N° d'ordre NNT : 2016LYSEi047 Année : 2016

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON
préparée au sein de
l'INSA de LYON

École Doctorale MEGA ED 162
Mécanique, Energétique, Génie Civil, Acoustique

Spécialité de doctorat :
Génie Mécanique

Soutenue publiquement le 19/05/2016, par :

Hassan AL AKHRAS
Ingénieur INSA de Lyon

Automatic Isogeometric Analysis Suitable Trivariate Models Generation – Application to Reduced Order Modeling

Devant le jury composé de :

David NERON	Professeur des Universités École Normale Supérieure de Cachan, France	Président du Jury
Jean-François REMACLE	Professeur Ordinaire Université Catholique de Louvain, Belgique	Rapporteur
Alain RASSINEUX	Professeur des Universités Université de Technologie de Compiègne, France	Rapporteur
Yuri BAZILEVS	Professor University of California, USA	Examinateur
Francisco CHINESTA	Professeur des Universités École centrale de Nantes, France	Examinateur
Michel ROCHETTE	Directeur de Recherche ANSYS, France	Examinateur
Anthony GRAVOUIL	Professeur des Universités INSA de LYON, France	Directeur de thèse
Thomas ELGUEDJ	Maître de Conférences HDR INSA de LYON, France	Co-directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **http://www.edchimie-lyon.fr** Sec : Renée EL MELHEM Bat Blaise Pascal 3e etage secretariat@edchimie-lyon.fr Insa : R. GOURDON	M. Stéphane DANIELE Institut de Recherches sur la Catalyse et l'Environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 avenue Albert Einstein 69626 Villeurbanne cedex directeur@edchimie-lyon.fr
E.E.A.	**ELECTRONIQUE, ELECTROTECHNIQUE, AUTOMATIQUE** **http://edeea.ec-lyon.fr** Sec : M.C. HAVGOUDOUKIAN Ecole-Doctorale.eea@ec-lyon.fr	M. Gérard SCORLETTI Ecole Centrale de Lyon 36 avenue Guy de Collongue 69134 ECULLY Tél : 04.72.18 60.97 Fax : 04 78 43 37 17 Gerard.scorletti@ec-lyon.fr
E2M2	**EVOLUTION, ECOSYSTEME, MICROBIOLOGIE, MODELISATION** **http://e2m2.universite-lyon.fr** Sec : Safia AIT CHALAL Bat Darwin - UCB Lyon 1 04.72.43.28.91 Insa : H. CHARLES Safia.ait-chalal@univ-lyon1.fr	Mme Gudrun BORNETTE CNRS UMR 5023 LEHNA Université Claude Bernard Lyon 1 Bât Forel 43 bd du 11 novembre 1918 69622 VILLEURBANNE Cédex Tél : 06.07.53.89.13 e2m2@ univ-lyon1.fr
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTE** **http://www.ediss-lyon.fr** Sec : Safia AIT CHALAL Hôpital Louis Pradel - Bron 04 72 68 49 09 Insa : M. LAGARDE Safia.ait-chalal@univ-lyon1.fr	Mme Emmanuelle CANET-SOULAS INSERM U1060, CarMeN lab, Univ. Lyon 1 Bâtiment IMBL 11 avenue Jean Capelle INSA de Lyon 696621 Villeurbanne Tél : 04.72.68.49.09 Fax :04 72 68 49 16 Emmanuelle.canet@univ-lyon1.fr
INFOMATHS	**INFORMATIQUE ET MATHEMATIQUES** **http://infomaths.univ-lyon1.fr** Sec :Renée EL MELHEM Bat Blaise Pascal 3e etage infomaths@univ-lyon1.fr	Mme Sylvie CALABRETTO LIRIS – INSA de Lyon Bat Blaise Pascal 7 avenue Jean Capelle 69622 VILLEURBANNE Cedex Tél : 04.72. 43. 80. 46 Fax 04 72 43 16 87 Sylvie.calabretto@insa-lyon.fr
Matériaux	**MATERIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec : M. LABOUNE PM : 71.70 –Fax : 87.12 Bat. Saint Exupéry Ed.materiaux@insa-lyon.fr	M. Jean-Yves BUFFIERE INSA de Lyon MATEIS Bâtiment Saint Exupéry 7 avenue Jean Capelle 69621 VILLEURBANNE Cedex Tél : 04.72.43 71.70 Fax 04 72 43 85 28 Ed.materiaux@insa-lyon.fr
MEGA	**MECANIQUE, ENERGETIQUE, GENIE CIVIL, ACOUSTIQUE** **http://mega.universite-lyon.fr** Sec : M. LABOUNE PM : 71.70 –Fax : 87.12 Bat. Saint Exupéry mega@insa-lyon.fr	M. Philippe BOISSE INSA de Lyon Laboratoire LAMCOS Bâtiment Jacquard 25 bis avenue Jean Capelle 69621 VILLEURBANNE Cedex Tél : 04.72 .43.71.70 Fax : 04 72 43 72 37 Philippe.boisse@insa-lyon.fr
ScSo	**ScSo*** **http://recherche.univ-lyon2.fr/scso/** Sec : Viviane POLSINELLI Brigitte DUBOIS Insa : J.Y. TOUSSAINT viviane.polsinelli@univ-lyon2.fr	Mme Isabelle VON BUELTZINGLOEWEN Université Lyon 2 86 rue Pasteur 69365 LYON Cedex 07 Tél : 04.78.77.23.86 Fax : 04.37.28.04.48

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Abstract

We present an effective method to automatically construct IsoGeometric Analysis (IGA) suitable trivariate B-spline models of complicated geometry and arbitrary topology. Our method takes as input a solid model defined by its triangulated boundary. The algorithm includes three main steps:

- Using cuboid decomposition, an initial polycube approximating the input boundary mesh is built. We begin by decomposing the triangulated input mesh into a set of pants patches. Such segmentation decomposes a complicated surface into a set of shapes that have a trivial topology: genus-0 surfaces with 3 boundaries. We then decompose each pants patch into a set of cuboid patches. Each cuboid is one boxed region enclosed by 6 disk-like surfaces.

- The polycube serves as the parametric domain of the tensor-product spline representation required for IGA. The polycube's nodes and arcs decompose the input model locally into quadrangular patches, and globally into hexahedral domains. Using aligned global parameterization, the nodes are re-positioned and the arcs are re-routed across the surface in a way to achieve low overall patch distortion, and alignment to principal curvature directions and sharp features.

- Based on the optimized polycube and parameterization, compatible B-spline boundary surfaces are reconstructed. The interior volumetric parameterization is computed using Coon's interpolation and the B-spline surfaces as boundary conditions.

This method can be applied to reduced order modeling for parametric studies based on geometrical parameters. For models with the same topology but different geometries, this method allows to have the same representation: i.e., meshes (or parameterizations, in the case of IGA) with the same topology.

The efficiency and the robustness of the proposed approach are illustrated by several examples from the mechanical and medical domains.

Acknowledgments

"I am not a genius, I am just curious. I ask many questions.
And when the answer is simple, then God is answering."

- Albert Einstein

Contents

Introduction

Isogeometric Analysis (IGA) is a new computational approach that offers the possibility of seamless integration between Computer-Aided Design (CAD) and Computer-Aided Engineering (CAE). This method uses the same type of mathematical representation (spline representation), for both geometry and physical solutions, and thus avoids the costly forth and back transformations between CAD and CAE. Nevertheless, a prerequisite for IGA is the availability of analysis-suitable models.

Reduced Order Models (ROMs) are usually thought of as computationally inexpensive mathematical representations that offer the potential for near real-time analysis. Such techniques show great abilities to accelerate solutions for linear and non-linear problems, provide well suited strategies for parametric studies, and are essential for scenarios where real-time simulation responses are desired. Their construction requires accumulating a certain number of precomputations called snapshots. Nevertheless, a prerequisite for ROM is the availability of solution vectors with the same dimension associated to all snapshots (in order to avoid a projection step between different snapshots).

In this thesis, the problem of generating isogeometric analysis-suitable models is addressed. In the context of reduced order modeling, for different geometrical instances of the same topological model, the issue of having isotopological solutions is addressed by generating the same isogeometrical representation for all objects.

Aiming at converting boundary representation models into analysis-suitable models, this thesis focuses on one of the most crucial problems of the conversion process: the generation of a volume parameterization.

Surface (i.e., bivariate) parameterization can be viewed as a mapping from a surface embedded in $\mathbb{R}^3$ to a parametric (or canonical) domain embedded in $\mathbb{R}^2$. Surface parameterization would suffice if only the surface geometry is of interest. In many cases, the surface will enclose a volume. The basic problem is to develop a volume (i.e., trivariate) parameterization in such a way that the surface parameterization is preserved. The canonical domain must have the same topology as the model but simplified geometrical features. For instance, genus-0 surface models can be mapped to a sphere. However for models with more complex geometry and arbitrary topology, more complex domains are required.

Due to its tensor-product nature, tensor-product spline fitting demands that the parametric domain (may be composed of a set of sub-domains) keeps regular. In addition, the construction of trivariate splines requires parametric domains in $\mathbb{R}^3$. A popular shape abstraction method is to use polycubes. A polycube is a set of cubes consistently glued together. It can be used to approximate very roughly the geometry of an object while faith-

fully replicating its topology. Due to its highly regular structure, the polycube is suitable for serving as the parametric domain of the tensor-product trivariate spline representation.

The polycube domain must satisfy two requirements: it must be topologically equivalent to the input model and its cube sub-domains must be consistent. If these requirements are satisfied, the model's boundary can be mapped to cover the boundaries of all parametric sub-domains seamlessly and consistently. Namely each patch of the model's boundary is mapped to one of six faces of one cube of the polycube.

The advantage of this approach is that each local mapping is tensor-product regular and the global mapping is seamless between different adjacent patches. In addition, it is inherently volumetric in the sense that the surface parameterization (between the model's boundary and the polycube's boundary) can be trivially turned into a volume parameterization using interpolation on each cube. However the quality of the resulting volume parameterization strongly depends on the placement of polycube's corners on the model's boundary.

The polycube consists of nodes and arcs embedded in the surface. Locally, neighboring nodes and arcs partition the surface into quadrilateral patches, and globally neighboring quadrilateral patches form hexahedral domains (i.e., cuboids). By using polycubes as parametric domains, the problem of finding a volume parameterization of a solid model is simplified to a problem of finding a surface parameterization between the model's boundary and the polycube's boundary.

The isoparametric lines of this parameterization are then extracted and serve to define the quadrilateral control mesh required for tensor-product spline representations. This control mesh is said to be good if its elements are uniform, and its edges are orthogonal and aligned with the surface's geometric features. In addition, the quality of the control mesh is mainly affected by singularities which are vertices touched by more or less than four edges. These vertices are particularly important because they are the only ones where the control mesh is not a regular grid.

A recent trend are aligned global parameterizations which adapt the parameterization to the geometry of the surface by fitting its gradient to a smooth direction field interpolating reliable principal curvature directions and geometric features. This field can be seen as a kind of proxy for the parameterization. In other words, each of the required properties for a good parameterization (such as uniformity, orthogonality and singularities) can be redefined in terms of desired properties of the field. Thus the task is shifted from the definition of a surface parameterization to the design of a smooth direction field on the surface.

An important property of direction fields is their singularities. A direction field singularity will generate an irregular vertex or a non-quadrilateral polygon in the control mesh. Direction field design is the generation of a smooth direction field from a set of constraints. These constraints can be topological (i.e., imposed singularities) and/or geometrical (i.e., imposed directions). Topologically, the direction field is constrained by the polycube structure. This means that the direction field must be singular only at the position of irregular nodes of the polycube. Geometrically, the direction field must follow the geometric features of the surface.

The direction field, and hence the aligned global parameterization, are constrained by the polycube structure. Depending on the quality of the polycube's embedding in the

surface, the resulting parameterization may contain large distortions or even local non-injectivities due to fold-overs. Based on the gradient of the parameterization's objective functional, the polycube's embedding is optimized so as to arrive to a local optimum of global embedding quality.

Using the optimized aligned global parameterization, a structured point grid is generated on each patch. This point grid is used to fit the boundary spline surfaces. The spline volume is then obtained by interpolating the reconstructed spline surfaces.

In order to generate a volume parameterization of a given solid models two main issues have to be addressed. The first is the computation of the polycube embedding, i.e., the placement of the polycube's nodes and arcs on the boundary surface of the input model. The second is the computation of the optimized aligned global parameterization used to extract the structured point grid needed for spline fitting.

This thesis is organized as follows: Chapter 1 gives a general presentation of the context and the problematics addressed in this thesis; Chapter 2 presents model decomposition and the polycube's generation; Chapter 3 presents model parameterization and the computation of the optimized aligned global parameterization; Chapter 4 illustrates some results and potential applications of the proposed approach.

Chapter 1

IGA Based ROM

Introduction

It is often desired to forecast the behavior of real phenomena which depend on the geometry of objects by performing a numerical simulation. An essential ingredient of numerical simulations is to accurately and simply describe the geometry of objects in a digital environment (i.e., computers). This is the main task of Computer Aided Design (CAD). Generally speaking, physical phenomenons are formulated using Partial Differential Equations (PDEs) and can not be formally solved. Another essential ingredient of numerical simulations is to numerically find approximate solutions for these PDEs. This is the main task of Computer Aided Engineering (CAE).

Despite the fact that CAD and CAE are both essential ingredients and very closely related in numerical simulations, they usually use very different numerical geometric representations. Most CAD systems usually employ the so-called spline representation, while most CAE systems usually employ the so-called mesh representation. The typical situation in engineering practice is that geometrical objects are encapsulated in CAD systems as spline objects, and mesh objects needed for CAE systems are generated from them. The construction of meshes is costly, time consuming and creates inaccuracies. As a consequence, novel methods are required in order to bridge the gap between the worlds of CAD and CAE. This is the principal motivation behind Isogeometric Analysis (IGA).

Many modern numerical models of real-life physical phenomenons pose challenges when used in numerical simulations, due to complexity and large size. Even if computer architectures or numerical methods greatly improve, simulations of complex numerical problems may lead to strong numerical difficulties and especially long computational times. As a consequence, novel methods are required in order to tackle not only non-linear problems but also large scale and parametric problems. In addition, such methods are essential for scenarios where real-time simulation responses are desired (e.g. applications in the medical domain). This is the principal motivation behind Reduced Order Models (ROMs).

1.1 Computational Modeling

Computer science provides attractive and efficient tools for various mathematical modeling fields and industrial purposes. Numerical simulations have become an essential part

of physics and engineering, and of even more unexpected fields such as biology and social science. Such simulations are indispensable in situations where an experiment cannot be performed as for instance the task of inspecting the stability of a building in case of an earthquake. However, even in cases where an experiment could potentially be performed (e.g. in the development of a new product), it often makes sense to run a simulation instead of the real-world experiment in order to reduce development cost and/or time.

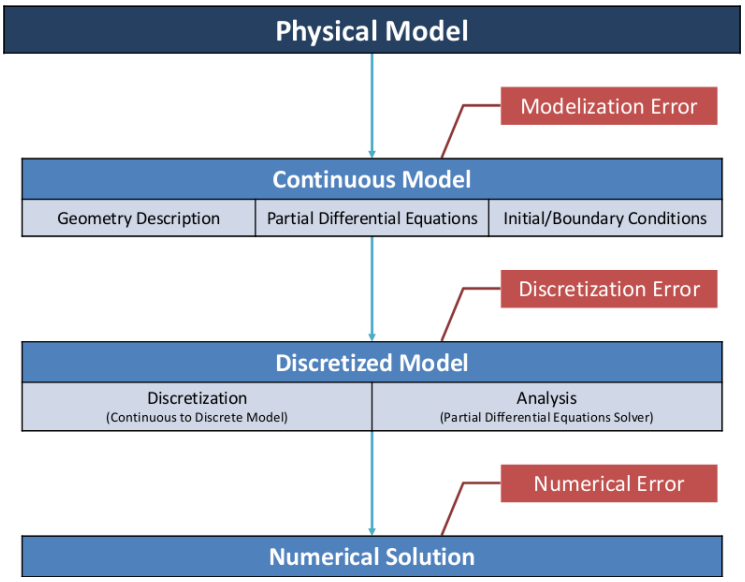


Figure 1.1: Overview of the different steps of computational modeling.

Before obtaining a workable numerical model, several modeling steps are required. These steps generate different kinds of error, which have to be estimated to qualify the relevancy of the final result. Figure 1.1 presents a brief overview of the different modeling phases:

- **Continuous model**: the physical model is put into equations by choosing the most suited mathematical models and governing laws. This step induces a modeling error.
- **Discretized model**: in general, the continuous model is formulated with PDEs and can not be formally solved. Geometry, characteristic variables and time are discretized, then numerical methods (e.g. isogeometric method, finite element method, finite difference method) are used for analysis. Analysis aims at finding approximate solutions for these PDEs. This step induces a disretization error.
- **Numerical resolution**: once the model is discretized, the problem is generally cast into an algebraic formulation. The resulting linear or non-linear system may be large, and suited solvers provide the desired solution. This step is affected by numerical errors (e.g. solver parameters, computer arithmetic, round-off errors).

The generation of a discretized model is the most time-consuming in a numerical analysis based on Finite Element Analysis (FEA). The numerical resolution might be time-consuming depending on the type/size of the numerical problem and may arouse memory storage issues.

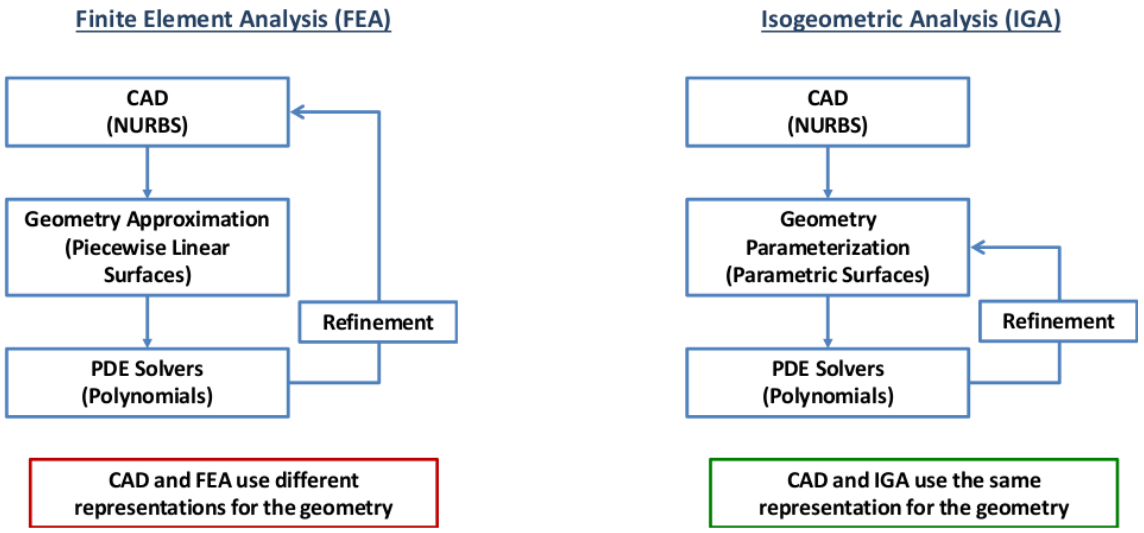


Figure 1.2: Geometrical representations in CAD and CAE: using FEA (a) and IGA (b).

In many of nowadays applications (e.g. car crash, flow around the wing of a plane, stability of a building), describing the accurate geometry of objects numerically is essential. With the rapid advances in scanning and acquisition techniques, it is now reasonably easy to obtain a numerical representation of existing objects through some automatic procedures. On step further, instead of replicating and enriching the real world in a digital environment, designers and engineers are able to utilize the enormous potential of today's modeling environments to create new complex objects. Due to this wide range of technologies and applications, there are different numerical geometric representations: existing objects are usually represented by the so-called mesh representation, and designed objects are usually represented by the so-called spline representation.

1.1.1 Computational Geometry: The Need for IGA

Spline representations of surfaces and more specifically tensor-product splines are widely used in CAD [Far02]. Splines are piecewise-defined by polynomial functions, and possesse a high degree of smoothness at the places where the polynomial pieces connect. In these representations, the geometry of a surface is usually defined by a structured quadrilateral control mesh. On the other hand, mesh representations of surfaces and more specifically polygonal meshes are widely used in CAE. Meshes are piecewise-defined by linear functions, and possesse only C^0 smoothness at the places where the linear pieces connect. In these representations, the geometry of a surface is usually defined by a triangle or quadrilateral mesh. Thus, the geometric representations in CAD and CAE are very different. The typical situation in engineering practice is that geometrical objects are encapsulated in CAD systems as spline objects, and mesh objects needed for CAE systems are generated from them. The construction of meshes is costly, time consuming and creates inaccuracies. In some instances mesh generation can be done automatically but in most circumstances it can be done at best semi-automatically. It is estimated that about 80% of overall

analysis time is devoted to mesh generation in the automotive, aerospace, and ship building industries [Hug05]. Once a mesh is constructed, refinement requires communication with the CAD system during each refinement iteration (Figure 1.2a). This link is very time-consuming and often unavailable.

Surface representation would only suffice if analysis only requires the surface geometry (e.g. stress or buckling analysis of a shell, linear analysis using the boundary element method). In many cases (e.g. non-linear analysis using the finite element method), the surface will enclose a volume and an analysis model will need to be created for the volume. The basic problem is to develop a volume (i.e., trivariate) representation of the solid in such a way that the surface (i.e., bivariate) representation is preserved. A volume spline scheme has gathered growing interest from both graphics and analysis research communities. Volume-based analysis has a computational advantage over traditional surface-based analysis because it is capable of expressing, in addition to the boundary, the interior physical space and materials of the model. It also promise to alleviate the burden of creating effective trivariate analysis-ready domains in many solid modeling and volume graphics applications.

The initial work on IGA was motivated by this existing gap between the worlds of CAD and CAE. IGA employs the same basis functions to represent both the geometry and the approximate solutions of PDEs, hence geometric representations in CAD and CAE are the same (Figure 1.2b). A primary goal of IGA is to have a geometrically exact representation no matter how coarse the discretization. Another goal is to simplify refinement by eliminating the need for communication with the CAD geometry once the initial analysis-suitable geometric representation is constructed.

1.1.2 Computational Analysis: The Need for ROM

Over the last decades, evolution of computer and supercomputer architectures provided a first response to more demanding communities. This hardware evolution with the High Performance Computing (HPC) was intuited by Moore's law [Moo65]. The overall computational improvement is also due to a simultaneous evolution of numerical methods [Glo09]. Numerical models, which take into account more realistic physical phenomenons, are becoming larger and more complex. Figure 1.3 presents different types of numerical models with increasing complexity. Even if computer architectures or numerical methods greatly improve, simulations of complex numerical problems may lead to strong numerical difficulties and especially long computational times.

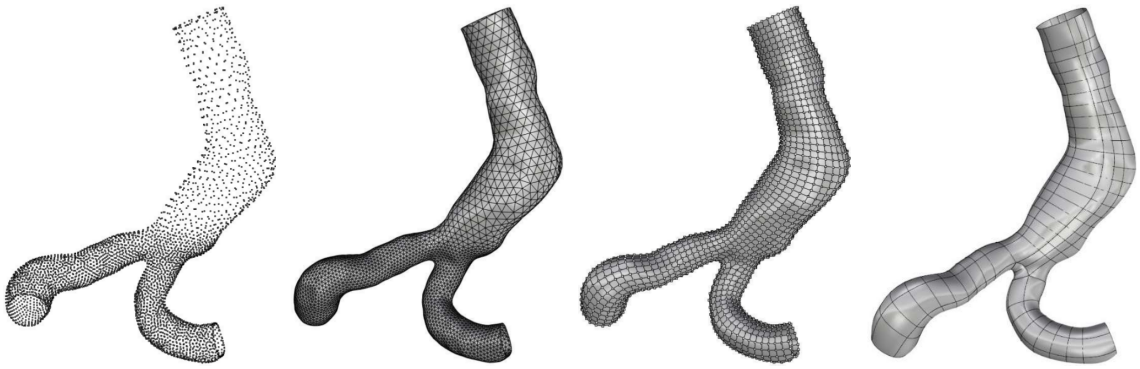
ROMs aim to lower the computational complexity of such problems. Reduced order modeling techniques show great abilities to accelerate solutions for numerous linear and non-linear problems, and provide well suited strategies for parametric studies. They are also essential for scenarios where real-time simulation responses are desired (e.g. applications in the medical domain). Such methods consist in solving a problem into a reduced subspace which is expected to capture the "most dominant trends" of the solution. There are specific tools to design reduced basis that capture the most relevant and the most contributory components (or modes) of the solution. Thereby, one can expect to write the

Figure 1.3: Different types of numerical models with increasing complexity: Mono-physics linear model (a); Multi-physics linear model (b); Multi-physics parametric linear model (c); Multi-physics parametric non-linear model (d).

solution with a sample of few modes (reducibility property). This result in cheap reduced basis computations sparing most of the computational work.

1.2 Isogeometric Analysis

IGA, introduced by Hughes et al. [Hug05], is a numerical analysis technique where the same basis functions are used to represent both the geometry and the approximate solutions of PDEs. This method is a promising approach to bridge the gap between CAD and CAE. The first advantage of IGA is that it uses higher-order continuity basis functions (such as B-splines) to approximate physical fields compared to classical finite elements used in FEA. Another advantage is that since IGA uses the same geometric representation for all design and analysis tasks, it allows to perform the analysis on the CAD geometry rather than the approximate piecewise linear geometry. Furthermore splines provide refinement possibilities and therefore refinement does not require any interaction with the original



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

Figure 1.4: Models acquisition using scanning techniques. These scanners create a point cloud (a) that is triangulated using a reconstruction software (b). Unfortunately, this type of output is not usable in a CAD context: a structured quadrilateral control mesh has to be created (c) in order to generate the sought spline representation (d).

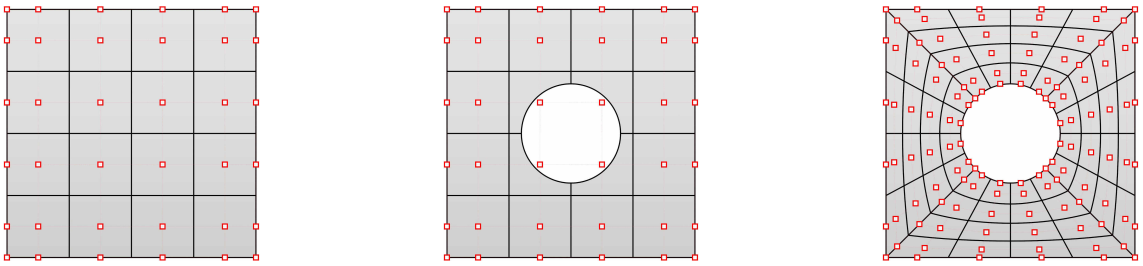
CAD model as in FEA. Simple operators such as knot insertion or degree elevation allow us to perform any kind of refinements.

Nevertheless, a prerequisite in IGA is the availability of solid models represented by trivariate tensor product splines. Geometry parameterization in IGA plays the same role as geometry discretization (mesh generation) in FEA. Unlike surface parameterization, volume parameterization involves parameterization of both boundary and interior of the model. In fact, due to computational complexity, volume parameterization is considered a big challenge and its quality has a big impact on the analysis [Coh10].

With the rapid advances in scanning and acquisition techniques, it is now reasonably easy to obtain a triangle mesh representation of an existing object through some automatic procedures. For instance, to obtain a triangle mesh from a real object, it is possible to use a range laser scanner [Lev00]. These scanners create a point cloud that is triangulated by a companion reconstruction software [Kaz06; Hor06]. The final result is a dense triangle mesh of the surface of the object.

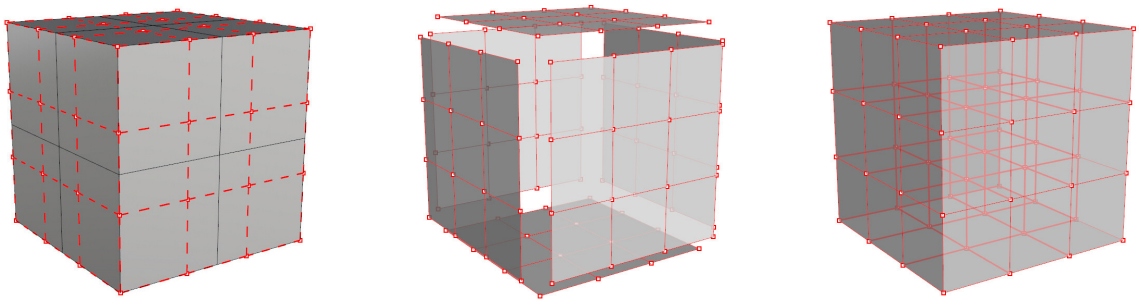
However in general, the obtained triangle meshes can not be directly used as control meshes of trivariate spline representation required for IGA. The reason is because their cells are triangular instead of quadrilateral, and because they are too dense as high-resolution meshes are usually used to capture the least detail of the objects. Hence a trivariate spline representation with respect to the given triangulated boundary has to be generated (Figure 1.4).

Non-Uniform Rational B-Splines (NURBS) are the most widely used in CAD [Pie97; Far99; Rog01; Coh01; Far02]. The major strengths of NURBS are that they are convenient for free-form surface modeling (through the manipulation of control mesh's points, the shape of the underlying surface can be smoothly modified, while preserving its initial continuity), can exactly represent all conic sections (e.g. circles, cylinders, spheres, ellipsoids), and there exist many efficient and numerically stable algorithms to generate



(a) (b) (c)

Figure 1.5: Boolean operations. A plate with the underlying control net (a). A circular hole is cut into the plate by trimming: the underlying control net is still the same (b). Analysis-suitable representation of the plate with hole (c).



(a) (b) (c)

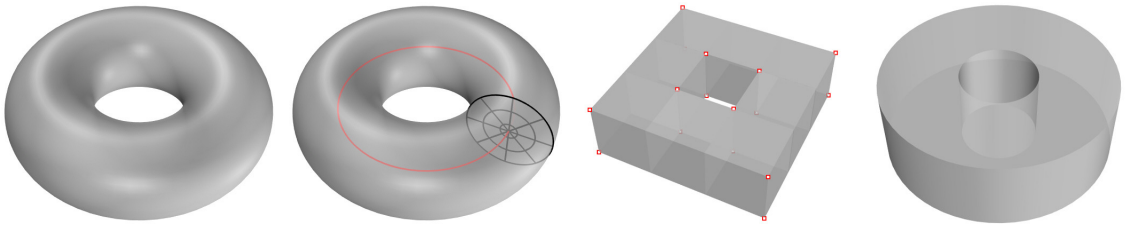
Figure 1.6: Boundary representations. A cube with the underlying control net (a). The cube appears as a volumetric object but the underlying description is defined as a set of boundary surfaces (b). Analysis-suitable trivariate representation of the cube (c).

NURBS objects. They also possess useful mathematical properties, such as the ability to be refined through knot insertion, C^{p-1}-continuity for p^{th}-order NURBS, and the variation diminishing and convex hull properties. Moreover, the normal and all the differential quantities (such as derivatives, geodesics, curvatures) of the surface can be precisely computed at arbitrary parametric values without resorting to any numerical approximations. In the context of this thesis, NURBS functions will be used as basis functions for IGA.

However in general, standard spline models can not be directly used in IGA. The reason is because CAD modelers usually use boolean operations that lead to trimmed surface models (Figure 1.5) and employ the Boundary Representation (B-Rep) method for representing volume models (Figure 1.6). Hence a volume parameterization with respects to the given boundary (possibly trimmed) spline surfaces has to be generated.

1.2.1 Analysis-Suitable Parameterization

Constructing analysis-suitable parameterization from a given solid model, defined by its boundary triangle mesh or boundary (possibly trimmed) spline surfaces, remains one of the most significant challenges in IGA. For a given object, various computational domains can be constructed with the same shape but with different parameterizations. Cohen et al.



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

Figure 1.7: A torus (a) and its different parametric domains: the cylindrical domain produces inevitable degenerate points along the tube axis (depicted in red) (b); the polycube domain uses at least eight cubes and generates some irregular points (depicted in red) (c); the generalized polycube domain uses one cube with two opposite faces glued together and don't generate any irregular points (d).

[Coh10] studied the parameterization of computational domains in IGA, and showed that the parameterization quality has a great impact on analysis results and efficiency. Pilgerstorfer et al. [Pil14] showed that in IGA the condition number of the stiffness matrix, which is a key factor for the stability of the linear system, depends strongly on the parameterization quality. One basic requirement for computational domains in IGA is that the resulting parameterization should have no self-intersections (i.e., the mapping from the parametric domain to physical domain should be injective). Xu et al. [Xu13c] proposed a linear and easy-to-check sufficient condition for the injectivity of a trivariate B-Spline parameterization. In addition, the parameterization should also satisfy these requirements: the isoparametric elements should be as uniform as possible, and the isoparametric structure should be as orthogonal as possible.

Existing techniques to build analysis-suitable parameterizations generally follow different trends. There are methods that exploit information about representation and topology of volumes such as tetrahedral meshes or swept volumes. Martin et al. [Mar09] proposed a parameterization method for a generalized cylinder-type volume defined by a tetrahedral mesh. This method is based on discrete volumetric harmonic functions. After remeshing the tetrahedral mesh with a hexahedral mesh, a trivariate B-spline for the solid model is generated by an iterative fitting method. However, in terms of spline construction, the cylinder-like domain produces inevitable degenerate points along the tube axis (Figure 1.7b). For genus-zero solids, using adaptive tetrahedral meshing and mesh untangling techniques, Escobar et al. [Esc11] proposed a method to construct trivariate T-spline representation. Aigner et al. [Aig09] proposed a variational approach to construct NURBS parameterization of swept volumes, which cover many free-form shapes in CAD system like blades or propellers. In this method, a spline approximation for a solid model is found by solving a minimization problem with several penalty terms corresponding to particular features of the shape.

For models which are given in boundary representation, some methods take as input a spline boundary of the model. These methods are restricted to models with an explicit specific topology: homeomorphic to a cube or a set of cubes. For models described by six boundary (non-trimmed) B-spline surfaces, Xu et al. [Xu13b] proposed a variational

harmonic method to construct a trivariate analysis-suitable parameterization. It consists in a nonlinear optimization problem, in which a regular term is integrated into the optimization formulation to achieve more uniform and orthogonal isoparametric structure. For models topologically equivalent to a set of cubes and bounded by (non-trimmed) B-spline surfaces, Xu et al. [Xu13a] studied the volume parameterization of the multi-block computational domain. First, they found a volume parameterization with good uniformity and orthogonality for each single-block computational domain by solving a constrained optimization problem, in which the constraint condition is the injectivity sufficient conditions. Then they extended their constrained optimization problem to the multi-block case by adding the continuity condition between the neighboring B-spline volumes to the constraint term. From six boundary (non-trimmed) B-spline surfaces, Wang et al. [Wan14] proposed an efficient method by combining divide-and-conquer, constraint aggregation and the hierarchical optimization technique to obtain valid trivariate B-spline solids. For models described by six boundary (non-trimmed) NURBS surfaces, Xu et al. [Xu14] proposed a method for the construction of high-quality trivariate analysis-suitable parameterization based on a Möbius transformation. After performing a reparameterization on the boundary surfaces to achieve high-quality isoparametric structure without changing the shape, the inner control points and weights are constructed using a variational harmonic metric.

Other methods take as input a triangulation of the model's boundary. For a genus-zero geometry, Zhang et al. [Zha12] proposed a robust and efficient approach to construct injective solid T-splines. They created a parametric mapping between the boundary triangle mesh and the boundary of a unit cube, which is the parameter domain of the solid T-spline. To do this, eight vertices have to be selected by the user, which correspond to the eight corners of the cube. Then twelve curves are found via calculating the shortest path between each pair of the selected vertices. Based on these curves, the input mesh is divided into six sub-meshes, and each one is associated with one face on the unit cube. Then the main work was to map each sub-mesh to a planar unit square using surface parameterization based on harmonic mapping. For more general problems where objects have complex geometry and arbitrary topology, there are methods based on shape decomposition and abstraction. A popular shape abstraction method is to use polycubes [Tar04]. A polycube is a solid composed of cubes (Figure 1.7c). It can be used to approximate very roughly the geometry of an object while faithfully replicating its topology. Due to its highly regular structure, the polycube can be used as the parametric domain for volume parameterization and spline modeling. However, in practice, due to the complexity of shapes, polycubes are usually constructed manually, entailing considerable effort. It is still challenging problem to automatically create polycubes for high genus geometry and use them in constructing analysis-suitable trivariate splines. Based on Morse theory, the work of Zhang et al. [Zha12] has been extended by Wang et al. [Wan13] to models with arbitrary topology. To extract the topology of the input geometry, they used the saddle points of a smooth harmonic scalar field defined over the mesh. Based on these saddle points, a polycube whose topology is equivalent to the input geometry is built and it serves as the parametric domain for the solid T-spline. A polycube mapping is then used to build a one-to-one correspondence between the input triangulation and the polycube's boundary. This method needs the interaction of the user at two stages to construct the polycube domain: to select two extremum constraints which controls the harmonic field, and to choose four seeds points which controls the polycube generation. Choosing these

inputs without considering geometric features and symmetry can affect the polycube generation and may yield asymmetric parameterization. Recently, Li et al. [Li12] extended the conventional polycube to a Generalized PolyCube (GPC), which enables the curved cuboid representation of the elementary sub-volumes (Figure 1.7d). This enables the polycube map approach to be applied to more complex objects. After decomposing the entire model into a group of T-shaped patches, they decompose each T-shape patch into four connected cube-like sub-patches to obtain the polycube. As for the method proposed by Wang et al. [Wan13], this method still needs the interaction of the user to select four seed points which controls the GPC generation. Choosing these inputs without considering geometric features can greatly affect the quality of the generated polycube.

1.2.2 Volume Parameterization

Our work investigates objects with complex geometry and arbitrary topology given in boundary representations. Our goal is to generate a trivariate parameterization with respects to a given solid model defined by its boundary triangle mesh or boundary (possibly trimmed) spline surfaces. In order to have a unified framework for both input types, our method takes as input a solid model defined by its triangulated boundary. Strong results based on the theory of Delaunay triangulation have been developed for triangle meshes [Fis06], and have been applied for the generation of a surface triangle mesh approximating a piecewise smooth surface [Jam15]. In the context of this article, we used the softwares ANSYS [ANS] and Rhinoceros [Rhia] to convert a CAD model defined by its boundary (possibly trimmed) NURBS surfaces into a surface triangle mesh.

Assume that we are given a solid model defined by its triangulated boundary. In order to convert a triangle mesh into a spline surface, one of the main problems we need to tackle is the extraction of a good quadrilateral control mesh of the surface. This control mesh is said to be good if its elements are uniform, and its edges are orthogonal and aligned with the principal curvature directions of the surface. These properties make the control mesh optimum in an approximation point of view, and greatly help to reduce unwanted oscillations on the final spline surface built from it. In addition, the quality of the final quadrilateral mesh is mainly affected by singularities which are vertices touched by more or less than four edges. These vertices are particularly important because they are the only ones where the quadrilateral mesh is not a regular grid. A singular vertex can have different singularity indices, depending on the number of edges touching it. The total sum of singularity indices is a topological invariant that depends on the genus of the surface [Ray08]. Hence, the generation of quadrilateral meshes is an inherently global problem since local changes in the mesh structure usually propagate globally across the mesh. So the main problem in converting a triangular mesh into a spline surface is equivalent to the problem of finding a proper quadrangulation of the surface (Figure 1.8).

The most popular and actively researched class of quadrilateral-meshing techniques is the family of parameterization-based quadrilateral meshing methods [Bom13b]. A parameterization can be viewed as a mapping from a surface embedded in $\mathbb{R}^3$ to a canonical

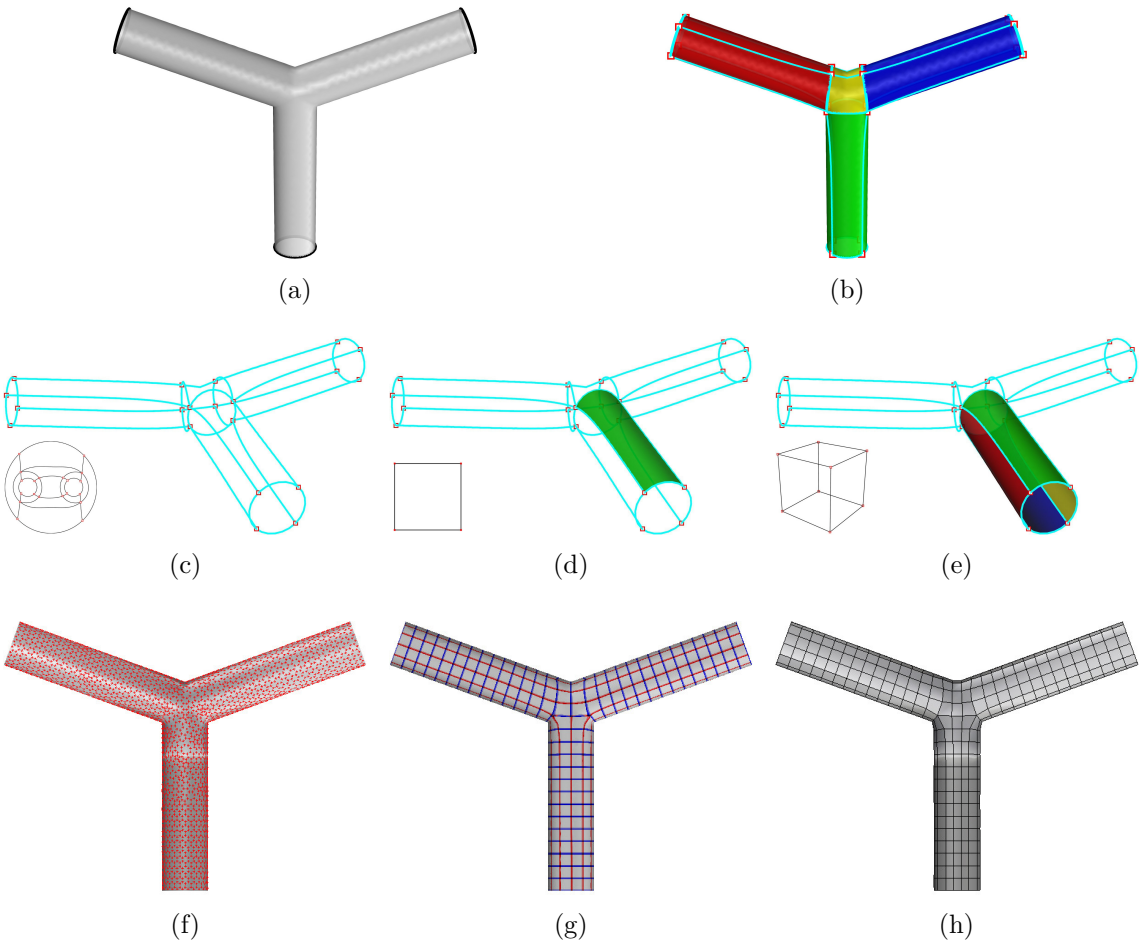


Figure 1.8: Quadrilateral mesh from aligned global parameterization. The input triangulated boundary surface homeomorphic to a pants patch (a). The polycube generated from cuboid decomposition (b). The polycube structure is equivalent to a quad layout (c). Locally, neighboring nodes and arcs partition the surface into quadrilateral patches (d). Globally, neighboring quadrilateral patches form cuboids (e). The cross field topologically conform with the quad layout, and geometrically aligned with the surface principal directions and boundaries (f). The global parameterization aligned with the cross field (g). The control quadrilateral mesh extracted from the global aligned parameterization (h).

domain embedded in $\mathbb{R}^2$. The isoparametric lines of this parameterization are then extracted and serve to define the quadrangulation, and hence the control mesh which has the desired properties. For surfaces with topologies other than the disk, the canonical domain must necessarily include discontinuities (e.g. cuts or seams). Methods based on local parameterization often lead to visible breakup of the isoparametric curves along seams. Global parameterization is of particular interest because it covers the entire surface, and hence can alleviate this problem when the translational and rotational discontinuities in the parameterization are compatible along cuts. A recent trend are aligned global parameterizations which adapt the parameterization to the geometry of the surface by fitting

the parameterization gradient to a smooth field interpolating principal directions and geometric features of the surface [Ray06; Käl07; Bom09]. Such techniques consist in first defining a cross field (i.e., a set of four orthogonal tangent unit vectors at each point of the surface), and then finding a parameterization such that its gradient field matches the cross field as much as possible. The cross field can be seen as a kind of proxy for the parameterization (more precisely, its gradient field). Interestingly, each of the required properties for a good parameterization (uniformity, orthogonality and singularities) can be redefined in terms of desired properties of the cross field. Thus the task is shifted from the definition of a good parameterization to the definition of a good cross field on the surface.

The definition of a good cross field implies, among other things, the good placing of singularities. In quadrilateral meshing, the field singularities play an important role. A field singularity will generate an irregular vertex or a non-quadrilateral polygon. On the global structure, each quadrilateral mesh implicitly defines a unique coarsest underlying patch layout with quadrilateral patches (or quadrilateral layout). Its patch borders are defined by those sequences of edges which form straight connections between irregular vertices [Bom11]. In other words, quadrilateral layouts are embedded graphs which partition surfaces into non-overlapping quadrilateral patches. In practice, the patch layout of a quadrilateral mesh is desired to be coarse and simple while still appropriately respecting the underlying geometry. In mesh processing, the quality of these patches is of high interest to support standard operations like high-order surface (e.g. B-splines or NURBS) fitting.

We are seeking a volume parameterization which involves parameterization of both boundary and interior of the model. A polycube is a solid composed of cubes. It can be used to approximate very roughly the geometry of an object while faithfully replicating its topology. Due to its highly regular structure, the polycube is suitable for serving as the parametric domain of the tensor-product spline representation required for trivariate NURBS-based IGA. To this end, starting with a triangulated boundary, we decompose the input model into cuboids in two steps: pants decomposition and cuboid decomposition. The obtained set of cuboids compose a polycube approximating the input model. The polycube consists of nodes and arcs embedded in the surface. Locally, neighboring nodes and arcs partition the surface into quadrilateral patches, and globally neighboring quadrilateral patches form hexahedral domains (i.e., cuboids). The process of decomposing the surface into cuboids using a set of nodes and arcs is equivalent to the process of constructing a quadrilateral layout. The polycube completely determines the topology of the quadrilateral layout (i.e., combinatorial structure). In addition, the polycube also adds another dimension to the quadrilateral layout because neighboring patches form cuboids.

In order to obtain a guiding field for the aligned global parameterization, we design a cross field on the surface mesh. Cross field design is the generation of a smooth cross field from a set of constraints. These constraints can be topological (i.e., imposed singularities) and/or geometrical (i.e., imposed directions). We strike for a balance between three important properties of the cross field: smoothness, singularity positions/indices,

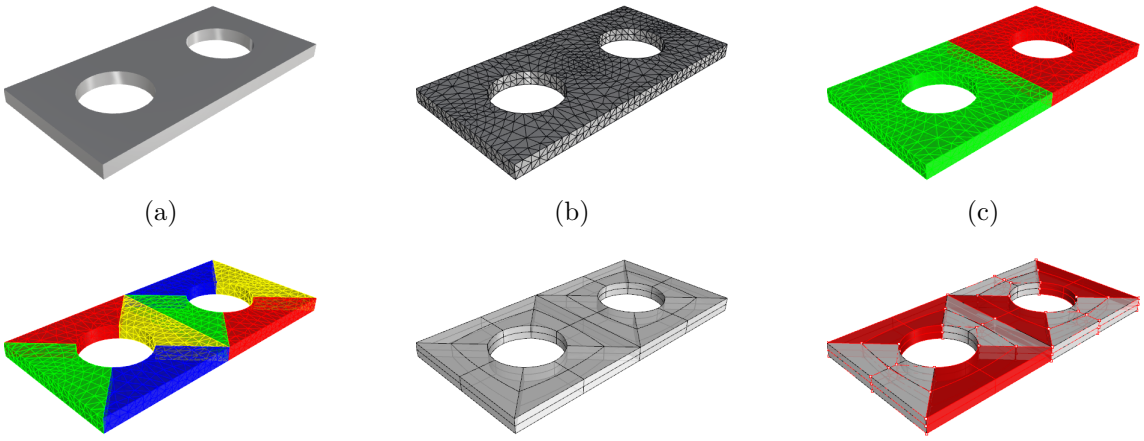


Figure 1.9: Trivariate NURBS Models Generation. The input solid region (a) along with its triangulated boundary surface (b). The boundary surface is first decomposed into pants patches (c). Each pants patch is further decomposed into 4 cuboids (d). For each cuboid, 6 compatible B-spline boundary surfaces are reconstructed (e). The volumetric parameterization from B-spline boundary surfaces (f).

and alignment with principal directions and geometric features. Topologically, the cross field is constrained by the polycube structure. This means that the cross field must be singular only at the position of irregular nodes of the polycube. Geometrically, the cross field must follow the principal directions and geometric features of the surface.

The aligned global parameterization is constrained by the polycube structure. Depending on the quality of the generated polycube, the resulting parameterization may contain large distortions or even local non-injectivities due to fold-overs. A practical solution to this problem is to optimize the geometric embedding of the polycube based on the gradient of the parameterization's objective functional so as to arrive to a local optimum of global embedding quality. The geometric embedding of the polycube describes the locations of its nodes and arcs as well as parameterizations of its patches. Based on the work of Campen et al. [Cam14], the initial geometric embedding of the polycube's nodes and arcs is optimized. Using global aligned parameterization, the nodes are re-positioned and the arcs are re-routed iteratively across the surface in a way to achieve low overall patch distortion, as well as alignment to principal curvature directions and geometric features.

1.2.3 Trivariate Models Generation

Our method takes as input a solid model defined by its triangulated boundary. The algorithm includes three main steps (Figure 1.9):

- The starting point of our algorithm is a triangulated mesh M bounding a solid model V. As the boundary of a solid region, $M = \partial V$ is a closed surface and can be of complex geometry and arbitrary topology. We begin by decomposing the input triangle mesh M into a set of pants patches. Such segmentation decomposes a complicated surface into a

set of shapes that have a trivial topology: genus-0 surfaces with 3 boundaries. The Euler characteristic χ for surfaces of most topological types are negative integers. For a pants patch $\chi = -1$, and therefore pants decomposition provides a canonical decomposition scheme for these surfaces. We then decompose each pants patch into a set of cuboid patches. Each cuboid is one boxed region enclosed by 6 disk-like surfaces. The idea is to generate nodes and arcs on each pants patch and decompose it into 4 connected components, each having 8 nodes and 12 arcs like a cuboid.

- The polycube's nodes and arcs decompose the input model locally into quadrilateral patches, and globally into hexahedral domains. In order to obtain a guiding field for the global parameterization, we design a cross field topologically conform with the polycube structure and geometrically following the principal directions and geometric features of the surface. The aligned global parameterization is constrained by the polycube structure, and may contain large distortions or even local non-injectivities due to fold-overs. Based on the gradient of the global aligned parameterization's objective functional, the geometric embedding of the polycube is optimize so as to arrive to a local optimum of global embedding quality.

- Using the quadrilateral mesh extracted from the optimized aligned global parameterization, a structured grid of points is generated on each patch, and then used to fit the boundary B-spline surfaces [Ma95]. For each cuboid, the volumetric parameterization is obtained using the reconstructed B-spline surfaces as boundary conditions. Keeping the boundary control points fixed, the interior control points of the B-spline solid are computed using Coons' interpolation [Wan14]. The positions of the interior control points are then adjusted by minimizing a Laplacien based energy.

1.3 Reduced Order Modeling

ROMs are usually thought of as computationally inexpensive mathematical representations that offer the potential for near real-time analysis. Generally speaking, numerical methods provide an algebraic formulation of the problem which has to be solved into a space $\mathcal{E}$ of dimension n (i.e., the number of degrees of freedom). The space $\mathcal{E}$ is spanned by the canonical basis composed of n unitary vectors (equaling 1 at a given degree of freedom and 0 elsewhere). Reduction methods consist in finding an appropriate basis composed of $r < n$ vectors (the so-called reduced basis) spanning a subset of $\mathcal{E}$ wherein the problem is solved. This way, computational work may be spared. For instance, in modal analysis of mechanical systems, the modal reduction method consists in choosing the subspace spanned by the first normal modes of the studied body (modal truncation method). A large literature shows how large is the field of applications of such techniques. Nevertheless, finding a basis providing both an attractive dimensional reduction and a relevant solution is challenging.

1.3.1 Reduction Methods

Reduction methods are distinguished between a posteriori approaches and a priori approaches. First, an a posteriori approach consists in prescribing a basis spanning a subspace before performing computations. Then, equations of the numerical model are pro-

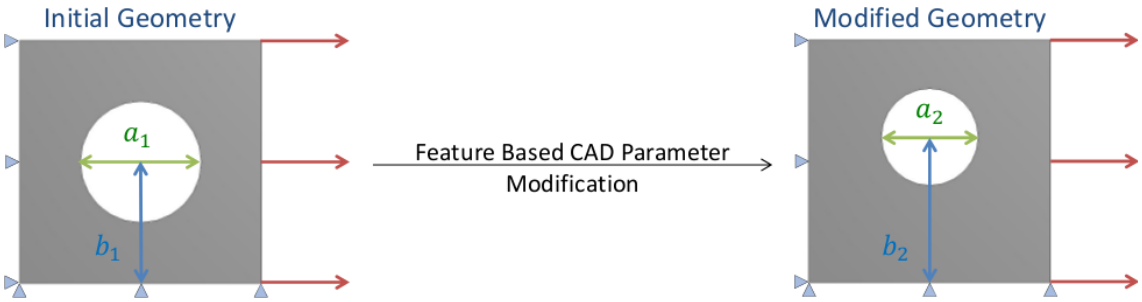


Figure 1.10: A parametric model where the geometric parameters a and b correspond to the diameter and center's position of the circular hole, respectively.

jected using e.g. a Galerkin method. Finally, sought solution is computed (online phase) using the resulting ROM. It generally requires a low computational effort and such a strategy is particularly suited for on-board computations and parametric problems. Both linear and non-linear problems can be tackled [Kun01; Ryc06; Ker11; Ams12]. Nevertheless, to design a working subspace some prior knowledge about the solution (called snapshots) are required. These prerequisites can be obtained from prior computations, analytic solutions, approximation of the solution on surrogate models, etc. Then, a relevant reduced basis for them is computed using for instance a Proper Orthogonal Decomposition (POD) or a Singular Value Decomposition (SVD). This preliminary phase is called offline phase and may be expensive.

On the contrary, a priori approach does not require prior knowledge about the solution. The reduced basis is computed and adapted on-the-fly during the resolution process. A widespread a priori reduced basis method is the Proper Generalized Decomposition (PGD) [Lad10; Chi10; Chi11; Bou13; Chi14]. It is a general method tackling various problems and consists in searching the solution into a low rank approximation (or separated form). Such a representation for the solution provides several advantages as far as the computational work is engaged and the storage memory is required. Nonetheless, drawbacks of this approach rely on the sustainability of the low rank format for the solution (lack of orthogonal properties on the basis).

Generally, efficient reduction methods involve a mixed approach between a posteriori and a priori ones. For instance, given a reduced basis, some calculations are performed on a ROM but results obtained do not satisfy a certain level of accuracy (a posteriori approach). Then, the considered reduced basis can be enriched to span an improved solution. A priori methods are able to fulfill this goal. To take another example, instead of initializing a priori methods from scratch, prior knowledge about the solution could be reused providing a first guess for the subspace to compute on-the-fly. That's why generally speaking, efficient strategies involve both a posteriori and a priori methods in complementary roles [Ryc05; Ker11; Gal11a; Gal11b; Hey13].

1.3.2 Snapshots and Parametric Studies

The main objective of parametric studies is the following: given a parametric model, for any value of an input parameter $\mu \in \mathbb{R}$ find the solution field $\mathbf{u}(\mu) \in \mathbb{R}^n$. In a structural mechanics context for instance, μ would be the value of an imposed displacement, a

(a) (b)

Figure 1.11: High-dimensional solution space represented by the triad Y and its embedded low-dimensional manifold on which the field variable $\mathbf{u}(\mu)$ resides (a). The solution corresponding to the parameter value μ^{new} is approximated by a linear combination of previously computed solutions $\mathbf{u}(\mu_i)_{1\leq i\leq m}$ corresponding to the previous design points (b). From [Ver03].

material behavior parameter, or even a shape parameter. The solution $\mathbf{u}(\mu)$ would be the displacement field, the stress field, or any additional variable computed by the behavior model on the domain. Figure 1.10 presents an example of a parametric model based on geometrical parameters.

The difficulty here stems from the fact that the solution field $\mathbf{u}(\mu)$ is a member of the infinite-dimensional solution space Y associated with the partial differential equations that are being solved. However, it can be intuited that the possible values of $\mathbf{u}(\mu)$ do not "cover" the entire solution space Y. If we imagine Y being reduced to a three-dimensional space, then $\mathbf{u}(\mu)$ can be conceived as lying on a curve or surface as depicted on Figure 1.11a. For example, again in a computational mechanics context, we expect that the displacement and stress fields which satisfies the governing equations does not vary randomly with the parameter μ, but in fact varies in a smooth fashion. In other words, the field variable is not some arbitrary member of the high-dimensional solution space associated with the partial differential equations; rather, it resides, or "evolves" on a much lower-dimensional manifold induced by the parametric dependence as depicted on Figure 1.11b [Ver03; Gal11a].

Let us define the considered parametric problem as a set of different linear static structural problems, one for each parametric configuration: $\mathbf{A}(\mu)\cdot\mathbf{u}(\mu)=\mathbf{b}(\mu)$, where $\mathbf{A}(\mu)$ is the usual stiffness matrix, $\mathbf{u}(\mu)$ the nodal displacement solution, and $\mathbf{b}(\mu)$ the external generalized forces. As explained previously, one can assume that the current problem, or current design point, still contains some characteristics of the previously computed ones. To compute a reduced basis approximation space for the solution subspace, a Gram-Schmidt process could be used for instance. By doing so, the different trends of snapshots can be identified and redundancy among snapshots is eliminated. Nevertheless, such an orthogonalization process does not provide the most dominant directions for snapshots. For that purpose, specific methods such as the POD and SVD are used. These methods originated mainly from the statistical field (principal component analysis, Karhunen-Loève decomposition, etc.) and aim to provide a reduced basis composed of most dominant vectors for snapshots.

Let $\mathbf{U}$ be a snapshot matrix. For instance, in the context of computational mechanics, generalized displacements (n degrees of freedom) for all snapshots corresponding to different parameters μ_j $(j = 1, \ldots, m)$ can be cast into $\mathbf{U}$ as follows:

$$\mathbf{U} = \begin{bmatrix} u_1(\mu_1) & \cdots & u_1(\mu_m) \\ \vdots & \ddots & \vdots \\ u_n(\mu_1) & \cdots & u_n(\mu_m) \end{bmatrix} = \begin{bmatrix} \mathbf{u}(\mu_1) & \cdots & \mathbf{u}(\mu_m) \end{bmatrix}. \tag{1.1}$$

$\mathbf{U} \in \mathbb{R}^{n \times m}$ is a real rectangular matrix and $\mathbf{u}(\mu_j)$ are snapshots of the generalized displacement field. According to the singular value theorem [Eck36], $\mathbf{U}$ can be factorized using SVD:

$$\mathbf{U} = \mathbf{\Psi}\mathbf{\Sigma}\mathbf{\Phi}^T = \begin{bmatrix} \boldsymbol{\psi}_1 & \cdots & \boldsymbol{\psi}_n \end{bmatrix} \begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_l \\ \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}_1^T \\ \vdots \\ \boldsymbol{\phi}_m^T \end{bmatrix}, \tag{1.2}$$

with $l = \min(n, m)$, $\mathbf{\Psi} \in \mathbb{R}^{n \times n}$ an unitary matrix containing left-singular vectors $\boldsymbol{\psi}$, $\mathbf{\Phi} \in \mathbb{R}^{m \times m}$ an unitary matrix containing right-singular vectors $\boldsymbol{\phi}$, and $\mathbf{\Sigma} \in \mathbb{R}^{n \times m}$ a rectangular diagonal matrix containing positive singular values σ in decreasing amplitudes. This decomposition is unique up to an arbitrary sign for pair $(\boldsymbol{\psi}, \boldsymbol{\phi})$. The SVD factorization can be rewritten into the following rank one expansion:

$$\mathbf{U} = \sum_{k=1}^{l} \sigma_k \boldsymbol{\psi}_k \boldsymbol{\phi}_k^T. \tag{1.3}$$

Given a matrix $\mathbf{U} \in \mathbb{R}^{n \times m}$ whose entries are denoted by u_{ij}, the matrix p-norm is defined as follows:

$$\|\mathbf{U}\|_p = \left[\sum_{i=1}^{n} \sum_{j=1}^{m} | u_{ij} |^p \right]^{\frac{1}{p}}. \tag{1.4}$$

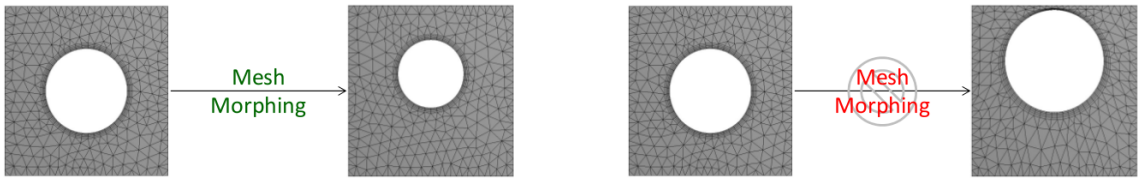
For $p = 2$, the Frobenius norm is defined and is denoted by $\|\mathbf{U}\|_F$. Taking into account only the $r \leq l$ first singular values of the SVD expansion of $\mathbf{U}$ allows to define a low rank approximation of $\mathbf{U}$ denoted by $\bar{\mathbf{U}}$:

$$\bar{\mathbf{U}} = \sum_{k=1}^{r} \sigma_k \boldsymbol{\psi}_k \boldsymbol{\phi}_k^T. \tag{1.5}$$

According to the Eckart-Young's low rank approximation theorem [Eck36], $\bar{\mathbf{U}}$ is the best approximation of rank r of $\mathbf{U}$ with respect to the Frobenius' norm. Moreover, if $r = k$ then $\bar{\mathbf{U}} = \mathbf{U}$. The relative error between the snapshot matrix $\mathbf{U}$ and its SVD approximation $\bar{\mathbf{U}}$ is given by:

$$\varepsilon(\mathbf{U}) = \frac{\|\mathbf{U} - \bar{\mathbf{U}}\|_F}{\|\mathbf{U}\|_F} = \sqrt{\frac{\sum_{i=r+1}^{k} \sigma_i^2}{\sum_{i=1}^{k} \sigma_i^2}}. \tag{1.6}$$

Generally, r is chosen in such a way that a great compression is gained and the corresponding low rank approximation is sufficiently accurate (e.g. $\varepsilon(\mathbf{U}) < 0.1$). The dimension r of the reduced basis is then of a key importance, because the gain in computational time is directly related to it: the smaller r, the faster the computations.



(a) (b)

Figure 1.12: A parametric model where the geometric parameters correspond to the center's position and diameter of the circular hole. For small variation of input parameters, general mesh morphing techniques will work well (a), but will fail for large variations (b).

Nevertheless, a good prerequisite for ROM construction (in order to avoid a projection step) is the availability of solution vectors with the same dimension associated to all snapshots (the snapshot matrix $\mathbf{U}$ is a real rectangular matrix). In other words, their construction requires accumulating a certain number of simulations on isotopological meshes (i.e., meshes with the same number of nodes and connectivity) while modifying the input parameters. However for parametric studies based on geometrical parameters, isotopological meshes can be difficult to obtain when using general mesh morphing techniques in the case of a large variation of input parameters (Figure 1.12).

1.3.3 Isotopological Snapshots Generation

A morphing is a transformation that changes one shape into another through a seamless transition. In other words, it is a one-to-one function which associates each point of the first shape to the corresponding point of the second shape. If both shape have the same topology, the morphing function between them can be computed by first finding a canonical domain which have the same topology as both shapes. Then we compute the parameterization functions between each shape and the canonical domain. Finally we can express the morphing function as a composition of the parameterization functions. See Figure 1.13 for an illustration.

For models with different geometries but same topology, a potential solution to generate isotopological snapshots is the use of analysis-suitable isogeometric parameterization with the same canonical domain. In addition, isogeometric meshes are based on the parameterization of the model and can handle a large amount of distortions, hence can adapt to large variation of input parameters. A summarizing workflow is proposed on Figure 1.14.

Conclusion

IGA is a numerical analysis technique where the same basis functions are used to represent both geometry and the approximate solutions of PDEs. However, a prerequisite for IGA is the availability of solid models represented by trivariate tensor product splines. Starting with the triangulated boundary of the solid model, we build a polycube approximating the input boundary mesh. Then we design a cross field topologically conform with the polycube structure, and geometrically following the principal directions and geometric features of the surface. A global parameterization whose gradient field is aligned with the

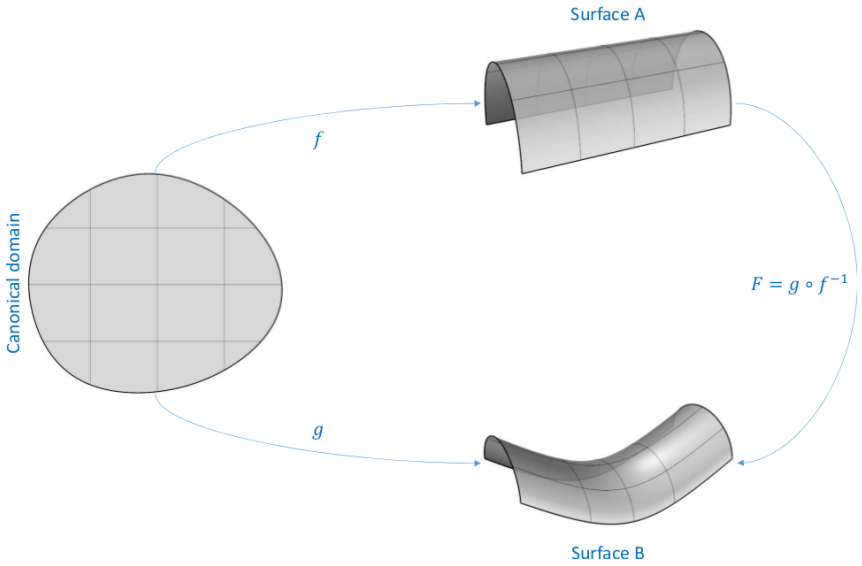
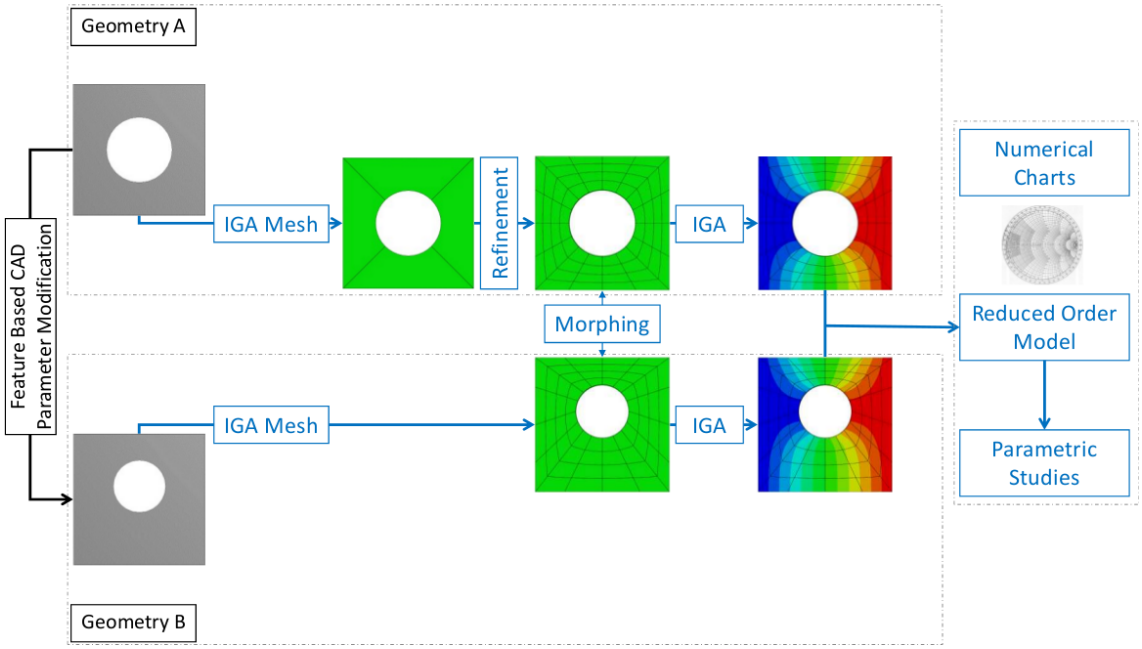


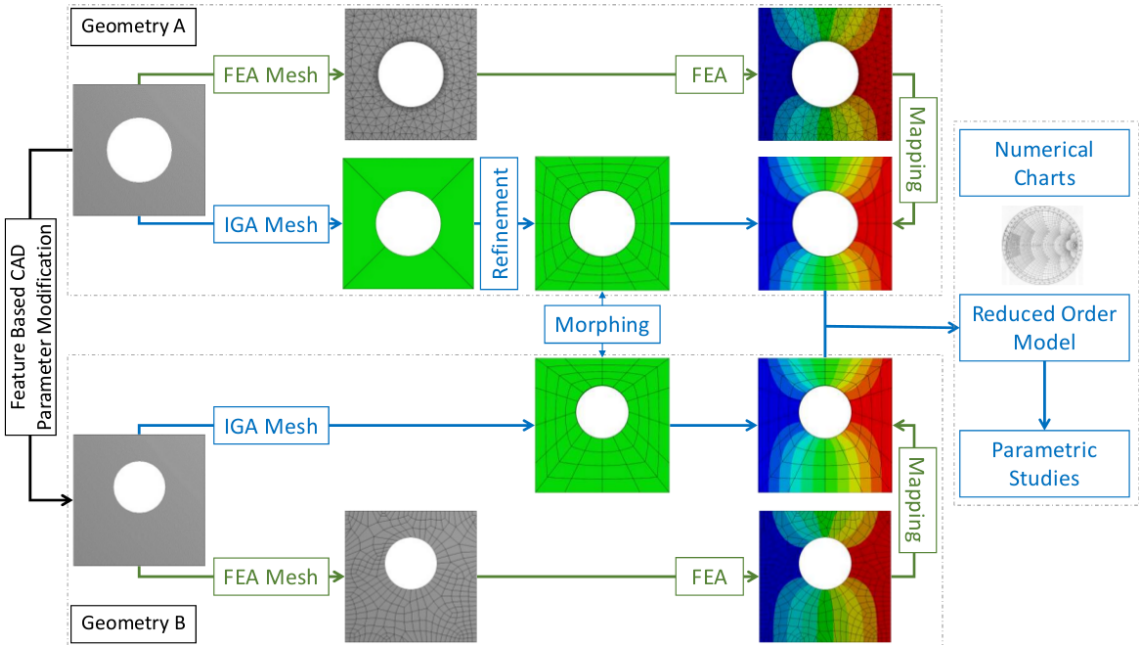
Figure 1.13: To compute the morphing function F between surface A and surface B, we begin by finding a canonical domain which have the same topology as both surfaces. Then we compute two parameterization functions f and g between the canonical domain and surfaces A and B, respectively. The morphing function F can by computed as a composition of the function f and g: $F = g \circ f^{-1}$. For shapes with different geometry but same topology, isotopological representations can be generated by using analysis-suitable isogeometric parameterization with the same canonical domain.

cross field is then computed. The isoparametric lines of this parameterization are then extracted and serve the define the control mesh for splines fitting.

ROMs are computationally inexpensive mathematical representations that offer the potential for near real-time analysis. In the context of parametric studies, their construction requires accumulating a certain number of system responses to different input excitations. The accumulated system responses are called snapshots. The ROM is constructed by using a compression method aiming at computing a basis spanning the snapshots. However, a prerequisite for ROM construction is the availability of snapshots with the same dimension. In other words, all pre-computations (corresponding to different input parameters) must be done on isotopological meshes. For models with different geometries but same topology, isotopological snapshots are generated using analysis-suitable isogeometric parameterization with the same canonical domain.



(a)



(b)

Figure 1.14: Isotopological snapshots generation: using IGA (a) and using FEA (b).

Chapter 2

Model Decomposition

Introduction

Volume-based spline modeling and analysis have gained much attention recently with many applications. Compared with surface splines, volume splines can represent both boundary and volumetric physical/material attributes of a model. This property makes volume representations highly preferable in many physically-bases applications including mechanical analysis. Hughes et al. [Hug05] have proposed IGA which uses trivariate spline basis functions to represent both the geometry and the approximate solutions of PDEs. Constructing analysis-suitable trivariate models from a given solid model, defined by its boundary triangle mesh or boundary (possible trimmed) spline surfaces, remains one of the most significant challenges in IGA [Cot09].

Due to the tensor-product nature, a single spline patch can only represent topological rectangles. Employing multiple patches seems a natural way of solving the issue to deal with various topological shapes. To handle models with complex geometry and arbitrary topology, methods in a divide-and-conquer fashion are the most promising. Compared with surface splines construction designed to extract features, volume splines construction mainly focus on finding part-aware component domains. Due to their tensor-product nature, spline fitting demands that the parametric domain keeps regular. In addition, the parametric domain must have the same topology of the model but simplified geometric features. A popular shape abstraction method is to use polycube domains. A polycube is a solid composed of cubes. It can be used to approximate very roughly the geometry of an object while faithfully replicating its topology. Due to its highly regular structure, the polycube can be used as the parametric domain required for trivariate spline fitting.

The current work investigates objects with complex geometry and arbitrary topology given in boundary representations. The first step toward a trivariate parameterization is the polycube generation. The needed background material in topology, geometric representations and surface parameterization techniques are introduced in Section 2.1. The input is a triangulation of the solid model's boundary. The boundary surface is decomposed into a set of cuboids in two steps: pants decomposition and cuboid decomposition (Section 2.2). Pants decomposition decomposes a complicated surface into a set of pants patches, i.e., shapes that have a trivial topology (Section 2.2.1). Cuboid decomposition

decomposes each pants patch into a set of cuboids, i.e., a boxed region enclosed by six disk-like patches (Section 2.2.2). This set of cuboids compose a generalized polycube approximating very roughly the input model's geometry while faithfully replicating its topology.

The novelty of the proposed method is the "geometry-aware" pants-to-cuboids decomposition algorithm. The algorithm is completely automatic and very robust even for low-quality and noisy meshes.

2.1 Background Material

2.1.1 Topology

This section briefly introduces the related background in topology. The reader is referred to the book of Hatcher [Hat01] for more details. *Topology* is the study of properties of a shape that are preserved under continuous deformations including stretching and bending, but not tearing or gluing. This includes such properties as continuity, connectedness and boundary.

Let X be a set. The elements x of X are usually called points. X is allowed to be the empty set. Let $\mathbf{N}$ be a function assigning to each point $x \in X$ a non-empty set N_x such that $x \in N_x$ and $N_x \subset X$. The set N_x is called the *neighborhood* of x with respect to $\mathbf{N}$. We denote by N the collection of sets N_x for all $x \in X$. Let X be a set and N a collection of subsets of X with the following properties: 1) the empty set $\emptyset \in N$ and the space $X \in N$; 2) the intersection of an arbitrary number of sets in N is also in N; 3) the union of a finite number of sets in N is also in N. Then we say that N is a *topology* on X and that the pair (X, N) is a *topological space* (Figure 2.1).

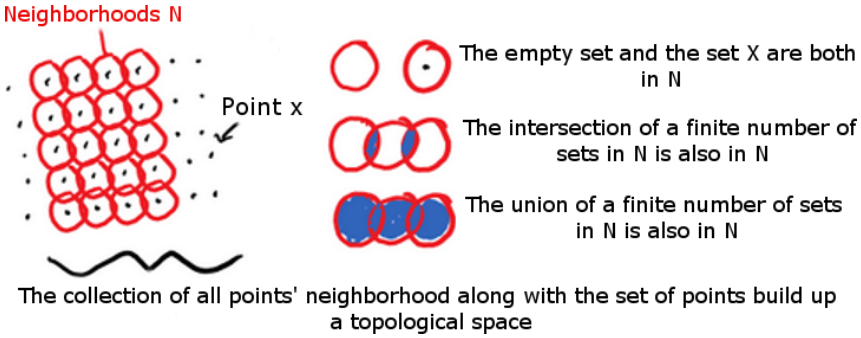


Figure 2.1: Topological space.

A *homeomorphism* is a continuous function between two topological spaces that has a continuous inverse function. Such transformation preserve all the topological properties of a given space. Two spaces with a homeomorphism between them are the same topologically and are called *homeomorphic* (Figure 2.2).

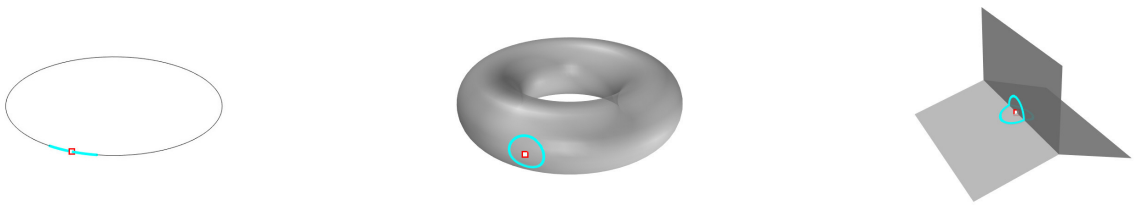
2.1.1.1 Manifolds

An *n-dimensional topological manifold* (or n-manifold) M is a topological space such that for each point $p \in M$, there exist an open neighborhood U of p in M and a continuous

Figure 2.2: Homeomorphic and non-homeomorphic surfaces.

bijective mapping $\mathbf{x} : D \to U$, where D is an open set in $\mathbb{R}^n$. In other words, M can be parameterized locally using maps $\mathbf{x}$. We will only be considering submanifolds of Euclidean spaces $\mathbb{R}^m$. Examples of manifold in $\mathbb{R}^3$ include curves (1-manifolds) and surfaces (2-manifolds) (Figure 2.3). A topological manifold $M \subseteq \mathbb{R}^m$ is a *smooth manifold* if for every point $p \in M$ there is a smooth regular map $\mathbf{x} : D \to M$, where $D \subseteq \mathbb{R}^n$ is open, such that $p \in \mathbf{x}(D)$. The smooth regular maps $\mathbf{x}$ are called *patches*. A mapping $f : M_1 \to M_2$ between smooth manifolds M_1 and M_2 is a smooth map if $\mathbf{x}_2^{-1} \circ f \circ \mathbf{x}_1$ is infinitely differentiable for any patches $\mathbf{x}_1$ of M_1 and $\mathbf{x}_2$ of M_2.

We will be interested in studying surfaces such as the closed unit disk which is not included in the current definition of a smooth manifold. An *n-dimensional manifold with boundary* (or n-manifold with boundary) M is a topological space such that for each point $p \in M$, there exist an open neighborhood U of p in M and a continuous bijective mapping $\mathbf{x} : D \to U$, where D is either an open set in $\mathbb{R}^n$ or an open set of the half plane $\mathbb{R}^n_+$. As before, we say that M is a smooth manifold with boundary if we can always choose maps $\mathbf{x}$ so that they are smooth and regular and we will refer to the smooth regular maps $\mathbf{x}$ as patches. A point $p \in M$ lies in the interior of M if there is an open neighborhood U of p in M and a continuous bijective mapping $\mathbf{x} : D \to U$, where D is an open set in $\mathbb{R}^n$. The set of all points that lie in the interior of M is called the *interior* of M and is denoted by $\text{int}(M)$. The set of all points p such that $p \notin \text{int}(M)$ is called the *boundary* of M and is denoted by ∂M. The boundary of a n-dimensional manifold is a $(n-1)$-dimensional manifold. If $\partial M = \emptyset$, M is a *n-dimensional manifold without boundary* (or n-manifold without boundary).



(a) (b) (c)

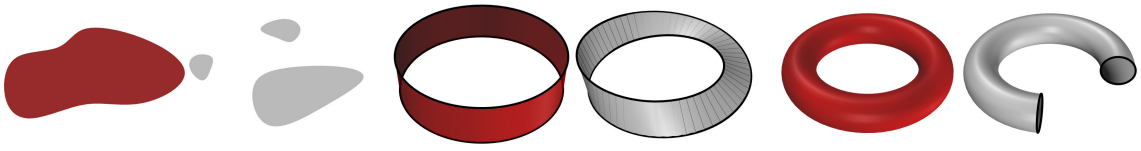
Figure 2.3: An n-manifold is a topological space that "locally looks like" the Euclidean space $\mathbb{R}^n$: curves are 1-manifolds (a) and surfaces are 2-manifolds (b). If the neighborhood of a point is not like $\mathbb{R}^n$, then the considered n-dimensional object is not a manifold (c).

Surfaces

A *surface* M is a 2-manifold, *i.e*, a topological space in which each point has a neighborhood homeomorphic to either the plane $\mathbb{R}^2$ or the closed half plane $\mathbb{R}^2_+$. Points with closed half-plane neighborhood are defined as the *boundary* ∂M of the surface M. In the context of this thesis, we will only consider connected, orientable, compact surfaces possibly with boundaries.

A *connected* surface is a topological space that cannot be represented as the union of two or more disjoint non-empty subsets (Figure 2.4a-left). If a surface can be represented in such a way, it is *disconnected* (Figure 2.4a-right). A surface in $\mathbb{R}^3$ is called *orientable*, if it is possible to distinguish between its two sides (Figure 2.4b-left). A *non-orientable* surface has a path which brings a traveler back to his starting point mirror-reversed (Figure 2.4b-right).

Triangulation is the division of a surface into a set of triangles, with the condition that each side (except the ones forming the boundary) of each triangle is entirely shared by two adjacent triangles. Every surface has a triangulation, sometimes with an infinite number of triangles. A *compact* surface is a surface admitting a finite number of triangles in its triangulation. A *closed* surface is a compact surface without boundaries (Figure 2.4c-left). Removing b disjoint open disks from a closed surface yields a compact surface with b disjoint boundary components (Figure 2.4c-right).



(a) (b) (c)

Figure 2.4: Connected (a-left) and disconnected (a-right) surfaces. Orientable (b-left) and non-orientable (b-right) surfaces. Compact surfaces without boundary (c-left) and with boundary (c-right).

The *genus* g of a connected, orientable, closed surface M is an integer representing the maximum number of cuttings along non-intersecting simple closed curves without rendering the resulting manifold disconnected. The genus g of a connected, orientable, compact surface M with boundaries is defined as the genus of the corresponding connected, orientable, closed surface (Figure 2.5).

Surfaces are topologically classified by their genus and number of boundaries, and characterized by the *Euler characteristic*. The Euler characteristic χ is a topological invariant, so that surfaces with different Euler characteristics cannot be homeomorphic (Figure 2.6). The Euler characteristic completely classify connected, orientable, compact surfaces up to homeomorphism. For a genus-g surface M with b boundary components, it is given by:

$$\chi(M) = 2 - 2g - b. \tag{2.1}$$

If M is a triangulated surface with vertices V, edges E, and faces F, then:

$$\chi(M) = |V| - |E| + |F|. \tag{2.2}$$

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

Figure 2.5: The genus of a surface is the maximum number of non-intersecting closed curves which can be drawn on it without disconnecting the surface: genus-0 (a-b) and genus-1 (c-d) surfaces.

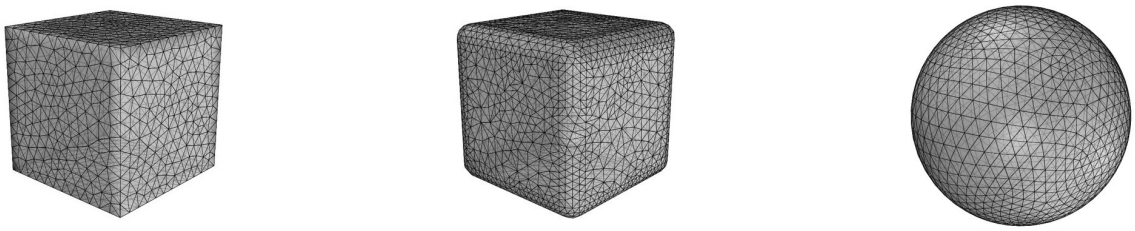


Figure 2.6: Different homeomorphic surfaces with the same Euler characteristic: $\chi = 2$.

2.1.1.2 Homotopy and Homology

A surface M is a topological space (2-manifold) where each point has a neighborhood homeomorphic to either the plane $\mathbb{R}^2$ (for interior points) or the half plane $\mathbb{R}^2_+$ (for boundary points). The surface M has genus-g and a boundary ∂M which is a collection of its b borders. The surface M considered here is compact, connected and oriented, so that each point has a unique normal vector $\mathbf{n}$.

Paths, Loops and Homotopy

A *path* p on a surface M is a continuous map $p : [0,1] \to M$. A *loop* is a closed path, meaning that the endpoints $p(0)$ and $p(1)$ coincide. The *concatenation* of two paths p and q, with $p(1) = q(0)$ is the path $p \circ q$ defined by:

$$(p \circ q)(t) = \begin{cases} p(2t) & \text{if } t \leq 1/2, \\ q(2t-1) & \text{if } t \geq 1/2. \end{cases} \tag{2.3}$$

Two paths p_1 and p_2 are *homotopic* $p_1 \equiv_t p_2$ if and only if one path can continuously evolve to the other one through a family of paths on the surface (Figures 2.7a-2.7b). Rigorously speaking, a *homotopy* between paths p_1 and p_2 is a continuous map $h : [0,1] \times [0,1] \to M$ subject to $h(0,t) = p_1$, $h(1,t) = p_2$, $h(s,0) = q_1$, $h(s,1) = q_2$, for all $s,t \in [0,1]$, where q_1 and q_2 are two paths joining $p_1(0)$ with $p_2(0)$ and $p_1(1)$ with $p_2(1)$, respectively.

A loop based at point x of the surface M is contractible if it is homotopic to the constant loop based at x, i.e., the loop that stays at x throughout. We denote the homotopy equivalence class of path p as $[p]$. The set of homotopy equivalence classes of loops based at

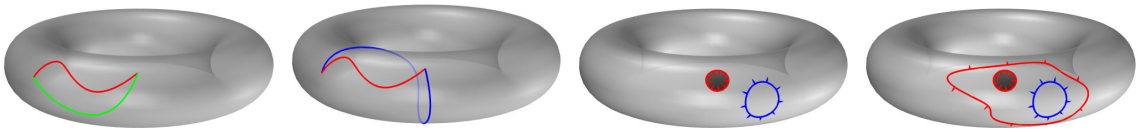
x forms a group under concatenation, called the fundamental group and denoted $\pi_1(M, x)$. The identity element of the fundamental group is the homotopy class of contractible loops.

Chain, Cycles and Homology

A *chain* γ on a surface M is an oriented 1-manifold embedded in M: $\gamma \subset M$. Chains are not necessarily connected and can be composed of multiple components. A *cycle* is a chain γ without boundaries: $\partial\gamma = \emptyset$. We introduce the following notions on chains and cycles:

- A chain γ is oriented, i.e., it has a unique tangent vector $\mathbf{t}_\gamma$ at each of its points which is also tangent to the surface M. Using this tangent vector, along with the surface normal $\mathbf{n}$, we can define a unique conormal vector $\mathbf{n}_\gamma = \mathbf{n} \times \mathbf{t}_\gamma$ on the chain, which ensures that $(\mathbf{t}_\gamma, \mathbf{n}_\gamma, \mathbf{n})$ forms a natural local orthonormal basis called the Dabroux frame.
- The reversal $-\gamma$ of a chain γ is the chain with opposite orientation: $\mathbf{t}_{-\gamma} = -\mathbf{t}_\gamma$ and $\mathbf{n}_{-\gamma} = -\mathbf{n}_\gamma$.
- ∂ is called the boundary operator, such that ∂M is the subset of points of the surface M with neighborhood homeomorphic to the half plane. This subset is a cycle γ. We can choose an orientation for this cycle γ by requiring its conormal to point outwards. In this case, we say that the cycle γ is a boundary.
- A chain γ is called exact if there exists a submanifold $\bar{M}$ of M such that $\gamma = \partial\bar{M}$. An exact chain is necessarily a cycle which is a boundary.
- If the surface M is a topological disk then $\gamma = \partial M$ is said contractible. The definition of contractibility for loops is that a loop is contractible if it is homotopic to a null loop. The definition of contractibility for cycles adds a notion of orientation as the reversal of a contractible boundary cycle is not necessarily contractible.

Two cycles γ_1 and γ_2 are *homologic* $\gamma_1 \equiv_l \gamma_2$ if and only if $\gamma_1 - \gamma_2$ is exact (Figures 2.7c-2.7d). More formally, the *homology* for cycles is defined as the quotient set of cycles over exact chains. In particular the zero of homology $\emptyset$ is the class of exact cycles. In other words, a cycle γ is exact if and only if $\gamma \equiv_l \emptyset$. Note that homological cycles might have different number of connected components. Homology is a more flexible concept than homotopy: homology allows to split a cycle in two or fusion two cycle into one whereas homotopy doesn't.



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

Figure 2.7: Homotopic (a) and non-homotopic (b) paths on a surface. Non-Homologic (c) and homologic (d) chains on a surface.

Cut Graphs, Homotopy and Homology Basis

Let $G = (V, E)$ be a graph with a node set V and an edge set $E \subset V \times V$. An embedding of the graph G in a surface M is a map which takes each node v of G to a distinct point on M and maps each edge $e = (v_i, v_j)$ of G to a curve on M which connects the images of v_i and v_j.

Cut Graph

A *cut graph* is an embedding of a graph G in a surface M whose compliment is a topological disk. A *system of loops* is a cut graph with only one node, i.e., all edges of the cut graph are loops that start and end at the same node.

The algorithm for computing a cut graph for a surface mesh M is relatively simple. A spanning dual tree $\mathcal{T}^*$ of dual edges is computed. The primal of all non-spanning dual tree edges is a cut graph which transforms M into a topological disk. The size of this cut graph can be significantly reduced by iteratively removing all open paths.

Homotopy Basis

Homotopy basis is a generalization of the system of loops. The homotopy basis of a genus-g surface consists of $2g$ loops whose homotopy classes generate the fundamental group $\pi_1(M, x)$ (Figure 2.8a). Every system of loops is a homotopy basis, but the converse is not true: homotopy bases can contain self-intersections or intersections that can not be removed by homotopy.

The algorithm for finding a greedy homotopy basis for a surface mesh M is actually very simple. A spanning tree $\mathcal{T}$ of primal edges and a spanning tree $\mathcal{T}^*$ of dual edges which does not cross $\mathcal{T}$ are computed. $2g$ primal edges will then not be contained in $\mathcal{T}$ nor crossed by dual edges in $\mathcal{T}^*$. Connecting these $2g$ primal edges to the root of $\mathcal{T}$ through $\mathcal{T}$ yields the $2g$ loops forming the homotopy basis.

Homology Basis

The set of cycles $H(M) = \left\{\gamma_i^H\right\}_{i=1,\dots,2g}$ is a *homology basis* on the genus-g surface M if it satisfies the following two conditions:

- Linear independence condition: $\sum a_i\gamma_i^H \equiv_l \emptyset \Leftrightarrow a_1 = \cdots = a_{2g} = 0$.
- Spanning condition: $\forall\gamma \in \mathcal{C}(M)$, $\exists a \in \mathbb{Z}^{2g}$ such that $\gamma \equiv_l \sum a_i\gamma_i^H$, where $\mathcal{C}(M)$ the set of all cycles on M.

In other words, any cycle γ on the surface M is homologic to a formal sum of the homology basis cycles. The homology basis of a genus-g surface consists of $2g$ cycles (Figure 2.8b). Any homotopy basis is also a homology basis, but not vice versa, since the cycles in a homotopy basis generally do not have a common point.

The tunnel and handle loops form a homology basis. Suppose a closed surface $S \subset \mathbb{R}^3$ separates $\mathbb{R}^3$ into a bounded space $\mathbb{I}$ and an unbounded space $\mathbb{O}$. Handle and tunnel loops on S can be defined as follows. A loop a_i is a tunnel if it spans a disk in the unbounded space $\mathbb{O}$. A loop b_i is a handle if it spans a disk in the bounded space $\mathbb{I}$. There are various

existing algorithms for computing these loops on surfaces. We use the algorithm proposed by Dey et al. [Dey07] which computes well defined handle and tunnel loops for three-dimensional models, and guarantees that the resulting loops are topologically correct and geometrically small.

(a) (b)

Figure 2.8: Homotopy (a) and homology (b) basis for a genus-2 surface. From [Eri05].

2.1.2 Geometric Representations

Surface representations are often divided into three major classes, namely implicit, explicit, and parametric representations. The main idea of implicit representations is based on the observation that a 2-manifold surface in $\mathbb{R}^3$ is of co-dimension 1 and consequently can be described as the kernel $K = \{\mathbf{p} \in \mathbb{R}^3 : f(\mathbf{p}) = 0\}$ of a single scalar function $f : \mathbb{R}^3 \mapsto \mathbb{R}$. For instance, a circle centered at the origin and with radius r can be described by the equation $f(\mathbf{p}) = 0 \rightarrow x^2 + y^2 - r^2 = 0$. However, explicitly evaluating points on the surface, for example to render the surface, is equivalent to a root finding process and thus typically very inefficient. In such situations explicit surface representations which are given as a set of geometric primitives like points or polygons are more advantageous. In computer graphics, the most prominent explicit representation is the triangle mesh, i.e. the object surface is given as a set of triangles.

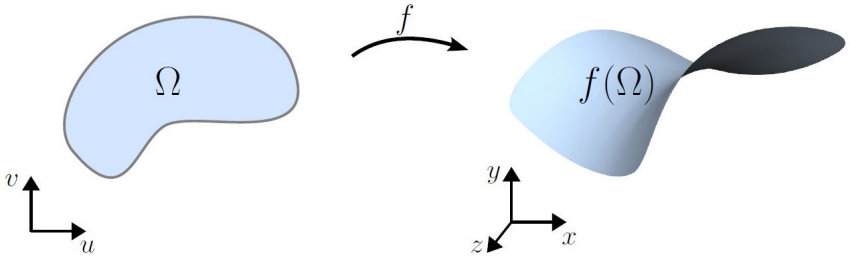


Figure 2.9: Parametric surface representation based on a continuous mapping between a domain $\Omega \subset \mathbb{R}^2$ and its embedding $f(\Omega) \subset \mathbb{R}^3$.

The main concept of parametric surface representations is to describe a surface through a mapping $\mathbf{f}(u, v) : \Omega \mapsto \mathbb{R}^3$ between a base domain $\Omega \subset \mathbb{R}^2$ and the embedding space $\mathbb{R}^3$ (Figure 2.9). In this setting the properties of the function (continuity, differentiability, etc.) are strongly connected to the shape of the surface and consequently the choice of an adequate function space is essential. Even more general is the concept of manifolds which enable the representation of topologically non-trivial objects by "stitching" several

parametric representations with the help of transition functions which guarantee compatibility in overlapping areas. CAD parametric representations often appear in the form of tensor product NURBS surfaces. The reason is that these piecewise polynomial surfaces are on the one hand equipped with a guaranteed smoothness but on the other hand still intuitively controllable by means on a net of control points.

2.1.2.1 Mesh Representations

In computers graphics, surface meshes are the classical representation of surfaces in $\mathbb{R}^3$. A *surface mesh* $M = (V, E, F)$ is formally a tuple of three sets, namely the *vertices* V, the *edges* E and the *faces* F. Each vertex $v_i \in V$ is equipped with the position of its embedding in $\mathbb{R}^3$: $\mathbf{p}(v_i) = \mathbf{p}_i \in \mathbb{R}^3$. Each edge $e_i \in E$ is a pair of two vertices: $e_i = (v_j, v_k)$. Each face $f_i \in F$ is a tuple of diverse vertices which are cyclically connected to form a topological polygon. For instance, all faces are triangles for a pure triangle surface mesh, and all faces are quadrilaterals for a pure quadrilateral surface mesh. We further require that the surface mesh be a topological 2-manifold. This requires that any two distinct faces of the mesh must either be disjoint, intersect each other at a common edge, or intersect each other at a common vertex. In addition, the faces of the mesh having a given vertex in common share edges in a cyclic way.

Neighborhood relations between vertices, edges and faces are defined in the usual graph theoretical sense. Two elements are said to be *incident* if the vertices of one are a subset of the vertices of the other. While *incidence* describes the neighborhood relation between elements of different dimensions, *adjacency* is a similar concept for entities of equal dimension. *Adjacent vertices* are incident to a common edges, *adjacent edges* are incident to a common vertex and *adjacent faces* overlap at a common edge. The valence of a vertex is defined to be the number of its incident edges. An edge is called *boundary edge* if it is incident to a single face, otherwise it is called *interior edge*. Vertices inherit the boundary property from edges: a vertex is called *boundary vertex* if it is adjacent to at least one boundary edge, otherwise it is an *interior vertex*.

An orientation of a face $f_i \in F$ of the mesh M is an ordering of its vertices $v_1, v_2, \ldots, v_n \in V$. We denote such an oriented face by $[v_1, v_2, \ldots, v_n]$. We say two orientations are the same if they differ from each other by an even permutation of the vertices. For example, the oriented triangle faces $[v_1, v_2, v_3]$ and $[v_2, v_3, v_1]$ have the same orientation, where $[v_1, v_2, v_3]$ and $[v_1, v_3, v_2]$ have opposite orientations. An orientation of an edge $e_i \in E$ is an ordering of its vertices $v_j, v_k \in V$. We will denote such an oriented edge by $[v_j, v_k]$. An edge e_i with vertices v_j and v_k has precisely two orientations, $[v_j, v_k]$ and $[v_k, v_j]$, so we say that these oriented edges have opposite orientations. An oriented face induces an orientation on its edges. For instance, the oriented triangle face $[v_1, v_2, v_3]$ induces the oriented edges $[v_1, v_2]$, $[v_2, v_3]$, and $[v_3, v_1]$. A surface mesh M is orientable if each face $f_i \in F$ can be oriented so that whenever two faces share a common edge, the orientations induced by the faces on the edge are opposite.

For a surface mesh $M = (V, E, F)$, its dual $M^* = (V^*, E^*, F^*)$ is given by an isomorphism which uniquely maps the primal k-dimensional entities to the dual $(2-k)$-dimensional ones and vice versa. More precisely, each vertex $v_i \in V$ is identified with the dual face $f_i^* \in F^*$, each edge $e_j \in E$ is identified with a dual edge $e_j^* \in E^*$, and each face $f_k \in F$ is identified with a dual vertex $v_k^* \in V^*$. The connectivity of the dual mesh is

uniquely inherited from the primal mesh: for instance, if two vertices are neighbored in the primal mesh, so will be the corresponding dual faces in the dual mesh.

Triangle Meshes

Triangle meshes are the most widely used. One reason is that a triangle (or 2-simplex) is in some sense the simplest 2-dimensional entity. A simplex is a generalization of the notion of a triangle to arbitrary dimensions. A k-simplex is the convex hull of its $k+1$ (non-collinear) vertices: for instance, a single point is a 0-simplex, a line segment is a 1-simplex, and a triangle is a 2-simplex. It is important to notice that a triangle mesh is not only an explicit surface representation but also possesses an intrinsic parameterization. Each triangle $(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k)$ can be parameterized by the barycentric linear mapping $\mathbf{\Gamma}(u, v) = u \cdot \mathbf{p}_i + v \cdot \mathbf{p}_j + (1 - u - v) \cdot \mathbf{p}_k$ with $u, v > 0$ and $u + v \leq 1$. In practice all these individual triangle mappings are often combined into one piecewise linear mapping.

Quadrilateral Meshes

Besides triangle meshes, today quadrilateral meshes enjoy a steadily increasing popularity. While triangle meshes easily enable adaptive element sizes, quadrilateral meshes exhibit a superior (mostly regular) structure. Quadrilateral meshes are often preferred over triangle meshes especially in animation and simulation. One reason is that their tensor-product nature easily generalizes to higher-order representations which are able to satisfy the C^2-continuity requirements that arise in many practical applications. However, it is important to notice that a general quadrilateral mesh is not an explicit geometry representation comparable to a triangle mesh. In contrast to a triangle, a quadrilateral might be non-planar and/or non-convex. Consequently apart from specialized applications which are restricted to the subset of convex and planar quadrilateral meshes, a quadrilateral mesh is usually used as the control mesh of a parametric surface like tensor-product NURBS.

It is well know that a simple polygon (*i.e.*, planar and non-intersecting) can always be triangulated [Fou84; Cha91]. In contrast to that, it is not always possible to quadrangulate a polygon. This observation indicates that the generation of quadrilateral meshes involve some global aspects which are not present in the generation of triangle meshes. To be useful in practice, a quadrilateral mesh typically has to fulfill strong quality requirements. Besides local properties like regularity, element orientation and element shape, also global properties like the patch structure usually play an important role. Consequently instead of local optimization strategies, as typically applied in the generation and optimization of triangle meshes, global optimization techniques are inevitable.

For a quadrilateral mesh, an interior vertex is called *regular vertex* if its valence is equal to 4, otherwise it is called *irregular* (or *singular* or *extraordinary*). Analogously, on the boundary a regular vertex is characterized by a valence of 3. Irregular vertices are particularly important because they are the only ones where the quadrilateral mesh is not a regular grid. They can have different singularity indices depending on their valence. The total sum of singularity indices is a topological invariant that depends on the genus of the surface [Ray08].

The relation between the valences $\mathrm{val}(v_i)$ of a quadrilateral mesh vertices $v_i \in V$ and

the genus g of the represented closed surface is given by [Bom12a]:

$$\sum_{i=1}^{|V|} (4 - \text{val}(v_i)) = 8\,(1 - g)\,. \tag{2.4}$$

The above formula is a necessary condition on the sum of vertex valences in a quadrilateral mesh which represents a genus-g surface. This formula shows that for closed surfaces a quadrilateral mesh where all vertices are regular can be found only if the genus of the surface is 1. A genus-0 surface will require a total valence defect of 8. Therefore a valid set of irregular vertices would be 8 irregular vertices with valence 3 (each having a valence defect of 1). However, there are infinitely many different possibilities, since positive and negative valence defects cancel out. For instance, a valence 3 and a valence 5 irregular vertices together have a valence defect of 0. Even for genus-1 surfaces, it is often desirable to introduce irregular vertices if the surface is more complex than a torus, like e.g. a coffee cup.

The *base complex* is a unique partitioning of a given quadrilateral mesh into rectangular patches. It can be constructed by connecting irregular vertices through straight chains of edges called *separatrices*. In other words, the base complex corresponds to a simplified version of the input quadrilateral mesh. Providing a high-quality quadrilateral mesh with a coarse base complex is of great interest, since a coarse base complex induces a simple patch layout which is desired for fitting NURBS patches.

2.1.2.2 Parametric Representations

We have introduced the mesh representation. Objects obtained by scanning technologies are often output as meshes. Although this representation is straight-forward, the large amount of sampled points of a scanned physical object makes it very inefficient in terms of memory and disk space for interactive editing. Furthermore, ensuring continuity (smoothness) when modifying the curved regions of the surface means many points have to be re-positioned accurately, which is a tedious task. These two problems can be overcome by representing the object in a parametric representation. Parametric representations of surfaces, and more specifically tensor-product splines, are widely used in CAD. In the context of this thesis, we have chosen NURBS as the target spline representation.

B-Spline Basis functions

A knot vector is a non-decreasing set of real coordinates in the parameter space, written:

$$\Xi = \{\xi_0, \ldots, \xi_i, \ldots, \xi_m\}, \tag{2.5}$$

where the element ξ is the i^{th} knot, and the half-open interval $[\xi_i, \xi_{i+1}[$ is the i^{th} knot span (or element). Knot spans can have zero length, since knots don't need to be distinct. A knot vector is called uniform when its knot spans are identical, otherwise it is called non-uniform.

Let p be the degree of the B-Spline. We will consider only non-periodic (or clamped or open) normalized knot vectors, which have the form:

$$\Xi = \{\underbrace{0, \ldots, 0}_{p+1}, \xi_{p+1}, \ldots, \xi_{m-p-1}, \underbrace{1, \ldots, 1}_{p+1}\}. \tag{2.6}$$

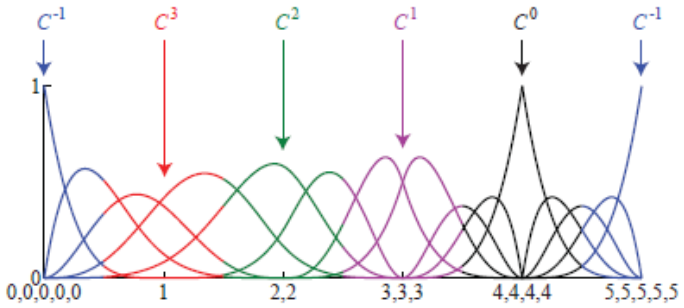


Figure 2.10: Quartic ($p = 4$) basis functions for an open, non-uniform knot vector $\Xi = \{0, 0, 0, 0, 0, 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5\}$. The continuity across an interior element boundary is a direct result of the polynomial order and the multiplicity of the corresponding knot value.

Given a knot vector $\Xi = \{\xi_0, \ldots, \xi_m\}$, the i^{th} B-spline basis function of degree p (order $p+1$), denoted by $N_{i,p}(\xi)$, can be efficiently computed using the Cox-De Boor recursive formula as follows:

$$\begin{aligned} N_{i,0}(\xi) &= \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \\ N_{i,p}(\xi) &= \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi). \end{aligned} \tag{2.7}$$

Once the degree p is fixed, the knot vector Ξ completely determines the B-Spline basis functions $N_{i,p}(\xi)$. Let $m+1$ be the number of knots of the non-periodic knot vector Ξ. Then there are $n+1$ basis functions, where $n = m - p - 1$. The B-Spline basis functions have the following properties (Figure 2.10):

- The $N_{i,p}(\xi)$ are piecewise polynomials, defined on the entire real line. Generally only the interval $[\xi_0, \xi_m]$ is of interest: $N_{i,p}(\xi) = 0$ if ξ is outside the interval $[\xi_i, \xi_{i+p+1}[$.
- The $N_{i,p}(\xi)$ are non-negative, i.e., $N_{i,p}(\xi) \geq 0$ for all i, p and ξ. In addition, in any given knot span $[\xi_j, \xi_{j+1}[$, at most $p+1$ of the $N_{i,p}(\xi)$ are non-zero, namely the functions $N_{j-p,p}(\xi), \ldots, N_{j,p}(\xi)$.
- For an arbitrary knot span $[\xi_i, \xi_{i+1}[$, the $N_{j,p}(\xi)$ for $j = i-p, \ldots, i$ form a partition of unity, i.e., $\sum_{j=i-p}^{i} N_{j,p} = 1$.
- All derivatives of $N_{i,p}(\xi)$ exist in the interior of a knot span. At a given knot, $N_{i,p}(\xi)$ is $p-k$ times continuously differentiable, where k is the multiplicity of the knot. Hence, increasing degree increases continuity, and increasing knot multiplicity decreases continuity.

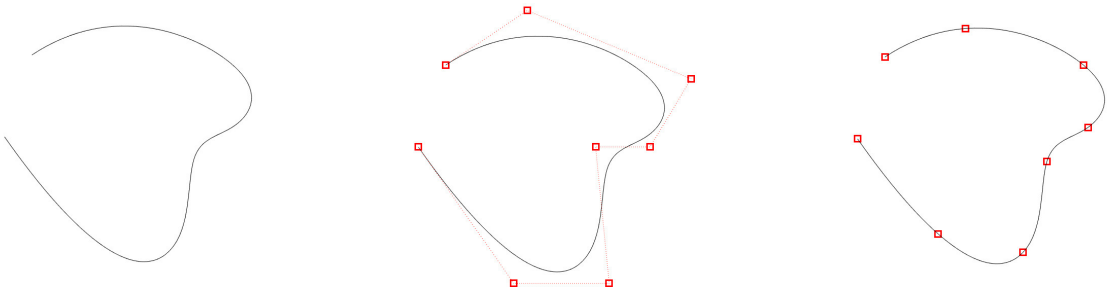
B-Spline Curves

B-spline curves in $\mathbb{R}^d$ are constructed by taking a linear combination of B-spline basis functions. The vector-valued coefficients of the basis functions are referred to as control

points. Let $\Xi = \{\xi_0, \ldots, \xi_m\}$ be a knot vector and $\{\mathbf{P}_i\}$ a set of control points with $\mathbf{P}_i \in \mathbb{R}^d$ for $i = 0, \ldots, n$. The B-Spline curve with degree p, where $p = m - n - 1$, is defined by (Figure 2.11):

$$C(\xi) = \sum_{i=0}^{n} N_{i,p}(\xi)\mathbf{P}_i, \tag{2.8}$$

where $N_{i,p}$ are B-Spline basis functions of order p, corresponding to knot vector Ξ.



(a) (b) (c)

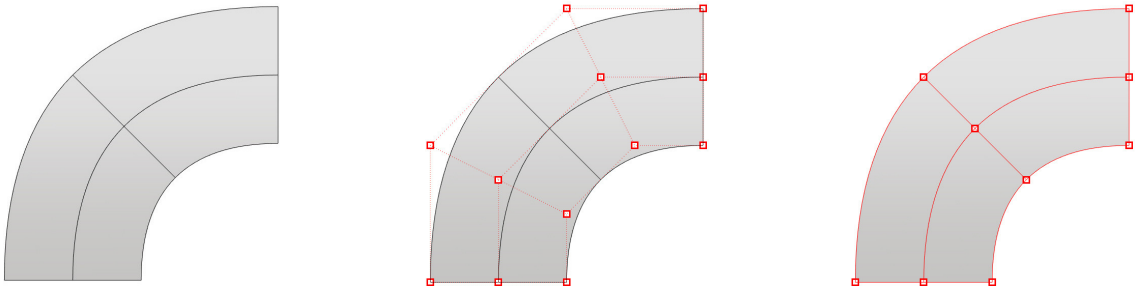
Figure 2.11: B-Spline curve (a), its control points (b) and its knot locations (c).

B-Spline surfaces

Parametric curves can be generalized to bi-parametric surface patches. One can consider the parametric surface as the Cartesian product (tensor product) of two parametric curves. Let $\Xi = \{\xi_0, \ldots, \xi_{m_0}\}$ and $\mathcal{H} = \{\eta_0, \ldots, \eta_{m_1}\}$ be knot vectors, and $\{\mathbf{P}_{i,j}\}$ a set of control points with $\mathbf{P}_{i,j} \in \mathbb{R}^d$ for $i = 0, \ldots, n_0$ and $j = 0, \ldots, n_1$. The B-Spline surface with degree p and q, where $p = m_0 - n_0 - 1$ and $q = m_1 - n_1 - 1$, is defined by (Figure 2.12):

$$S(\xi, \eta) = \sum_{i=0}^{n_0} \sum_{j=1}^{n_1} N_{i,p}(\xi) M_{j,q}(\eta) \mathbf{P}_{i,j}, \tag{2.9}$$

where $N_{i,p}(\xi)$ and $M_{j,q}(\eta)$ are univariate B-spline basis functions of order p and q, corresponding to knot vectors Ξ and $\mathcal{H}$, respectively.



(a) (b) (c)

Figure 2.12: B-Spline surface (a), its control points (b) and its knots locations (c).

B-Spline Volumes

Tensor product B-Spline volumes are defined in analogous fashion to tensor product B-Spline surfaces. Let $\Xi = \{\xi_0, \ldots, \xi_{m_0}\}$, $\mathcal{H} = \{\eta_0, \ldots, \eta_{m_1}\}$ and $\mathcal{Z} = \{\zeta_1, \ldots, \zeta_{m_2}\}$ be knot vectors, and $\{\mathbf{P}_{i,j,k}\}$ a set of control points with $\mathbf{P}_{i,j,k} \in \mathbb{R}^d$ for $i = 0, \ldots, n_0$, $j = 0, \ldots, n_1$ and $k = 0, \ldots, n_2$. The B-Spline volume with degree p, q and r, where $p = m_0 - n_0 - 1$, $q = m_1 - n_1 - 1$ and $r = m_2 - n_2 - 1$, is defined by:

$$V(\xi, \eta, \zeta) = \sum_{i=0}^{n_0} \sum_{j=0}^{n_1} \sum_{k=0}^{n_2} N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta) \mathbf{P}_{i,j,k}, \tag{2.10}$$

where $N_{i,p}(\xi)$, $M_{j,q}(\eta)$ and $L_{k,r}(\zeta)$ are univariate B-spline basis functions of order p, q and r, corresponding to knot vectors Ξ, $\mathcal{H}$ and $\mathcal{Z}$, respectively.

NURBS Curves, Surfaces and Volumes

A NURBS geometry is similar to a non-uniform B-spline geometry except that the control points are defined in homogeneous coordinates rather than Cartesian coordinates. Consider a control point $\mathbf{P} = (x_i, y_i, z_i) \in \mathbb{R}^3$, its coordinates in homogeneous space can be simply written as $(x_i \cdot w_i, y_i \cdot w_i, z_i \cdot w_i, w_i)$, where w_i is the weight of the control point. Given a knot vector $\Xi = \{\xi_0, \ldots, \xi_m\}$, the i^{th} NURBS basis function $R_{i,p}$ of degree p is expressed as:

$$R_{i,p}(\xi) = \frac{w_i N_{i,p}(\xi)}{\sum_{j=0}^{n} w_j N_{j,p}(\xi)}, \tag{2.11}$$

where $N_{i,p}(\xi)$ is the i^{th} B-spline basis functions of order p, and $n = m - p - 1$. NURBS curves, surfaces and volumes can be expressed exactly as their corresponding B-Spline entities by simply replacing the B-Spline basis functions with NURBS basis functions.

2.1.3 Parameterization Techniques

Model parameterization is the fundamental basis and powerful geometry processing tool with versatile applications such as meshing processing and spline fitting. Surface parameterization can be viewed as a mapping from a surface M embedded in $\mathbb{R}^2$ to canonical domain D embedded in $\mathbb{R}^2$. For topologies of M other than the disk, the domain D must necessarily include discontinuities (e.g. cuts or seams). The ideal parameterization is isometric, i.e., it fully preserves areas and angles. For surfaces, an isometric parameterization is not possible in the general case. Therefore many approaches to surface Euclidean parameterization attempt to find a mapping which is either conformal (i.e., no angular distortion), or equiareal (i.e., no area distortion). A key fact about conformal maps is that they always exist, as guaranteed by the uniformization theorem.

Surface uniformization means that all metric surfaces can be conformally mapped to one of the three canonical domains: the sphere, the plane, and the hyperbolic space. Ricci flow is a parabolic system of partial differential equations which acts like the heat equation to spread the curvature of a Riemannian metric evenly over the surface to produce a metric of constant curvature. Computational discrete Ricci flow is the practical method to compute surface uniformization [Jin08].

In this chapter, only the parameterization of disk-like surfaces and multiply connected genus zero surfaces will be considered:

- For the parameterization of a disk-like triangulated surface M, we use the discrete harmonic mapping. We construct a harmonic function $f : M \to \mathbb{R}$ such that $\Delta f = 0$. The surface boundary ∂M is first mapped to the boundary of the parameter domain and then the parameterization for the interior vertices is obtained by solving a linear system:

$$\sum_{j \in N_i} w_{ij} \left[f(v_j) - f(v_i) \right] = 0, \tag{2.12}$$

  where $v_i, v_j \in S$, N_i is the set of indices of vertices adjacent to v_i, and w_{ij} is a scalar weight assigned to the directed edge $e_{ij}(v_i, v_j)$. Different parameterization methods assign different weights w_{ij} for each edge. Here, we choose the mean value coordinates weights introduced by Floater [Flo03]:

$$w_{ij} = \frac{\lambda_{ij}}{\sum_{k \in N_i} \lambda_{ik}} \text{ and } \lambda_{ij} = \frac{\tan(\theta_{ij}/2) + \tan(\phi_{ij}/2)}{\|v_j - v_i\|}, \tag{2.13}$$

  where θ_{ij} and ϕ_{ij} are the angles the edge $e_{ij}(v_i, v_j)$ makes with its two immediate neighboring edges at v_i.

- For the parameterization of multiply connected genus zero surfaces, we use the Generalized Koebe's Method presented by Zeng et al. [Zen09]. According to Koebe's uniformization theory, all genus zero multiply connected surfaces can be mapped to a planar disk with multiply circular holes. Furthermore, this kind of mappings are angle preserving and differ by Möbius transformations.

2.2 Part-Aware Partitioning

Due to its tensor-product nature, B-spline fitting demands that the parametric domain (may be composed of a set of sub-domains) keeps regular. In addition, the construction of volumetric splines requires parametric domains in $\mathbb{R}^3$. If the parametric domain is composed by a set of sub-domains, consistency between different local domains is required. If all these requirements are satisfied, the surface mesh of the model can be mapped to cover the boundaries of all local parametric domains seamlessly and consistently. To satisfy these requirements, the parametric domain (or the set of sub-domains) must have the same topology of the model but simplified geometrical features. For genus-0 models, the most simple way is to map them to a sphere without considering their geometrical features. However for models with more complex geometry and arbitrary topology, more complex domains are required.

A commonly used part-aware component is cylinder-like domains [Mar09]. Martin et al. [Mar10] have extended this domain to mimic more complex shapes. However, in terms of spline construction, the cylinder-like domain produces inevitable degenerate points along the tube axis. To avoid such problem, a more popular shape abstraction method is to use polycube domains. A polycube is a domain composed by gluing a set of cubes together. Each patch of the boundary mesh of the input model maps to one of six faces of one

cube. The advantage of this mapping method is that each mapping patch is tensor-product regular and the global mapping is seamless between different adjacent patches. The parameters between neighboring patches can transform consistently to each other simply by linear parameter transformation or rotation. Polycubes allow the decomposition of an object into a set of larger hexahedral pieces. However, the quality of the resulting hexahedral representation strongly depends on the placement of polycube corners on the input triangle mesh.

Tarini et al. [Tar04] pioneered the concept of polycube maps for seamless texture mapping with low angle and area distortion. The domain construction and mapping are computed through simple projection. The map between the given 3D shape and the polycube requires the projection of the vertices from the 3D shape to the polycube. This extrinsic method may not produce a valid one-to-one map if the polycube differs from the modeled shape significantly. Wang et al. [Wan07] presented an intrinsic method to construct the polycube map which avoids the projection of the vertices on a 3D model to the polycube domain. Their approach first maps the model and the polycube to a common canonical domain to guarantee bijectivity. The map between the given 3D shape and polycube requires computing a global surface parameterization. Based on the uniformization theorem, this global parameterization maps models, depending on their Euler characteristic χ, to either the sphere $\mathbb{S}^2$ ($\chi > 0$), the Euclidean plane $\mathbb{E}^2$ ($\chi = 0$) or the hyperbolic disk $\mathbb{H}^2$ ($\chi < 0$). This intrinsic method, though theoretically sound to guarantee a bijection, may not be practically useful for a topologically complicated surface. It is known that embedding models with negative Euler characteristic is error-prone when points are very close to the boundary of the hyperbolic disk due to the numerical rounding error. Hence, this method is not practical and much less numerical stable to construct polycube maps of large-scale models with negative Euler Characteristic.

Several methods have been developed to improve user control. Wang et al. [Wan08] presented a technique where the user can interactively control the desired locations and the number of singularities of the polycube map (i.e. the corners in polycubes) which facilitates the manifold spline construction. Xia et al. [Xia11] allowed users to sketch curve constraints to control the polycube map.

Automatic methods are usually difficult to control. Lin et al. [Lin08] used the Reeb graph to segment the surface and then developed an automatic method to construct polycube map. However, their segmentation method may not work for shapes with complicated topology and geometry and does not guarantee bijection between the polycube and the input model. He et al. [He09] proposed a divide-and-conquer algorithm by slicing the model along an axis direction. Slicing along an axis produces very complex domain structure so Gregson et al. [Gre11] proposed a deformation-based method which is less prone to over-segmentation. These methods work best for axis-aligned geometric models without any twist bend, and spiral. Livesu et al. [Liv13] and Huang et al. [Hua14] introduced Polycuts and L1-Polycubes to improve the corner configuration on the conventional polycube.

Recently, Li et al. [Li12] extended the conventional polycube to a Generalized PolyCube (GPC), which enables the curved cuboid representation of the elementary subvolumes decomposed via shape analysis. This enables the polycube map approach to be applied to more complex objects. The drawback to their method is that polycube corners had to be selected manually by the user.

2.2.1 Pants Decomposition

(a) (b) (c)

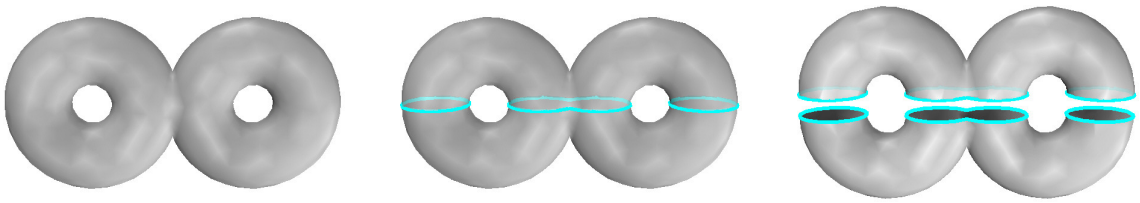
Figure 2.13: Different geometric representations of a pants patch: a disk with 2 boundaries (a); a T-shaped geometry with 3 boundaries (b); a sphere with 3 boundaries (c).

2.2.1.1 Definition

Pants decomposition has been studies by Hatcher et al. [Hat00] and work has been done to investigate the optimal segmentation of a given surface into pants patches by Verdière et al. [Ver07]. Pants decomposition provides an elegant topological tool to study the consistent segmentation of high-genus surfaces systematically.

Let $M_{g,b}$ be a surface of genus g with b boundary components. A *pants patch* is a genus-0 surface (topological sphere) with 3 boundary components (Figure 2.13). A *pants decomposition* of $M_{g,b}$ is a collection of pairwise disjoint simple cycles that splits the surface into a set of pants patches (Figure 2.14).

We assume that M is a surface with negative Euler characteristic, i.e., M is none of the surfaces $M_{0,0}$ (topological sphere), $M_{0,1}$ (topological disk), $M_{0,2}$ (topological cylinder), and $M_{1,0}$ (topological torus). In this case pants decomposition of M do exist, and each pants decomposition consists of $3g + b - 3$ curves and divides M into $2g + b - 2$ pants patches. $\chi = -1$ for a pants patch and therefore pants decomposition provides a canonical decomposition scheme for these surfaces.



(a) (b) (c)

Figure 2.14: A pants decomposition of a surface (a) is a collection of pairwise disjoint simple cycles (b) that splits the surface into a set of pants patches (c).

2.2.1.2 Algorithm

A consistent pants decomposition algorithm was presented by Li et al. [Li08]. The homology basis of a genus-g surface M consists of $2g$ cycles. We will use the homology

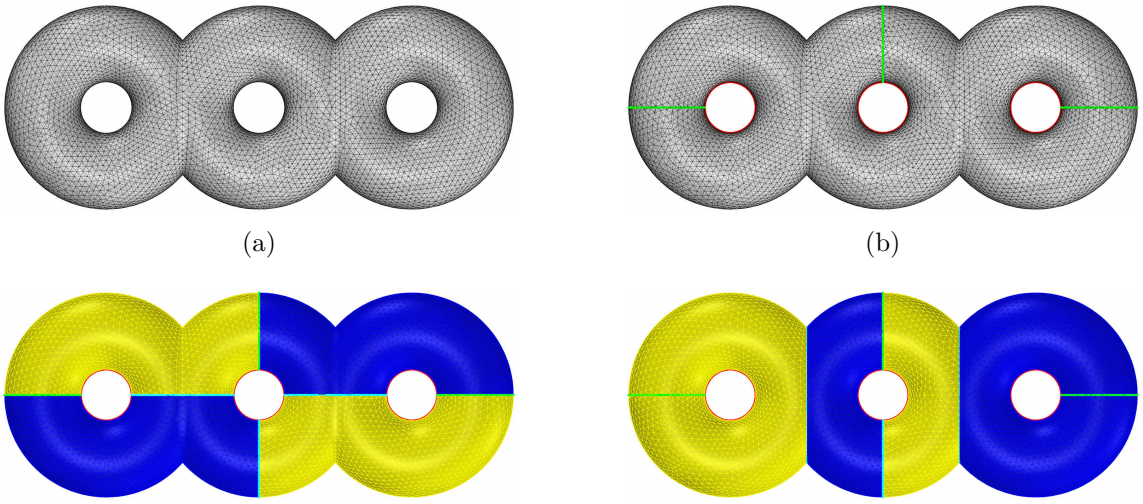


Figure 2.15: Pants decomposition of a 3-torus. Triangulated boundary surface (a) along with its handle and tunnel loops in green and red, respectively (b). Pants decomposition using loops with shortest distance (c), and loops passing through areas of minimum curvature (d).

basis formed by the handle and tunnel loops. From this homology basis, we can pick a subset H composed of g simple and pairwise disjoint handle loops $\{h_1, \ldots, h_g\}$. Slicing a genus-g surface M with b boundary components along its g handle loops will lead to a genus-0 surface $\bar{M}$ with $2g+b$ boundary components. We denote these $2g+b$ boundaries as $W = w_1, \ldots, w_{2g+b}$. We iteratively pick two boundaries w_i and w_j from W and compute a new simple cycle w_{ij} to bound them, i.e., w_{ij} is homotopic to $w_i \circ w_j$. The three cycles w_i, w_j and w_{ij} bound a pants patch T_k. We remove this pants patch T_k from $\bar{M}$. The patch left is still genus-0 but its boundary number reduces by 1: the two cycles w_i and w_j are removed, and one new cycle w_{ij} is inserted. This is iteratively performed until $|W| = 3$. This idea is formulated in Algorithm 1, and the operation that traces a cycle w_{ij} homotopic to cycle $w_i \circ w_j$ is formulated in Algorithm 2. Figure 2.16 gives an illustration for the complete algorithm.

In Algorithm 2 Step 8, the "metric" can also be adapted to favor loops passing through areas of symmetry or minimum curvature. For instance, in Figure 2.15c, pants decomposition was performed using loops with shortest distance, whereas in Figure 2.15d, pants decomposition was performed using loops passing through areas of minimum curvature. All the tracing of cutting cycles are from Dijkstra's algorithm [Dij59] conducted on the weighted triangle mesh. Therefore, we can integrate different geometric criteria into the weight of each triangle edge [Zha14]. The favored edges will have a smaller weights so that traced cycles will more likely go through them. Different geometric criteria can be used to guide the pants decomposition:

- Shortest length: shortest cutting cycles discretely approximate the geodesics and are simply desirable in most scenarios. For an edge $e = (v_i, v_j)$, its shortest-length weight for the Dijkstra tracing is defined as the Euclidean distance $\omega_l(e) = |v_j - v_i|^2$ between

Algorithm 1 Pants Decomposition Algorithm.

Input: Triangulated genus-g surface M with b boundary components and its g handle loops.

Output: Set of $2g+b-2$ pants patches $T = \{T_1, \ldots, T_{2g+b-2}\}$, where $M = \bigcup T_i$.

1: $k \leftarrow 1$
2: Slice M along all its handle loops and get a genus-0 surface $\bar{M}_k$ with $2g+b$ boundaries.
3: Put all boundaries of $\bar{M}_k$ in a set $W = \{w_1, \ldots, w_{2g+b}\}$.
4: **while** $|W| > 3$ **do**
5: Select two boundaries w_i and w_j from W and compute a loop w_{ij} homotopic to $w_i \circ w_j$.
6: $\{w_i, w_j, w_{ij}\}$ bound a pants patch T_k. Remove T_k from $\bar{M}_k$: $\bar{M}_k \leftarrow \bar{M}_k \setminus T_k$.
7: Remove w_i and w_j from W, and add w_{ij} into W.
8: $k \leftarrow k+1$
9: **end while**

Algorithm 2 Homotopic cycle computation.

Input: Triangulated genus-0 surface M with b boundary components $\{w_1, \ldots, w_b\}$.

Output: A cycle w_{ij} homotopic to cycle $w_i \circ w_j$.

1: Compute shortest path connecting w_i to w_j.
2: Slice M along this path to get one new large boundary c_{ij}.
3: Connect all other boundaries together using shortest paths.
4: Slice M along these paths to get one new large boundary c_k. M becomes a topological cylinder.
5: Compute the shortest path γ connecting the cylinder's two boundaries c_{ij} and c_k.
6: Slice M along the path γ: every point $p_i \in \gamma$ splits into a pair $(p_i, \bar{p}_i)$.
7: Trace all paths connecting points pairs $(p_i, \bar{p}_i)$.
8: Among these paths, w_{ij} is the one that has the minimal length.

its vertices.

- Symmetry: many models have intrinsic symmetric patterns. One may prefer to cut the surface along its symmetry plane. We can define a scalar value $d(v)$ on each vertex v using its Euclidean distance to the symmetry plane. For an edge $e = (v_i, v_j)$, its symmetry weight for the Dijkstra tracing is defined as $\omega_s(e) = 0.5\,[d(v_i) + d(v_j)]$.

- Minima rule: human perception often cuts the surfaces along concave regions, which is know as the minima rule [Lee05]. We define a value on each vertex by the minimum curvature value, which is calculated by the tensor field computation [All03]. Since the range of minimum curvature values are too diverse, we normalize the values. If $\kappa(v)$ is the minimum curvature value at a vertex v, the normalized value is $r(v) = (\kappa(v) - \mu)/\sigma$, where μ is the mean and σ is the standard deviation of $\kappa(v)$ over all vertices of the mesh. For an edge $e = (v_i, v_j)$, its minima-rule weight for the Dijkstra tracing is defined as $\omega_m(e) = 0.5\,[r(v_i) + r(v_j)]$.

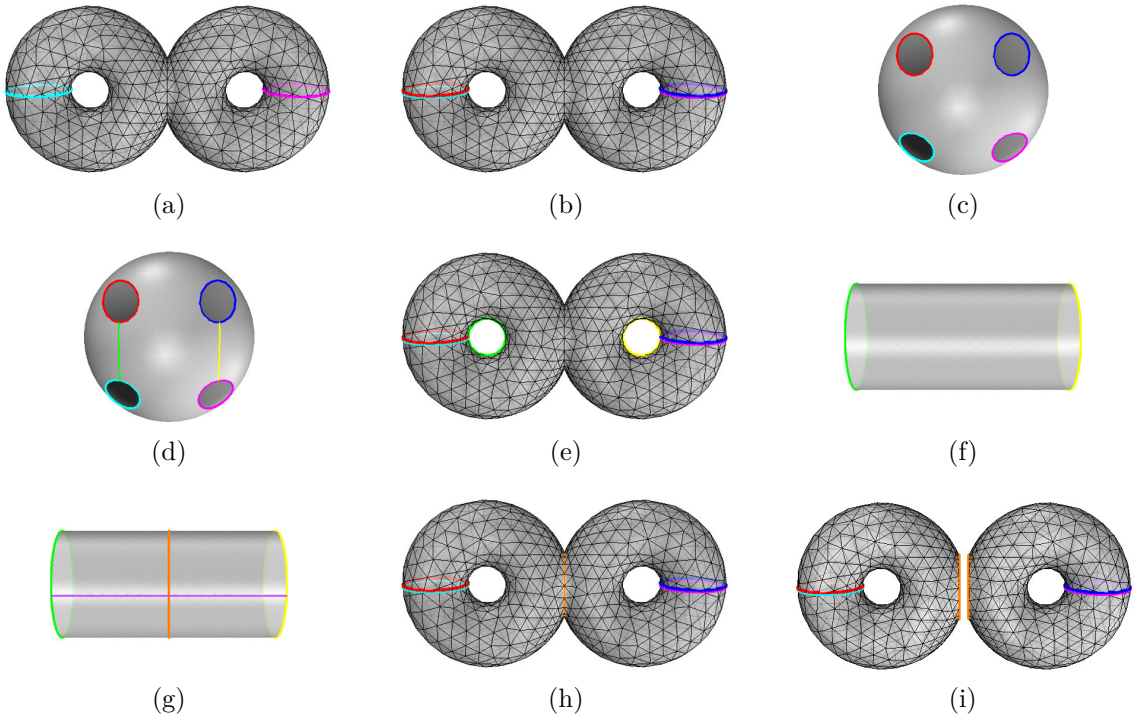
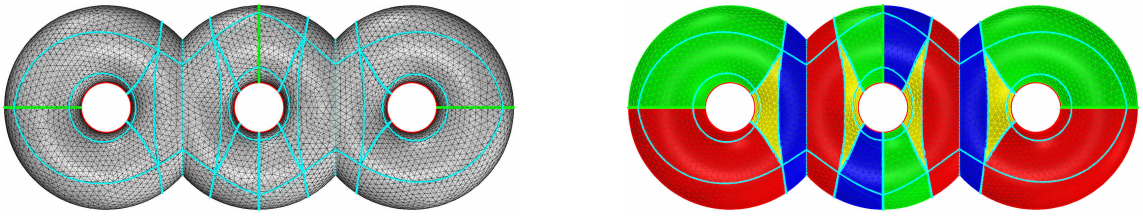


Figure 2.16: Pants decomposition algorithm overview. Triangulated boundary genus-2 surface along with its 2 handle loops (a). The surface is sliced along these loops (b). The surface become a topological sphere with 4 boundaries (c). 2 boundaries are selected and the shortest path between them is computed. The shortest path connecting the remaining 2 boundaries is also computed (d). The surface is sliced along these two shortest paths (e) and become a topological cylinder (f). The shortest path γ connecting the cylinder's two boundaries is computed. The cylinder is then sliced along the path γ and every point $p \in \gamma$ is split into a pair of points. Between all paths connecting all pair of points, we choose the path ω which has the minimal length (g-h). Finally, the surface is sliced along the path ω to get two pants patches (i).

2.2.2 Cuboid Decomposition

After decomposing the boundary surface into a set of pants patches, we decompose each pants patch T_i into a set of 4 cuboids $\{C_{ij}\}$. The idea is to generate corners and polyedges on each pants patch and decompose it into 4 connected components, each having 8 corners and 12 polyedges like a cuboid (illustrated in Figure 2.17).

As input, we have a set of pants patches. The 3 boundaries of a given pants patch will be arbitrary denoted by B_1, B_2 and B_3. We process these pants patches one by one in an arbitrary order. To guarantee corner alignment, when we determine one pants patch's result, we transfer its corners on the boundaries of the adjacent pants patches if they are not processed yet. The proposed algorithm is very robust even for low-quality and noisy meshes (Figure 2.23).



(a) (b)

Figure 2.17: Cuboid decomposition of a 3-torus: the polyedge structure (a) and the cuboid organization (b).

2.2.2.1 Cutting Curves

Step 1 We generate 3 cutting curves W_1, W_2, and W_3 (illustrated in Figure 2.18):

a) We compute 3 discrete harmonic functions: f_1, f_2 and f_3. To compute f_i, we set $f_i = 0$ for vertices on the boundary B_i and $f_i = 1$ for vertices on the remaining two boundaries B_j and B_k. Then we solve $\Delta f_i = 0$ using mean value coordinates (Figure 2.18a).

b) Each harmonic function f_i has one minimum component and two maximum components. Hence, the function f_i must have one saddle point. We denote the saddle points of the functions f_1, f_2 and f_3 by s_1, s_2 and s_3, respectively (Figure 2.18b).

c) Let $\bar{f}_i = \min\{f_i(s_i), f_i(s_j), f_i(s_k)\}$. Then the cutting curve W_i is defined as the isoparametric curve of the function f_i for the value $\bar{f}_i$ (Figure 2.18c).

2.2.2.2 Boundary Corners

Step 2 We generate all corners on the boundaries B_1, B_2 and B_3 (illustrated in Figure 2.19):

a) We conformally map the pants patch to a planar disk with 2 circular holes in the parametric domain, using circular conformal mapping [Zen09]. The boundary B_k is mapped to the outer circle of the planar disk and the boundaries B_i and B_j are mapped to the inner circular holes (Figures 2.19a and 2.19b).

b) In the parametric domain, by intersecting the line passing through the centers of the inner circles with the 3 boundaries, we get 6 intersection points (Figure 2.19c). By projecting these points back to the pants patch, we obtain 6 points s_{ij}, s_{ik}, s_{ji}, s_{jk}, s_{ki} and s_{kj} (Figure 2.19d). Throughout, these points will be called "seed points".

c) Each boundary B_i has 2 seed points s_{ij} and s_{ik}. Let $\gamma_i : [0, 1] \to \mathbb{R}^3$ be an arc-length parameterization of the boundary B_i with $\gamma_i(0) = \gamma_i(1) = s_{ij}$. We define the points $\gamma_i(0.125)$, $\gamma_i(0.375)$, $\gamma_i(0.625)$ and $\gamma_i(0.875)$ as the corners of the boundary B_i.

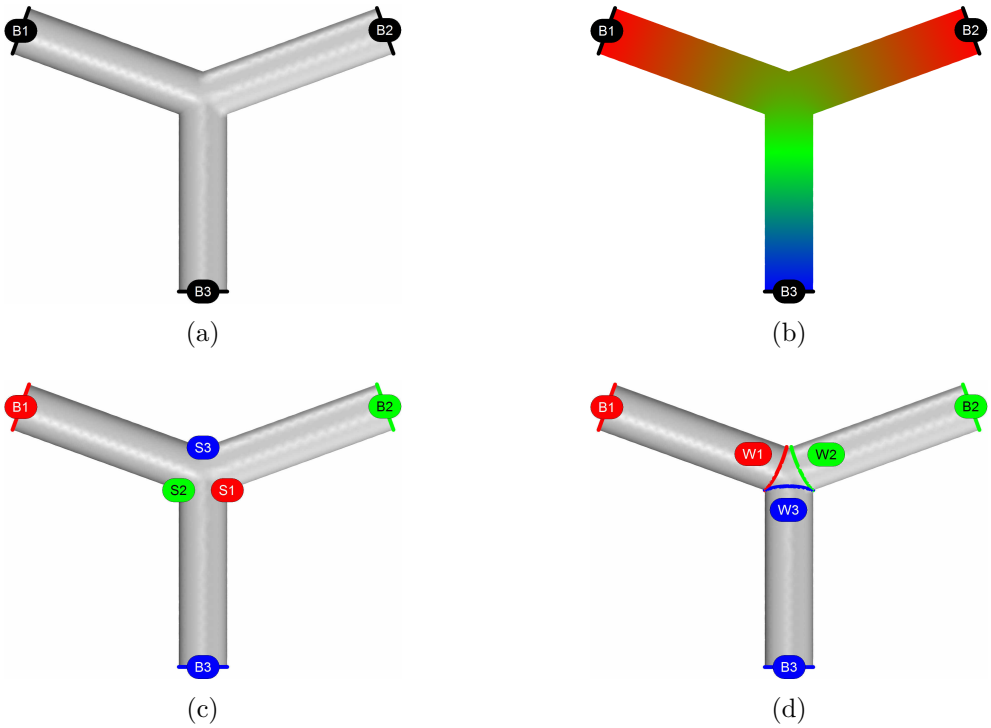


Figure 2.18: Generation of cutting curves. A pants patch (a) and the discrete harmonic function f_1 (b). Saddle points of the three discrete harmonic functions f_1, f_2 and f_3 (c). The three generated cutting curves using saddle points (d).

At the end of this step, each boundary of the pants patch has 4 corners. Throughout, we will denote by $\{P_1, P_2, P_3, P_4\}$ the corners on B_1, $\{P_5, P_6, P_7, P_8\}$ the corners on B_2, and $\{P_9, P_{10}, P_{11}, P_{12}\}$ the corners on B_3 (Figure 2.20a). We also pair the corners on different boundaries as follows: $\{P_1, P_9\}$, $\{P_2, P_{10}\}$, $\{P_3, P_7\}$, $\{P_4, P_8\}$, $\{P_5, P_{11}\}$, and $\{P_6, P_{12}\}$ (Figure 2.20b).

2.2.2.3 Boundary Polyedges

Step 3 We trace the 6 polyedges between each pair of corners in 3 passes. At each pass, we trace 2 polyedges (illustrated in Figure 2.21):

a) We remove a long branch by cutting along its cutting line W_k. After filling the cutting hole, the resulting patch is a topological cylinder with 2 boundaries B_i and B_j. We denote this patch by P_k (Figure 2.21a).

b) We map the topological cylinder P_k to a cylindrical domain $[u, v]$ following the approach of [Mar09]. We first set $u = 0$ for vertices on B_i and $u = 1$ for vertices on B_j, then solve $\Delta u = 0$. We trace an iso-v curve along ∇u from the seed vertex s_{ik} on the boundary B_i to the boundary B_j. We slice P_k along this iso-curve and get two duplicated boundary paths. We finally set $v = 0$ and $v = 1$ on them respectively and solve $\Delta v = 0$ (Figure 2.21b).

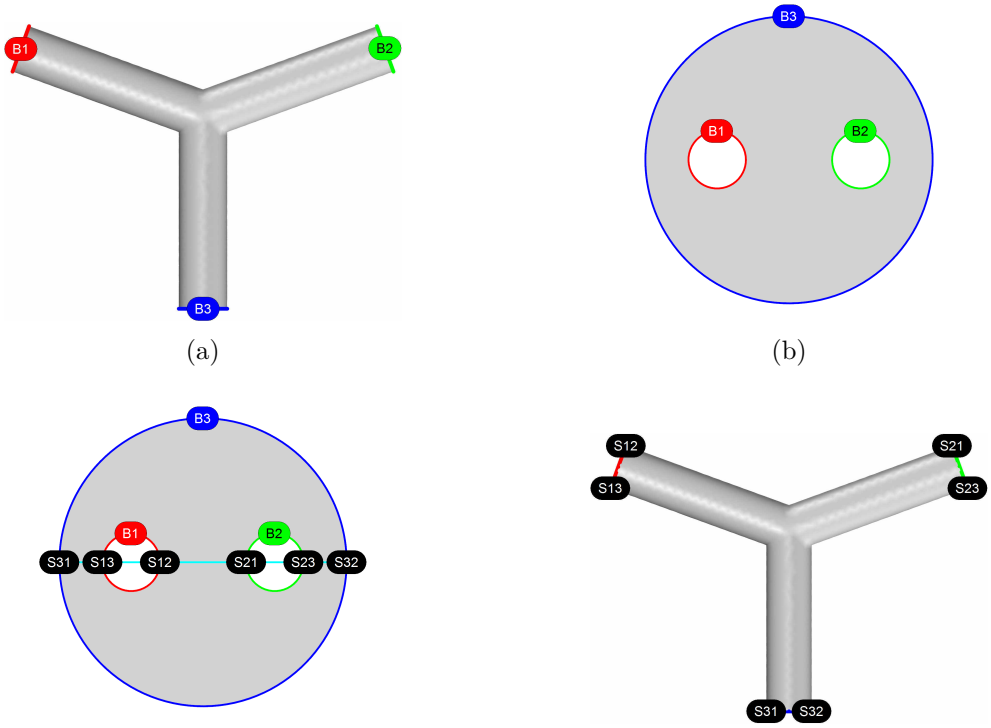


Figure 2.19: Generation of boundary seed points. A pants patch (a) and its cicular conformal map (b). Seed points locations in the parametric space (c) and in the physical space (d).

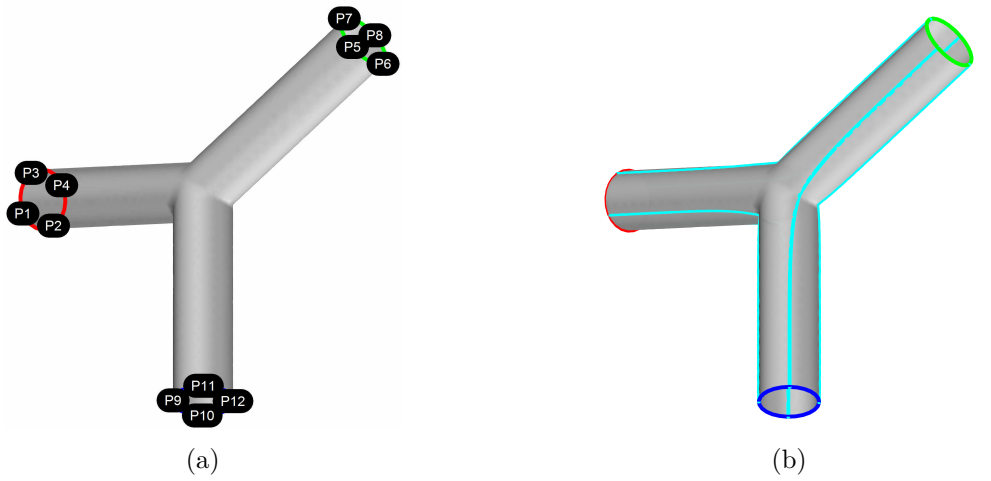


Figure 2.20: Corner points on the pants patch's boundaries (a) and the polyedges connecting each pair of corner points (b).

c) On B_i and B_j, we select the 2 pairs of corners that are the furthest from the seed points s_{ik} and s_{jk}. We map all 4 corners to the cylindrical domain. For each pair

of corners, we trace the straight line between them on the cylindrical domain (Figure 2.21c), then project this parametric straight line back to the patch P_k and get the resulting polyedge (Figure 2.21d).

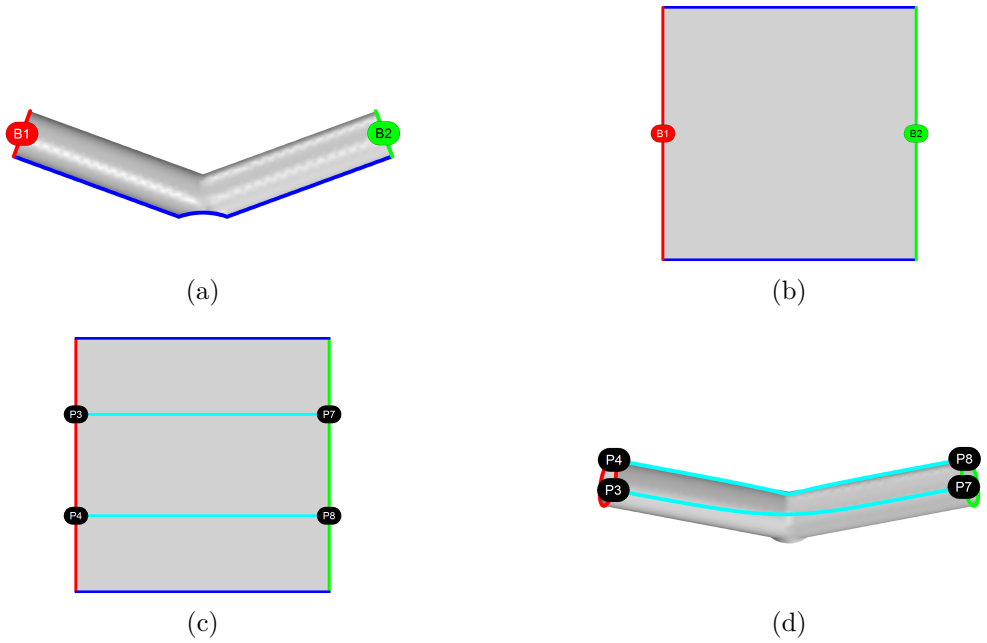


Figure 2.21: Generation of polyedges between corner pairs. The topological cylinder after removing a long branch (a) and its cylindrical parametric domain (b). Polyedges between pairs of corners in the parametric space (c) and in the physical space (d).

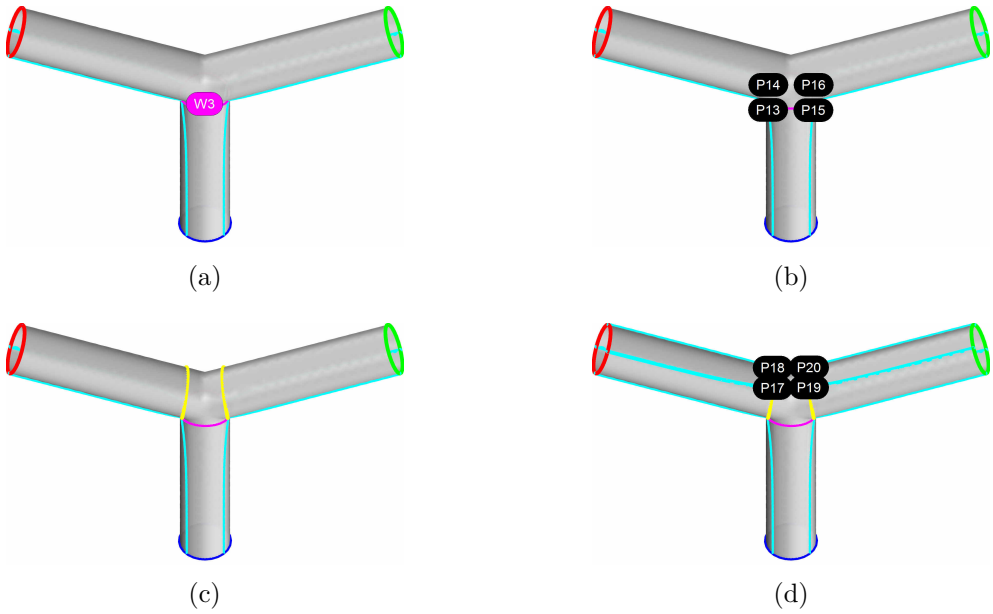
2.2.2.4 Central Cuboid

Step 4 We now generate the remaining corners and polyedges of the central cuboid (illustrated in Figure 2.22):

a) The 4 corners $\{P_{13}, P_{14}, P_{15}, P_{16}\}$ are the intersection between the cutting curve W_3 (Figure 2.22a) and the polyedges generated in the previous step using subpatches P_1 and P_2 (Figure 2.22b).

b) Using the cylindrical parameterization of subpatch P_3, we trace polyedges between the corner pairs $\{P_{13}, P_{14}\}$ and $\{P_{15}, P_{16}\}$ following the method presented in the previous step (Figure 2.22c).

c) The 4 corners $\{P_{13}, P_{14}, P_{15}, P_{16}\}$ are the intersection between the newly traced polyedges and the polyedges generated in the previous step using subpatch P_3 (Figure 2.22d).



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

Figure 2.22: Generation of corners and polyedges of the central cuboid.

Conclusion

The objective is to generate a trivariate parameterization with respects to a given solid model defined by its boundary mesh or boundary (possibly trimmed) spline surfaces. In order to have a unified framework for both input types, the input is a solid model defined by its triangulated boundary. The first step toward this objective is the generation of an initial polycube approximating the input boundary mesh. The polycube approximates very roughly the geometry of the model while faithfully replicating its topology. Due to its regular structure, the polycube is suitable for serving as the parametric domain of the tensor-product spline representation required for IGA.

The generalized polycube domain is obtained after two steps of domain decomposition based on topology and geometry. The model is split into a set of cuboids based on solving harmonic functions on the surface. The boundary triangle mesh is first decomposed into a set of pants patches. Such segmentation decomposes a complicated surface into a set of shapes that have a trivial topology: a pants patch is a genus-0 surface with 3 boundaries. Each pants patch is then decomposed into a set of cuboids: a cuboid is a boxed region enclosed by 6 disk-like surfaces. The novelty of the proposed method is the "geometry-aware" pants-to-cuboids decomposition algorithm. The algorithm is completely automatic and very robust even for low-quality and noisy meshes. In can be also integrated to existing volume parameterization techniques (e.g. [Li12], [Wan13]) in order to render them completely automatic.

The second main step toward generating a trivariate parameterization is presented in the next chapter (Chapter 3). It consists in computing an aligned global parameterization between the cuboids' boundaries and the boundary of the generated generalized polycube

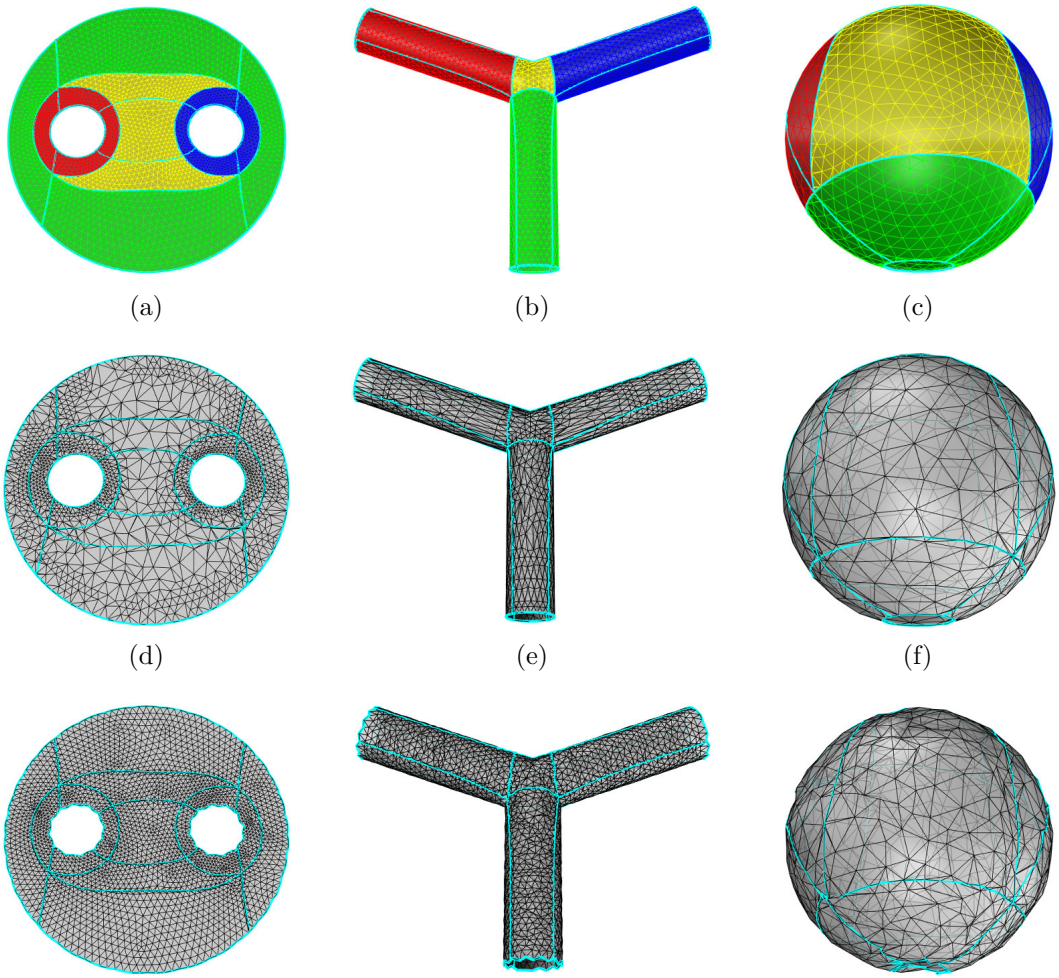


Figure 2.23: Cuboid decomposition of different geometries of a pants patch (a-c). The algorithm a very robust even for low quality (d-f) and noisy (g-i) meshes.

domain. The parameterization must be topologically conform with the polycube structure, and geometrically must follow the geometric features of the model.

Chapter 3

Model Parameterization

Introduction

Surface manipulation and representation is becoming increasingly important, with applications ranging from special effects for films and video-games to digital fabrication and architectural design. Despite significant research efforts, there is still a large technological gap between the acquisition of models and the tools used to process, edit, and render them. Acquired objects do not possess a high level structure; they often start as a point cloud of surface points from which a triangle mesh is derived. Conversely, the majority of models used in CAD applications are smooth surfaces with a shape that is controlled by a coarse grid of quadrilaterals. Depending on the application, the quadrilaterals can be either subdivided or used as control points for defining high-order polynomial surfaces.

There is a fundamental difference between the mesh and the target spline representation. A mesh is an enumerated sampled representation of the geometry, whereas a spline surface requires more structure. The gap between both representations is filled by constructing an abstract representation of the object, i.e., a parameterization. In order to produce a "geometry-meaningful" parameterization that fulfills the anisotropy, the iso-value lines of the parameterization are guided by an anisotropy-adapted direction fields. Such parameterizations are called aligned global parameterization. Since the parameterization is adapted to the geometry, the iso-value lines define a natural quadrilateral control mesh of the surface, optimum from an approximation theory point of view. Parameterization-based quadrilateral remeshing techniques are introduced in Section 3.1.

The anisotropy-adapted direction field is represented by a cross field, i.e., a set of four orthogonal directions at each point of the surface. A smooth cross field, topologically conform with the polycube, is designed on the surface. This means that the cross field is singular only at the position of irregular nodes of the polycube. Within this topologically fixed space of cross fields, a smooth cross field interpolating principal directions and geometric features of the surface is computed. The design of such cross field is presented in Section 3.2.

The global parameterization is then found such that its gradient field matches the cross field as much as possible. The approach of using the polycube's structure to constrain the cross field (and hence the parameterization) is inherently volumetric in the sense that

the volume parameterization can be trivially computed just by interpolating the surface parameterization of each cuboid. Constraining the parameterization by the polycube structure may lead to large distortions or even local non-injectivities due to fold-overs. This problem is solved by re-positioning the polycube's nodes based on the gradient of the aligned global parameterization's objective functional with respect to their positions, so as to arrive at a local optimum of global embedding quality. The computation and optimization of such parameterization are presented in Section 3.3.

The novelty of the proposed method is the design of a cross field topologically conforming the polycube structure and geometrically interpolating the geometrical features of the model by only solving two linear systems. This is done by combining two different discretization approaches of direction fields on surfaces. Previous methods (e.g. [Ray08]) relies on computing such field using mixed-integer optimization formulations.

3.1 Quadrilateral Remeshing

The conversion of a triangle mesh into a quadrilateral mesh is a difficult problem that has been deeply studied during the last decade. Many different approaches have been proposed and transferred from academia to commercial modeling and CAD softwares. However, there are still many open problems to solve to provide a fully automatic pipeline that converts an unstructured model into a high-level representation that can be directly used in a conventional modeling pipeline.

The creation of triangle meshes has been extensively studied in the past few decades, and many robust algorithms are currently available in production-quality software libraries. Its main limitation is that it does not contain any global structure, and it is thus difficult to edit or use as a control grid for higher-order surfaces. This is why quadrilateral meshes are ubiquitous in CAD and CAE applications: they naturally represent a pair of directions and have good numerical properties for discretizing PDEs.

3.1.1 Quality Requirements

The quality of a triangular mesh is usually determined by the shape and size of the triangles; equilateral and uniformly sized triangles are often preferred for rendering smooth surfaces. High-quality triangular meshes also exhibit better numerical properties when used for solving partial differential equations. For more sophisticated geometric modeling and processing applications (like CAD and CAE), quadrilateral meshes are often preferred over triangle meshes. However, the generation and handling of quad meshes is significantly more difficult due to the anisotropic nature of quadrilaterals. While for high quality triangle meshes it is usually sufficient to have a fairly regular vertex distribution, good quadrilateral meshes have additional orientation and consistency constraints to satisfy.

A quadrilateral mesh is usually used as a control grid for a subdivision or NURBS surface. Similar quality metrics could be defined for quadrilateral meshes: the quadrilaterals elements should be as uniform and as orthogonal as possible. In addition, the quality of the final surface is mainly affected by the distribution of singularities which are vertices touched by more or less than four edges. These vertices are particularly important because they are the only ones where the quadrilateral mesh is not a regular grid.

A singular vertex can have a different singularity index, depending on the number of edges touching it. The total sum of singularity indices is a topological invariant that depends on the genus of the surface [Ray08]. Thus, the quality of a quadrilateral mesh depends of the number and the geometric location of the singularities. Hence, the optimization of quadrilateral meshes is an inherently global problem since local changes in the mesh structure usually propagate globally across the mesh. This is not the case for triangle meshes where mesh optimization can be performed based on local operations.

3.1.2 Parameterization-Based Techniques

Quadrilateral-remeshing techniques have a long tradition within the graphics community and nice surveys exist [All05; Bom13b]. Early works tried to generate oriented quadrilateral elements by explicitly tracing lines along the principal curvature directions [All03; Mar04], resulting in quad-dominant meshes. Most recent parameterization based techniques are very successful in generating curvature oriented all-quadrilateral meshes. They are able to automatically find adequate singularity positions, e.g. by non-linearly smoothing the cross field induced by the principal curvature directions [Käl07], optimizing a non-linear objective function [Ray06; Hua08; Zha10], or solving a mixed-integer problem [Bom09]. Typically these methods can generate quadrilateral meshes with a nice angle and edge-length distribution as well as adequate singularities. However, the quality of the induced quad mesh is often not sufficient. The major reason for this imperfection is that mesh singularities are usually placed based on geometric considerations but otherwise fairly independently from each other. An important consequence of this is that geodesically neighboring singularities are not properly connected to form a nicely shaped patch layout. In practice such a clean patch layout is highly desirable to support standard operations like texturing or NURBS fitting.

Instead of using the principal curvature directions as a guiding, another class of algorithms directly exploits a patch layout with specified topology to generate quadrilateral meshes via a global parameterization. The patch layout is constructed manually [Ton06; Bom08] or derived automatically from a Morse-Smale complex [Don06]. If all patches of the layout are topologically equivalent to quadrilaterals then we talk about quadrilateral patch layout (or for short quad layout). A quad layout is an embedded graph which partitions the surface into a set of non-overlapping quadrilateral patches. In practice, the quad layout of a quadrilateral mesh is desired to be coarse and simple while still appropriately respecting the underlying geometry. In mesh processing, the quality of these patches is of high interest to support standard operations like high-order surface fitting (e.g. NURBS). Automatic generation of pure quadrilateral patch layouts on manifold meshes has received a lot of interest in recent years. Powerful automatic algorithms have been developed [Cam12; Bom13a; Raz15]. In the context of this thesis, we will use our method based on cuboid decomposition to automatically generate the initial quad layout. The advantage of our method is that adjacent quad layout's patches form cuboids which adds an extra dimension to the standard quad layout, and allows the construction of trivariate splines.

The polycube consists of nodes and arcs embedded in the surface. Locally, neighboring nodes and arcs partition the surface into quadrilateral patches. Globally, neighboring quadrilateral patches form hexahedral domains (i.e., cuboids). The process of decomposing the surface into cuboids using a set of nodes and arcs is equivalent to the process of

constructing a quad layout. The polycube completely determines the topology of the quad layout (i.e., combinatorial structure). In addition, the polycube adds another dimension to the quad layout because neighboring patches form cuboids. The geometric embedding of the polycube describes the locations of its nodes and arcs as well as parameterizations of its patches. Based on the work of Campen et al. [Cam14], the initial geometric embedding of the polycube's nodes and arcs is optimized. Using aligned global parameterization, the nodes are re-positioned and the arcs are re-routed across the surface in a way to achieve low overall patch distortion, as well as alignment to principal curvature directions and sharp features.

3.2 N-Symmetry Direction Fields

Spline surfaces require a quadrilateral control mesh of the object. The edges of an optimal quadrilateral mesh of an object should be orthogonal and aligned with the curvature principal directions of the surface [dAz00]. This type of mesh, called anisotropy-adapted control mesh, helps to reduce unwanted oscillations in the final spline surface. To create such control mesh, we first compute a guidance direction field that will steer the placement of the edges, then we generate the mesh itself. In other words, this direction field captures the anisotropy of the surface.

Many algorithms in computer graphics and geometry processing are based on smooth direction fields (unit tangent vector fields) defined over a surface. For instance, such a direction field was used to place hatch strokes in non-photorealistic rendering [Her00], to steer the orientation of features in texture synthesis [Pra00; Tur01; Dic02], or to remesh a surface with cells aligned with the principal curvature directions [All03]. These applications use objects of higher symmetry than simple direction fields, i.e. objects invariant by rotation of π or $\pi/2$. Such objects are called N-symmetry direction fields, where N represent the order of symmetry. N-symmetry direction fields on surfaces are fields that associate to every point of the surface a set of N unary vectors forming equal angles between radially consecutive directions. For example, a 2-symmetry (line) field can be used to guide texture placement, a 4-symmetry (cross) field can be used for quadrangular remeshing [Bom09], and a 6-symmetry field can be used for triangular and hexagonal remeshing [Nie10].

Direction fields are different from vector fields in the way that their topology is closely related to the topology of the surface on which they are defined. In particular, their singularities can not be defined as zeroes (because by definition, they have unit norm), but as holes in their definition domain. In quad remeshing, the field singularities play an important role. For instance, a singularity will generate an irregular vertex or a non-quadrilateral polygon. A variety of methods were proposed for N-symmetry direction field construction. A number of methods rely on manually placed singularities [Ray08; Lai10; Cra10] and other methods compute singularity positions automatically [Ray09; Bom09; Knö13].

Classical direction field design is much studied [Tri00; The02; Zha06]. There are also works on 2-symmetry direction fields (or tensor fields) design [Zha07]. However, these methods do not provably control singularities. Ray et al. [Ray08] were the first to present

a method for designing higher symmetry direction fields on arbitrary surfaces. They were also the first to give a formal definition of N-symmetry direction fields and rigorous mathematical tools that link the topology of the direction field to the indices of singularities and hence the Poincaré-Hopf Index Theorem. This section is organized as follows:

- We introduce the notion of N-symmetry direction field, that generalizes direction fields. We also introduce the definitions of turning number and index to characterize the singularities of a N-symmetry direction field. We finally introduce an analog of the Poincaré-Hopf theorem, implying that the indices of the singularities of a N-symmetry direction field defined on a manifold surface sum to its Euler characteristic.

- We introduce a discrete representation of N-symmetry direction fields for triangular meshes. The values of the field are defined on the faces of the mesh. In addition, we introduce the notion of connections angles or period jumps. There are one-forms attached to the edges of the dual mesh. They represent the variations of direction between two adjacent faces and enables representing singularities of arbitrary indices.

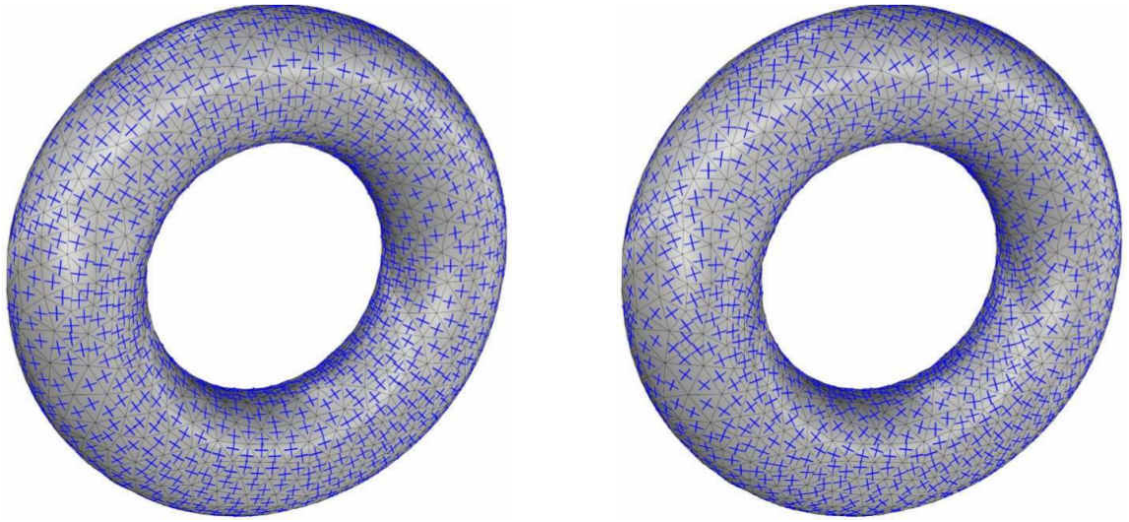
- We want to design a smooth 4-symmetry direction field (cross field) that is topologically conform with the quad layout induced by the polycube. We present an algorithm for N-symmetry direction field topological design. From a user-defined set of singularities, the algorithm constructs a smooth N-symmetry direction field satisfying these constraints. If the indices of the user-defined singularities satisfies the Poincaré-Hopf theorem, the constructed field has no other singularity.

- Within the topologically fixed space of cross field topologically conform with the polycube, we strive to find a smooth cross field that interpolates sparse directional constraints, corresponding to reliable principal directions and sharp features. We present an algorithm for N-symmetry direction field geometrical design. From a user-defined set of fixed directions, the algorithm constructs a smooth N-symmetry direction field satisfying these constraints and the given topology. We also add the use of soft constraints which, while mostly achieving accurate alignment to constraints directions, provides some freedom around singularities.

3.2.1 Continuous Representation

This section presents the fundamental tools for studying N-symmetry direction fields defined over surfaces with boundaries, and especially to study their topology. Topology is the study of properties which are invariant by continuous deformations. In other words, topology tries to answer under what condition are two objects homeomorphic (*i.e.*, topologically equivalent). For oriented surfaces with boundaries, the answer is that they need to have the same genus g and the same number of boundaries b. For this reason, we say that the two integers (g, b) are surfaces' Topological Degrees of Freedom (TDoF). We want to answer the same question for N-symmetry direction fields and find their TDoF.

We will see that the TDoF of N-symmetry direction field are the turning numbers of a homology basis of cycles. Moreover, turning numbers generalize singularities which alone do not control all of the field topology (Figure 3.1). The intuitive idea behind turning numbers is that when following a closed cycle, a N-symmetry direction field might do an arbitrary number of N^{th} turns before coming back to its starting point. For instance,



(a) (b)

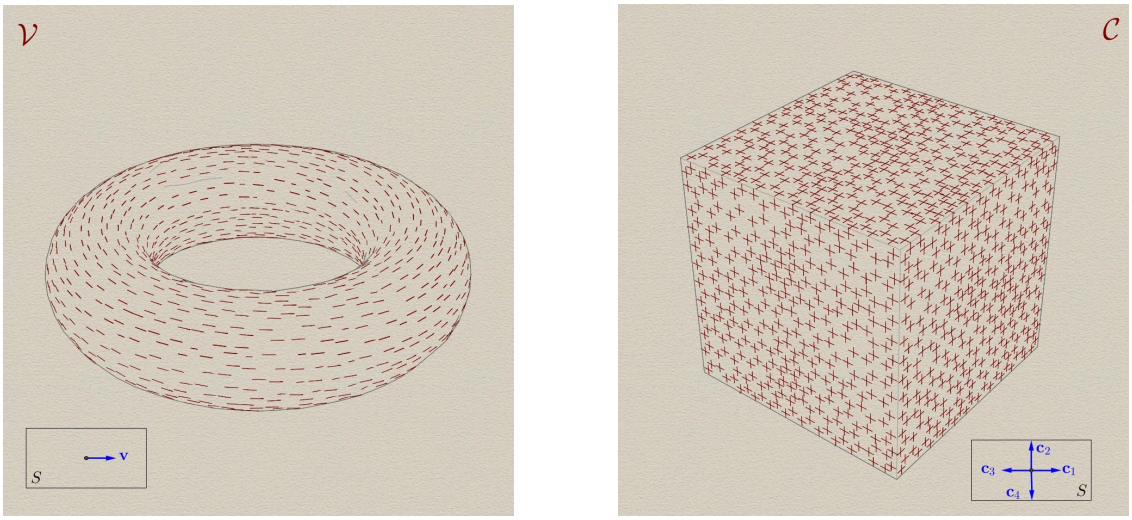
Figure 3.1: The turning number associated to a generator cycle defines topology that cannot be captured by singularity indices: two topologically different direction fields defined on a torus having no singularities but different turning numbers around homology generators (a-b).

imagine that we are traveling on earth along a cycle with a compass giving us the north direction. Then using the compass, we can count the number of turns while following the cycle. If we turn around a tree (assuming that there are no trees at earth poles), we will get one turn. But if we follow the equator or turn around a pole, we will get zero turn. We can do the same on any surface, and with any N-symmetry direction field defined over it. The number of turns is called the turning number of the field along the cycle (it depends on both the cycle and the direction field). These turning numbers capture the topology of the N-symmetry direction field: two N-symmetry direction fields have the same topology if and only if they have the same turning numbers along $2g + \max(b-1, 0)$ cycles of a homology basis (i.e., a basis for cycles on a 2-manifold). This shows that the topology of a N-symmetry direction field is entirely defined by $2g + \max(b-1, 0)$ integers (multiple of $1/N$), which are its TDoF.

3.2.1.1 Continuous Direction Fields

A *direction field* defined on a surface S is a tangent unit vector field: at each point of the surface, there exists a direction $\mathbf{w}$ such that $\|\mathbf{w}\| = 1$ and $\mathbf{w} \cdot \mathbf{n} = 0$, where $\mathbf{n}$ is the normal of S. A *N-symmetry direction field* $\mathcal{W}$ is a multivalued direction field: at each point of the surface S, there exists a N-symmetry direction $\mathbf{w}$ which is a set of N directions $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$ preserved by rotations of $2\pi/N$ around the normal $\mathbf{n}$ of S (Figure 3.2). In the following, we will omit the term "N-symmetry" for brevity.

The set of derivable directions fields of a surface S is denoted by $\mathcal{D}_N(S)$. Two direction fields $\mathcal{W}_1, \mathcal{W}_2 \in \mathcal{D}_N(S)$ are called homotopic if there exists a continuous function $\Gamma : [0,1] \to \mathcal{D}_N(S)$ such that $\Gamma(0) = \mathcal{W}_1$ and $\Gamma(1) = \mathcal{W}_2$. In other terms, two direction fields



(a) (b)

Figure 3.2: N-symmetry direction fields. A 1-symmetry direction field (i.e., unit vector field) $\mathcal{V}$ on a torus: at each point of the surface S, there exists one direction $\mathbf{v}$ (a). A 4-symmetry direction field (i.e, cross field) $\mathcal{C}$ on a cube: at each point of the surface S, there exists one 4-symmetry direction $\mathbf{c}$ which is a set of 4 directions $\{\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4\}$ invariant by a rotation of $\pi/2$ around the normal (b).

are homotopic (i.e., have the same topology) if they can be continuously transformed one into another. As this is a relation of equivalence, homotopy classes can be defined as the sets of all direction fields equivalent to a given one. Homotopy classes of direction fields can be characterized by the turning numbers of the field along some cycles.

3.2.1.2 Turning Numbers

The turning numbers of a direction field along a cycle corresponds to the number of rotations of the field along this cycle, and are characteristic of homotopy classes of direction fields, hence of their topology. The turning number $T_{\mathcal{W}}(\gamma)$ of a direction field $\mathcal{W}$ around a cycle γ can be expressed as [Ray08]:

$$T_{\mathcal{W}}(\gamma) = \frac{1}{2\pi} \oint_{\gamma} (\kappa_{\mathcal{W}} - \kappa_{\gamma}) ds, \tag{3.1}$$

where $\kappa_{\mathcal{W}}$ is the geodesic curvature of the direction field $\mathcal{W}$ around the cycle γ, and κ_{γ} is the geodesic curvature of the tangent vector of the cycle γ. The turning number is always an integer multiple of $1/N$, and corresponds intuitively to the number of N^{th} turns the direction field $\mathcal{W}$ does in the Darboux frame along γ.

Turning numbers have two fundamental properties which make them useful for studying direction field topology:

- Let S be a surface (2-manifold with boundary ∂S) embedded in $\mathbb{R}^3$ with genus g and b

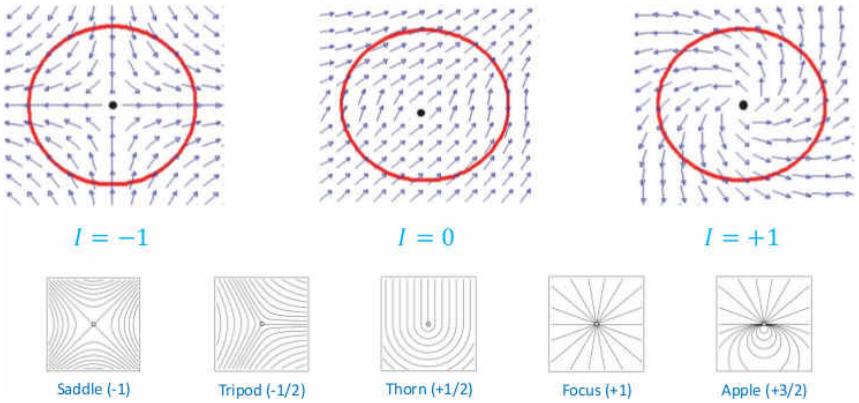


Figure 3.3: The index of singularity I of a vector field $\mathcal{V}$ at a point $\mathbf{p}$ is the number of counter-clockwise rotations that the vectors make as we travel counter-clockwise along a loop around $\mathbf{p}$.

boundaries, then [Ray08]:

$$T_{\mathcal{W}}(\partial S) = \chi(S), \tag{3.2}$$

where $\chi(S) = 2 - 2g - b$ is the Euler characteristic of S.

- Two direction fields $\mathcal{W}_1$ and $\mathcal{W}_2$ defined over a surface S are homotopic if and only if they have the same turning numbers along the cycles of an homology basis H of S [Ray08]:

$$\mathcal{W}_1 \equiv \mathcal{W}_2 \Leftrightarrow \forall \gamma \in H(S),\ T_{\mathcal{W}_1}(\gamma) = T_{\mathcal{W}_2}(\gamma), \tag{3.3}$$

where $\equiv$ denotes the homotopic equivalence.

3.2.1.3 Singularities

For a vector field $\mathcal{V}$ defined on a surface S, a singularity is a point $\mathbf{p} \in S$ such that $\mathcal{V}(\mathbf{p}) = \mathbf{0}$. Singularities can be of various types, according to how the vector field winds around the singularity. This property is referred to as index in vector field topology [Tri02] (Figure 3.3). On a surface S, The Poincaré-Hopf Index Theorem states that the indices of all singularities sum to its Euler characteristic. Direction fields are different from vector fields in the way that their singularities can not be defined as zeroes (because by definition, they have unit norm), but as holes in their definition domain.

For direction fields, the concept of replacing singularities with holes allows singularities to be handled as borders. This way the singularity is characterized by the behevior of the direction field along the boundary of the hole. The other advantage of this concept is that borders have indices, which correspond to the index of the singularity obtained if we contract the border to a single point. Hence, the turning numbers of cycles around singular points characterize the singularities in the direction field. For a direction field $\mathcal{W}$, let $\mathbf{p}$ be a singular point and $\partial\mathbf{p}$ the border of a disk covering $\mathbf{p}$. The index $I_{\mathcal{W}}(p)$ of the singular point $\mathbf{p}$ is related to the turning number $T_{\mathcal{W}}$ around the cycle $\partial\mathbf{p}$ by the following equation [Ray08]:

$$I_{\mathcal{W}}(\mathbf{p}) = 1 + T_{\mathcal{W}}(\partial\mathbf{p}). \tag{3.4}$$

With this definition for the index of singular points in a direction field, the Poincaré-Hopf theorem can be generalized to N-symmetry direction fields defined on a surface S [Ray08]:

$$\sum_{i=1}^{s} I_i = \chi(S), \tag{3.5}$$

where s is the number of singular point $\mathbf{p}_i$ with index I_i, and $\chi(S)$ is the Euler characteristic of the surface S.

3.2.2 Discrete Representation

With this understanding of continuous direction fields topology, we introduce the notion of connection angles and period jumps and use them to build a discrete field representation. We work with a triangulated connected 2-manifold mesh M with a set of vertices V, edges E and faces F. Its dual mesh M^* has a set of vertices V^*, edges E^* and faces F^*. The dual mesh doesn't have to be explicitly constructed since dual quantities can be stored on the corresponding primal elements. M is also oriented: each face $f \in F$ has coherent normal, and each edge $e \in E$ has an orientation. This allows to define a unique orientation for each dual edge $e^* \in E^*$.

3.2.2.1 Discrete Direction Fields

The first step of the discretization of a direction field $\mathcal{W}$ on a mesh M is the choice of tangent planes. Storing directions at vertices V requires the definition of tangent planes in terms of extrinsic geometry. Storing directions at faces F is more natural because tangent planes can be defined intrinsically. Hence, in the context on this thesis, direction fields will be represented at faces F of the mesh M, or equivalently at vertices V^* of the dual mesh M^*.

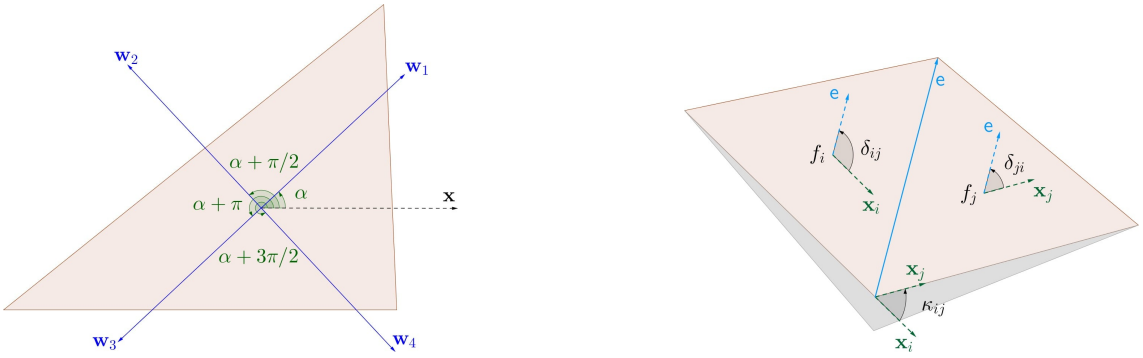
The direction field is then sampled by defining one direction on each face. This is done by choosing a local orthonormal frame $(\mathbf{x}, \mathbf{y})$ on each face f. $\mathbf{x}$ is the unit vector along one of the oriented edges e of f, and $\mathbf{y} = \mathbf{n} \times \mathbf{x}$ with $\mathbf{n}$ being the normal of f. The direction $\mathbf{w}$ defined on f can then be expressed in terms of polar coordinates. Since $\mathbf{w}$ has unit norm, it is completely parameterized by the polar angle α it forms with $\mathbf{x}$ (Figure 3.4a). Angles in adjacent faces can be represented in a common coordinate frame by flattening both faces along their common edge. Given an angle α_i^i in face f_i, it can be expressed in an adjacent face f_j as:

$$\alpha_i^j = \alpha_i^i - \delta_{ij} + \delta_{ji} = \alpha_i^i + \kappa_{ij} \text{ with } \kappa_{ij} = -\delta_{ij} + \delta_{ji} = -\kappa_{ji}, \tag{3.6}$$

where δ_{ij} and δ_{ji} are the angles between the shared edge e and the reference direction $\mathbf{x}_i$ and $\mathbf{x}_j$ respectively, and $\kappa_{ij} \in]-\pi, \pi]$ represents the angle between reference directions $\mathbf{x}_i$ and $\mathbf{x}_j$ (Figure 3.4b).

3.2.2.2 Connection Angles and Period Jumps

However by representing the direction field only at a set of points, its topology cannot be accessed. In particular, there is no information about the behavior of the direction field between the discretization points. Assume that we want to interpolate a direction $\mathbf{w}$



(a) (b)

Figure 3.4: N-symmetry direction fields representation. A 4-symmetry direction $\mathbf{w}$ on a face f is a set of directions $\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \mathbf{w}_4\}$ defined as the image of a reference direction $\mathbf{x}$ by rotations of $\alpha + k(\pi/2), k \in \{0, 1, 2, 3\}$ (a). Angles in adjacent faces f_i and f_j can be expressed in a common coordinate frame using the rotation angle κ_{ij} between reference directions $\mathbf{x}_i$ and $\mathbf{x}_j$ (b).

along an edge $[AB]$ knowing $\mathbf{w}_A$ and $\mathbf{w}_B$. If the direction field is continuous, the angular variation ω along $[AB]$ verifies:

$$\omega_{AB} = \int_A^B d\alpha = \angle(\mathbf{w}_A, \mathbf{w}_B) + (2\pi/N)p_{AB}, \tag{3.7}$$

where $\angle(\mathbf{w}_A, \mathbf{w}_B) \in]-\pi, \pi]$ is the angle between $\mathbf{w}_A$ and $\mathbf{w}_B$, and $p \in \mathbb{Z}$ is an integer. The angular variation ω has infinitely many possible values for all the possible values of p (Figure 3.5a). This ambiguity can be solved by (Figure 3.5b):

- specifying one direction at A or B and the real ω. $\omega \in \mathbb{R}$ is called the *connection angle* and it specifies the amount of rotation the direction undergoes when passing from A to B. We will refer to this method, developed by Crane et al. [Cra10], as the *connection based discretization*. In this case, an angle α_i^i defined on face f_i, expressed as α_i^j in an adjacent face f_j, is in general equal to α_j^j:

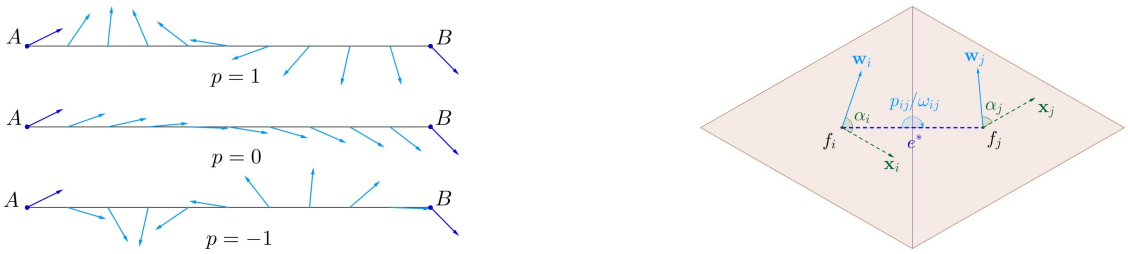
$$\alpha_i^j = \alpha_i^i + \kappa_{ij} + \omega_{ij} \text{ and } \alpha_i^j = \alpha_j^j. \tag{3.8}$$

  Hence a direction angle on one face and connection angles on all dual edges are required to completely define the discrete direction field.

- specifying the directions at A and B and the integer p. $p \in \mathbb{Z}$ is called the *period jump* [Li06] and it specifies the number of N^{th} turns the direction $\mathbf{w}$ undergoes when passing from A to B. We will refer to this method, developed by Ray et al. [Ray08], as the *period jump based discretization*. In this case, an angle α_i^i defined on face f_i, expressed as α_i^j in an adjacent face f_j, is in general different than α_j^j:

$$\alpha_i^j = \alpha_i^i + \kappa_{ij} + (2\pi/N)p_{ij} \text{ and } \alpha_i^j \neq \alpha_j^j. \tag{3.9}$$

  Hence direction angles on all faces and period jumps on all dual edges are required to completely define the discrete direction field.



(a) (b)

Figure 3.5: N-symmetry direction fields discretization. Between two directions given at points A and B, different interpolations are possible (a). This ambiguity can be solved by specifying an angle on one face and a real ω. ω is called the connection angle and it specifies the amount of rotation a direction undergoes when traversing a dual edge. This ambiguity can be also solved by specifying angles on both faces and an integer p. p is called the period jump and it specifies the number of N^{th} turns a direction undergoes when traversing a dual edge (b).

For the sake of simplicity, the angle α_i^i defined on face f_i will be simply denoted by α_i. In the context of this thesis, we use both discretization methods. Connection based discretization (3.8) is used for topological design. This is done by finding the proper connection angles on dual edges satisfying the topological constraints. Period jump based discretization (3.9) is used for geometrical design. This is done by fixing period jumps on all dual edges and then fitting direction angles on primal faces to the geometrical constraints.

3.2.3 Topological and Geometrical Design

In order to obtain a guiding field for the aligned global parameterization, we design a 4-symmetry direction field (i.e., cross field) $\mathcal{C}$ on the surface. Direction field design is the generation of a smooth direction field from a set of constraints. These constraints can be topological (i.e., imposed singularities) and/or geometrical (i.e., imposed directions). We strike for a balance between three important properties of direction fields : smoothness, singularity positions/indices, and alignment with local geometry. Topologically, $\mathcal{C}$ must be conform with the quad layout $\mathcal{L}$ induced by the polycube. This means that $\mathcal{C}$ must be singular only at the position of irregular nodes of $\mathcal{L}$. Geometrically, $\mathcal{C}$ must follow the principal directions and sharp features of the surface.

The basic idea of our designing mechanism is to find the right values of direction angles and connections angles/period jumps over the whole triangular mesh that correspond to the desired direction field topology and geometry. The design procedure consists of the following two steps:

- Topologic step: for a genus-g surface with b boundaries, the topology of the direction field is entirely defined by its turning numbers along $2g$ homology generator cycles, $\max(b-1,0)$ boundary cycles, and s cycles around singularities. We need to fix all these TDoF in order to restrict to direction fields topologically compatible with the quad

layout induced by the polycube. We obtain the correct values of connection angles, and hence the topology of the direction field is fixed.

- Geometric step: within the topologically fixed space of direction fields, we want to find a smooth direction field interpolating sparse directional constraints and sharp features. This way, the direction field can be made adapted to the anisotropy of the triangular mesh. After computing the fixed values of period jumps accommodating the given set of directional constraints, we obtain the direction angles such that the so-created direction field has the smoothest geometry.

3.2.3.1 Smoothness Energy

A common requirement on direction fields is smoothness. For a mesh M, the curvature of a direction field $\kappa_{\mathcal{W}}$ along a dual edge e^* is the angle difference between neighboring faces represented in a common coordinate frame [Ray08]:

$$\kappa_{\mathcal{W}}(e^*_{ij}) = \omega_{ij} = \alpha_j - [\alpha_i + \kappa_{ij} + (2\pi/N)p_{ij}]\,, \tag{3.10}$$

where e^*_{ij} is the dual edge oriented from face f_i to face f_j, $\omega_{ij} \in \mathbb{R}$ is the connection angle, $\alpha_i \in \mathbb{R}$ and $\alpha_j \in \mathbb{R}$ are direction angles in faces f_i and f_j respectively, $\kappa_{ij} \in]-\pi, \pi]$ is the angle between local frames, and $p_{ij} \in \mathbb{Z}$ is the period jump along the dual edge e^*_{ij}.

The smoothness energy of a direction field $E_s(\mathcal{W})$ can be measured as its integrated squared curvature $\kappa_{\mathcal{W}}$ [Ray08]:

$$E_s(\mathcal{W}) = \sum_{e^*_{ij} \in E^*} \left\| \kappa_{\mathcal{W}}(e^*_{ij}) \right\|^2 \tag{3.11}$$

$$= \sum_{e^*_{ij} \in E^*} \|\omega_{ij}\|^2 \tag{3.12}$$

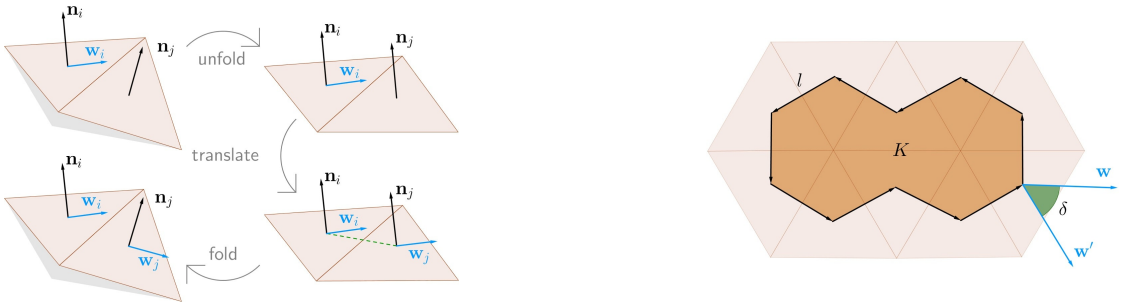
$$= \sum_{e^*_{ij} \in E^*} \|\alpha_i + \kappa_{ij} + (2\pi/N)p_{ij} - \alpha_j\|^2\,. \tag{3.13}$$

Topological design is based on the minimization of energy (3.12), and geometrical design is based on the minimization of energy (3.13).

3.2.3.2 Topological Design

The problem solved by the algorithm can be formalized as follows: given a triangulated surface S of genus g, and a subset of vertices $V_s = v_i \in V$ with desired non-zero indices I_i respectively (such that $\sum I_i = 2 - 2g$), the algorithm generates a direction field such that the index of v_i is equal to I_i and that provably does not contain any other singularity.

On a smooth surface S, a *connection* Ω describes how a direction rotates locally when moved an infinitesimal distance along a given direction. Once this local information is provided, by integrating these infinitesimal rotations along a curve, a direction can be mapped between distinct tangent planes. This process is called *parallel transport*. For a mesh M, a connection can be represented as an angle ω on each dual edge e^*. This angle represents the integrated rotation that a direction undergoes when passing from one face



(a) (b)

Figure 3.6: Discrete connections. Transport using Levi-Civita connection can be described as follows: unfold the faces isometrically into the plane, translate the vector across the shared dual edge, and then fold the faces back into their initial configuration (a). In general, a vector parallel transported around a loop l will not end up where it started. The angle between the initial direction $\mathbf{w}$ and the transported direction $\mathbf{w}'$ is called the holonomy δ. Every connection has an associated curvature K. If l bounds a region then the curvature K over this region equals the holonomy δ around the loop (b).

to its neighbor. Once a connection Ω is defined, there is a straightforward way to construct a direction field $\mathcal{W}$ on the mesh M. Namely, starting with an arbitrary face and an initial direction, this direction is transported to adjacent faces until the entire mesh is covered.

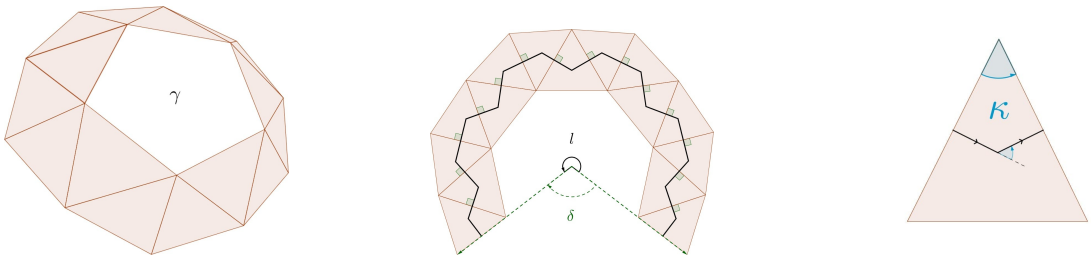
The simplest connection is obtained by setting connection angles to zero on all dual edges: this is the Levi-Civita connection (Figure 3.6a). In general for a curved surface S, a direction parallel transported around a loop l by a connection will not return to its original orientation. The difference in angle between the initial and final directions is called the *holonomy* δ of the connection around the loop l. Every connection has also an associated *curvature* K. In particular if the loop l bounds a region of the surface S, then the curvature K of the connection over this region equals its holonomy δ around the loop l (Figure 3.6b).

A cycle γ is a sequence of primal faces that form a loop (Figure 3.7a). This loop is defined as the sequence of dual edges orthogonal to primal edges shared by adjacent faces of γ after unfolding the cycle isometrically to the plane (Figure 3.7b). From this definition, the discrete geodesic curvature κ_γ at a face equals the signed angle between corresponding primal edges (Figure 3.7c). In this case, the holonomy δ_γ of the Levi-Civita connection around the cycle γ can be defined as [Myl12]:

$$\delta_\gamma = -\sum_{e^* \in \gamma} \kappa_\gamma(e^*) \Rightarrow \delta_\gamma = -\kappa_\gamma, \tag{3.14}$$

where κ_γ is the total discrete geodesic curvature of the cycle γ.

Because of holonomy, depending on which connection is used, the direction field might not be consistently defined. A direction field is considered to be consistent only if directions are mapped to themselves modulo $2\pi/N$ by parallel transport. For instance, the holonomy of the Levi-Civita connection equals the Gaussian curvature, so in general a direction transported around a closed loop is not mapped back to itself. As a consequence, transport



(a) (b) (c)

Figure 3.7: Cycles holonomy. A cycle γ is a sequence of primal faces (a). By unfolding γ isometrically to a plane, a loop l is defined as the sequence of dual edges orthogonal to primal edges shared by adjacent faces of γ (b). Within a face, l turns an angle κ, computed as the oriented angle between corresponding adjacent primal edges (c).

from one point to another will depend on the choice of path. Crane et al. [Cra10] showed that transport via trivial connections, i.e., connections with globally vanishing holonomy, is path-independent. However zero holonomy everywhere implies zero curvature everywhere. According to the Gauss-Bonnet theorem, the total Gaussian curvature of a surface is equal to $2\pi\chi$, where χ is the Euler characteristic of the surface. To overcome this problem, the total curvature is distributed over the mesh in a way that doesn't interfere with parallel transport. In particular, curvature will be concentrated at boundaries and/or at a set of vertices called *singularities* in increment of $2\pi/N$. In this case, for a given cycle γ, a direction might do an arbitrary number of N^{th} turns before coming back to its starting point. This quantity is called the turning number.

The turning number T_γ of the direction field $\mathcal{W}$ along the cycle γ [Ray08]:

$$2\pi T_\gamma = \sum_{e^* \in \gamma} \kappa_{\mathcal{W}}(e^*) - \sum_{e^* \in \gamma} \kappa_\gamma(e^*), \tag{3.15}$$

where $\kappa_{\mathcal{W}}$ is the curvature of the direction field $\mathcal{W}$, and κ_γ is the geodesic curvature of the cycle γ.

By substituting the curvature $\kappa_{\mathcal{W}}$ of a direction field along a dual edge by its connection angle ω (3.10), and the total geodesic curvature κ_γ of a cycle by its holonomy δ_γ (3.14), the turning number T_γ (3.15) of the direction field $\mathcal{W}$ along the cycle γ can also be expressed as:

$$2\pi T_\gamma = \sum_{e^* \in \gamma} \omega(e^*) + \delta_\gamma \Rightarrow \sum_{e^* \in \gamma} \omega(e^*) = -\delta_\gamma + 2\pi T_\gamma. \tag{3.16}$$

In terms of connections, this states that the trivial connection angles around a given cycle should cancel the holonomy found with Levi-Civita, and potentially add a certain amount of curvature in order to respect the Gauss-Bonnet theorem.

The topology of a direction field is entirely defined by the turning numbers of the direction field along $2g$ homology generator cycles, $\max(b-1, 0)$ boundary cycles, and s cycles around singularities [Ray08]. We need to fix all topological degrees of freedom in order to restrict to cross fields topologically compatible with the quad layout $\mathcal{L}$. The turning

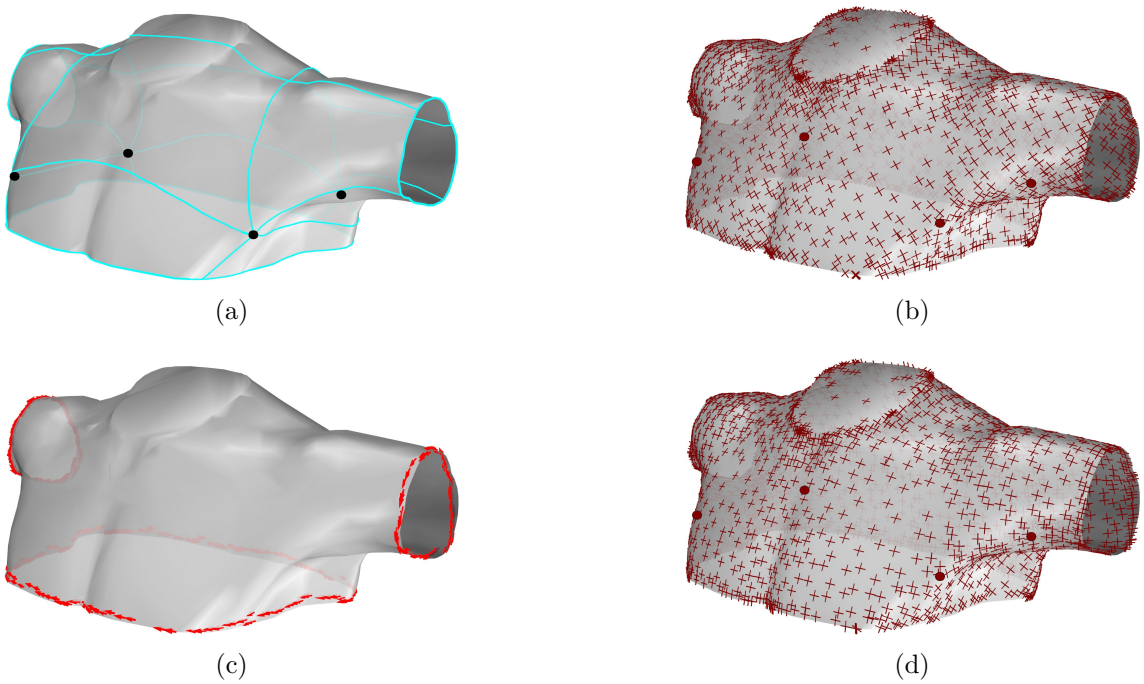


Figure 3.8: N-symmetry direction fields topological design. The TORSO model along with its quad layout $\mathcal{L}$ obtained from polycube decomposition. The quad layout singular nodes are depicted by circles (a). The cross field $\mathcal{C}$ obtained by setting the initial direction angle to a random value. The cross field singular vertices (depicted by circles) are conform with the quad layout $\mathcal{L}$ singular nodes (b). The directional constraints on boundary faces corresponding to the direction of boundary edges (c). The cross field obtained by setting the initial direction angle to the mean value of all directional constraint angles expressed in a common face. Locally, the cross field directions don't respect the directional constraints on each face (d).

numbers around singularities, homology generators and boundary cycles are determined using the method described by Campen et al. [Cam14]. For an irregular node its turning number is determined from its valence m as $-0.25m$. For a homology generator or boundary cycle, the turning number needs to be fixed to $\pm 0.25\,(n_n - n_a)$, where n_a is the number of arcs crossed by the cycle and n_n the number of nodes in the closest homotopic arc cycle.

Crane et al. [Cra10] presented an algorithm that, given the above turning numbers, determines the appropriate trivial connection by minimizing energy $E_s(\mathcal{W})$ (3.12). Finally, we still need to define one direction to construct the direction field. To obtain this direction, all directional constraints are transported from their respective faces to a common face f_0. The initial direction angle α_0 is defined to be the mean of all directional constraint angles expressed in f_0 (Figure 3.8).

3.2.3.3 Geometrical Design

Within this topologically fixed space, we now strive to find a smooth direction field that interpolates sparse directional constraints, corresponding to reliable principal directions, feature curves directions, or user specified intents. In other terms, given a mesh M and a subset of faces $F^c \subset F$ with constrained directions α^c, we search for the smoothest interpolating direction field $\mathcal{W}$ restricted to a given topology.

In the general case ($\mathcal{W}$'s topology is free), as presented by Bommes et al. [Bom09], the smoothest interpolating direction field $\mathcal{W}$ is computed by finding integer valued period jumps p on dual edges and real valued direction angles α on faces minimizing the direction field smoothness energy $E_s(\mathcal{W})$ (3.13) subject to directional constraints α^c on the set of faces F^c. In our case, the direction field $\mathcal{W}$ is restricted to a given topology so the integer valued period jumps p are fixed, and the real valued angles α are the only variables in the constrained optimization problem. For the mesh M, let $|E|$ be the number of its edges and $|F|$ the number of its faces. Then the energy $E_s(\mathcal{W})$ can be expressed as:

$$E_s(\mathcal{W}) = \sum_{e^*_{ij} \in E^*} \|\alpha_i + \kappa_{ij} + (2\pi/N)p_{ij} - \alpha_j\|^2 = \|\mathbf{A}\boldsymbol{\alpha} - \mathbf{b}\|^2, \tag{3.17}$$

where $\mathbf{A} \in \mathbb{R}^{|E|\times|F|}$, $\mathbf{b} \in \mathbb{R}^{|E|}$, and $\boldsymbol{\alpha} \in \mathbb{R}^{|F|}$ is the vector of unknowns corresponding to direction angles on faces. For the k^{th} dual edge e^*_{ij} oriented from face f_i to face f_j, $\mathbf{A}(k,i) = 1$, $\mathbf{A}(k,j) = -1$, and $\mathbf{b}(k) = -[\kappa_{ij} + (2\pi/N)p_{ij}]$. The constrained minimization problem can then be formulated as follows:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^{|F|}} \|\mathbf{A}\boldsymbol{\alpha} - \mathbf{b}\|^2 \text{ subject to } \alpha_i = \alpha_i^c \ , \ \forall f_i \in F^c. \tag{3.18}$$

For a given face $f^c \in F^c$, it is not inherently clear by which of $\mathcal{W}$'s N directions the directional constraint is to be interpolated. This choice has an important effect on the fixed values of the period jumps. Assume that for a given face f_i, the direction angle is given by α_i, and the period jumps are given by p_{i0}, p_{i1} and p_{i2}. If the angle α_i is rotated by a multiple of $2\pi/N$ around the normal (i.e., $\bar{\alpha}_i = \alpha_i + k \cdot 2\pi/N$), in order to preserve the direction field topology, this change must be compensated by updating the affected period jumps (i.e, $\bar{p}_{ij} = p_{ij} - k, \ j = 0, 1, 2$). Using a modified version of the technique for reducing the search space [Bom09], we proceed as follows to compute the fixed values of period jumps accommodating the given set of directional constraints (Figure 3.9):

1. We construct a forest of Dijkstra trees of the dual mesh, where each constrained face in F_c is the root of a separate tree such that no tree connects constrained faces [Bom09].

2. In each constrained face f_i^c, we normalize the direction angle α_i and the constraint angle α_i^c to $]-\pi/N; \pi/N]$. This choice is justified by the fact that a direction field is invariant by rotations of $2\pi/N$ around the normal. We then find the integer $k \in \mathbb{Z}$ minimizing the absolute difference $|\ \alpha_i + k \cdot 2\pi/N - \alpha_i^c\ |$. Finally, the interpolating direction in face f_i^c is set to be $\bar{\alpha}_i = \alpha_i + k \cdot 2\pi/N$.

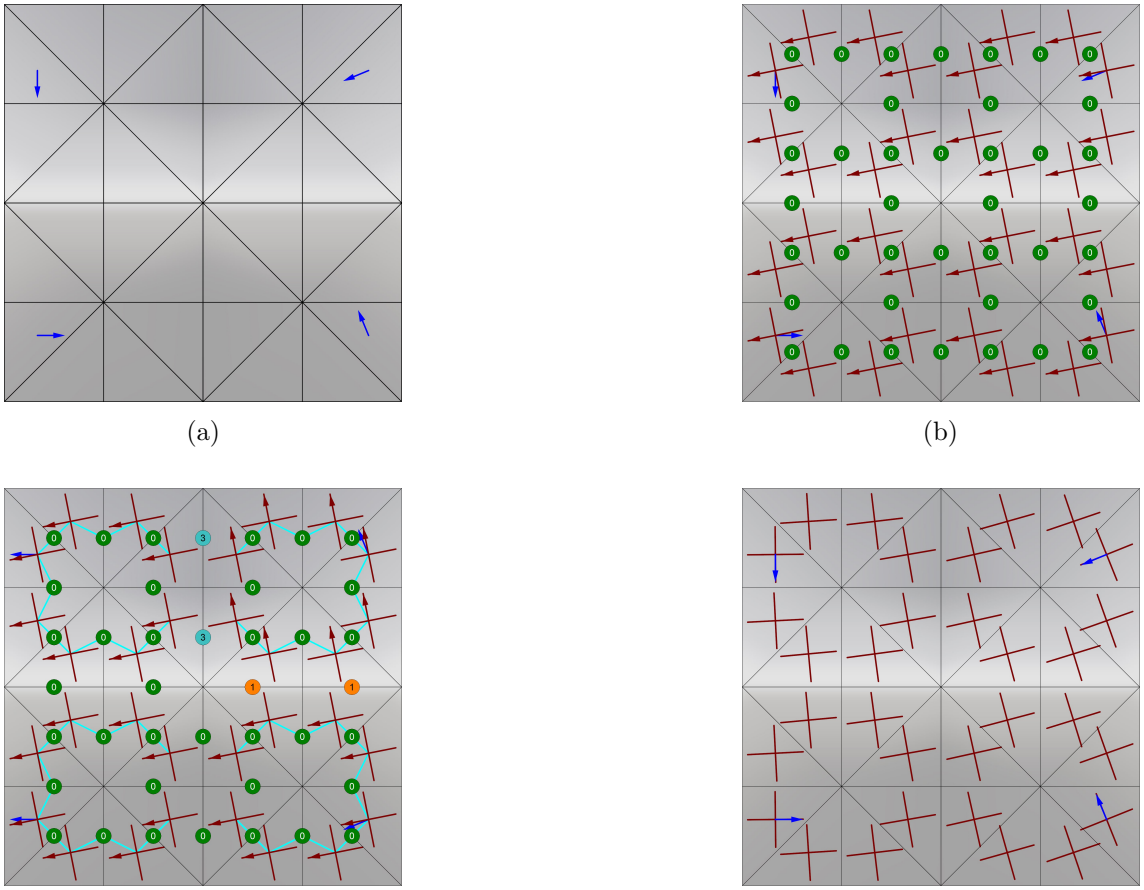


Figure 3.9: N-symmetry direction fields directional constraints interpolation. We are searching for the smoothest cross field with no singularities and interpolating 4 directional constraints (a). The cross field satisfying topological constraints and constructed using the mean value of the directional constraints. The primary directions are indicated by an arrow on each face, and the period jumps are indicated by a value on each edge (b). The reconstructed cross field starting at constrained faces and following the forest of Dijkstra trees (c). The cross field satisfying topological and geometrical constraints (d).

3. Starting at constrained faces f^c with interpolating directions $\bar{\alpha}$, we reconstruct the direction field using the connection angles and following the forest of Dijkstra trees. We note that at this point, the direction field is unchanged : it was only discretized differently so its period jumps could agree with the directional constraints.

4. The fixed period jumps p are defined to be the integers minimizing the direction field curvature $\kappa_{\mathcal{W}}$ (3.10) on each dual edge.

The geometrical direction field is the solution of a symmetric linear system obtained by setting the gradient of the energy E_s (3.18) to zero (Figure 3.10):

$$\frac{\partial E_s(\mathcal{W})}{\partial \boldsymbol{\alpha}} = \mathbf{0} \Rightarrow \mathbf{A}^t\mathbf{A} \cdot \boldsymbol{\alpha} = \mathbf{A}^t\mathbf{b}. \tag{3.19}$$

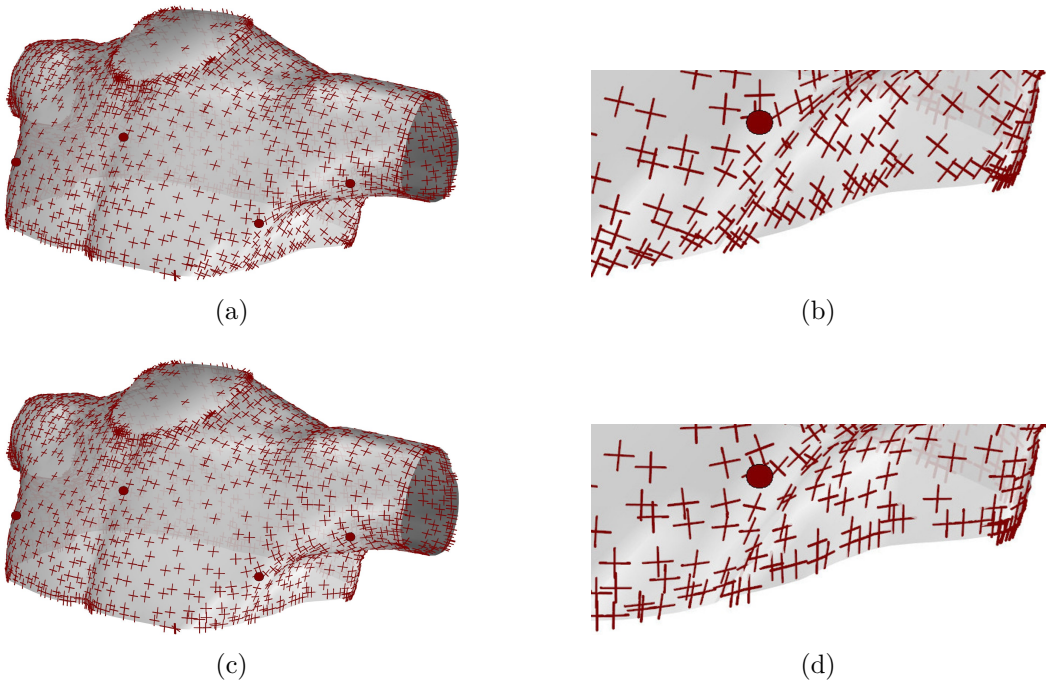


Figure 3.10: N-symmetry direction fields geometrical design. The topological cross field doesn't satisfy all directional constraints on boundary faces corresponding to boundary edges (a-b). The geometrical cross field satisfies all directional constraints while preserving the same topology (c-d).

Hard constraints can be difficult to handle because in some cases additional singularities should arise to exactly satisfy all the constraints. To prevent undesired singularities and have some freedom around imposed singularities, the constrained minimization problem (3.18) can be modified to integrate soft directional constraints:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^{|F|}} (1-\lambda)\left\|\mathbf{A}\boldsymbol{\alpha}-\mathbf{b}\right\|^2 + \lambda\left\|\boldsymbol{\alpha}-\hat{\boldsymbol{\alpha}}\right\|^2 \text{ subject to } \alpha_i = \alpha_i^c \ , \ \forall f_i \in F^c, \tag{3.20}$$

where $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^{|F|}$ is the set of soft constraints on faces, and the parameter $\lambda \in [0,1]$ is used to control the trade off between smoothness and fitting. A value of $\lambda = 0.75$ proved to perform very well in practice, and has been used in all the examples.

3.3 Aligned Global Parameterization

Once the guidance direction field is computed, we now study how to generate a coordinate system of the object (i.e., global parameterization), aligned with the direction field. Every mesh M can be cut to a topological disk M_c, by removing a set of seam edges from the mesh. In this case, a global parameterization (Figure 3.11) is a piecewise linear map from the mesh $M_c \in \mathbb{R}^3$ to a topological disk domain $D \in \mathbb{R}^2$. Since the parameterization should be piecewise linear, it is sufficient to assign a (u, v) parameter value to each face corner of the mesh. The parameterization should be locally oriented according to the guiding field. This implies that the gradients ∇u and ∇v of the piecewise linear scalar

fields u and v should follow the cross field directions $\mathbf{u}^c$ and $\mathbf{v}^c$ on each face. The function u and v are found by minimizing the following energy functional:

$$E_p = \int_S \left[\|\nabla u - h\mathbf{u}^c\|^2 + \|\nabla v - h\mathbf{v}^c\|^2 \right] dS, \tag{3.21}$$

where h is a scale factor that sets the correspondence between the length scale of the parametric domain and the surface. For a mesh M, the discrete version of this energy functional is:

$$E_p = \sum_{f \in F} \left[\|\nabla u - h\mathbf{u}^c\|^2 + \|\nabla v - h\mathbf{v}^c\|^2 \right] A_f, \tag{3.22}$$

where A_f is the area of the face f.

The most popular and actively researched class of quadrilateral-remeshing techniques is the family of parameterization based quad meshing methods. They all strive to generate an integer-grid map [Bom13a], *i.e.* a parameterization of the input surface into $\mathbb{R}^2$ such that the canonical grid of integer isolines forms a quad mesh when mapped back onto the surface in $\mathbb{R}^3$. Parameterization based quad meshing methods can roughly be divided into those that employ some form of harmonic parameterization [Don06; Ton06; Hua08; Zha10] and those that generate parameterizations which minimize some alignment energy, e.g. to achieve curvature and/or feature alignment [Ray06; Käl07; Bom09; Myl10; Myl13]. Given an input mesh $M = (V, E, T)$, an integer-grid map Γ is the union of linear maps $\Gamma_i : \mathbb{R}^3 \to \mathbb{R}^2$ that map each triangle $(\mathbf{p}_i, \mathbf{q}_i, \mathbf{r}_i) \in \mathbb{R}^{3\times 3}$ of M to a triangle $(\mathbf{u}_i, \mathbf{v}_i, \mathbf{w}_i) \in \mathbb{R}^{2\times 3}$ in the plane. Moreover, it satisfies three constraints:

- The transition functions $\mathbf{g}_{ij}$ mapping the chart of triangle f_i to the chart of the adjacent triangle f_j have to be grid automorphisms [Käl07], i.e. be of the form

$$\mathbf{g}_{ij}(\mathbf{a}) = R_{90}^{p_{ij}}\mathbf{a} + \mathbf{t}_{ij}, \tag{3.23}$$

  where R_{90} is a rotation by $\pi/2$, $p_{ij} \in \mathbb{Z}$ is the matching, and $\mathbf{t}_{ij} \in \mathbb{Z}^2$ is an integer translation.

- Singular points have to be mapped to integer coordinates, i.e.

$$\Gamma(\mathbf{s}_i) \in \mathbb{Z}^2, \ \forall \mathbf{s}_i \in S, \tag{3.24}$$

  where S is the set of singular points in M.

- The image of each triangle has to have a positive area:

$$\det(\mathbf{v}_i - \mathbf{u}_i, \mathbf{w}_i - \mathbf{u}_i) > 0. \tag{3.25}$$

3.3.1 Seamless Parameterization

We are interested in global parameterizations where the transition functions across seems are not arbitrary but of a very restricted class: rigid transformations with a rotation angle of some multiple of $\pi/2$. In addition for quadrangulation, it is necessary that the transitional part of the transitions are integral (3.23). In this case, we talk about integral seamless parameterization. We follow the method of Bommes et al. [Bom09] to compute such parameterization:

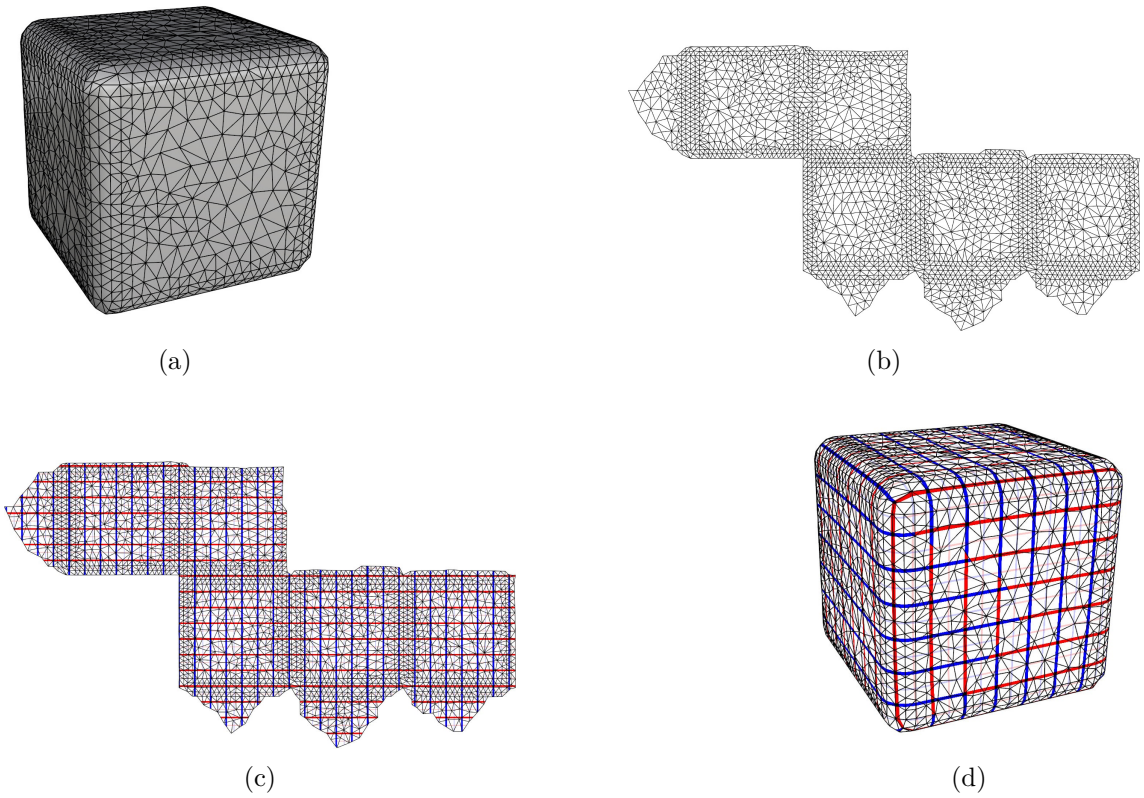
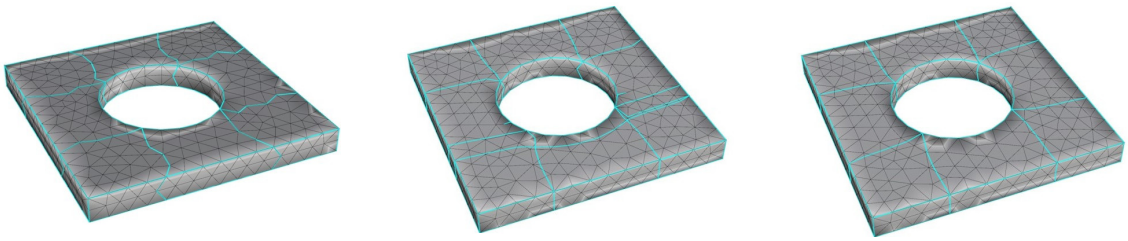


Figure 3.11: Global parameterization. A triangulated mesh (a) and its parameterization (b). The intersection between the Cartesian Grid and the parameterization (c) is inversely mapped onto the mesh in order to obtain a quadrilateral mesh (d).

- A cut is a connected graph G of mesh edges, such that $M \setminus G$ is topologically equivalent to a disk $D \in \mathbb{R}^2$. The appropriate cut graph is computed in two steps. First we start from a random face and grow a topological disk by constructing a spanning tree of the dual mesh. Thus the primal of all non spanning tree edges is already a cut graph G which transforms M into a topological disk D. The size of this cut graph can be significantly reduced by iteratively removing all open paths. In the second step, paths connecting the cross field singularities to the cut graph are added.

- The cross-field is made consistent: the angles representing the field are changed so that the period jumps across all non-cut edges are equal to zero. It is possible to achieve this since the cut passes through all singularities. As the period jumps are all equal to zero at non-cut edges, if we arbitrarily label one of the directions of the field on a face f as $\mathbf{u}^c$, the label can be consistently propagated to all other faces. The $\pi/2$ rotated vectors of the cross-field are labeled $\mathbf{v}^c$. The vectors $\mathbf{u}^c$ and $\mathbf{v}^c$ are the target values for the gradients of the parametric coordinates ∇u and ∇v in the face f.

- The parameterization is computed as a solution to the constrained minimization problem



(a) (b) (c)

Figure 3.12: The input quad layout induced by the polycube (a). The intermediate state of tracing iso-parameter curves from the quad layout nodes in an unconstrained parameterization to obtain the base complex (b). The base complex of a parameterization with node connection constraints; now the base complex is structurally equivalent to the input quad layout (c).

of the energy E_p (3.22):

$$\min_{(u,v)} \sum_{f \in F} \left[\|\nabla u - h\mathbf{u}^c\|^2 + \|\nabla v - h\mathbf{v}^c\|^2 \right] A_f, \tag{3.26}$$

where A_f is the area of the face f, and h is a scale factor that sets the correspondence between the length scale of the parametric domain and the surface.

The constraints imposed on (u, v) correspond to transitions across seams. We want the match across seams to be the same as for the guiding cross-field: if the $\mathbf{u}^c$ direction across a seam is transformed to a $\mathbf{v}^c$ direction, then the parametric directions are transformed in the same way. More precisely, if two faces f_i and f_j share a cut edge e_{ij}, with parametric positions of endpoint corners $\mathbf{p}_1^i$ and $\mathbf{p}_2^i$ on one side of the cut, and $\mathbf{p}_1^j$ and $\mathbf{p}_2^j$ on the other side, these are related by:

$$\mathbf{p}_1^j = R_{90}^{p_{ij}} \mathbf{p}_1^i + \mathbf{t}_e \text{ and } \mathbf{p}_2^j = R_{90}^{p_{ij}} \mathbf{p}_2^i + \mathbf{t}_e, \tag{3.27}$$

where R_{90} is a rotation by $\pi/2$, p_{ij} is the period jump of the cross-field along the edge e_{ij}, and $\mathbf{t}_e$ is an unknown translation. The constraints (3.27) are then incorporated as linear constraints into parameterization problem (3.26) using elimination of variables.

3.3.2 Arcs Embedding Optimization

A parameterization of this kind induces a base complex, which can be extracted by tracing iso-parametric curves (i.e., separatrices) from the nodes. With iso-parametric we mean that either the u or v parameter is constant among the curve when taking transitions into account. We now further constrain the parameterization problem (3.26) using node connection constraints, derived from the structure of the layout, to accomplish that this induced base complex is structurally equivalent to the quad layout $\mathcal{L}$. This ultimately allows us to derive an embedding for the quad layout $\mathcal{L}$ from the global parameterization.

We have to ensure for each arc of the quad layout $\mathcal{L}$ that the two incident nodes ($\mathbf{n}_1$ and $\mathbf{n}_2$) lie on a common iso-parametric curve. In other words, we want to change the

parameterization so that a parametric line starting at $\mathbf{n}_1$ passes through $\mathbf{n}_2$. If a mesh can be parametrized without seams, the requirement of nodes alignment easily translates into a constraint on parameterization: both nodes should be on the same parametric line, i.e., share the same u or v value. For parameterizations with cuts, the situation is more complicated. A parametric line on the surface undergoes a jump to a different point and direction in the parametric space when it crosses a cut. While the rotation is entirely determined by the cross field to which the parameterization is aligned, the positional jump depends on the parameterization itself. The resulting constraint will depend not only on the pair of nodes ($\mathbf{n}_1$, $\mathbf{n}_2$), but also on the variable translational parts $\mathbf{t}_e$ at the cut edges we cross. In the general case, consider a path crossing cut edges e_i, $i = 0, ..., m$ between nodes $\mathbf{n}_1$ and $\mathbf{n}_2$. Assuming the final direction of the path is $\lambda(u, v) \in \{u, v\}$, and that $\mathbf{p}_1(u_1, v_1)$ and $\mathbf{p}_2(u_2, v_2)$ are the parametric positions of $\mathbf{n}_1$ and $\mathbf{n}_2$ respectively, then the complete constraint has the form [Myl10]:

$$\left[\left(\prod_{i=0}^{m} R_{m-i}\right)\mathbf{p}_1 + \sum_{i=0}^{m-1}\left(\prod_{j=0}^{m-i-1} R_{m-j}\right)\mathbf{t}_{e_i} + \mathbf{t}_{e_m}\right]_{\lambda(u,v)} = \left[\mathbf{p}_2\right]_{\lambda(u,v)}, \tag{3.28}$$

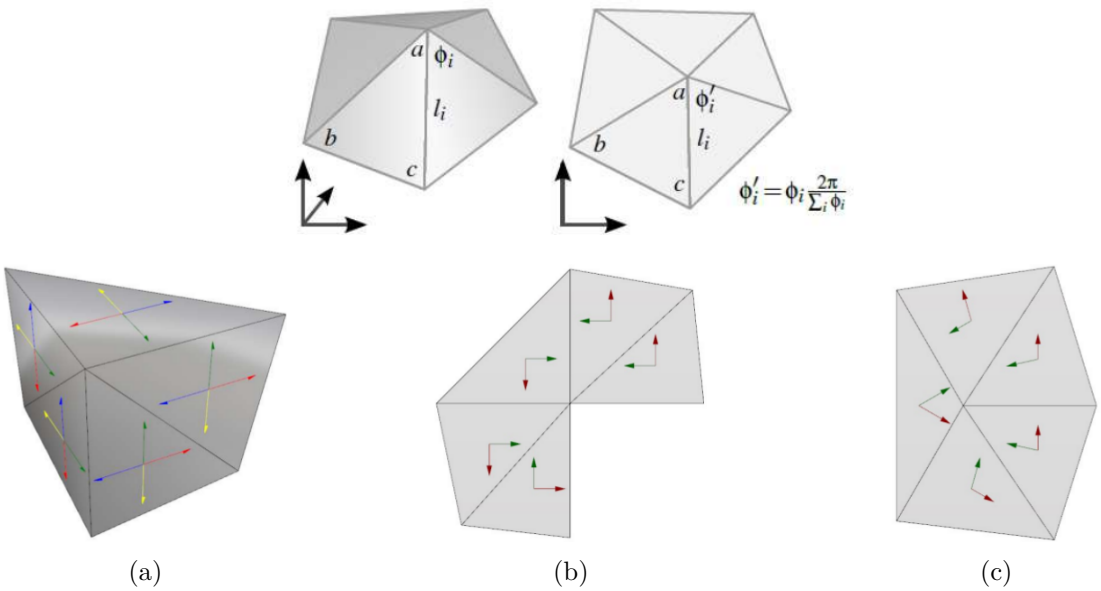
for either $\lambda(u, v) = u$ or $\lambda(u, v) = v$, where this subscript means taking the u or v coordinate.

3.3.3 Nodes Embedding Optimization

While the described constrained parameterization procedure optimizes the embedding of arcs and patches, the nodes remain fixed due to the very nature of the setup. This not only restrains the achievable quality, it further gives rise to large distortions or even local non-injectivities due to fold-overs. After computing a global parameterization, the quad layout's nodes are re-positioned based on the gradient of the parameterization's objective functional with respect to their positions. This is done iteratively until a local optimum of global embedding quality is reached. Campen et al. [Cam14] described a method to perform this re-positioning using an easy-to-implement estimator of the objective functional's true gradient.They advocated a strategy to move the nodes so as to arrive at a solution with lower residual. Thus, while taking care of the distortion problems, this strategy simultaneously optimizes the nodes' embedding, basically by exploiting the information the distortion provides.

Technically speaking, we are going to optimize (3.26) not only w.r.t. the parameters u and v (thus the arcs' and patches' embedding) but also w.r.t. the geometric positions of quad layout's nodes on the mesh M. We tackle this non-linear problem using a strategy which optimizes (3.26) A) with respect to u and v (with fixed nodes) and B) with respect to the node positions (with fixed u and v) in an alternating manner. In this way the large problem A ($O(\mid M \mid)$ variables) can still be solved as a simple linear problem as described in the previous section. The smaller non-linear problem B ($O(\mid \mathcal{L} \mid)$ variables), is addressed using a gradient descent strategy. In order to locally move a node $\mathbf{n}$ in such a way that the residual of the energy E_p (3.21) decreases, we need to move this node in a gradient descent manner in direction

$$\mathbf{d}(\mathbf{n}) = -\left(\frac{\partial}{\partial n_\xi} E(n_\xi, n_\eta), \frac{\partial}{\partial n_\eta} E(n_\xi, n_\eta)\right), \tag{3.29}$$



(a) (b) (c)

Figure 3.13: The physical embedding of a vertex (a), its isometric map (b), and its geodesic polar map (c).

where (ξ, η) are 2D coordinates in some local coordinate chart of the mesh M around the node $\mathbf{n}$ and (n_ξ, n_η) expresses the current position of node $\mathbf{n}$ accordingly. Note the E depends on n_ξ and n_η because nodes are embedded in vertices (i.e., node positions are vertex positions).

The gradient (3.29) of energy (3.21) depends on n_ξ and n_η in multiple ways: they appear directly in the discrete gradient operator ∇ and in A_t, but indirectly also in the cross field (i.e., in $\mathbf{u}^c$ and $\mathbf{v}^c$), as well as in the parameterization (u, v). If we neglect these indirect dependencies, i.e., consider $\hat{E}$ where $\mathbf{u}^c$ and $\mathbf{v}^c$ as well as (u, v) are fixed, we can analytically derive:

$$\frac{\partial}{\partial n_\xi}\hat{E} = \sum_{t \in T(\mathbf{n})} \left[2 \left(\frac{\partial}{\partial n_\xi} \nabla_t u \right)^T (\nabla_t u - \mathbf{u}_t^c)\, A_t + \frac{\partial A_t}{\partial n_\xi} \parallel \nabla_t u - \mathbf{u}_t^c \parallel^2 \right.$$
$$\left. + 2 \left(\frac{\partial}{\partial n_\xi} \nabla_t v \right)^T (\nabla_t v - \mathbf{v}_t^c)\, A_t + \frac{\partial A_t}{\partial n_\xi} \parallel \nabla_t v - \mathbf{v}_t^c \parallel^2 \right]. \quad (3.30)$$

$$\frac{\partial}{\partial n_\eta}\hat{E} = \sum_{t \in T(\mathbf{n})} \left[2 \left(\frac{\partial}{\partial n_\eta} \nabla_t u \right)^T (\nabla_t u - \mathbf{u}_t^c)\, A_t + \frac{\partial A_t}{\partial n_\eta} \parallel \nabla_t u - \mathbf{u}_t^c \parallel^2 \right.$$
$$\left. + 2 \left(\frac{\partial}{\partial n_\eta} \nabla_t v \right)^T (\nabla_t v - \mathbf{v}_t^c)\, A_t + \frac{\partial A_t}{\partial n_\eta} \parallel \nabla_t v - \mathbf{v}_t^c \parallel^2 \right]. \quad (3.31)$$

Note that the sum is only over triangles $T(\mathbf{n})$ incident to node $\mathbf{n}$, as all other terms vanish due to independence from n_ξ and n_η. The corresponding approximate gradient $\hat{\mathbf{d}}(\mathbf{n})$ can thus be very efficiently evaluated based on only the 1-ring neighborhood.

In order to compute the gradient descent vector $\hat{\mathbf{d}}$ for a node curretly embedded in vertex $\mathbf{a}$, we first need to obtain a local 2D coordiante system of $\mathbf{a}$'s 1-ring. To this end we employ the commonly used geodesic polar map [Wel94], effectively flattening the 1-ring to the plane while preserving radial lengths (l_i) and relative angles by uniformly scaling the inner angles ϕ_i incident to $\mathbf{a}$ such that they sum to 2π; origin and axes in the plane can be chosen arbitrarily (Figure 3.13).

Let (a_ξ, a_η), (b_ξ, b_η) and (c_ξ, c_η) denote the 2D coordinates of a triangle t's vertices in this system, and (a_u, a_v), (b_u, b_v) and (c_u, c_v) their current (u, v) parameters. $\mathbf{u}_t^c$ and $\mathbf{v}_t^c$ are the first and second cross field vectors in t (expressed in the 2D system). The per triangle gradient $(\frac{\partial}{\partial a_\xi}\hat{E}_t, \frac{\partial}{\partial a_\eta}\hat{E}_t)^T$ at center vertex $\mathbf{a}$ in triangle t can then be computed based on well-known expressions for triangle area A and triangle gradient ∇. The expressions related to the triangle's area A are given by:

$$A = \frac{1}{2}\left[a_\xi\left(b_\eta - c_\eta\right) + b_\xi\left(c_\eta - a_\eta\right) + c_\xi\left(a_\eta - b_\eta\right)\right], \tag{3.32}$$

$$\frac{\partial}{\partial a_\xi}A = \frac{1}{2}\left(b_\eta - c_\eta\right) \;;\; \frac{\partial}{\partial a_\eta}A = \frac{1}{2}\left(c_\xi - b_\xi\right). \tag{3.33}$$

The expressions related to the triangle's gradient ∇ are given by:

$$\mathbf{H}_u = a_u\left(b_\eta - c_\eta, c_\xi - b_\xi\right)^T + b_u\left(c_\eta - a_\eta, a_\xi - c_\xi\right)^T + c_u\left(a_\eta - b_\eta, b_\xi - a_\xi\right)^T, \tag{3.34}$$

$$\mathbf{H}_v = a_v\left(b_\eta - c_\eta, c_\xi - b_\xi\right)^T + b_v\left(c_\eta - a_\eta, a_\xi - c_\xi\right)^T + c_v\left(a_\eta - b_\eta, b_\xi - a_\xi\right)^T, \tag{3.35}$$

$$\nabla u = \frac{1}{2A}\mathbf{H}_u \;;\; \nabla v = \frac{1}{2A}\mathbf{H}_v, \tag{3.36}$$

$$\frac{\partial}{\partial a_\xi}\nabla u = \frac{1}{2A}\left(0, b_u - c_u\right)^T - \frac{1}{2A^2}\mathbf{H}_u\frac{\partial}{\partial a_\xi A}, \tag{3.37}$$

$$\frac{\partial}{\partial a_\eta}\nabla u = \frac{1}{2A}\left(c_u - b_u, 0\right)^T - \frac{1}{2A^2}\mathbf{H}_u\frac{\partial}{\partial a_\eta A}, \tag{3.38}$$

$$\frac{\partial}{\partial a_\xi}\nabla v = \frac{1}{2A}\left(0, b_v - c_v\right)^T - \frac{1}{2A^2}\mathbf{H}_v\frac{\partial}{\partial a_\xi A}, \tag{3.39}$$

$$\frac{\partial}{\partial a_\eta}\nabla v = \frac{1}{2A}\left(c_v - b_v, 0\right)^T - \frac{1}{2A^2}\mathbf{H}_v\frac{\partial}{\partial a_\eta A}, \tag{3.40}$$

The final gradient descent vector $\hat{\mathbf{d}}(\mathbf{a})$ is then computed by summation over the triangles $T(\mathbf{a})$ incident to $\mathbf{a}$:

$$\hat{\mathbf{d}}(\mathbf{a}) = -\left(\frac{\partial}{\partial a_\xi}\hat{E}, \frac{\partial}{\partial a_\eta}\hat{E}\right)^T = -\sum_{t\in T(\mathbf{a})}\left(\frac{\partial}{\partial a_\xi}\hat{E}_t, \frac{\partial}{\partial a_\eta}\hat{E}_t\right)^T. \tag{3.41}$$

Conclusion

Aligned global parameterizations adapt the parameterization to the geometry of the surface by fitting the parameterization gradient to a smooth cross field interpolating principal directions and geometric features of the surface. Interestingly, each of the required properties for a good parameterization (uniformity, orthogonality and singularities) can be redefined in terms of desired properties of the cross field. Thus the task is shifted from the definition of a good parameterization to the definition of a good cross field on the surface.

The anisotropy-adapted direction field is represented by a cross field, i.e., a set of four orthogonal directions at each point of the surface interpolating principal curvature directions and geometrical features (e.g. boundaries, sharp features). The cross field's orthogonal property will ensure that the iso-parametric structure is as orthogonal as possible. In addition, the quality of the quadrilateral mesh is mainly affected by the distribution of singularities. To this end, the structure of the generated polycube in the previous chapter (Chapter 2) will be used. Topologically, the guiding cross field is conform with the polycube's structure. In other words, the cross field is singular only at the position of irregular nodes of the polycube. The novelty of the proposed method is the design of a cross field satisfying both topological and geometrical constraints by only solving two linear systems.

The parameterization is found by a constrained global minimization of an energy functional using functions of the actual parameters u and v (coordinates in the parametric space) of the surface. The parameterization should be locally oriented according to the guiding direction field. This implies that the gradients ∇u and ∇v of the piecewise linear scalar fields u and v should follow the cross field directions $\mathbf{u}^c$ and $\mathbf{v}^c$. A scale factor that sets the correspondence between the length scale of the parametric domain and the surface is integrated in the energy functional. This factor will ensure that the quadrilateral elements are as uniform as possible. The parameterization is constrained by the polycube structure. While the described constrained parameterization optimizes the embedding of the polycube's arcs and patches, its nodes remain fixed. This may give rise to large distortions or even local non-injectivities due to fold-overs. A practical solution to this problem is to re-position the polycube's nodes based on the gradient of the parameterization's objective functional with respect to their positions, so as to arrive at a local optimum of global embedding quality.

The approach of using the polycube's structure to constrain the parameterization is inherently volumetric. The polycube decomposes the solid model into hexahedral domains, i.e., cuboids. By computing an aligned global surface parameterization between the cuboids' boundaries and the polycube's boundary, the volume parameterization can be trivially computed just by gridding the interior of each cuboid.

Extraction of spline surfaces and volumes from the computed aligned global parameterization is presented in the next chapter (Chapter 4). Using the quadrilateral mesh extracted from the optimized aligned global parameterization, a structured point grid is generated and used to fit the boundary spline surfaces. The spline volume is then obtained using the reconstructed spline surfaces as boundary conditions.

Chapter 4

Applications

Introduction

As it is pointed by Cottrell et al. [Cot09], one of the most significant challenges towards IGA is constructing analysis-suitable parameterizations from models given in boundary representations. In the context of this thesis, a potential solution to this problem is obtained by the following strategy. The input is the model's boundary triangle mesh. This triangle mesh is approximated by a polycube (Chapter 2). Due to its regular structure, the polycube serves as the parametric domain required for tensor-product trivariate splines fitting. The parameterization between the boundary triangle mesh and the boundary of the polycube is computed using global parameterization aligned to a cross field. The cross field is topologically conform with the polycube structure, and geometrically interpolates the model's geometric features (Chapter 3). The approach of using the polycube's structure to compute a parameterization is inherently volumetric. Spline surfaces and volumes construction is presented in Section 4.1.

Moreover isogeometric representation reduces the number of parameters needed to describe the geometry, which is of particular interest for shape optimization and analysis. In addition, it can be applied to reduced order modeling for parametric studies based on geometrical parameters. For models with the same topology but different geometries, this method allows to have the same isotopological representation required for ROM construction (in order to avoid a projection step). An application is presented in Section 4.2.

4.1 Analysis-Suitable Trivariate Models

Using the quadrilateral mesh extracted from the optimized aligned global parameterization, a structured grid of points is generated on each patch, and then used to fit the boundary B-Spline surfaces. For each cuboid, the B-Spline solid is obtained using the reconstructed B-Spline surfaces as boundary conditions. Keeping the boundary control points fixed, the interior control points of the B-Spline solid are computed using Coons' interpolation. The positions of the interior control points are then adjusted by minimizing a Laplacien based energy.

4.1.1 Spline Surfaces Reconstruction

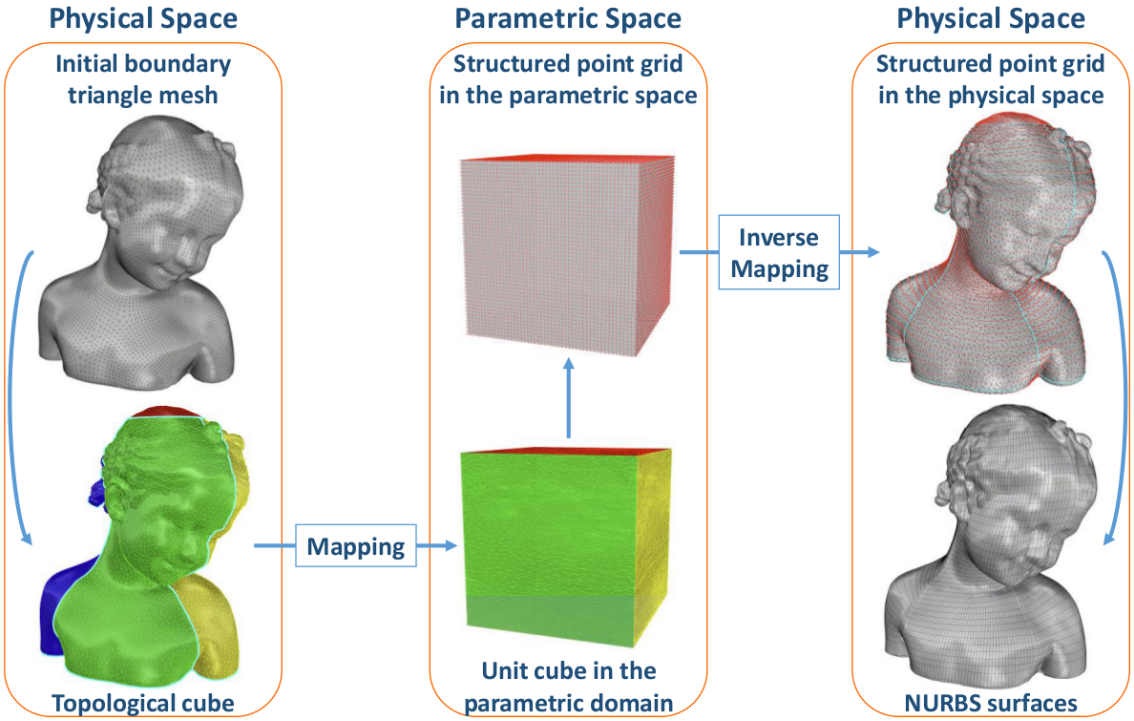


Figure 4.1: B-Spline Surfaces reconstruction.

Let $S(u, v)$ be a B-Spline surface. A structured point grid is extracted from the parameterization on each cuboid, and then used to reconstruct the boundary B-Spline surfaces. For each cuboid, six compatible B-Spline boundary surfaces with optional $G1$ smoothness constraint are reconstructed as follows:

$$\min_{X,Y,Z} \sum_{i=1}^{6} \left[\lambda_1^i \cdot E_i^{sretching} + \lambda_2^i \cdot E_i^{bending} + \lambda_3^i \cdot E_i^{error}\right], \tag{4.1}$$

$$\text{subject to } C_1 \cdot X = 0,\ C_2 \cdot Y = 0,\ C_3 \cdot Z = 0, \tag{4.2}$$

$$\text{with } E_i^{sretching} = \int_0^1 \int_0^1 \left(\|S_{i,u}\|^2 + \|S_{i,v}\|^2\right) du \cdot dv, \tag{4.3}$$

$$\text{and } E_i^{bending} = \int_0^1 \int_0^1 \left(\|S_{i,uu}\|^2 + 2\|S_{i,uv}\|^2 + \|S_{i,vv}\|^2\right) du \cdot dv, \tag{4.4}$$

where X, Y, Z are the vectors of the coordinates of the surface control points, and E_i^{error} is the deviation from the fitted B-spline surface to the boundary triangle mesh, calculated by the method presented by Ma et al. [Ma95]. A larger λ_1^i and λ_2^i will make the surface smoother and suppress mesh folding, and a larger λ_3^i will reduce the fitting error. The constraints (4.2) represent simplified G1 linear constraints presented by Shi et al. [Shi04] that ensure G1 continuity across both the shared edges and corners. In the context of this thesis, the software *Rhinoceros* [Rhia] was used for B-Spline surfaces fitting.

Figure 4.1 illustrates the method for B-Spline surfaces reconstruction. The input is a boundary triangle mesh topologically equivalent to a cube. Using aligned global parameterization, a parametric mapping between the boundary triangle mesh and the boundary of the unit cube in the parametric domain is built. A regular and structured points grid on the parametric cube is generated, and then mapped to the original boundary triangle mesh using barycentric coordinates. The structured points grid in the physical space is used to reconstruct 6 compatible B-Spline surfaces.

4.1.2 Spline Volumes Reconstruction

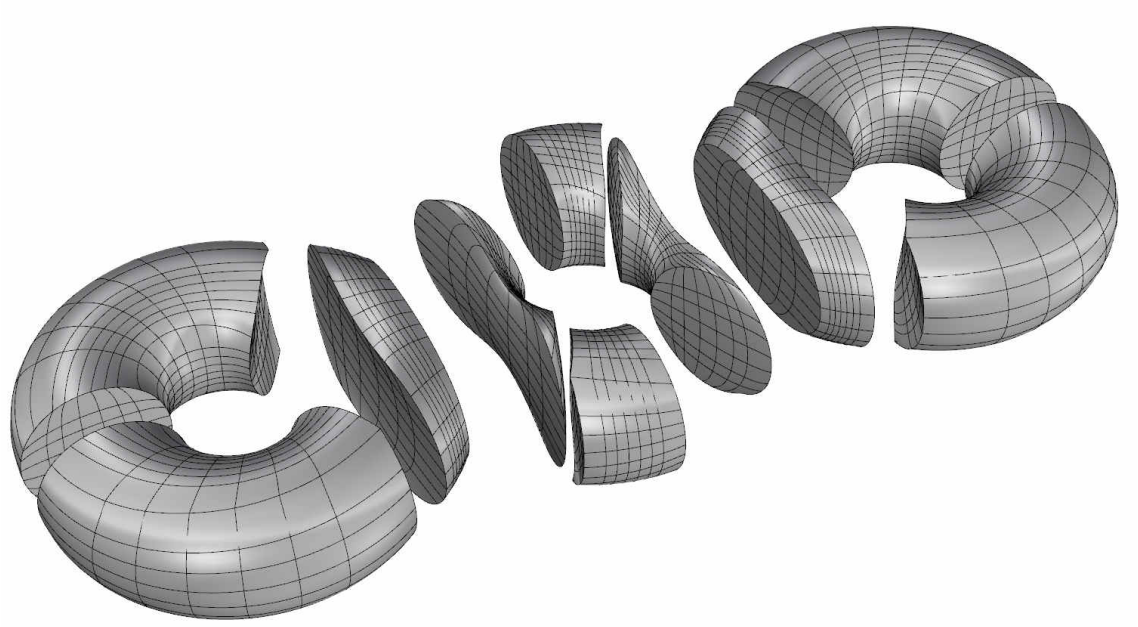


Figure 4.2: B-Spline solid reconstruction of a 3-torus: exploded view with some cuboids removed to show the interior parameterization.

For each cuboid, the B-Spline solid is obtained using the reconstructed B-Spline surfaces as boundary conditions. Keeping the boundary control points fixed, the interior control points of the B-Spline solid are computed using Coons' interpolation. The inputs are six valid and compatible B-Spline boundary surfaces, $S_1(u,v)$, $S_2(u,v)$, $S_3(u,w)$, $S_4(u,w)$, $S_5(v,w)$, and $S_6(v,w)$. Let $\psi_0^u = 1-u$, $\psi_0^v = 1-v$, $\psi_0^w = 1-w$,$\psi_1^u = u$, $\psi_1^v = v$, $\psi_1^w = w$, then the Coons solid $V(u,v,w)$ is defined as follows [Wan14]:

$$\begin{aligned} V(u,v,w) &= \psi_0^w S_1(u,v) + \psi_1^w S_2(u,v) + \psi_0^v S_3(u,w) + \psi_1^v S_4(u,w) + \psi_0^u S_5(v,w) \\ &+\psi_1^u S_6(v,w) - [\psi_0^v\psi_0^w S_1(u,0) + \psi_1^v\psi_0^w S_1(u,1) + \psi_0^v\psi_1^w S_2(u,0) + \psi_1^v\psi_1^w S_2(u,1) \\ &+\psi_0^u\psi_0^w S_1(0,v) + \psi_1^u\psi_0^w S_1(1,v) + \psi_0^u\psi_1^w S_2(0,v) + \psi_1^u\psi_1^w S_2(1,v) + \psi_0^u\psi_0^v S_3(0,w) \\ &+\psi_1^u\psi_0^v S_3(1,w) + \psi_0^u\psi_1^v S_4(0,w) + \psi_1^u\psi_1^v S_4(1,w)] + [\psi_0^u\psi_0^v\psi_0^w S_1(0,0) \\ &+\psi_1^u\psi_0^v\psi_0^w S_1(1,0) + \psi_0^u\psi_1^v\psi_0^w S_1(0,1) + \psi_1^u\psi_1^v\psi_0^w S_1(1,1) + \psi_0^u\psi_0^v\psi_1^w S_2(0,0) \\ &+\psi_1^u\psi_0^v\psi_1^w S_2(1,0) + \psi_0^u\psi_1^v\psi_1^w S_2(0,1) + \psi_1^u\psi_1^v\psi_1^w S_2(1,1)]. \end{aligned} \tag{4.5}$$

This Coons volume is then used to generate the initial internal control points of the B-spline solid. We uniformly distribute the parameters u, v, w and evaluate the equation (4.5) to obtain a set of inner points of the Coon's volume $T_c\left(\frac{i}{n}, \frac{j}{m}, \frac{k}{l}\right)$, $i = 1, \ldots, n-1$, $j = 1, \ldots, m-1$, $k = 1, \ldots, l-1$.

We then adjust the positions of the interior control points by minimizing the Laplacian based energy given by:

$$E(p_{i,j,k}) = \sum_{\lambda \in N_{i,j,k}} w_\lambda \cdot \|p_{i,j,k} - p_\lambda\|, \tag{4.6}$$

where $p_{i,j,k}$ is an internal control point, $N_{i,j,k}$ is the set of indices of control points adjacent to $p_{i,j,k}$, and w_λ is a weight. In our implementation we simply use the uniform weight $w_\lambda = 1/6$. We move the grid points iteratively and we stop when changes of all nodes are smaller than a given threshold.

4.1.3 Trivariate Models Examples

The software *Rhinoceros 5* [Rhia] is used for visualisation, and *RhinoCommon* [Rhib] for NURBS manipulations. In addition, for solving the various sparse linear systems, the SuperLU [Dem99], SuiteSparseQR [Dav08], CHOLMOD [Che08], and CoMISo [Bom12b] solvers are used. Figures 4.3-4.4 demonstrate the method for models with high genus, Figure 4.5 for models with sharp features, and Figures 4.6-4.7 for models with boundaries.

The "Tetra" and "Block" models are courtesy of the AIM@ShapeProject. We thank Aline Brunon (LaMCoS) for her help in providing the "Bust" model, and Daniel Silva Soto (University of Sheffield) for his help in providing the "Aortic Cross" model.

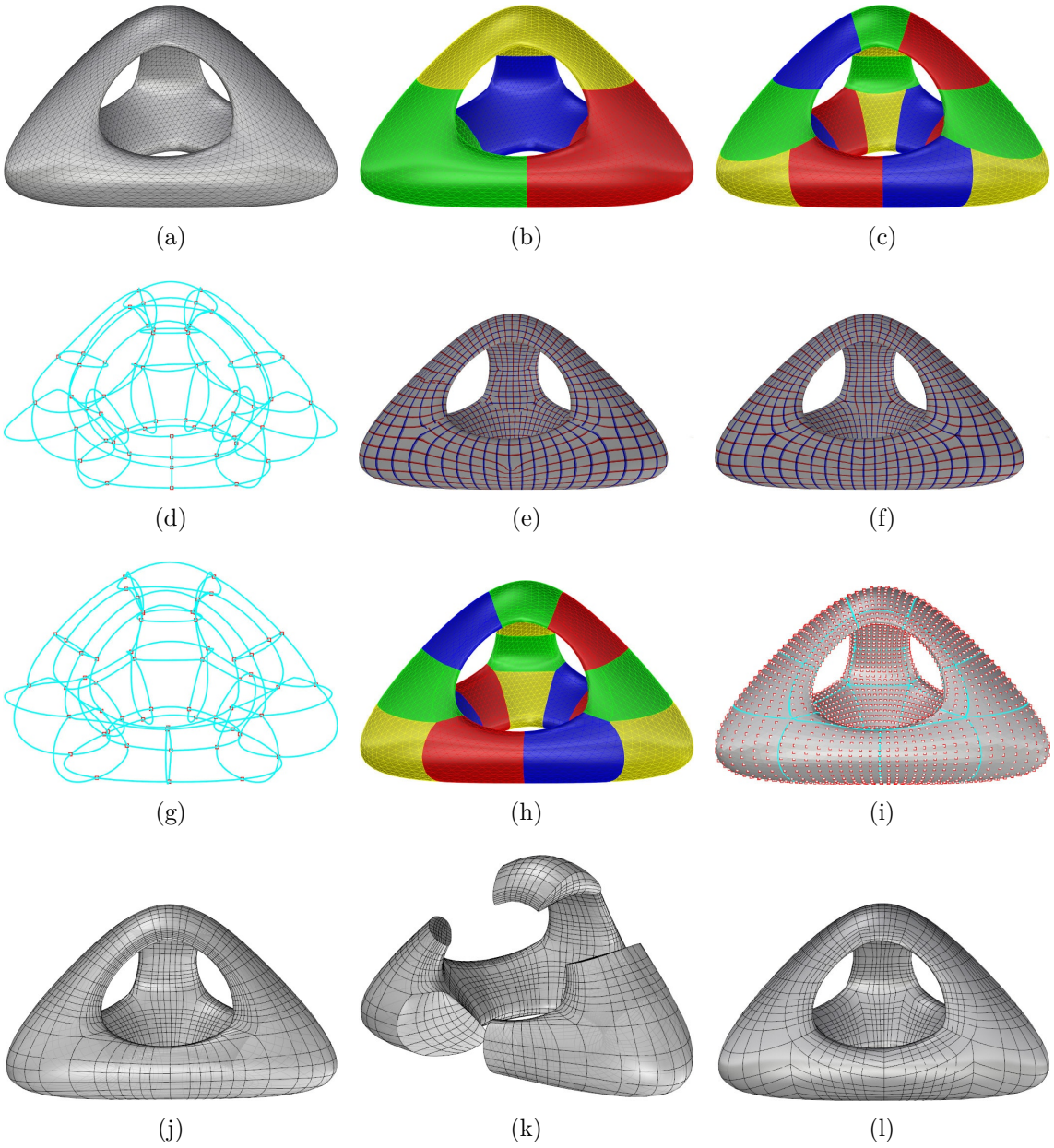


Figure 4.3: Tetra model: input boundary triangle mesh (a); initial pants decomposition (b); initial (c) and optimized (h) cuboid decomposition; initial (d) and optimized (g) polycube embedding; initial (e) and optimized (f) aligned global parameterization; extracted structured point grid (i); reconstructed B-Spline boundary surfaces (j); reconstructed B-Spline solid (k). Reconstructed B-Spline boundary surfaces using local harmonic parameterization: notice the tangential discontinuity of the isoparametric lines across different patches (l).

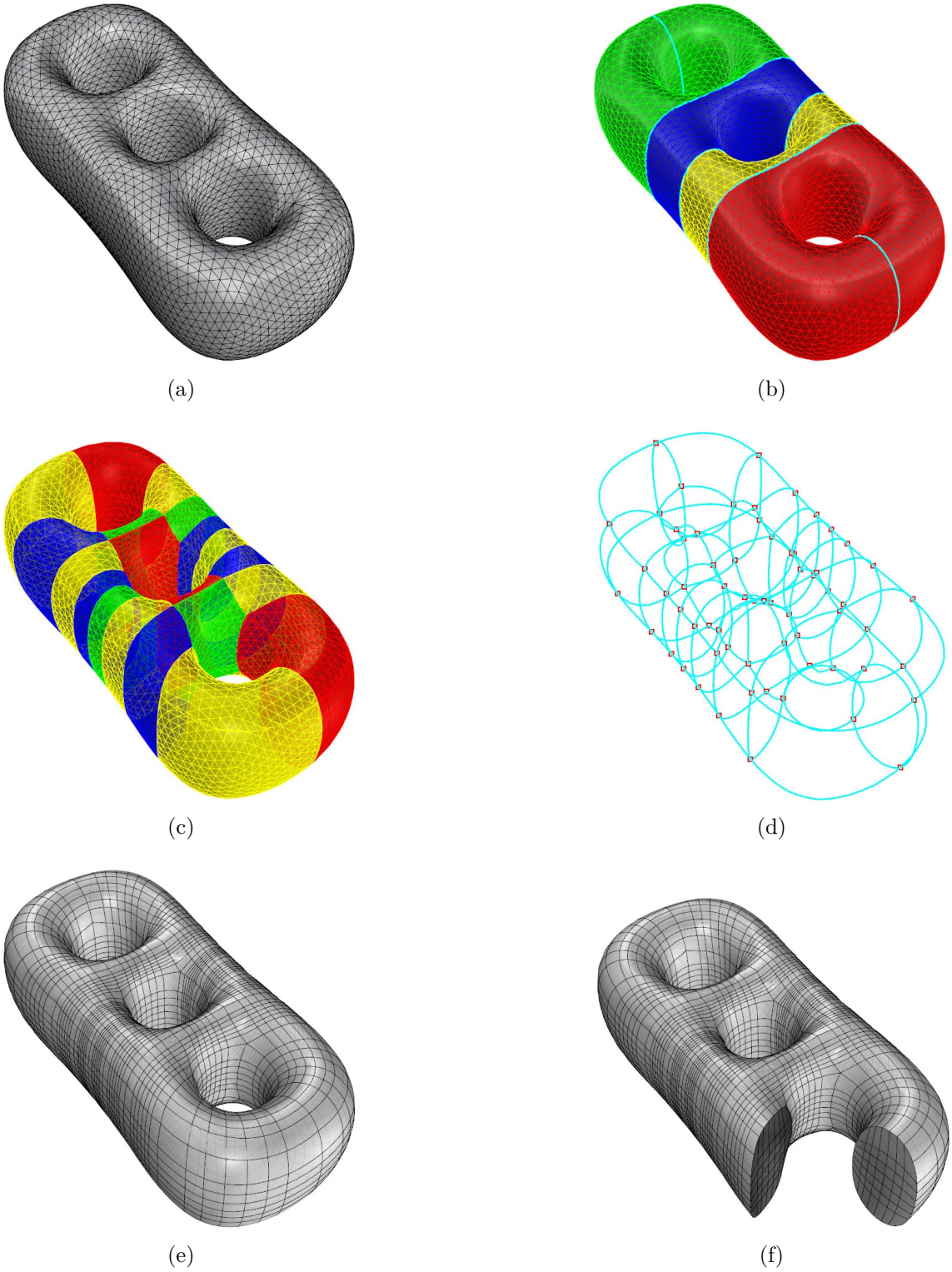


Figure 4.4: Block model: input triangulated boundary mesh (a); pants decomposition (b); cuboid decomposition (c); optimized polycube embedding (d); B-spline boundary surfaces (e); output B-spline volume (f).

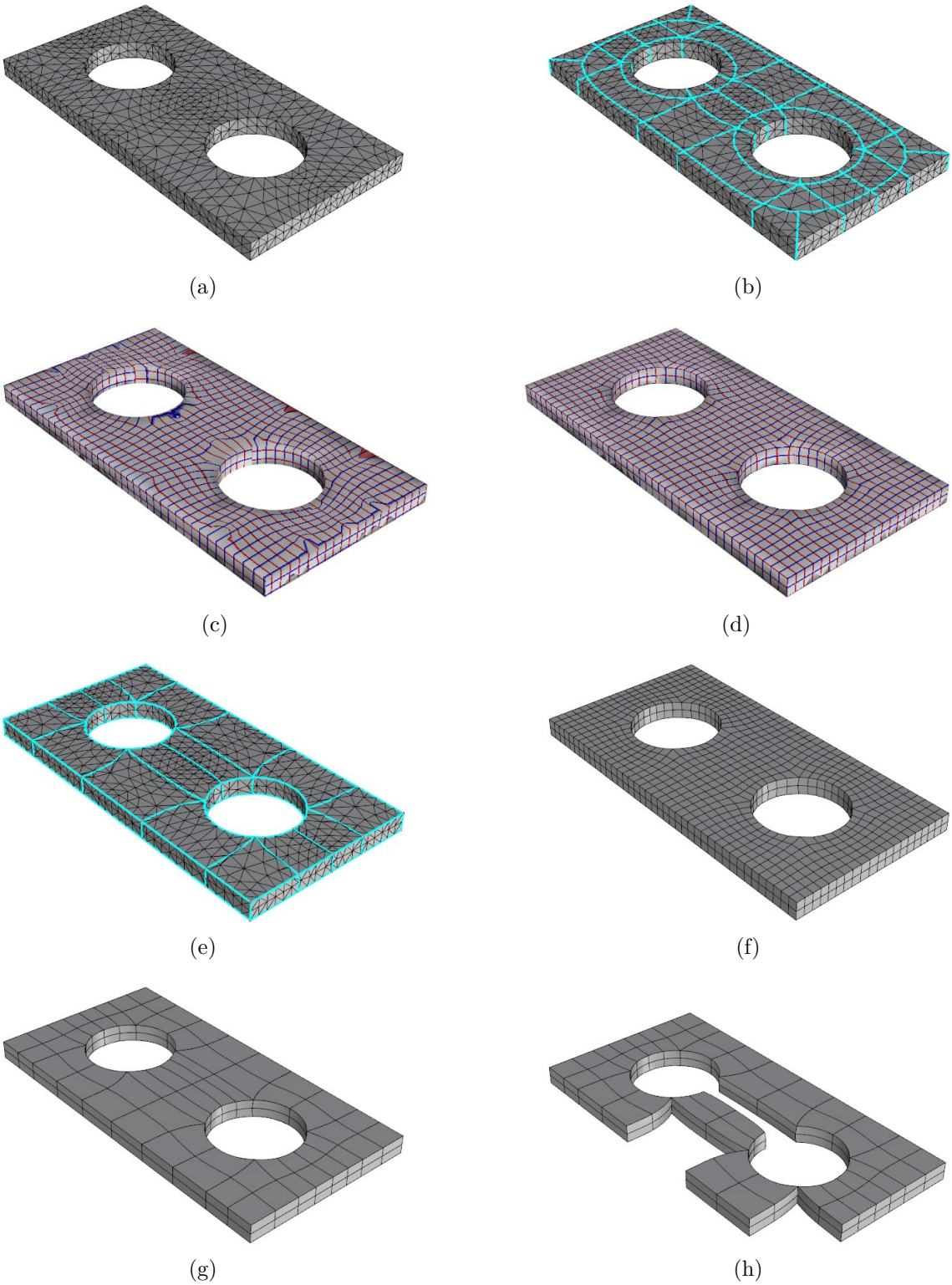


Figure 4.5: Plate model: boundary triangle mesh (a); initial (b) and optimized (e) polycube embedding; initial (c) and optimized (d) aligned global parameterization; extracted quad mesh (f); B-Spline boundary surfaces (g); B-Spline solid (h).

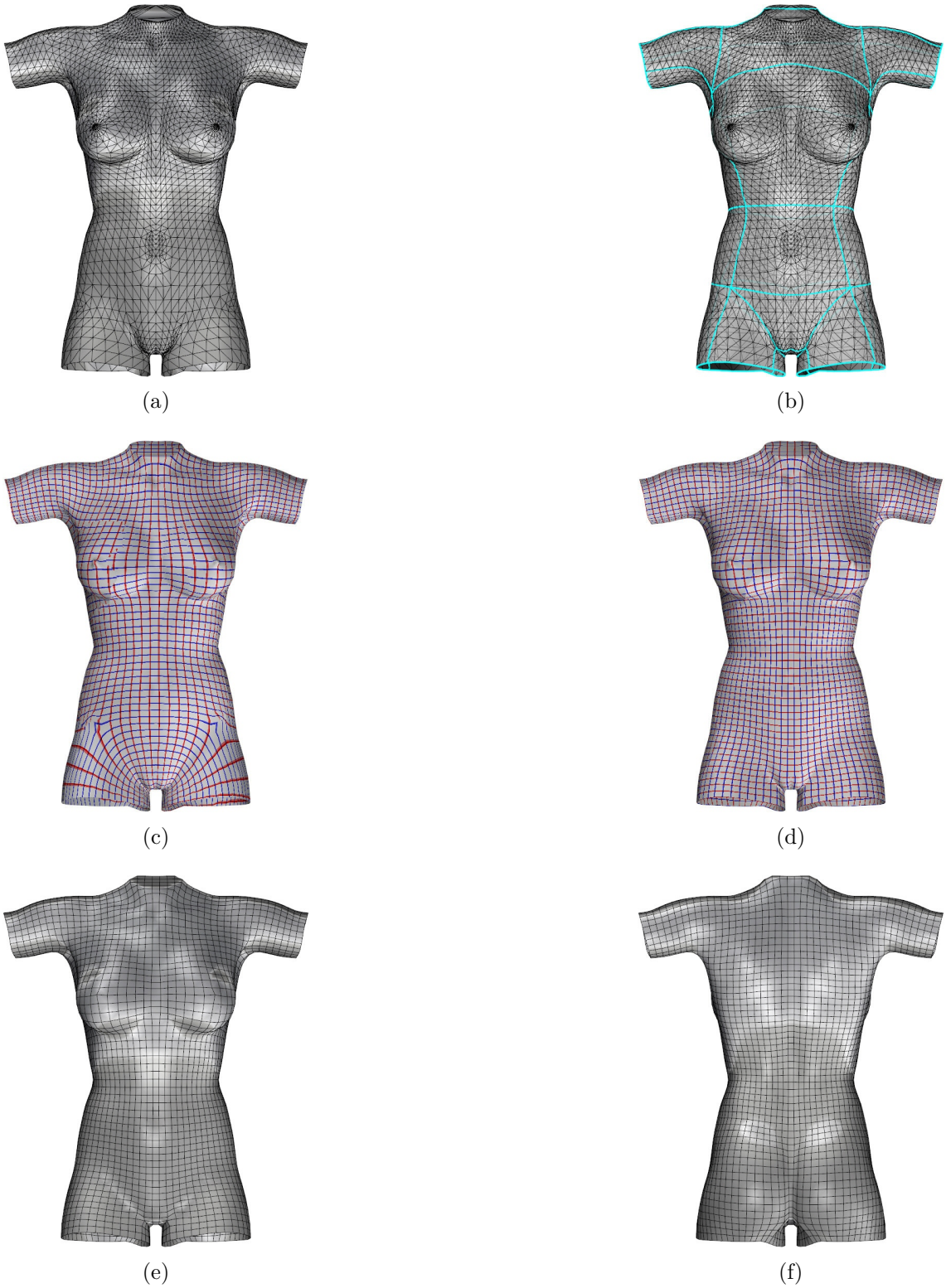


Figure 4.6: Bust model: boundary triangle mesh (a); initial polycube embedding (b); initial (c) and optimized (d) aligned global parameterization; extracted quad mesh (e-f).

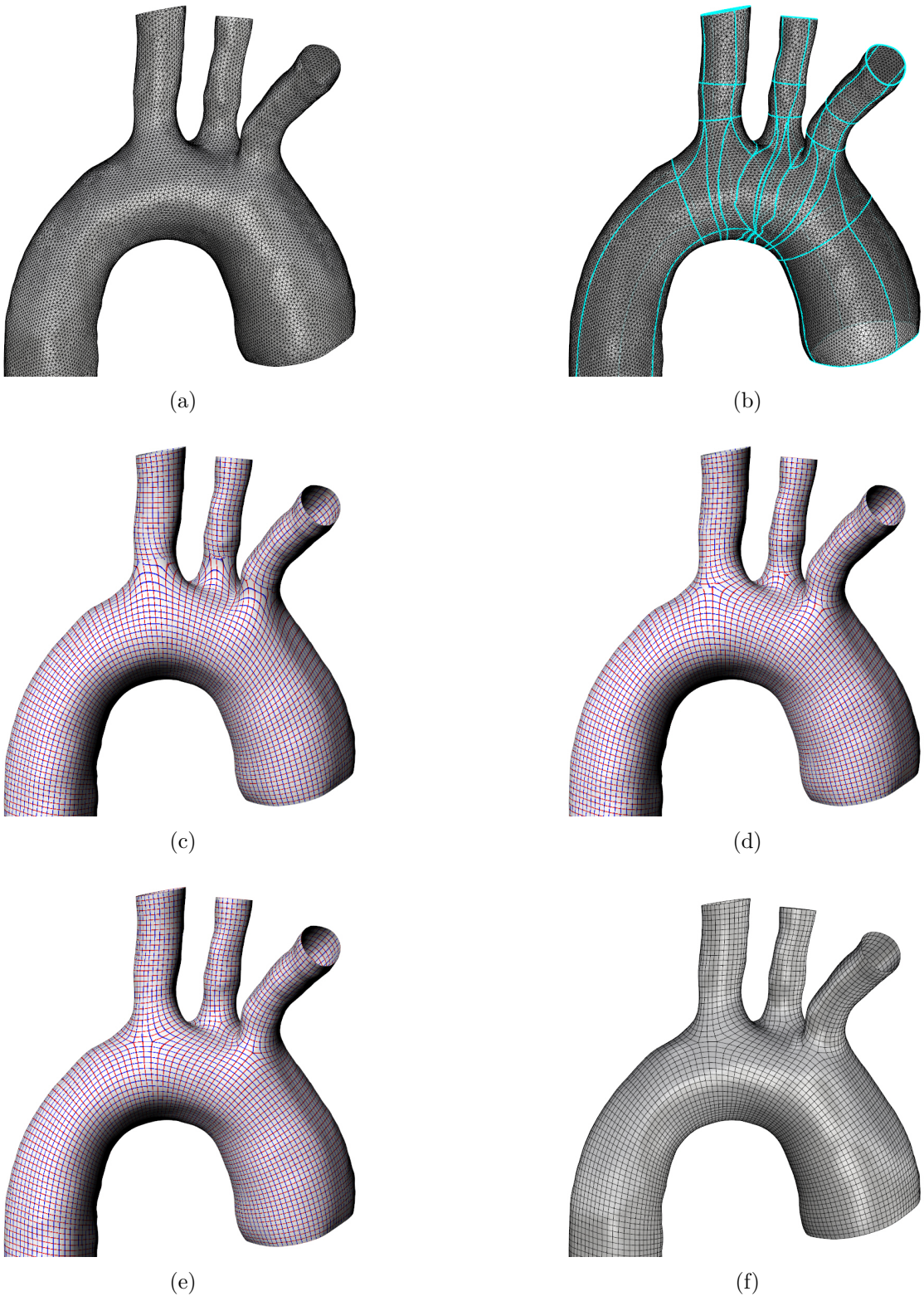


Figure 4.7: Aortic-cross model: boundary triangle mesh (a); initial polycube embedding (b); initial (c), intermediate (d) and optimized (e) aligned global parameterization; extracted quad mesh (f).

4.2 Statistical Shape Analysis

Statistical shape analysis is an analysis of the geometrical properties of some set of shapes by statistical methods. It has applications in various fields including medical imaging. For instance, it could be used to quantify differences between normal and pathological vessel, bone or organ shapes. Important aspects of shape analysis are to obtain a measure of distance between shapes, to estimate mean shapes from (possible random) samples, to estimate shape variability within samples, to perform clustering and to test for differences between shapes [Zie94; Dry98]. One of the main methods used is principal component analysis. Such techniques require the creation of a proper isotopological shape model for all members of the population. An isogeometric representation over all shapes can be a useful tool. This technique is going to be illustrated on a database of abdominal aortas that present an aneurysm.

4.2.1 Abdominal Aortic Aneurysm

The aorta is the main blood vessel that supplies blood to the abdomen, pelvis, and legs. Abdominal Aortic Aneurysm (AAA) is a localized dilatation of abdominal aorta that exceeds standard diameter length by more than 50%. It is the most prevalent type of aortic aneurysm.

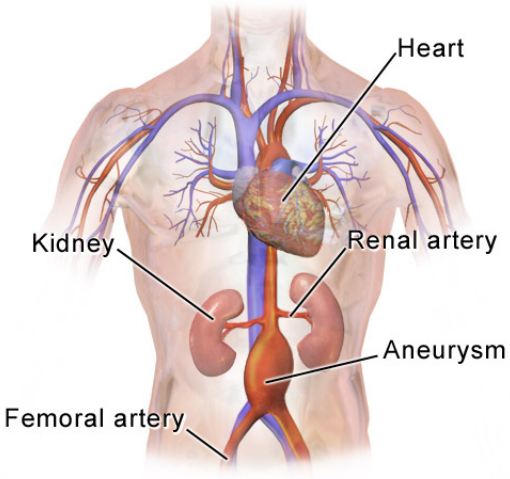


Figure 4.8: The aorta is the largest artery in the body. An abdominal aortic aneurysm is a ballooning of that large artery in the abdomen. From [Med16].

EndoVascular Aneurysm Repair (EVAR) is a type of endovascular surgery used to treat pathology of the aorta, most commonly an AAA. The procedure involves the placement of an expandable stent graft within the aorta to treat aortic disease avoiding open surgery. Performing statistical shape analysis on a population of pathological aortas can have many advantages:

- It allows to define a metric on the human variability. Hence for each new patient, the closest patient in the existing database can be identified. This allows the clinician to get

information corresponding to the closest patient: stent dimensions, eventual problems which occurred during or after the intervention, etc.

- It gives access to the population variability and can be used to make sure that the set of existing stents covers all the potential patients (useful for stent design and manufacturing).

- It allows to generate "virtual patients". For instance, virtual patients which are on the limits of the population variability can be generated in order to train surgeons.

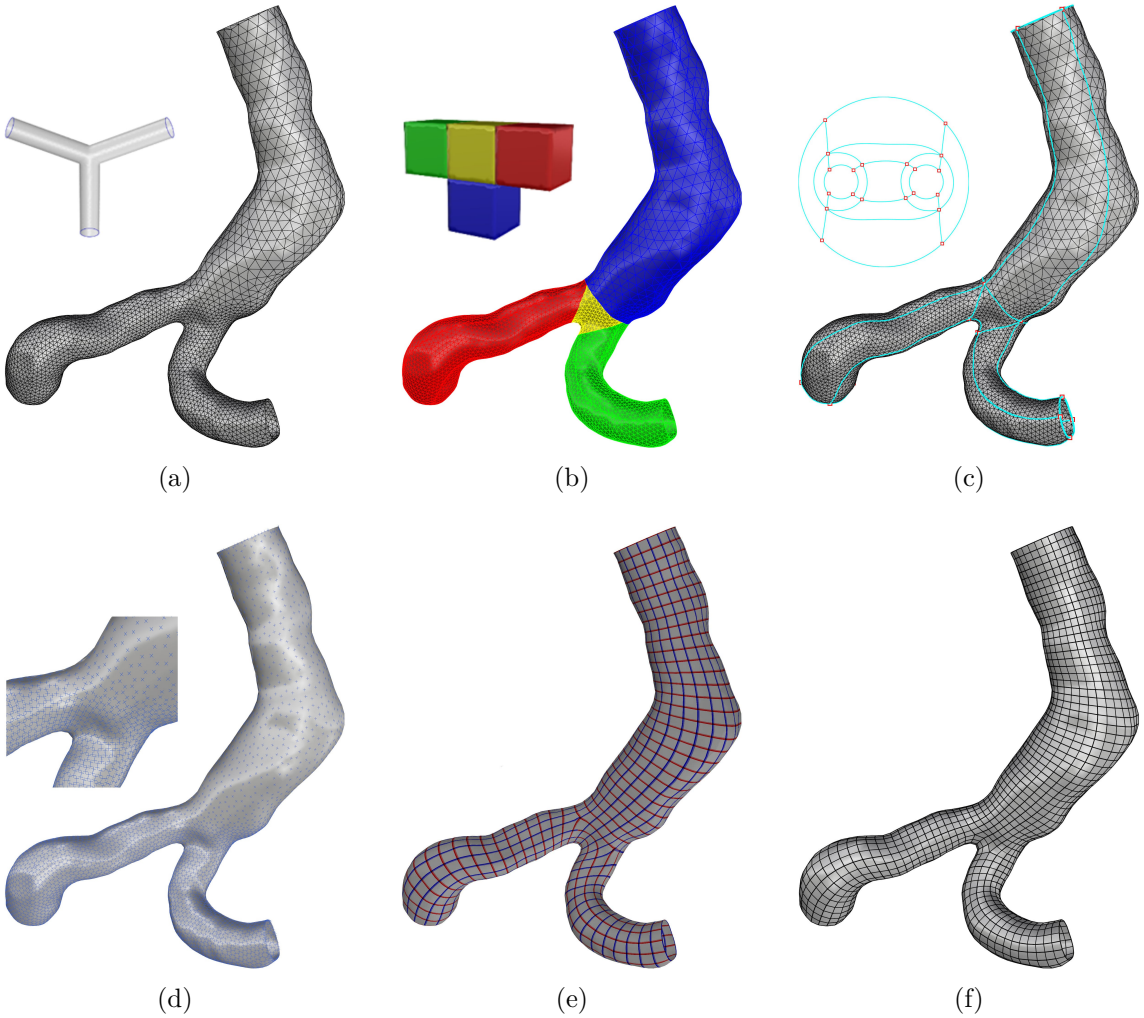
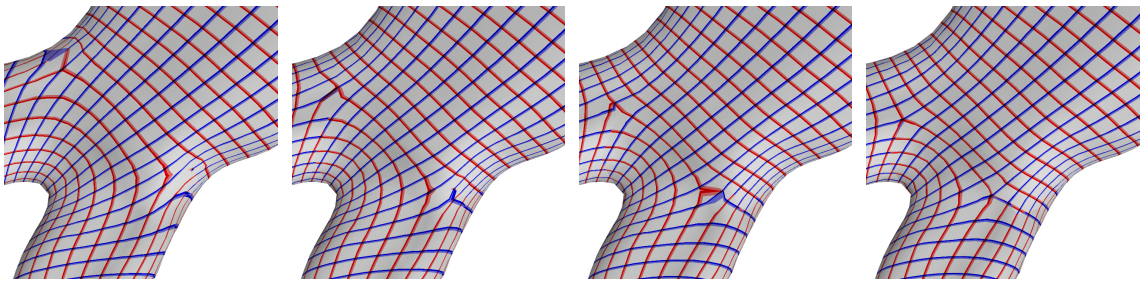


Figure 4.9: Algorithm overview. The input triangulated boundary surface homeomorphic to a pants patch (a). The polycube generated from cuboid decomposition (b). The embedding of the quad layout induced by the polycube (c). The cross field topologically conform with the quad layout, and geometrically aligned with the surface principal directions and boundaries (d). The global parameterization aligned with the cross field (e). The control quad mesh extracted from the global aligned parameterization (f).

4.2.2 Isotopological Representations

The input data is a database of 90 aortas' meshes. These surface triangle meshes are obtained from preliminary analysis of preoperative CT scans (using the software EndoSize, Therenva). This analysis is conducted by the vascular surgery department of the CHU (Centre Hospitalier Universitaire, Rennes, France) and the LTSI (INSERM U1099, Université de Rennes 1, France) as part of a research protocol on EVAR interventions. The study protocol was approved by the CHU's ethics committee and patient consents were obtained before including their anonymous data into the database.

Each input triangle mesh is homeomorphic to a pants patch. Using the cuboid decomposition algorithm, a polycube approximating the input mesh is generated. The quad layout induced by the polycube decomposes the input mesh locally into quadrilateral patches and globally into hexahedral domains. A global parameterization aligned with a cross field is then computed. The cross field is topologically conform with the quad layout, and geometrically aligned with the surface principal directions and boundaries. The parameterization is optimized until a local optimum of global embedding quality of the polycube is reached. The control quadrilateral mesh is finally extracted from the optimized global aligned parameterization. Figure 4.9 illustrates the complete algorithm, and Figure 4.10 illustrates the aligned global parameterization during the course of the optimization of the polycube's embedding.



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

Figure 4.10: The evolution of the aligned global parameterization after 0 (a), 5 (b), 10 (c), and 15 (d) iterations. In the beginning severe distortions and inversions are present which successively vanish in the course of the optimization.

For the subsequent principal component analysis using singular value decomposition, all aortas must have the same representation (in order to avoid a projection step), i.e., meshes with the same number of nodes and connectivity. This can be implicitly achieved by mapping all the input meshes to the same canonical domain. In addition since each decomposition is optimized to fit the given input geometry, not only the output meshes will have the same topology, but they have the optimal geometry representing each shape. Figure 4.11 illustrates the morphing between two aortas with different morphologies, and Figure 4.12 illustrates isotopological meshes for different patients.

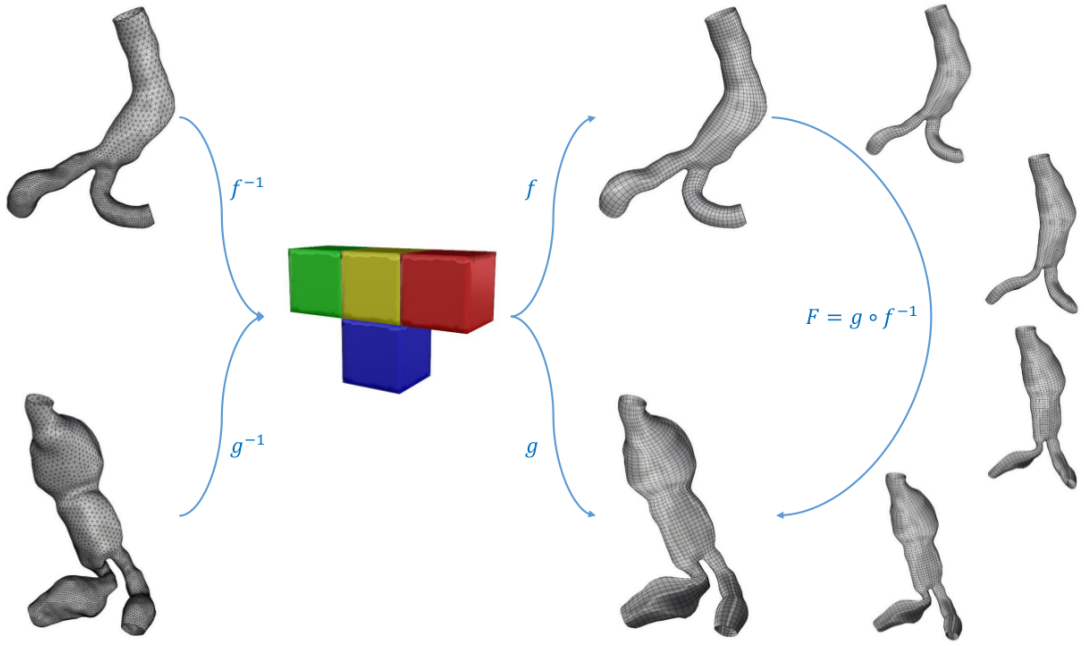


Figure 4.11: Morphing between two aortas with different morphologies.

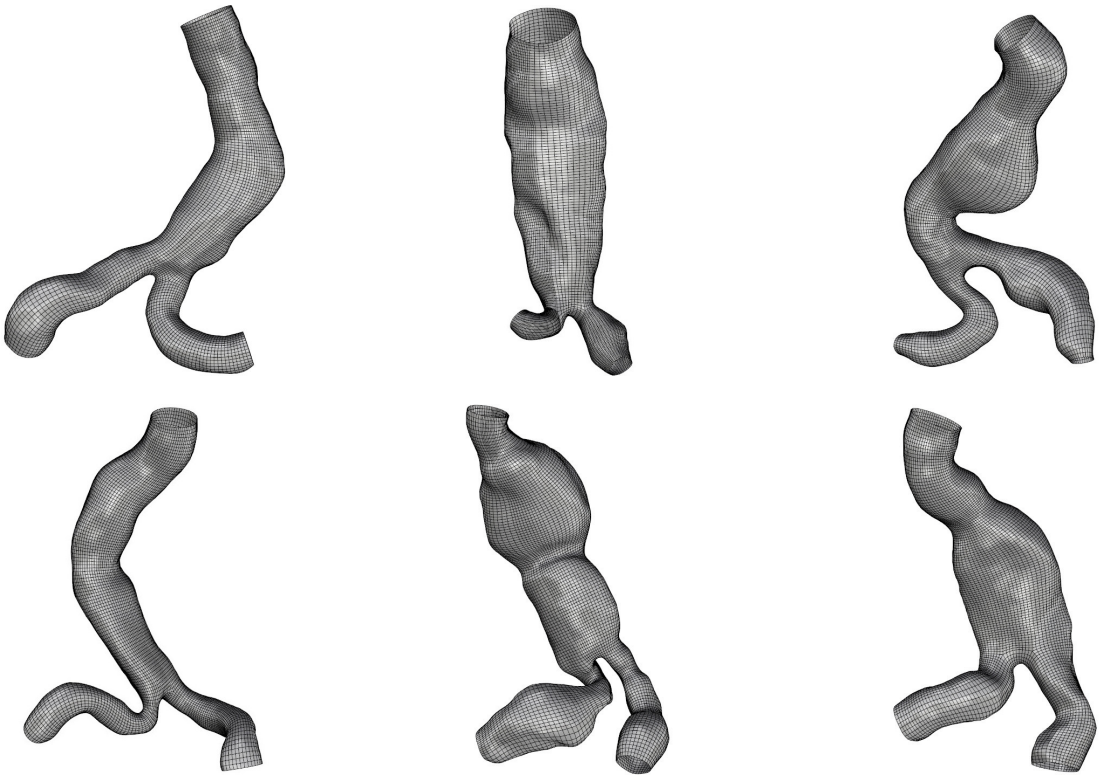


Figure 4.12: Isotopological meshes for aortas with different morphologies.

4.2.3 Singular Value Decomposition

Consider a set of m shapes $\mathbf{x}_j$ $(j = 1, \ldots, m)$, each of them consists of a set of n points $\mathbf{p}_i$ $(i = 1, \ldots, n)$, where each shape is represented as:

$$\mathbf{x}_j = \begin{bmatrix} x_1 & y_1 & z_1 & \cdots & x_n & y_n & z_n \end{bmatrix}^T, \tag{4.7}$$

where (x_i, y_i, z_i) are the coordinates of a point $\mathbf{p}_i$ of the shape $\mathbf{x}_j$. In our case, the set of aortas composes the set of shapes and each aorta is represented by the set of points of its mesh (Figure 4.13). All aortas are represented by isotopological meshes. Hence each mesh can be considered as a snapshot and the snapshots matrix is given by:

$$\mathbf{U} = \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_m \end{bmatrix}. \tag{4.8}$$

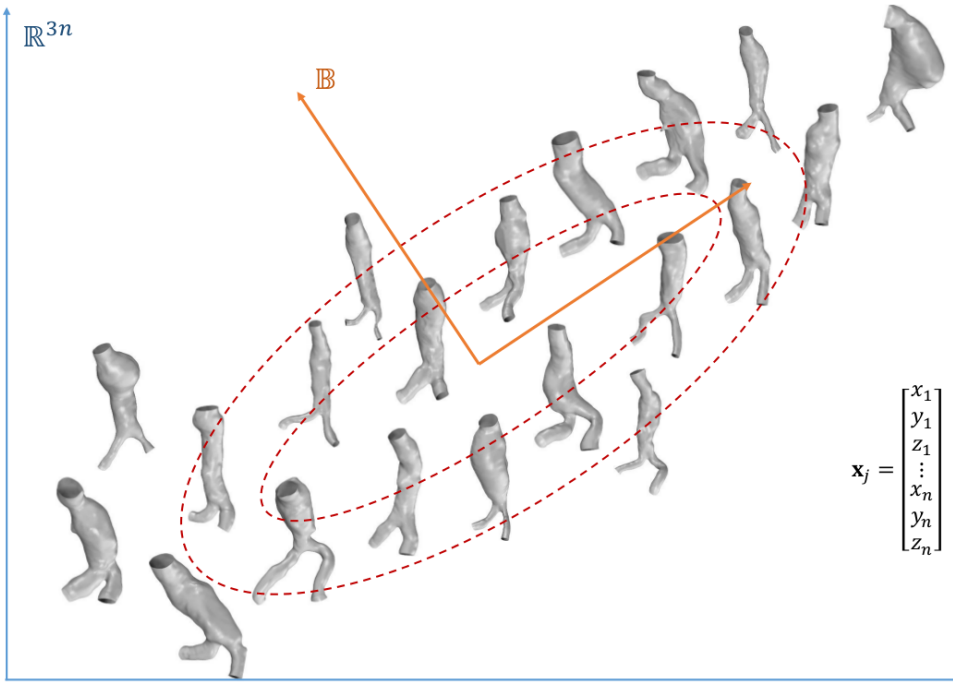


Figure 4.13: Each shape is a vector in $\mathbb{R}^{3n}$. The goal is to find a new subspace $\mathbb{B}$ with better coordinate system which reflects the distribution of the data. In such basis, few coordinates suffice to represent a high dimensional vector.

The goal is to project the data onto a low-dimensional linear subspace that best explains their variation. This subspace can be found using principal component analysis. The SVD factorization of $\mathbf{U}$ can be written as:

$$\mathbf{U} = \mathbf{\Psi}\mathbf{\Sigma}\mathbf{\Phi}^T = \sum_{k=1}^{l} \sigma_k \boldsymbol{\psi}_k \boldsymbol{\phi}_k^T \text{ with } l = \min(n, m), \tag{4.9}$$

where $\mathbf{\Psi} \in \mathbb{R}^{n \times n}$ contains left-singular vectors $\boldsymbol{\psi}_i = \begin{bmatrix} \psi_{i,1} & \cdots & \psi_{i,n} \end{bmatrix}^T$, $\mathbf{\Phi} \in \mathbb{R}^{m \times m}$ contains right-singular vectors $\boldsymbol{\phi}_j = \begin{bmatrix} \phi_{j,1} & \cdots & \phi_{j,m} \end{bmatrix}^T$, and $\mathbf{\Sigma} \in \mathbb{R}^{n \times m}$ contains positive singular values σ_k in decreasing amplitude (Figure 4.14).

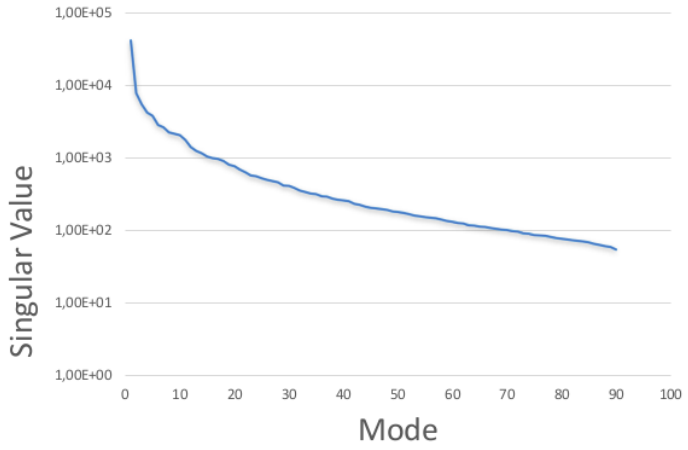


Figure 4.14: The amplitude of the singular values decreases with respect to the modes.

An existing snapshot $\mathbf{x}_j$ $(j = 1, \ldots, m)$ can be written as a linear combination of left singular vectors $\boldsymbol{\psi}$ amplified by the singular values σ and the values of right singular vectors ϕ (Figure 4.15):

$$\mathbf{x}_j = \sum_{k=1}^{l} \sigma_k \boldsymbol{\psi}_k \phi_{k,j} = \sum_{k=1}^{l} \alpha_k^j \boldsymbol{\psi}_k \text{ with } \alpha_k^j = \sigma_k \cdot \phi_{k,j}. \tag{4.10}$$

The left singular vectors $\boldsymbol{\psi}$ are called modes, and the value α_k^j represents the coefficient of the snapshot $\mathbf{x}_j$ with respect to the mode $\boldsymbol{\psi}_k$. These modes capture the principal variations of the population and the weight of each mode is given by its associated singular value. The first modes capture global characteristics of the population. For instance, the first mode represents a scaling (Figures 4.16a-c) and the second mode contains mainly the orientation (Figures 4.16d-f) over the population of aortas. For higher modes, local effects are more present (Figures 4.16g-i). The mean shape $\mathbf{x}_m$ is given by:

$$\mathbf{x}_m = \sum_{k=1}^{l} \alpha_k^m \boldsymbol{\psi}_k \text{ with } \alpha_k^m = \frac{1}{m} \sum_{j=1}^{m} \alpha_k^j. \tag{4.11}$$

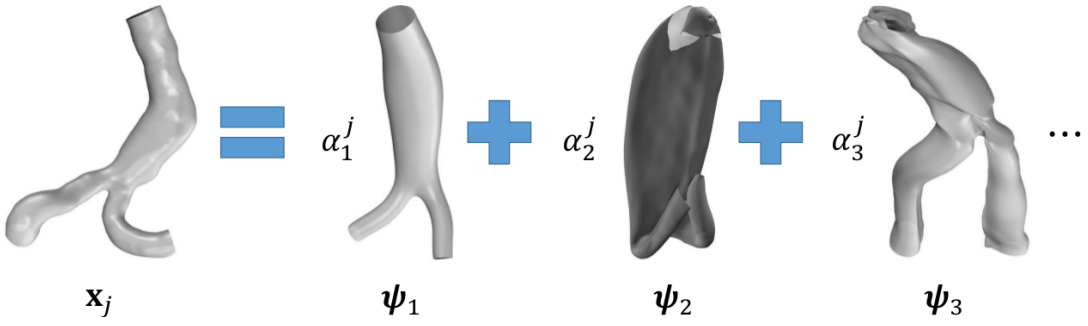


Figure 4.15: Each snapshot can be written as a linear combination of modes. The first mode is homogeneous to a shape and the other modes are homogeneous to a variation.

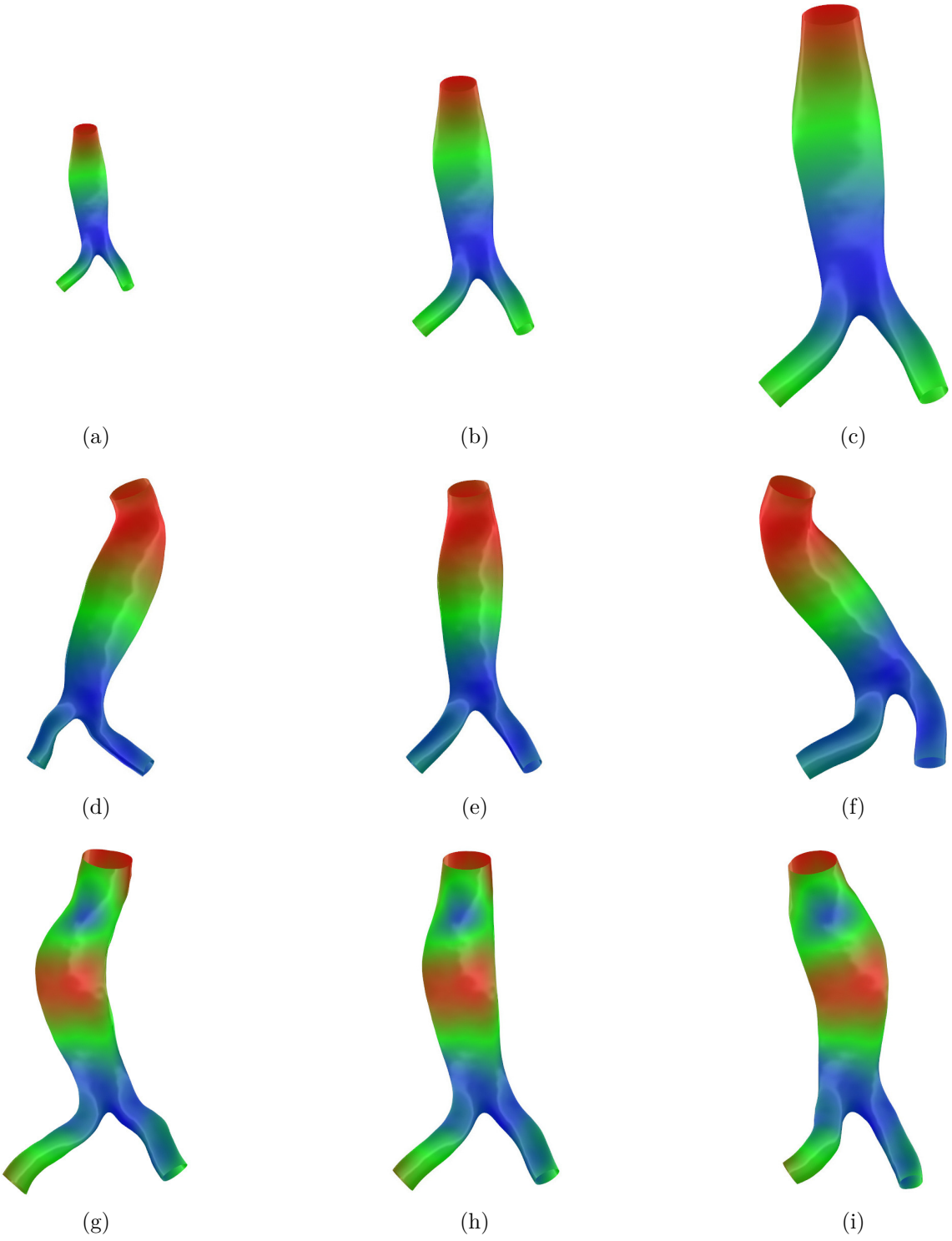


Figure 4.16: Three configurations are depicted corresponding to the minimum (a,d,g), mean (b,e,f) and maximum (c,f,i) coefficients of the represented mode. Between the configurations corresponding to the minimum and maximum coefficients, the nodes which move the most are depicted in red and the ones which move the less are depicted in blue.

Taking into account only the $r \leq k$ first modes allows to define a low rank approximation of $\mathbf{x}_j$ denoted by $\bar{\mathbf{x}}_j$:

$$\bar{\mathbf{x}}_j = \sum_{k=1}^{r} \alpha_k^j \boldsymbol{\psi}_k. \tag{4.12}$$

The approximation basis of rank r will be called the truncated basis of dimension r. The low rank approximation snapshot matrix $\bar{\mathbf{U}}$ of $\mathbf{U}$ is:

$$\bar{\mathbf{U}} = \sum_{k=1}^{r} \sigma_k \boldsymbol{\psi}_k \boldsymbol{\phi}_k^T. \tag{4.13}$$

The relative error between the snapshots matrix $\mathbf{U}$ and its SVD approximation $\bar{\mathbf{U}}$ of rank r is given by:

$$\varepsilon(\mathbf{U}) = \sqrt{\frac{\sum_{i=r+1}^{k} \sigma_i^2}{\sum_{i=1}^{k} \sigma_i^2}}. \tag{4.14}$$

This relative error with respect to the rank of the SVD approximation is depicted in Figure 4.17. For instance, if the truncated basis is of dimension 20, then the relative error ε is about 4.95%.

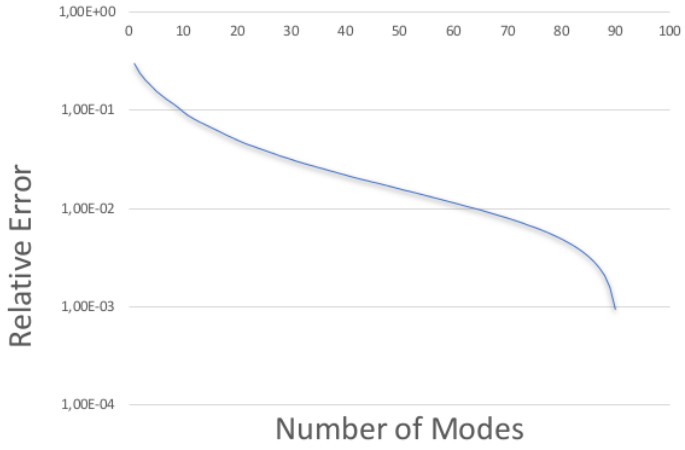


Figure 4.17: Residual square sum of singular values with respect to the number of modes.

A snapshot $\mathbf{x}_k$ can be projected into the truncated basis and its coefficient α_k^j with respect to the mode $\boldsymbol{\psi}_k$ can be computed as:

$$\alpha_k^j = \mathbf{x}_j \cdot \boldsymbol{\psi}_k. \tag{4.15}$$

Figures 4.18-4.19 represent, for two snapshots, a comparison between the exact and projected shapes using a truncated basis of different dimensions. The dimension of the truncated basis, and hence the number of modes, have a direct effect on the precision of the projection. The choice of the number of modes depends on the desired application, and is beyond the scope of this thesis.

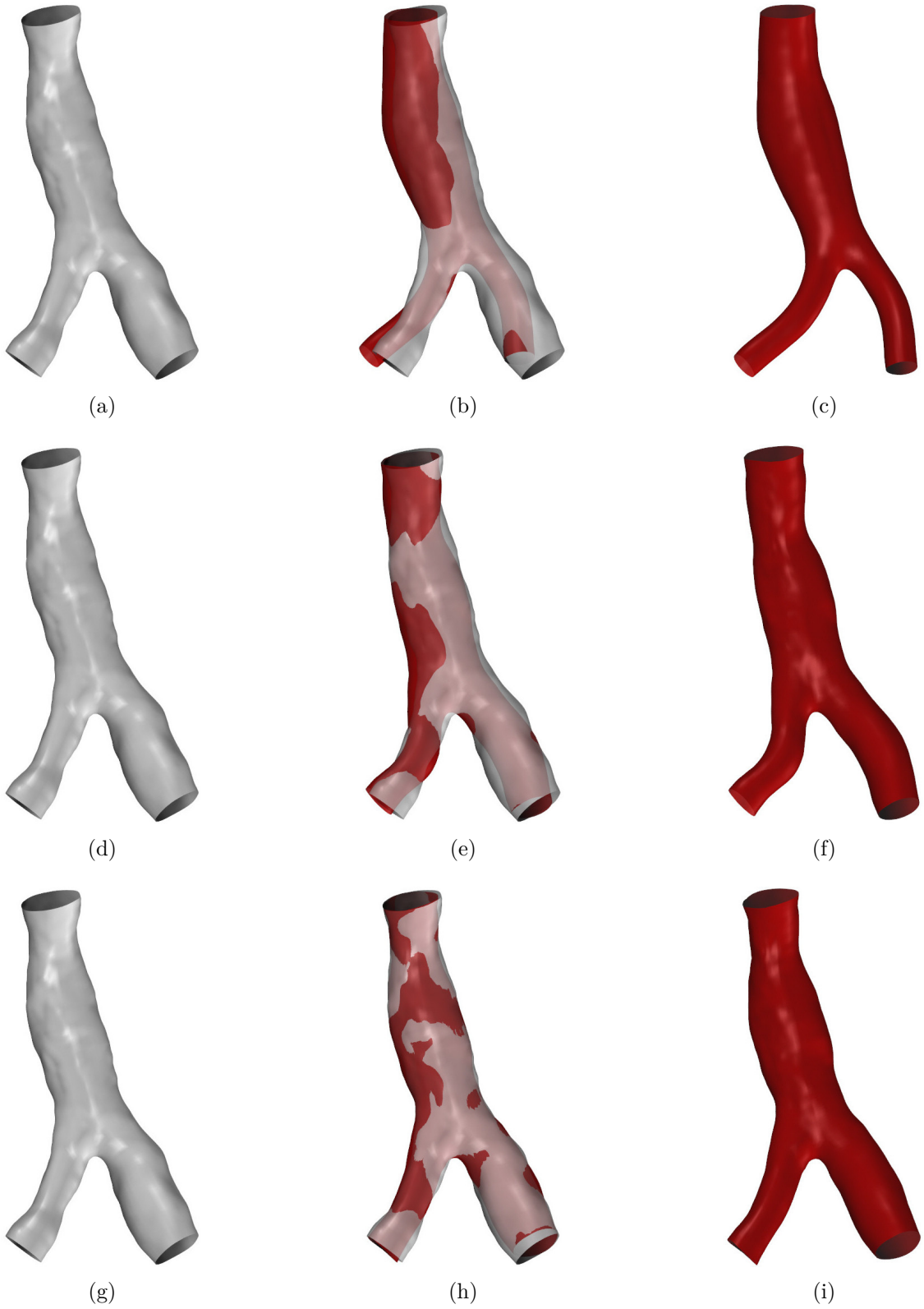


Figure 4.18: The exact (a,d,g) and the projected shapes using 10 (c), 20 (f) and 30 (i) modes. The mean projection error is around 2.3 mm (b), 1.1 mm (e) and 0.6 mm (h).

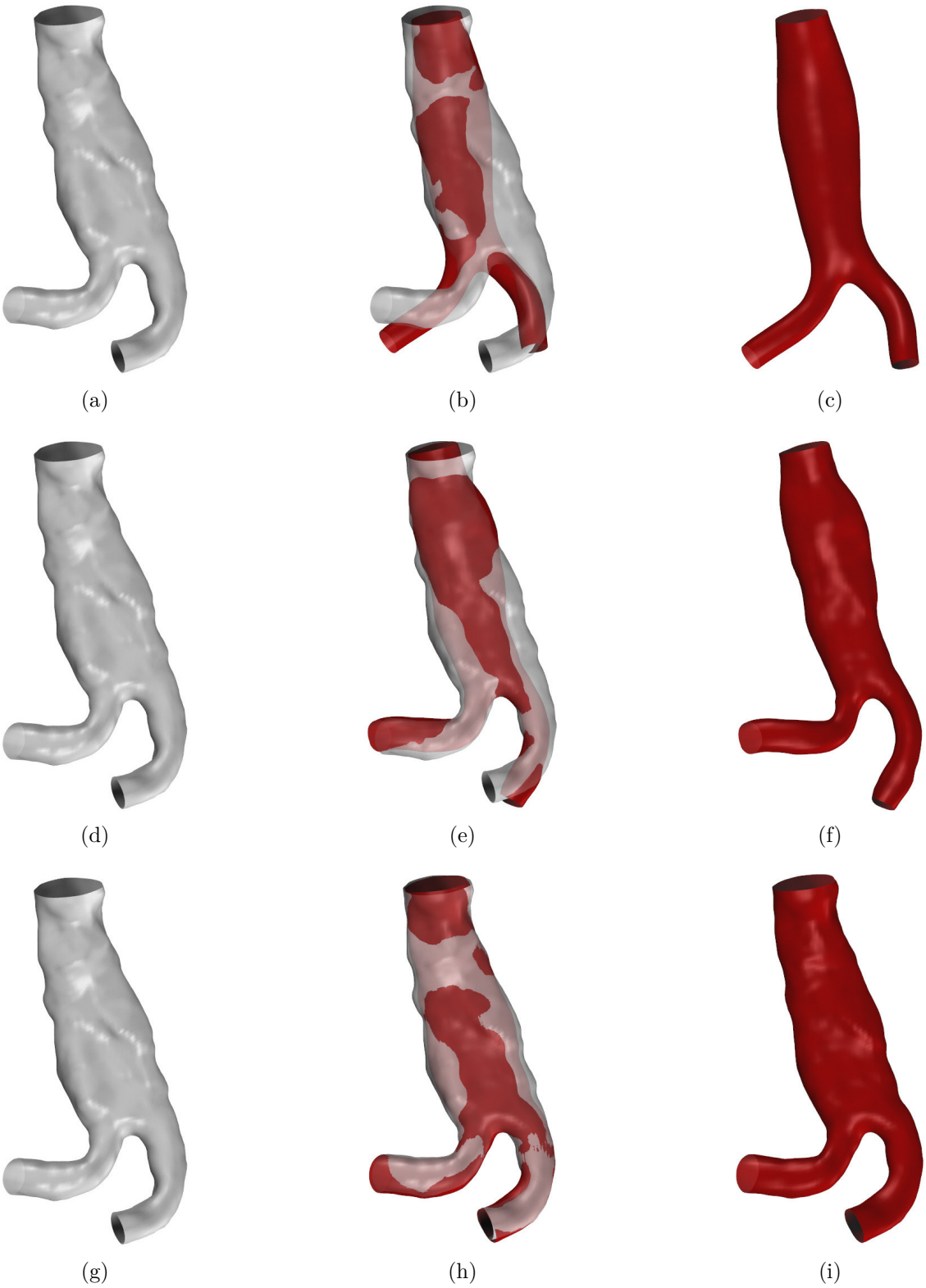


Figure 4.19: The exact (a,d,g) and the projected shapes using 10 (c), 20 (f) and 30 (i) modes. The mean projection error is around 3.6 mm (b), 1.9 mm (e) and 0.7 mm (h).

Conclusion

The efficiency and the robustness of the proposed approach were illustrated by several examples from the mechanical and medical domains. It has been successfully applied to high-genus models with boundaries and sharp features. For models with the same topology but different geometries, this method allows to have the same representation: meshes (or parameterizations) with the same topology. This is very useful in the context of reduced order modeling in order to avoid a potential projection step.

Conclusion

In this thesis, an effective method to automatically construct isogeometric analysis-suitable trivariate models of complicated geometry and arbitrary topology. The input is a solid model defined by its boundary triangle mesh. One of the most crucial problems of the conversion process is the generation of a volume parameterization.

A parameterization is a mapping from an object embedded in $\mathbb{R}^3$ to a parametric domain. The canonical domain must have the same topology as the model but simplified geometrical features. By using polycubes as parametric domains, the problem of finding a volume parameterization of a solid model is simplified to a problem of finding a surface parameterization between the model's boundary and the polycube's boundary. The surface parameterization can be trivially turned into a volume parameterization using interpolation on each cube.

The first step toward a trivariate parameterization is the polycube generation. A polycube is a set of cubes consistently glued together. It can be used to approximate very roughly the geometry of an object while faithfully replicating its topology. Due to its highly regular structure, the polycube can be used as the parametric domain required for tensor-product trivariate spline fitting.

The boundary triangle mesh is decomposed into a set of pants patches. Such segmentation decomposes a complicated surface into a set of shapes that have a trivial topology. A pants patch is a genus-0 surface with 3 boundaries. The Euler characteristic χ for surfaces of most topological types are negative integers. For a pants patch $\chi = -1$, and therefore pants decomposition provides a canonical decomposition scheme for these surfaces. Each pants patch is further decomposed into a set of cuboids. A cuboid is a boxed region enclosed by 6 disk-like patches.

The polycube consists of nodes and arcs embedded in the surface. Locally, neighboring nodes and arcs partition the surface into quadrilateral patches, and globally neighboring quadrilateral patches form hexahedral domains (i.e., cuboids).

The second step toward a trivariate parameterization is the computation of a surface parameterization between the model's boundary and the polycube's boundary. A recent trend are aligned global parameterizations which adapt the parameterization to the geometry of the surface by fitting its gradient to a smooth direction field interpolating reliable principal curvature directions and geometric features. Interestingly the required properties for a good parameterization (such as uniformity, orthogonality and singularities) can be redefined in terms of desired properties of the field.

The anisotropy-adapted direction field is represented by a cross field, i.e., a set of four orthogonal directions at each point of the surface. A balance between three important properties of direction fields is desired: smoothness, singularity positions/indices, and alignment with geometrical features. A smooth cross field, topologically conform with the polycube structure, is designed on the surface. This means that the cross field is singular only at the position of irregular nodes of the polycube. Within this topologically fixed space of cross fields, a smooth cross field interpolating principal curvature directions and geometric features of the surface is computed.

The global parameterization is then found such that its gradient field matches the cross field directions as much as possible by solving a constrained global minimization problem. The parameterization is constrained by the polycube structure. While the described constrained parameterization optimizes the embedding of the polycube's arcs and patches, its nodes remain fixed. This may give rise to large distortions or even local non-injectivities due to fold-overs. A practical solution to this problem is to re-position the polycube's nodes based on the gradient of the parameterization's objective functional with respect to their positions, so as to arrive at a local optimum of global embedding quality.

Using the quadrilateral mesh extracted from the optimized aligned global parameterization, a structured grid of points is generated on each patch, and then used to fit the boundary spline surfaces. For each cuboid, the volume parameterization is obtained using the reconstructed spline surfaces as boundary conditions. Keeping the boundary control points fixed, the interior control points of the spline solid are computed using Coons' interpolation. The positions of the interior control points are then adjusted by minimizing a Laplacien based energy.

The efficiency and the robustness of the proposed approach were illustrated by several examples from the mechanical and medical domains. It has been successfully applied to high-genus models with boundaries and sharp features. For different geometrical instances of the same topological model, this method allows to have the same representation. This is very useful in the context of reduced order modeling to address the issue of having isotopological solutions and avoid potential projection steps.

Limitations and Perspectives

A given surface admits infinitely many pants decompositions. Figures 4.20a-c illustrates different pants decomposition for a genus-2 model. In general, not all pants decomposition results are suitable for the following cuboid decomposition algorithm. For instance, it is the case of the pants decomposition illustrated in Figure 4.20c. As already mentioned, the proposed pants-to-cuboids algorithm is very robust and still going to generate the corresponding cuboid decomposition even for very distorted pants patches (Figure 4.20d). However, the extracted volume parameterization is greatly affected by the quality of the initial pants decomposition and might contain very distorted elements (Figures 4.20e-f). In practice satisfying results were obtained if the pants decomposition is guided by the different geometric criteria (shortest length, symmetry, minima rule).

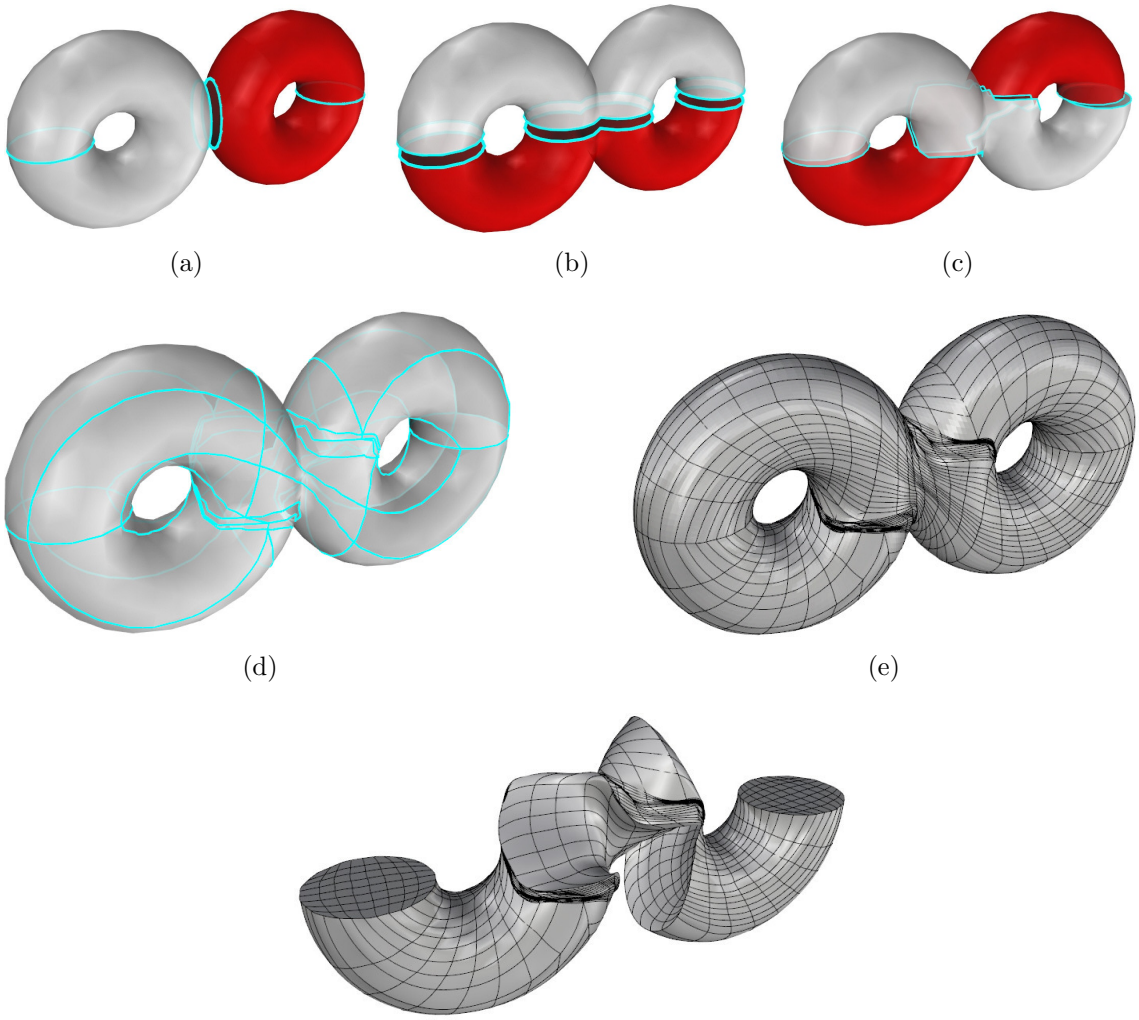


Figure 4.20: Different pants decompositions of a 2-torus (a-c). The cuboid decomposition (d) corresponding to the most distorted pants decomposition (c), along with its surface (e) and volume (f) parameterizations. The volume parameterization is greatly affected by the quality of the initial pants decomposition.

Pants patches can be geometrically assimilated to three branches meeting together. Typically, but not exhaustive, examples of models having such configuration are illustrated in Figures 4.21a-c. The proposed cuboid decomposition algorithm generates only T-shaped configurations (Figure 4.21a). However other configurations might be more optimal in some cases in order to capture the geometrical features of the model. For instance in the case of the plate model, Figures 4.21d-g illustrates two cuboids decompositions using different configurations and its impact on the generated quadrilateral control mesh. In order to select the most suitable configuration in a general way, a potential solution is to extract the skeleton of the model and select the cuboids' configuration in a skeleton-aware manner.

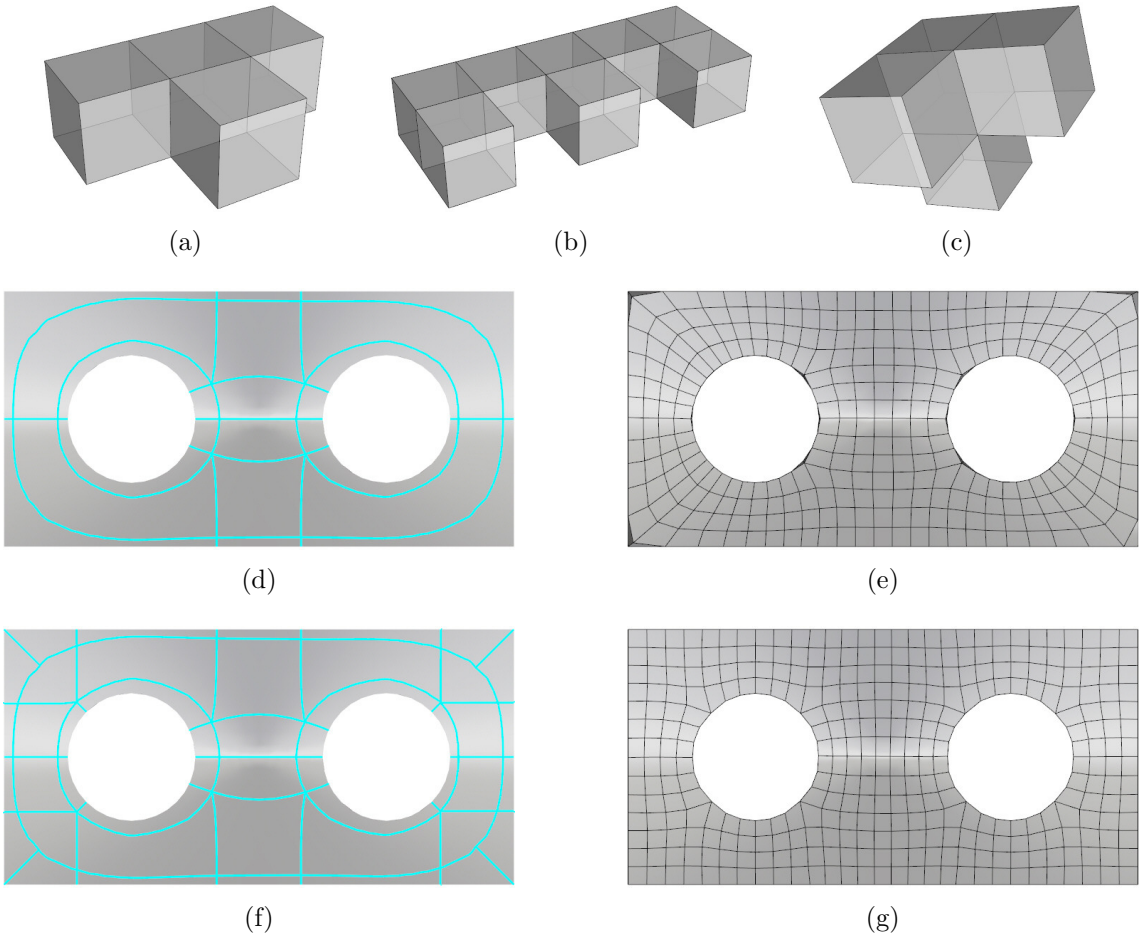


Figure 4.21: Different possible cuboids' configurations for a given pants patch (a-c). The cuboids decomposition for the plate model (d) using the first configuration (a) and the extracted quadrilateral mesh (e). The cuboids decomposition for the plate model (f) using the second configuration (b) and the extracted quadrilateral mesh (g). The quality of the generated quadrilateral control mesh is greatly affected by the choice of the cuboids' configuration.

Bibliography

[Aig09] M. Aigner, C. Heinrich, B. Juttler, E. Pilgerstorfer, B. Simeon, and A. V. Vuong. "Swept Volume Parameterization for Isogeometric Analysis". In: *Proceedings of the 13th IMA International Conference on Mathematics of Surfaces XIII*. 19-44. 2009.

[All03] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. "Anisotropic Polygonal Remeshing". In: *ACM Transactions on Graphics* 22.3 (2003), pp. 485–493.

[All05] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene. *Recent Advances in Remeshing of Surfaces*. Tech. rep. AIM@SHAPE Network of Excellence, 2005.

[Ams12] D. Amsallem, M. Zahr, and C. Farhat. "Nonlinear model order reduction based on local reduced-order bases". In: *International Journal for Numerical Methods in Engineering* 92.10 (2012), pp. 891–916.

[ANS] ANSYS. http://www.ansys.com/.

[Bom12a] D. Bommes. "Quadrilateral Surface Mesh Generation for Animation and Simulation". PhD thesis. 2012.

[Bom13a] D. Bommes, M. Campen, H.-C. Ebke, P. Alliez, and L. Kobbelt. "Integer-Grid Maps for Reliable Quad Meshing". In: *ACM Transactions on Graphics* 32.4 (2013), 98:1–98:12.

[Bom11] D. Bommes, T. Lempfer, and L. Kobbelt. "Global Structure Optimization of Quadrilateral Meshes". In: *Computer Graphics Forum* 30.2 (2011), pp. 375–384.

[Bom13b] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. Silva, and D. Zorin. "Quad-Mesh Generation and Processing: a survey". In: *Computer Graphics Forum* 32.6 (2013), pp. 51–76.

[Bom08] D. Bommes, T. Vossemer, and L. Kobbelt. "Quadrangular Parameterization for Reverse Engineering". In: *Proceedings of the 7th International Conference on Mathematical Methods for Curves and Surfaces*. MMCS'08. 2008, pp. 55–69.

[Bom09] D. Bommes, H. Zimmer, and L. Kobbelt. "Mixed-Integer Quadrangulation". In: *ACM Transactions on Graphics* 28.3 (2009), 77:1–77:10.

[Bom12b] D. Bommes, H. Zimmer, and L. Kobbelt. "Practical Mixed-Integer Optimization for Geometry Processing". In: *Proceedings of the 7th International Conference on Curves and Surfaces* (2012), pp. 193–206.

[Bou13] L. Boucinha, A. Gravouil, and A. Ammar. “Space-time proper generalized decompositions for the resolution of transient elastodynamic models”. In: *Computer Methods in Applied Mechanics and Engineering* 255 (2013), pp. 67–88.

[Cam12] M. Campen, D. Bommes, and L. Kobbelt. “Dual Loops Mesh: Quality Quad Layouts on Manifolds”. In: *ACM Transactions on Graphics* 31.4 (2012), 110:1–110:11.

[Cam14] M. Campen and L. Kobbelt. “Quad Layout Embedding via Aligned Parameterization”. In: *Computer Graphics Forum* 33.8 (2014), pp. 69–81.

[Cha91] B. Chazelle. “Triangulating a simple polygon in linear time”. In: *Discrete Computational Geometry* 6.5 (1991), pp. 485–524.

[Che08] Y. Chen, T. Davis, W. Hager, and S. Rajamanickam. “Algorithm 887: CHOLMOD, Supernodal Sparse Cholesky Factorization and Update Downdate”. In: *ACM Transactions on Mathematical Software* 35.3 (2008), 22:1–22:14.

[Chi10] F. Chinesta, A. Ammar, and E. Cueto. “Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models”. In: *Archives of Computational methods in Engineering* 17.4 (2010), pp. 327–350.

[Chi14] F. Chinesta, R. Keunings, and A. Leygue. “The Proper Generalized Decomposition for Advanced Numerical Simulations”. In: *Springer International Publishing* (2014).

[Chi11] F. Chinesta, P. Ladevèze, and E. Cueto. “A short review on model order reduction based on proper generalized decomposition”. In: *Archives of Computational Methods in Engineering* 18 (2011), pp. 395–404.

[Coh10] E. Cohen, T. Martin, R. Kirby, T. Lyche, and R. Riesenfeld. “Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 199 (2010), pp. 334–356.

[Coh01] E. Cohen, R. Riesenfeld, and G. Elber. *Geometric Modeling with Splines: An Introduction*. A.K. Peters, 2001.

[Cot09] J. Cottrell, T. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. Ed. by Wiley. 2009.

[Cra10] K. Crane, M. Desbrun, and P. Schröder. “Trivial Connections on Discrete Surfaces”. In: *Computer Graphics Forum (SGP)* 29.5 (2010), pp. 1525–1533.

[Dav08] T. Davis. “A multithreaded multifrontal sparse QR factorization”. In: *University of Florida* (2008).

[dAz00] E.-F. d’Azevedo. “Are bilinear quadrilaterals better than linear triangles?” In: *SIAM J. of Scientific Computing* 22.1 (2000), pp. 198–217.

[Dem99] J. Demmel, S. Eisenstat, J. Gilbert, X. Li, and J. Liu. “A supernodal approach to sparse partial pivoting”. In: *SIAM J. Matrix Analysis and Applications* 20.3 (1999), pp. 720–755.

[Dey07] T. K. Dey, K. Li, and J. Sun. “On Computing Handle and Tunnel Loops”. In: *International Conference on Cyberworlds* (2007), pp. 357–366.

[Dic02] J.-M. Dichler, K. Maritaud, B. Levy, and D. Chazanfarpour. “Texture particles”. In: *Computer Graphics Forum* 21.3 (2002).

[Dij59] E. W. Dijkstra. “A note on two problems in connexion with graphs”. In: *Numerische Mathematik* 1 (1959), pp. 269–271.

[Don06] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. Hart. “Spectral Surface Quadrangulation”. In: *ACM Transactions on Graphics* 25.3 (2006), pp. 1057–1066.

[Dry98] I.-L. Dryden and K.-V. Mardia. “Statistical Shape Analysis”. In: *John Wiley & Sons* (1998).

[Eck36] C. Eckart and G. Young. “The approximation of one matrix by another of lower rank”. In: *Psychometrika* 1 (1936), pp. 211–218.

[Eri05] J. Erickson and K. Whittlesey. “Greedy Optimal Homotopy and Homology Generators”. In: *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA ’05. 2005, pp. 1038–1046.

[Esc11] J. Escobar, J. M. Cascon, E. Rodriguez, and R. Montenegro. “A new approach to solid modeling with trivariate T-splines based on mesh optimization”. In: *Computer Methods in Applied Mechanics and Engineering* 200 (2011), pp. 3210–3222.

[Far99] G. Farin. *NURBS Curves and Surfaces: from Projective Geometry to Practical Use*. A.K. Peters, 1999.

[Far02] G. Farin. *Curves and Surfaces for CAGD : A Pracical Guide*. Morgan Kaufmann Publishers Inc., 2002.

[Fis06] M. Fisher, B. Springborn, A. Bobenko, and P. Schroder. “An Algorithm for the Construction of Intrinsic Delaunay Triangulations with Applications to Digital Geometry Processing”. In: *ACM SIGGRAPH 2006 Courses*. SIGGRAPH ’06. 2006, pp. 69–74.

[Flo03] M. Floater. “Mean value coordinates”. In: *Computer Aided Geometric Design* 20.1 (2003), pp. 19–27.

[Fou84] A. Fournier and D.-Y. Montuno. “Triangulating simple polygons and equivalent problems”. In: *ACM Transactions of Graphics* 3.2 (1984), pp. 153–174.

[Gal11a] F. Galland. “An adaptive model reduction approach for 3D fatigue crack growth in small scale yielding conditions”. PhD thesis. INSA de Lyon, 2011.

[Gal11b] F. Galland, A. Gravouil, E. Malvesin, and M. Rochette. “A global model reduction approach for 3d fatigue crack growth with confined plasticity”. In: *Computer Methods in Applied Mechanics and Engineering* 200.5 (2011), pp. 699–716.

[Glo09] S. Glotzer, S. Kim, P. Cummings, A. Deshmukh, M. Head-Gordon, G. Karniadakis, L. Petzold, C. Sagui, and M. Shinozuka. “International assessment of research and development in simulation based engineering and science”. In: *World Technology Evaluation Center* (2009).

[Gre11] J. Gregson, A. Sheffer, and E. Zhang. "All-Hex Mesh Generation via Volumetric PolyCube Deformation". In: *Computer Graphics Forum (Special Issue of Symposium on Geometry Processing 2011)* 30.5 (2011).

[Hat01] A. Hatcher. *Algebraic Topology*. http://www.math.cornell.edu/ hatcher, 2001.

[Hat00] A. Hatcher, P. Lochak, and L. Schneps. "On the teichmüller tower of mapping class groups". In: *Journal Für Die Reine Und Angewandte Mathematik* 521 (2000), pp. 1–24.

[He09] Y. He, H. Wang, C.-W. Fu, and H. Qin. "A Divide-and-conquer Approach for Automatic Polycube Map Construction". In: *Computers and Graphics* 33.3 (2009), pp. 369–380.

[Her00] A. Hertzmann and D. Zorin. "Illustrating Smooth Surfaces". In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '00. 2000, pp. 517–526.

[Hey13] C. Heyberger, P.-A. Boucard, and D. Néron. "A rational strategy for the resolution of parametrized problems in the PGD framework". In: *Computer Methods in Applied Mechanics and Engineering* 259 (2013), pp. 40–49.

[Hor06] A. Hornung and L. Kobbelt. "Robust Reconstruction of Watertight 3D Models from Non-uniformly Sampled Point Clouds Without Normal Information". In: *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*. SGP '06. 2006.

[Hua14] J. Huang, T. Jiang, Z. Shi, Y. Tong, H. Bao, and M. Desbrun. "L1-based construction of polycube maps from complex shapes". In: *ACM Transactions of Graphics* 33.3 (2014), 25:1–25:11.

[Hua08] J. Huang, M. Zhang, J. Ma, X. Liu, L. Kobbelt, and H. Bao. "Spectral Quadrangulation with Orientation and Alignment Control". In: *ACM Transactions on Graphics* 27.5 (2008), 147:1–147:9.

[Hug05] 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* 195.39-41 (2005), pp. 4135–4195.

[Jam15] C. Jamin, P. Alliez, M. Yvinec, and J.-D. Boissonnat. "CGALmesh: A Generic Framework for Delaunay Mesh Generation". In: *ACM Transactions on Mathematical Software* 41.4 (2015), 23:1–23:24.

[Jin08] M. Jin, J. Kim, F. Luo, and X. Gu. "Discrete Surface Ricci Flow". In: *IEE Transactions on Visualization and Computer Graphics* 14.5 (2008).

[Käl07] F. Kälberer, M. Nieser, and K. Polthier. "QuadCover - Surface Parameterization using Branched Coverings". In: *Computer Graphics Forum* 26.3 (2007), pp. 375–384.

[Kaz06] M. Kazhdan, M. Bolitho, and H. Hoppe. "Poisson Surface Reconstruction". In: *Eurographics Symposium on Geometry Processing* (2006), pp. 61–70.

[Ker11] P. Kerfriden, P. Gosselet, S. Adhikari, and S. Bordas. “Bridging proper orthogonal decomposition methods and augmented newton-krylov algorithms: An adaptive model order reduction for highly nonlinear mechanical problems”. In: *Computer Methods in Applied Mechanics and Engineering* 200.850-866 (2011).

[Knö13] F. Knöppel, K. Crane, U. Pinkall, and P. Schröder. “Global Optimal Direction Fields”. In: *ACM Transactions of Graphics* 32.4 (2013), 59:1–59:10.

[Kun01] K. Kunisch and S. Volkwein. “Galerkin proper orthogonal decomposition methods for parabolic problems”. In: *Numerische Mathematik* 90 (2001).

[Lad10] P. Ladevèze, J.-C. Passieux, and D. Néron. “The LATIN multiscale computational method and the proper generalized decomposition”. In: *Computer Methods in Applied Mechanics and Engineering* 199 (2010), pp. 1287–1296.

[Lai10] Y.-K. Lai, M. Jin, X. Xie, Y. He, J. Palacios, E. Zhang, S.-M. Hu, and X. Gu. “Metric Driven RoSy Field Design and Remeshing”. In: *IEEE Transactions on Visualization and Computer Graphics* 16.1 (2010), pp. 95–108.

[Lee05] Y. Lee, S. Lee, A. Shamir, D. Cohen-Or, and H.-P. Seidel. “Mesh Scissoring with Minima Rule and Part Salience”. In: *Computer Aided Geometric Design* 22.5 (2005), pp. 444–465.

[Lev00] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. “The Digital Michelangelo Project: 3D Scanning of Large Statues”. In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’00. 2000, pp. 131–144.

[Li12] B. Li, X. Li, K. Wang, and H. Qin. “Surface mesh to volumetric spline conversion with generalized poly-cubes”. In: *IEEE Tansactions On Visualization And Computer Graphics* 19.9 (2012), pp. 1539–1551.

[Li06] W.-C. Li, B. Valletand, N. Ray, and B. Lévy. “Representing Higher-Order Singularities in Vector Field on Piecewise Linear Surfaces”. In: *IEEE Tansactions On Visualization And Computer Graphics* 12.5 (2006), pp. 1315–1322.

[Li08] X. Li, X. Gu, and H. Qin. “Surface Matching Using Consistent Pants Decomposition”. In: *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*. SPM ’08. 2008, pp. 125–136.

[Lin08] J. Lin, X. Jin, Z. Fan, and C. Wang. “Automatic polycube-maps”. In: *Geometric Modeling and Processing* (2008), pp. 3–16.

[Liv13] M. Livesu, N. Vining, A. Sheffer, J. Gregson, and R. Scateni. “Polycut: monotone graph-cuts for polycube base-complex construction”. In: *ACM Transactions of Graphics* 32.6 (2013), 171:1–171:12.

[Ma95] W. Ma and J. Kruth. “Parameterization of randomly measured points for least squares tting of b-spline curves and surfaces”. In: *Computer-Aided Design* 27.9 (1995), pp. 663–675.

[Mar04] M. Marinov and L. Kobbelt. “Direct Anisotropic Quad-Dominant Remeshing”. In: *Proceedings of the Computer Graphics and Applications, 12th Pacific Conference*. PG ’04. 2004, pp. 207–216.

[Mar10] T. Martin and E. Cohen. “Volumetric parameterization of complex objects by respecting multiple materials”. In: *Computer and Graphics* 26.6 (2010), pp. 648–664.

[Mar09] T. Martin, E. Cohen, and M. Kirby. “Volumetric Parameterization and Trivariate B-spline Fitting Using Harmonic Functions”. In: *Computer Aided Geometric Design*. SPM ’08 26 (2009), pp. 648–664.

[Med16] B. Medical. “Abdominal Aneurysm”. 2016.

[Moo65] G. Moore. “Cramming more components onto integrated circuits”. In: *Proceedings of the EEE* 86.1 (1965).

[Myl10] A. Myles, N. Pietroni, D. Kovacs, and D. Zorin. “Feature-aligned T-meshes”. In: *ACM Transactions on Graphics* 29.4 (2010), 117:1–117:11.

[Myl12] A. Myles and D. Zorin. “Global Parameterization by Incremental Flattening”. In: *ACM Transactions of Graphics* 31.4 (2012), 109:1–109:11.

[Myl13] A. Myles and D. Zorin. “Controlled-Distortion Constrained Global Parameterization”. In: *ACM Transactions of Graphics* 32.4 (2013), 105:1–105:14.

[Nie10] M. Nieser, J. Palacios, K. Polthier, and E. Zhang. “Hexagonal Global Parameterization of Arbitrary Surfaces”. In: *ACM SIGGRAPH ASIA 2010 Sketches* (2010), 5:1–5:2.

[Pie97] L. Piegl and W. Tiller. *The NURBS Book*. Springer-Verlag New York, Inc., 1997.

[Pil14] E. Pilgerstorfer and B. Jüttler. “Bounding the Influence of Domain Parameterization and Knot Spacing on Numerical Stability in Isogeometric Analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 268 (2014), pp. 589–613.

[Pra00] E. Praun, A. Finkelstein, and H. Hoppe. “Lapped textures”. In: *Proceedings of ACM SIGGRAPH* (2000), pp. 465–470.

[Ray06] N. Ray, W. C. Li, B. Lévy, A. Sheffer, and P. Alliez. “Periodic Global Parameterization”. In: *ACM Transactions on Graphics* 25.4 (2006), pp. 1460–1485.

[Ray09] N. Ray, B. Vallet, L. Alonso, and B. Lévy. “Geometry Aware Direction Field Processing”. In: *ACM Transactions on Graphics* 29.1 (2009), 1:1–1:11.

[Ray08] N. Ray, B. Vallet, W. C. Li, and B. Lévy. “N-Symmetry Direction Field Design”. In: *ACM Transactions on Graphics* 27.2 (2008), 10:1–10:13.

[Raz15] F. Razafindrakaza, U. Reitebuch, and K. Polthier. “Perfect Matching Quad Layouts for Manifold Meshes”. In: *Computer Graphics Forum* 34.5 (2015), pp. 219–228.

[Rhia] Rhinoceros. http://www.rhino3d.com/.

[Rhib] RhinoCommon. http://wiki.mcneel.com/developer/rhinocommon.

[Rog01] D. Rogers. *An Introduction to NURBS: With Historical Perspective*. Morgan Kaufmann Publishers Inc., 2001.

[Ryc05] D. Ryckelynck. “A priori hyperreduction method: an adaptive approach”. In: *Journal of Computational Physics* 202.1 (2005), pp. 346–366.

[Ryc06] D. Ryckelynck, F. Chinesta, E. Cueto, and A. Ammar. “On the a priori model reduction: Overview and recent developments”. In: *Archives of Computational Methods in Engineering* 13.91-128 (2006).

[Shi04] X. Shi, T. Wang, P. Wu, and F. Liu. “Reconstruction of convergent G1 smooth B-spline surfaces”. In: *Computer Aided Geometric Design* 21.9 (2004), pp. 893–913.

[Tar04] M. Tarini, K. Hormann, P. Cignoni, and C. Montan. “PolyCube-Maps”. In: *ACM Transactions of Graphics* 23.3 (2004), pp. 853–860.

[The02] H. Theisel. “Designing 2D vector fields of arbitrary topology”. In: *Computer Graphics Forum (Proceedings Eurographics 2002)* 21.3 (2002), pp. 595–604.

[Ton06] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun. “Designing Quadrangulations with Discrete Harmonic Forms”. In: *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*. SGP ’06. 2006, pp. 201–210.

[Tri02] X. Tricoche. “Vector and Tensor Field Topology Simplification, Tracking, and Visualization”. PhD thesis. 2002.

[Tri00] X. Tricoche, G. Scheuermann, and H. Hagen. “Higher Oder Singularities in Piecewise Linear Vector Fields”. In: *Proceedings of IMA Conference on the the Mathematics of Surfaces* (2000), pp. 99–113.

[Tur01] G. Turk. “Texture synthesis on surfaces”. In: *Proceedings of ACM SIGGRAPH* (2001), pp. 347–354.

[Ver07] E. Verdière and F. Lazarus. “Optimal pants decomposition and shortest homotopic cycles on an orientable surface”. In: *Journal of the ACM* 54.4 (2007).

[Ver03] K. Veroy. “Reduced-basis methods applied to problems in elasticity: Analysis and applications”. PhD thesis. Massachusetts Institute of Technology, 2003.

[Wan07] H. Wang, Y. He, X. Li, X. Gu, and H. Qin. “Polycube Splines”. In: *Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling*. SPM ’07. 2007, pp. 241–251.

[Wan08] H. Wang, M. Jin, Y. He, X. Gu, and H. Qin. “User-controllable Polycube Map for Manifold Spline Construction”. In: *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*. SPM ’08. 2008, pp. 397–404.

[Wan13] W. Wang, Y. Zhang, L. Liu, and T. J. R. Hughes. “Trivariate solid T-spline construction from boundary triangulations with arbitrary genus topology”. In: *Computer-Aided Design* 45 (2013), pp. 351–360.

[Wan14] X. Wang and X. Qian. “An optimization approach for constructing trivariate B-spline solids”. In: *Computer-Aided Design* 46 (2014), pp. 179–191.

[Wel94] W. Welch and A. Witkin. “Free-form Shape Design Using Triangulated Surfaces”. In: *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’94. 1994, pp. 247–256.

[Xia11] J. Xia, I. Garcia, Y. He, S.-Q. Xin, and G. Patow. “Editable Polycube Map for GPU-based Subdivision Surfaces”. In: *Symposium on Interactive 3D Graphics and Games*. 2011, pp. 151–158.

[Xu13a] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. “Analysis-suitable volume parameterization of multi-block computational domain in isogeometric applications”. In: *Computer-Aided Design* 45 (2013), pp. 395–404.

[Xu13b] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. “Constructing analysis-suitable parameterization of computational domain from CAD boundary by variational harmonic method”. In: *Journal of Computational Physics* 252 (2013), pp. 275–289.

[Xu13c] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. “Optimal analysis-aware parameterization of computational domain in 3D isogeometric analysis”. In: *Computer-Aided Design* 45 (2013), pp. 812–821.

[Xu14] G. Xu, B. Mourrain, A. Galligo, and T. Rabczuk. “High-quality construction of analysis-suitable trivariate NURBS solids by reparameterization methods”. In: *Computational Mechanics* 54.5 (2014), pp. 1303–1313.

[Zen09] W. Zeng, X. Yin, M. Zhang, F. Luo, and X. Gu. “Generalized Koebe’s Method for Conformal Mapping Multiply Connected Domains”. In: *2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling*. SPM ’09. 2009, pp. 89–100.

[Zha07] E. Zhang, J. Hays, and G. Turk. “Interactive Tensor Field Design and Visualization on Surfaces”. In: *IEE Transactions on Visualization and Computer Graphics* 13.1 (2007), pp. 94–107.

[Zha06] E. Zhang, K. Mischaikow, and G. Turk. “Vector Field Design on Surfaces”. In: *ACM Transactions on Graphics* 25.4 (2006), pp. 1294–1326.

[Zha14] K. Zhang and X. Li. “Searching Geometry-aware Pants Decomposition in Different Isotopy Classes”. In: *Geometry, Imaging, and Computing* 1.3 (2014), pp. 367–393.

[Zha10] M. Zhang, J. Huang, X. Liu, and H. Bao. “A Wave-based Anisotropic Quadrangulation Method”. In: *ACM Transactions on Graphics* 29.4 (2010), 118:1–118:8.

[Zha12] Y. Zhang, W. Wang, and T. J. R. Hughes. “Solid T-spline Construction from Boundary Representations for Genus-Zero Geometry”. In: *Computer Methods in Applied Mechanics and Engineering* 201 (2012).

[Zie94] H. Ziezold. “Mean Figures and Mean Shapes Applied to Biological Figure and Shape Distributions in the Plane”. In: *Biometrical Journal* 36 (1994), pp. 491–510.

UNIVERSITÉ DE LYON

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

FOLIO ADMINISTRATIF

THÈSE DE L'UNIVERSITÉ DE LYON OPERÉE AU SEIN DE L'INSA DE LYON

NOM : AL AKHRAS

DATE DE SOUTENANCE : 19 mai 2016

PRÉNOM : Hassan

TITRE : Automatic Isogeometric Analysis Suitable Trivariate Models Generation - Application to Reduced Order Modeling

NATURE : Doctorat

NUMÉRO D'ORDRE : 2016LYSEi047

ÉCOLE DOCTORALE : Mécanique, Energétique, Génie Civil, Acoustique (MEGA ED 162)

SPÉCIALITÉ : Génie Mécanique

RÉSUMÉ :

Cette thèse présente un algorithme automatique pour la construction d'un modèle NURBS volumique à partir d'un modèle représenté par ses bords (maillages ou splines). Ce type de modèle est indispensable dans le cadre de l'analyse isogéométrique utilisant les NURBS comme fonctions de forme. Le point d'entrée de l'algorithme est une triangulation du bord du modèle. Après deux étapes de décomposition, le modèle est approché par un polycube. Ensuite un paramétrage surfacique entre le bord du modèle et celui du polycube est établi en calculant un paramétrage global aligné à un champ de direction interpolant les directions de courbure principales du modèle. Finalement, le paramétrage volumique est obtenu en se basant sur ce paramétrage surfacique. Dans le contexte des études paramétriques basées sur des paramètres de formes géométriques, cette méthode peut être appliquée aux techniques de réduction de modèles pour obtenir la même représentation pour des objets ayant différentes géométries mais la même topologie.

MOTS-CLÉS : Analyse Isogéométrique ; Réduction de Modèle ; Paramétrage Volumique

LABORATOIRE DE RECHERCHE : LaMCoS – Laboratoire de Mécanique des Contacts et des Structures
UMR CNRS 5259 – INSA de Lyon
18-20 rue des Sciences
69621 Villeurbanne Cedex, France

DIRECTEURS DE THÈSE : Pr. Anthony GRAVOUIL et Dr. Thomas ELGUEDJ

PRÉSIDENT DU JURY : Pr. David NERON

COMPOSITION DE JURY :

Pr. Francisco CHINESTA	Pr. David NERON
Pr. Jean-François REMACLE	Dr. Michel ROCHETTE
Pr. Alain RASSINEUX	Pr. Anthony GRAVOUIL
Pr. Yuri BAZILEVS	Dr. Thomas ELGUEDJ