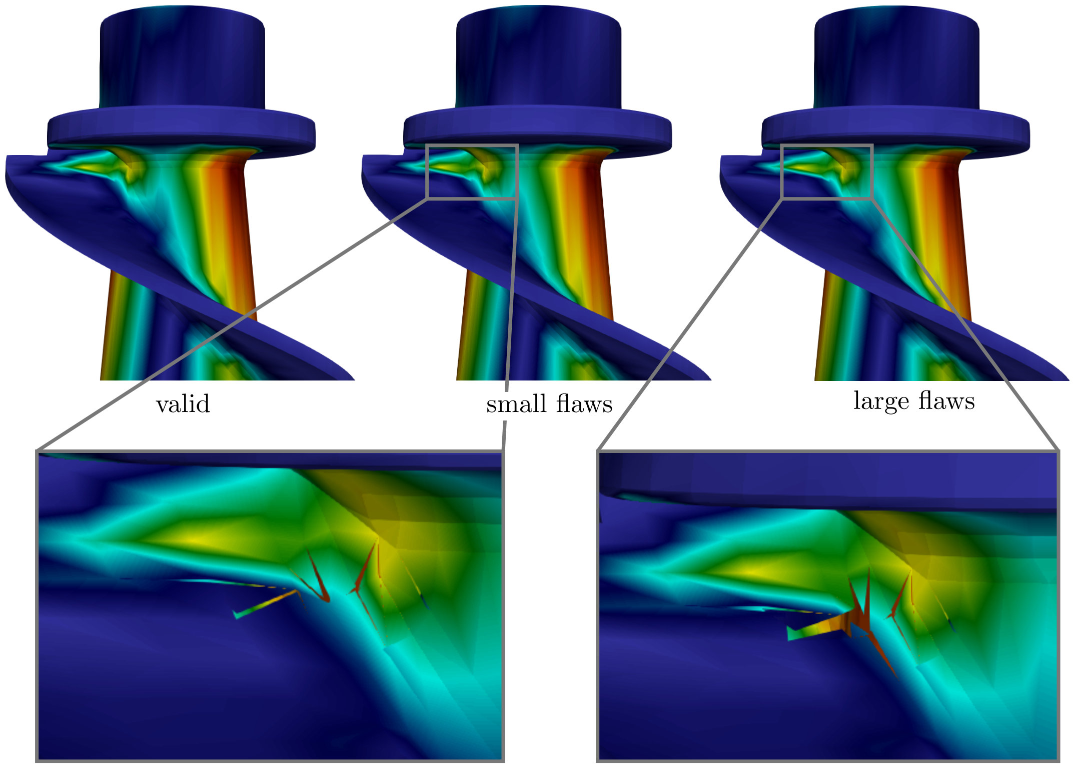


Figure 13.13: Von Mises stresses on the valid and the small- and large-flawed model.

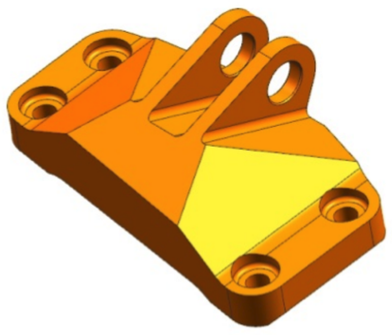
larger number of secondary PMCs on the cut cells. Since the flaws are introduced only in a small region, the number of ambiguous PMCs is very small. A comparison of ambiguous PMCs and PMCs that led to a different classification with respect to the valid model, shows the robustness of the secondary PMC – about 70% of the ambiguous test are classified 'correctly', i.e., the 'majority decision' results in the same classification as does the valid model. Considering that only $\sim 0.0012\%$ (for the small flaws) and $\sim 0.0034\%$ (for the large flaws) of the PMCs led to a 'wrong' classification, it is not surprising that the von Mises stresses showed no visual deviation. Nevertheless, apart from the fact that the flaws are limited to a specific region, the 'defectiveness' of the model is higher than most of the real-world CAD models, which renders this example a valid prove of concept of the proposed method.

Flaw size	Number of PMCs	Number of PMCs which differ from the valid model	Number of PMCs on cut cells	Number of ambi-uous PMCs
small	113 029 521	1 353	24 934 803	4 549
large	113 118 354	3 882	75 026 028	15 749

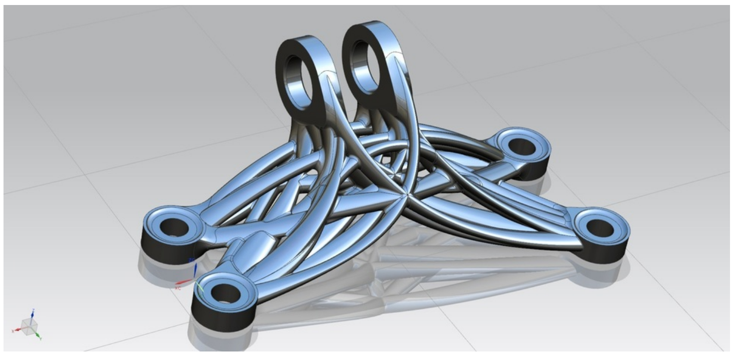
Table 13.3: Statistics of the secondary PMC.

13.3 Flawed engine bracket

In 2013, a collaboration of GrabCAD® and General Electric® arranged a competition to find the optimal design of an engine bracket for a turbofan [75]. The submitted designs were evaluated, and the top ten models were produced with additive manufacturing. The model depicted in Figure 13.14 was designed by Sean Morrissey[3].



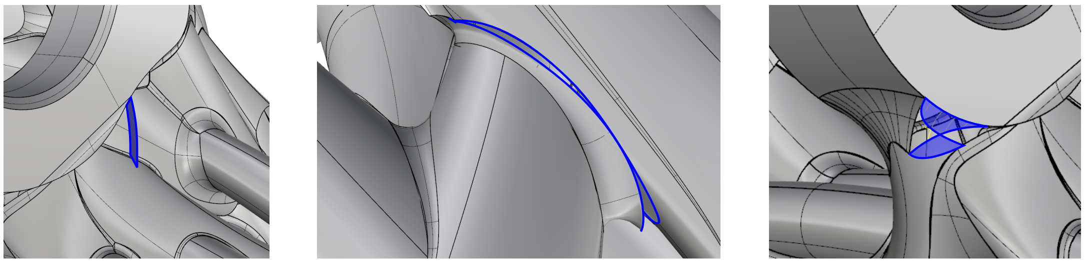
(a) Initial design [75].



(b) Submitted design [118].

Figure 13.14: General Electric design challenge for the optimal shape of a jet engine bracket.

The model was submitted in the proprietary PTC Creo Elements/Pro® '.prt'-format and was additionally provided in the open CAD formats IGES and STEP. In an attempt to perform a numerical simulation, it turned out that the conversions to IGES and STEP had introduced numerous flaws, as illustrated in the Figures 13.15 and 13.16. Interestingly, most of the flaws occur either in the IGES or the STEP file. The predominant reasons for flaws are unconnected trimmed spline-patches that mainly cause gaps and (self-)intersections. But also artifacts and incorrectly constructed surfaces appear. A triangulation of the IGES model with 155 150 triangles yields 8 382 free edges. The STEP model is less flawed – a comparable triangulation with 152 844 has only 575 free edges. Due to the complex shape of the model, an automatized geometry healing is not applicable, as most of the flaws are either fixed unreasonably or are not resolved at all. However, with the approach presented in this thesis, it is possible to simulate without any further treatment of the flaws.



(a) (b) (c)

Figure 13.15: Flaws in the IGES model: (a) Artifact causing intersections, (b) one of many gaps, and (c) incorrectly defined surface.

[3]https://grabcad.com/sean.morrissey-1

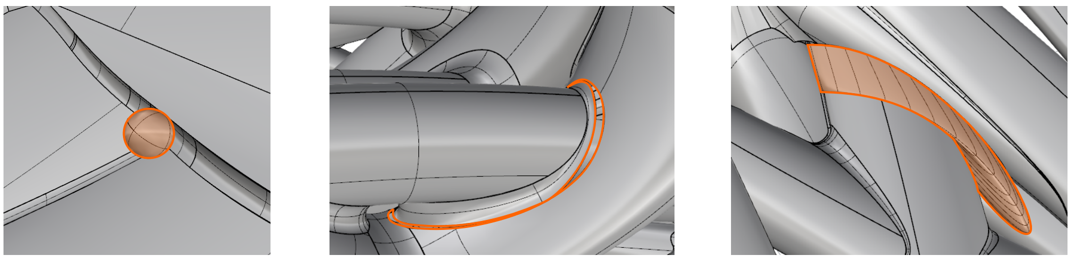


Figure 13.16: Flaws in the STEP model: (a) Artifact, (b) one of many gaps, and (c) an incorrectly joined surface, causing gaps and intersections.

In the following, the submitted model is evaluated regarding its load-carrying capacities based on the original four predefined loading conditions from General Electric® (see Figure 13.17), i.e., a vertical, a horizontal, a diagonal force and a torque around the $z-$axis. The corresponding simulations are carried out in the *foot-pound-second* (FPS) system, since the simulation settings and the models are provided within this system. The outer dimensions of the model are $6.704\,in \times 4.025\,in \times 2.387\,in$ ($17.03\,cm \times 10.22\,cm \times 6.06\,cm$). The computational domain is chosen slightly larger with an offset of $0.06\,in$ ($0.15\,cm$) in each direction and discretized with $27 \times 16 \times 10$ finite cells. The polynomial degree of the hierarchical Legendre polynomials is $p = 4$. For the integration moment-fitting with $2(p+1)$ quadrature points in each direction is used. The moment-fitting weights are computed on an octree with a maximum subdivision depth of $k = 4$ and $p + 1$ integration points in each direction.

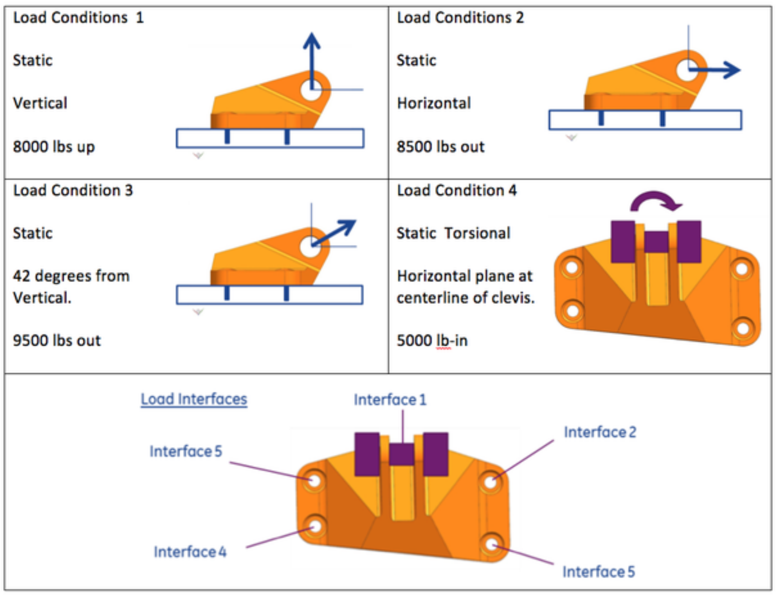


Figure 13.17: Loading conditions as defined by General Electric® [75].

The reconstruction tree $\widehat{T}_{geo}$ for the PMC is set up on each finite cell and has a maximum subdivision depth of $n_{max} = 3$ in the case of the IGES model and $n_{max} = 4$ for the STEP model (see Figure 13.18).

Figure 13.19 shows the critical flaws, i.e., the largest gaps, for the triangulation of the IGES and the STEP model in detail. For the IGES model, the largest gap is due to the incorrectly defined surface depicted in Figure 13.15(c), and for the STEP model, the gap is depicted in Figure 13.16(b). As can also be seen, most of the IGES model surfaces are not connected, leading to many gaps between the trimmed surfaces.

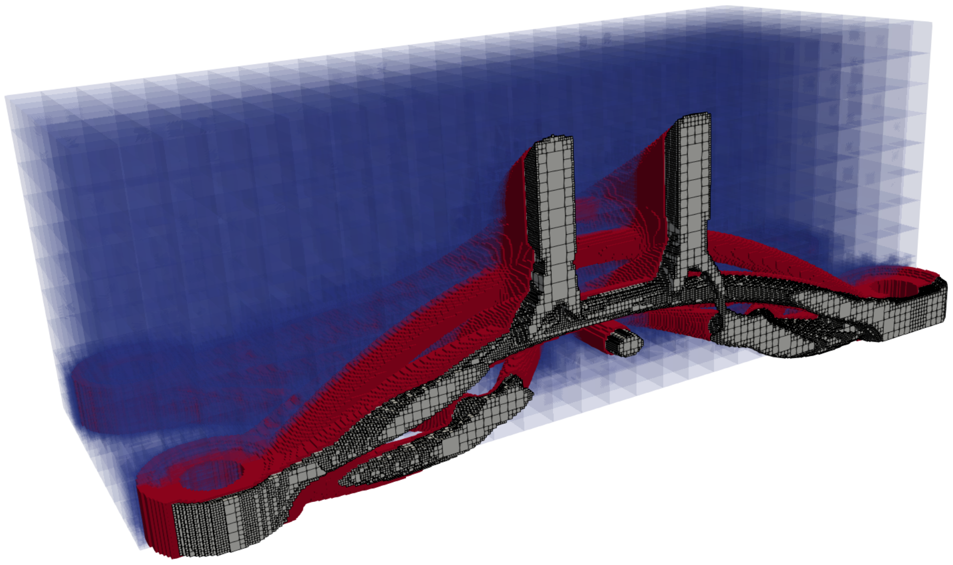


Figure 13.18: Reconstruction tree $\widehat{T}_{geo}$ of the STEP model with a maximum subdivision depth of $n_{max} = 4$.

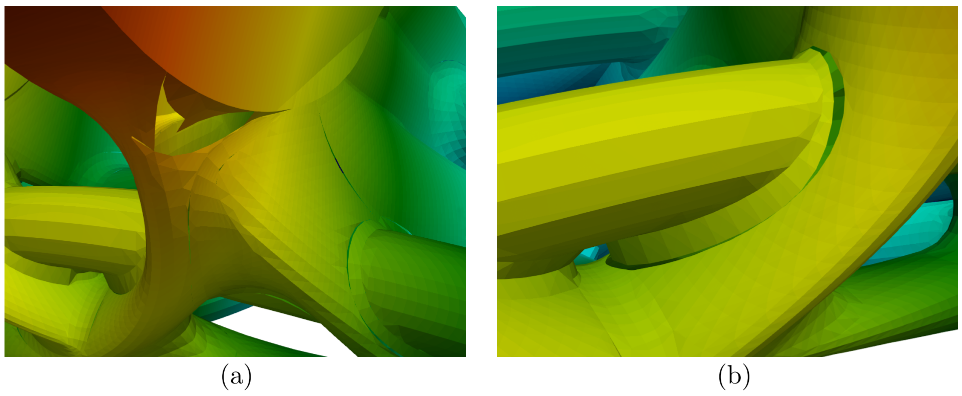


Figure 13.19: Critical flaws on (a) the IGES model and (b) the STEP model.

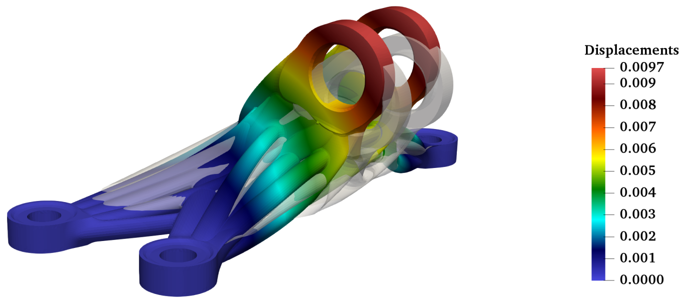
For the printing material, the high-strength and corrosion-resistant titanium alloy Ti-6Al-4V was specified. As material properties for the simulations, a Young's modulus of 15736.6 ksi (10850 $\frac{kN}{cm^2}$) and a Poisson's ratio of 0.34 were chosen. Material failure is assumed, if the yield strength of 131 ksi (90, 3 $\frac{kN}{cm^2}$) is exceeded.

The mounting is realized with homogeneous Dirichlet boundary conditions applied in a weak sense using the penalty method. Displacements in $z-$direction are restricted on the four bottom surfaces around the support holes, and in $x-$ and $y-$directions on the four lateral surfaces of the support holes. The four load cases are applied to the two lateral surfaces of the loading holes. For this, the respective forces and the torque are translated into sets of surface tractions $\bar{t}_i$, i.e., the Neumann boundary conditions. Depending on which lateral surface these tractions are applied, they are referred to as Neumann 1 and Neumann 2. The areas of lateral surfaces of the loading holes are $0.567\,in^2$ $(3.658\,cm^2)$ each. Table 13.4 shows the surface tractions for the respective load cases. As the torque's lever arm, the distance between the barycenters of the lateral loading surfaces is used $s = 1.125\,in$ $(s = 2.86\,cm)$.

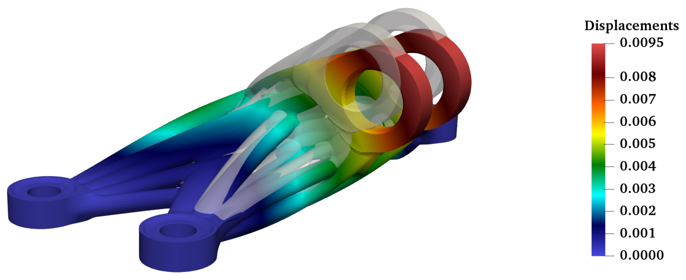
Load case	Load		Direction	Neumann 1		Neumann 2	
				(ksi)	$\left(\frac{kN}{cm^2}\right)$	(ksi)	$\left(\frac{kN}{cm^2}\right)$
vertical	$8\,000\,lb$	$36.59\,kN$	z	7.06	4.86	7.06	4.86
horizontal	$8\,500\,lb$	$37.81\,kN$	y	7.50	5.17	7.50	5.17
diagonal	$9\,500\,lb$	$42.26\,kN$	z	6.23	4.29	6.23	4.29
			y	5.61	3.86	5.61	3.86
torque	$5\,000\,lb\text{-}in$	$56.49\,kNcm$	y	-7.84	-5.40	7.84	5.40

Table 13.4: Surface tractions of the different load cases for the Neumann boundary conditions.

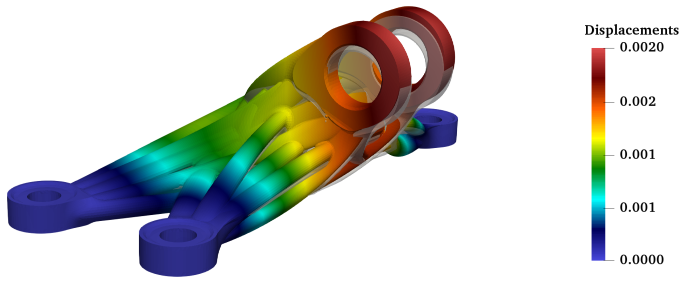
Figures 13.20 and 13.21 show the displacements and von Mises stresses for the different load cases. For the vertical and horizontal load cases, the highest stress concentrations are around the bottom's support. As can be seen, the design can only withstand the diagonal load condition, as only here the maximum stresses of $126\,ksi$ do not exceed the yield stress of $131\,ksi$. For the torque condition, the sheering of the two loading flaps leads to very high stress concentrations at the joining rod between the flaps. The general stress distribution implies that the model did not result from topology optimization.



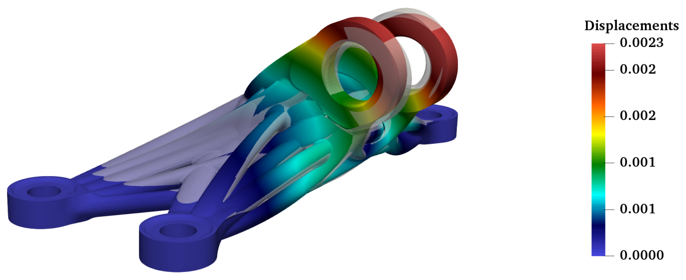
(a) Vertical load case.



(b) Horizontal load case.

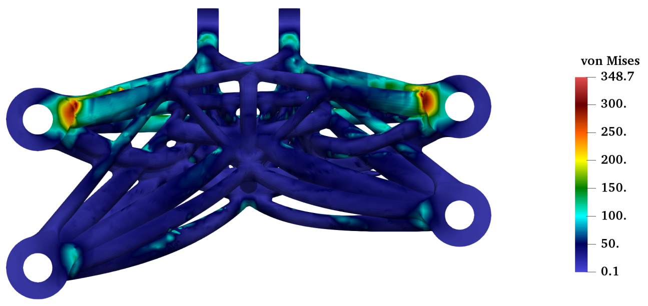


(c) Diagonal load case.

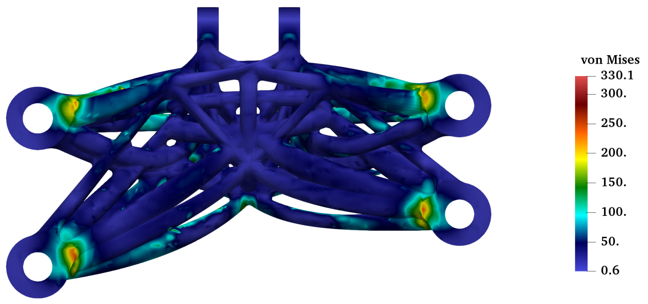


(d) Torque load case.

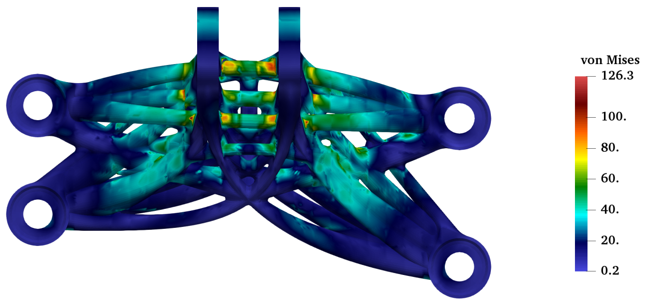
Figure 13.20: Displacements in $(inch)$ for the different load cases – warped by a scaling factor of $s = 50$.



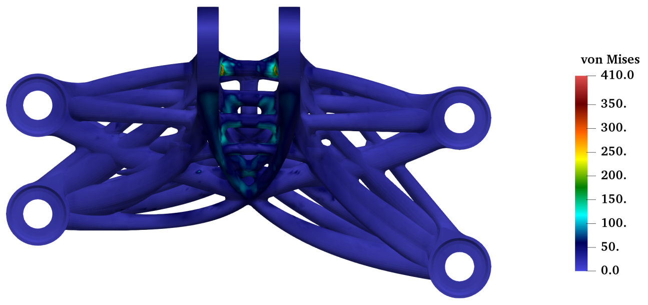
(a) Vertical load case (bottom view).



(b) Horizontal load case (bottom view).



(c) Diagonal load case (top view).



(d) Torque load case (top view).

Figure 13.21: Von Mises stresses in (ksi) for the different load cases.

Chapter 14

Applications of solid-based procedural and CSG models

The following four solid-based procedural/CSG examples involve several extended primitives and operations. The first two examples illustrate the application of the PMC for a sweep and a loft, respectively. The third example focuses on a variety of extended operations and the last example shows the applicability to practical applications. Additionally, several geometrical and topological changes are introduced, illustrating the advantageous geometry independence of the FCM from the CAD model. It is noteworthy, that in all involved steps only the accurate procedural/CSG models are used, i.e., from the CAD model description and during the entire simulation. Only for the visualization of the results, a surface triangulation is carried out, based on marching cubes. However, this conversion into a B-rep model is not mandatory as volumetric post-processing is a possible option as well.

14.1 Coil spring

The example of a coil spring illustrates the PMC on sweeps. The underlying CAD model consists of three sweeps that are combined with two union operations (see Figure 14.1). All three sweeps are created by moving a circular sketch of radius $r_{\text{sketch}} = 1$ along the respective sweep paths. For the bottom and top sweep these paths are circles with a radius of $r_{\text{torus}} = 10$. The resulting tori are aligned to the $x-y-$plane and located at $z_{\text{bottom}} = 0$ and $z_{\text{top}} = 24$, respectively. The computational domain ranges from $z = 0$ to $z = 24$, thus, clipping the bottom and top torus into half-tori. The sweep path of the coil is described by a helical NURBS curve of degree $p = 2$, the open knot vector

$$\Xi = [0,0,0,\ 1,1,\ 2,2,\ 3,3,\ 4,4,\ 5,5,\ 6,6,\ 7,7,\ 8,8,\ 9,9,\ 10,10\ ,11,11,\ 12,12,12]\,,$$

and the NURBS control points $\tilde{\boldsymbol{P}}_i$ with the alternating weights $w_1 = 1$ and $w_2 = \frac{1}{\sqrt{2}}$

$$\tilde{\boldsymbol{P}}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \\ w_i \end{bmatrix} = \begin{bmatrix} 10 & 10 & 0 & -10 & -10 & -10 & 0 & 10 & 10 & 10 & 0 & -10 \\ 0 & 10 & 10 & 10 & 0 & -10 & -10 & -10 & 0 & 10 & 10 & 10 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ w_1 & w_2 & w_1 & w_2 & w_1 & w_2 & w_1 & w_2 & w_1 & w_2 & w_1 & w_2 \end{matrix} \cdots$$

$$\cdots \begin{matrix} -10 & -10 & 0 & 10 & 10 & 10 & 0 & -10 & -10 & -10 & 0 & 10 & 10 \\ 0 & -10 & -10 & -10 & 0 & 10 & 10 & 10 & 0 & -10 & -10 & -10 & 0 \\ 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \\ w_1 & w_2 & w_1 & w_2 & w_1 & w_2 & w_1 & w_2 & w_1 & w_2 & w_1 & w_2 & w_1 \end{bmatrix}.$$

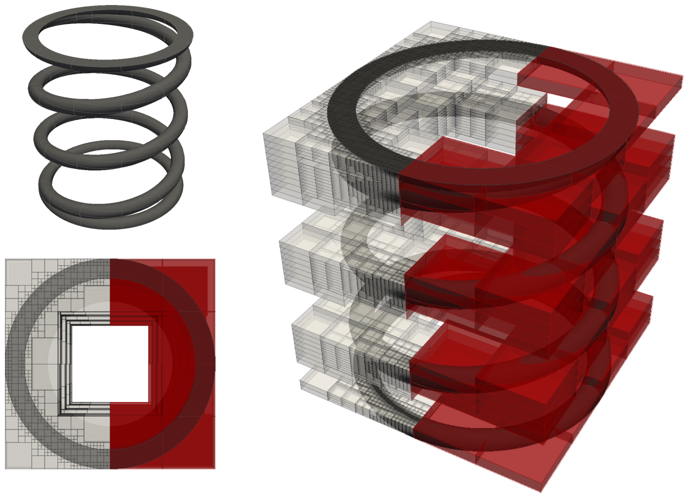
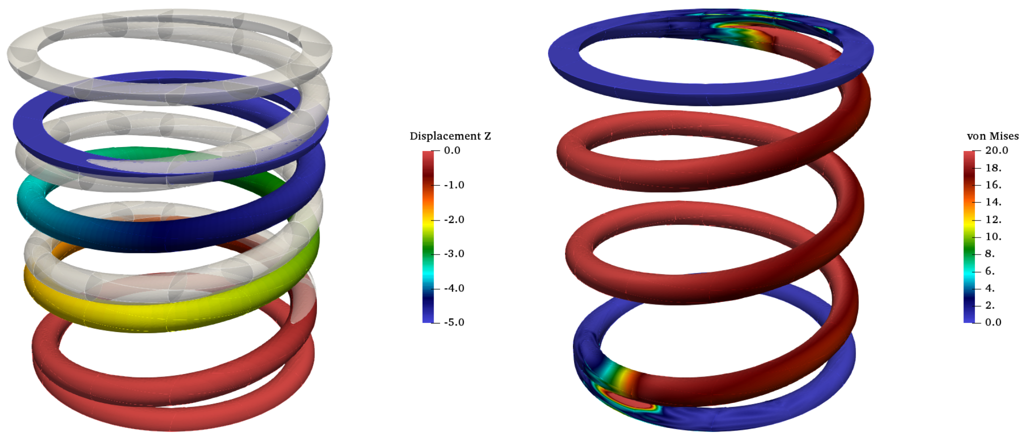


Figure 14.1: Coil spring model consisting of three sweeps, with the finite cell discretization (red) and octree partitioning for the integration of cut cells (gray).

The computational domain is discretized with $4 \times 4 \times 24$ finite cells using integrated Legendre shape functions with a polynomial degree of $p = 7$. The integration is carried out on an octree with a maximum partitioning depth of $k = 6$. On both intersection surfaces, homogeneous Dirichlet boundary conditions are applied in all directions, except for the vertical displacement on the top face, which is predefined by $\overline{u}_z = -5$.

Figure 14.2 shows the resulting displacements and the von Mises stresses, which are smooth throughout most parts of the domain and exhibit singular behavior at the intersections of the bottom and top torus with the coil. As in any finite element computation, local stresses at singularities can not be accurately resolved without an appropriate refinement.



(a) Displacements (b) Stresses

Figure 14.2: Displacements and von Mises stresses.

14.2 Loft

This example illustrates the PMC for the second class of extended primitives: lofts. For this a numerical simulation of a complicated pipe elbow with two flanges is carried out. The pipe is modeled as Boolean difference of two lofts. The lofts start with a circular and end with a rectangular sketch profile. The loft path is, for both cases, a quadratic B-spline which is defined by five control points. The model was initially constructed as procedural CAD model with Autodesk Inventor® and is – for the simulation – transformed into a CSG tree, as illustrated in Figure 14.3. The dimensions of the example are depicted in Figure 14.4.

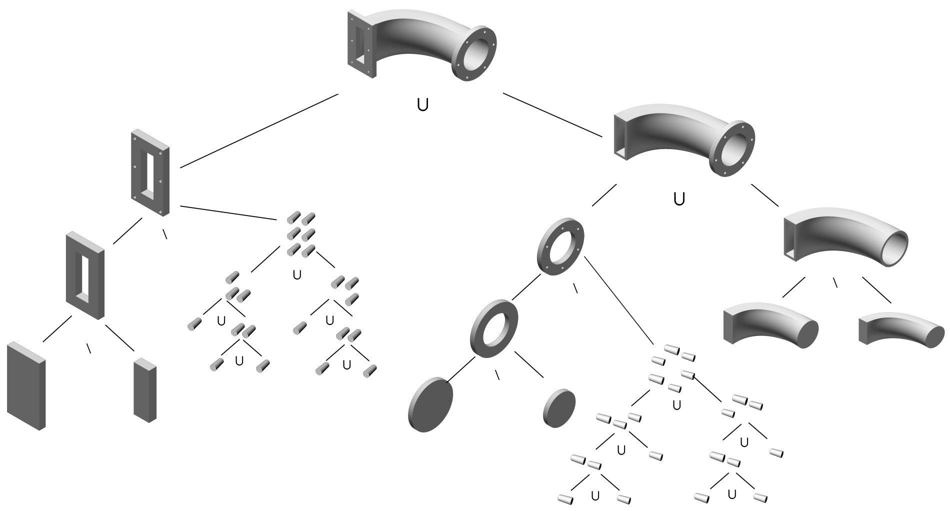


Figure 14.3: CSG tree of the pipe elbow model.

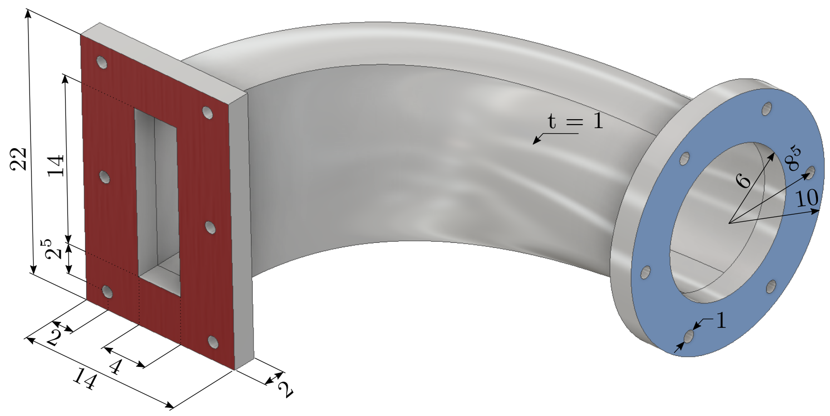


Figure 14.4: Dimensions of the loft.

One big advantage of procedural models is the possibility to subsequently change the model with only minimal effort. In the context of classical FEM, it is essential to distinguish between geometrical and topological changes. Geometrical changes alter the parameters which define a primitive. Depending on the extent of the respective alterations, it might be possible to morph the FEM mesh to fit the changed geometry. In the case of topological changes, however, it is not possible to adapt the FEM mesh, but a re-meshing is required. Topological changes can appear due to adding or deleting primitives with the corresponding operations. But also a geometrical change might lead to a topological change. For instance, consider the increase of the diameter of a borehole close to the boundary of a part. At some point, the diameter will be so large that the drilling becomes a cove.

In contrast to classical FEM, the FCM is insensitive to geometrical and topological changes since the discretization is independent of the geometrical shape. The different form is resolved on the integration level, leading to a different composed integration. In the current example, a geometrical change is realized with a shift of one control point of the loft path (see Figure 14.5).

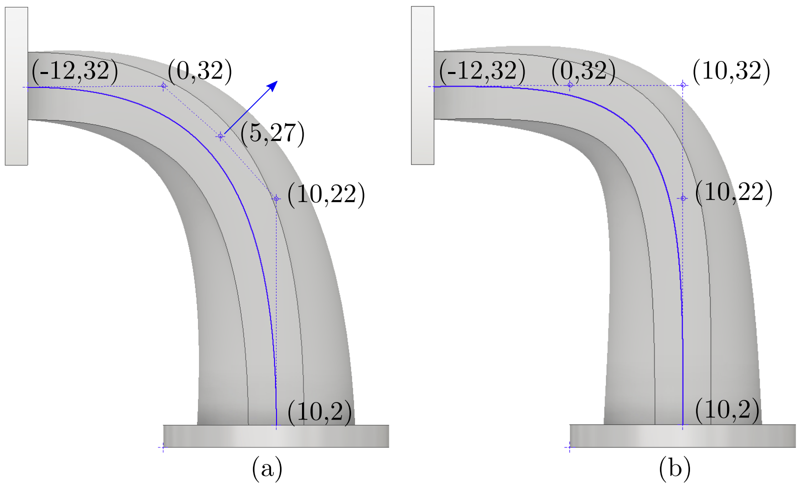
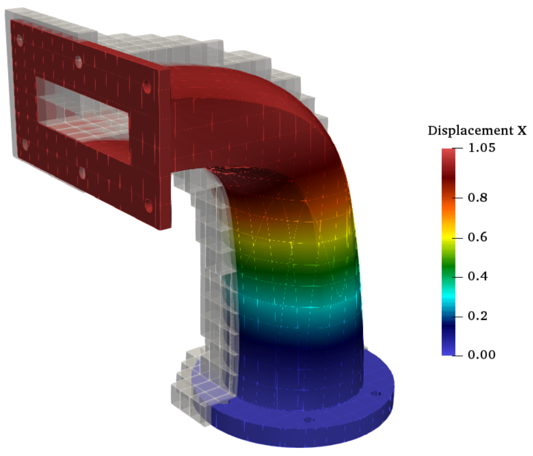


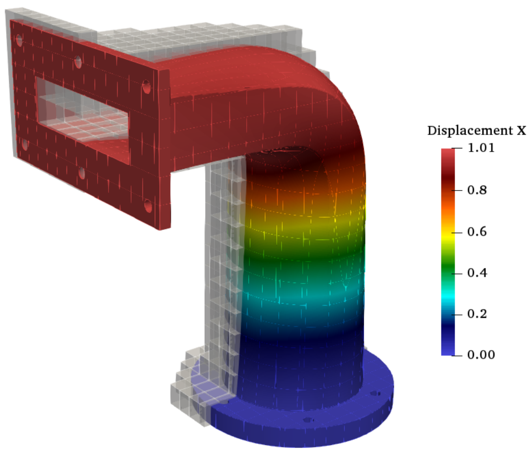
Figure 14.5: Loft path of (a) the original model and (b) a variation, in which one control point is shifted.

For the simulations, the computational domain is discretized with $20 \times 15 \times 20$ finite cells that employ integrated Legendre polynomials with a polynomial degree of $p = 5$. For the integration, the cells are partitioned with an octree with a maximum subdivision depth of $k = 5$. Homogeneous Dirichlet boundary conditions fix the round base plate. As a loading, a predefined deflection $\overline{u}_x = 1$ is applied onto the left, rectangular plate. Figures 14.6 and 14.7 show the displacements and von Mises stresses and the finite cell mesh for both variations of the elbow pipe.

As pointed out before, also a topological change does not affect the setup of the FCM. Figure 14.8 illustrates a simple modification of the model. In the initial model, six boreholes are drilled into the rectangular plate. For the modification, the number of drillings reduces to four. The initial boundary conditions would hardly impact the results, which is why a more expressive Neumann boundary condition replaces the prescribed deflection. More precisely, a moment is applied, via added washers at each hole. For this, the upper and lower rows of washers are loaded with tractions of $\overline{t}_x = 1$ and $\overline{t}_x = -1$, respectively. Figures 14.9 and 14.10 show the (warped) displacements and von Mises stresses in the region around the topological changes.

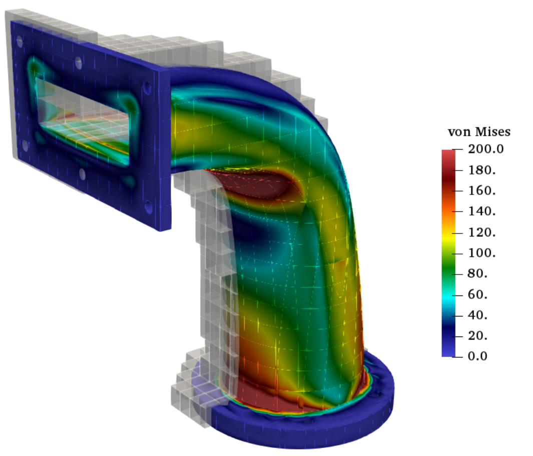


(a)

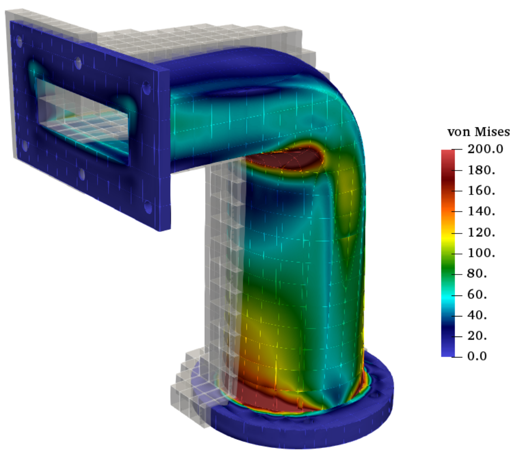


(b)

Figure 14.6: Displacements of (a) the original model and (b) the variation with changed loft path.



(a)



(b)

Figure 14.7: Von Mises stresses of (a) the original model and (b) the variation with changed loft path.

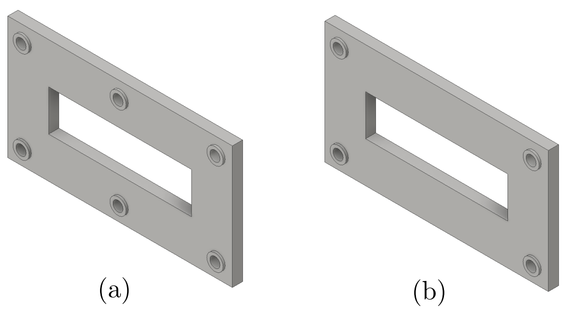


Figure 14.8: Topological changes on the rectangular plate: (a)&(b) Washers added to the holes and (b) deletion of the middle holes.

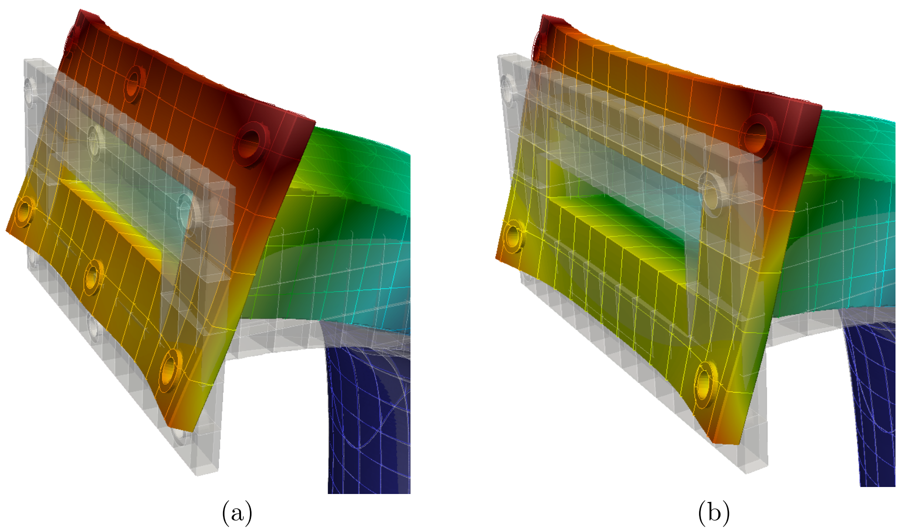


Figure 14.9: Displacements at the rectangular plate for the two different topologies, scaled by a factor of $s = 200$.

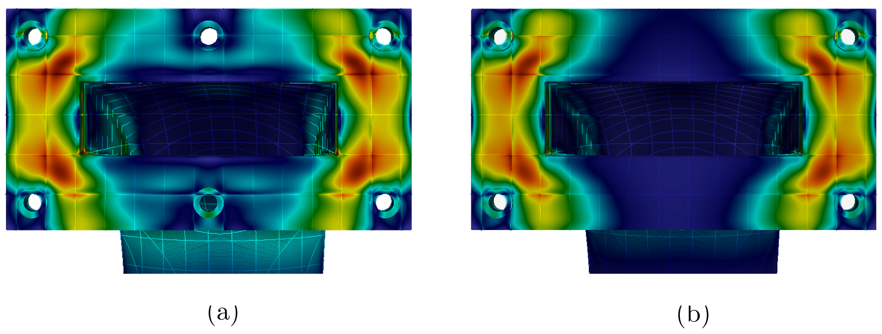


Figure 14.10: Von Mises stresses at the rectangular plate for the two different topologies.

14.3 Extended operations

Modern solid-based procedural CAD systems, provide – apart from extended primitives – also extended operations. As explained in Section 7.2.2, these operations can be represented by CSG trees on their own. This example illustrates several common operations, as provided, for instance by Autodesk Inventor®. Starting from a cube, four edges are chamfered, two edges are filleted, three holes are drilled, and one shell operation is applied (see Figure 14.11).

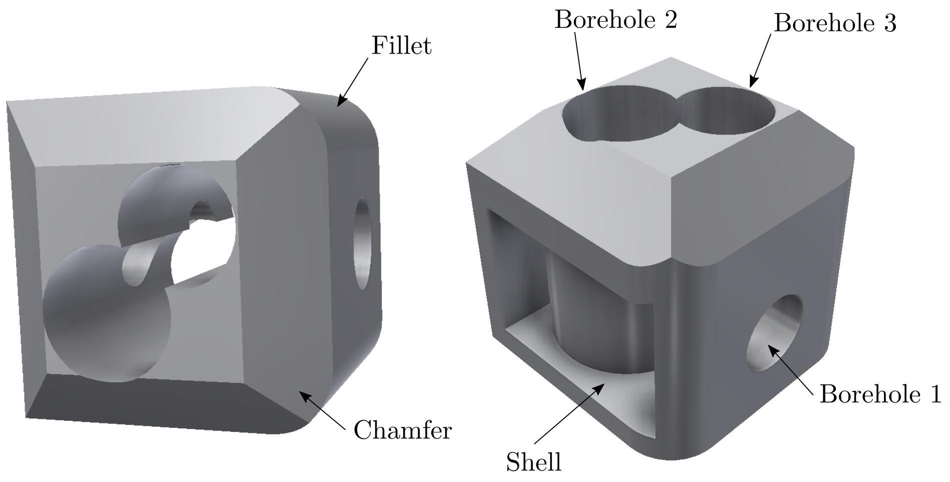


Figure 14.11: Extended operations applied on a cube: chamfer, fillet, drilling a hole, and shell operation.

The extended operations fillet, chamfer, and drilling a hole are comparably easy to translate into a respective CSG tree. Shells, on the other hand, are more elaborate. The shell operation is applied to one surface, caving the volume while keeping a predefined distance to all other surfaces. In the present model, it is possible to model the shell operation manually. To this end, the shell operation is put to the beginning of the construction chain, where a smaller cuboid is subtracted from the initial cube. Subsequently, the other subtractive extended operations are modified so that first, a larger primitive is added, from which the actual fillet, chamfer, or hole is then subtracted. Certainly, this approach is not straightforward for more sophisticated cases. However, as explained in Section 7.2.2, the 'shell' procedure is not a solid-based operation since it requires knowledge about the distances to the boundaries.

To illustrate another advantage of the FCM, two almost identical models are created, subsequently meshed, and the resulting meshes' quality is compared. In the initial model, borehole one is created by subtracting a cylinder. For the modified model, this hole is composed of two extruded half-cylinders, which are shifted by 0.05% with respect to the model's dimension. This shift does not have any significant influence on the results of the simulation. However, it does lead to a considerable increase in the effort needed for the meshing for a classical FEM mesh. Figure 14.12 shows the resolution of the meshes, created by Netgen [119], around borehole one. Although the shift is minimal and only applied at one hole, the mesh of the second model has around 18 times more elements.

Moreover, many elements are very badly shaped. Even if this mesh represents the 'exact' geometry of the CAD model, this is probably not the intended finite element discretization. Similarly, there are often seemingly unmotivated refinements in practical applications, so considerable engineering effort has to be spent on remodeling a structure before an efficient numerical analysis can be carried out.

For the FCM simulation, $10 \times 10 \times 10$ finite cells using integrated Legendre polynomials of degree

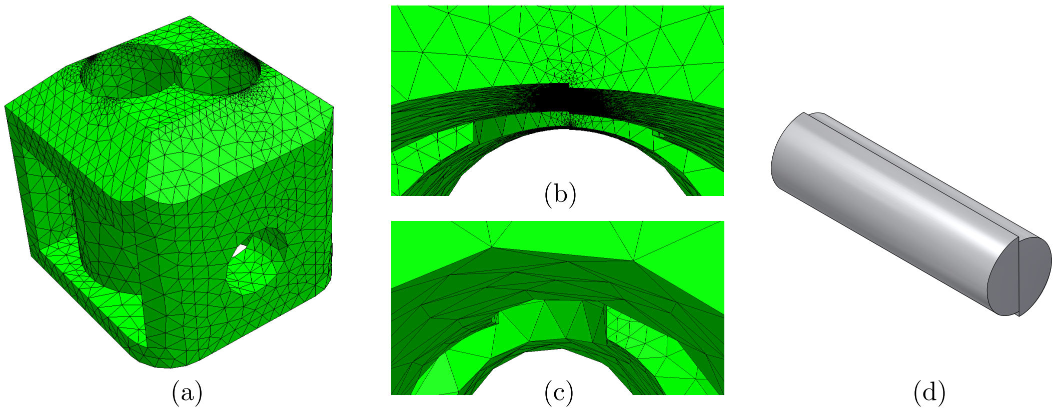


Figure 14.12: Models meshed with Netgen [119]: (a) Entire mesh of the initial model, (b) refinement around the hole that is constructed with two slightly shifted half-cylinders, and (c) mesh around a hole without the (unnecessary) geometric detail. (d) Two slightly shifted half-cylinders are subtracted to generate the modified model (shift not to scale).

$p = 5$ and an octree with a partitioning depth of $k = 4$ are used. The bottom surface is fixed, and on the top face, a predefined deflection of $\overline{u}_z = -0.5$ is applied in $z-$direction. Figure 14.13 shows the displacements, and von Mises stresses for the initial model. A close-up of the displacements and von Mises stresses around borehole one is depicted in 14.14. The results of the FCM computation for the two models (full cylinder versus two shifted half-cylinders) are identical up to the used geometrical precision, proving that the proposed method is robust against imprecise CAD modeling.

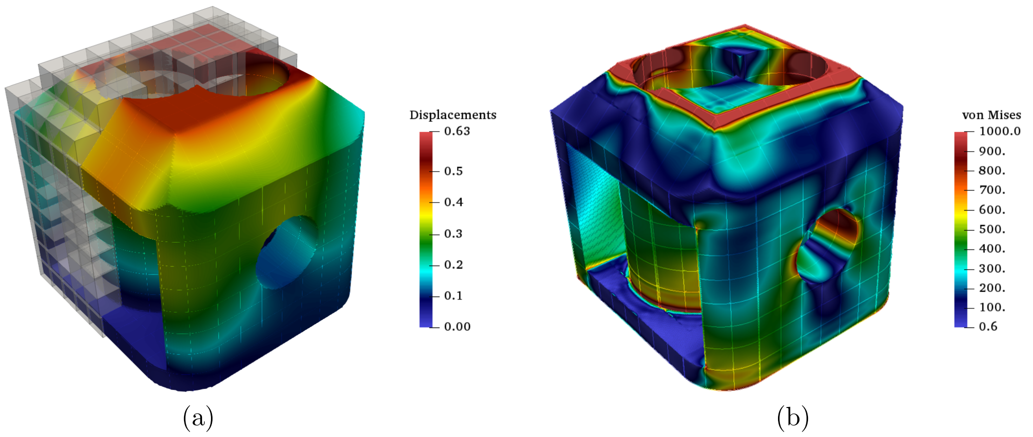


Figure 14.13: Qualitative results of the numerical simulation: (a) Displacements with finite cell mesh and (b) von Mises stresses.

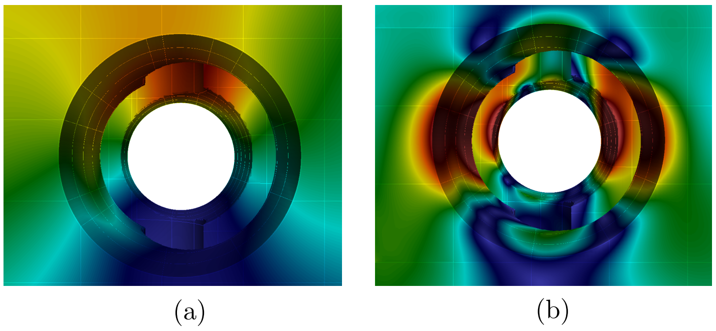


Figure 14.14: Magnification of borehole one: (a) Displacements and (b) von Mises stresses are similar for both, the original and the modified model.

14.4 Steel framework node

The following model of a steel framework joint serves as practical application for the solid-based procedural modeling. As illustrated in Figure 14.15 the joint consists of an IPE beam with a flange attached on its top surface. The flange connects the beam with two diagonal and one vertical pipe, or strut. Similar to the preceding examples the initial model is altered which yields five different variants (see Figure 14.15). Two modifications are created with geometrical changes of the initial geometry (a). More precisely, for model (b) the angle between the diagonal struts and the beam is changed, and for model (c) the diameters of the diagonal and vertical struts are increased and decreased, respectively. The other two models result from topological changes. For model (d) two fillets – representing weld seams – are added to the vertical flange[1], and for model (e) the flange has a hemispheric cutout.

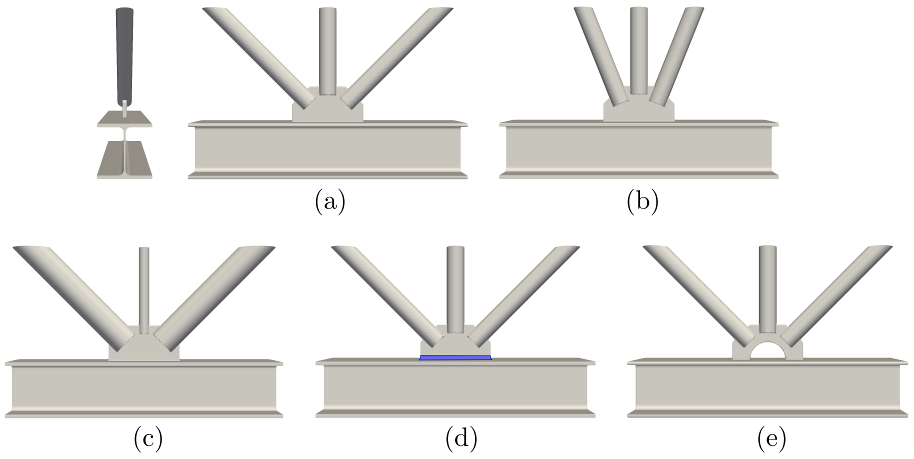


Figure 14.15: Variants of a steel framework node: (a) initial design, (b) changed angle of the diagonal struts, (c) changed diameter of the struts, (d) added weld seam at the flange, and (e) hemispherical cutout in the flange.

[1] *Remark:* This change might also be created by adapting the geometry. Yet, in the present case, additional primitives were added to the CSG tree.

For the numerical simulation, the computational domain is discretized with $20 \times 10 \times 10$ finite cells using hierarchical Legendre polynomials with $p = 5$. The subdivision depth of the integration octree is $k = 5$. The IPE beam is clamped on the left and right surface. The diagonal struts are loaded with a compressive force and the vertical strut with a tensile force in the corresponding normal directions. Additionally, all struts are loaded with a lateral force. The boundary conditions are applied in a weak sense, using the penalty method. For this, the required surfaces are recovered with the marching cubes algorithm directly from the model.

Figures 14.16 to 14.20 show the qualitative displacements and von Mises stresses for all five variants. It is noteworthy that all simulations are carried out with the same discretization.

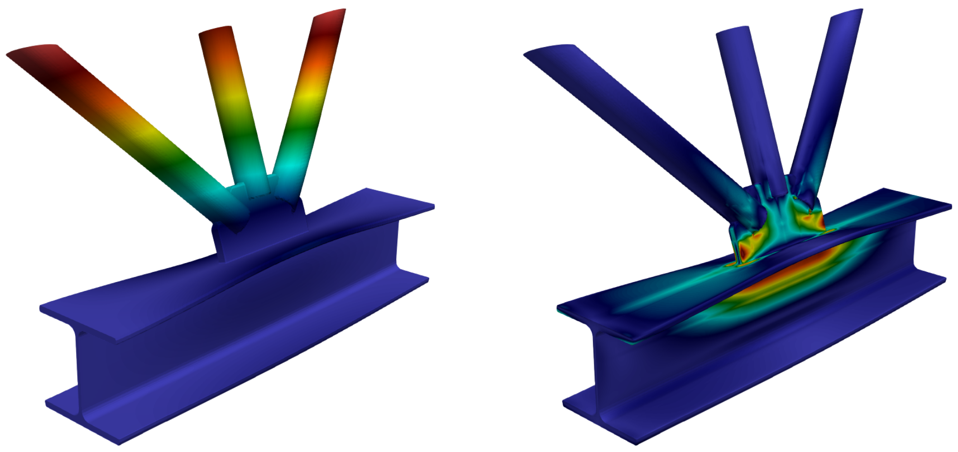


Figure 14.16: Qualitative displacements and von Mises stresses for the initial model.

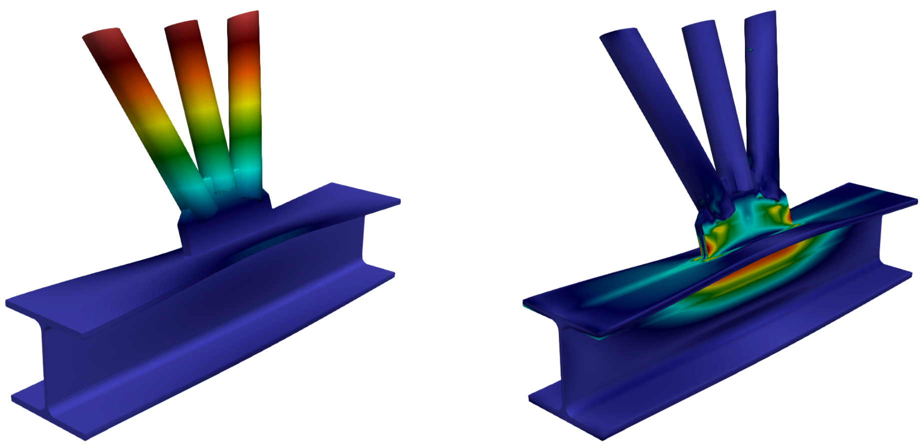


Figure 14.17: Qualitative displacements and von Mises stresses for the model with changed angle between the struts and the beam.

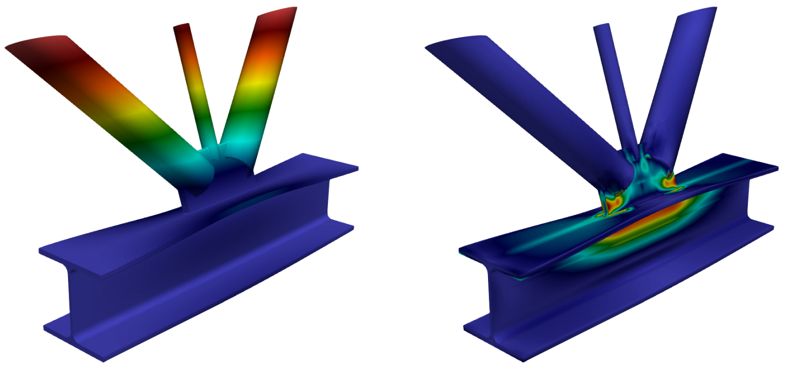


Figure 14.18: Qualitative displacements and von Mises stresses for the model with changed diameters of the struts.

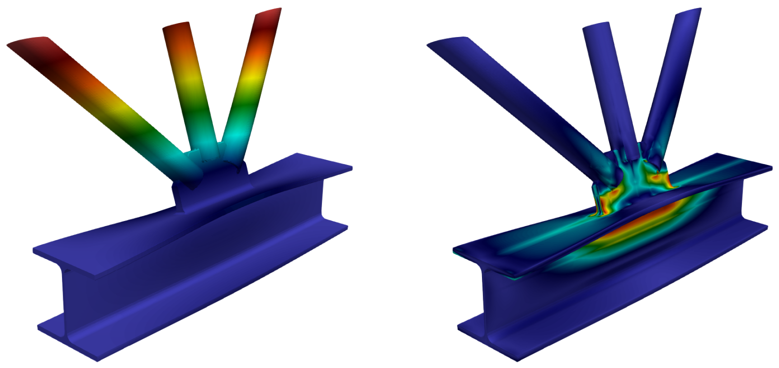


Figure 14.19: Qualitative displacements and von Mises stresses for the model with weld seams along the flange.

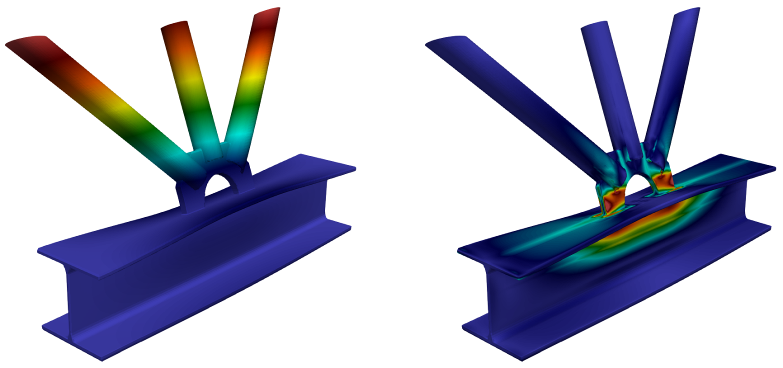


Figure 14.20: Qualitative displacements and von Mises stresses for the model with a hemispherical cutout.

Chapter 15

Applications of functionally graded materials based on V-models

As described in Section 12.1, the V-rep framework already offers a reliable and robust PMC for V-models that allows direct numerical simulations with homogeneous material distributions, similar to the preceding examples. Thus, the following five examples focus on the V-models' capability to model and represent functionally graded materials. The first, simple example of a cuboid with prescribed heterogeneous material serves as a verification for the method. The second example, a coupled heat, thermo-elastic simulation of a curved thermal protection tile, emphasizes the applicability to examples of engineering relevance. In contrast to the first two examples, the FGM of the third example is not modeled by a heterogeneous material distribution within the V-model, but with a fully resolved, continuously changing microstructure. In the fourth example, the underlying tiles of the microstructure of the third example are evaluated in terms of a homogenization, which are then used in the fifth example to perform a simulation on a homogenized constructive FGM.

15.1 Cuboid with sinusoidal material distribution

As a benchmark problem, a cuboid with varying material distribution in $z-$direction is chosen[1]. The cuboid is a trivariate B-spline and is created with GuIrit [73]. As the spline basis functions are initially linear in each direction, they cannot represent the material function $E(z)$. For this reason, a degree elevation to $r = 3$ and subsequent multiple knot insertions in $z-$direction were carried out, yielding a knot-vector of $W = [0, 0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1, 1]$. The control points in $z-$direction are depicted in Figure 4.2a. The cuboid has assigned a constant Poisson ratio of $\nu = 0.3$. For the functionally graded Young's modulus, an analytical function is given

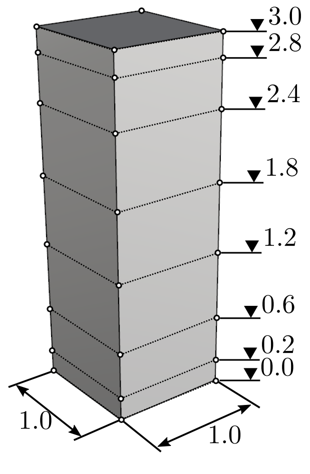
$$E(z) = 10^6 + 5 \cdot 10^4 \cdot sin(z\pi)\,. \tag{15.1}$$

The material 'coordinates' μ_i^E of the control points are computed with the least squares method, using $n_s = 100$ sample points, according to Equation (9.3)

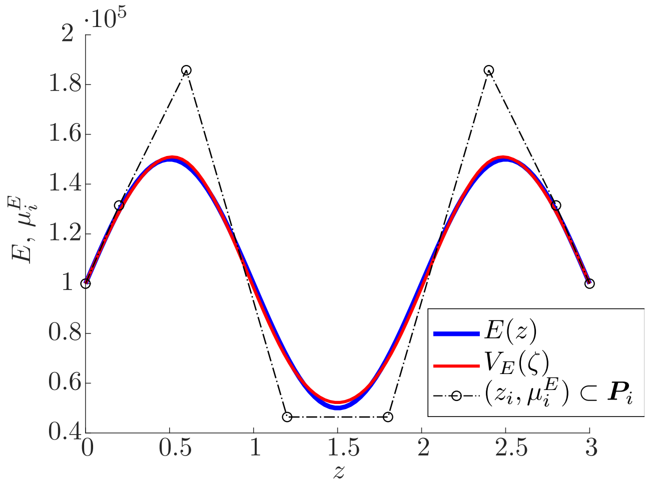
$$\boldsymbol{\mu}^E = [100000,\ 131438,\ 185772,\ 46415,\ 46415,\ 185772,\ 131438,\ 100000]\,. \tag{15.2}$$

Figure 15.1 depicts the dimensions of the cuboid with the three-dimensional control point mesh and the resulting least squares approximation of the material distribution.

[1] *Remark:* This benchmark example is chosen to be of simple shape to able to obtain a highly precise reference solution. Yet, as the structure is embedded in a larger domain, the solution is non-trivial for immersed boundary methods.



(a)



(b)

Figure 15.1: (a) Dimensions of the cuboid. (b) Least squares fitting of the material function $E(z)$ yields the material coordinates μ_i^E.

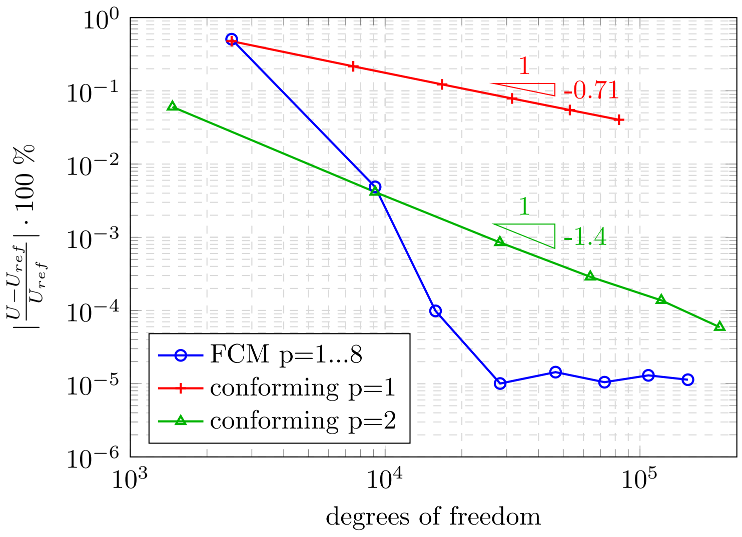
For the simulation, the cuboid is embedded into a slightly larger fictitious domain (± 0.1 in each direction, ergo $1.2 \times 1.2 \times 3.2$) and discretized by $6 \times 6 \times 16$ finite cells deploying hierarchic Legendre shape functions. Homogeneous Dirichlet boundary conditions are applied in $x-$direction on the left, in $y-$direction on the front, and in $z-$direction on the bottom surface using the penalty method. The cuboid is loaded on the top surface with a traction of $f = -1000$ in $z-$direction.

To prove the validity of the FCM for multi-material FGM, a convergence study is carried out. The polynomial degrees of the Legendre ansatz function are increased from $p = 1$ to $p = 8$, and the corresponding strain energies are compared to a reference solution U_{ref}, which was computed by a boundary-conforming $p-$FEM analysis. A composed integration is used, which can accurately recover the cuboid's exact shape, thus minimizing integration errors – similar to the smart octree [50]. In order to compare the convergence behavior of the FCM with the standard FEM, two additional convergence studies using $h-$refinement are carried out on boundary conforming FEM discretizations, with polynomial degrees of $p = 1$ and $p = 2$, respectively.

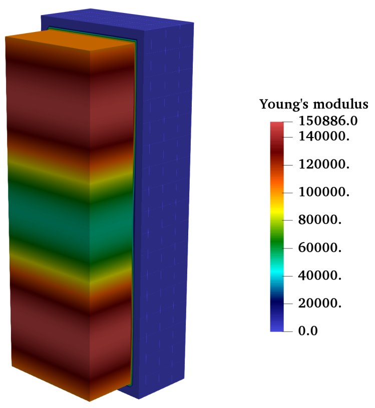
As depicted in Figure 15.2a, the FCM shows a pre-asymptotic exponential convergence until it reaches the numerical precision of the underlying Irit library at $p = 4$, whereas the $h-$studies show algebraic convergence – as expected[2]. Obviously, in terms of degrees of freedom, the FCM outperforms classical $h-$versions.

Figure 15.3 shows the displacements and the von Mises stresses on the surface of and inside the deformed cuboid. As can be seen, the lower stiffness regions are undergoing a more extensive deformation, leading to a bulge in which the stresses are minimal.

[2] *Remark:* Since the relative error is stated in percent, the actual precision is the order of 10^{-7}, which corresponds to the accuracy of the geometric modeler Irit which uses single precision.

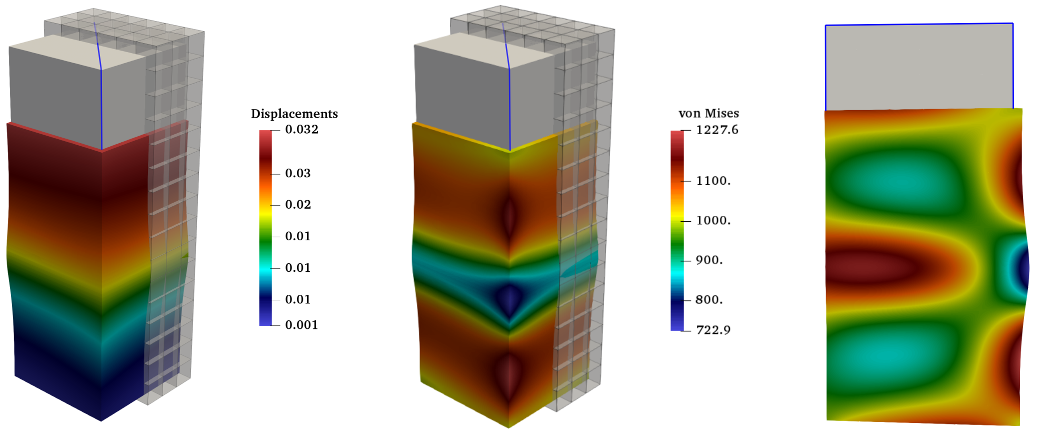


(a)



(b)

Figure 15.2: (a) Relative error in the strain energy for the FCM for polynomial degrees $p = 1...8$ and for boundary fitted $h-$studies with $p = 1$ and $p = 2$, respectively. (b) Distribution of the Young's inside the cuboid and in the fictitious domain.



(a) (b) (c)

Figure 15.3: Results for the linear elastic simulation: (a) Displacements and (b) von Mises stresses warped around the undeformed cuboid (grey block) embedded into the finite cell mesh. (c) Diagonal cross section (at the blue frame) showing the stress distribution inside the cuboid. The deformation is scaled by a factor of $s = 20$.

15.2 Curved thermal shielding tile

To demonstrate the applicability for real-world problems, coupled simulations for three different variants of a curved thermal shielding tile are carried out. Such tiles are needed for high-temperature applications, such as re-entrance shielding for spacecraft or the inner coating of fusion power plants. The tiles consist of a load-carrying zone made of titanium Ti and an insulating zone made of porous silica SiO_2 with a porosity of 70%. Both materials have similar melting points of $\Theta_{Ti} = 1.668°C$ for titanium and $\Theta_{SiO_2} = 1.710°C$ for silica, which allows a fabrication with additive manufacturing using e.g., powder bed laser melting.

Particular focus is laid on the continuity of the transition zone between these materials. The first discontinuous tile consists of two distinct domains where both domains are assumed to be homogeneous titanium and silica, respectively, i.e., there is no transition zone. Hence, the first tile is not an FGM but a composite material. The material is changed C^0-continuously in the second tile, and C^1-continuously in the third tile. In order to evaluate the stresses under a heat load, a coupled simulation is carried out. An initial thermal simulation provides the temperature distribution, which is then used to apply thermal strains for the subsequent thermo-elastic simulation. Consequently, the model will deform due to the different thermal expansion ratios. However, this deformation is hindered by the different Young's moduli in the transition zone, then leading to internal stresses.

The underlying V-model (see Figure 15.4) consists of one V-cell and was generated by extruding a curved two-dimensional B-spline surface $5\,cm$ in $z-$direction. The control point mesh of the curved surface is defined as follows[3]

$$\boldsymbol{P}_i^{surface} = \begin{bmatrix} 0 & 0 & 0 & 0 & \quad 4 & 4 & 7 & 7 & \quad 8 & 8 & 14 & 14 & \quad 12 & 12 & 21 & 21 \\ 0 & 10 & 20 & 30 & \quad 0 & 10 & 20 & 30 & \quad 0 & 10 & 20 & 30 & \quad 0 & 10 & 20 & 30 \\ 0 & 0 & 7 & 7 & \quad 0 & 0 & 7 & 7 & \quad 0 & 0 & 0 & 0 & \quad 0 & 0 & 0 & 0 \end{bmatrix}. \tag{15.3}$$

The knot vectors in $x-$ and $y-$ direction read $U = V = [0, 0, 0, 0.5, 1, 1, 1]$. Consequently, the surface has polynomial degrees of $p_x = p_y = 2$.

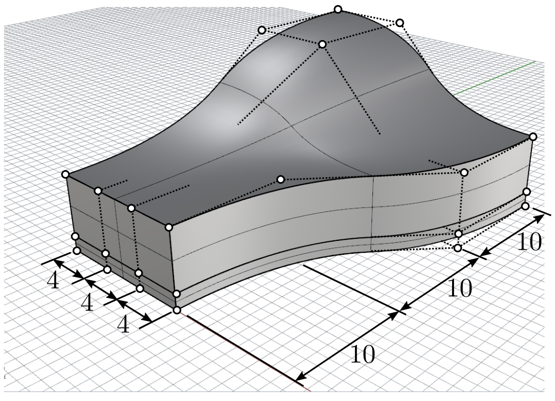
The extrusion yields a V-cell with a polynomial degree of $p_z = 1$ and a knot vector $W = [0, 0, 1, 1]$. Thus, the volumetric control point mesh consists of twice the initial mesh of the surface, where the second half of the control points have an offset of $\Delta z = 5$ cm. For the C^1- continuous tile, the polynomial degree in $z-$direction is increased to $p_z = 2$. To construct the discontinuous tile, the V-model was split at $\Delta z_{split} = 1.25\,cm$ using knot-insertion. The knot vectors and the offsets of the control points in $z-$direction for all tiles read as follows

$$\boldsymbol{Z}_{Discont.} = [0,\, 0,\, 0.25,\, 0.25,\, 1,\, 1]\,, \tag{15.4}$$
$$\boldsymbol{Z}_{C^0} = [0,\, 0,\, 0.15,\, 0.35,\, 1,\, 1]\,, \tag{15.5}$$
$$\boldsymbol{Z}_{C^1} = [0,\, 0,\, 0,\, 0.05,\, 0.29,\, 0.5,\, 1,\, 1,\, 1]\,, \tag{15.6}$$
$$\boldsymbol{\Delta z}_{Discont.} = [0,\, 1.25,\, 1.25,\, 5]\,, \tag{15.7}$$
$$\boldsymbol{\Delta z}_{C^0} = [0,\, 0.75,\, 1.75,\, 5]\,, \tag{15.8}$$
$$\boldsymbol{\Delta z}_{C^1} = [0,\, 0.2,\, 0.8,\, 1.7,\, 3.6,\, 5]\,. \tag{15.9}$$

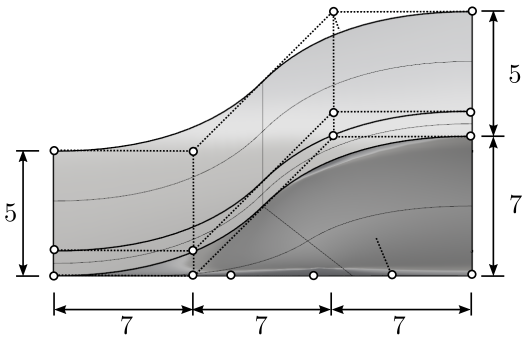
The resulting material distributions are depicted in Figure 15.5 exemplary for the Young's modulus. The other material properties are distributed similarly. The parameters for the B-splines were chosen such that the integral of the material over the thickness is equal for all three tiles.

To perform the coupled simulation, four different material parameters are required for both materials (see Table 15.1). The properties were taken from AZO Materials and averaged if necessary [120]. Due to the porosity of the silica, the respective Young's modulus E_{SiO_2} and the thermal conductivity κ_{SiO_2}

[3] *Remark:* Blank columns indicate a new row of control points in $x-$direction



(a)



(b)

Figure 15.4: Model dimensions in (cm) and control point mesh of the discontinuous tile in (a) isometric view, and (b) from the back side.

must be adapted. This is implemented with the Gibson-Ashby criteria, which provide simple formulas to estimate the properties based of the porosity [121, 122]

$$\kappa_r = (1-\phi)^{3/2}\,, \tag{15.10}$$

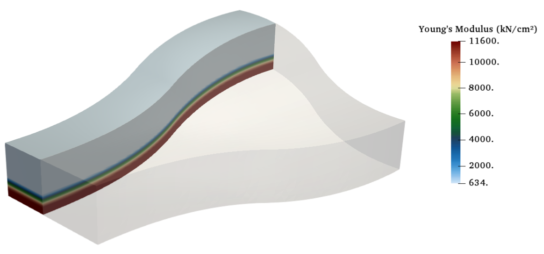
$$E_r = (1-\phi)^2\,, \tag{15.11}$$

where ϕ is the porosity (in this example $\phi = 0.7$) and κ_r and E_r are the weighting factors for the thermal conductivity and Young's modulus, respectively. By contrast, the Poisson's ratio ν_{SiO_2} and the thermal expansion α_{SiO_2} require no adjustment [123].

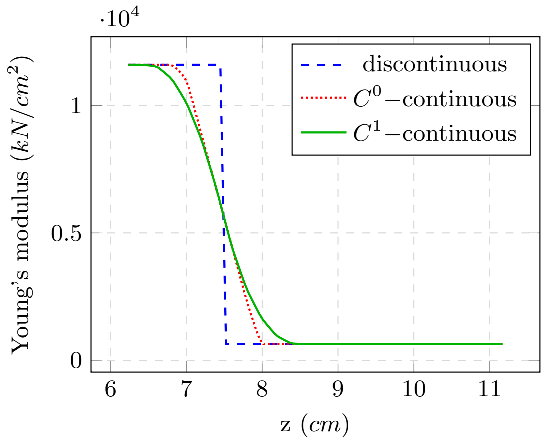
Property	Symbol	Titanium	Silica (70% porosity)	Units
Young's Modulus	E	$11,600$	634	kN/cm^2
Poisson's Ratio	ν	0.36	0.17	$-$
Thermal conductivity	κ	0.216	$2.3 \cdot 10^{-3}$	W/cmK
Thermal expansion	α	$8.6 \cdot 10^{-6}$	$6.5 \cdot 10^{-7}$	$1/K$

Table 15.1: Material properties of titanium and porous silica for the coupled simulation.

The simulation is based on $16 \times 23 \times 9$ finite cells with a polynomial degree of $p = 3$ and an integration subdivision depth of $n = 3$. For the preceding heat simulations, Dirichlet boundary conditions are applied with a prescribed heat of $1000°C$ on the top surface and $20°C$ on the bottom surface. The resulting temperature inside the tiles is then transferred as a body strain to perform a thermo-elastic simulation. Additionally, the tiles are clamped at the bottom surface. Since the higher-order shape functions cannot represent jumps in the material distribution, the simulations of the discontinuous tile are carried out on two separate meshes – one for each domain – which are 'glued' together in a weak sense along their coupling surface using the penalty method [106]. Both meshes are equally discretized with $16 \times 23 \times 9$ finite cells. Thus, on each mesh, only material jumps from the physical into the fictitious domain appear, which can be decently represented by the FCM. To resolve the critical regions, the multi-level *hp*-method [41] is used with a refinement depth of two around the coupling surface for the two meshes of the discontinuous tile and a refinement depth of one in the transition zones of the continuous tiles (see Figure 15.6).

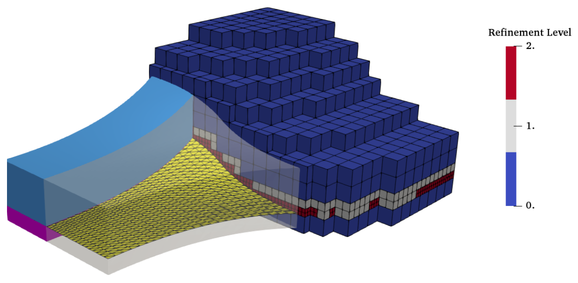


(a)

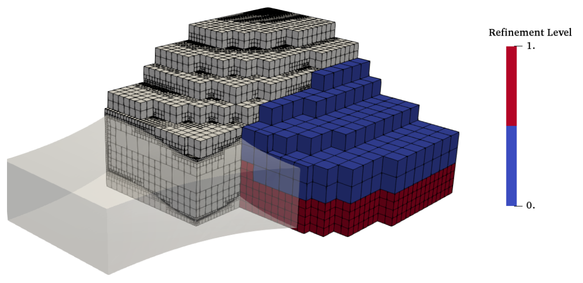


(b)

Figure 15.5: Material distribution of the Young's modulus (a) inside the C^0-continuous tile and (b) plotted at $x = 5\,cm$, $y = 25\,cm$ over the thickness.



(a)



(b)

Figure 15.6: Discretizations of (a) the discontinuous tile – the mesh is refined twice around the coupling surface (yellow), which divides the upper (light blue) and lower domain (purple), and (b) the C^0-continuous tile, where the FCM mesh is refined once in the transition zone (blue cells are unrefined, and red cells refined once). The grey mesh in the background corresponds to the octree for the integration. The C^1-continuous tile is meshed and refined analogously.

To visualize the results inside the tiles, a cut through the model is investigated at $x = 5\,cm$. Figure 15.7 shows the temperature distribution and displacements of the C^0-continuous tile. However, the temperature and the displacement distributions are almost identical for all tiles. More relevant are the stress distributions. As shown in Figure 15.8, a stress concentration occurs at the coupling surface of the discontinuous tile. Figures 15.9 and 15.10 plot the temperature distribution, displacements, and stresses over the height at $x = 5\,cm$ and $y = 25\,cm$. The discontinuous material distribution yields a C^0-continuous heat and displacement distribution, which then entails a discontinuous stress distribution with a maximum peak at the interface, i.e., the coupling surface. This stress concentration is critical as it will potentially cause delamination. The C^0-continuous material distribution, on the other hand, ensures a continuous and much smaller stress distribution throughout the entire domain. This effect can be further augmented by using a C^1-continuous material distribution. Continuous materials, on the other hand, involve a larger heat flux. For the one-dimensional case, the thermal resistance is reduced to $\sim 86\%$ for the C^0-continuous and $\sim 75\%$ for the C^1-continuous material with respect to the discontinuous material distribution.

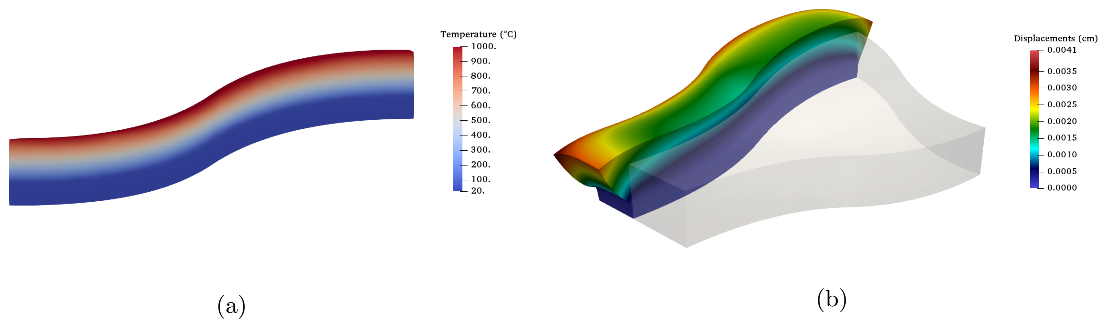


Figure 15.7: Cut through the C^0-continuous tile at $x = 5\,cm$: (a) Temperature distribution, and (b) displacements warped by a scaling factor of $s = 1000$.

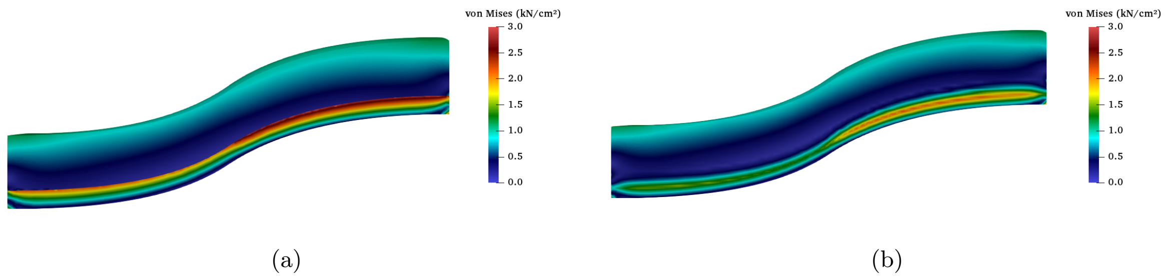
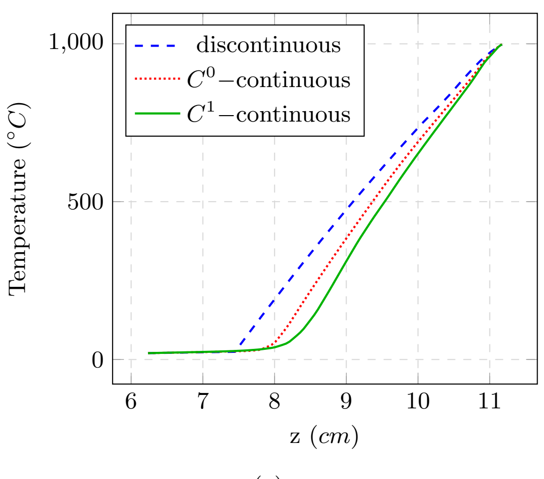
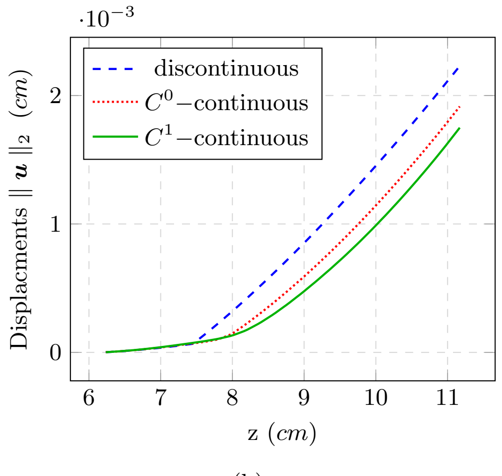


Figure 15.8: Cut through the tiles at $x = 5\,cm$ showing the von Mises for (a) the discontinuous, and (b) the C^0-continuous thermal shielding tile. The stress distribution of the C^1-continuous tile looks very similar to the C^0-continuous tile.



(a)



(b)

Figure 15.9: Comparison of (a) the temperature, and (b) the displacements of the discontinuous and continuous tiles at $x = 5\,cm$, $y = 25\,cm$ over the thickness.

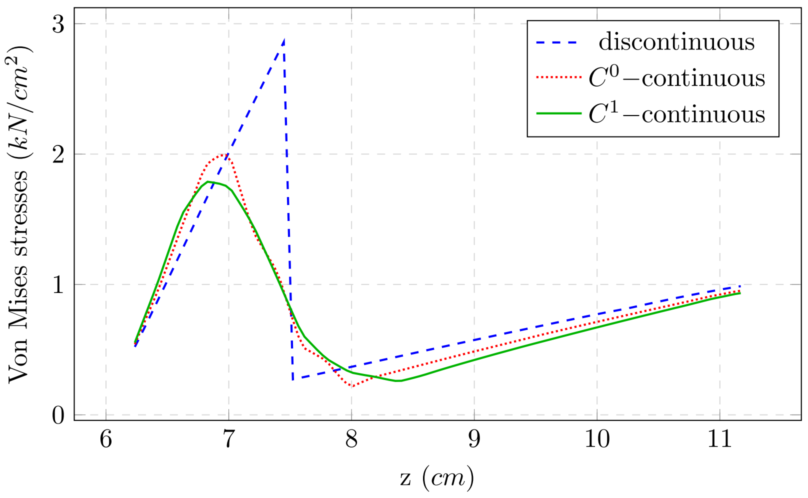


Figure 15.10: Comparison of the von Mises stresses of the discontinuous and continuous tiles at $x = 5\,cm$, $y = 25\,cm$ over the thickness.

15.3 Functionally graded microstructure

As an example of a single-material FGM, a linear-elastic simulation on the microstructure, depicted in Figure 9.3, is carried out. It resembles a porous, foam-like microstructure stiffened by an outer shell. To generate this model, a continuously changing microstructure is created with GuIrit. Different unit tiles – each composed of seven trivariate B-splines – are used to tile a parametrically described ruled body (see Figure 7.8). The unit tiles have a growing stiffness from bottom to top, realized by an increasing diameter of the rod in $x-$direction[4]. The resulting microstructure consists of $6 \times 6 \times 9$ unit tiles and

[4] *Remark:* Due to the rotation of the ruled body, the stiffer direction is changing from bottom to top

an overall number of 2268 trivariate B-splines, or V-cells. A direct simulation on this V-model leads to unreasonably high runtimes due to the complexity of the inverse mapping. However, since in this example, the FGM is not modeled within the individual V-cells, but as a single-material continuously changing microstructure, it is possible to carry out a simulation significantly faster on an auxiliary B-rep model. To this end, the V-cells' B-spline surfaces are extracted, and inner surfaces, between consecutive V-cells, are deleted. The resulting B-rep model consists of 8064 B-spline surfaces. With a B-rep CAD tool (Rhinoceros®), the shell is added as a B-rep volume and combined with the microstructure using the Boolean union operation. Subsequently, the microstructure on the outer side of the shell is trimmed away using the trimming operation with the shell volume's outer surface. Finally, the computational model is extracted with a Boolean intersection with the computational domain. Figure 15.11 depicts the selection of the computational domain and the final model with the respective surfaces for the boundary conditions.

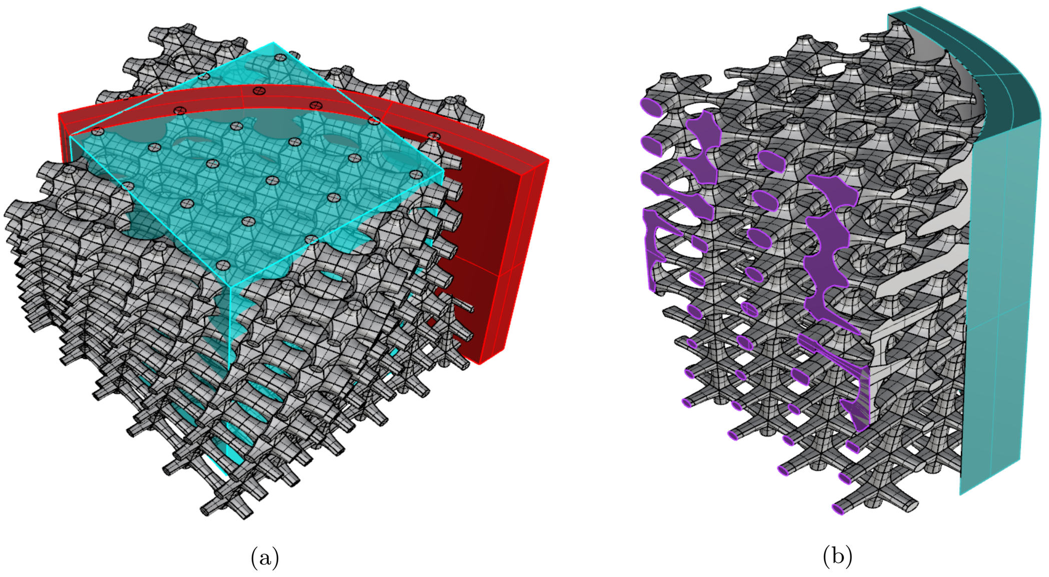

(a) (b)

Figure 15.11: (a) Selection of the computational domain (turquoise). An outer shell (red) is embedded into a microstructure. (b) Intersection of the microstructure with $\Omega_\cup$ leading to the physical domain Ω_{phy}. Boundary conditions are applied on the highlighted intersection surfaces.

For the simulation, homogeneous Dirichlet boundary conditions are applied on the cutting planes of the shell (see Figure 15.11b – highlighted in turquoise). The top and bottom surface restrict the displacements in $x-$ and $z-$direction, and the front and back surface restrict the displacements in $x-$ and $y-$direction. A prescribed deflection of $\overline{u}_x = 0.1$ is applied on the surfaces on the left side (see Figure 15.11b – highlighted in purple). All boundary conditions are enforced in a weak sense with the penalty method. For the material properties, a Young's modulus of $E = 100$ and a Poisson's ratio of $\nu = 0.3$ are chosen. The computational domain is discretized with $20 \times 20 \times 20$ finite cells, employing Legendre polynomials with a polynomial degree $p = 4$. The subdivision depth of the octree for the integration is set to $n = 4$.

Figures 15.12 and 15.13 show the qualitative displacements and the von Mises stresses. Since the shape functions are poorly suited to represent holes inside one finite cell (i.e., 'material–void–material'), the microstructure needs to be finely discretized. An alternative is the local enrichment with shape functions specifically designed to represent the material jumps, as presented in [124].

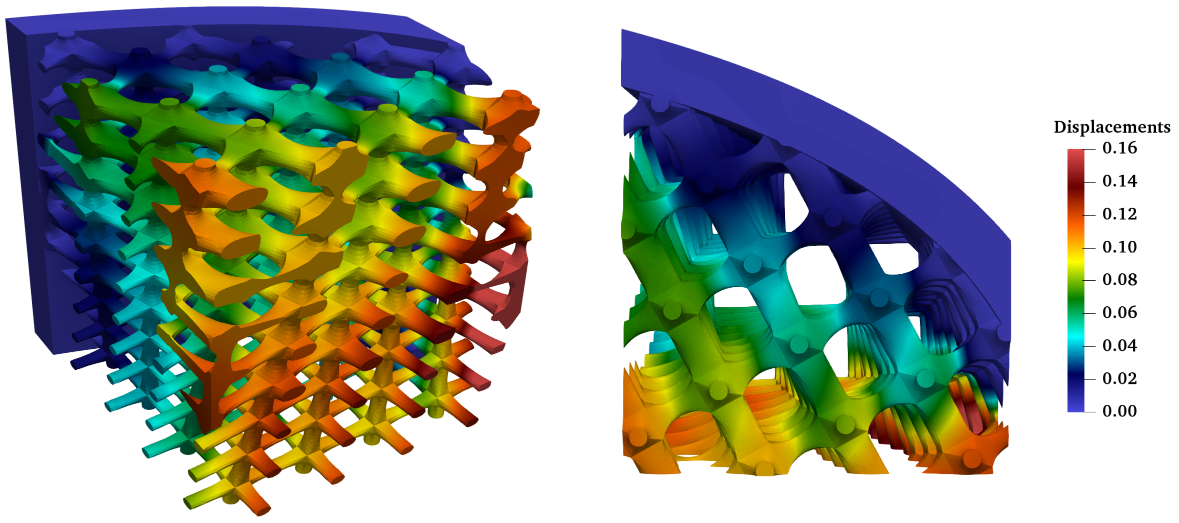


Figure 15.12: Qualitative displacements.

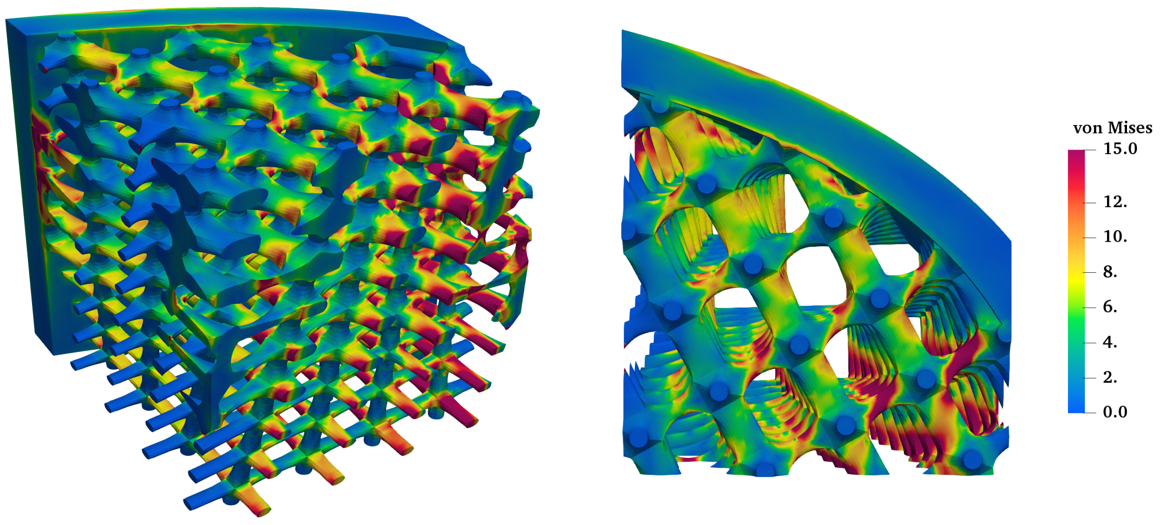


Figure 15.13: Qualitative von Mises stresses.

15.4 Material characterization database for unit tiles

A fully resolved numerical simulation of a microstructure – as presented in Example 15.3 – is computationally very demanding in both memory consumption and simulation time. For large-scale microstructures (as in Example 15.5), fully-resolved computations need to be carried out on a high-performance computer, or might even be not applicable at all. A remedy is offered by homogenization. As explained in Section 12.3, for a functionally graded microstructure it is sufficient to compute the effective material tensors $\boldsymbol{C}^*_{Ti}$ only for several representative unit tiles, and interpolate the material properties in-between, according to the parametrization of the microstructure.

Two parameters are used to characterize the unit tiles in the Examples 15.3, 15.4 and 15.5, the diameter d of the rod in $x-$direction and rotation angle ψ around the $z-$axis. To compute the respective microscopic material behaviors, homogenization simulations are carried out for unit tiles with three different configurations of the diameter of the rod in $x-$direction ($d = 0.2\,mm$, $d = 0.3\,mm$, and $d = 0.4\,mm$), yielding the unrotated, effective material tensors $\boldsymbol{C}^*_{Ti}$.

For the homogenization simulations, the microstructure material is considered to be steel with a Young's modulus of $E = 210\,GPa$, and a Poisson's ratio of $\nu = 0.3$. Each tile is discretized with $11 \times 11 \times 11$ finite cells of polynomial degree $p = 5$. For the domain integration, the moment-fitting approach with the depth of an underlying octree of $d = 6$ is chosen. As the structures under consideration are, in good approximation, geometrically periodic, periodic boundary conditions are the natural choice for transferring the macroscopic quantities to the microscopic unit cells.

Figure 15.14 shows the displacement fields under shear load for the unit tiles in the unrotated configuration. The resulting homogenized material tensors for the tiles 1, 2 and 3 are summarized in the Equations (15.12), (15.13) and (15.14), respectively. One can identify different material behaviors, which is expected due to the geometrical features of the respective unit tiles. The orientation and the thickness of the rods has an important effect on the final material behavior. Tile 1 shows a cubic macroscopic material symmetry with three independent elasticity coefficients [125], namely C_{11}, C_{12} and C_{44}

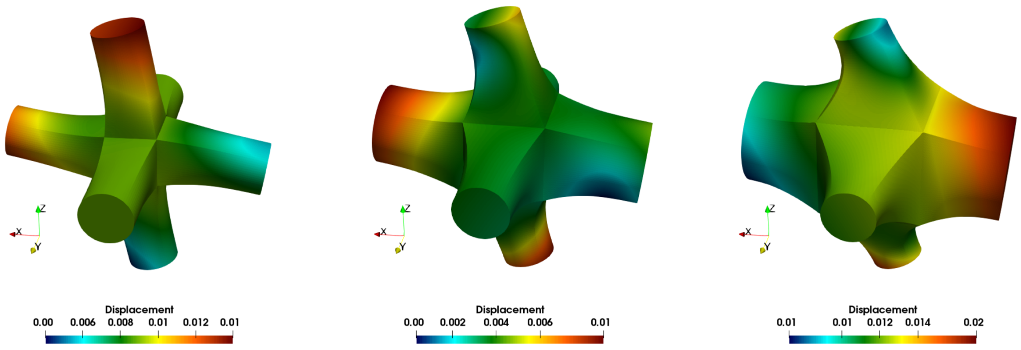
$$\boldsymbol{C}^*_{T1} = \begin{bmatrix} 7895.81 & 432.89 & 432.89 & 0 & 0 & 0 \\ 432.89 & 7895.81 & 432.89 & 0 & 0 & 0 \\ 432.89 & 432.89 & 7895.81 & 0 & 0 & 0 \\ 0 & 0 & 0 & 200.71 & 0 & 0 \\ 0 & 0 & 0 & 0 & 200.71 & 0 \\ 0 & 0 & 0 & 0 & 0 & 200.71 \end{bmatrix} \tag{15.12}$$

Due to the stiffer direction in $x-$direction, tile 2 and 3 show a tetragonal effective material symmetry with $C_{11}, C_{22}, C_{44}, C_{55}, C_{12}$ and C_{23} as independent entries:

$$\boldsymbol{C}^*_{T2} = \begin{bmatrix} 18246.81 & 1026.56 & 1026.56 & 0 & 0 & 0 \\ 1026.56 & 11066.80 & 659.81 & 0 & 0 & 0 \\ 1026.56 & 659.81 & 11066.80 & 0 & 0 & 0 \\ 0 & 0 & 0 & 769.49 & 0 & 0 \\ 0 & 0 & 0 & 0 & 590.69 & 0 \\ 0 & 0 & 0 & 0 & 0 & 769.49 \end{bmatrix} \tag{15.13}$$

$$\boldsymbol{C}^*_{T3} = \begin{bmatrix} 33809.00 & 2037.73 & 2037.73 & 0 & 0 & 0 \\ 2037.73 & 14770.28 & 997.14 & 0 & 0 & 0 \\ 2037.73 & 997.14 & 14771.08 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2022.10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1375.86 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2022.17 \end{bmatrix} \tag{15.14}$$

The material tensors $\boldsymbol{C}'_{Ti}$ for the second changing parameter – the rotation around the $z-$axis – can be computed by a coordinate transformation, and thus require no homogenization simulations. The Bond-Transformation matrices [126] can be used to rotate the effective elasticity tensor by a matrix-matrix multiplication. Assume the following ordering of the macroscopic stresses σ^M_{ij} and strains ε^M_{ij} in the



(a) Unit tile 1. (b) Unit tile 2. (c) Unit tile 3.

Figure 15.14: Homogenization simulation of the unrotated unit tiles: Displacement field, warped by a scaling factor of $s = 10$.

Voigt notation

$$\begin{bmatrix} \sigma_{11}^M \\ \sigma_{22}^M \\ \sigma_{33}^M \\ \sigma_{12}^M \\ \sigma_{23}^M \\ \sigma_{13}^M \end{bmatrix} = \begin{bmatrix} C_{11}^* & C_{12}^* & C_{13}^* & C_{14}^* & C_{15}^* & C_{16}^* \\ C_{12}^* & C_{22}^* & C_{23}^* & C_{24}^* & C_{25}^* & C_{26}^* \\ C_{13}^* & C_{23}^* & C_{33}^* & C_{34}^* & C_{35}^* & C_{36}^* \\ C_{14}^* & C_{24}^* & C_{34}^* & C_{44}^* & C_{45}^* & C_{46}^* \\ C_{15}^* & C_{25}^* & C_{35}^* & C_{45}^* & C_{55}^* & C_{56}^* \\ C_{16}^* & C_{26}^* & C_{36}^* & C_{46}^* & C_{56}^* & C_{66}^* \end{bmatrix} \begin{bmatrix} \varepsilon_{11}^M \\ \varepsilon_{22}^M \\ \varepsilon_{33}^M \\ \varepsilon_{12}^M \\ \varepsilon_{23}^M \\ \varepsilon_{13}^M \end{bmatrix} . \tag{15.15}$$

Then, the transformation of the effective elastic tensor reads as follows

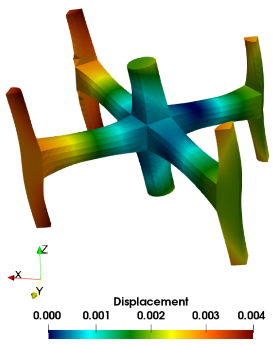
$$\boldsymbol{C}' = \boldsymbol{M}\boldsymbol{C}^*\boldsymbol{N}^{-1} , \tag{15.16}$$

where $\boldsymbol{C}^*$ is the unrotated and $\boldsymbol{C}'$ the rotated effective elasticity tensor. $\boldsymbol{M}$ and $\boldsymbol{N}$ are the Bond-stress and the Bond-strain transformation matrices, respectively. For a rotation around the $z-$axis, the Bond strain and stress matrices are defined as follows

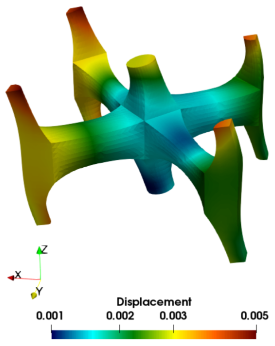
$$\boldsymbol{M} = \begin{bmatrix} cos^2(\alpha) & sin^2(\alpha) & 0 & sin(2\alpha) & 0 & 0 \\ sin^2(\alpha) & cos^2(\alpha) & 0 & -sin(2\alpha) & 0 & 0 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \\ -\frac{sin(2\alpha)}{2} & \frac{sin(2\alpha)}{2} & 0 & cos(2\alpha) & 0 & 0 \\ 0 & 0 & 0 & 0 & cos(\alpha) & -sin(\alpha) \\ 0 & 0 & 0 & 0 & sin(\alpha) & cos(\alpha) \end{bmatrix} \tag{15.17}$$

$$\boldsymbol{N} = \begin{bmatrix} cos^2(\alpha) & sin^2(\alpha) & 0 & \frac{sin(2\alpha)}{2} & 0 & 0 \\ sin^2(\alpha) & cos^2(\alpha) & 0 & -\frac{sin(2\alpha)}{2} & 0 & 0 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \\ -sin(2\alpha) & sin(2\alpha) & 0 & cos(2\alpha) & 0 & 0 \\ 0 & 0 & 0 & 0 & cos(\alpha) & -sin(\alpha) \\ 0 & 0 & 0 & 0 & sin(\alpha) & cos(\alpha) \end{bmatrix} \tag{15.18}$$

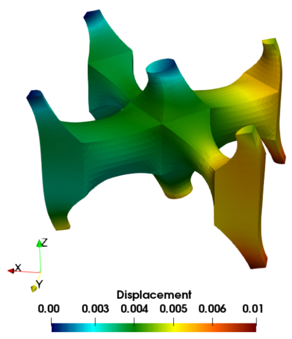
As a numerical verification for the correct choice of the RVEs, additional homogenization simulations are carried out for a rotation angle of $\psi = 45^\circ$ and compared to the results to the Bond transformations (see Figure 15.15).



(a) Unit tile 1.



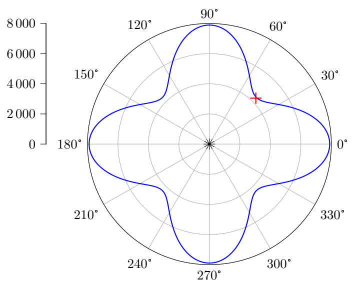
(b) Unit tile 2.



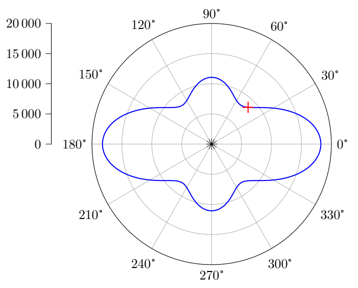
(c) Unit tile 3.

Figure 15.15: Homogenization simulation of the rotated unit tiles at $\psi = 45^\circ$: Displacement field, warped by a scaling factor of $s = 10$.

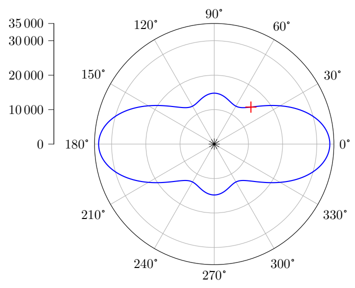
Figure 15.16 shows exemplarily the material property C_{11} of the three unit tiles for an arbitrary rotation angle ψ. Here, also the material property C_{22} can be read out at a phase shift of $\Delta\psi = \pm 90^\circ$. The results of the verification simulations at $\psi = 45^\circ$ are denoted with red markers ('+'). As can be seen, tile 1 is symmetric in both $x-$ and $y-$direction, whereas tile 2 and 3 have a stiffer direction. Moreover, as expected, the absolute stiffness is rising with increasing diameter. For a more detailed overview of all rotated material properties, please refer to [15].



(a) Unit tile 1.



(b) Unit tile 2.



(c) Unit tile 3.

Figure 15.16: Material property C_{11} in (MPa) of the unit tiles for a rotation around the $z-$axis.

Given a set of different (an-)isotropic unit tiles that can be used to construct such microstructures, it is possible to create a look-up table of homogenized materials, which can then be utilized to simulate different macroscopic load cases. Table 15.2 is a snippet of such a look-up table, and shows the effective elasticity tensors for the two varying material properties. The material properties in-between can be interpolated. This Table 15.2, will be used in the following Example 15.5 to compute a large-scale microstructure with interpolated homogenized material properties.

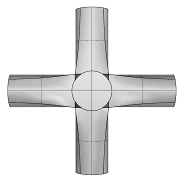
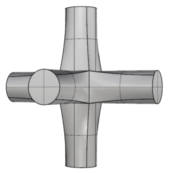
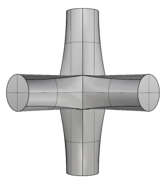
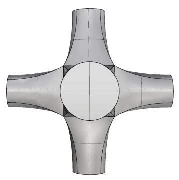
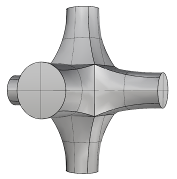
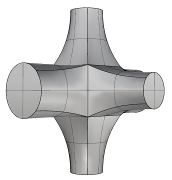
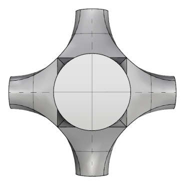
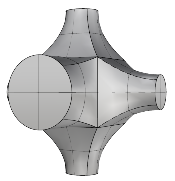
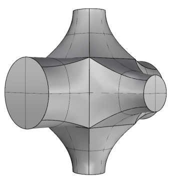
		Rotation around the $z-$axis		
		0°	22.5°	45°
Diameter of the rod in $x-$direction	$0.2\,mm$	 $C_{11} = 7.9\ GPa$ $C_{22} = 7.9\ GPa$	 $C_{11} = 6.1\ GPa$ $C_{22} = 6.1\ GPa$	 $C_{11} = 4.4\ GPa$ $C_{22} = 4.4\ GPa$
	$0.3\,mm$	 $C_{11} = 18.2\ GPa$ $C_{22} = 11.1\ GPa$	 $C_{11} = 14.2\ GPa$ $C_{22} = 9.1\ GPa$	 $C_{11} = 8.6\ GPa$ $C_{22} = 8.6\ GPa$
	$0.4\,mm$	 $C_{11} = 33.8\ GPa$ $C_{22} = 14.8\ GPa$	 $C_{11} = 26.5\ GPa$ $C_{22} = 13.0\ GPa$	 $C_{11} = 15.2\ GPa$ $C_{22} = 15.2\ GPa$

Table 15.2: Exemplary look-up table for the effective elasticity tensors (here represented by C_{11} and C_{22}) for changing diameters of the rod in $x-$direction and rotations around the $z-$axis.

15.5 Homogenized microstructure

Based on the unit tiles' material characterization in Example 15.4, a linear-elastic simulation with homogenized material is carried out on a large microstructure. To this end, the model of Example 15.3 is considered to be a part of a larger structure, as illustrated in Figure 15.17. Similar to Example 15.3, the corresponding geometric parts are modeled as B-rep models. For the simulation, the model is subdivided into an outer shell and an infill. The shell is modeled as solid isotropic material. In contrast, the infill is a homogenized microstructure which continuously changes the two known parameters: (a) the rotation angle ψ around the $z-$axis varies from 0° at the bottom to 90° at the top, and (b) the diameter of the rod d (in x-direction at $z = 0$) increases from the center axis of the infill $d = 0.2\,mm$ towards the interface of the shell $d = 0.4\,mm$. A uniaxial compression state is achieved by applying a uniform deflection of $\overline{u}_z = -1.0$ on the top surface and restricting the displacements in $z-$direction on the bottom surface. Three additional point-bearings block the rigid body motions.

For the simulation, the model is embedded into a fictitious domain and discretized with $15 \times 15 \times 15$ finite cells that employ hierarchic Legendre functions with a polynomial degree of $p = 4$. For integration, the moment-fitting approach is used. The underlying octree has a subdivision depth of $k = 4$. At the interface between shell and infill, the mesh is refined to capture the material discontinuity. As the unit tiles' homogenization was carried out with periodic boundary conditions, the behavior at the interface between the shell and the infill is not captured precisely. However, the affected domain is small compared to the overall structure. Thus, the introduced error is negligible. If the microscopic stress state at the

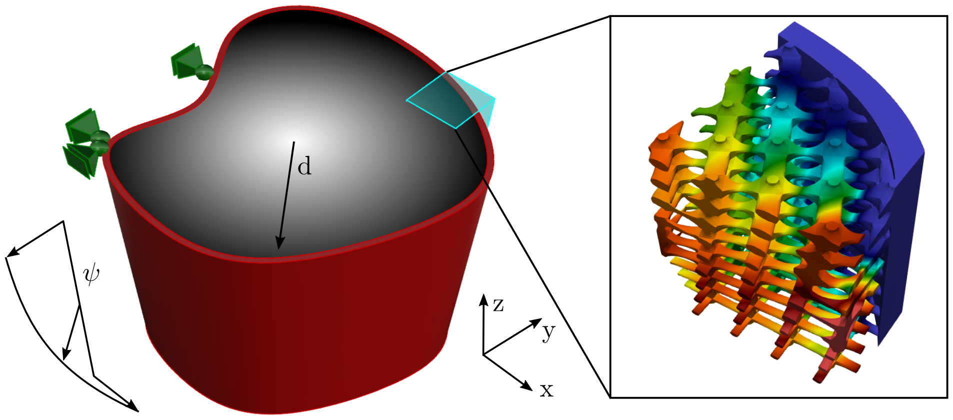
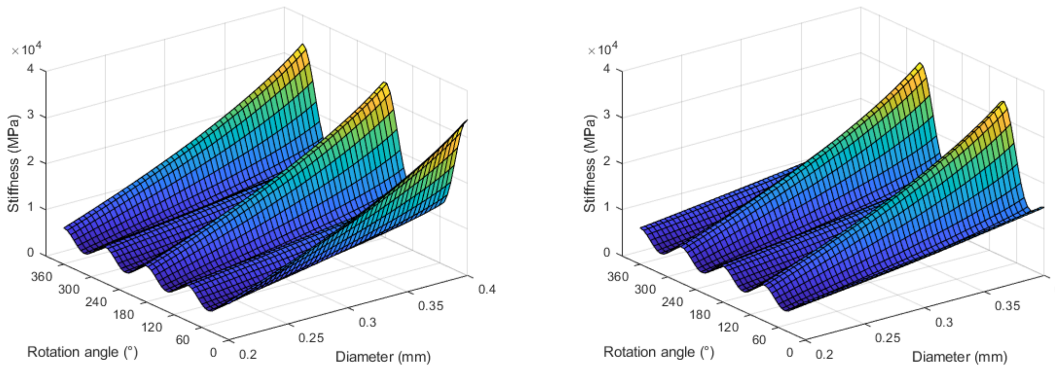


Figure 15.17: Structure consisting of a solid shell (red) and a homogenized microstructure (gray scale).

interface is of interest, then a geometrically resolving simulation as in Example 15.3 needs to be carried out.

The material matrix $\boldsymbol{C}(\psi, d)$ has 13 independent material coefficients that depend on the rotation angle and diameter of the stiffer rod, and thus, on the location of the respective quadrature point. To determine the corresponding material coefficients, the values from the look-up table (see Table 15.2) are interpolated using spline fitting. Figure 15.18 shows exemplarily the interpolation for the material coefficients C_{11} and C_{22}.

$$\boldsymbol{C}(\psi, d) = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & 0 & 0 \\ & C_{22} & C_{23} & C_{24} & 0 & 0 \\ & & C_{33} & C_{34} & 0 & 0 \\ & & & C_{44} & 0 & 0 \\ & & & & C_{55} & C_{56} \\ symm. & & & & & C_{66} \end{bmatrix} \tag{15.19}$$



(a) C_{11} (b) C_{22}

Figure 15.18: Spline based interpolation of the material coefficients C_{11} and C_{22}.

Figure 15.19 shows the displacements in $x-$direction and von Mises stresses of the structure under uniaxial compression in $z-$direction. The load is mainly transferred through the stiffer shell, yet the infill's contribution cannot be neglected. Due to the uniaxial compression, the rotation angle ψ of the microstructure has only little influence. The thickness of the rod d, on the other hand, can be deducted directly from the stress field of the infill.

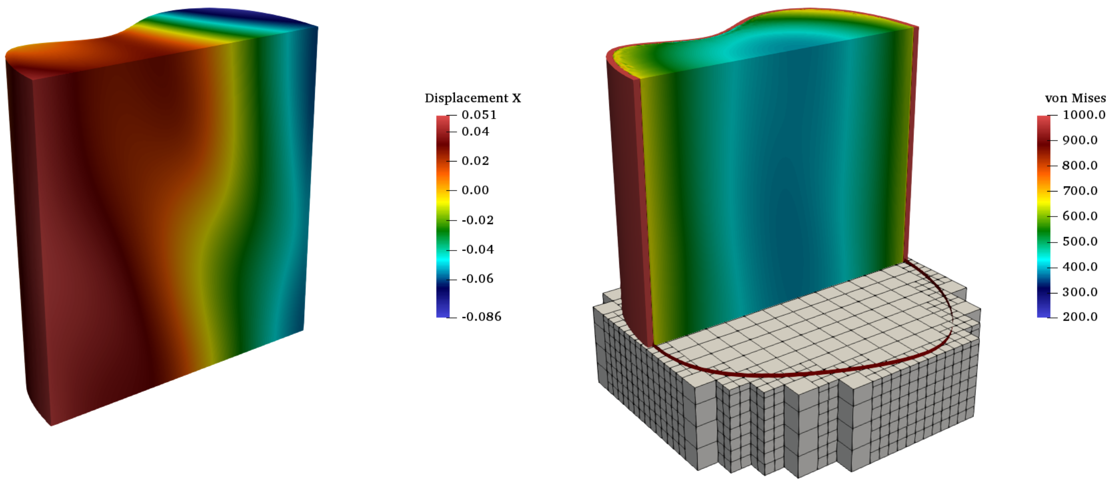


Figure 15.19: (a) Displacements in $x-$direction, and (b) von Mises stresses with the refined finite cell mesh.

Similar to the examples in Chapter 14, a geometrical or topological change does not affect the overall workflow. Also, the settings for the simulation can remain unchanged. Moreover, since the sole information required by the FCM is a PMC, it is possible to combine different model representations with the Boolean operations. For instance, a CSG primitive or V-model can be combined with a B-rep model. Of course, regarding a simulation with FGM, not all combinations of model representations and operations are straightforward, since B-rep or CSG primitives do not directly provide the required material properties. Yet, always applicable is subtracting an arbitrary primitive from a V-model, since the void is automatically filled with fictitious material. Figure 15.20 illustrates a Boolean difference of the microstructure with a CSG cylinder. The resulting hole is 'drilled' through the entire structure, leading to a topological change that would require a re-meshing in classical FEM or IGA. As can be seen in Figure 15.20, the infill contributes less to the load transfer, and high stress concentrations appear at the wall of the hole.

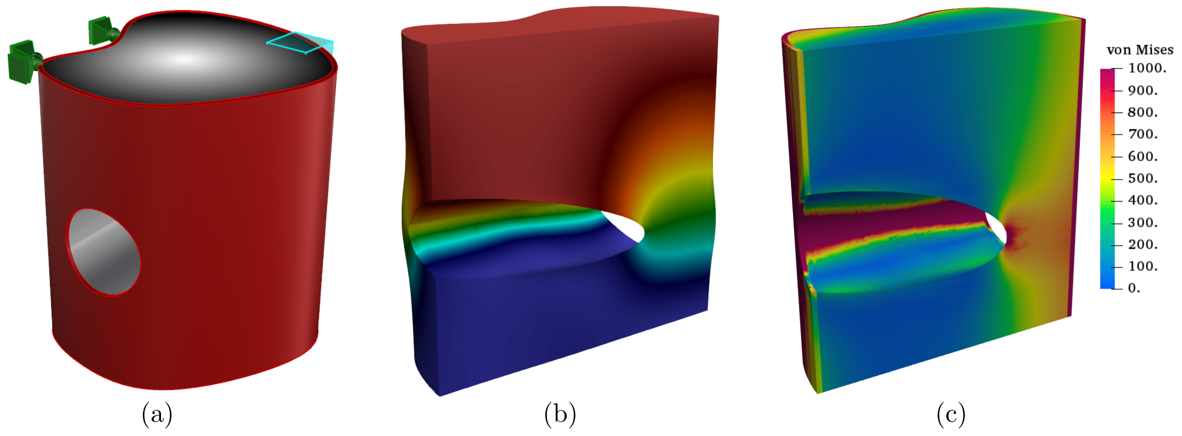


Figure 15.20: (a) Model with subtracted cylinder, (b) displacements in $z-$direction (warped by a scaling factor of $s = 2$), and (c) von Mises stresses.

Summary, conclusion and outlook

Summary

The objective of this thesis was the direct numerical simulation of different CAD representations – namely, flawed B-rep models, solid procedural/CSG models, and functionally graded material models. Traditionally, a numerical simulation method requires a preceding transition process from the CAD geometry to an analysis-suitable representation. In the case of classical FEM, this transition process typically comprises geometry healing, meshing, and mesh healing or mesh adaption. But also modern approaches like IGA require at least a valid CAD model. Unfortunately, especially geometry healing is difficult to automatize. The finite cell method turns out to be the perfect choice for the simulation framework for all kinds of geometrical representations. Its geometrical independence allows implementing a true direct simulation approach to overleap the entire transition process from the potentially invalid model to the numerical simulation. In its core, the FCM requires only one single information from the CAD geometry – an unambiguous statement about the membership of an arbitrary point, i.e., an answer to whether a point is inside or outside of the model. Depending on the geometric representation, the point membership needs to be determined differently.

Within the scope of this thesis, point membership classification methods for flawed B-rep and solid-based procedural/CSG models were developed and presented. The PMC for the flawed B-rep model involves a flaw-size dependent approximation with a space tree. This tree is constructed, such that inside and outside is always distinguishable. In order to preserve accuracy, the approximation tree is combined with a secondary PMC that is carried out only in the vicinity of the surfaces.

By contrast, the PMC for the solid-based procedural and CSG models takes advantage of the knowledge on the model's construction history. For this, the question of point membership is transferred through the construction history, or CSG tree and ultimately posed to the primitives. For each primitive and operation, a separate, perfectly matching test is presented that provides a robust and accurate point membership classification.

Finally, the FCM was extended to V-models, and their capability to represent functionally graded multi- and single-materials was exploited. For the latter – which corresponds to continuously changing microstructures – the FCM allows the extraction of the unit cells' effective material behavior based on homogenization simulations. With this information, it is possible to simulate large-scale, continuously changing microstructures.

Conclusion

Flawed B-rep models: Unquestionably, a simulation on flawed geometries will inevitably lead to modeling errors. Since an error corresponds to the deviation from an exact reference value – and a valid reference model is in general not available – the question arises: "Error referred to what?". The presented two-stage PMC allows narrowing down the answer. For this, the secondary ray tracing PMC close to the boundary is analyzed. This second test is carried out multiple times in different directions. In the case of an ambiguous result – i.e., at least one statement differs from the others – the majority of statements decide on the point membership. The number of ambiguous tests compared to the total amount of secondary PMCs is a good indicator of the degree of the defectiveness and the achievable accuracy. By carrying out a bracketing simulation – meaning, in two simulations, all ambiguous points are counted at first as inside and subsequently as outside – an upper and lower boundary for the internal energy and, thus, the error can be determined. As expected, the modeling error depends on the characteristic size of the flaws. However, since the influence of the flaws is very localized, the method is converging to 'some' solution, even in the case of large flaws. A thorough parameter study and simulations on complex, severely flawed B-rep models prove the applicability for real-world problems and show that highly accurate results can be obtained.

Solid-based/CSG models: In comparison to B-rep models, the amount of possible shapes that can be created with solid-based modeling is limited. This limitation is even more severe for classical CSG with its simple primitives and the classical three Boolean operations. Nevertheless, in application areas in which free form shapes are of less relevance, such as in mechanical engineering, solid-based procedural modeling is widely used. Regarding the numerical simulation with the FCM, one advantageous property is the immense robustness towards all kinds of flaws due to the closed analytical definition of the primitives and operations. Since, for all simple and extended/complex primitives and almost all operations, a separate, accurate, and robust PMC can be found, the FCM is ideally suited for the simulation of such models. In this context, only the shell operation is more evolved and requires user-interaction, as it is not a solid-based operation. Beneficial regarding the computational effort is the fact that the FCM can fully exploit the knowledge about the modeling history by an adaptive PMC test on the construction, or CSG tree. Adaptive means here that typically not the entire tree and, thus, not all primitives need to be queried because some operations can directly exclude full branches. The algorithm can be further improved by introducing intersection tests with the bounding boxes of the individual primitives. Regarding the modeling aspect, another significant advantage over B-rep is that subsequent modifications of the model can be carried out easily and require no complicated remodeling or adaption steps. The FCM behaves similarly flexible. I contrast to classical FEM, where a geometrical or topological change requires a re-meshing, the discretization of the FCM remains unchanged.

Functionally graded materials with V-reps: Additive manufacturing as arising new production technique gains more and more importance and attention. Due to its freedom in producible shapes, AM has the potential to change product development completely. However, one of the big problems is the significant difference between the scales – the laser is in the range of microns, whereas the laser paths' length can add up to kilometers. Such large deviations are tremendously challenging to represent by a numerical analysis, be it for the actual manufacturing process, or the final product. Exploiting AM's full capabilities, it is possible to generate two different kinds of functionally graded materials, multi- and single-material FGM, where the latter corresponds, for instance, to continuously changing microstructures. The V-rep framework offers the possibility to design both types of FGM. Although initially developed for IGA, the FCM renders to be ideally suited for the subsequent numerical simulation. The underlying Irit library already offers a robust and fast PMC test and additionally provides the corresponding material properties during the simulation. Furthermore, the FCM provides a remedy for the simulation of multi-scale problems based on homogenization. The effective material behavior of the small building blocks of microstructures – the unit cells – is classified, enabling purposeful modeling and simulation of large-scale microstructures. Several numerical examples prove the applicability for single- and multi-material FGM.

Outlook

Potential for further developments exists for all aspects covered within this thesis. In particular, this is the case for the direct simulation of flawed B-rep geometries, where the presented approach is a proof-of-concept of a truly working solution for one of the most urgent open questions in computer-aided manufacturing. Other direct approaches are not able to circumvent an elaborate, preparatory geometry healing step. Invalid CAD geometries were recently identified as a major pitfall for IGA. In this context, several further developments are directly sensible, for instance (i) an adaption of the PMC for curved shells – which is the main application of IGA, (ii) a tighter connection of FCM with IGA, in which the PMC of the FCM is used for the integration of the spline basis functions of IGA, or (iii) an extension to classical FEM. For the latter, consider a model, which is, in general, valid and flawed only in some regions of minor interest. Such a model could be meshed and simulated classically, wherever possible. In flawed regions, the FCM could then provide a best-possible approximation. Additional improvements in the existing approach could comprise implementing a more effective data structure for the boundary triangulation (e.g., with an $kd-$tree), the implementation, and comparison with other secondary PMC methods (e.g., based on point clouds), or an extension to curved boundary surfaces. In any case, the PMC for flawed B-rep models should be applied to more real-world applications to illustrate further its accuracy, flexibility, and potential of significant workload reduction.

Regarding the PMC on CSG-like models, a possible extension could be the implementation of a plugin for a commercial solid-based procedural CAD software, such as Siemens NX®, SolidWorks®, or Autodesk Inventor® that would allow a direct numerical simulation inside the CAD tool. Another topic could be rearranging the construction history into a balanced or differently optimized CSG tree, which could render many PMCs unnecessary and thus further improve the computational effort.

The collaboration of V-reps and FCM showed great potential for modeling and simulating multi-scale microstructures based on homogenization and classification of the underlying unit cells. It is sensible to advance this concept to an automatized generation of functionally contributing microstructures, i.e., microstructures that are not merely infill, but which are contributing to the overall behavior, such as load-carrying, thermal-, chemical-, or electric conductivity/resistance, etc. A possible workflow could involve the following steps: At first a complete determination and classification of the effective behavior of various sets of unit cells in all kinds of different shapes, orientations, etc. are carried out using homogenization with the FCM. Subsequently, a preliminary numerical analysis with isotropic, homogeneous material obtains the object's general response towards some exposure, e.g., the principal stress trajectories. And finally, an automatized selection of the best-suited unit cells composes the microstructure. Numerical analyses on this material can be quickly and efficiently carried out since the homogenized material is known a priori. Another development could address a simulation of the manufacturing process. For this, the functionality of the Irit library to directly create a G-code for a V-model could be used.

Bibliography

[1] M. Webster. *Definition of ENGINEERING*. May 29, 2020. URL: https://www.merriam-webster.com/dictionary/engineering (visited on 05/29/2020).

[2] M. J. Turner, R. W. Clough, H. C. Martin, and L. J. Topp. "Stiffness and Deflection Analysis of Complex Structures". In: *Journal of the Aeronautical Sciences* 23.9 (Sept. 1, 1956), pp. 805–823. DOI: 10.2514/8.3664.

[3] M. Mäntylä. *An Introduction to Solid Modeling*. Principles of Computer Science Series 13. Rockville: Computer Science Press, 1988. 401 pp. ISBN: 978-0-88175-108-6.

[4] J. J. Shah and M. Mäntylä. *Parametric and Feature-Based CAD/CAM: Concepts, Techniques, and Applications*. John Wiley & Sons, 1995. 646 pp. ISBN: 978-0-471-00214-7.

[5] J. D. Foley, A. V. Dam, S. K. Feiner, J. F. Hughes, and R. L. Phillips. *Introduction to Computer Graphics*. Addison-Wesley, 1997. 557 pp. ISBN: 978-0-201-60921-9.

[6] F. Massarwi and G. Elber. "A B-Spline Based Framework for Volumetric Object Modeling". In: *Computer-Aided Design*. SPM 2016 78 (Sept. 2016), pp. 36–47. ISSN: 0010-4485. DOI: 10.1016/j.cad.2016.05.003.

[7] P. T. Boggs, A. (E. Althsuler, A. R. (E. Larzelere, E. J. Walsh, R. L. Clay, and M. F. (N. L. Hardwick. *DART System Analysis*. SAND2005-4647. Sandia National Laboratories, Aug. 1, 2005. DOI: 10.2172/876325.

[8] W. Pöschl. "B-Spline Finite Elements and Their Efficiency in Solving Relativistic Mean Field Equations". In: *Computer Physics Communications* 112.1 (July 1, 1998), pp. 42–66. ISSN: 0010-4655. DOI: 10.1016/S0010-4655(98)00003-4.

[9] P. Kagan and A. Fischer. "Integrated Mechanically Based CAE System Using B-Spline Finite Elements". In: *Computer-Aided Design* 32.8 (Aug. 1, 2000), pp. 539–552. ISSN: 0010-4485. DOI: 10.1016/S0010-4485(00)00041-5.

[10] K. Höllig, U. Reif, and J. Wipper. "Weighted Extended B-Spline Approximation of Dirichlet Problems". In: *SIAM Journal on Numerical Analysis* 39.2 (2001), pp. 442–462. DOI: 10.1137/S0036142900373208.

[11] T. J. R. 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 (Oct. 1, 2005), pp. 4135–4195. ISSN: 0045-7825. DOI: 10.1016/j.cma.2004.10.008.

[12] L. F. Leidinger, M. Breitenberger, A. M. Bauer, S. Hartmann, R. Wüchner, K. .-. Bletzinger, F. Duddeck, and L. Song. "Explicit Dynamic Isogeometric B-Rep Analysis of Penalty-Coupled Trimmed NURBS Shells". In: *Computer Methods in Applied Mechanics and Engineering* 351 (July 1, 2019), pp. 891–927. ISSN: 0045-7825. DOI: 10.1016/j.cma.2019.04.016.

[13] B. Wassermann, S. Kollmannsberger, T. Bog, and E. Rank. “From Geometric Design to Numerical Analysis: A Direct Approach Using the Finite Cell Method on Constructive Solid Geometry”. In: *Computers & Mathematics with Applications*. High-Order Finite Element and Isogeometric Methods 2016 74.7 (Oct. 1, 2017), pp. 1703–1726. ISSN: 0898-1221. DOI: 10.1016/j.camwa.2017.01.027.

[14] B. Wassermann, S. Kollmannsberger, S. Yin, L. Kudela, and E. Rank. “Integrating CAD and Numerical Analysis: ‘Dirty Geometry’ Handling Using the Finite Cell Method”. In: *Computer Methods in Applied Mechanics and Engineering* 351 (July 1, 2019), pp. 808–835. ISSN: 0045-7825. DOI: 10.1016/j.cma.2019.04.017.

[15] B. Wassermann, N. Korshunova, S. Kollmannsberger, E. Rank, and G. Elber. “Direct Simulation of Functionally Graded Materials Using V-Models and the Finite Cell Method”. In: *Preprint. Submitted to Advanced Modeling and Simulation in Engineering Sciences* (2020). DOI: https://arxiv.org/abs/2007.10433.

[16] J. Bonet and R. D. Wood. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge ; New York, NY, USA: Cambridge University Press, 1997. 248 pp. ISBN: 0-521-57272-X.

[17] G. A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Chichester, New York: Wiley, 2000. 455 pp. ISBN: 0-471-82304-X.

[18] M. Breitenberger, A. Apostolatos, B. Philipp, R. Wüchner, and K. .-. Bletzinger. “Analysis in Computer Aided Design: Nonlinear Isogeometric B-Rep Analysis of Shell Structures”. In: *Computer Methods in Applied Mechanics and Engineering*. Isogeometric Analysis Special Issue 284 (Feb. 1, 2015), pp. 401–457. ISSN: 0045-7825. DOI: 10.1016/j.cma.2014.09.033.

[19] B. Philipp, M. Breitenberger, I. D’Auria, R. Wüchner, and K.-U. Bletzinger. “Integrated Design and Analysis of Structural Membranes Using the Isogeometric B-Rep Analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 303 (May 2016), pp. 312–340. ISSN: 00457825. DOI: 10.1016/j.cma.2016.02.003.

[20] J. W. Rudnicki. *Fundamentals of Continuum Mechanics*. 1st ed. Chichester, West Sussex, United Kingdom: Wiley, Nov. 10, 2014. 218 pp. ISBN: 978-1-118-47991-9.

[21] B. D. Reddy. *Introductory Functional Analysis: With Applications to Boundary Value Problems and Finite Elements*. Texts in Applied Mathematics 27. New York: Springer, Nov. 20, 1997. 472 pp. ISBN: 978-0-387-98307-3.

[22] O. Zienkiewicz, R. Taylor, and J. Zhu. *The Finite Element Method: Its Basis and Fundamentals*. 6th ed. Butterworth-Heinemann, 2005. ISBN: 0-7506-6320-0.

[23] C. A. Felippa. *Introduction to Finite Element Methods*. Lecture Notes. Boulder, Colorado, USA: Department of Aerospace Engineering Sciences, University of Colorado at Boulder, 2013.

[24] T. J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Mineola, NY: Dover Publications, 2000. 682 pp. ISBN: 0-486-41181-8.

[25] I. Babuska, B. Szabo, and I. Katz. “The P-Version of the Finite Element Method”. In: *SIAM Journal on Numerical Analysis* 18.3 (June 1, 1981), pp. 515–545. ISSN: 0036-1429. DOI: 10.1137/0718033.

[26] A. Düster, E. Rank, and B. A. Szabó. “The P-Version of the Finite Element Method and Finite Cell Methods”. In: *Encyclopedia of Computational Mechanics*. Ed. by E. Stein, R. Borst, and T. J. R. Hughes. Vol. 2. Chichester, West Sussex: John Wiley & Sons, 2017, pp. 1–35. ISBN: 978-1-119-00379-3.

[27] B. A. Szabó and I. Babuška. *Finite Element Analysis*. New York: John Wiley & Sons, 1991. 368 pp. ISBN: 0-471-50273-1.

[28] E. Heikkola, Y. A. Kuznetsov, and K. N. Lipnikov. “Fictitious Domain Methods for the Numerical Solution of Three-Dimensional Acoustic Scattering Problems”. In: *Journal of Computational Acoustics* 07.03 (Sept. 1, 1999), pp. 161–183. ISSN: 0218-396X. DOI: 10.1142/S0218396X99000126.

[29] U. Hetmaniuk and C. Farhat. “A Finite Element-Based Fictitious Domain Decomposition Method for the Fast Solution of Partially Axisymmetric Sound-Hard Acoustic Scattering Problems”. In: *Finite Elements in Analysis and Design.* 14th Robert J. Melosh Competition 39.8 (May 1, 2003), pp. 707–725. ISSN: 0168-874X. DOI: 10.1016/S0168-874X(03)00055-6.

[30] E. Burman and P. Hansbo. “Fictitious Domain Finite Element Methods Using Cut Elements: I. A Stabilized Lagrange Multiplier Method”. In: *Computer Methods in Applied Mechanics and Engineering* 199.41-44 (Oct. 2010), pp. 2680–2686. ISSN: 00457825. DOI: 10.1016/j.cma.2010.05.011.

[31] J. Bishop. “Rapid Stress Analysis of Geometrically Complex Domains Using Implicit Meshing”. In: *Computational Mechanics* 30 (Apr. 1, 2003), pp. 460–478. DOI: 10.1007/s00466-003-0424-5.

[32] W. K. Liu et al. “Immersed Finite Element Method and Its Applications to Biological Systems”. In: *Computer Methods in Applied Mechanics and Engineering* 195.13-16 (Feb. 15, 2006), pp. 1722–1749. ISSN: 0045-7825. DOI: 10.1016/j.cma.2005.05.049.

[33] T. Ye, R. Mittal, H. S. Udaykumar, and W. Shyy. “An Accurate Cartesian Grid Method for Viscous Incompressible Flows with Complex Immersed Boundaries”. In: *Journal of Computational Physics* 156.2 (Dec. 10, 1999), pp. 209–240. ISSN: 0021-9991. DOI: 10.1006/jcph.1999.6356.

[34] A. M. Roma, C. S. Peskin, and M. J. Berger. “An Adaptive Version of the Immersed Boundary Method”. In: *Journal of Computational Physics* 153.2 (Aug. 10, 1999), pp. 509–534. ISSN: 0021-9991. DOI: 10.1006/jcph.1999.6293.

[35] M. García-Ruíz and G. Steven. “Fixed Grid Finite Elements in Elasticity Problems”. In: *Engineering Computations* 16.2 (Jan. 1, 1999), pp. 145–164. DOI: 10.1108/02644409910257430.

[36] F. Daneshmand and M. Kazemzadeh-Parsi. “Static and Dynamic Analysis of 2D and 3D Elastic Solids Using the Modified FGFEM”. In: *Finite Elements in Analysis and Design* 45 (Sept. 1, 2009), pp. 755–765. DOI: 10.1016/j.finel.2009.06.003.

[37] E. Nadal, J. J. Ródenas, J. Albelda, M. Tur, J. E. Tarancón, and F. J. Fuenmayor. “Efficient Finite Element Methodology Based on Cartesian Grids: Application to Structural Shape Optimization”. In: *Abstract and Applied Analysis* 2013 (2013), pp. 1–19. ISSN: 1085-3375, 1687-0409. DOI: 10.1155/2013/953786.

[38] O. Marco, R. Sevilla, J. J. Ródenas, and M. Tur. “Numerical Simulation from Medical Images: Accurate Integration by Means of the Cartesian Grid Finite Element Method”. In: *Computational Modeling of Objects Presented in Images. Fundamentals, Methods, and Applications.* Ed. by Y. J. Zhang and J. M. R. S. Tavares. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 255–260. ISBN: 978-3-319-09994-1. DOI: 10.1007/978-3-319-09994-1_24.

[39] A. Düster, J. Parvizian, Z. Yang, and E. Rank. “The Finite Cell Method for Three-Dimensional Problems of Solid Mechanics”. In: *Computer Methods in Applied Mechanics and Engineering* 197.45–48 (Aug. 15, 2008), pp. 3768–3782. ISSN: 0045-7825. DOI: 10.1016/j.cma.2008.02.036.

[40] D. Schillinger and E. Rank. “An Unfitted Hp-Adaptive Finite Element Method Based on Hierarchical B-Splines for Interface Problems of Complex Geometry”. In: *Computer Methods in Applied Mechanics and Engineering* 200.47-48 (Nov. 2011), pp. 3358–3380. ISSN: 00457825. DOI: 10.1016/j.cma.2011.08.002.

[41] N. Zander, T. Bog, M. Elhaddad, F. Frischmann, S. Kollmannsberger, and E. Rank. “The Multi-Level Hp-Method for Three-Dimensional Problems: Dynamically Changing High-Order Mesh Refinement with Arbitrary Hanging Nodes”. In: *Computer Methods in Applied Mechanics and Engineering* 310 (Oct. 1, 2016), pp. 252–277. ISSN: 0045-7825. DOI: `10.1016/j.cma.2016.07.007`.

[42] N. Zander, M. Ruess, T. Bog, S. Kollmannsberger, and E. Rank. “Multi-Level Hp-Adaptivity for Cohesive Fracture Modeling”. In: *International Journal for Numerical Methods in Engineering* 109.13 (Mar. 30, 2017), pp. 1723–1755. ISSN: 1097-0207. DOI: `10.1002/nme.5340`.

[43] V. Varduhn, M.-C. Hsu, M. Ruess, and D. Schillinger. “The Tetrahedral Finite Cell Method: Higher-Order Immersogeometric Analysis on Adaptive Non-Boundary-Fitted Meshes”. In: *International Journal for Numerical Methods in Engineering* 107.12 (Jan. 1, 2016), pp. 1054–1079. ISSN: 1097-0207. DOI: `10.1002/nme.5207`.

[44] D. Kamensky, M.-C. Hsu, D. Schillinger, J. A. Evans, A. Aggarwal, Y. Bazilevs, M. S. Sacks, and T. J. R. Hughes. “An Immersogeometric Variational Framework for Fluid–Structure Interaction: Application to Bioprosthetic Heart Valves”. In: *Computer Methods in Applied Mechanics and Engineering*. Isogeometric Analysis Special Issue 284 (Feb. 1, 2015), pp. 1005–1053. ISSN: 0045-7825. DOI: `10.1016/j.cma.2014.10.040`.

[45] S. Duczek and U. Gabbert. “The Finite Cell Method for Polygonal Meshes: Poly-FCM”. In: *Computational Mechanics* (June 27, 2016), pp. 1–32. ISSN: 0178-7675, 1432-0924. DOI: `10.1007/s00466-016-1307-x`.

[46] F. de Prenter, C. V. Verhoosel, G. J. van Zwieten, and E. H. van Brummelen. “Condition Number Analysis and Preconditioning of the Finite Cell Method”. In: *Computer Methods in Applied Mechanics and Engineering*. Special Issue on Isogeometric Analysis: Progress and Challenges 316 (Supplement C Apr. 1, 2017), pp. 297–327. ISSN: 0045-7825. DOI: `10.1016/j.cma.2016.07.006`.

[47] J. Parvizian, A. Düster, and E. Rank. “Finite Cell Method”. In: *Computational Mechanics* 41.1 (Apr. 2007), pp. 121–133. ISSN: 0178-7675. DOI: `10.1007/s00466-007-0173-y`.

[48] F. de Prenter. “Preconditioned Iterative Solution Techniques for Immersed Finite Element Methods: With Applications in Immersed Isogeometric Analysis for Solid and Fluid Mechanics”. PhD Thesis. Eindhoven: Eindhoven University of Technology, June 27, 2019.

[49] M. Dauge, A. Düster, and E. Rank. “Theoretical and Numerical Investigation of the Finite Cell Method”. In: *Journal of Scientific Computing* 65.3 (Mar. 5, 2015), pp. 1039–1064. ISSN: 0885-7474, 1573-7691. DOI: `10.1007/s10915-015-9997-3`.

[50] L. Kudela, N. Zander, S. Kollmannsberger, and E. Rank. “Smart Octrees: Accurately Integrating Discontinuous Functions in 3D”. In: *Computer Methods in Applied Mechanics and Engineering* 306 (July 1, 2016), pp. 406–426. ISSN: 0045-7825. DOI: `10.1016/j.cma.2016.04.006`.

[51] S. Hubrich, P. D. Stolfo, L. Kudela, S. Kollmannsberger, E. Rank, A. Schröder, and A. Düster. “Numerical Integration of Discontinuous Functions: Moment Fitting and Smart Octree”. In: *Computational Mechanics* (July 18, 2017), pp. 1–19. ISSN: 0178-7675, 1432-0924. DOI: `10.1007/s00466-017-1441-0`.

[52] M. Joulaian, S. Hubrich, and A. Düster. “Numerical Integration of Discontinuities on Arbitrary Domains Based on Moment Fitting”. In: *Computational Mechanics* 57.6 (June 2016), pp. 979–999. ISSN: 0178-7675, 1432-0924. DOI: `10.1007/s00466-016-1273-3`.

[53] B. Müller, F. Kummer, and M. Oberlack. “Highly Accurate Surface and Volume Integration on Implicit Domains by Means of Moment-Fitting”. In: *International Journal for Numerical Methods in Engineering* 96.8 (Nov. 23, 2013), pp. 512–528. ISSN: 1097-0207. DOI: `10.1002/nme.4569`.

[54] M. Ruess, Y. Bazilevs, D. Schillinger, N. Zander, and E. Rank. “Weakly Enforced Boundary Conditions for the NURBS-Based Finite Cell Method”. In: *European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS).* Vienna, Austria, 2012. ISBN: 978-3-9502481-9-7.

[55] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn, and E. Rank. “Weakly Enforced Essential Boundary Conditions for NURBS-Embedded and Trimmed NURBS Geometries on the Basis of the Finite Cell Method”. In: *International Journal for Numerical Methods in Engineering* 95.10 (Sept. 7, 2013), pp. 811–846. ISSN: 00295981. DOI: 10.1002/nme.4522.

[56] S. Kollmannsberger, A. Özcan, J. Baiges, M. Ruess, E. Rank, and A. Reali. “Parameter-Free, Weak Imposition of Dirichlet Boundary Conditions and Coupling of Trimmed and Non-Conforming Patches”. In: *International Journal for Numerical Methods in Engineering* 101.9 (Mar. 2, 2015), pp. 670–699. ISSN: 1097-0207. DOI: 10.1002/nme.4817.

[57] Y. Guo and M. Ruess. “Nitsche’s Method for a Coupling of Isogeometric Thin Shells and Blended Shell Structures”. In: *Computer Methods in Applied Mechanics and Engineering.* Isogeometric Analysis Special Issue 284 (Feb. 1, 2015), pp. 881–905. ISSN: 0045-7825. DOI: 10.1016/j.cma.2014.11.014.

[58] S. Fernández-Méndez and A. Huerta. “Imposing Essential Boundary Conditions in Mesh-Free Methods”. In: *Computer Methods in Applied Mechanics and Engineering* 193.12-14 (Mar. 2004), pp. 1257–1275. ISSN: 00457825. DOI: 10.1016/j.cma.2003.12.019.

[59] I. Babuska. “The Finite Element Method with Penalty”. In: *Mathematics of Computation* 27.122 (Apr. 1973), pp. 221–228. ISSN: 00255718. DOI: 10.2307/2005611.

[60] T. Zhu and S. N. Atluri. “A Modified Collocation Method and a Penalty Formulation for Enforcing the Essential Boundary Conditions in the Element Free Galerkin Method”. In: *Computational Mechanics* 21.3 (Apr. 23, 1998), pp. 211–222. ISSN: 0178-7675, 1432-0924. DOI: 10.1007/s004660050296.

[61] L. Demkowicz. *Computing with Hp-Adaptive Finite Elements, Vol. 1: One and Two Dimensional Elliptic and Maxwell Problems.* Applied Mathematics and Nonlinear Science Series. Boca Raton: Chapman & Hall/CRC, 2007. 1 p. ISBN: 1-58488-671-4.

[62] D. D’Angella, S. Kollmannsberger, E. Rank, and A. Reali. “Multi-Level Bézier Extraction for Hierarchical Local Refinement of Isogeometric Analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 328 (Jan. 2018), pp. 147–174. ISSN: 00457825. DOI: 10.1016/j.cma.2017.08.017.

[63] E. Rank. “Adaptive Remeshing and H-p Domain Decomposition”. In: *Computer Methods in Applied Mechanics and Engineering* 101.1–3 (Dec. 1992), pp. 299–313. ISSN: 0045-7825. DOI: 10.1016/0045-7825(92)90027-H.

[64] D. Schillinger, A. Düster, and E. Rank. “The Hp-d-Adaptive Finite Cell Method for Geometrically Nonlinear Problems of Solid Mechanics”. In: *International Journal for Numerical Methods in Engineering* 89.9 (2012), pp. 1171–1202. ISSN: 1097-0207. DOI: 10.1002/nme.3289.

[65] N. Zander. “Multi-Level Hp-FEM: Dynamically Changing High-Order Mesh Refinement with Arbitrary Hanging Nodes”. PhD Thesis. München: Technische Universität München, 2017.

[66] B. A. Szabó and I. Babuška. *Introduction to Finite Element Analysis: Formulation, Verification, and Validation.* Chichester, West Sussex: Wiley, 2011. ISBN: 978-0-470-97728-6.

[67] R. Verfürth. “A Review of a Posteriori Error Estimation Techniques for Elasticity Problems”. In: *Computer Methods in Applied Mechanics and Engineering* 176.1–4 (July 6, 1999), pp. 419–440. ISSN: 0045-7825. DOI: 10.1016/S0045-7825(98)00347-8.

[68] M. Ainsworth and J. T. Oden. “A Posteriori Error Estimation in Finite Element Analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 142.1–2 (Mar. 15, 1997), pp. 1–88. ISSN: 0045-7825. DOI: `10.1016/S0045-7825(96)01107-3`.

[69] D. D’Angella, N. Zander, S. Kollmannsberger, F. Frischmann, E. Rank, A. Schröder, and A. Reali. “Multi-Level Hp-Adaptivity and Explicit Error Estimation”. In: *Advanced Modeling and Simulation in Engineering Sciences* 3.1 (Dec. 1, 2016), p. 33. ISSN: 2213-7467. DOI: `10.1186/s40323-016-0085-5`.

[70] B. A. Szabó, A. Düster, and E. Rank. “The P-Version of the Finite Element Method”. In: *Encyclopedia of Computational Mechanics*. Ed. by E. Stein. Chichester, West Sussex: John Wiley & Sons, 2004. ISBN: 978-0-470-09135-7.

[71] L. Piegl and W. Tiller. *The NURBS Book*. Monographs in Visual Communication. Berlin Heidelberg: Springer-Verlag, 1995. ISBN: 978-3-642-97385-7. DOI: `10.1007/978-3-642-97385-7`.

[72] P. Bézier. *The Mathematical Basis of the UNISURF CAD System*. London; Boston: Butterworths, 1986. ISBN: 978-0-408-22175-7.

[73] G. Elber. *Guirit - A Graphics User Interface to Irit - Home Page*. Sept. 5, 2019. URL: `http://www.cs.technion.ac.il/%20~gershon/GuIrit/` (visited on 09/05/2019).

[74] H.-J. Bungartz, M. Griebel, and C. Zenger. *Introduction to Computer Graphics*. Charles River Media, 2004. 290 pp. ISBN: 978-1-58450-332-3.

[75] GrabCad. *General Electric Jet Engine Bracket Challenge - GrabCAD*. 2013. URL: `https://grabcad.com%20/challenges/ge-jet-engine-bracket-challenge` (visited on 11/27/2017).

[76] A. Bundy and L. Wallen. “Constructive Solid Geometry”. In: *Catalogue of Artificial Intelligence Tools*. Ed. by A. Bundy and L. Wallen. Symbolic Computation. Berlin, Heidelberg: Springer, 1984, pp. 21–22. ISBN: 978-3-642-96868-6. DOI: `10.1007/978-3-642-96868-6_40`.

[77] G. Elber. *The IRIT Modeling Environment - Home Page*. July 1, 2020. URL: `http://www.cs.technion.ac.il/~irit/` (visited on 07/01/2020).

[78] G. Elber, Y.-J. Kim, and M.-S. Kim. “Volumetric Boolean Sum”. In: *Computer Aided Geometric Design*. Geometric Modeling and Processing 2012 29.7 (Oct. 1, 2012), pp. 532–540. ISSN: 0167-8396. DOI: `10.1016/j.cagd.2012.03.003`.

[79] E. Brivadis, A. Buffa, B. Wohlmuth, and L. Wunderlich. “Isogeometric Mortar Methods”. In: *Computer Methods in Applied Mechanics and Engineering* 284 (July 31, 2014). DOI: `10.1016/j.cma.2014.09.012`.

[80] T. W. Sederberg, G. T. Finnigan, X. Li, H. Lin, and H. Ipson. “Watertight Trimmed NURBS”. In: *ACM SIGGRAPH 2008 Papers*. New York, NY, USA: ACM, 2008. DOI: `10.1145/1399504.1360678`.

[81] W. E. Lorensen and H. E. Cline. “Marching Cubes: A High Resolution 3D Surface Construction Algorithm”. In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’87. New York, NY: ACM Press, 1987, pp. 163–169. ISBN: 978-0-89791-227-3. DOI: `10.1145/37401.37422`.

[82] G. Butlin and C. Stops. “CAD Data Repair”. In: *Proceedings of the 5th International Meshing Roundtable*. 1996, pp. 7–12.

[83] H. Gu, T. R. Chase, D. C. Cheney, T. “ Bailey, and D. Johnson. “Identifying, Correcting, and Avoiding Errors in Computer-Aided Design Models Which Affect Interoperability”. In: *Journal of Computing and Information Science in Engineering* 1.2 (May 1, 2001), pp. 156–166. ISSN: 1530-9827. DOI: `10.1115/1.1384887`.

[84] N. M. Patrikalakis, T. Sakkalis, and G. Shen. “Boundary Representation Models: Validity and Rectification”. In: *The Mathematics of Surfaces IX*. Ed. by R. Cipolla and R. Martin. London: Springer, 2000, pp. 389–409. ISBN: 978-1-4471-0495-7. DOI: 10.1007/978-1-4471-0495-7_23.

[85] C. M. Hoffmann. *Geometric and Solid Modeling: An Introduction*. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. San Mateo, Calif: Morgan Kaufmann, 1989. 338 pp. ISBN: 978-1-55860-067-6.

[86] B. Rumpe. *Modeling with UML: Language, Concepts, Methods*. Springer International Publishing, 2016. ISBN: 978-3-319-33932-0.

[87] A. Goldberg and D. Robson. *Smalltalk-80: The Language and Its Implementation*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1983. ISBN: 978-0-201-11371-6.

[88] S. Suresh. *Fundamentals of Functionally Graded Materials: Processing and Thermomechanical Behaviour of Graded Metals and Metal-Ceramic Composites*. London: IOM Communications Ltd, 1998. vi, 165. ISBN: 978-1-86125-063-6.

[89] S. Bohidar, R. Sharma, and P. Mishra. “Functionally Graded Materials: A Critical Review”. In: *Int J Res (IJR)* 1 (Jan. 1, 2014), pp. 289–301. ISSN: 2348-6848.

[90] N. Noda. “Thermal Stresses in Functionally Graded Materials”. In: *Journal of Thermal Stresses* 22.4-5 (June 1, 1999), pp. 477–512. ISSN: 0149-5739. DOI: 10.1080/014957399280841.

[91] C. Zhang et al. “Additive Manufacturing of Functionally Graded Materials: A Review”. In: *Materials Science and Engineering: A* 764 (Sept. 9, 2019), p. 138209. ISSN: 0921-5093. DOI: 10.1016/j.msea.2019.138209.

[92] B. Zhang, P. Jaiswal, R. Rai, and S. Nelaturi. “Additive Manufacturing of Functionally Graded Material Objects: A Review”. In: *Journal of Computing and Information Science in Engineering* 18.4 (Dec. 1, 2018), p. 041002. ISSN: 1530-9827. DOI: 10.1115/1.4039683.

[93] F. Yan, W. Xiong, and E. Faierson. “Grain Structure Control of Additively Manufactured Metallic Materials”. In: *Materials* 10 (Nov. 2, 2017), p. 1260. DOI: 10.3390/ma10111260.

[94] G. H. Loh, E. Pei, D. Harrison, and M. D. Monzón. “An Overview of Functionally Graded Additive Manufacturing”. In: *Additive Manufacturing* 23 (Oct. 1, 2018), pp. 34–44. ISSN: 2214-8604. DOI: 10.1016/j.addma.2018.06.023.

[95] A. O. Aremu, J. P. J. Brennan-Craddock, A. Panesar, I. A. Ashcroft, R. J. M. Hague, R. D. Wildman, and C. Tuck. “A Voxel-Based Method of Constructing and Skinning Conformal and Functionally Graded Lattice Structures Suitable for Additive Manufacturing”. In: *Additive Manufacturing* 13 (Jan. 1, 2017), pp. 1–13. ISSN: 2214-8604. DOI: 10.1016/j.addma.2016.10.006.

[96] D. Geraldes. “Orthotropic Modelling of the Skeletal System”. PhD Thesis. London: Imperial College of Science, Technology and Medicine, Mar. 1, 2013.

[97] J. Jiang, X. Xu, and J. Stringer. “Support Structures for Additive Manufacturing: A Review”. In: *Journal of Manufacturing and Materials Processing* 2 (Sept. 20, 2018). DOI: 10.3390/jmmp2040064.

[98] A. Clausen, N. Aage, and O. Sigmund. “Exploiting Additive Manufacturing Infill in Topology Optimization for Improved Buckling Load”. In: *Engineering* 2 (June 1, 2016), pp. 250–257. DOI: 10.1016/J.ENG.2016.02.006.

[99] A. Bandyopadhyay and B. Heer. “Additive Manufacturing of Multi-Material Structures”. In: *Materials Science and Engineering: R: Reports* 129 (July 1, 2018), pp. 1–16. ISSN: 0927-796X. DOI: 10.1016/j.mser.2018.04.001.

[100] J. Koopmann, J. Voigt, and T. Niendorf. “Additive Manufacturing of a Steel–Ceramic Multi-Material by Selective Laser Melting”. In: *Metallurgical and Materials Transactions B* 50.2 (Apr. 1, 2019), pp. 1042–1051. ISSN: 1543-1916. DOI: 10.1007/s11663-019-01523-1.

[101] K.-H. Shin and D. Dutta. “Constructive Representation of Heterogeneous Objects”. In: *Journal of Computing and Information Science in Engineering* 1.3 (June 1, 2001), pp. 205–217. ISSN: 1530-9827. DOI: 10.1115/1.1403448.

[102] X. Wu, W. Liu, and M. Y. Wang. “A CAD Modeling System for Heterogeneous Object”. In: *Adv. Eng. Softw.* 39.5 (May 2008), pp. 444–453. ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2007.03.002.

[103] E. L. Doubrovski, E. Y. Tsai, D. Dikovsky, J. M. P. Geraedts, H. Herr, and N. Oxman. “Voxel-Based Fabrication through Material Property Mapping: A Design Method for Bitmap Printing”. In: *Computer-Aided Design.* Material Ecology 60 (Mar. 1, 2015), pp. 3–13. ISSN: 0010-4485. DOI: 10.1016/j.cad.2014.05.010.

[104] V. Chandru, S. Manohar, and C. E. Prakash. “Voxel-Based Modeling for Layered Manufacturing”. In: *IEEE Computer Graphics and Applications* 15.6 (Nov. 1995), pp. 42–47. ISSN: 0272-1716. DOI: 10.1109/38.469516.

[105] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* New York, NY, USA: Springer-Verlag New York, Inc., 1985. ISBN: 978-0-387-96131-6.

[106] M. Elhaddad, N. Zander, T. Bog, L. Kudela, S. Kollmannsberger, J. S. Kirschke, T. Baum, M. Ruess, and E. Rank. “Multi-Level Hp-Finite Cell Method for Embedded Interface Problems with Application in Biomechanics”. In: *International Journal for Numerical Methods in Biomedical Engineering* 34.4 (2018), e2951. ISSN: 2040-7947. DOI: 10.1002/cnm.2951.

[107] L. Kudela, S. Kollmannsberger, U. Almac, and E. Rank. “Direct Structural Analysis of Domains Defined by Point Clouds”. In: *Computer Methods in Applied Mechanics and Engineering* 358 (Jan. 1, 2020), p. 112581. ISSN: 0045-7825. DOI: 10.1016/j.cma.2019.112581.

[108] A. F. Möbius. “Über die Bestimmung des Inhaltes eines Polyëders”. In: *Berichte über die Verhandlungen der Königlich Sächsischen Gesellschaft der Wissenschaften.* Vol. Mathematisch-Physische Klasse. Berichte über die Verhandlungen der Königlich Sächsischen Gesellschaft der Wissenschaften 17. Leipzig, 1865, pp. 31–68.

[109] M. Shimrat. “Algorithm 112: Position of Point Relative to Polygon”. In: *Communications of the ACM* 5.8 (Aug. 1, 1962), p. 434. ISSN: 0001-0782. DOI: 10.1145/368637.368653.

[110] A. Becciu, A. Fuster, M. Pottek, B. van den Heuvel, B. ter Haar Romeny, and H. van Assen. “3D Winding Number: Theory and Application to Medical Imaging”. In: *International Journal of Biomedical Imaging* 2011 (2011), e516942. ISSN: 1687-4188. DOI: 10.1155/2011/516942.

[111] T. Möller and B. Trumbore. “Fast, Minimum Storage Ray-Triangle Intersection”. In: *Journal of Graphics Tools* 2.1 (Jan. 1, 1997), pp. 21–28. ISSN: 1086-7651. DOI: 10.1080/10867651.1997.10487468.

[112] T. W. Sederberg, D. C. Anderson, and R. N. Goldman. “Implicit Representation of Parametric Curves and Surfaces”. In: *Computer Vision, Graphics, and Image Processing* 28.1 (Oct. 1, 1984), pp. 72–84. ISSN: 0734-189X. DOI: 10.1016/0734-189X(84)90140-3.

[113] G. Strang. *An Analysis of the Finite Element Method.* In collab. with G. J. Fix. Englewood Cliffs, N.J: Prentice-Hall, 1973. 306 pp. ISBN: 0-13-032946-0.

[114] S. Nemat-Nasser, M. Hori, and J. Achenbach. *Micromechanics: Overall Properties of Heterogeneous Materials.* North-Holland Series in Applied Mathematics and Mechanics. Elsevier Science, 2013. ISBN: 978-1-4832-9151-2.

[115] R. Hill. “Elastic Properties of Reinforced Solids: Some Theoretical Principles”. In: *Journal of the Mechanics and Physics of Solids* 11.5 (Sept. 1, 1963), pp. 357–372. ISSN: 0022-5096. DOI: 10.1016/0022-5096(63)90036-X.

[116] N. Korshunova, J. Jomo, G. Lékó, D. Reznik, P. Balázs, and S. Kollmannsberger. “Image-Based Material Characterization of Complex Microarchitectured Additively Manufactured Structures”. In: *Preprint accepted. To be published in Computers & Mathematics with Applications* (2019). DOI: https://arxiv.org/abs/1912.07415.

[117] F. Feyel. “A Multilevel Finite Element Method (FE2) to Describe the Response of Highly Non-Linear Structures Using Generalized Continua”. In: *Computer Methods in Applied Mechanics and Engineering.* Multiscale Computational Mechanics for Materials and Structures 192.28 (July 18, 2003), pp. 3233–3244. ISSN: 0045-7825. DOI: 10.1016/S0045-7825(03)00348-7.

[118] S. Morrissey. *Sean Morrissey - GrabCAD.* July 22, 2013. URL: https://grabcad.com/sean.morrissey-1 (visited on 05/19/2020).

[119] J. Schöberl. *NETGEN.* 2003. URL: http://www.hpfem.jku.at/netgen/ (visited on 12/22/2016).

[120] A. Materials. *AZO Materials - Material Science - Material Engineering.* Jan. 19, 2020. URL: https://www.azom.com/ (visited on 01/19/2020).

[121] I. J. Gibson and M. F. Ashby. “The Mechanics of Three-Dimensional Cellular Materials”. In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 382.1782 (July 8, 1982), pp. 43–59. DOI: 10.1098/rspa.1982.0088.

[122] W. Pabst, T. Uhlířová, E. Gregorová, and A. Wiegmann. “Young’s Modulus and Thermal Conductivity of Closed-Cell, Open-Cell and Inverse Ceramic Foams – Model-Based Predictions, Cross-Property Predictions and Numerical Calculations”. In: *Journal of the European Ceramic Society* 38.6 (June 1, 2018), pp. 2570–2578. ISSN: 0955-2219. DOI: 10.1016/j.jeurceramsoc.2018.01.019.

[123] W. Pabst and E. Gregorová. “Critical Assessment 18: Elastic and Thermal Properties of Porous Materials – Rigorous Bounds and Cross-Property Relations”. In: *Materials Science and Technology* 31.15 (Dec. 8, 2015), pp. 1801–1808. ISSN: 0267-0836. DOI: 10.1080/02670836.2015.1114697.

[124] G. Legrain, N. Chevaugeon, and K. Dréau. “High Order X-FEM and Levelsets for Complex Microstructures: Uncoupling Geometry and Approximation”. In: *Computer Methods in Applied Mechanics and Engineering* 241-244 (Oct. 2012), pp. 172–189. ISSN: 00457825. DOI: 10.1016/j.cma.2012.06.001.

[125] S. C. Cowin and S. B. Doty. “Modeling Material Symmetry”. In: *Tissue Mechanics.* New York, NY: Springer, 2007, pp. 139–167. ISBN: 978-0-387-49985-7. DOI: 10.1007/978-0-387-49985-7_5.

[126] W. L. Bond. “The Mathematics of the Physical Properties of Crystals”. In: *Bell System Technical Journal* 22.1 (1943), pp. 1–72. ISSN: 1538-7305. DOI: 10.1002/j.1538-7305.1943.tb01304.x.