Smooth representation of thin shells and volume structures for isogeometric analysis

DISSERTATION

Zur Erlangung des akademischen Grades eines
Doktor-Ingenieur
an der Fakultät Bauingenieurwesen
der Bauhaus-Universität Weimar

vorgelegt von

Chiu Ling Chan
(interner Doktorand)

geboren am 1. Juni 1986 in Perak, Malaysia

Mentor:
Herr Prof. Dr.-Ing. Timon Rabczuk, Bauhaus-Universität Weimar
Gutachter:
Herr Prof. Dr. rer. nat. habil. Klaus Gürlebeck, Bauhaus-Universität Weimar
Herr Prof. Dr.-Ing. Stéphane Bordas, Université Du Luxembourg

Tag der Disputation: 23 Juni 2020

To my whole family ...

Acknowledgements

I wish to express my sincere gratitude and appreciation to my advisor Prof. Dr. Ing. Timon Rabczuk, for his supervision and inspiration throughout this research. I want to thank him for introducing me to the fascinating fields of image processing and isogeometric analysis. This dissertation would not have been possible without his support and encouragement.

I am also grateful to all people who have contributed towards shaping this dissertation. In particular, I would like to thank Dr. Cosmin Anitescu for his guidance and suggestion throughout the study. I am also thankful to Prof. Jessica Zhang in our collaboration on image processing. Thanks also go to my colleagues at the Institute of Structural Mechanics for their help and friendly support.

Furthermore, I would like to thank the German Academic Exchange Service (DAAD) for supporting and funding the study. I also appreciate the financial support from Bauhaus Research School.

Finally, I would especially like to thank my family for their support and love all the time.

Weimar, Apr 2019

Chiu Ling Chan

Abstract

The purpose of this study is to develop self-contained methods for obtaining smooth meshes which are compatible with isogeometric analysis (IGA). The study contains three main parts. We start by developing a better understanding of shapes and splines through the study of an image-related problem. Then we proceed towards obtaining smooth volumetric meshes of the given voxel-based images. Finally, we treat the smoothness issue on the multi-patch domains with C^1 coupling. Following are the highlights of each part.

First, we present a B-spline convolution method for boundary representation of voxel-based images. We adopt the filtering technique to compute the B-spline coefficients and gradients of the images effectively. We then implement the B-spline convolution for developing a non-rigid images registration method. The proposed method is in some sense of "isoparametric", for which all the computation is done within the B-splines framework. Particularly, updating the images by using B-spline composition promote smooth transformation map between the images. We show the possible medical applications of our method by applying it for registration of brain images.

Secondly, we develop a self-contained volumetric parametrization method based on the B-splines boundary representation. We aim to convert a given voxel-based data to a matching C^1 representation with hierarchical cubic splines. The concept of the osculating circle is employed to enhance the geometric approximation, where it is done by a single template and linear transformations (scaling, translations, and rotations) without the need for solving an optimization problem. Moreover, we use the Laplacian smoothing and refinement techniques to avoid irregular meshes and to improve mesh quality. We show with several examples that the method is capable of handling complex 2D and 3D configurations. In particular, we parametrize the 3D Stanford bunny which contains irregular shapes and voids.

Finally, we propose the Bézier ordinates approach and splines approach for C^1 coupling. In the first approach, the new basis functions are defined in terms of the Bézier-Bernstein polynomials. For the second approach, the new basis is defined as a linear combination of C^0 basis functions. The methods are not limited to planar or bilinear mappings. They allow the modeling of solutions to fourth order partial differential equations (PDEs) on complex geometric domains, provided that the given patches are G^1 continuous. Both methods have their advantages. In particular, the Bézier approach offer more degree of freedoms, while the spline approach is more computationally efficient. In addition, we proposed partial degree elevation to overcome the C^1-locking issue caused by the over constraining of the solution space. We demonstrate the potential of the resulting C^1 basis functions for application in IGA which involve fourth order PDEs such as those appearing in Kirchhoff-Love shell models, Cahn-Hilliard phase field application, and biharmonic problems.

Contents

1 Introduction 1
  1.1 Background, motivation and problem description 1
  1.2 Summary of major work 2
  1.3 Outline of dissertation 4

2 Isogeometric analysis framework 5
  2.1 B-splines and NURBS 5
  2.2 Bézier decomposition 7
  2.3 Hierarchical refinement with PHT-splines 10
    2.3.1 PHT-Spline Space 12
    2.3.2 Computing the control points 13
  2.4 Analysis using splines 14
    2.4.1 Galerkin method 15
    2.4.2 Linear elasticity 17
  2.5 Numerical examples 19
    2.5.1 Infinite plate with circular hole 19
    2.5.2 Open spanner 19
    2.5.3 Pinched cylinder 20
    2.5.4 Hollow sphere 21
  2.6 Conclusions 22

3 B-splines for Image processing 23
  3.1 Previous work 23
  3.2 Voxelized images 25
    3.2.1 B-spline level set representation 26
  3.3 Registration framework 29
  3.4 Non-rigid registration 31
    3.4.1 The governing equation 32
    3.4.2 Diffeomorphisms 33
    3.4.3 Regularization 34
    3.4.4 Moving image update 34

3.4.5 Parallel computing and Matlab implementation 35
3.5 Multiresolution registration 36
3.6 Numerical examples 39
3.6.1 Examine on synthetic transformation 39
3.6.2 C-shaped image 39
3.6.3 3D registration 41
3.6.4 3D registration of medical image 41
3.7 Conclusions 44

4 Volumetric parameterization 47
4.1 Previous studies 47
4.2 B-spline boundary representation 49
4.3 Volumetric parametrization of voxelized data 50
4.3.1 Classifying the boundary elements 51
4.3.2 Determining the adjustable vertices 52
4.3.3 Adjusting vertices 53
4.3.4 Smoothing and untangling the elements 59
4.3.5 Refinement 59
4.4 Numerical Examples 59
4.4.1 Infinite plate with circular hole 60
4.4.2 Yeti footprint 61
4.4.3 Open spanner 61
4.4.4 Spherical hole in an infinite domain 63
4.4.5 Stanford Bunny 64
4.5 Conclusions 65

5 C^1 coupling 67
5.1 Previous studies 67
5.2 Geometrically continuous isogeometric functions 69
5.2.1 Bézier ordinates approach 72
5.2.2 Spline approach 75
5.3 Smooth isogeometric functions in 2D 77
5.3.1 C^1 continuity condition in 2D 78
5.3.2 C^1 coupling in 2D 80
5.3.3 C^1 coupling of finer meshes 84
5.4 Multi-patch coupling 86
5.4.1 Localization using MDS 94
5.5 Partial degree elevation 97
5.5.1 Partial degree elevation for higher dimension 99
5.6 C^1 coupling for surface in space 101
5.7 C^1 coupling for 3D 102

5.8 Numerical results 105
    5.8.1 Linear elasticity 105
    5.8.2 Kirchhoff-Love model 108
    5.8.3 Cahn-Hilliard phase field model 110
    5.8.4 Biharmonic problem 112
5.9 Conclusions 117

6 Concluding remarks 120
6.1 Essential results 120
6.2 Future work 121

References 123

Curriculum Vitae 133

List of Figures

2.1 B-spline example: (a) B-spline, and (b) basis functions 6
2.2 Representation of circle with NURBS 7
2.3 (a) Basis functions generated over Ξ^{ξ}, (b) modified and new basis functions when a basis vertex at $\frac{5}{8}$ is inserted. 11
2.4 Modification of basis functions in 2D, (a) basis functions at level ℓ, and (b) basis functions at level $\ell+1$. 13
2.5 Domains in isogeometric analysis 18
2.6 Infinite plate with circular hold example, (a) C^1 continuous NURBS mesh with control points, (b) refined mesh of (a), (c) σ_{xx} displacement of (b), (d) C^0 continuous NURBS mesh with control points, (e) refined mesh of (d), and (f) σ_{xx} displacement of (e). 20
2.7 Spanner example, (a) Splines mesh, (b) displacement and Von Mises stress. 21
2.8 Pinched cylinder example, (a) pinched cylinder, (b) displacement of pinched cylinder. 21
2.9 Hollow sphere example, (a) cross section, (b) analysis result. 22

3.1 (a) Interpolation using the exact interpolation method, (b) interpolation in[1], and (c) interpolation using (3.8). 29
3.2 Multi-resolution example, first row: (a) the moving image and (b) the fixed image. 37
3.2 Multi-resolution example (cont.), second row: result from the first level; (c) deformed image, (d) forward deformation grids, and (e) inverse deformation grids. Third row: result from the second level; (f) deformed image, (g) forward deformation grids, and (h) inverse deformation grids. Fourth row: result from the third level; (i) deformed image, (j) forward deformation grids, and (k) inverse deformation grids. 38

3.3 First row: (a) the fixed Image, (b) the moving Image, and (c) synthetic forward deformation grid. Second row: result from the optical flow method; (d) deformed image, (e) intensity differences, and (f) forward deformation grid. Third row: result from our proposed method; (g) deformed image, (h) intensity differences, and (i) forward deformation grid. 40
3.4 Comparison of our method (using the composition update) with the additive method. First row: (a) the moving image and (b) the fixed image. Second row: results from the additive method; (c) deformed image, (d) forward transformation grid, and (e) inverse transformation grid. Third row: results from the composition method; (f) deformed image, (g) forward transformation grid, and (h) inverse transformation grid. 42
3.5 Registration of synthetic 3D images. (a) The moving image, (b) the fixed image, (c) deformed image computed using $\gamma = 1.54$, and (d) deformed image computed using $\gamma = 0.1$. 43
3.6 Comparison of the average of segmentation result from 10 model sets between the proposed method and the optical flow method. 44
3.7 2D slice of brain image. First row: (a) the moving image and (b) the fixed image. Second row: (c) result produced using our proposed method, (d) result form the optical flow method. 45

4.1 (a) Spanner image, (b) contour from gray scale function, (c) contour from level set function, and (d) contour from level set after resetting the intensity to 0 and 255. 51
4.2 2D example of the boundary matching algorithm, (a)initial mesh and element classification, (b) after adjusting the selected vertices of boundary elements, (c) zoomed in view of the adjusted vertices and (d) the resulting bounding after adjustment. 54
4.3 3D classification of adjust direction (a) vertex belongs to one boundary element, (b) vertex shared by two adjacent elements, and (c) vertex shared by four elements. 55
4.4 Osculating circle example, (a) circle template with repeated control points, (b) alternate circle template, (c) element 10 with repeated control points, (d) element 10 with separated control points, and (d) Yeti footprint and a scaled and rotated circle template at a selected contour point. 57
4.5 Sphere template (the interior 8 elements which approximate the unit sphere are shown) and surface error. 58

4.6 Synthetic benchmark: plate with a hole (a) coarse mesh, (b) σ_{xx} component of the stress field, (c) convergence plot, (d) σ_{xx} stress on a finer mesh. 60
4.7 Yeti footprint example, (a) Yeti footprint image, (b) active area of the PHT-mesh after adjusting vertices, (c) the result using a finer mesh, and (d) the scaled Jacobians of the finer mesh. 61
4.8 Open spanner example, (a) spanner image, (b) result of adaptive refinement, (c) result of volumetric representation using B-splines, and (d) results of uniform refinement along boundary. 62
4.8 Open spanner example (cont.), (e) zoomed-in figure of the upper corned at the end of the spanner, (f) zoomed-in figure of the lower corner at the end of the spanner, (g) the analysis result, and (h) the scaled Jacobians. 63
4.9 Cube with a hollow sphere example: (a) The von Misses stresses on a coarse mesh, and (b) Convergence plots for the energy norm. 64
4.10 Stanford bunny example, (a) level set representation of the Stanford Bunny, (b) the bottom part of the bunny, (c) solid splines representation of the bunny, and (d) the analysis result. 66

5.1 The geometric mapping $G^{(1)}$ and $G^{(2)}$ defined on $[0,1]$. 70
5.2 Isogeometric functions on $\Omega = [0,1]$ (a) with local index, and (b) with global index. 71
5.3 1D Bernstein basis functions and their first derivatives 73
5.4 Bézier coefficient in $\Omega^{(1)}$ and $\Omega^{(2)}$ (a) with local index and (b) with global index. 74
5.5 (a) Smooth solution space obtained from C^1 coupling and (b) comparison of solutions. 75
5.6 (a) C^0 isogeometric functions on a finer mesh (a) local index and (b) global index. 75
5.7 (a) Solution space formed by original functions and new functions (b) convergence plot. 77
5.8 Mapping form parameter space to physical space. 78
5.9 Basis functions in parameter space with local index (C^{-1} continuity). . 80
5.10 Basis functions in the parameter space with global index (C^0 continuity). 80
5.11 Removed functions and original functions in the parameter space. . . 81
5.12 Bézier coefficients on the two patch domain. 82
5.13 Plate with circular hole formed by finer mesh. 84
5.14 Plate with circular hole example (a) relevant coefficients. 84
5.14 Plate with circular hole example (Cont.) (b) removed functions. . . . 85
5.15 Three patch example: (a) A three-patch domain where the Bézier coefficients relevant to the common boundary are represented by red dots, (b) A Type 2 basis function, (c) and (d) x and y derivatives of (b). . . . 88

5.16 Fibe patch circle example: (a) A five-patch circle, (b) a Type 2 basis function, (c) and (d) x and y derivatives of (b). 89
5.17 Circle example: (a) course mesh and distribution of quadratic isogeometric functions (b) quadratic new function, (c) and (d) x and y derivatives of (b). 91
5.18 Circle example: (a) fine mesh and distribution of quadratic isogeometric functions (b) quadratic new function, (c) and (d) x and y derivatives of (b). 92
5.19 Circle example:(a) fine mesh and distribution of cubic isogeometirc functions, (b) C^1 continuous new function, (c) and (d) x and y derivatives of (b). 93
5.20 Localization example: (a) relevant coefficients and minimal MDS, (b) localized basis function, (c)-(d) x and y derivatives of (b). 96
5.21 (a) solution space formed by cubic basis functions, (b) solution space formed by quartic basis functions, and (c) solution space with combination of cubic and quartic basis functions. 98
5.22 Basis functions in partial degree elevations: (a) cubic basis functions, (b) combined basis functions, and (c) quartic basis functions. 100
5.23 Degree elevation in 2D multi-patch domain with extraordinary vertices. 100
5.24 Mapping from parametric space to physical space in 3D. 102
5.25 Removed basis functions in 3D. 103
5.26 Patch test of multi-patch coupling: (a) von Misses stresses, and (b) displacement in x direction. 105
5.27 The infinite plate with a hole example with degree elevation: (a) Plate with hole fine mesh, (b) σ_{yy} component of the stress field, (c) convergence plot, (d) Geometry description and applied boundary conditions. 107
5.28 L-shaped bracket: (a) bracket formed by 18 patches and boundary conditions, (b) stress component σ_{11} of (a). 108
5.29 Kirchhoff-Love plate example: (a) deflection of the plate, and (b) convergence plot. 109
5.30 Pinched cylinder example: (a) cylinder subjected to forces from opposite at the midspan, (b) half cylinder mesh in space, (c) displacement. 109
5.31 Cahn-Hilliard phase field model in a two patch planar domain: (a) two patch mesh, (b)-(i) evolution of the components. 111
5.32 Cahn-Hilliard phase field model on the surface of a sphere: (a) sphere composed of six patches (b) initial distribution of binary components, (c)-(d) evolution of components in chronological order. 113
5.33 Square domain:(a) square composed of two patches, (b) convergence curve, (c) solution of two patch domain, and (d) solution of one patch. 114
5.34 Annulus example (a) annulus composed composed of four patches, (b) convergence curves, (c) solution given by $p = 4$, and (d) error 115

5.35 Circle example (a) circle composed composed of five patches, (b) convergence curves, (c) solution affected by C^1 locking, and (d) solution of partial degree elevation 116
5.36 Cube example (a) solution of two patch domain computed with p=3, and (b) convergence curves. 117
5.37 Cube example: (a) one patch mes, (b) solution computed with p=3, and (c) convergence curves. 118

List of Tables

2.1 Control points and weight of NURBS in Figure 2.2 8

3.1 Comparison of average computation time per iteration 44

5.1 Comparison of condition numbers 106

Nomenclature

Symbols and abbreviations regularly used in the thesis are listed. Symbols less frequently used, or have different meanings in different contexts, are defined where they are used.

Symbol	**Description**
Δ	Laplace operator
∇	Gradient operator
Ξ	Knot vector
x, y, z	Coordinates in physical space
ξ, η, ζ	Coordinates in parameter space
$\bar{\xi}, \bar{\eta}, \bar{\zeta}$	Coordinates in reference space
Ω	Physical domain
$\hat{\Omega}$	Parameter domain
$\bar{\Omega}$	Reference domain
e	Element
Ω_e	Domain of element e in Ω
p	Degree
d	Dimension
$N_{i,p}(\xi)$	B-spline basis functions
P	Control points
$B(\xi)$	B-splines
$R(\xi)$	NURBS
W	Weights in NURBS
ς	Bézier ordinates
$\beta(\xi)$	Bernstein polynomials
$G(\xi)$	Geometric mapping
$T(\xi)$	PHT-splines
$\mathcal{S}$	spline space
$\mathcal{T}$	T-mesh
ν	Poission's ration
$\mathcal{N}$	Cardinal basis function
$\mathcal{B}$	Cardinal B-spline function
$\omega_{i,p}(x)$	isogeometric functions

Abbreviations	**Description**
CAD	Computer Aided Design

IGA	Isogeometric Analysis
PDEs	Partial Differential Equations
CT	Computer Tomography
NURBS	Non-uniform Rational B-splines
PHT-Splines	Polynomial Splines over Hierarchical T-meshes
FFD	Free Form Deformation
SSD	Sum of Square Differences
IBM	Immersed Boudary Method
IFEM	Immersed Finite Element Method
FCM	Finite Cell Method
WEB-splines	Weighted Extended B-splines

Chapter 1

Introduction

1.1 Background, motivation and problem description

The premise of isogeometric analysis (IGA) has been to provide a numerical method by which the basis functions used in computer-aided design (CAD) to represent the geometry are also used to compute the approximate solution fields[2]. IGA offers some appealing features such as exact and efficient geometry representation, simplified mesh refinement, and increased smoothness of the approximate solution. In particular, the basis functions used for the discretization of the approximation space and the geometry description can have smoothness, up to C^{p-1}, where p is the polynomial degree of the basis. This is a significantly higher continuity order compared to the Lagrange polynomials commonly used in the finite element analysis, which are typically restricted to C^0. As a result, better efficiency in terms of degrees of freedom is observed when the solution is suitably smooth, and fourth (as well as higher) order PDEs can be approximated using standard discretization methods.

A challenging task in IGA is obtaining a smooth volumetric mesh that represents a designed object. We note that the design process is often done in the CAD environment, where the geometry is usually provided in boundary form and that it often uses Boolean operations (trim curves and surfaces). This representation is unsuitable for direct use in analysis. Moreover, there are demands for generating analysis-suitable models from sources other than CAD[3], such as voxelized data provided by CT-scans or other specialized imaging equipment. The latter in particular has important uses in medical fields (for the study of the mechanical characteristics of implants) or applications related to digital image correlation. However, it is not easy to generate the analysis model when little *a priori* structure is known about the domain.

A part of the problem in obtaining smooth volumetric meshes is that a tensor-product basis can represent easily “box-like” domains leading to accurate geometry descriptions with few degrees of freedom, however, it is less suitable for representing complex domains. Typically a multi-patch description is required for complex shape representation. Unfortunately, when multiple parameter spaces (patches) are joined together to describe the physical domain, there is typically a loss of continuity which occurs at the patch boundaries. This decrease of smoothness is dictated by the geometry description, where C^0 parameterizations are normally used to deal with kinks and corners in the domain. Multiple patches also need to be used to describe more complex shapes such as those with inclusions, since a single patch is limited by its underlying tensor product structure to describing relatively simple geometries.

There is readily a rich literature on methods for working with voxel-based data as well as dealing with the continuity issues on multi-patch domains. Some of them are suitable for a particular application or shape with simple geometry; whereas others require manual intervention which heavily depends on the user expertise. A simple example is to obtain mesh with desirable shape and continuity by free-form deformation, i.e. adjusting the mesh vertices or control points. This method involves trial and error which is time consuming. Moreover, for the less experienced user, tweaking the mesh by moving the control points may result in irregular shapes or singularities which affect the accuracy in the analysis process. We intended to improve the existing methods in terms of flexibility and efficiency, which leads to the primary objective of this dissertation - to develop self-contained computational methodologies that can be used to obtain smooth volumetric meshes for analysis applications. We set several milestones toward this goal, dealing with interrelated topics. More detailed literature reviews on each topic are given in the corresponding chapter.

B-splines and non-uniform rational B-splines (NURBS), which are often used in CAD systems, are the most common shape functions in IGA. We start the study by determining the application of B-splines for representing voxelized data. An efficient method for boundary representation will be presented, and its implementation in a closely related field - image registration, will be discussed. We then extend the idea to develop a scheme for conversion of a given voxel-based data into an analysis suitable single patch model such that continuity is guaranteed. For multi-patch representation, we propose two constructive approaches for C^1 coupling which are compatible for general geometries.

1.2 Summary of major work

In this dissertation, we present self-contained methods for obtaining smooth volumetric mesh representation of desired shapes as well as the applications. Following are three main parts of this dissertation:

1. Investigate the usage of B-splines for voxel-based images representation and processing.
2. Conversion of the voxelized data into analysis suitable model.
3. C^1 coupling on multi-patch domains.

As a starting point, a convolution method for B-splines representation of curves and surfaces appearing in a voxel-based image is presented. We propose to compute the B-splines coefficients and gradients of the images by using filters. This approach reduces the computational time significantly. Based on the B-splines convolution, we propose an image registration algorithm. In the proposed method, both the image and the deformation are represented in the framework of B-splines. These allow us to use B-splines composition for updating the deformed image and yield smooth transformation mappings. Since the method is in some sense of "isoparametric"; it has the potential to tighten the integration of shape representation and analysis process. Several examples are presented, including registration of medical images.

Furthermore, we propose a method to convert a given voxel-based image to an analysis suitable C^1 volumetric mesh. We determine a matching active domain in a mesh based on the spline representation of the image. The mesh is constructed by using a hierarchical basis of C^1 cubic polynomials which allows more flexibility in the refinement and boundary representation. In order to obtain good geometry approximation, we use a circle temple to adjust the control points along the boundary of the active domain. The adjustment is made according to the normal and curvature of the B-splines image representation. This idea is developed based on differential geometry of curves, where the osculating circle can best describe the geometry of a smooth curve. For 3D, we used sphere template to facilitate the geometry approximation. To avoid singular points in the mesh, we implement Laplacian smoothing to untangle the irregular shapes. The appealing features of the proposed method are that complex domains can be handled in a single-patch framework and it does not involves numerical optimization. We demonstrate the method for handling complex shapes in several 2D and 3D examples.

Finally, we propose two constructive approaches for the approximation of C^1 solution space on the multi-patch domain, namely the Bézer ordinates approach and spline approach. The proposed methods allow the modeling of solutions to fourth order PDEs on complex geometric domains, provided that the given patches have G^1 continuity. The methods are compatible with 2D, 3D, as well as surfaces in space. In the Bézier ordinates approach, the C^1 basis functions are given in terms of Bézier-Bernstein polynomial. In the splines approach, they are defined as a linear combination of C^0 basis functions. The Bézier ordinates and splines coefficients are numerically computed to fulfill the continuity constraints derived according to the geometry of the physical domain. For some geometries, the over-constrained solution space will lead to C^1 locking. We present a partial degree elevation approach to overcome this issue, where the high computation cost for degree elevating the entire domain can be avoided. The Bézier ordinates approach which offers more degree of freedoms is useful for handling domains with complex shapes, while the splines approach which is more computationally efficient is particularly suitable for higher dimensions. We show the application in Kirchoff-Love shell models, Cahn-Hilliard phase field, and biharmonic problems.

1.3 Outline of dissertation

Following the introduction, in Chapter 2 we give a brief overview of IGA and splines. We present the B-splines, NURBS, Bézier decomposition, and Polynomial Splines over Hierarchical T-meshes (PHT Splines) which are used in the later chapters. We also discussed the application for solving the linear elasticity problems.

In Chapter 3, we present a method for B-splines images representation via convolution and filtering. We then give an introduction of image registration framework. It was followed by proposing a method for non-rigid diffeomorphic registration, where B-splines composition for updating the deformed images is presented.

Volumetric parametrization is presented in Chapter 4. The discussion starts with the determination of the active domain in the PHTmesh according to the boundary of the given voxel-based image. Next, we present the construction of a circle template and its usage for geometry approximation. It is followed by the discussion of sphere template for 3D application. Finally, Laplacian smoothing for untangling the irregular shapes and refinement method for improving the result are discussed.

In Chapter 5, we propose the Bézier ordinates approach and spline approach for construction of C^1 continuous basis functions on the multi-patch domain. Firstly, we describe the preliminaries by using a simple 1D example. We then discuss the extension of the ideas to 2D, surfaces in space, and 3D. We also present a localization method using minimal determining set and partial degree elevation approach to overcome C^1 locking.

Finally, the summary of the major conclusion of this study is given in Chapter 6. We also briefly describe possibilities for future work related to the applications of splines in image processing, volumetric parametrization and smooth approximations on complex geometries.

Chapter 2

Isogeometric analysis framework

We will present the framework of IGA[4] in this chapter. In general, IGA is formulated based on B-splines and NURBS. The fundamentals of these splines are presented in Section 2.1. Other splines which are also used in IGA and provide more flexibility are T-splines. They were introduced in[5] to allow meshes with T-junctions. Relevant studies of T-splines in IGA applications can be found in[6],[7] and[8]. PHT-splines[9] are developed based on T-splines; they have improved local refinement property.

The PHT-spline and related computational techniques are presented in Section 2.3. Meanwhile, the Bézier decomposition which is also used in the later chapter is given in Section 2.2. We also describe the application of spline for analysis by using the Galerkin method and linear elasticity problem in Section 2.4. Some numerical examples are provided in Section 2.5, and a conclusion is given in Section 2.6.

2.1 B-splines and NURBS

Given a knot vector $\Xi^{\xi} = \{\xi_0, \xi_1, \cdots, \xi_{p+n+1}\}$ in the parametric domain, the set of basis functions $N_{i,p}(\xi)$ of polynomial degree p are defined recursively as follows:

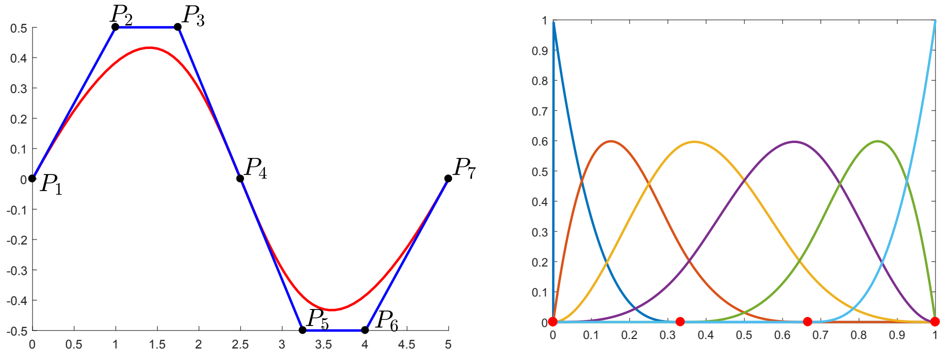
$$\begin{aligned} N_{i,0}(\xi) &= \begin{cases} 1 & : \text{if} \quad \xi_i \le \xi \le \xi_{i+1} \\ 0 & : \text{otherwise} \end{cases} \\ N_{i,p}(\xi) &= \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi), \end{aligned} \tag{2.1}$$

where $\xi_i \in \mathbb{R}$ is the ith knot. The knot-vector and the polynomial degree determine the basis functions. For a knot-vector with no repeated values, the continuity of the basis is C^{p-1}, i.e., the first $p-1$ derivatives are continuous at all points in the domain. The knot values in a knot vector may be repeated, in which case the continuity order at that point is reduced by 1 for each repetition.

A B-spline $B(\xi)$ of degree p is defined in terms of basis functions as:

$$B(\xi) = \sum_{i=1}^{n} P_i N_{i,p}(\xi) \tag{2.2}$$

where P_i denotes the control points. The points on the spline corresponding to $N(\xi_i)$ are called the knot points. An example of a B-spline of degree $p = 3$ defined over the knot vector $\Xi^{\xi} = \{0,0,0,0,1/3,2/3,1,1,1,1\}$ is shown in Figure 2.1. In Figure 2.1(a), the spline is represented by the red curve, the blue lines represent the control polygon, and the black dots are the control points. The corresponding basis functions are shown in Figure 2.1(b), where the red dots represent the knots. B-spline have several appealing properties including geometry invariance under translation and rotation. Besides, the basis functions form a partition of unity. Moreover, the *h*-/*p*-/*k*-refinement properties of the basis function makes them suitable for adaptive analysis with B-splines.



(a)

(b)

Figure 2.1: B-spline example: (a) B-spline, and (b) basis functions

Tensor products of 1D B-spline basis functions are used to represent the surface and volume. For 2D, it is defined as:

$$B(\xi,\eta) = \sum_{i=1}^{n} \sum_{j=1}^{n} P_{ij} N_{i,p}(\xi) N_{j,p}(\eta) \tag{2.3}$$

where $P_{ij} = \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix}$ are the 2D control points corresponding to the basis functions. It can be extended directly to 3D as:

$$B(\xi,\eta,\zeta) = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} P_{ijk} N_{i,p}(\xi) N_{j,p}(\eta) N_{k,p}(\zeta) \tag{2.4}$$

and the control points in 3D is given by $P_{ijk} = \begin{bmatrix} x_{ijk} \\ y_{ijk} \\ z_{ijk} \end{bmatrix}$.

A NURBS $R(\xi) \in \mathbb{R}^2$ is the projection of a B-spline in $\mathbb{R}^3$. It is defined by the basis functions and points with weights as:

$$R(\xi) = \sum_{i=1}^{n} P_i \frac{W_i N_{i,p}(\xi)}{\sum_{\hat{i}=1}^{n} W_{\hat{i}} N_{\hat{i},p}(\xi)} \tag{2.5}$$

where W_i represent the weights. The advantage of NURBS is that it allows exact geometry representation which leads to more accurate result in modeling and analysis. Figure 2.2 shows the example of the boundary of a circle formed by NURBS of degree 2. In Figure 2.2 the red curve is the NURBS, the black dots denote the control points, and the blue lines represent the control polygon. The knot vector used to generate the circle is given by $\Xi^{\xi} = \{0,0,0,1/4,1/4,1/2,1/2,3/4,3/4,1,1,1\}$, while the control points and the corresponding weight are listed in Table 2.1.

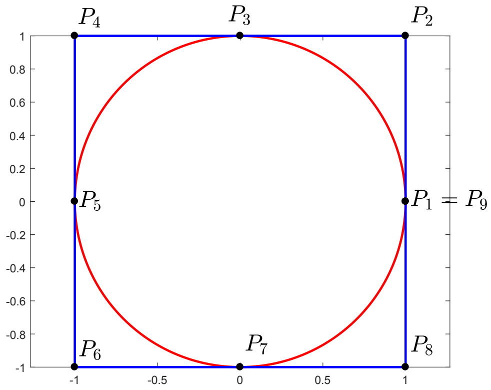


Figure 2.2: Representation of circle with NURBS

2.2 Bézier decomposition

Since the B-Splines are non-local (i.e., they span up to $p+1$ elements) or knot-spans, their implementation in standard finite element codes may be cumbersome. Therefore, it is advantageous to obtain a local representation of the values of the basis functions corresponding to each element independently. This provides an element structure for the isogeometric analysis that is compatible with existing finite element computations [10]. We note that this only applies for $p > 1$, as linear B-Splines ($p = 1$) are equivalent to the standard linear finite element basis in 1D or on tensor-product quadrilateral or

x	y	weight
1	0	1
1	1	$\frac{1}{\sqrt{2}}$
0	1	1
-1	1	$\frac{1}{\sqrt{2}}$
-1	0	1
-1	-1	$\frac{1}{\sqrt{2}}$
0	-1	1
1	-1	$\frac{1}{\sqrt{2}}$
1	0	1

Table 2.1: Control points and weight of NURBS in Figure 2.2

hexahedral meshes.

Local representation of the basis functions on each element can be accomplished by using Bézier decomposition which is based on the idea of knots insertion. Suppose the p degree basis functions are defined on the knot $\Xi^{\xi} = \{\xi_1, \xi_2, \dots, \xi_{n+p+1}\}$. Bézier decomposition requires all the interior knots of Ξ^{ξ} to be repeated by p times. Let $\{\tilde{\xi_1}, \tilde{\xi_2}, \dots, \tilde{\xi_m}\}$ be the set of repeated knots, for each knots $\tilde{\xi_j} \in [\xi_k, \xi_{k+1}[$, $j = \{1, 2, \dots m\}$, we compute ς^j as:

$$\varsigma^j = \begin{bmatrix} \rho_1 & 1-\rho_2 & 0 & \dots & & & 0 \\ 0 & \rho_2 & 1-\rho_3 & 0 & \dots & & 0 \\ 0 & 0 & \rho_3 & \rho_4 & 0 & \dots & 0 \\ \vdots & & & & & & \\ 0 & \dots & & & 0 & \rho_{(n+j-1)} & 1-\rho_{(n+j)} \end{bmatrix}, \tag{2.6}$$

where ρ_i, $i = 1, 2, \dots, n+j$ is defined as:

$$\rho_i = \begin{cases} 1 & : \text{if} \quad 1 \le i \le k-p \\ \frac{\tilde{\xi} - \xi_i}{\xi_{i+p} - \xi_i} & : \text{if} \quad k-p+1 \le i \le k \\ 0 & : \text{if} \quad i \ge k+1 \end{cases}. \tag{2.7}$$

We then define the basis functions $N(\xi) = \{N_{i,p}(\xi)\}_{i=1}^{n}$ as linear combination of Bernstein polynomials ($\beta(\xi) = \{\beta_{i,p}(\xi)\}_{i=1}^{n+m}$):

$$N(\xi) = \varsigma\beta(\xi), \tag{2.8}$$

where ς is know as the Bézier ordinates, it is a $n \times (n \times m)$ matrix given by:

$$\varsigma = (\varsigma^m)^T (\varsigma^{m-1})^T \dots (\varsigma^1)^T. \tag{2.9}$$

For a particular nonzero knot-spans $[\xi_i, \xi_{i+1}]$, the Bézier decomposition enable the basis functions over it to be defined locally as:

$$N^{\hat{i}}(\xi) = \varsigma^{\hat{i}} \beta^{\hat{i}}(\xi), \tag{2.10}$$

where $\varsigma^{\hat{i}}$ refer to the corresponding $p+1 \times p+1$ entries in ς for this span. In this study, we compute the Bernstein polynomial $\beta^{\hat{i}}(\xi) = \{\beta_{j,p}(\xi)\}_{j=1}^{p+1}$ in a reference space $\bar{\Omega} = [-1, 1]$ as follows:

$$\bar{\beta}_{j,p}(\bar{\xi}) = \frac{1}{2^p} \binom{p}{j-1} (1-\bar{\xi})^{p-(j-1)} (1-\bar{\xi})^{j-1}, \; j = 1, 2, \dots, p+1. \tag{2.11}$$

$\binom{p}{j-1}$ is the binomial coefficients defined as:

$$\binom{p}{j-1} = \frac{p!}{(j-1)!(p+1-j)!}, \; 1 \leq j \leq p+1, \tag{2.12}$$

and the mapping $G_\xi : [-1, 1] \to [\xi_i, \xi_{i+1}]$ is given by:

$$G_\xi(\bar{\xi}) = \frac{1}{2}(\xi_{i+1} - \xi_i)\bar{\xi} + \frac{1}{2}(\xi_i + \xi_{i+1}). \tag{2.13}$$

A similar concept can be applied for Bézier decomposition in a higher dimension. In 2D, the local Bézier ordinates $\varsigma^{\widehat{ij}}$ is defined as:

$$\varsigma^{\widehat{ij}} = \varsigma_\xi^{\hat{i}} \otimes \varsigma_\eta^{\hat{j}},$$

and in 3D $\varsigma^{\widehat{ijk}}$ is given by:

$$\varsigma^{\widehat{ijk}} = \varsigma_\xi^{\hat{i}} \otimes \varsigma_\eta^{\hat{j}} \otimes \varsigma_\zeta^{\hat{k}},$$

where $\varsigma_\xi^{\hat{i}}$, $\varsigma_\eta^{\hat{i}}$ and $\varsigma_\zeta^{\hat{k}}$ are the $\hat{\imath}$th, $\hat{\jmath}$th, and $\hat{k}$th extraction operators in ξ, η and ζ directions respectively.

2.3 Hierarchical refinement with PHT-splines

The polynomial splines introduced in[9] are related to B-splines of reduced continuity over hierarchical T-meshes. Local refinement is an appealing property of the PHT-splines, which makes their use suitable for fitting arbitrary domains. Isogeometric analysis methods using PHT-splines and various refinement strategies have been presented in[11,12,13,14].

In 1D, the initial PHT-spline representation is of the form:

$$T(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) P_i, \tag{2.14}$$

where P_i are the control points and $N_{i,p}(\xi)$ are the cubic B-spline basis functions defined over the knot vector

$$\Xi^{\xi} = \left\{0, \ldots, 0, \frac{1}{n}, \frac{1}{n}, \ldots, \frac{n-1}{n}, \frac{n-1}{n}, 1, \ldots, 1\right\}.$$

The begin and end knots are repeated $p+1$ times, while the interior knots are repeated once. Therefore PHT splines have C^1 continuity. For each interior vertex ξ_i, there are two basis functions supporting $[\xi_{i-1}, \xi_{i+1}]$; we call these points *basis vertices*. The associated basis functions are determined by the local knot vectors $(\xi_{i-1}, \xi_{i-1}, \xi_i, \xi_i, \xi_{i+1})$, and $(\xi_{i-1}, \xi_i, \xi_i, \xi_{i+1}, \xi_{i+1})$ respectively. For each newly inserted basis vertex, there will be two additional new basis functions correspond to it. Therefore two new control points need to be calculated, while the previous control points stay fixed. In particular, the refined PHT-spline can be expressed as follows:

$$\hat{T}(\xi) = \sum_{i=1}^{n} \hat{N}_{i,p}^{\ell}(\xi) \cdot P_i + \sum_{k=1}^{2} \hat{N}_{k,p}^{\ell+1}(\xi) \cdot P_k, \tag{2.15}$$

where $\hat{T}(\xi)$ is the new geometry parametrization, $\hat{N}_{i,p}^{\ell}(\xi)$ are the (possibly modified) basis functions from the previous level ℓ and $\hat{N}_{k,p}^{\ell+1}(\xi)$ are the new basis functions. If the geometry parametrization is fixed, then $\hat{T}(\xi) = B(\xi)$.

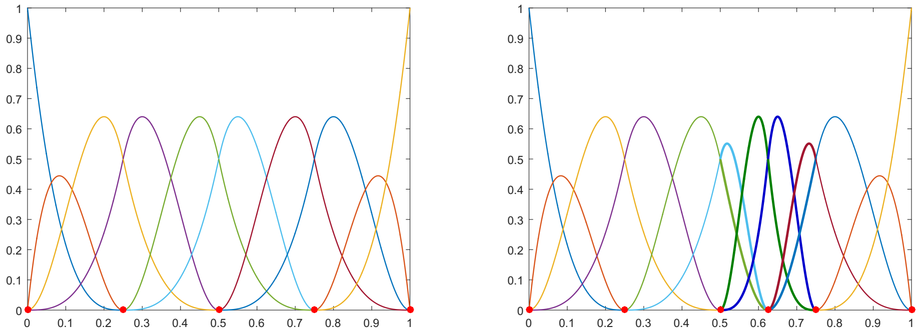
In order to modify the basis functions, the Bézier ordinates representation is used. Suppose an element e is refined at level ℓ. The corresponding Bézier ordinates of e are subdivided using the DeCasteljau algorithm (see formula (14.15) in[15]) into two parts. The basis function $N_{i,p}^{\ell}(\xi)$ is modified to $\hat{N}_{i,p}^{\ell}(\xi)$ by resetting the Bézier ordinates associated with the new knot to zero. Let the new knot be given by $\xi_{\hat{i}}$, then the two new basis functions are defined through the knot vector $(\xi_{\hat{i}-1}, \xi_{\hat{i}-1}, \xi_{\hat{i}}, \xi_{\hat{i}}, \xi_{\hat{i}+1})$, and

$(\xi_{\hat{i}-1}, \xi_{\hat{i}}, \xi_{\hat{i}}, \xi_{\hat{i}+1}, \xi_{\hat{i}+1})$ respectively.

Figure 2.3 illustrates the change of basis functions when a new knot is inserted, which is different from standard B-spline knot insertion. The basis functions in Figure 2.3(a) are defined by the knot vector

$$\Xi^{\xi} = \{0, 0, 0, 0, \frac{1}{4}, \frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}, \frac{3}{4}, 1, 1, 1, 1\},$$

and Figure 2.3(b) shows the modified basis functions when a basis vertex at $\frac{5}{8}$ is inserted. We observe that in this case, the only modifications to the basis functions take place on the element that is being refined. Even in higher dimensions, an element can be refined by changing the Bézier representation of the basis on the current element and its immediate neighbor only. The reduced overlap between the basis functions allows for more efficient and localized refinement schemes compared to other local spline bases, such as hierarchical B-splines and T-splines.



(a) (b)

Figure 2.3: (a) Basis functions generated over Ξ^{ξ}, (b) modified and new basis functions when a basis vertex at $\frac{5}{8}$ is inserted.

We note that in Figure 2.3, all functions except the modified basis functions which have support on the boundary of the refined knot span are B-splines. A PHT-spline can be converted to a B-spline by repeatedly inserting vertices until all the knot spans are at the same refinement level. Also similarly to T-splines, a PHT-spline mesh is determined entirely in terms of the local knot vectors and the Bézier ordinates of the basis functions on each knot span. The definition of basis functions for higher dimension PHT-splines can be generalized using a similar method. For dimension d, a new knot (basis vertex) introduces 2^d new basis functions.

2.3.1 PHT-Spline Space

For simplicity, we briefly describe the concept of T-meshes in 2D. Referring to[16], the spline space over a given T-mesh is defined as:

$$\mathcal{S}(\mathcal{T}) :\ = \{s \in C^{1,1}(\Omega) : s|_e \in \mathbb{P}_3 \,\text{for any}\, e \in \mathcal{T}\}, \tag{2.16}$$

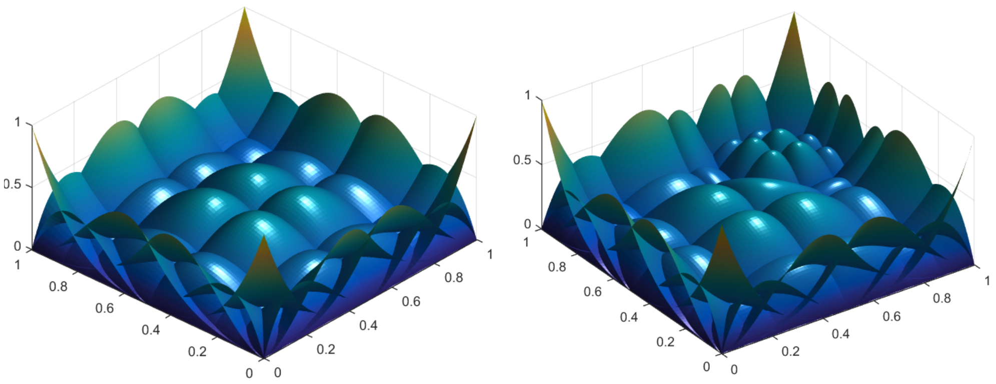
where $\mathbb{P}_3$ is the space of all cubic polynomials, $\mathcal{T}$ denotes the mesh, and $C^{1,1}(\Omega)$ is the space consisting of all bivariate functions which are continuous in Ω with order 1 along both spacial directions.

Refinement is always done by "cross-insertion", i.e. splitting a square element into four elements in 2D and a cube element into eight sub-cubes in 3D. The basis vertices in 2D are those located at the intersection of four elements in the interior, and all the boundaries and corner vertices. The basis vertices in 3D are located at the intersection of eight elements in the interior, four elements on the faces as well as all the edges and corner vertices.

Since the PHT-spline is a cubic spline with C^1 continuity, the dimension of the PHT-spline space over the mesh $\mathcal{T}$ is given by:

$$\dim(\mathcal{S}(\mathcal{T})) = 2^d \cdot (\text{number of basis vertices}).$$

We denote the spline space at level ℓ as $\mathcal{S}(\mathcal{T}_\ell)$. At the new level, the new spline space $\mathcal{S}(\mathcal{T}_{\ell+1})$ will be changed such that the basis functions inherited from the level ℓ will have minimal changes when compared to the basis functions at level $\ell+1$. In particular, they have the same shape as those in level ℓ outside the elements that contain new basis vertices. This can be seen by comparing the shape of basis functions before and after a new vertex is inserted (see Figure 2.4(a) and Figure 2.4(b)). This is an essential feature of the PHT-splines, which reduces the overhead associated with local refinement. The T-vertices (located at T-junctions) have no basis function associated with them. We note that the neighbor of a refined element only needs to be changed when a T-vertex is converted to a basis vertex. This is different from T-splines where the addition of an edge to the mesh can trigger additional refinements further away from the inserted edge. In 3D, the modification of existing basis functions and the computation of new ones can be done using a similar method, taking advantage of the local tensor product structure of the refined element. Compared to 2D, eight basis functions are supporting each vertex instead of four. Thus, adding a new vertex introduces eight new basis functions to the spline space. The modification of basis functions and computation of new basis functions can be done similarly.



(a) (b)

Figure 2.4: Modification of basis functions in 2D, (a) basis functions at level ℓ, and (b) basis functions at level $\ell+1$.

2.3.2 Computing the control points

For a linear mapping between the parameter space and the physical space, the control points of the initial mesh $(\ell = 1)$ are set as the locations of the Greville Abscissae, while the control points at level $\ell > 1$ can be computed using the *geometric information* of the basis functions[9]. The location of the basis vertex determines the geometric information in the physical space and the values of the derivatives of the mapping evaluated at the basis vertex. In 1D, the geometric information of $T(\xi)$ is given by the linear operator:

$$\mathscr{L}T(\xi) = \left(T(\xi), \frac{\partial T(\xi)}{\partial \xi}\right).$$

Suppose the spline is defined as:

$$T(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) \cdot P_i\,, \tag{2.17}$$

then the geometric information of the spline at an arbitrary knot ξ_a denoted $(\mathscr{L}T(\xi_a))$ is given by:

$$\begin{aligned} \mathscr{L}T(\xi_a) &= \textstyle\sum_{j=1}^{2} \mathscr{L}(N_{i_j,p}(\xi_a) \cdot P_i) \\ &= P \cdot \mathscr{N}\,, \end{aligned} \tag{2.18}$$

where i_1 and i_2 represent the two basis indices corresponding to the knot ξ_a. Note that P and $\mathscr{N}$ are $1{\times}2$ and $2{\times}2$ matrices respectively. Hence, P can be solved from:

$$P = \mathscr{L}T(\xi_a) \cdot \mathscr{N}^{-1}\,. \tag{2.19}$$

This method can be extended to finding P when dealing with higher dimensions. We refer to[9] for an explanation of computing P in 2D. Here we discuss the computation of control points in 3D. The geometric information in 3D is defined as follows:

$$\begin{aligned}\mathscr{L}T(\xi,\eta,\zeta) = [T(\xi,\eta,\zeta),\ &\tfrac{\partial T(\xi,\eta,\zeta)}{\partial \xi},\ \tfrac{\partial T(\xi,\eta,\zeta)}{\partial \eta},\ \tfrac{\partial T(\xi,\eta,\zeta)}{\partial \zeta},\\ &\tfrac{\partial T^2(\xi,\eta,\zeta)}{\partial \xi \partial \eta},\ \tfrac{\partial T^2(\xi,\eta,\zeta)}{\partial \xi \partial \zeta},\ \tfrac{\partial T^2(\xi,\eta,\zeta)}{\partial \eta \partial \zeta},\ \tfrac{\partial T^3(\xi,\eta,\zeta)}{\partial \xi \partial \eta \partial \zeta}].\end{aligned} \tag{2.20}$$

Thus, for a basis vertex (ξ_a, η_a, ζ_a), the control points can be obtained from the relation:

$$P = \mathscr{L}T(\xi_a, \eta_a, \zeta_a) \cdot \mathscr{N}^{-1}. \tag{2.21}$$

Taking advantages of the closed form of B-splines and assuming the new basis vertex is in the middle of the refined element, elementary operations show that $\mathscr{N}$ is a 8×8 matrix given by:

$$\mathscr{N} = W \otimes [V \otimes U], \tag{2.22}$$

where $\otimes$ is the Kronecker product. Moreover

$$U = \begin{bmatrix} (1-\lambda) & \lambda \\ -\delta & \delta \end{bmatrix}, \quad V = \begin{bmatrix} (1-\mu) & \mu \\ -\vartheta & \vartheta \end{bmatrix}, \text{ and } W = \begin{bmatrix} (1-\nu) & \nu \\ -\gamma & \gamma \end{bmatrix},$$

where $\delta = \frac{1}{\Delta\xi_1+\Delta\xi_2}$, $\vartheta = \frac{1}{\Delta\eta_1+\Delta\eta_2}$, $\gamma = \frac{1}{\Delta\zeta_1+\Delta\zeta_2}$, $\lambda = \delta\Delta\xi_1$, $\mu = \vartheta\Delta\eta_1$, and $\nu = \gamma\Delta\zeta_1$. $\Delta\xi_i$, $\Delta\eta_i$ and $\Delta\zeta_i$, $i = 1, 2$ are the differences between ξ_a, η_a, and ζ_a and the neighbor knots in the negative and positive direction respectively.

2.4 Analysis using splines

In this section, we discuss solving boundary problem using spline. Suppose we would like to solve a boundary value problem as below:

$$\begin{aligned}\Delta u &= f \,\text{in}\, \Omega,\\ u &= g \,\text{on}\, \partial\Omega.\end{aligned} \tag{2.23}$$

The weak form of (2.23) is obtained by considering trial solutions ($u : \Omega \to \mathbb{R}$), multiplying both sides by the weighting function (w) and integration by parts:

$$\int_\Omega \nabla w \nabla u \, d\Omega = \int_\Omega w f \, d\Omega. \tag{2.24}$$

The collection of trial solutions (S) is given by:

$$S = \{u | u \in H^1(\Omega), u|_{\partial\Omega} = g\}. \tag{2.25}$$

The weighting functions are denoted by a set $\mathcal{V}$ as:

$$\mathcal{V} = \{w | w \in H^1(\Omega), w|_{\partial\Omega} = 0\}, \tag{2.26}$$

where $H^1(\Omega)$ is the Sobolev space. The weak form can be rewritten as:

$$a(w,u) = L(w), \tag{2.27}$$

where

$$a(w,u) = \int_\Omega \nabla w \nabla u \, d\Omega, \tag{2.28}$$

and

$$L(w) = \int_\Omega w f \, d\Omega. \tag{2.29}$$

The solution to (2.24) and (2.27) is known as the weak solution. In the next section, we will discuss finding the solution using Galerkin method.

2.4.1 Galerkin method

The weak form of the problem can be converted into a system of algebraic equations using the Galerkin method. The main concept of Galerkin method is to approximate S and $\mathcal{V}$ with the corresponding finite-dimensional approximations S^h and $\mathcal{V}^h$, which are typically the subsets of S and $\mathcal{V}$:

$$S^h \subset S$$
$$\mathcal{V}^h \subset \mathcal{V}.$$

Suppose $g^h \in S^h$ such that $g^h|_{\partial\Omega} = g$, then there is an unique $v^h \in \mathcal{V}^h$ for every $u^h \in S^h$ such that

$$u^h = v^h + g^h. \tag{2.30}$$

Assume that g^h exists, the Galerkin form of the problem can therefore be written as: given g^h, find $u^h = v^h + g^h$ such that for all w^h in $\mathcal{V}^h$,

$$a(w^h, u^h) = L(w^h). \tag{2.31}$$

Referring to (2.30), (2.31) can be rewritten as:

$$a(w^h, v^h) = L(w^h) - a(w^h, g^h). \tag{2.32}$$

In IGA, the solution space is formed by linear combination of a set of basis functions $N_A : \Omega \rightarrow \mathbb{R}, A = 1, \ldots, n_{nb}$ where for all $w^h \in \mathcal{V}^h$ there exists $c_A, A = 1, \ldots, n_{eq}$ such that

$$w^h = \sum_{A=1}^{n_{eq}} N_A c_A. \tag{2.33}$$

The function g^h is given similarly as:

$$g^h = \sum_{A=n_{eq}+1}^{n_{nb}} N_A g_A. \tag{2.34}$$

By using (2.33) and (2.34) the trial function can be expressed as:

$$u^h = \sum_{A=1}^{n_{eq}} N_A d_A + \sum_{A=n_{eq}+1}^{n_{nb}} N_A g_A. \tag{2.35}$$

By substituting (2.33) and (2.35) into (2.32), we obtain:

$$\sum_{A=1}^{n_{eq}} c_A \left(\sum_{B=1}^{n_{eq}} a(N_A, N_B) d_B - L(N_A) + a(N_A, g^h) \right) = 0. \tag{2.36}$$

Since c_A are arbitrary, we have:

$$\sum_{B=1}^{n_{eq}} a(N_A, N_B) d_B = L(N_A) - a(N_A, g^h). \tag{2.37}$$

Let K_{AB} and F_A be

$$K_{AB} = a(N_A, N_B), \tag{2.38}$$

and

$$F_A = L(N_A) - a(N_A, g^h). \tag{2.39}$$

They can be written in matrix form as:

$$\mathbf{K} = [K_{ab}], \tag{2.40}$$

$$\mathbf{F} = \{F_a\}, \tag{2.41}$$

$$\mathbf{d} = \{d_a\}. \tag{2.42}$$

Therefore, for $A, B = 1, \ldots, n_{eq}$, the algebraic equation can be express as:

$$\mathbf{Kd} = \mathbf{F}, \tag{2.43}$$

where **K** is the stiffness matrix and **F** is the force vector. The displacement vector **d** can be solve from:

$$\mathbf{d} = \mathbf{K}^{-1}\mathbf{F}. \tag{2.44}$$

Finally, the solution can be obtained by substituting (2.44) into (2.35).

2.4.2 Linear elasticity

In this section, we show the examples of application of spline-based IGA on the problem of linear elasticity. Given a domain Ω, bounded by $\Gamma = \overline{\Gamma_t \cup \Gamma_u}$, $\Gamma_t \cap \Gamma_u = \emptyset$, where displacements and tractions ($\bar{\mathbf{t}}$) are prescribed on Γ_u and Γ_t respectively. The weak form of a linear elastostatics problem is given by: find the displacement $\mathbf{u}$ in the trial space, such that for all test functions $\delta\mathbf{u}$ in the test place satisfy the following statement:

$$\int_\Omega \varepsilon(\mathbf{u}) : \mathbf{D} : \varepsilon(\delta\mathbf{u}) d\Omega = \int_{\Gamma_t} \bar{\mathbf{t}} \cdot \delta\mathbf{u} d\Gamma + \int_\Omega \mathbf{b} \cdot \delta\mathbf{u} d\Omega. \tag{2.45}$$

In (2.45), $\mathbf{D}$ is known as the elasticity matrix, while $\mathbf{b}$ and $\varepsilon = \frac{1}{2}(\nabla\mathbf{u} + \nabla^T\mathbf{u})$ denote the body force and displacement gradient respectively. By using the the Galerkin method, $\mathbf{u}$ and $\delta\mathbf{u}$ can be expressed as:

$$\mathbf{u}(x) = \sum_I^n N_I(\xi)\mathbf{u}_I, \tag{2.46}$$

and

$$\delta\mathbf{u}(x) = \sum_I^n N_I(\xi)\delta\mathbf{u}_I. \tag{2.47}$$

Note that N_I is the basis functions, n represents the number of control points, $\mathbf{u}_I = [u_{xI}, u_{yI}]^T$ is the nodal unknown vector, and $\delta\mathbf{u}_I$ denotes the nodal displacement variations.

By substituting (2.46) and (2.47) into (2.45), we obtain the discrete equation:

$$\mathbf{K}\mathbf{u} = \mathbf{f}, \tag{2.48}$$

where

$$\mathbf{K}_{IJ} = \int_\Omega \mathbf{B}_I^T \mathbf{D} \mathbf{B}_J d\Omega, \tag{2.49}$$

and

$$\mathbf{f}_I = \int_{\Gamma_t} N_I \bar{\mathbf{t}} d\Gamma + \int_\Omega N_I \mathbf{b} d\Omega. \tag{2.50}$$

The strain-displacement $\mathbf{B}_I$ in 2D is defined as:

$$\mathbf{B}_I = \begin{bmatrix} N_{I,x} & 0 \\ 0 & N_{I,y} \\ N_{I,y} & N_{I,x} \end{bmatrix}, \tag{2.51}$$

where $N_{I,x}$ represents the first order partial derivatives of N_I with respect to x. We can then obtain the solution $\mathbf{u}$ by solving the linear system (2.48).

Note that spline basis functions are defined in the parameter space, while the quadrature rule are defined in the reference domain. The relation between the domains is illustrated in Figure 2.5, and the domain integral in (2.38) and (2.50) can be computed as follows:

$$\begin{aligned}\int_{\Omega} f(x,y)d\Omega &= \bigcup_{e=1}^{nel} \int_{\Omega_e} f(x,y)d\Omega_e \\ &= \bigcup_{e=1}^{nel} \int_{\Omega_e} f(x(\xi),y(\eta))|J_\xi|d\hat{\Omega}_e \\ &= \bigcup_{e=1}^{nel} \int_{\overline{\Omega}_e} f(\overline{\xi},\overline{\eta})|J_\xi||J_{\overline{\xi}}|d\overline{\Omega}_e, \end{aligned} \tag{2.52}$$

where $\bigcup$ is the assembly operator, and *nel* represents the number of finite elements which are defined as non-zeros knot spans. Meanwhile, $|J_\xi|$ denotes the determinant of Jacobian of the transformation from the parametric domain to the physical domain, while $|J_{\overline{\xi}}|$ represents the determinant of Jacobian of the transformation from the reference element to the parametric domain. The reference element contains the Bernstein polynomials of degree p, as described in Section 2.2. The integration can then be computed using Gauss-Legendre quadrature.

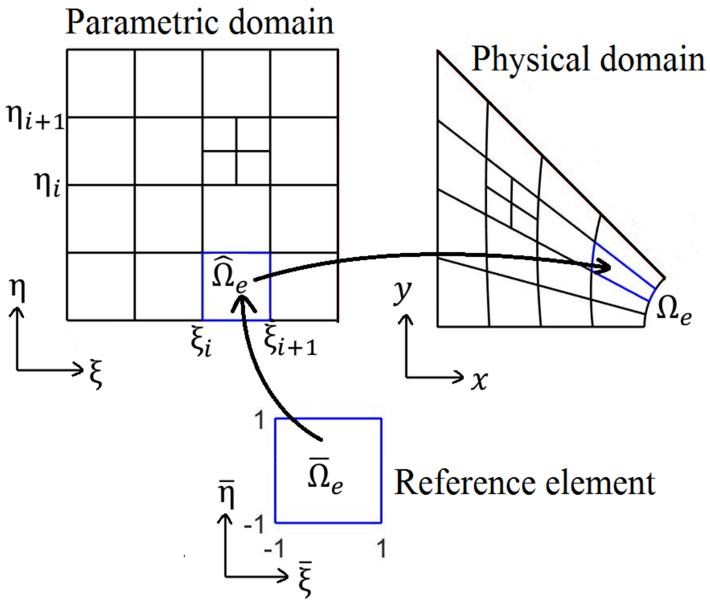


Figure 2.5: Domains in isogeometric analysis

2.5 Numerical examples

In this section, we show some 2D and 3D benchmark examples of spline-based isogeometric analysis for linear elasticity problem.

2.5.1 Infinite plate with circular hole

Firstly, we show the 2D example of an infinite plate with circular hole. The plate is subject to constant in-plane tension at infinity, and the exact solution of this problem is given by:

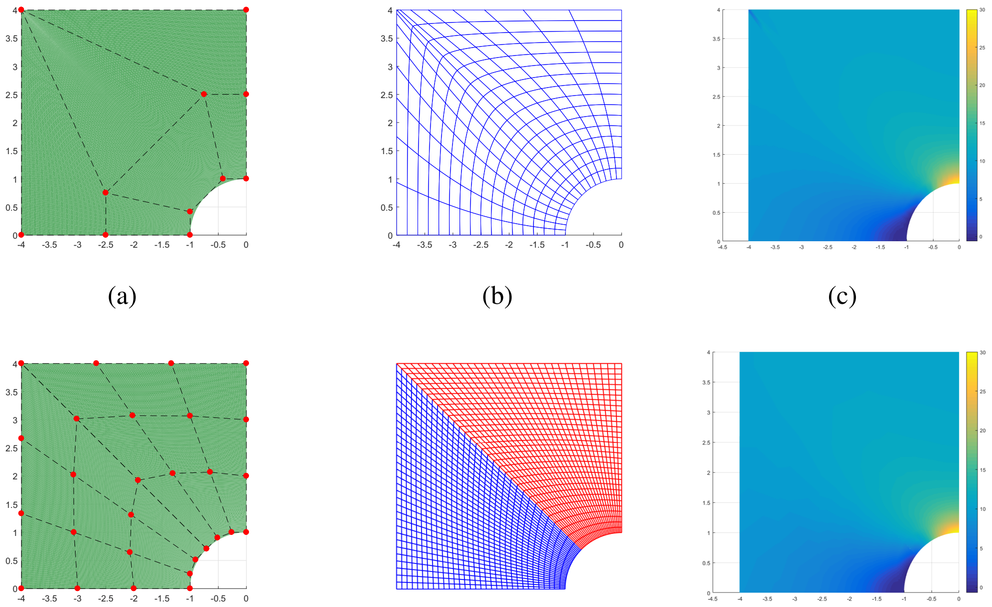
$$\begin{aligned} \sigma_{rr} &= \frac{\sigma_\infty}{2}\left(1-\frac{R^2}{r^2}\right)+\frac{\sigma_\infty}{2}\left(1-\frac{4R^2}{r^2}+\frac{3R^4}{r^4}\right)\cos 2\theta \\ \sigma_{\theta\theta} &= \frac{\sigma_\infty}{2}\left(1+\frac{R^2}{r^2}\right)-\frac{\sigma_\infty}{2}\left(1+\frac{3R^4}{r^4}\right)\cos 2\theta \\ \sigma_{r\theta} &= -\frac{\sigma_\infty}{2}\left(1+\frac{2R^2}{r^2}-\frac{3R^4}{r^4}\right)\sin 2\theta\,, \end{aligned} \tag{2.53}$$

where σ_∞ is the remote stress. R is the hole radius and r and θ are polar coordinates with respect to the center of the hole. In this example, we set $\sigma_\infty = 10$ and $R = 1$. The domain of this problem is infinite, and the problem is symmetric about the origin. Therefore, we represent the model with one quadrant of a square plate with vicinity of the hole as shown in Figure 2.6, and apply symmetric boundary condition on it.

In this example, we set the isotropic material of the plate to be: Young modulus $E = 3 \times 10^5$ and Poisson ratio $\nu = 0.3$. There are various ways for constructing this models. The geometry of the model can be represented by using a quadratic element that is C^1 continuous. Figure 2.6(a) shows the control points of the quadratic element. Note that there are two overlapping points at the upper left corner of the model, which cause a singularity in parametrization. The refined mesh is shown in Figure 2.6(b), and the corresponding result is given in Figure 2.6(c). In the result, we can observe that there is a small area with less accurate computed stresses at the upper left corner. Alternatively, the model can be represented by using two quadratic elements with C^0 continuity as show in Figure 2.6(d). The solution will converge to exact solution (see Figure 2.6(f)) as the mesh is refined as shown in Figure 2.6(e).

2.5.2 Open spanner

Next we show a more complex 2D example - the open spanner. In this example, the spanner is assumed to be made from linear elastic material with Young modulus $E =$



(a) (b) (c)

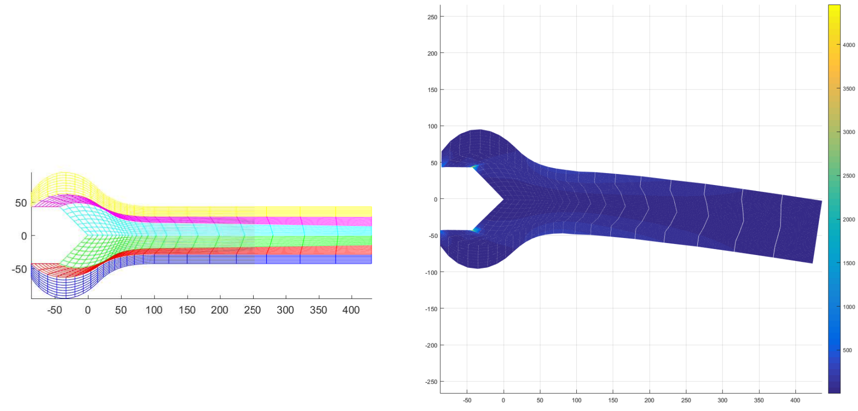
(d) (e) (f)

Figure 2.6: Infinite plate with circular hold example, (a) C^1 continuous NURBS mesh with control points, (b) refined mesh of (a), (c) σ_{xx} displacement of (b), (d) C^0 continuous NURBS mesh with control points, (e) refined mesh of (d), and (f) σ_{xx} displacement of (e).

3×10^5 and Poisson ratio $\nu = 0.3$. Traction is applied at the end of the handle and the jaw is fixed. The spline mesh of the spanner is illustrated in Figure 2.7(a), while the displacement and the Von Mises stress is shown in Figure 2.7(b). We note that some elements in the interior have a high aspect ratio, which may lead to less accurate results, and the continuity between the different regions of the domains is only C^0.

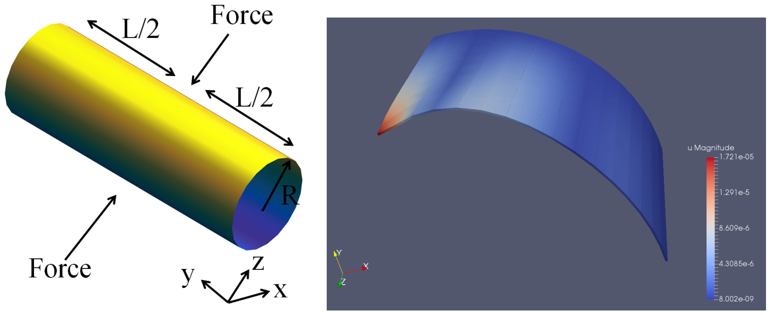
2.5.3 Pinched cylinder

Following is the thick-wall shell example of the pinched cylinder. The cylinder is subjected to two opposite loads at the middle of the cylinder as shown in Figure 2.8(a). Since the stress and boundary condition are symmetric, the result can be illustrated by using one-eighth of the cylinder. In this example, the cylinder is modeled with one cubic element through the thickness. The Young modulus and Poisson ration are set as $E = 3 \times 10^6$ and $\nu = 0.3$ respectively. Figure 2.8(b) shown the displacement of the cylinder.



(a) (b)

Figure 2.7: Spanner example, (a) Splines mesh, (b) displacement and Von Mises stress.

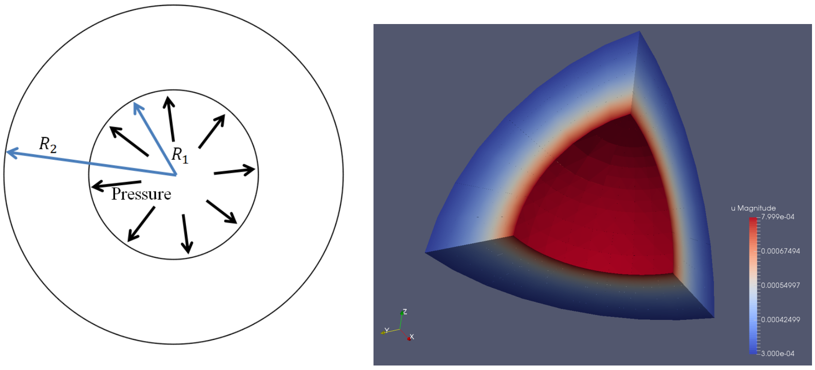


(a) (b)

Figure 2.8: Pinched cylinder example, (a) pinched cylinder, (b) displacement of pinched cylinder.

2.5.4 Hollow sphere

Finally, we show the 3D example of a hollow sphere. The elastic material of the sphere is set as: Young modulus $E = 1 \times 10^3$, and Poisson ration $\nu = 0.3$. In this example, the pressure is applied on the interior surface. Figure 2.9(a) shown the cross-section of the hollow sphere, where the black arrows represent the direction of the pressure, and the blue arrows labeled with R_1 and R_2 denote the interior and outer radius of the sphere. We model one right of the hollow sphere by using one patch. The analysis result is illustrated in Figure 2.9(b).



(a) (b)

Figure 2.9: Hollow sphere example, (a) cross section, (b) analysis result.

2.6 Conclusions

In this chapter, we introduced the splines and computation schemes which are the building block for the development of methods that will be discussed in the upcoming chapters. A more sophisticated application of B-spline method for handling voxelized data and relevant implementation are conducted in Chapter 3.

From the numerical examples, we can observe the difficulty of avoiding singularities for complex shape representation by using a single patch domain. An associated approach for this issue is presented in Chapter 4. Moreover, we show that for problems with the symmetric boundary conditions, single patch models are sufficient to illustrate the stress distribution and displacement. For other situations, multi-patch representations will be needed. Reasonable results are obtained for the multi-patch domains with C^0 continuity for linear elasticity problems. However, for problems involving higher order PDEs, higher continuity is required. More details on the continuity of multi-patch domain will be discussed in Chapter 5.

Chapter 3

B-splines for Image processing

Recall that the goal of this dissertation is to develop the self-contained methods for obtaining smooth volumetric meshes for analysis application. To achieve the goal, it is important that we develop a better understanding of spline for representing shapes in a form that is useful for the construction of analysis model. In this chapter, we determine the B-splines representation of voxel-based images and their application in image registration.

Relevant work on smooth representation of images and image registration will be presented in Section 3.1, followed by an overview for the processing of scan-based images and our approach for convolution strategy in the level set representation in Section 3.2. In Section 3.3 we discuss the registration framework, while the proposed image registration method is presented in Section 3.4. Multiresolution registration will be discussed in Section 3.5. Finally, we show some numerical examples in Section 3.6, and a conclusion is given in Section 3.7. We have also made the Matlab implementation of the proposed method available online at **https://github.com/stellaccl/cdmffd-image-registration**. We refer to the included README file for a description of the installation procedures.

3.1 Previous work

The level set method was devised by Osher and Sethian[17] as a simple and versatile method for computing and analyzing the motion of an interface in two and three dimensions[18]. Verhoosel et al.[19] implemented it for obtaining continuously differentiable internal surface of voxel-based data, where the level set coefficients are computed by using convolution-based strategy. The method involving integration which is computational expensive. In this study, we proposed to use the filtering technique for effectively compute the coefficients as well as the gradients of the level set representation. Also, we examine our convolution approach in a closely related filed - image registration.

Image registration is an image processing technique which aim to find the optimal spatial transformation that maps one image to another. In general, image registration can be classified into rigid and non-rigid registration. Rigid registration assumes images can be aligned with one another through rotation or translation while localized stretching of an image is necessary for non-rigid registration. Non-rigid registration is useful for instances of patient motion and atlas registration applications. There are various methods for non-rigid registration[20,21,22,23,24,25]. In particular, Thirion[24] approached the registration problem as a diffusion model. This was proven to be an approximation of a second order gradient descent based on the sum of squared intensity differences (SSD)[23]. Vemuri[25] presented a deformation method by considering an evolving model for the image. Cachier[20] introduced the iconic feature based (IFB) algorithm, where intensity, similarity, and geometric distance were taken into consideration. In[21,22], the image was deformed by an L^2-gradient flow which is derived through minimizing a predefined energy function.

Free form deformation (FFD) is commonly used to deform the image in non-rigid registration[1,21,22,26,27,28,29]. In FFD, the image is transformed by deforming the B-spline object in which it is embedded, and the B-spline control points are used to describe the motion of the image. A gradient based FFD for non-rigid local deformation was discussed in[28] and a further improvement by using a B-spline fitting approach was proposed in[1]. The advantage of B-spline based FFD methods is that the multiresolution scheme can be applied using B-spline subdivision. Instead of the conventional subdivision, hierarchical B-splines were used in FFD to reduce the number of control points. In the standard FFD method, the image is embedded in a given grid which is defined by B-splines. The gradient is computed directly from the pixel-based image on the location of the control point of this grid. The gradient computed is used to update the B-spline grid and the image is updated by interpolating the intensity value along the deformed grid. In this paper, we propose an alternative FFD method, where we compute the gradient from a smooth B-spline level set representation of the image. In addition, instead of interpolation, we update the image with composition.

A challenge in non-rigid registration is to obtain a smooth deformation field that is invertible. Such deformation is known as diffeomorphic registration. Two common approaches to achieve diffeomorphic registration include the elastic[30,31] and fluid models[32,33,34]. The elastic deformation is based on the Green St. Venant strain tensor. It is however, only valid for small deformations. The fluid model is based on the rate of deformation tensor, which can handle large deformations. Vercauteren[35] presented a diffeomorphic registration method by applying ideas from the Lie Group theory. Another approach[27] is to introduce constraints on the control points such that their displacement is sufficiently small which results in a diffeomorphic B-spline transformation.

Despite the availability of well-developed registration methods, we present an alternative image deformation model compared to FFD in which we represent the image using a B-spline level set. Different from the standard FFD, where optimization is performed on the locations of the control points, the displacements are explicitly computed at each red registration step, in a manner similar to the optical flow Thirions Demons method. The gradients of the intensity value can be evaluated directly in terms of the derivatives of the basis functions. The explicit displacement field is projected on the space of B-spline transformations using composed updates which provides a smoothing effect to the deformation mapping. We also employ filtering techniques in the registration process. Since filtering is commonly used in image processing, there are efficient algorithms available which help minimize the computational cost.

3.2 Voxelized images

Features and the intensity of the image are the standard information provided by the voxel-based images. The feature-based details require the detection of salient and distinctive objects (closed boundary regions, line intersections, edges, corners, etc.). A shortcoming of using feature-based information is that they require the detection which may need to be done manually, and the certainty of geometry approximation depends on the accuracy of feature detection. On the other hand, the intensity-based method employs the pixel intensity levels for determining the geometric information. The complexity of detecting the feature can thus be avoided. Therefore, it is more flexible than the feature-based because most of the important information available from the raw data is taken into consideration. Below is a simple introduction to the intensity model.

Suppose a voxel-based image is contained in a domain $\Omega = \otimes_{d=1}^{n_d}[0, L_d] \in \mathbb{R}^{n_d}$ which is partitioned by a set of pixels or voxels Ω_{voxel}^e, $e \in \{1, 2, \ldots, n_{voxel}\}$. n_d represents the physical dimension, in the following discussion we assume $n_d = 3$. The intensity model can be defines by the gray scale function $\mathscr{G} : \Omega \to \mathbb{R}$ as follows:

$$\mathscr{G}(X) = \begin{cases} \mathscr{I}_1 & : X \in \Omega_{voxel}^1 \\ \mathscr{I}_2 & : X \in \Omega_{voxel}^2 \\ \vdots & \\ \mathscr{I}_{m_{voxel}} & : X \in \Omega_{voxel}^{n_{voxel}} \end{cases}$$

with $X = (x, y, z)$ being the coordinates of the voxel centers and $\mathscr{I}_\imath$ $(\imath = 1, 2, \ldots, n_{voxel})$ is the gray scale value in the voxels $\{\Omega_{voxel}^e\}_{e=1}^{n_{voxel}}$. This discrete representation is non-smooth and would significantly affect the calculation of gradient. A smooth representation of the image is, therefore, necessary for more accurate gradient evaluations.

3.2.1 B-spline level set representation

We propose to compute the smooth approximation of the image based on a discrete cubic B-spline kernel K_B. The elements in K_B are computed using the cardinal basis function $\mathcal{N}^3(x)$ defined as follows:

$$\mathcal{N}^3(x) = \begin{cases} b_0(x) = \frac{(2+x)^3}{6} & : -2 \leqslant x < -1 \\ b_1(x) = \frac{2}{3} - x^2 - \frac{x^3}{2} & : -1 \leqslant x < 0 \\ b_2(x) = \frac{2}{3} - x^2 + \frac{x^3}{2} & : 0 \leqslant x < 1 \\ b_3(x) = \frac{(2-x)^3}{6} & : 1 \leqslant x < 2 \\ 0 & : \text{otherwise} \end{cases} \tag{3.1}$$

Given the gray values $\mathcal{G}(x,y,z)$, we would like to find the coefficient $c_{i,j,k}$ such that the cubic cardinal B-spline function $\mathcal{B}(x,y,z)$ interpolates the intensity of the image:

$$\mathcal{B}(x,y,z) = \sum_{i=l}^{l+3} \sum_{j=m}^{m+3} \sum_{k=n}^{n+3} c_{i,j,k} \mathcal{N}^3(x-i) \mathcal{N}^3(y-j) \mathcal{N}^3(z-k) \tag{3.2}$$

with

$$l = \lceil x-2 \rceil, \quad m = \lceil y-2 \rceil, \quad n = \lceil z-2 \rceil.$$

Equation (3.2) is also known as the B-spline level set function. Solving for the coefficients $c_{i,j,k}$ can be performed using a matrix framework, e.g., by a L^2 projection or least squares fitting. A more efficient approach is to use digital filtering techniques. Using the fact that the B-splines are translation invariant, $\mathcal{G}$ can be evaluated at the voxel center (x,y,z) by using convolution as follows:

$$\mathcal{G}(x,y,z) = (K_B * c_{i,j,k}) \tag{3.3}$$

where K_B is the discrete B-spline kernel. The convolution operation, in this case, can be viewed as sliding a kernel K_B over the coefficients array c, this process is known as “filtering with convolution”. In the rest of the study, we refer to it as filtering. $c_{i,j,k}$ can be solved by an inverse filtering:

$$c_{i,j,k} = (K_B)^{-1} * \mathcal{G}(x,y,z). \tag{3.4}$$

Unser[36] presented an efficient computation method for $c_{i,j,k}$ by using casual and anti-casual recursive filter. The resulting B-spline model is an exact interpolation of the data. In our study, however, we prefer a smoother approximation to avoid oscillation caused by noise or Gibbs phenomenon. We implement the approach in[19] and compute the coefficients as follows:

$$\begin{aligned} c_{i,j,k} &= \frac{\int_\Omega \hat{\mathcal{N}}^3_{i,j,k}(x,y,z)g(x,y,z)\,dx\,dy\,dz}{\int_\Omega \hat{\mathcal{N}}^3_{i,j,k}(x,y,z)\,dx\,dy\,dz} \\ &= \frac{1}{A}\int_\Omega \hat{\mathcal{N}}^3_{i,j,k}(x,y,z)g(x,y,z)\,dx\,dy\,dz \end{aligned} \tag{3.5}$$

where $\hat{\mathcal{N}}^3_{i,j,k}(x,y,z) = \mathcal{N}^3(x-i)\mathcal{N}^3(y-j)\mathcal{N}^3(z-k)$, and $A = \int_\Omega \hat{\mathcal{N}}^3_{i,j,k}(x,y,z)\,dx\,dy\,dz$. Since $\mathscr{G}(x,y,z)$ is a piecewise constant function and $\hat{\mathcal{N}}^3_{i,j,k}(x,y,z)$ is a piecewise polynomial, we obtain

$$\begin{aligned} &\int_\Omega \hat{\mathcal{N}}^3_{i,j,k}(x,y,z)\mathscr{G}(x,y,z)dx\,dy\,dz \\ &= \int_\Omega \sum_{i=l}^{l+3}\sum_{j=m}^{m+3}\sum_{k=n}^{n+3} \mathcal{N}^3(x-i)\mathcal{N}^3(y-j)\mathcal{N}^3(z-k)\mathscr{G}(x,y,z)\,dx\,dy\,dz \\ &= K_B * M, \end{aligned} \tag{3.6}$$

where K_B is a kernel given by:

$$K_B(\hat{\imath},\hat{\jmath},\hat{k}) = a_{\hat{\imath}}a_{\hat{\jmath}}a_{\hat{k}} \quad , \quad \hat{\imath},\hat{\jmath},\hat{k} = 0,1,2,3. \tag{3.7}$$

and $a_{\hat{\imath}} = \int_{p_{\hat{\imath}}}^{p_{\hat{\imath}}+1} b_{\hat{\imath}}(x)\,dx$, $p_{\hat{\imath}} = \hat{\imath}-2$. Moreover, matrix M is composed of gray scale values $\mathscr{I}$ of the corresponding integration domain. Thus, we can compute the coefficient efficiently by:

$$c_{i,j,k} = \frac{1}{A}(K_B * \mathscr{G}(x,y,z)). \tag{3.8}$$

Note that h_a is a $4\times4\times4$ matrix whose entries only depend on the cardinal B-spline β^3 and can be precomputed. In addition, h_a is a separable kernel which has a computational advantage, since applying it to a 3D image can be done by 1D filtering in each direction. Quantitatively, if $h_a \in \mathbb{R}^{p\times q\times r}$is non-separable, then filtering requires $\mathscr{O}(p\cdot q\cdot r)$ multiplication and additions for each voxel. For a separable kernel, the computation cost decreases to $\mathscr{O}(p+q+r)$. Efficient algorithms exist for performing these operations, including those that exploit parallelism on multi-core systems or GPUs.

With the coefficients $c_{i,j,k}$ obtained using the approach in[19], the following properties hold:

1. Conservation of average gray scale intensity

$$\frac{1}{A}\int_{\Omega}\mathcal{B}(X)dX = \frac{1}{A}\int_{\Omega}\mathcal{G}(X)dX = \frac{1}{n_{voxel}}\sum_{e=1}^{n_{voxel}} c_e. \tag{3.9}$$

2. Local Boundedness

$$\min_{X\in\Omega^e_{supp}}(\mathcal{G}(X)) \leq \mathcal{B}(X) \leq \max_{X\in\Omega^e_{supp}}(\mathcal{G}(X)) \quad \forall X \in \Omega^e_{voxel}. \tag{3.10}$$

   where $\Omega^e_{supp} = \bigcup_{e\in\mathcal{I}_e}\Omega^e_{voxel}$, and $\mathcal{I}_e = \{e : \hat{\mathcal{N}}^3(X)\big|_{\Omega^e_{voxel}} \neq 0\}$

3. Approximation to a Gaussian Kernel
   Substituting (3.5) into (3.2) we obtain

$$\mathcal{B}(X) = \int_{\Omega}\left[\sum_{i,j,k}\frac{\hat{\mathcal{N}}^3_{i,j,k}(X)\hat{\mathcal{N}}^3_{i,j,k}(Y)}{A}\right]\mathcal{G}(Y)dY. \tag{3.11}$$

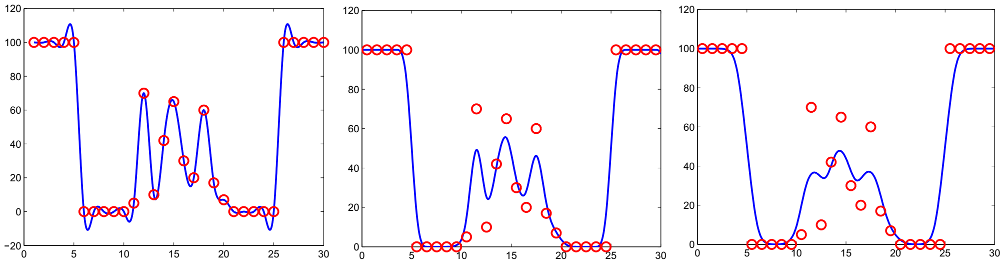
   Thus $\mathcal{B}(X)$ is the integral transform of intensity value $\mathcal{G}(X)$ with the kernel

$$k(X,Y) = \sum_{i,j,k}\frac{\hat{\mathcal{N}}^3_{i,j,k}(X)\hat{\mathcal{N}}^3_{i,j,k}(Y)}{A}. \tag{3.12}$$

   This kernel is an approximation to the Gaussian filter.

An alternative way to compute the interpolation coefficients, commonly used in FFD, is to set the coefficient $c_{i,j,k}$ to the values of the input data[1]. This method is very easy to implement, but it does not take into account the overlap between the basis functions. The resulting spline is less smooth for noisy input than the convolution approach.

We illustrate with a 1D example the difference between computing coefficient using the exact interpolation method in[36], the method used in[1] and the method using (3.8). In Figure 3.1, the red circles represent the data values. Figure 3.1(a) and Figure 3.1(b) show interpolation using the exact interpolation method in[36] and interpolation used in FFD respectively. For both methods, oscillations occur at the places where there is a sharp change of data value. Figure 3.1(c) shows interpolation using the convolution strategy discussed in[19], whereby a smoother approximation is obtained. Such smooth interpolation is desirable in the context of image registration, because it provides a filtering effect that smoothes irregular intensity caused by noise. Thus, it gives a representation of the image that is suitable for computing the gradient of the intensity. Another advantage of representing the image using a B-spline model is that it is second-order continuously differentiable. We will discuss in the following section fast



(a) (b) (c)

Figure 3.1: (a) Interpolation using the exact interpolation method, (b) interpolation in ^{1}, and (c) interpolation using (3.8).

computation of gradients from the coefficients $c_{i,j,k}$ by filtering.

Before implementing the B-splines level set discussed above for obtaining smooth analysis model, we would like the examine it in a closely related field - image registration. The registration framework will be given in the following section.

3.3 Registration framework

Given a fixed image I_1 and a moving image I_0, image registration is an optimization problem that aims to find the spatial transformation $\mathbf{T}$ such that

$$I_0 \circ \mathbf{T} \approx I_1. \tag{3.13}$$

A similarity criterion is used to measure the differences between the two input images. There are several similarity measures - e.g. direct correlation, mean square error and sum of square differences (SSD). In this paper, we will adopt the SSD which is more computationally straightforward. The SSD is defined as

$$SSD(I_1, I_0) = \sum_{\wp \in \Omega} |I_1(\wp) - I_0(\wp)|^2, \tag{3.14}$$

where Ω is the region domain of the image, and the sum is taken over all pixels (or voxels) $\wp$. A simple optimization of (3.14) will lead to unstable and non-smooth solutions. This problem can be avoided by adding the regularization term $\mathrm{Reg}(\mathbf{T})$ to the global energy function as:

$$E(\mathbf{T}) = SSD(I_1, I_0) + \sigma_T \mathrm{Reg}(\mathbf{T}) \tag{3.15}$$

where σ_T is a parameter to control the amount of regularization needed. The combination of SSD and regularization provides a well-posed optimization framework. However, it is quite challenging to optimize (3.15) when $\mathbf{T}$ depends on a large number of parameters. Cachier[20] proposed a solution to this problem by adding a hidden variable and by defining the global energy function in terms of two transformations (**C** and **T**) as follows:

$$E(\mathbf{C},\mathbf{T}) = SSD(I_1, I_0 \circ \mathbf{C}) + \sigma_x \mathrm{dist}(\mathbf{T},\mathbf{C})^2 + \sigma_T \mathrm{Reg}(\mathbf{T}). \tag{3.16}$$

Here $\mathrm{dist}(\mathbf{C},\mathbf{T}) = ||\mathbf{T}-\mathbf{C}||$ and σ_x accounts for the spatial uncertainty between the transformation **C** and **T**. The optimization is initialized by setting $\mathbf{T}_0$ equal to the identity transformation (Id) at iteration 0 and performing the following steps at each iteration $n \geq 1$:

- For a given $\mathbf{T}_{n-1}$, find $\mathbf{C}_n$ that minimizes

  $$E_1(\mathbf{C}_n;\mathbf{T}_{n-1}) = SSD(I_1, I_0 \circ \mathbf{C}_n) + \sigma_x \mathrm{dist}(\mathbf{T}_{n-1},\mathbf{C}_n)^2.$$

- For the $\mathbf{C}_n$ obtained from the previous step, find $\mathbf{T}_n$ that minimizes

  $$E_2(\mathbf{T}_n;\mathbf{C}_n) = \sigma_x \mathrm{dist}(\mathbf{T}_n,\mathbf{C}_n)^2 + \sigma_T \mathrm{Reg}(\mathbf{T}_n).$$

  If $\mathrm{Reg}(\mathbf{T}_n) = \parallel \nabla \mathbf{T}_n \parallel^2$ this minimization can be efficiently computed using a convolution of Gaussian kernel K_G with $\mathbf{C}_n$, i.e. $\mathbf{T}_n \leftarrow K_G * \mathbf{C}_n$. The symbol $*$ denotes the convolution operator.

Thirion[24] presented an efficient method to compute a force that is equivalent to the computation of an update to $\mathbf{T}_n$, as discussed in the first step above. The main idea is to deform the image by using a diffusion model and defining a force acting on each pixel (so-called the "demon" force analogy to Maxwell's demon). The force is modified to include a normalization that is necessary to avoid instabilities due to small gradient values. Thirion used updates of the form:

$$\mathbf{V}_{n+1} = \frac{(I_1 - I_0 \circ \mathbf{T}_n)\nabla I_1}{||\nabla I_1||^2 + (I_1 - I_0 \circ \mathbf{T})^2}, \quad \mathbf{T}_0 = \mathrm{Id}, \quad \mathbf{T}_n = \mathrm{Id} + \mathbf{V}_n. \tag{3.17}$$

Here $\mathbf{V}_n$ is a vector field representing the displacement of each pixel at step n. Vercauteren[35] derived a more general displacement field as:

$$\mathbf{V}_{n+1} = \frac{(I_1 - I_0 \circ \mathbf{T}_n)\mathbf{J}^{\wp}}{\parallel \mathbf{J}^{\wp} \parallel^2 + \sigma_x^{-2}(I_1 - I_0 \circ \mathbf{T}_n)}, \tag{3.18}$$

where $\mathbf{J}^{\wp}$ can also be defined as either the gradient of the moving image or the average of the gradient of the fixed and moving image $\frac{\nabla I_1 + \nabla(I_0 \circ \mathbf{T})}{2}$. σ_x is a parameter used to control the step size of the displacement field. It can be shown[35] that the maximum displacement is bounded by:

$$\parallel \mathbf{V}_{n+1} \parallel \leqslant \frac{\sigma_x}{2}.$$

Vemuri[25] proposed to deform the image using the level set method. The idea is to let the image evolve in the direction normal to its contour lines. The vector field is updated using the following governing equation:

$$\mathbf{V}_{n+1} = (I_1 - I_0 \circ (\mathrm{Id} + \mathbf{V}_n)) \frac{\nabla(K_G * (I_0 \circ (\mathrm{Id} + \mathbf{V}_n)))}{||\nabla(K_G * (I_0 \circ (\mathrm{Id} + \mathbf{V}_n)))|| + \rho}, \tag{3.19}$$

where K_G represents the Gaussian Kernel and ρ is a small positive constant that acts as a stabilizer. Though Thirion[24], Vercauteren[35] and Vemuri[25] used different approaches, the vector field updates are computed similarly in all these methods. In this study, we will adopt an approach similar to that in[35] and use the parameter σ_x to control the step size.

There are two ways to update the transformation **T** from a given displacements field **V**[35]:

1. Additive update

$$\mathbf{C}_n = \mathbf{T}_{n-1} + \mathbf{V}_n = \mathbf{V}_1 + \mathbf{V}_2 + \cdots + \mathbf{V}_n.$$

2. Composite update

$$\mathbf{C}_n = \mathbf{T}_{n-1} \circ (\mathrm{Id} + \mathbf{V}_n) = \mathbf{V}_1 \circ \mathbf{V}_2 \circ \cdots \circ \mathbf{V}_n.$$

Additive updates are easy to implement. However, they are less efficient for large deformations and have no geometric meaning. For example if $\mathbf{V}_n$ represents a rotation by angle α, then a rotation by angle 2α is naturally represented by $\mathbf{V}_n \circ \mathbf{V}_n = \mathbf{V}_n^2$, which is not the same as $\mathbf{V}_n + \mathbf{V}_n$, in particular for large α. According to Ashburner[37], the additive method is only suitable for small deformations. The inverse transformation of the additive method obtained through subtraction is just an approximation. Therefore, it is hard to enforce a one-to-one mapping that is invertible and can correctly represent large deformations. Composite updates provide a natural way of working with diffeomorphic transformations (since the composition of two diffeomorphisms is also diffeomorphism). However, composite updates are more difficult to implement. They normally require the interpolation of the image intensity and the gradients "in-between" the pixels, which can introduce errors in the registration algorithm. In this study, we perform the update of transformation by using B-spline composition. More details of this procedure will be discussed in the next section.

3.4 Non-rigid registration

In this section, we will present an image registration approach which is a hybrid of FFD and optical flow method. Meanwhile, we will discuss the update of transformation by composing the B-spline transformation from the previous step with a linear update $(\mathrm{Id} + \mathbf{V})$, resulting in another B-spline transformation of the same polynomial degree.

3.4.1 The governing equation

In this study, we deform the image using the following displacement $\mathbf{V}(X)$ at each point X:

$$\mathbf{V}(X) = \frac{(I_1(X) - \mathcal{B}(X))\nabla\mathcal{B}(X)}{\| \nabla\mathcal{B}(X) \|^2 + \gamma \cdot (I_1(X) - \mathcal{B}(X))^2}. \tag{3.20}$$

An important difference in this equation compared to (3.17), (3.18) and (3.19) is that we use the B-spline representation to evaluate the gradient values and that $\mathcal{B}(X)$ is a B-spline representation of $I_0 \circ \mathbf{T}$. This can be done by finding the partial derivative of the basis using a filtering method as discussed in the previous section. This vector update contains the registration parameter γ corresponding to $\frac{1}{\sigma_x^2(x)}$ in (3.18) which allows more control over the step size. To compute the gradient

$$\nabla\mathcal{B}(X) = \begin{bmatrix} \frac{\partial\mathcal{B}(X)}{\partial x} \\ \frac{\partial\mathcal{B}(X)}{\partial y} \\ \frac{\partial\mathcal{B}(X)}{\partial z} \end{bmatrix},$$

the partial derivative of $\mathcal{B}(X)$ with respect to x is given using (3.2) by:

$$\frac{\partial\mathcal{B}(X)}{\partial x} = \frac{\partial\mathcal{B}(x,y,z)}{\partial x} = \sum_{i=l}^{l+3}\sum_{j=m}^{m+3}\sum_{k=n}^{n+3} c_{i,j,k} \frac{\partial\mathcal{N}^3(x-i)}{\partial x} \mathcal{N}^3(y-j)\mathcal{N}^3(z-k). \tag{3.21}$$

Using the fact that we need to evaluate the gradients only at the voxel centers, it is suffices to precompute the values of the cardinal B-spline $\hat{\mathcal{N}}^3$ and its gradients at the knot midpoints -1.5, -0.5, 0.5 and 1.5. In particular, we define the discrete kernel K_{B_x} by:

$$K_{B_x}(\hat{i},\hat{j},\hat{k}) = \frac{\partial\mathcal{N}^3(x_{\hat{i}})}{\partial x} \times \mathcal{N}^3(x_{\hat{j}}) \times \mathcal{N}^3(x_{\hat{k}}) \quad , \quad \hat{i},\hat{j},\hat{k} = 0,1,2,3, \tag{3.22}$$

where $x_0 = -1.5, x_1 = -0.5, x_2 = 0.5, x_3 = 1.5$. Now (3.21) can be written in the form of the convolution

$$\frac{\partial\mathcal{B}(X)}{\partial x} = (K_{B_x} * c)(X). \tag{3.23}$$

The corresponding kernel K_{B_y} and K_{B_z} of variable y and z can be computed using a similar procedure. By using these kernels, $\nabla\mathcal{B}(X)$ can be computed efficiently by the filtering technique:

$$\nabla\mathcal{B}(X) = \begin{bmatrix} (K_{B_x} * c)(X) \\ (K_{B_y} * c)(X) \\ (K_{B_z} * c)(X) \end{bmatrix}. \tag{3.24}$$

We also compute the composition of transformation using B-splines. Let $\mathbf{T}_{n-1}$ be the previous transformation and $\mathbf{V}_n$ be the updated vector field, we compose $\mathbf{V}_n$ with $\mathbf{T}_{n-1}$ as:

$$\begin{aligned}\mathbf{T}_n(X) &= \mathbf{T}_{n-1} \circ (\mathrm{Id}+\mathbf{V}_n)(X) \\ &= \sum_{i=l}^{l+3}\sum_{j=m}^{m+3}\sum_{k=n}^{n+3} \mathbf{C}_{i,j,k}^{(n-1)} \hat{\mathcal{N}}_{i,j,k}^{3}((\mathrm{Id}+\mathbf{V}_n)(X)) \\ &= \sum_{i=l}^{l+3}\sum_{j=m}^{m+3}\sum_{k=n}^{n+3} \mathbf{C}_{i,j,k}^{(n)} \hat{\mathcal{N}}_{i,j,k}^{3}(X)\end{aligned} \tag{3.25}$$

where $\mathrm{Id}(X) = X$ is the identity transformation. The coefficients $\mathbf{C}_{i,j,k}^{(n)} = [c_{i,j,k}^{x,(n)}, c_{i,j,k}^{y,(n)}, c_{i,j,k}^{z,(n)}]$ are obtained by convolution of the x, y and z components of the previous transformation $\mathbf{T}_{n-1}^{x}(X)$, $\mathbf{T}_{n-1}^{y}(X)$ and $\mathbf{T}_{n-1}^{z}(X)$ composed with the displacement $(\mathrm{Id}+\mathbf{V}_n)(X)$. However, note that here the basis $\hat{\mathcal{N}}^3$ is evaluated at points that are not equally spaced. This requires the evaluation of cubic polynomials at arbitrary points. However, the computation of $\mathbf{C}_{i,j,k}^{(n)}$ can be performed efficiently using filtering with the B-spline kernel K_B as follows:

$$\begin{aligned} c_{i,j,k}^{x,(n)} &= \frac{1}{A}(K_B * \mathbf{T}_n^x(X)), \\ c_{i,j,k}^{y,(n)} &= \frac{1}{A}(K_B * \mathbf{T}_n^y(X)), \\ c_{i,j,k}^{z,(n)} &= \frac{1}{A}(K_B * \mathbf{T}_n^z(X)). \end{aligned} \tag{3.26}$$

We note that the repeated convolution using (3.26) only affects the transformation mapping and can be used to provide a smoothing effect. The source image is converted to a B-spline level-set only once via convolution, as described in more detail in Section 3.4.4, so there is no degradation of the image quality during the registration.

3.4.2 Diffeomorphisms

It is important for many applications that the deformation field is smooth and injective. Finding such deformations is known as diffeomorphic registration. Referring to Rueckert[27], it can be achieved by constraining the displacement to be a fraction of the distance between the voxel centers. Let $\Delta c_{i,j,k} = c_{i,j,k}^{(n)} - c_{i,j,k}^{(n-1)}$ be the displacement of control point $c_{i,j,k}^{(n-1)}$. Moreover the deformation is locally injective if $\max|\Delta c_{i,j,k}^{x}| < \frac{1}{K}$, $\max|\Delta c_{i,j,k}^{y}| < \frac{1}{K}$ and $\max|\Delta c_{i,j,k}^{z}| < \frac{1}{K}$, where $K \approx 2.48$ when using uniform cubic

B-spline functions[38]. Using the fact[19] that the displacement of the control points is bounded by the displacement of the voxels, and equations (3.18) and (3.20) we write:

$$\max\|\Delta\mathbf{C}_{i,j,k}\| \leqslant \|\mathbf{V}(X)\| \leqslant \frac{\sigma_x}{2} = \frac{1}{2\sqrt{\gamma}}.$$

Thus, in order to obtain diffeomorphic registration, we can set γ such that

$$\frac{1}{2\sqrt{\gamma}} \leqslant \frac{1}{K}.$$

Since $K \approx 2.48$, we have $\gamma \geqslant 1.5376$. Therefore diffeomorphic transformation can be obtained by choosing such γ in (3.20). Since the composition of diffeomorphic transformations is diffeomorphic, this property is preserved by the overall registration.

3.4.3 Regularization

The transformations can be regularized by relying on physical models (i.e., elastic model, fluid model, etc.). Note that the B-spline convolution strategy that we used for computing the coefficient for B-spline model of the image (discussed in section 3) and also vector transformation (discussed in section 4.1) already have a regularization effect. However, if more regularization is necessary, we can apply a Gaussian filter to the vector field. The convolution of a vector field V with Gaussian kernel K_G to give a smoothed displacement is performed as:

$$\mathbf{V}(x,y,z) * K_G = \sum_{i=-n}^{n}\sum_{j=-n}^{n}\sum_{k=-n}^{n} \mathbf{V}(x,y,z) K_G(x-i, y-j, z-k), \quad n = 6 \times \sigma_G, \tag{3.27}$$

where x,y and z are the point position of vector V, and σ_G is the standard deviation of Gaussian filter. Applying the Gaussian smoothing to the displacement field at each step could be considered as a rough simulation of a fluid model[23].

3.4.4 Moving image update

Interpolation is applied to update the image with the corresponding transformation. A natural way to perform this update, since both the image and the transformation are represented in terms of B-splines, is to use composition. In particular,

$$\mathcal{B}(X) = I_0 \circ \mathbf{T}_n(X) = \sum_{i,j,k} c_{i,j,k} \mathcal{N}^3(\mathbf{T}_n(X)). \tag{3.28}$$

The coefficients $c_{i,j,k}$ are the coefficients of the B-spline level set function of the source image. This composition can be easily converted to a level set function using (3.2) for use in the next step. To summarize, we use the Algorithm 1 to find the optimal spatial mapping $\mathbf{T}$. This algorithm is implemented in refImage2D.m and regImage3D.m.

```
Algorithm 1: Algorithm for image registration
Input  : source image $I_0$, target image $I_1$
Output: $\begin{bmatrix} c^x_{i,j,k} \\ c^y_{i,j,k} \\ c^z_{i,j,k} \end{bmatrix}$ such that $I_0 \circ \sum_{i,j,k} \begin{bmatrix} c^x_{i,j,k} \\ c^y_{i,j,k} \\ c^z_{i,j,k} \end{bmatrix} \hat{\mathcal{N}}_{i,j,k}(X) \approx I_1(X)$
1: Compute $c_{i,j,k}$ using (3.8) such that $\mathcal{B}(x,y,z) \approx I_0(x,y,k)$.
2: Let $\mathbf{T}_0 = \mathrm{Id}(X)$ , compute $\begin{bmatrix} c^x_{i,j,k} \\ c^y_{i,j,k} \\ c^z_{i,j,k} \end{bmatrix}$ by using (3.26).
3: For each iteration:
    (a) Compute $\nabla\mathcal{B}(X)$ using (3.24).
    (b) Compute $\mathbf{V}_n(X)$ using (3.20).
    (c) Regularize if necessary $\mathbf{V}_n = \mathbf{V}_n * K_G$.
    (d) Evaluate $\mathbf{T}_n = \mathbf{T}_{n-1} \circ (\mathrm{Id} + \mathbf{V}_n)$ using (3.25).
    (e) Compute $\begin{bmatrix} c^x_{i,j,k} \\ c^y_{i,j,k} \\ c^z_{i,j,k} \end{bmatrix}$ using (3.26).
    (f) Compute $c_{i,j,k}$ such that $\mathcal{B}(X) = I_0 \circ \mathbf{T}_n$ using (3.8).
```

3.4.5 Parallel computing and Matlab implementation

In image registration, we need to find a solution (the spatial transformation) from a given data set (the images). Solving this problem for large data sets requires a lot of memory. Also, it involves computationally intensive tasks. Thus, parallel computing which take advantage of multi-core processors is suitable for image registration. MPI (Message Passing Interface) and OpenMP (Open Multi-Processing) are the two often used communications protocols for parallel computing. MPI is used in distributed memory systems, where computational tasks are distributed to different nodes. The running code on each node does not have access to the memory of the other nodes, therefore, they communicate via a messaging interface. A very simple MPI implementation is provided in Matlab by the “parfor” command which can be used to parallelise a for loop, where each iteration is independent of the others. When this command is issued, Matlab starts worker-threads that perform the computations in each loop and return them to the main thread. We used “parfor” in the following subroutines:

- computing the coefficients $c_{i,j,k}$ in (3.8), which is implemented in img2coef2D.m and img2coef3D.m. In these functions, we filtered the data using discrete kernel in each spatial dimension by using the function bspline1dfilt.m.
- composing the transformations in (3.25), which is implemented in BsplineCompose2D.m and BsplineCompose3D.m. The outermost sum is parallelised by using the "parfor" loop.
- composing the B-Spline level set representation of the moving image I_0 with the current transformation in (3.28). This is implemented in BsplineComposeImage2D.m and BsplineComposeImage3D.m. The outermost sum is parallelised by using the "parfor" loop.

There are also some subroutines in our code which are parallelised automatically by Matlab. These included the use of imfilter in computing the gradient and also the Gaussian filter (if used). We note there is also a GPU-optimized implementation of imfilter provided by GPUArray / imfilter. Besides MPI, the computation can be further enhanced by using the OpenMP protocol. OpenMP targets shared memory systems, and uses a thread based parallelism. It is suitable for the case where the computational task for each thread is small compared to the size of the variables involved. In Matlab, the "parfor" loops will be converted to OpenMP when it is compiled to a mex file via the built-in "coder" toolbox.

3.5 Multiresolution registration

The proposed method can be improved when used in combination with a coarse to fine multiresolution scheme. First, the computational load is reduced significantly when the computations are performed on a coarse version of the image. Second, large deformations can be handled fast and efficiently since the transformation field is coarse. Recall that the goal of image registration is, for given I_0 and I_1, to find the transformations $\mathbf{C}$ and $\mathbf{T}$ such that

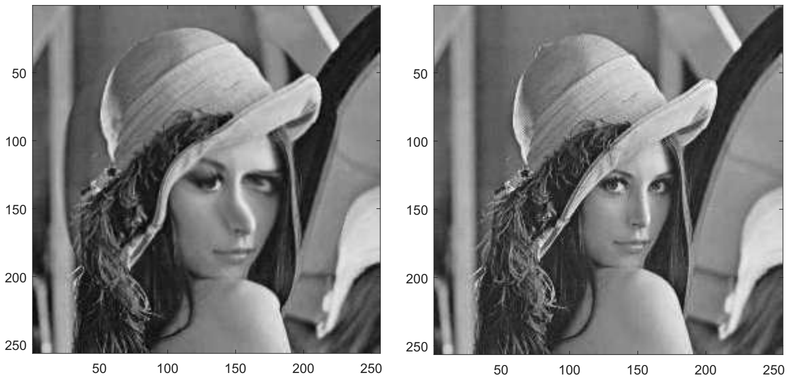
$$\min E(\mathbf{C},\mathbf{T})$$

where $E(\mathbf{C},\mathbf{T})$ is defined in (3.16).

In the multiresolution framework, we solve a sequence of registration problems successively from the coarser level ($\ell = 1$) to the finest level ($\ell = \textit{number of levels}$). The scaling factor for the image at level (ℓ) is given by $\frac{1}{2^{\textit{number of levels}-\ell}}$. By using this scaling factor, we reduce the image size in each direction by half in each level. In the first level, we can obtain I_0^1 and I_1^1 directly by scaling I_0 and I_1 with the scale factor. For $\ell > 1$, we scale the I_0 and I_1 according the scaling factor of the corresponding level, followed by composing the transformation field ($\mathbf{T}^{\ell-1}$) from level ($\ell-1$). Then the transformation field ($\mathbf{T}^{\ell}$) is obtained by performing Algorithm 1 using I_0^{ℓ} and I_1^{ℓ}. We also update the deformation grids in a similar way so that the information is inherited and the complete transformation can be seen clearly in the final deformation grids.

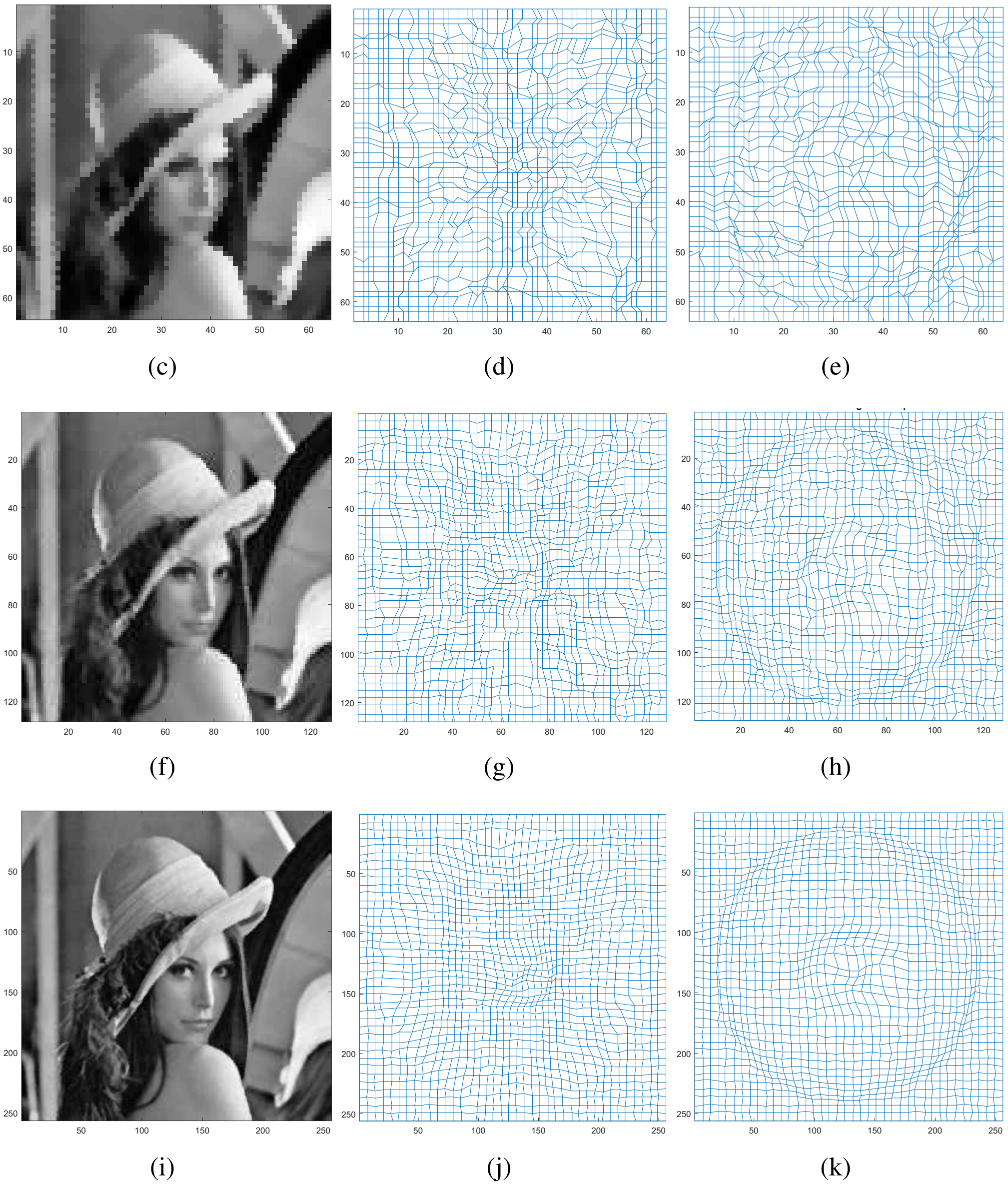
The multiresolution algorithm is implemented in the functions Multiresolution2D.m and Multiresolution3D. Note that it is not always the case that more multiresolution levels result in faster and better registration. The number of levels should depend on the image size and also on the type of the image. If there are a lot of small details in the image, a very coarse level would not be able to capture the deformation very well.

In the following example, we illustrate the multi-resolution registration discussed above using the Lena image. We perform a three level multiresolution registration on it. In the first row, Figures 3.2(a) and 3.2(b) are the given fixed and moving image with size 256×256 pixels. The results from the coarse to fine levels are shown in the second to the fourth row. The image sizes of the first and second levels are 64×64 pixels, and 128×128 pixels respectively, and the images in the final level have the original size. Note that it is difficult to illustrate the transformation clearly with dense deformation grids, thus, the deformation grids for all levels are plotted using 40×40 uniformly spaced grid lines. We can observe that the large deformation can be registered well in the coarse level, and the fine detail is progressively recovered at the finer levels. Also, the resulting deformation grids does not show only the deformation of its level, but the up-scaled transformation from the previous levels. The final deformation grids (Figures 3.3(k) and 3.3(j)) show the deformation information of the entire registration process. Please refer to main2DLena.m for the Matlab implementation of this example.



(a) (b)

Figure 3.2: Multi-resolution example, first row: (a) the moving image and (b) the fixed image.



(c) (d) (e)

(f) (g) (h)

(i) (j) (k)

Figure 3.2: Multi-resolution example (cont.), second row: result from the first level; (c) deformed image, (d) forward deformation grids, and (e) inverse deformation grids. Third row: result from the second level; (f) deformed image, (g) forward deformation grids, and (h) inverse deformation grids. Fourth row: result from the third level; (i) deformed image, (j) forward deformation grids, and (k) inverse deformation grids.

3.6 Numerical examples

We examine our image registration method through several examples. We use the similarity ratio, Rs defined as:

$$Rs = 1 - \frac{||I_1 - I_0 \circ \mathbf{T}||}{||I_1 - I_0||}$$

to evaluate the result. Perfect registration will result in $Rs = 1$, and a low Rs reflects a poor match between the source and target images. Note that the core aspect of registration is to produce the transformation between the moving and fixed images that is physically plausible. The transformation maps from our algorithm are illustrated in the 2D examples.

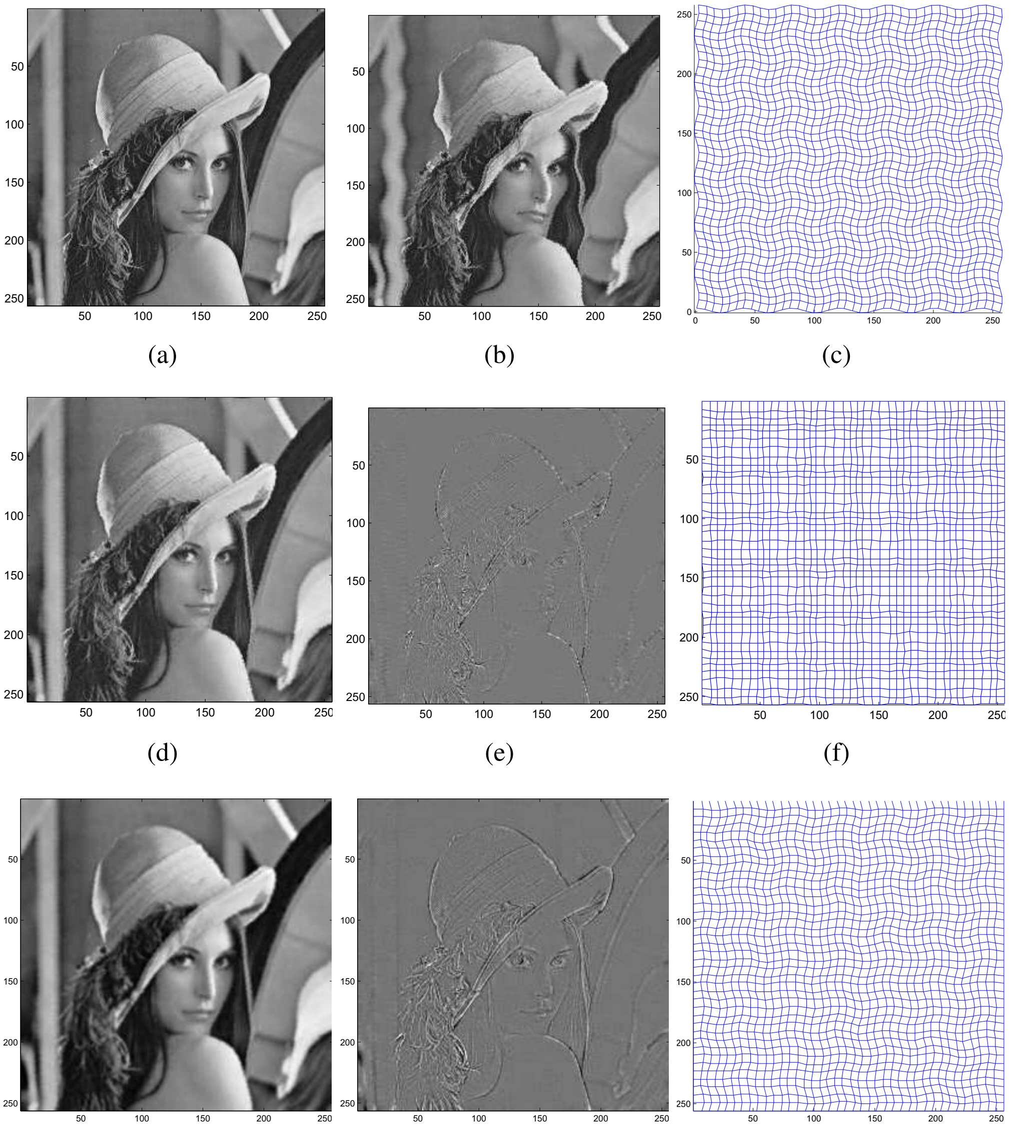
3.6.1 Examine on synthetic transformation

We first evaluate the registration algorithm by using an image deformed with a known synthetic transformation. The photographic image is distorted using a sinusoidal function as shown in Figure 3.3(c). This synthetic transformation is used such that the same deformation pattern is repeated in the image, and we can examine the ability of our proposed method for recovering the sinusoidal transformation. We compare the deformation grids obtained by the exact transformation, the proposed method and a diffeomorphic optical flow implementation[39] using Matlab[40], respectively. Please refer to main2DLenaSynthetic.m for the Matlab code of this example.

In Figure 3.3, both registered images from our method and the optical flow method look very similar to the fixed image. However, the deformation grids from the optical flow method are unable to recover the deformation grids well. On the other hand, there is a very good match between the computed transformation from the proposed method with the exact transformation. This is true even in the parts of the image that have uniform intensity (such as the background).

3.6.2 C-shaped image

Next we consider the benchmark example of C-shape synthetic images, where we seek to deform a circle into a C-shaped region. The diffeomorphic optical flow method with the implementation in[40] cannot correctly capture the large deformation in this synthetic image, therefore we compare our result with the additive method. In Figure 3.4, the first row shows the moving image (Figure 3.4(a)) and the fixed image (Figure 3.4(b)). The second row shows the results of the additive update and the results of our method using composition update are shown in the third row. The Rs of the composition method and additive method is 0.9369 and 0.9504 respectively. Though the



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 3.3: First row: (a) the fixed Image, (b) the moving Image, and (c) synthetic forward deformation grid. Second row: result from the optical flow method; (d) deformed image, (e) intensity differences, and (f) forward deformation grid. Third row: result from our proposed method; (g) deformed image, (h) intensity differences, and (i) forward deformation grid.

additive method obtains a higher *Rs* value, the corresponding deformation grids (Figure 3.4(g) and Figure 3.4(h)) looks chaotic and is unable to provide useful information for practical use. This deficiency is due to the additive update only involving vector addition; the fact that large deformation should be based on composition is neglected. The composite method updates the displacement field by warping it with the previous transformation, providing a smoother and more accurate outcome. It is shown in Figure 3.4(d) and Figure 3.4(e) that the transformation map from the composite method can depict clearly how the image is deformed. Please refer to main2DCShape.m for the Matlab code of this example.

3.6.3 3D registration

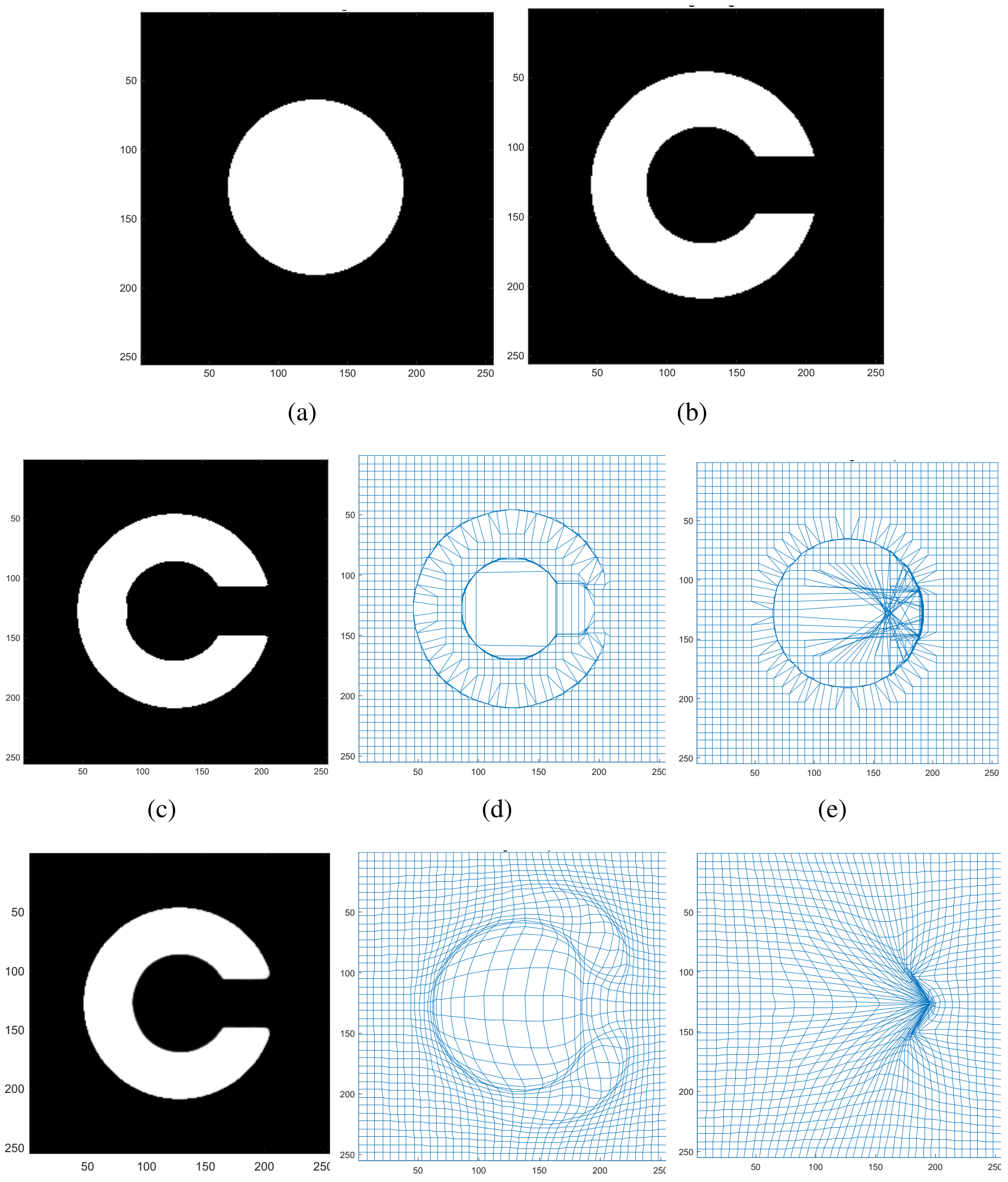
3D registration of synthetic images is illustrated in Figure 3.5. In this example, we register a sphere to a star-shape object. The moving image (sphere) is shown in Figure 3.5(a). Figure 3.5(b) shows the fixed image (the star-shape object), which is generated by distributing cones slightly above a small sphere. The star is generated in a way that the fluid characteristic of diffeomorphic registration can be illustrated clearly. Figure 3.5(c) and Figure 3.5(d) show the registration results generated using $\gamma = 1.54$ and $\gamma = 0.1$ respectively. In diffeomorphic registration, tears or folding of parts of the image should not occur. However, an approximation can be obtained as can be seen in Figure 3.5(c). We can observe that the cone remains attached to the sphere, but the part that links the cone to the sphere is compressed to be narrower. When γ is set to be at least 1.5376, diffeomorphisms are guaranteed, and the output is smooth. Note that the inequality $\gamma \geqslant 1.5376$ is not always sharp. Thus, when γ is set to a lower value, it may better able to capture sharp corners or other ”non-smooth” feature of the image (see Figure 3.5(d)). The main Matlab file that run this is example is main3D.m.

3.6.4 3D registration of medical image

In the last example, we test our method on 3D medical images of size 181 x 217 x 181 voxels. We demonstrate the application of our algorithm on registering the anatomical models of brain images obtained from[41]. These models consist of 12 classes of labels, one representing the background, and each of the others represents different classes of brain tissues. We computed the Dice similarity for the overlap regions. Given two different labels, denoted by w_1 and w_2, the Dice similarity is given by:

$$D(w_1, w_2) = \frac{\mathcal{F}(w_1 \cup w_2)}{\mathcal{F}(w_1) + \mathcal{F}(w_2)} \tag{3.29}$$

where $\mathcal{F}(w)$ is a function that returns the volume of w. For identical labels, $D(w_1, w_2) = 1$ indicates a perfect overlap and small value of $D(w_1, w_2)$ implies a low overlapping. In these examples, we used three resolution levels for registration, the average computation time used in an iteration for each level is shown in Table 3.1. We compared our method again with the diffeomorphic version of the optical flow method[39]. The Dice

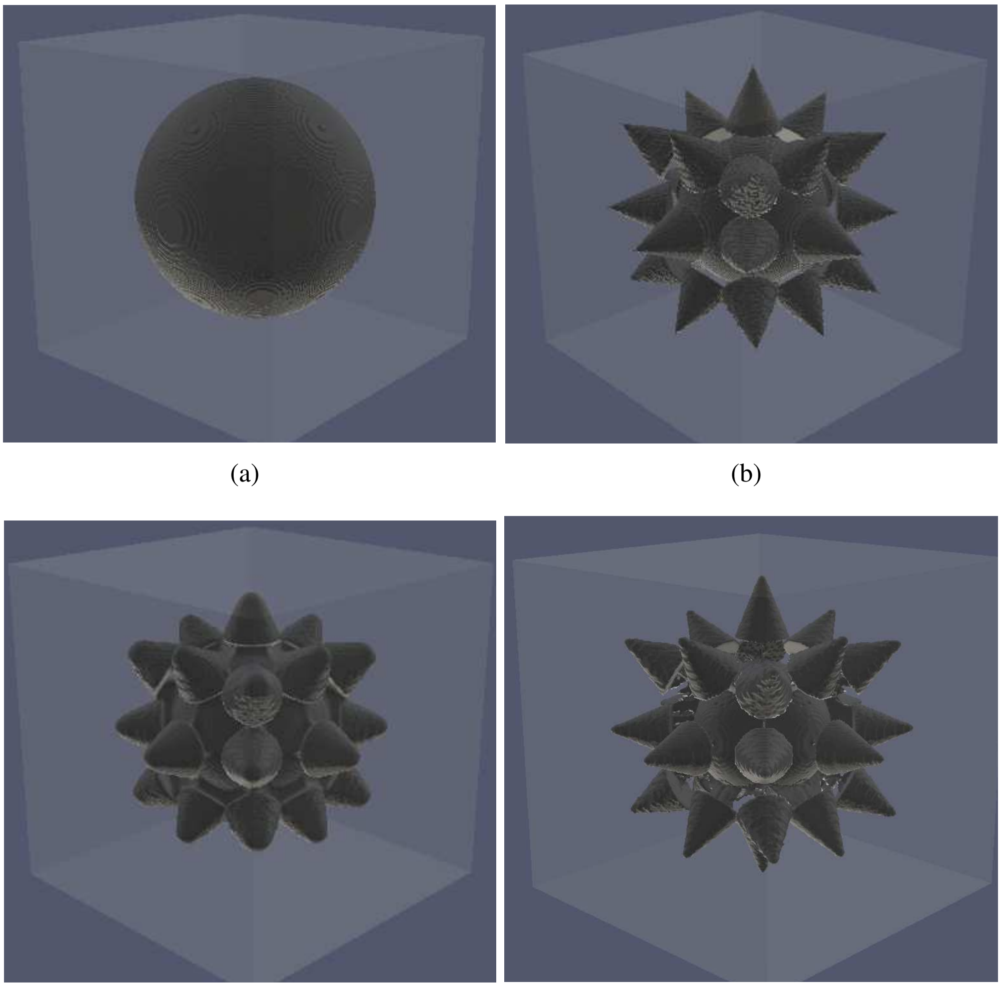


(a) (b)

(c) (d) (e)

(f) (g) (h)

Figure 3.4: Comparison of our method (using the composition update) with the additive method. First row: (a) the moving image and (b) the fixed image. Second row: results from the additive method; (c) deformed image, (d) forward transformation grid, and (e) inverse transformation grid. Third row: results from the composition method; (f) deformed image, (g) forward transformation grid, and (h) inverse transformation grid.



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

Figure 3.5: Registration of synthetic 3D images. (a) The moving image, (b) the fixed image, (c) deformed image computed using $\gamma = 1.54$, and (d) deformed image computed using $\gamma = 0.1$.

Table 3.1: Comparison of average computation time per iteration

method	proposed method	optical flow method
level 3	1.9631sec	1.3420 sec
level 2	3.6069 sec	9.2602 sec
level 1	20.3873 sec	85.8869 sec

similarity is computed for all the labels and the average of 10 model sets is summarized in a chart (Figure 4.2). From the chart, we can see that our proposed method produces a better match between the tissue types compared to the optical flow method. The 2D slices of a selected set of models are given in Figure 3.7. The first row are the slices of the fixed image (Figure 3.7(a)) and the moving image (Figure 3.7(b)). The comparison of results is given in the second row. The Matlab file that shows this example is given in main3DDiceSimilarity.m, however, to use this code, the reader should download the anatomic Brain model from[41] and put them in the current directory.

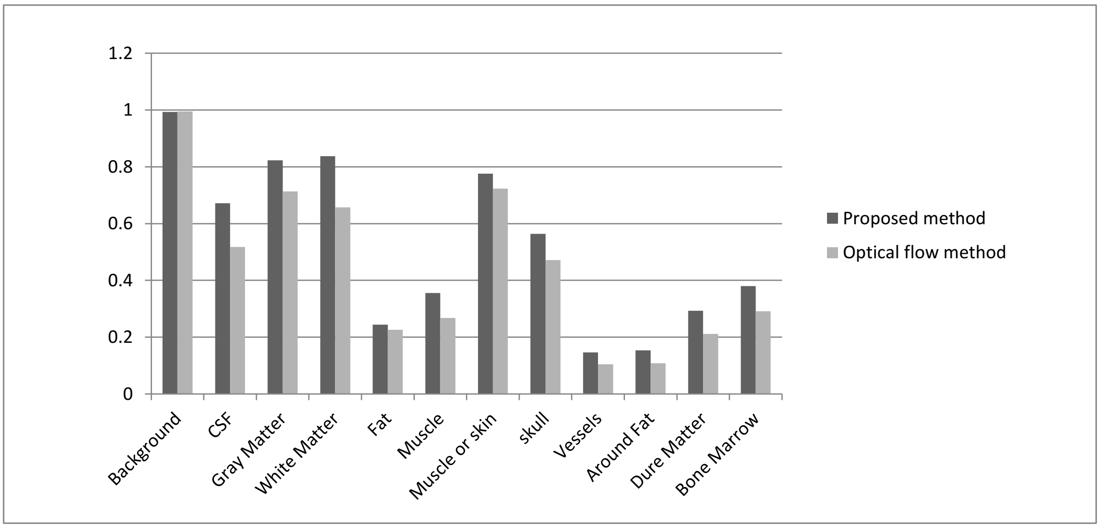
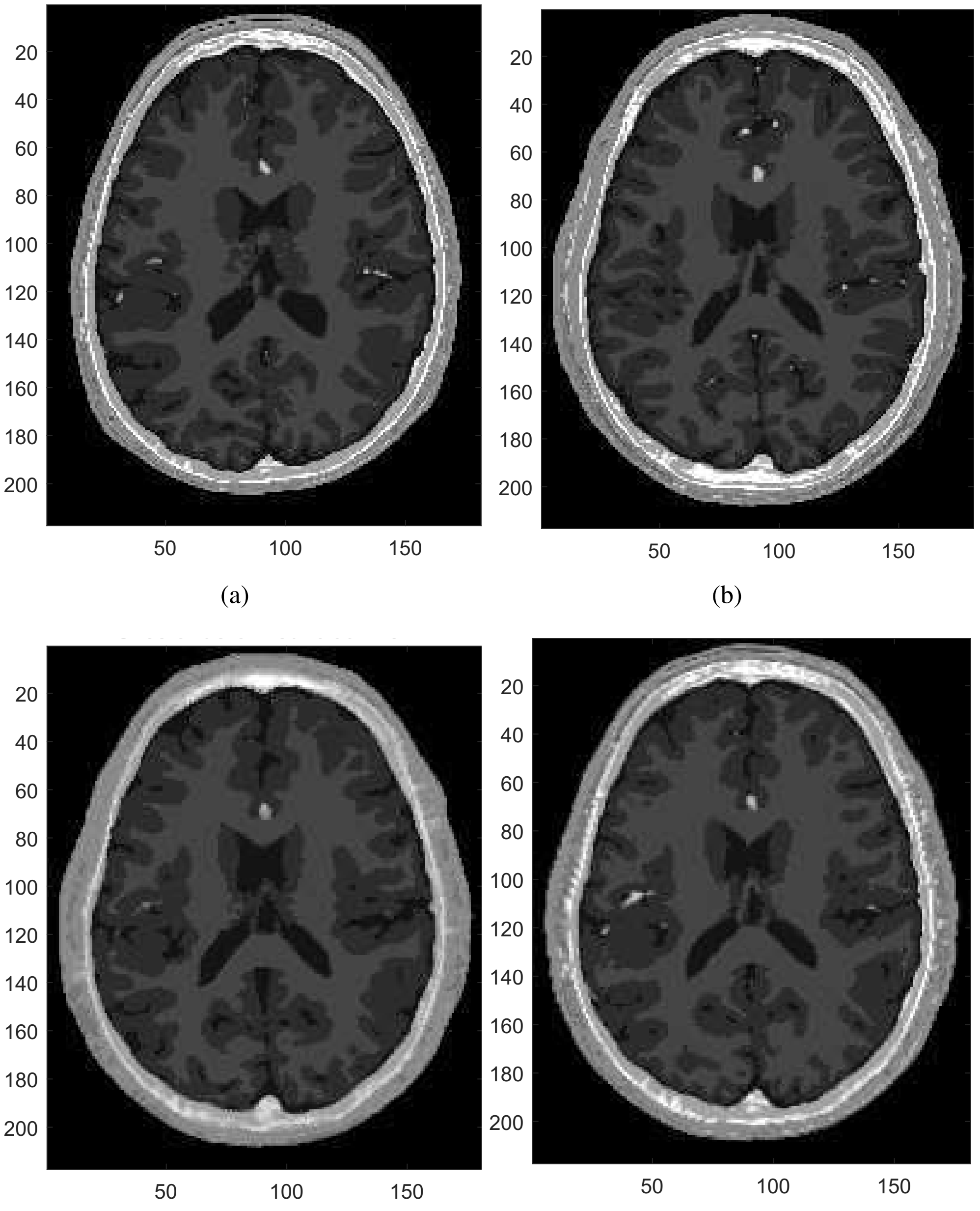


Figure 3.6: Comparison of the average of segmentation result from 10 model sets between the proposed method and the optical flow method.

3.7 Conclusions

In this chapter, we have discussed the representation of pixel and voxel-based curves and surfaces using B-splines convolution. We implement the filtering technique to improve the computation efficiency of B-splines coefficients. On this basis, we have proposed a non-rigid image registration algorithm. An appealing advantage of the proposed registration algorithm is that it is done within the B-splines framework. The



(a) (b)

(c) (d)

Figure 3.7: 2D slice of brain image. First row: (a) the moving image and (b) the fixed image. Second row: (c) result produced using our proposed method, (d) result form the optical flow method.

boundaries and deformation of the images are both represented in terms of B-splines, which allow the update of images using B-splines composition. We also perform the computation tasks in the registration process by using filters which enables parallel computing and reduces the computation cost significantly. It is shown in the numerical section that our proposed method can produce a smooth transformation map between the images (even for large deformations). We also showed possible medical applications of our approach, whereby we applied it for registration of brain images. Compared to the optical flow method, the output from our proposed method shows a better match between the registered image and the target image.

Chapter 4

Volumetric parameterization

In the previous chapter, we have studied the usage of B-splines for image representation, and its implementation in the image-related problem. In this chapter, we will discuss its application for developing the analysis model of voxel-based data. It is challenging to generate the analysis suitable mesh of complex shape when little *a priori* structure is known about the domain. A tensor-product basis can represent easily "boxy" domains leading to accurate geometry descriptions with few degrees of freedom. However, a tensor-product basis is less suitable for domains with inclusions or other irregular features.

In this contribution, we propose an automatic method to construct a PHT-spline model that matches the geometry features of a given physical domain. While the proposed method is applicable to other types of input data, such as a boundary spline representations or trimmed NURBS surfaces, we focus on obtaining volumetric descriptions of objects of interest from the voxel-based image.

The literature reviews are given in Section 4.1, followed by the discussion for obtaining smooth boundary representation from B-splines level set in Section 4.2. The proposed method is presented in Section 4.3. In Section 4.4, we illustrate our approach with several examples in two and three dimensions and show good performance on some standard benchmark test problems. Finally, a conclusion is given in Section 4.5.

4.1 Previous studies

For a given boundary representation, but the domain can be approximated by a (curved) rectangle or cube, then a volumetric description can be constructed using a Gordon-Coons parametrization[42]. However, in general, it cannot be guaranteed that the obtained geometry will be suitable for analysis. A discussion on using B-splines focused on obtaining meshes of complex geometries through an optimization method is presented in[43]. Alternatively,[44] presented a parametrization using volumetric T-splines. Another approach using an Isogeometric Boundary Element Method[45], which only requires discretization of the boundary to represent the geometry and conduct the analysis.

The approach used in this work is in some ways related to other methods for dealing with complex boundaries, such as the Immersed Boundary Method (IBM), Immerse Finite Element Method (IFEM)[46], and the Finite cell method (FCM)[47]. IBM and IFEM are widely used in fluid mechanics. However, they tend to introduce geometric modeling errors which negatively impacts the accuracy of the solutions. The idea of FCM was to apply adaptive analysis over a mesh of simple geometry, using a highly refined integration mesh to capture the limits of the domain. Nevertheless, this approach can lead to ill-conditioning if the elements cut by the boundary are too small, and the approximate geometry representation typically limits the accuracy of the method. Another related method is the Immersed Particle Method[48], which is based on the same concept as IFEM but has been developed for modeling fluid-structure interaction. In[49], a surface reconstruction from scanned image using B-spline is presented. The idea of this method is to project the grid points extracted from voxel data in the normal direction onto the surface. The Weighted Extended B-Splines (WEB-splines) [50] provide another interesting approach for representing free form geometries in the context of finite element simulations. In this method, B-splines are marked as inner or outer based on the location of their support and are weighted by a distance function. The resulting basis is non-polynomial and not fully compatible with existing CAD implementation. In[51], an immersed B-spline method was proposed to interpolate the domain boundaries using an isoparametric basis. However, the normal and curvature information of the surface is not considered and seems more difficult to incorporate along with non-uniform refinement due to the C^2 continuity of the basis.

Also related to our work, we can mention Cartesian grid methods, which is another approach for determining solid meshes. These methods are also suitable for analyzing voxel-based data, and the idea is to use the voxel structure as the computational domain. However, because many 3D images obtained using current technology can contain hundreds of millions of voxels, these methods are typically restricted to linear finite element approximations. Meshfree particle methods which have been developed based on Cartesian grids are introduced in[52]. In the recent method presented in[53], transfinite mapping functions are used at the elements cut by the boundary to avoid integration error. Reparameterization methods[54] are another approach to volumetric parametrization. Boundary reparameterization is performed as a pre-processing step to improve the isoparametric structure. The optimal control points and weights of the reparameterized surface are then obtained using a variational harmonic metric. The idea of using harmonic mappings to determine a planar parametrization is also explored in[55].

An appealing advantage of our method is that a complex domain can be meshed automatically by using only one patch of PHT-splines. In addition, an initial boundary representation such as a boundary triangulation is not necessary. Thus the method is in this sense self-contained. The key aspect of our proposed method is that we match the boundary of the solid in the physical domain by approximating the boundary with a single template consisting of a circular-arc in 2D and a sphere-surface in 3D. This approach allows the construction of C^1 representation of a domain of arbitrary complexity using only translations, rotations, and scaling operations at the boundary nodes. The resulting geometric mapping may contain non-regular points, which can also be found on a circle or sphere parametrization based on a tensor-product basis. However, different from the immersed boundary or weighted B-spline method, the domain representation is based on just Bernstein-Bézier polynomials with the usual control points and defined on a hierarchical mesh. Moreover, the reduced overlap between the basis functions allows for more local and granular control over the parametrization at the mesh vertices.

4.2 B-spline boundary representation

In this section, we discuss the usage of B-spline level set method described in Section 3.2 for obtaining the boundary description which is suitable for volumetric parameterize. Recall that a voxelized image is represented by a discontinuous gray scale function $\mathscr{G} : \Omega \to \mathbb{R}$ as follows:

$$\mathscr{G}(X) = \begin{cases} \mathscr{I}_1 & : X \in \Omega^1_{voxel} \\ \mathscr{I}_2 & : X \in \Omega^2_{voxel} \\ \vdots & \\ \mathscr{I}_{m_{voxel}} & : X \in \Omega^{n_{voxel}}_{voxel} \end{cases}$$

where $\Omega \subset \mathbb{R}^d$ is the image domain, X represents the voxel position and $\mathscr{I}_i$, $i = 1, 2, \ldots, n_{voxel}$ is the gray scale value in the image of each voxel $\{\Omega^e_I\}_{e=1}^{n_{voxel}}$. In this study, we assume that the domain of interest (*computational domain*) in the image has a lower gray scale intensity than the rest of the image. Then it can be approximated as the set $\{X : \mathscr{G}(X) \leq \tilde{T}\}$, where $\tilde{T}$ is a user-definable threshold which determines the boundary of the domain. Moreover, if $\hat{\mathscr{G}}(X)$ is a suitably smoothed representation of $\mathscr{G}(X)$, then the boundary is represented by the contour line $\{X : \hat{\mathscr{G}}(X) = \tilde{T}\}$ and the following fundamental properties of level set function hold (in both 2D and 3D):

1. The unit normal vector to the contour line (boundary) at point X is given by $\frac{\nabla \hat{\mathscr{G}}}{|\nabla \hat{\mathscr{G}}|}$.

2. The curvature of the contour line is $\nabla \cdot \frac{\nabla \hat{\mathscr{G}}}{|\nabla \hat{\mathscr{G}}|}$.

Though it is possible to obtain a contour line directly from pixel data, the boundary may contain jagged edges or noise artifacts from the image and the approximate gradients computed from the pixel intensity values may be inaccurate. Smooth boundary representation of a given image can be obtained by computing an approximation of it using a B-spline level set function

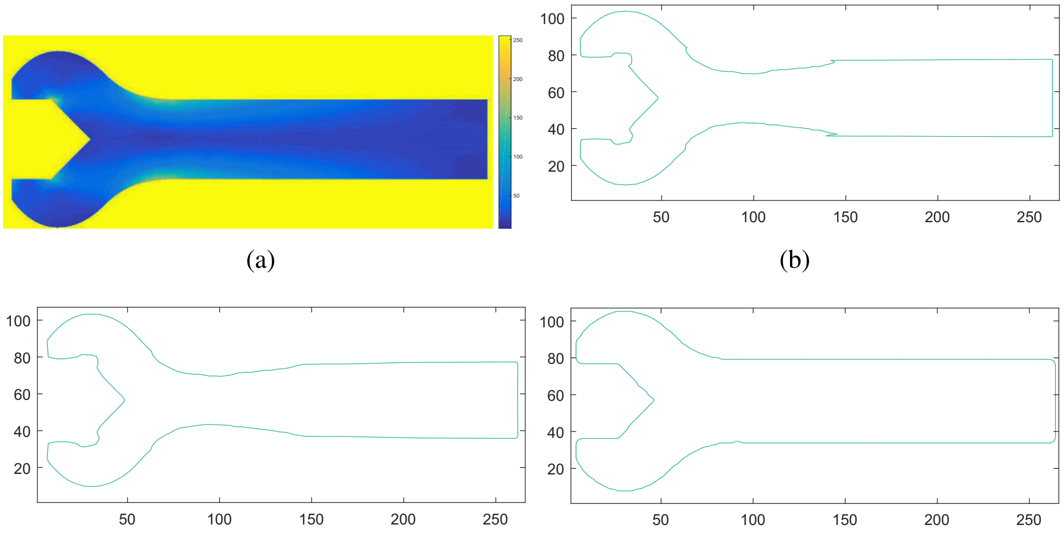
$$\mathcal{B}(X) \approx \mathscr{G}(X),$$

where the computation of $\mathcal{B}(X)$ is described in the Section 3.2. We note that by adjusting the threshold value, a good level set representation of the domain of interest can be obtained. To further ensure that the gradients around the boundary are predictable and not affected by image noise, a pre-processing step before convolution can be used in the case where the image is not black and white. We set the intensity for all the voxels outside the computational domain to 255, while intensity of the voxels inside the computational domain is set to 0. This essentially converts the gray scale image to a black and white image where black represents the domain of interest.

The comparison of the boundary obtained using the above mentioned approach and boundary obtained directly form the discrete gray scale function is given in Figure 4.1. In Figure 4.1(a), we reproduce a spanner image from[45]. The contour obtained by the B-spline level set representation is given in Figure 4.1(c), and Figure 4.1 is the contour obtained from the gray scale function for a particular intensity threshold. It is clear that the contour in Figure 4.1(c) is smoother and less prone to compression artifacts. The improved level set boundary representation after resetting the image to black and white colors is shown in Figure 4.1(d).

4.3 Volumetric parametrization of voxelized data

Here, we propose an approach to obtain a smooth mesh representation of an image using PHT-splines $T(\xi)$. The method involves refinement of boundary elements and also adjustment of vertices. For a given image $\mathscr{G}(X)$, we first obtain the boundary description by computing $\mathcal{B}(X)$ as discussed in Section 4.2. We then generate a domain Ω which has a much simpler geometry, but with a resolution which at least captures the coarse details of the image. In 2D, we consider Ω to be a rectangular tensor-product mesh, while in 3D, a cuboid is used as Ω. Followed by embed $\mathcal{B}(X)$ into Ω, and set the points where the level set value is below the threshold $\tilde{T} = 200$ as the physical domain. We then proceed with the Algorithm 2. Each step in Algorithm 2 is presented in more detail in the following subsections.



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

Figure 4.1: (a) Spanner image, (b) contour from gray scale function, (c) contour from level set function, and (d) contour from level set after resetting the intensity to 0 and 255.

```
Algorithm 2: Volumetric parametrization
Input   : level set function $\mathcal{B}(X)$, PHT-mesh, $\Omega$ embedded with $\Omega_{phys}$,
          intensity threshold $\tilde{T}$ , $\theta\%$
Output: Modified PHT-mesh, where the boundary of active domain $\Omega_{active}$
          matches the boundary of $\Omega_{phys}$
1: Determine and classify boundary elements. (Algorithm 3)
2: Determine adjustable vertices. (Algorithm 4)
3: Adjust vertices using the circle or sphere template. (Algorithm 5)
4: Smooth and untangle distorted elements.
5: If a better boundary representation is desirable, refine the boundary elements and
   their neighbors, and repeat from 1.
```

4.3.1 Classifying the boundary elements

We classify the elements in Ω into 4 groups as follows:

- Boundary element of type 1 are elements cut by the contour of Ω_{phys} and more than $\theta\%$ of the volume has intensity less than $\tilde{T}$.
- Boundary element of type 2 are elements cut by the contour of Ω_{phys} and less than $\theta\%$ of the volume has intensity greater than or equal to $\tilde{T}$.

- Internal elements are elements inside the physical domain.
- External elements are elements outside the physical domain.

The parameter $\theta\%$ is chosen based on the image data in order to prevent very small elements from being considered in the computational domain. A value of $\theta = 50$ works well in most cases. The classification of each element is illustrated in Figure 4.2(a), with the red color cells, green color cells, blue color cells and white color cells representing Boundary elements of type 1, Boundary elements of type 2, internal elements and external elements respectively.

The goal is to obtain an active domain $\Omega_{active} \subset \Omega$ such that $\Omega_{active} \approx \Omega_{phys}$. The Ω_{active} consists of boundary elements type 1 and internal elements, i.e. $\Omega_{active} = \Omega_{B1} \cup \Omega_{In}$. Our proposed approach is to deform the boundary elements of type 1 and 2 such that the boundary of Ω_{active} matches the boundary of Ω_{phys}. We note that the exterior elements and elements of type 2 are classified as *inactive elements*. While they are part of the hierarchical mesh and have associated control points, these elements are not considered during the assembly and solution of the given PDE.

Algorithm 3: Determining and classifying boundary element

```
Input  : PHT-mesh, level set function ($\mathcal{B}(X)$), intensity threshold $\tilde{T}$ , $\theta\%$
Output: Boundary elements
1: Determine equally-spaced sample points in $\mathcal{P}$.
2: For the sample points, compute the intensity value of these points using $\mathcal{B}(X)$.
3: If the intensity value is less than given intensity threshold $\tilde{T}$, set the point as an
   internal point, otherwise, set it as an external point.
4: Compute the number of internal points and external points within an element.
5: If more than $\theta\%$ of the points are internal points, set the element as Boundary
   element type 1.
6: If less than $\theta\%$ of the points are internal points, set the element as Boundary
   element type 2.
7: If all points are internal points, set the element as Internal element.
8: If all points are external points, set the element as External element.
9: Set the Boundary element type 1 and the Internal elements as the active elements.
```

4.3.2 Determining the adjustable vertices

After classifying the elements, we select the *adjustable vertices* from the boundary elements as follows:

- A vertex belonging to a Boundary element of type 1 is marked “adjustable” if the intensity value at the corresponding location is greater than or equal to the threshold $\tilde{T}$ (it lies outside Ω_{phys}).

- A vertex belonging to a Boundary element of type 2 is marked “adjustable” if the intensity value at the corresponding location is less than the threshold $\tilde{T}$ (it lies inside Ω_{phys}).

We would like to adjust (move) the selected vertices so that they touch the boundary of Ω_{phys}, hence enforcing that the boundary of Ω_{active} resembles the boundary of physical domain. The control points associated to the vertices are then adjusted by using a predefined template for the unit circle or sphere, as discribed in the following subsections.

Algorithm 4: Determining adjustable vertices

```
Input  : Boundary element type 1 or 2
Output: Adjustable vertices, intensity threshold $\tilde{T}$
1: Compute the intensity value at each vertex.
2: For Boundary element type 1, mark the vertices with intensity value greater than $\tilde{T}$ as adjustable vertices.
3: For Boundary element type 2, mark the vertices with intensity value less then $\tilde{T}$ as adjustable vertices.
```

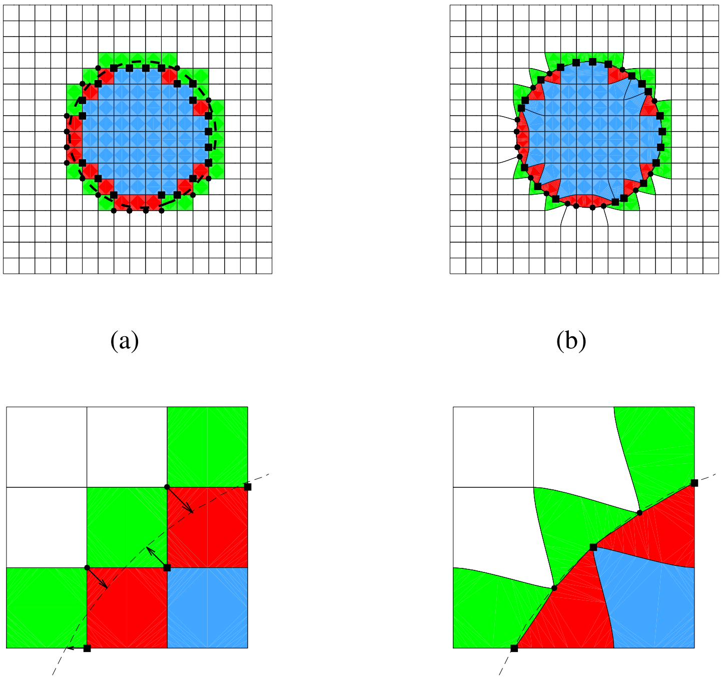
4.3.3 Adjusting vertices

In our proposed method, the vertices are moved to the boundary in a similar way as presented in[51] and[49]. In[51], the mesh is modified by weighting the B-splines so that mesh vertices of the boundary elements move to the closest location on the boundary. In[49], the vertices are first smoothed before they are projected in the normal direction onto the boundary using the Newton-Raphson method. The control points are then recovered by an interpolation method which involves solving a linear system of equations.

The proposed method intends to move the vertices of the boundary elements by taking advantage of the fact that each basis vertex is associated with 2^d control points. We move the vertices by assigning new locations to the corresponding control points, which determine locally the geometric parametrization. The locations of the control points and mesh geometry are obtained from the translated, scaled, and rotated circle or sphere template. Therefore, an advantage of the proposed method is that the geometric information of the level set function (such as normal and curvature) can be approximated by the PHT-splines.

4.3.3.1 Determining the new vertex location

We move an adjustable vertex along the edge or diagonal of an element according to type of elements that they are connected to. In 2D, if the adjustable vertex is shared by two boundary elements of the same type, we adjust it along the shared edge of these elements otherwise it is adjusted along the diagonal direction. A 2D example of the proposed method is illustrated in Figure 4.2. In the first figure (Figure 4.2(a)), we embed the level set representation of the image into an uniform mesh and classify the boundary elements. The Boundary element type 1, Boundary element type 2, and internal elements are highlighted with red, green and blue color respectively. The union of the red and the blue elements forms the active domain, which will be used for analysis. Adjustable vertices are marked with black dots and squares, where the dots are the adjustable vertices of Boundary element type 1, and squares are the adjustable vertices of Boundary element type 2. Figure 4.2(b) shows the result after adjustment.



(a) (b)

(c) (d)

Figure 4.2: 2D example of the boundary matching algorithm, (a)initial mesh and element classification, (b) after adjusting the selected vertices of boundary elements, (c) zoomed in view of the adjusted vertices and (d) the resulting bounding after adjustment.

In 3D, the determination of the adjusted direction is classified into 3 cases as follows:

1. Adjustable vertex belongs to only one boundary element (see Figure 4.3(a)).
2. Adjustable vertex shared by two adjacent boundary elements (see Figure 4.3(b)).
3. Adjustable vertex shared by four boundary elements (see Figure 4.3(c)).

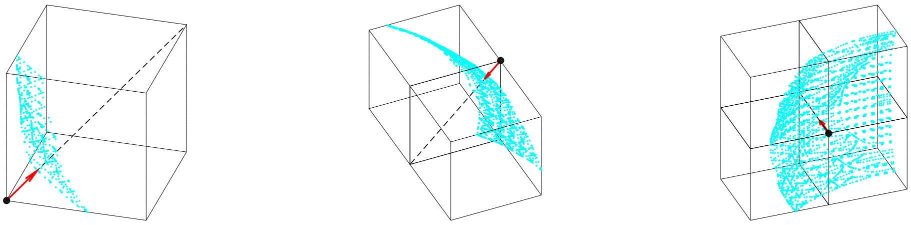

(a) (b) (c)

Figure 4.3: 3D classification of adjust direction (a) vertex belongs to one boundary element, (b) vertex shared by two adjacent elements, and (c) vertex shared by four elements.

In Figure 4.3, the black lines represent the element edges, the blue dots are the level set contour (boundary of the physical domain) cutting through the elements, and the black dots are the adjustable vertices. Figure 4.3(a) illustrates the case when the adjustable vertex belongs to only one element. In this case, we adjust the vertex along the cube diagonal. In Figure 4.3(b), the adjustable vertex is shared by two adjacent elements, and the shared face diagonal is selected as the adjustable direction. For the third case, where the adjustable vertex is shared by four elements and these four elements share an edge, as shown in Figure 4.3(c), we move the adjustable vertex along the shared edge. In all these figures, the red arrow represents the moving direction of the adjustable vertex, and the black dashes represent the diagonal. After determining the adjustable direction, we apply the bisection method to compute the cut location of the physical boundary at the selected edge or diagonal. In 3D, for the adjustable vertices which do not belong to any of these three cases, we set the nearest point on the physical boundary as the cut location.

4.3.3.2 Circle and sphere templates

Consider a 2D cubic PHT mesh $T(\xi,\eta)=\sum_{j=1}^{n}N_j(\xi,\eta)P_j$, where $N_j(\xi,\eta)$ are the basis functions and P_j are the control points. Suppose (ξ_a,η_a) is an adjustable vertex;

there will be four basis functions associated with it. The geometric information of this vertex can be written in the matrix form as:

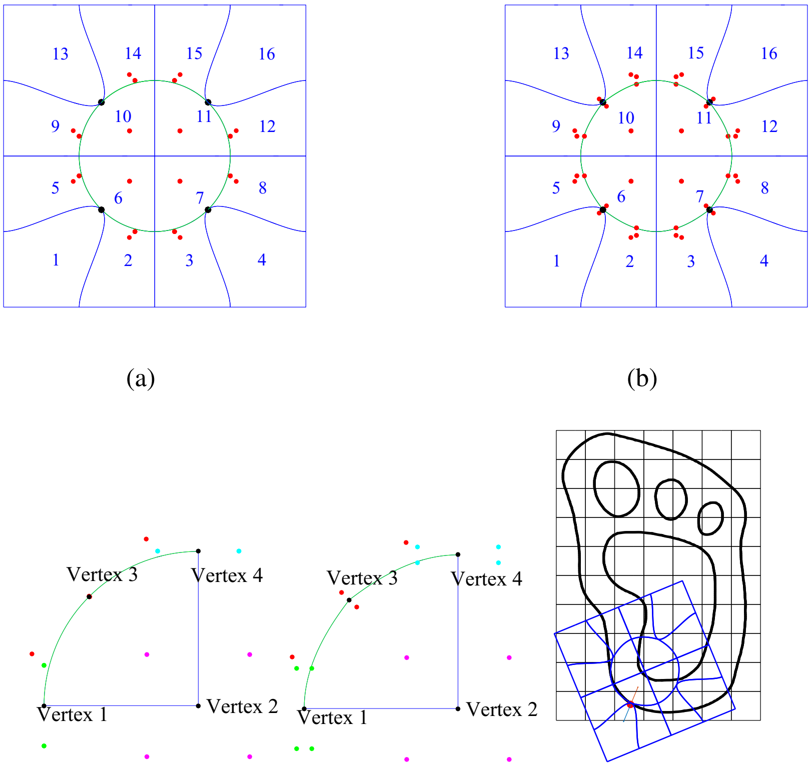
$$\mathscr{L}T(\xi_a,\eta_a) = \sum_{i=1}^{4} \mathscr{L}N_{j_i}(\xi_a,\eta_a)P_{j_i}, \tag{4.1}$$

where

$$\mathscr{L}N_{j_i}(\xi_a,\eta_a) = [N_{j_i}(\xi_a,\eta_a) \quad \frac{\partial N_{j_i}(\xi_a,\eta_a)}{\partial \xi} \quad \frac{\partial N_{j_i}(\xi_a,\eta_a)}{\partial \eta} \quad \frac{\partial N_{j_i}^2(\xi_a,\eta_a)}{\partial \xi \partial \eta}].$$

We would like to find the control point P_{j_i} such that at the vertex, the outer normal of the boundary edge matches that of the contour line of the level set. The normal of an element in the parameter space and the normals in the physical space are related to the first derivatives of the mapping T by Nansen's formula, which is often used when imposing Neumann boundary conditions in isogeometric analysis. We note that there are several cases depending on whether the adjustable vertex is a corner vertex of the active domain or if it is shared by more than one active elements. Also, the magnitude of the normal vectors and the values of the cross derivatives of the mapping (i.e. $\frac{\partial T}{\partial \xi \partial \eta}$), which correspond to the "corner twists" of the element, are free parameters which need to be estimated. In[9], a local parametrization using quadratic polynomials is suggested as the basis for extracting the geometric information. This procedure may however be computationally expensive. For these reasons, we propose to use a template that is scaled and translated to each element, as shown in Figure 4.4.

A good way to analyze the shape of a curve is to use the osculating circle. The osculating circle at a point x on a curve is the circle that approximates the curve by matching its tangent and curvature[56]. Instead of solving a linear system, we propose to use the corresponding control points from a circle template. The resulting control points will generate an arc passing through the adjusted vertices and approximating the geometric information of the physical boundary. First, we generate a circle template that approximates the unit circle. This template is formed by 16 elements, and the approximating circle is composed by four elements labeled 6, 7, 10 and 11 (see Figure 4.4(a)). In the template, the control points and the vertices are represented by the red dots and black dots respectively, and the circular arcs are approximated by the edges highlighted with green color. We note that some of the control points are repeated, which results in a singularity in the geometric mapping and may cause problems in analysis. We can avoid this problem by slightly separating the repeated control points. The control points of the element labeled 10 before and after the separation are shown in Figure 4.4(c) and 4.4(d) respectively. In these figures, the black dots represented the vertices, and the associative control points of each vertex are marked with different colors as: Vertex 1 - green , Vertex 2 - purple, Vertex 3 - red, and Vertex 4 -cyan. In 3D, we approximate the boundary using a sphere template as shown in Figure 4.5. The



(a) (b)

(c) (d) (e)

Figure 4.4: Osculating circle example, (a) circle template with repeated control points, (b) alternate circle template, (c) element 10 with repeated control points, (d) element 10 with separated control points, and (d) Yeti footprint and a scaled and rotated circle template at a selected contour point.

distance error from the actual unit sphere is plotted, and we can see that even with eight elements the error is relatively low.

To use the circle template, we start by computing the curvature (κ) and normal direction at the cut location from the level set function. Next, we generate a circle using the template by setting the radius $r = 1/\kappa$. The circle is then rotated such that the normal at the corresponding vertex matches with the normal obtained from the level set. A good boundary approximation can be obtained by replacing the associated control points with those obtained from the rotated and scaled circle template. Figure 4.4(e) illustrates how a selected part of the contour is approximated by using the circle template. The circle template is scaled and rotated to match with the geometric information at the given point (represented by a dot in the 4.4(e)) of the contour, and we will approximate the boundary with an arc of the circle template. We note that the

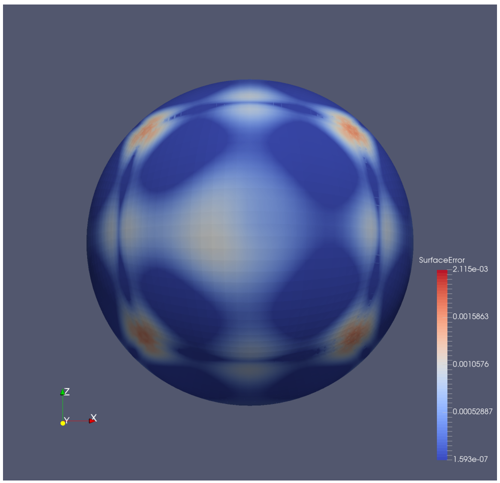


Figure 4.5: Sphere template (the interior 8 elements which approximate the unit sphere are shown) and surface error.

scaling should take into account both the curvature at the boundary and the size of the element to avoid fold-overs. We use the sphere template in a similar way to the circle template.

We mention that the adjustment may lead to distorted elements and non-regular points. However, the mesh quality can be improved with a smoothing operation which will be discussed in Section 4.3.4. In addition, a better approximation to the boundary can be obtained by refining the boundary elements. The refinement strategy is discussed in the Section 4.3.5.

Algorithm 5: Adjusting vertices

```
Input   : Adjustable vertex
Output: The new locations of the control points associated with the vertex
1: Find the adjustable direction.
2: Compute the cut location using the bisection method, and assign it as the new
   location of the corresponding vertex.
3: Compute normal and curvature at the new location from the level set function
   $\mathcal{B}(X)$.
4: Scale and rotate the circle template according to the normal and curvature values,
   and translate it to the new location.
5: Relocate the control points associated to the adjustable vertex according to those
   obtained from the circle or sphere template.
```

4.3.4 Smoothing and untangling the elements

In some cases, moving the vertices to the cut locations may lead to distortion. To address this problem, we apply a Laplacian smoothing operation after adjusting the vertices. Suppose P_j and P_i are the positions of j and i vertices respectively, while $\hat{P}_i$ denotes the new position of vertex i. The smoothing operation is described as:

$$\hat{P}_i \leftarrow P_i + \lambda L(P_i),\, 0 < \lambda < 1 \tag{4.2}$$

where λ is a regularization term, and $L(P_i)$ is the Laplacian smoothing defined as

$$L(P_i) = \frac{1}{n_i} \sum_{j=1}^{n_i} P_j - P_i. \tag{4.3}$$

where n_i represents the number of neighbors to the vertex i. In the numerical experiments shown below, we found that $\lambda = 0.1$ works well in most cases.

4.3.5 Refinement

In order to obtain a better boundary representation, a multi-level refinement is performed along the boundary elements. For a given PHT-mesh, refinement at each level can be generated by successively inserting new basis vertices onto the parametric domain of the boundary elements. Referring to Section 4.3.3, the vertices are adjusted by translating, scaling and rotating the associated control points. Since there are no specific control points associated to a vertex at the T-junction, it is preferable to avoid T-junctions on the boundary. Thus, in addition to refining the boundary elements, the neighbor elements are refined as well. On each refinement level, we carry out the following steps:

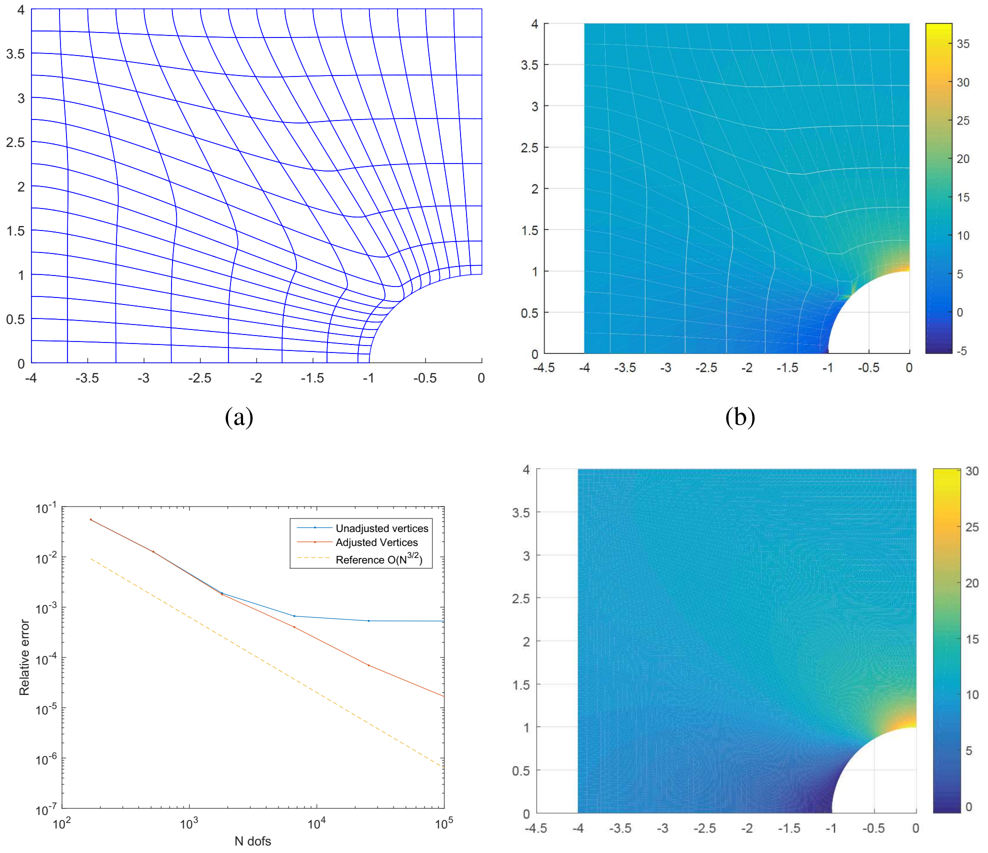
1. Determine Boundary elements of type 1 and Boundary elements of type 2.
2. Determine the elements bounding both types of Boundary elements.
3. Insert new basis vertices in the elements obtained from 1 and 2 as discussed in Section 2.3.

4.4 Numerical Examples

We illustrate the accuracy and efficiency of the proposed method with several 2D and 3D examples. The first example is a synthetic benchmark problem - a hole in an infinite plate. The second example is the Yeti footprint example used to demonstrate the efficiency of the proposed method for handling complex shapes. The third example is an open spanner example where we show the application to elastostatic analysis. We examine the proposed method in 3D using the spherical hole in an infinite domain benchmark, and a problem using the Stanford bunny as a computational domain is shown as the last example. For those examples involving analysis, the Young's modulus is set as $E = 10^5$ and the Poisson's ratio is set as $\nu = 0.3$.

4.4.1 Infinite plate with circular hole

We start by showing the example of infinite plate with circular hole subject to uniaxial tension. The problem statement and exact solution are similar as those given in Section 2.5.1.



(a) (b)

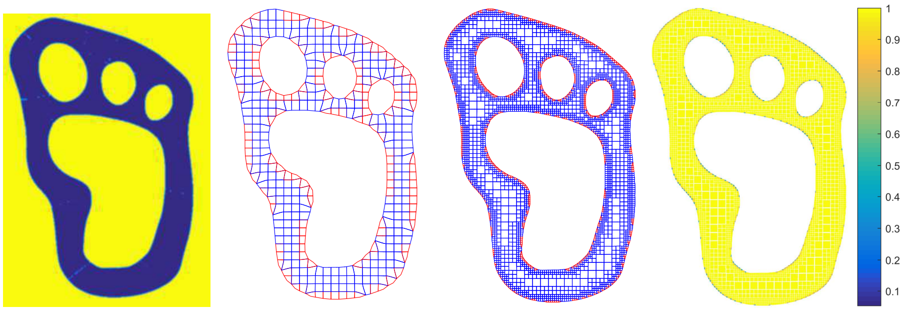
(c) (d)

Figure 4.6: Synthetic benchmark: plate with a hole (a) coarse mesh, (b) σ_{xx} component of the stress field, (c) convergence plot, (d) σ_{xx} stress on a finer mesh.

A PHT-spline representation of the active domain of the plate is shown in Figure 4.6(a), and the analysis result is illustrated in Figure 4.6(b). The analysis result can be improved when more refinement steps are performed, as shown in Figure 4.6(d). Figure 4.6(c) shows a comparison of the convergence rates. The red lines shows the convergence of the error in the energy norm for this problem when we refine the mesh and also adjust the vertices along the boundary using the proposed method. The blue line shows the result when only uniform refinement is carried out. We can see that after a few refinements, the convergence rate levels off. The convergence rate when the vertices are moved to the curved boundary at each refinement step is stable and close to the reference rate which is represented by the dashed line.

4.4.2 Yeti footprint

To demonstrate the ability of the proposed method in handling complicated shapes, including those with holes or cracks, we use the Yeti footprint as the next example. In Figure 4.7, we illustrate the representation of Yeti footprint with only one PHT-spline patch. Comparing to the example shown in[57], where 21 patches are used to form the domain, the proposed method is more straight-forward. Figure 4.7(a) shows the Yeti footprint image obtained from[57]. Figure 4.7(b) is the result of approximating the Yeti image with a coarse mesh, where we can see that the boundary is already well-resolved. However, if a better approximation is needed, the boundary representation can be further enhanced with additional refinements. The result of a finer mesh is shown in Figure 4.7(c). The scaled Jacobians of the last mesh are plotted in Figure 4.7(d).



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

Figure 4.7: Yeti footprint example, (a) Yeti footprint image, (b) active area of the PHT-mesh after adjusting vertices, (c) the result using a finer mesh, and (d) the scaled Jacobians of the finer mesh.

4.4.3 Open spanner

In this example, we demonstrate an application of the proposed method to elastostatic analysis. Figure 4.8(a) is the spanner image obtained from[45]. The volumetric representation of the spanner using adaptive refinement is shown in Figure 4.8(b), where we can see that sharp corners can be captured better by refining the elements around the corners. The result of volumetric representation using B-splines is given in Figure 4.8(c); the spanner can be represented well, but, it contains many elements. Thus, the computation is time consuming. The result of uniform refinement along the boundary is shown in Figure 4.8(d). It is clear that by refining only the boundary we can obtain a good boundary approximation, and since there are fewer elements and fewer degree of freedom, the computation is more efficient. The zoomed-in figures of the upper and lower corners at the end of the spanner are shown in Figures 4.9(c) and 4.9(d) respectively. The analysis results when the corners of the jaw are fixed and uniform traction is applied at the right edge of the handle is shown in Figure 4.9(e), while Figure 4.9(f) shows the scaled Jacobians of the parametric mesh.

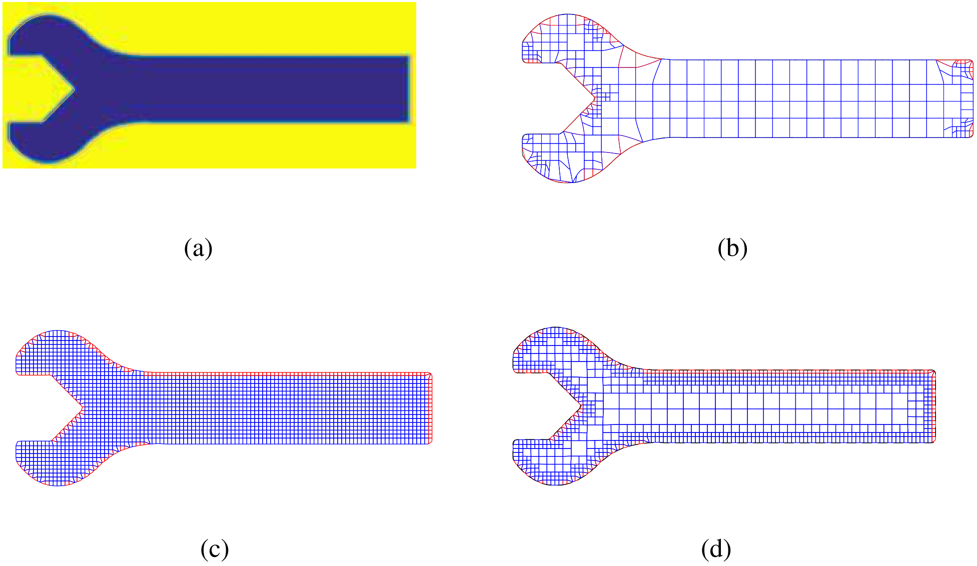
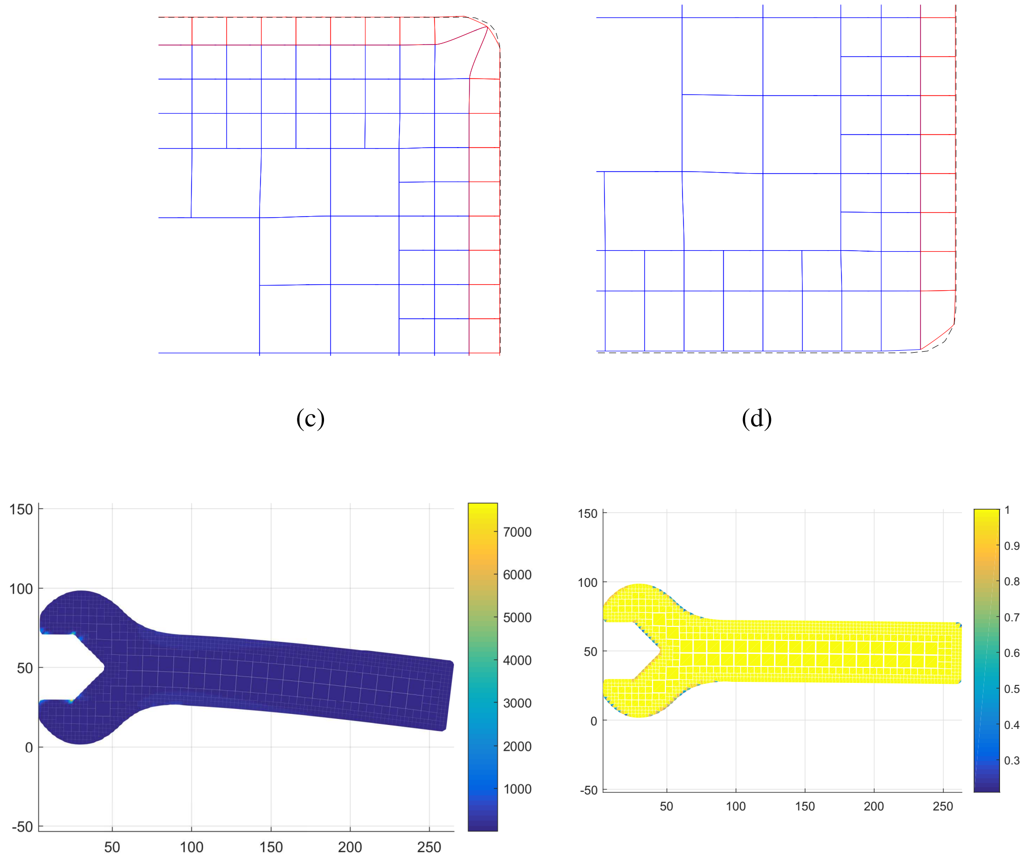

(a) (b) (c) (d)

Figure 4.8: Open spanner example, (a) spanner image, (b) result of adaptive refinement, (c) result of volumetric representation using B-splines, and (d) results of uniform refinement along boundary.



(c) (d)

(e) (f)

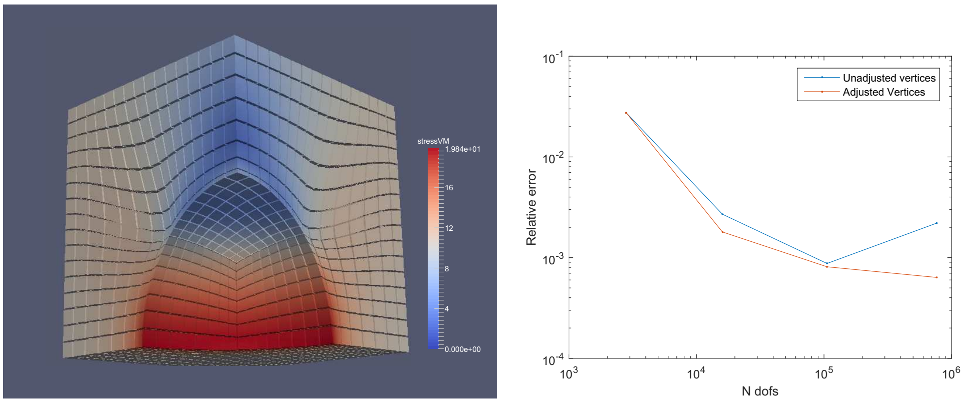
Figure 4.8: Open spanner example (cont.), (e) zoomed-in figure of the upper corned at the end of the spanner, (f) zoomed-in figure of the lower corner at the end of the spanner, (g) the analysis result, and (h) the scaled Jacobians.

4.4.4 Spherical hole in an infinite domain

The problem of a spherical hole in an infinite domain is shown is this section. In this problem, an uniaxial tension is applied at infinity in the domain, and the exact stresses are given in terms of spherical coordinates (R, θ, β) as follows[58]:

$$
\begin{aligned}
\sigma_{RR} &= \sigma_\infty \cos^2\beta + \frac{\sigma_\infty}{(7-5\nu)}\left(\frac{a^3}{R^3}(6-5(5-\nu)\cos^2\beta) + \frac{6a^2}{R^5}(3\cos^2\beta - 1)\right)\\
\sigma_{\theta\theta} &= \frac{3\sigma_\infty}{2(7-5\nu)}\left(\frac{a^3}{R^3}(5\nu-2+5(1-2\nu)\cos^2\beta) + \frac{a^5}{R^5}(1-5\cos^2\beta)\right)\\
\sigma_{\beta\beta} &= \sigma_\infty \sin^2\beta + \frac{\sigma_\infty}{2(7-5\nu)}\left(\frac{a^3}{R^3}(4-5\nu+5(1-2\nu)\cos^2\beta) + \frac{3a^5}{R^5}(3-7\cos^2\beta)\right)\\
\sigma_{R\beta} &= \sigma_\infty\left(-1+\frac{1}{7-5\nu}\left(-\frac{5a^3(1+\nu)}{R^3}+\frac{12a^5}{R^5}\right)\right)\sin\beta\cos\beta
\end{aligned}
\tag{4.4}
$$

where a represents the radius of the spherical hole and σ_∞ is the uniaxial tension at infinity. Since it is not feasible to model an infinite domain, we consider the solution on part of the domain as in Figure 4.9(a). We compare our results with results obtained from uniform refinement without changing the geometry. The comparison is shown in Figure 4.9(b). The red line is the result of refinement with changing the geometry by adjusting the vertices at the boundary, while the blue line is the result from refinement without adjusting the vertices. We can see that the result is better when we change the geometry at the boundary to approximate the desired shape.



(a) (b)

Figure 4.9: Cube with a hollow sphere example: (a) The von Misses stresses on a coarse mesh, and (b) Convergence plots for the energy norm.

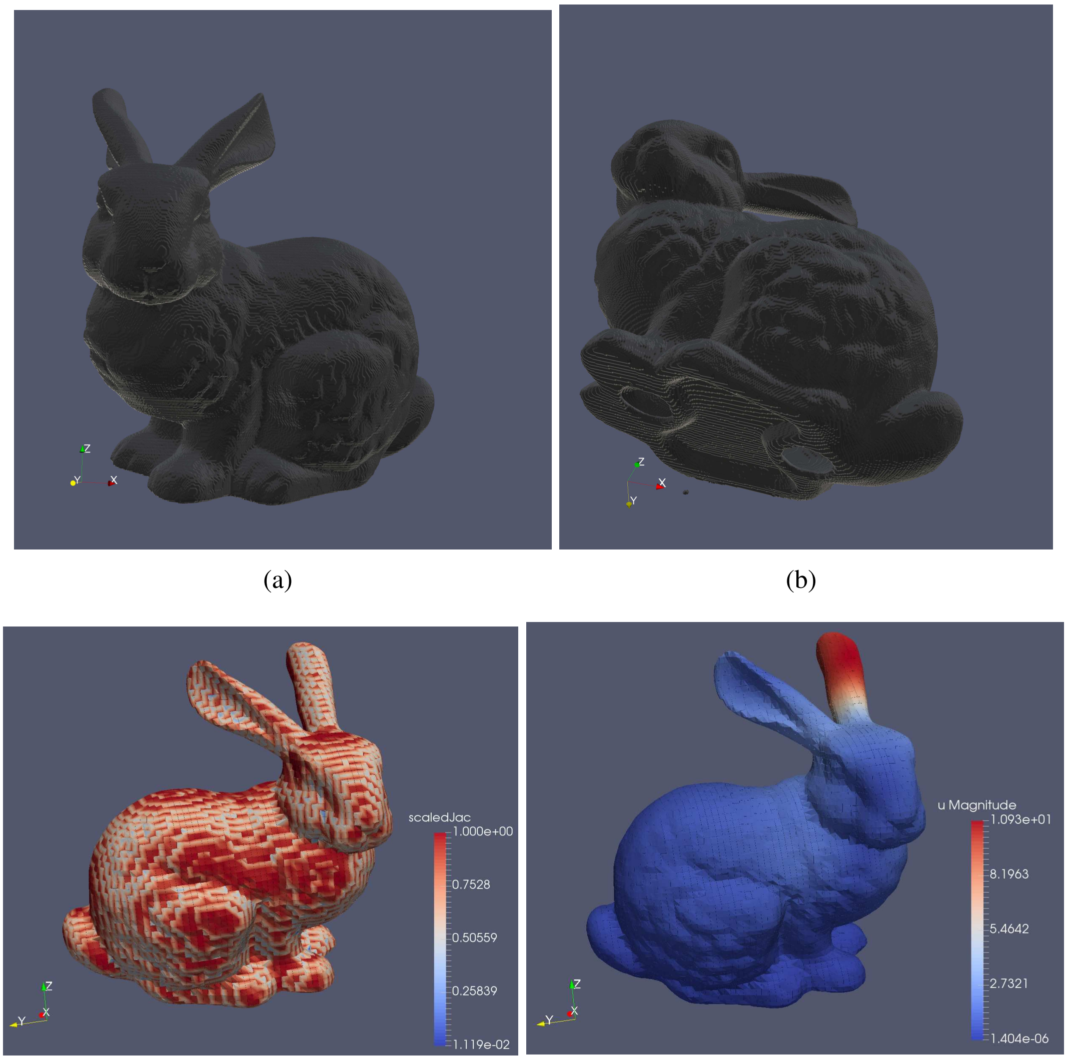
4.4.5 Stanford Bunny

In this example, we demonstrate the ability of the proposed method to handle complex 3D objects. We use the CT-scan data of the Stanford bunny given in[59]. We resize the data to 512 × 512 × 512 voxels such that the aspect ratio of the bunny is scaled to its proper size. Thus, the initial data has approximately 134 million voxels. We consider an approximation of the domain with 64 × 64 × 64 elements. After pre-processing the bunny image to obtain the level set function, the active region of the solid spline is now formed by only 13,754 hexahedral elements. Because the bunny is hollow, most of the domain is formed by inactive elements which do not contribute to the computations. The ability to model objects with voids is one the advantages of our proposed method. In addition, the resulting spline model uses only one patch. Figure 4.10(a) and Figure 4.10(b) shows the bunny and its base respectively. The volumetric representation and the scaled Jacobians are shown in Figure 4.10(c). The analysis result is illustrated in Figure 4.10(d), where we apply the displacement at one of the ears and fix the base.

4.5 Conclusions

In this chapter, we presented a volumetric parametrization method for obtaining analysis suitable meshes from given voxelized data. The proposed method starts with getting the B-splines boundary representation of the data via convolution and uses it to determine the active domain in the parametric mesh. We then extract information such as normal and curvature from the B-spline boundary. The concept of the osculating circle is employed to impose the geometric information onto the boundary of the active domain, where we use a circle template (sphere template for 3D) to facilitate the geometric approximation. Moreover, we apply the Laplacian smoothing operation to untangle the distorted element.

In the numerical section, we show that highly accurate results can be obtained due to the enhanced geometry approximation and high continuity within the parametric mesh. In particular, the examples of hole in an infinite plate shows improved results compared to the singular parametrization given in Chapter 2. Meanwhile, we also show that the proposed method can handle complex 2D and 3D shapes which contain voids or irregular geometric features with the Yeti footprint and Stanford Bunny examples.



(a) (b)

(c) (d)

Figure 4.10: Stanford bunny example, (a) level set representation of the Stanford Bunny, (b) the bottom part of the bunny, (c) solid splines representation of the bunny, and (d) the analysis result.

Chapter 5

C^1 coupling

The volumetric parametrization method discussed in the previous chapter is suitable for obtaining an analysis model from voxelized data. For a structure with known geometry, a volumetric mesh can be constructed according to the geometric information. In particular, the construction of an object with complex shapes often requires multi-patches which are typically stitched together with only C^0 continuity. In this chapter, we present two approaches for C^1 coupling on multi-patch domains.

A literature reviews is given in Section 5.1. Then we explain the concept of geometrically continuous isogeometric functions, and present the Bézier ordinates approach and splines approach by using a simple 1D example in Section 5.2. Extension of the methods to 2D is given in Section 5.3, followed by C^1 coupling on multi-patch domain in Section 5.4. A method for handling the C^1 locking issues is discussed in Section 5.5. The C^1 coupling for surfaces and 3D are presented in Sections 5.6 and 5.7 respectively, while the numerical studies are given in Section 5.8 and the conclusion is in Section 5.9.

5.1 Previous studies

Several methods have been proposed in engineering literature to deal with the decreased smoothness at the patch interfaces, such as the bending strip method or Kirchhoff-Love shells[60], which employs a fictitious material along the patch interface to approximately satisfy the kinematic constraints. This method was also used in[61] to couple trimmed NURBS patches. Other approaches include mortar methods[62,63,64], where adjacent patches are related by a master-slave relation and Lagrange multipliers are used to enforce continuity in a variational sense. The most accurate and stable development to date seems to be the Nitsche's method[65,66,67,68], where a mesh-dependent penalty (stabilization) term is used. These methods allow the coupling of non-conforming patches, however, they may lead to semi-definite saddle point problems in the case of mortar or Lagrange multiplier method. Nitsche's method also requires the determination of the stabilization parameters, which increases the overall cost.

A different way of ensuring that the desired continuity requirements are satisfied is to construct a basis for the approximation space, where the basis functions themselves have the required smoothness. Piecewise polynomial bases with C^1 continuity have been derived using symbolic algebra for planar bilinear patches in[69,70]. A similar approach was used to obtain C^2 bases for planar domains in[71,72]. These functions span a subspace of the C^0 or C^{-1} spline spaces defined on each patch individually and therefore result in a reduction in the number of degrees of freedom compared to the full (with less regularity) approximation space. A possible issue with this method was noted in[73], where it was shown that the over-constraining of a piecewise polynomial subspace of a given degree over certain geometries could result in a loss of approximation properties (C^1 locking). This has led to the study of analysis-suitable (AS) parameterizations[74], which include bilinear and bilinear-like planar mappings.

A related problem is the construction of smooth spline spaces over unstructured meshes, such as those obtained using T-Splines or even standard finite element mesh generators. Particular attention has been devoted to the parametrization around extraordinary vertices (interior vertices on quadrilateral meshes which have valence different than 4), where the desired smoothness properties have to be imposed through additional constraints. This leads to the "capping problem", where a ring of elements around the extraordinary vertex needs to be adjusted, for which various techniques have been proposed in[75,76,77]. In[78], smooth polar splines have been used at the extraordinary vertex, while a mathematical analysis of the dimension and basis construction on arbitrary topologies is presented in[79]. A construction using Hermite splines with optimal approximation properties is described in[80], while a more general method for smooth approximations over unstructured meshes is given in[81].

Another common approach for constructing smooth surfaces is through the use of subdivision surfaces, for which applications, in particular to thin-shell analysis have seen increased interest, see[82,83]. However, while subdivision surfaces provide a convenient way to construct smooth surfaces, they are not yet widely used for engineering applications. Alternatively, T-Splines have been used more extensively for modeling surface and plane geometries[6,8], using both unstructured and hierarchical meshes. Both the T-Spline and subdivision surface meshes require particular attention near the extraordinary vertices where the smoothness or approximation properties may be reduced when used in the analysis. Moreover, several approaches have been developed for discretizing the interior of the domain from a boundary triangulation or CAD surface geometry using T-Splines[84,85], or Bézier tetrahedra[86,87].

In this study, we follow a constructive approach for the approximation space, while leaving the geometric parametrization unchanged. We consider general geometries which are not limited to planar or bilinear mappings. Two approaches for constructing the C^1 basis on the multi-patch domain will be presented. In the first approach, the C^1 basis are given in terms of Bézier-Bernstein polynomials, whose Bézier ordinates are numerically computed based on the given geometry. This approach offers more degree of freedoms, therefore, it is suitable for handling complex geometry. However, it requires a high computational cost. In the second approach, we compute the C^1 basis functions as a linear combination of the C^0 basis functions corresponding to the common boundaries shared among the patches. An appealing advantage of this method compared to using Bézier elements is that the continuities within one patch are inherited by the resulting global basis functions automatically, whereas, in the first approach, this has to be done by providing additional continuity constraints in the linear system. Therefore, this approach is computationally efficient for finer meshes.

The problem of C^1 locking is overcome by localized degree elevation along the patch boundaries which restores the optimal convergence of the approximation in terms of degrees of freedom. To ensure that the basis functions have local support, a minimal determining set (MDS)[81] is computed. The method is also applied to non-planar surfaces, with the underlying assumption that the surfaces have G^1 (normal vector) continuity along the patch interfaces. This allows for the study of manifold-based 4th order PDEs on smooth surfaces as in[88,89,90].

5.2 Geometrically continuous isogeometric functions

We will start the discussion with several preliminary concept that are needed for C^1 coupling. For clarity, we discuss the relevant concept by using a simple example in 1D. The discussion focused on the functions of degree $p = 3$; however, a similar concept can be applied for degree $p = 2$ or higher.

In IGA, the p degree basis functions $N_{i,p}(\xi)$ are defined in the parameter space ($\hat{\Omega}$) which is formed by the knot vector $\Xi^{\xi} = \{\xi_0, \xi_1, \dots, \xi_n\}$. A B-spline segment is parameterized by a linear combination of basis functions with the corresponding control points $P_i \in \mathbb{R}$, $i = \{0, 1, \dots, n\}$ as follows:

$$G(\xi) = \sum_{i=0}^{n} N_{i,p}(\xi) P_i = x, \tag{5.1}$$

where $x \in \mathbb{R}$ is the image of ξ in the physical space.

Geometry objects with the desired shapes can be constructed in the physical space Ω by using the geometric mapping $G : \hat{\Omega} \to \Omega$ defined in (5.1). The isogeometric functions $\omega_{i,p}(x)$ in Ω can be obtained by composing the basis functions $N_{i,p}(\xi)$ with the inverse of the geometric mapping as follows:

$$\omega_{i,p}(x) = N_{i,p} \circ G^{-1}(x). \tag{5.2}$$

For the rest of this paper, unless the specification of the degree is needed, the subscript p will be suppressed for simplicity purposes.

We illustrate the concept of the C^1 coupling through an example in 1D. Consider two parameter spaces $\hat{\Omega}^{(1)}$ and $\hat{\Omega}^{(2)}$, each formed by the knot vector $\Xi^{\xi} = \{0,0,0,0,1,1,1,1\}$ and cubic basis functions. They define a parametrization by the geometric mappings $G^{(1)}$ and $G^{(2)}$ to the physical space $\Omega = (0,1)$ with the boundary $\Gamma = \{0,1\}$. The physical space is composed of two sub-domains $\Omega = \Omega^{(1)} \cup \Omega^{(2)}$, where each $\Omega^{(\ell)}$, $\ell \in \{1,2\}$, is assumed as one patch, and the superscript (ℓ) denotes the patch index. In Figure 5.1, $\Omega^{(1)}$ and $\Omega^{(2)}$ are represented by the red and blue lines respectively. We assume that the patches are joined at $\mathcal{P} = 0.5$ as is illustrated in Figure 5.1.

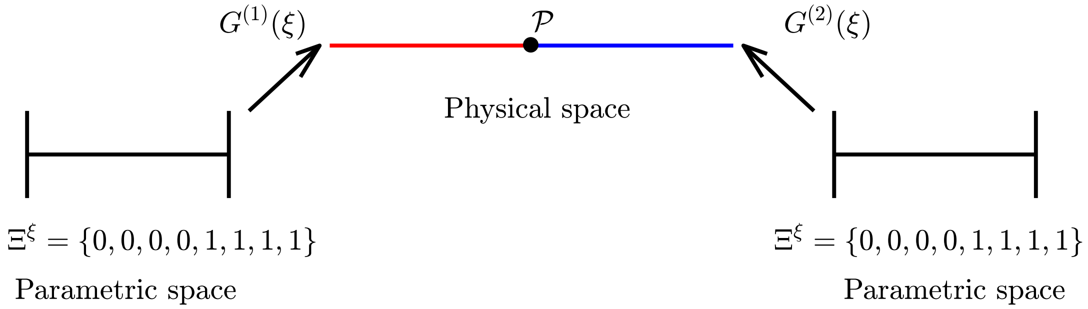


Figure 5.1: The geometric mapping $G^{(1)}$ and $G^{(2)}$ defined on $[0,1]$.

Suppose we seek the solution u that satisfies the biharmonic equation:

$$u^{(4)}(x) = f(x), \tag{5.3}$$

with the boundary condition $u : \Gamma \to \mathbb{R}$ given by:

$$\begin{aligned} u(0) = 0, u(1) = 0 \\ u'(0) = 0, u'(1) = 0. \end{aligned} \tag{5.4}$$

We choose $f = -24$ such that the exact solution of (5.3) subject to (5.4) is given by:

$$u(x) = -x^2(1-x)^2. \tag{5.5}$$

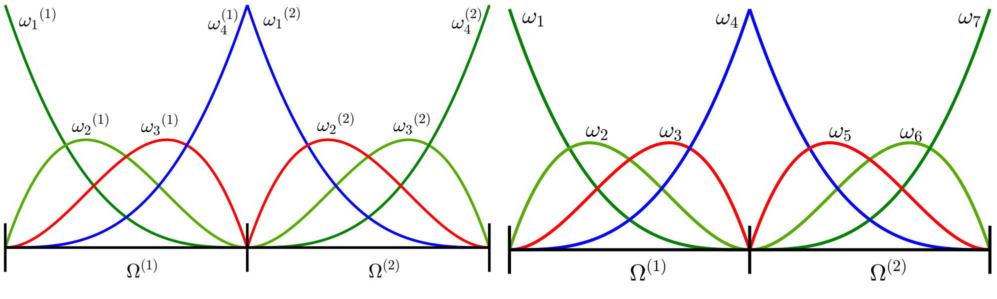
We want to compute an approximated solution u^h to u, of the form:

$$u_h(x) = \sum_{i=1}^{n} c_i \omega_i(x), \tag{5.6}$$

such that

$$\int_\Omega u_h'' \omega_i'' d\Omega = \int_\Omega u'' \omega_i'' d\Omega, \tag{5.7}$$

and u_h satisfies the boundary condition (5.4). In order to compute 5.7, ω_i should be at least C^1 continuous on Ω.



(a) (b)

Figure 5.2: Isogeometric functions on $\Omega = [0,1]$ (a) with local index, and (b) with global index.

Figure 5.2(a) shows the $\omega_i^{(\ell)}$ of each patch ℓ. The functions which have support at $\mathcal{P}$ are C^0 continuous if and only if

$$\omega_4^{(1)}(\mathcal{P}) = \omega_1^{(2)}(\mathcal{P}).$$

Thus, the two patches are connected by identifying the isogeometric functions which have support at the point $\mathcal{P}$. Connectivity between the patches can be indicated by converting the local index of the functions to global index as shown in Figure 5.2(b). To avoid confusion, the functions denoted by a global index will not have the superscript which denotes the patch index. We can observe that ω_3, ω_4, and ω_5 have only C^0 continuity (see the isogeometric functions highlighted with red and blue).

The goal of our study is eliminate the ω_i which possess C^0 continuous and replace them with globally C^1 continuous functions without changing the geometry parametrization. We classify the functions used in our construction accordingly into three groups:

- removed functions (functions which introduce C^0 continuity)
- new functions (functions which are globally C^1 continuous)

- original functions (functions remain unchanged)

Refer to Figure 5.2(b), ω_3 , ω_4 and ω_5 are the removed functions. The original functions are those preserved in the solution space (the green curves).

The main challenge is to construct the new functions $\psi_i(x)$ which are belong the space:

$$V = \{\psi_i \in C^1(\Omega) : \psi|_{\Omega^{(\ell)}} \in \hat{\Omega}^{(\ell)} \circ (G^{(\ell)})^{-1}, \ell = 1,2\}. \tag{5.8}$$

Suppose $\psi_i(x)$ can be separated into two parts:

$$\psi_i(x) = \psi_i^{(1)}(x) + \psi_i^{(2)}(x),$$

with each part having support on the corresponding patch. $\psi_i(x)$ is required to be C^1 across the common point $\mathcal{P}$, i.e. it needs to fulfill the following continuity condition:

$$\frac{d\psi_i^{(1)}(\mathcal{P})}{dx} = \frac{d\psi_i^{(2)}(\mathcal{P})}{dx}. \tag{5.9}$$

In this study, we present two different approaches to enforce C^1 continuity to $\psi_i(x)$, which are:

1. Bézier ordinates approach, and
2. Splines approach.

Each approach will be presented in the following subsections.

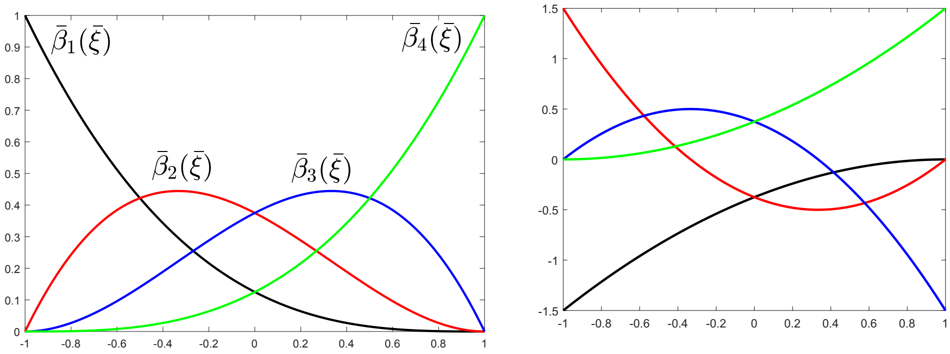
5.2.1 Bézier ordinates approach

Let $\hat{\psi}_i^{(\ell)}(\xi)$ be the corresponding function of $\psi_i^{(\ell)}(x)$ in the parametric space, it can be defined in terms of Bézier ordinates and Bernstein basis functions as follows:

$$\hat{\psi}_i^{(\ell)}(\xi) = \sum_{j=1}^{p+1} \varsigma_j^{(\ell)} \beta_j^{(\ell)}(\xi), \tag{5.10}$$

where $\beta_j^{(\ell)}(\xi)$ can be computed in a reference space $\bar{\Omega} = [-1, 1]$ (see Section 2.2). Figure 5.3(a) shows the cubic Bernstein functions $\bar{\beta}_j^{(\ell)}(\bar{\xi})$ plotted in the reference space $\bar{\Omega}$, their derivatives are given in Figure 5.3(b).

It is shown in the plotting that at $\bar{\xi} = -1$, the derivatives of $\bar{\beta}_3^{(\ell)}(\bar{\xi})$ and $\bar{\beta}_4^{(\ell)}(\bar{\xi})$ are zero. Thus, they have no influence on the derivatives of $\hat{\psi}_i^{(\ell)}(\xi)$ at ξ_i. In contrast, the derivatives of $\bar{\beta}_1^{(\ell)}(\bar{\xi})$ and $\bar{\beta}_2^{(\ell)}(\bar{\xi})$ are non-zero at $\bar{\xi} = -1$; which mean that they contribute to the derivatives of $\hat{\psi}_i^{(\ell)}(\xi)$ at ξ_i. We known that $\bar{\beta}_j^{(\ell)}(\bar{\xi})$ are scaled by the Bézier ordinates ς_j. Therefore, the associated Bézier coefficient of $\bar{\beta}_1^{(\ell)}(\bar{\xi})$ and $\bar{\beta}_2^{(\ell)}(\bar{\xi})$ (ς_1 and ς_2) will have influence on the derivatives at that end point. A similar concept is used to determine ς_j which are affecting the derivatives at the other end point ($\bar{\xi} = 1$). In this study, we address the influencing ς_j as "*relevant coefficients*".



(a) (b)

Figure 5.3: 1D Bernstein basis functions and their first derivatives

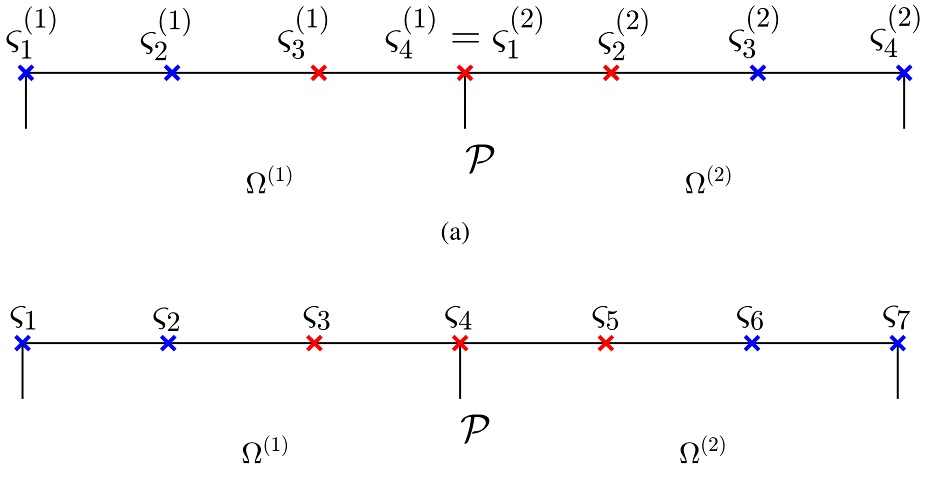
In Figure 5.4(a), the crosses represent the coefficients of $\bar{\beta}_j(\bar{\xi})$ on the two-patch domain $\Omega^{(1)}$ and $\Omega^{(2)}$. Since the patches are C^0 continuous at $\mathcal{P}$, the Bézier ordinates $\varsigma_4^{(1)} = \varsigma_1^{(2)}$ are equal at $\mathcal{P}$. The Bézier coefficients with global index are shown in Figure 5.4(b). In the figures, the relevant coefficients are marked with red; we need to determine the suitable values for them such that obtain C^1 continuous functions. The value of the other coefficients (denoted by the blue dots) will be set as zero.

Suppose (5.9) can be rewrite as:

$$\frac{d\hat{\psi}_i^{(1)}}{dx} \circ G^{(1)^{-1}}(\mathcal{P}) = \frac{d\hat{\psi}_i^{(2)}}{dx} \circ G^{(2)^{-1}}(\mathcal{P}), \tag{5.11}$$

we set the Bézier ordinates which has no influence at $\mathcal{P}$ to be zero ($\varsigma_0 = 0$, $\varsigma_1 = 0$, and $\varsigma_4 = 0$), and obtain the following:

$$\varsigma_3 \frac{d\beta_3^{(1)}}{dx} \circ G^{(1)^{-1}}(\mathcal{P}) + \varsigma_4 \left(\frac{d\beta_4^{(1)}}{dx} \circ G^{(1)^{-1}}(\mathcal{P}) - \frac{d\beta_1^{(2)}}{dx} \circ G^{(2)^{-1}}(\mathcal{P}) \right) - \varsigma_5 \frac{d\beta_2^{(2)}}{dx} \circ G^{(2)^{-1}}(\mathcal{P}) = 0. \tag{5.12}$$



(b)

Figure 5.4: Bézier coefficient in $\Omega^{(1)}$ and $\Omega^{(2)}$ (a) with local index and (b) with global index.

(5.12) can be represent in the matrix form as:

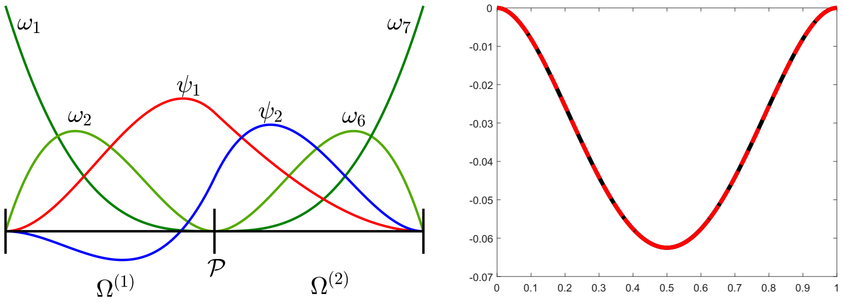
$$M\varsigma = 0,$$

where $M = \left[\frac{d\beta_3^{(1)}}{dx} \circ G^{(1)^{-1}}(\mathcal{P}) \quad \frac{d\beta_4^{(1)}}{dx} \circ G^{(1)^{-1}}(\mathcal{P}) - \frac{d\beta_1^{(2)}}{dx} \circ G^{(2)^{-1}}(\mathcal{P}) \quad -\frac{d\beta_2^{(2)}}{dx} \circ G^{(2)^{-1}}(\mathcal{P})\right]$, and $\varsigma = \begin{bmatrix}\varsigma_2 & \varsigma_3 & \varsigma_4\end{bmatrix}^T$. The values of ς is therefore given by the basis of the null space of M.

For this example, $M = [-9, 18, -9]$, and

$$\text{null}(M) = \begin{bmatrix} 0.816496580927726 & -0.408248290463863 \\ 0.526598632371090 & 0.236700683814455 \\ 0.236700683814455 & 0.881649658092773 \end{bmatrix}.$$

There are two basis vectors in the nullspace, each of them is used to construct a new function. In the new solution space, we replace the removed functions with the new functions as shown in Figure 5.5(a). In the figure, the green curves are the original functions, while the red and blue curves are the new functions (see Figure 5.2 for the old solution spaces). For coarse mesh where there is only one element in each patch, the relative error is given by 0.06205. The error can be further reduced through refinement. The comparison of the exact solution and approximating solution computed with 16 elements in each patch is given in Figure 5.5(b). In the figure, the dark line represents the approximating solution, while the red dashes denote the exact solution. The overlapping of red dashes on the dark line indicates that we obtain a good approximation, with the relative error given by 9.4682×10^{-7}.

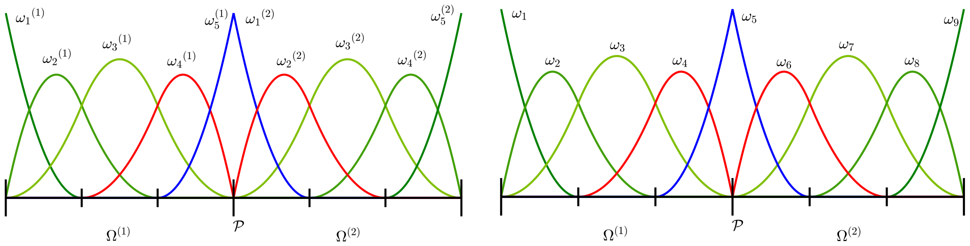


(a) (b)

Figure 5.5: (a) Smooth solution space obtained from C^1 coupling and (b) comparison of solutions.

5.2.2 Spline approach

In this section, we describe the spline approach by using a finer mesh, where each patch is formed by three elements. The isogometric functions before C^1 coupling are shown in Figures 5.6(a) and 5.6(b), labeled with local and global index respectively. We can observe that the functions are smooth and continuous across the elements within one patch. However, ω_4, ω_5, and ω_6, have only C^0 continuity (see the isogometric functions highlighted with red and blue). Hence, they are the removed functions.



(a) (b)

Figure 5.6: (a) C^0 isogeometric functions on a finer mesh (a) local index and (b) global index.

For the spline approach, we define the new functions $\psi_i \in V$ as a linear combination of the removed functions. Therefore, $\psi_i(x)$ are defined as :

$$\psi_i(x) = c_1^i \omega_4(x) + c_2^i \omega_5(x) + c_3^i \omega_6(x). \tag{5.13}$$

where c_j^i, $j = 1,2,3$ are scalar coefficients. We know that $\psi_i(x)$ can be expressed as sum of $\psi_i^{(\ell)}(x)$ which has local support on patch ℓ as:

$$\psi_i(x) = \psi_i^{(1)}(x) + \psi_i^{(2)}(x).$$

For this example, the removed functions with global and local index are related as following (see Figure 5.6):

$$\omega_4(x) = \omega_4^{(1)}(x),\, \omega_5(x) = \omega_5^{(1)}(x) + \omega_1^{(2)}(x), \text{and}\, \omega_6(x) = \omega_2^{(2)}(x).$$

Therefore, $\psi_i^{(1)}(x)$ and $\psi_i^{(2)}(x)$ can be defined as:

$$\psi_i^{(1)}(x) = c_1^i \omega_4^{(1)}(x) + c_2^i \omega_5^{(1)}(x),$$

and

$$\psi_i^{(2)}(x) = c_2^i \omega_1^{(2)}(x) + c_3^i \omega_2^{(2)}(x).$$

Hence (5.9) can be rewritten as:

$$c_1^i \frac{d\,\omega_4^{(1)}(\mathcal{P})}{dx} + c_2^i \frac{d\,\omega_5^{(1)}(\mathcal{P})}{dx} = c_2^i \frac{d\,\omega_1^{(2)}(\mathcal{P})}{dx} + c_3^i \frac{d\,\omega_2^{(2)}(\mathcal{P})}{dx} \tag{5.14}$$

Let $M\mathbf{c}^i = 0$ be the matrix form of (5.14), where

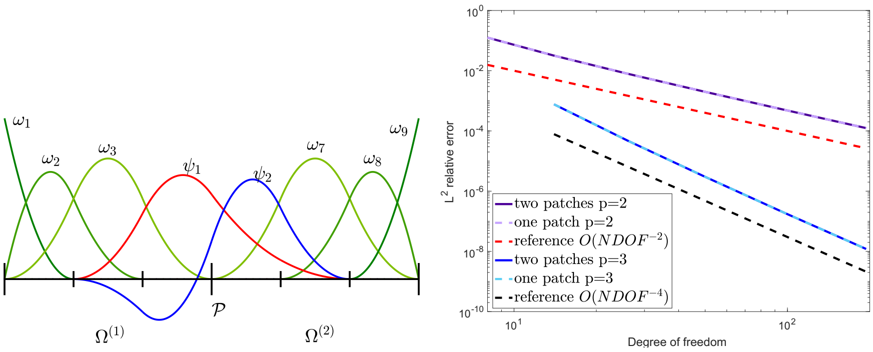
$$M = \begin{bmatrix} \frac{d\,\omega_4^{(1)}(\mathcal{P})}{d\xi} & \frac{d\,\omega_5^{(1)}(\mathcal{P})}{d\xi} - \frac{d\,\omega_1^{(2)}(\mathcal{P})}{d\xi} & -\frac{d\,\omega_2^{(2)}(\mathcal{P})}{d\xi} \end{bmatrix}$$

and $\mathbf{c}^i = \begin{bmatrix} c_1^i & c_2^i & c_3^i \end{bmatrix}^T$. We need to determine the values of $\mathbf{c}^i$ for construction of ψ_i, where $\mathbf{c}^i$ are given by the basis of the null space of M.

For the this example, $M = \begin{bmatrix} -2 & 4 & -2 \end{bmatrix}$, and

$$\text{null}(M) = \begin{bmatrix} 0.816496580927726 & -0.408248290463863 \\ 0.526598632371090 & 0.236700683814455 \\ 0.236700683814455 & 0.881649658092773 \end{bmatrix}.$$

Therefore, there are two new functions. Figure 5.7(a) shows the new cubic functions (blue and red lines) which are globally C^1 continuous. Together with the original functions (plotted in green), they form the smooth solution space for the two patch domain. Comparison of the convergence rate between one patch and two patches are given in Figure 5.7(b). In the figure, the purple line and light purple dashes are the convergence rates when the solution space is formed by quadratic basis functions, while the blue line and light blue dashes represent the convergence rate when cubic basis functions form the solution space. The reference convergence plots for $p = 2$ and $p = 3$ are represented by the red and dark dashes respectively. We can observe from the figure that the results are converging at the optimal rate for both $p = 2$ and $p = 3$. Meanwhile, we obtain zero error (up to machine precision) when the solution space is constructed with quartic basic functions ($p = 4$). For 20 degrees of freedom, the relative error is given by $7.3089.10^{-15}$ for two patch domain and $2.6917.10^{-14}$ for one patch domain.



(a) (b)

Figure 5.7: (a) Solution space formed by original functions and new functions (b) convergence plot.

Remarks

Note that for both approaches, the new functions highlighted in blue are negative. Typically, negativity will not affect the analysis result and the geometry is still represented by a non-negative basis. However, if the shape needs to be altered, new C^1 functions have to be recomputed based on the new geometry.

5.3 Smooth isogeometric functions in 2D

All the concepts of C^1 coupling in 1D are readily be extended to 2D, 3D, and shells. The discussion in this section will be focused on 2D. An effectively way to imposing the continuity conditions through the graph surface of the isogeometric functions will be discussed. Besides the computation methods, we will also discuss the advantages of each approach.

For all the discussion in 2D, $N_i(\xi, \eta)$ and $\beta_i(\xi, \eta)$ will be considered as the tensor product of 1D basis functions, and tensor product of 1D Bernstein functions respectively.

5.3.1 C^1 continuity condition in 2D

Suppose two parametric spaces are mapped to the physical space subdomains $\Omega^{(1)}$ and $\Omega^{(2)}$ by the corresponding geometric mapping $G^{(\ell)} : \hat{\Omega} \to \Omega^{(\ell)}$, $\ell \in \{1,2\}$. In the physical space, the patches are attached at a common edge $\tau = G(1,\eta) = G(0,\eta)$ (red line) as shown in Figure 5.8.

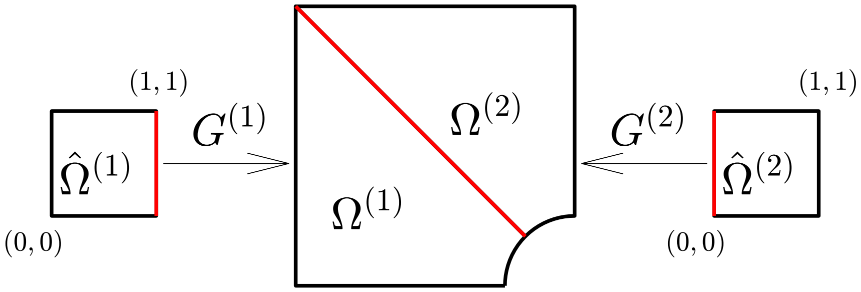


Figure 5.8: Mapping form parameter space to physical space.

The geometric mapping in 2D is defined as:

$$G^{(\ell)}(\xi,\eta) = \sum_{i=1}^{n_N} N_i(\xi,\eta) P_i^{(\ell)} = (x,y), \tag{5.15}$$

where $P_i^{(\ell)} \in \mathbb{R}^2$ are the associate control point of the basis functions $N_i(\xi,\eta)$, and n_N is the number of basis functions in each element. Meanwhile, the isogeometric functions on patch ℓ in the physical space are given by:

$$\omega_i^{(\ell)}(x,y) = N_i^{(\ell)} \circ {G^{(\ell)}}^{-1}(x,y),\, (x,y) \in \Omega^{(\ell)}.$$

Suppose the new basis functions can be written as:

$$\psi(x,y) = \psi^{(1)}(x,y) + \psi^{(2)}(x,y),$$

where $\psi^{(\ell)}$ has support on patch ℓ. In order to achieve C^1 continuity along τ, the derivatives of the functions $\psi^{(1)}$ and $\psi^{(2)}$ have to be equal, i.e.

$$\nabla \psi^{(1)}(x,y) = \nabla \psi^{(2)}(x,y), \quad \forall (x,y) \in \tau. \tag{5.16}$$

The continuity condition can be imposed through the graph surface of $\psi^{(\ell)}(x,y)$. We refer to the following results from[91]:

Proposition 5.3.1. *A necessary and sufficient condition for ψ to be contained in V is that their graph surfaces have G^1 continuity, or equivalently, all the points along the common boundary of the corresponding graph surfaces have the same tangent planes.*

In the following, we will therefore compute $\psi(x,y)$ whose graph surfaces are G^1 continuous across τ. Let the $\hat{\psi}^{(\ell)}(\xi,\eta)$ be corresponding functions of $\psi^{(\ell)}(x,y)$ in the parameter space. The graph surface of $\psi^{(\ell)}$ can be written as:

$$F^{(\ell)}(\xi,\eta) = (G^{(\ell)}(\xi,\eta), \hat{\psi}^{(\ell)}(\xi,\eta))^T. \tag{5.17}$$

We note that the tangent plane at a point (ξ,η) of the graph surface is spanned by the first derivatives: $\frac{\partial F^{(\ell)}(\xi,\eta)}{\partial \xi}$ and $\frac{\partial F^{(\ell)}(\xi,\eta)}{\partial \eta}$.

In order to achieve G^1 continuity across τ, for all the points along $\tau = G^{(1)}(1,\eta) = G^{(2)}(0,\eta)$, the tangent planes from $F^{(1)}(1,\eta)$ and $F^{(2)}(0,\eta)$ have to be coplanar. This leads to the following continuity condition:

$$\det\left(\frac{\partial F^{(1)}(1,\eta)}{\partial \xi}, \frac{\partial F^{(1)}(1,\eta)}{\partial \eta}, \frac{\partial F^{(2)}(0,\eta)}{\partial \xi}\right) = 0. \tag{5.18}$$

In the above continuity condition, $\frac{\partial F^{(2)}(0,\eta)}{\partial \eta}$ is neglected because it is identical to $\frac{\partial F^{(1)}(1,\eta)}{\partial \eta}$. Equation (5.18) contains a 3×3 matrix, by applying the cofactor expansion to it, we obtain:

$$\alpha(1,\eta)\frac{\partial \hat{\psi}^{(1)}(1,\eta)}{\partial \xi} + \beta(1,\eta)\frac{\partial \hat{\psi}^{(1)}(1,\eta)}{\partial \eta} + \gamma(0,\eta)\frac{\partial \hat{\psi}^{(2)}(0,\eta)}{\partial \xi} = 0, \tag{5.19}$$

where

$$\alpha(1,\eta) = \det\begin{bmatrix} \frac{\partial G_1^{(1)}(1,\eta)}{\partial \eta} & \frac{\partial G_1^{(2)}(0,\eta)}{\partial \xi} \\ \frac{\partial G_2^{(1)}(1,\eta)}{\partial \eta} & \frac{\partial G_2^{(2)}(0,\eta)}{\partial \xi} \end{bmatrix},$$

$$\beta(1,\eta) = -\det\begin{bmatrix} \frac{\partial G_1^{(1)}(1,\eta)}{\partial \xi} & \frac{\partial G_1^{(2)}(0,\eta)}{\partial \xi} \\ \frac{\partial G_2^{(1)}(1,\eta)}{\partial \xi} & \frac{\partial G_2^{(2)}(0,\eta)}{\partial \xi} \end{bmatrix},$$

and

$$\gamma(0,\eta) = \det\begin{bmatrix} \frac{\partial G_1^{(1)}(1,\eta)}{\partial \xi} & \frac{\partial G_1^{(1)}(1,\eta)}{\partial \eta} \\ \frac{\partial G_2^{(1)}(1,\eta)}{\partial \xi} & \frac{\partial G_2^{(1)}(1,\eta)}{\partial \eta} \end{bmatrix}.$$

The condition given in (5.19) is used to determine the continuity constraints such that enforce G^1 continuity between $F^{(1)}(\xi,\eta)$ and $F^{(2)}(\xi,\eta)$. More details will be presented in the following sections.

5.3.2 C^1 coupling in 2D

In this sections, we present construction of new functions ($\psi_i(x,y)$) in 2D. We begin by considering the simple case where each patch in Figure 5.8 contains only one element. The basis functions of $N_i^{(\ell)}$ on each patch in the parameter space are denoted by blue dots in Figure 5.9. For the edge parallel to the η direction, the basis functions $N_i{}^{(\ell)}$ labeled on the column immediately next to the edge have support on it, while the edge parallel to the ξ direction is contained in support of the $N_i{}^{(\ell)}$ labeled on the row next to it.

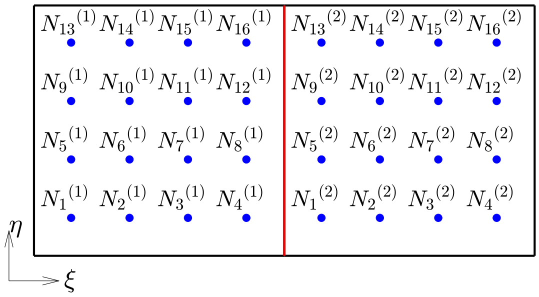

$N_{13}^{(1)}$ $N_{14}^{(1)}$ $N_{15}^{(1)}$ $N_{16}^{(1)}$
$N_9^{(1)}$ $N_{10}^{(1)}$ $N_{11}^{(1)}$ $N_{12}^{(1)}$
$N_5^{(1)}$ $N_6^{(1)}$ $N_7^{(1)}$ $N_8^{(1)}$
$N_1^{(1)}$ $N_2^{(1)}$ $N_3^{(1)}$ $N_4^{(1)}$
$N_{13}^{(2)}$ $N_{14}^{(2)}$ $N_{15}^{(2)}$ $N_{16}^{(2)}$
$N_9^{(2)}$ $N_{10}^{(2)}$ $N_{11}^{(2)}$ $N_{12}^{(2)}$
$N_5^{(2)}$ $N_6^{(2)}$ $N_7^{(2)}$ $N_8^{(2)}$
$N_1^{(2)}$ $N_2^{(2)}$ $N_3^{(2)}$ $N_4^{(2)}$
η
ξ

Figure 5.9: Basis functions in parameter space with local index (C^{-1} continuity).

Recall that functions are C^0 across τ if and only if $N_i^{(1)}(1,\eta) = N_i^{(2)}(0,\eta)$. Thus, the two patches share the functions which support τ. The functions labeled by their global index i are shown in the Figure 5.10, where the functions with red label are shared by the two patches.

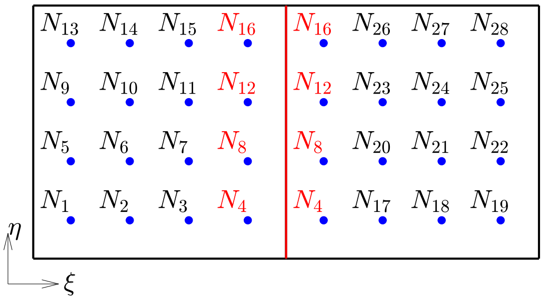

N_{13} N_{14} N_{15} N_{16} N_{16} N_{26} N_{27} N_{28}
N_9 N_{10} N_{11} N_{12} N_{12} N_{23} N_{24} N_{25}
N_5 N_6 N_7 N_8 N_8 N_{20} N_{21} N_{22}
N_1 N_2 N_3 N_4 N_4 N_{17} N_{18} N_{19}
η
ξ

Figure 5.10: Basis functions in the parameter space with global index (C^0 continuity).

The C^1 coupling procedure starts with determining the removed functions and the original functions. For simplicity, we use the symbols $\hat{\mathcal{R}}$ and $\mathcal{R}$ to denote the remove functions in the parameter space and physical space respectively. The removed functions are marked with red dots and labeled accordingly in Figure 5.11. Though $\hat{\mathcal{R}}_1$, $\hat{\mathcal{R}}_4$, $\hat{\mathcal{R}}_7$ and $\hat{\mathcal{R}}_{10}$ does not have support on τ, they meet with $\hat{\mathcal{R}}_3$, $\hat{\mathcal{R}}_6$, $\hat{\mathcal{R}}_9$ and $\hat{\mathcal{R}}_{12}$ of the neighbor patch with C^0 continuous at τ. Therefore, these functions are also classified as the removed functions. The other functions belong to the group of original functions.

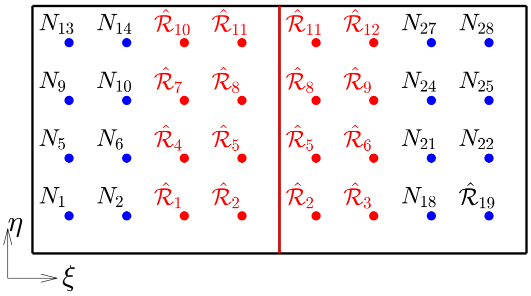


Figure 5.11: Removed functions and original functions in the parameter space.

5.3.2.1 Bézier ordinate approach

Recall that our task is to obtain new basis functions $\psi_i^{(\ell)}(x,y)$ which their graph surface $F_i^{(\ell)}(\xi,\eta)$ are G^1 continuous at the common boundary (i.e. fulfill the continuity condition (5.18)). Suppose $\hat{\psi}_i^{(\ell)}(\xi,\eta)$ is the corresponding function of $\psi_i^{(\ell)}(x,y)$ in the parameter space. For the Bézier ordinate approach, we define $\hat{\psi}_i^{(\ell)}(\xi,\eta)$ in the Bernstein Bézier form as:

$$\hat{\psi}_i^{(\ell)}(\xi,\eta) = \sum_{\iota=1}^{n_\beta} \varsigma_\iota^{(\ell)} \beta_\iota(\xi,\eta), \tag{5.20}$$

where n_β is the number of Bernstein functions, and $\varsigma_\iota^{(\ell)}$ are the coefficients associated to $\beta_\iota(\xi,\eta)$. Therefore, the graph surface of $\psi_i^{(\ell)}(x,y)$ can be express in the terms of Bézier ordinates as:

$$F_i^{(\ell)}(\xi,\eta) = \Big(G^{(\ell)}(\xi,\eta), \sum_{\iota=1}^{n_\beta} \varsigma_\iota^{(\ell)} \beta_\iota(\xi,\eta)\Big)^T. \tag{5.21}$$

We need to select the relevant coefficient, where the concept is similar to those discussed in 1D. If a particular edge is part of the common boundary, the influencing coefficients of this edge are the relevant coefficients. Figure 5.12 illustrate the coefficients in the two patch domain; the crosses represent the coefficients. The relevant coefficients are marked with red. Meanwhile, the remaining coefficients are represented by the blue crosses.

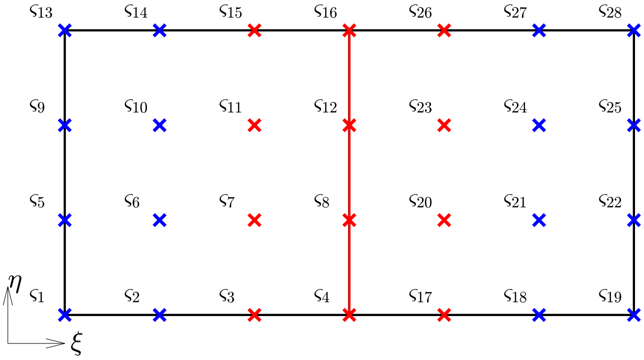


Figure 5.12: Bézier coefficients on the two patch domain.

The procedure is followed by selecting uniformly distributed collocation points along the common boundary. Each point is substitute into the continuity conditions (5.18), where $F_i^{(\ell)}(\xi,\eta)$ is given by (5.21). For each point, we will obtain a continuity constraint, which is then used to form a row in the homogeneous linear system:

$$M\varsigma^i = 0, \quad \varsigma^i = (\varsigma_\iota)_{\iota=1,\ldots,n_\varsigma},$$

where n_ς denotes the number of relevant coefficients. We then seek for the nullspace of M. Each basis vector in the nullspace contains the values of the relevant coefficients ς^i of a C^1 continuous isogeometric function $(\psi_i(x,y))$. Note that sufficient collocation points are needed to avoid underdetermined system which would result in an incorrectly large nullspace.

5.3.2.2 Spline approach

For the spline approach, we define the new functions as a linear combination of the removed functions:

$$\psi_i(x,y) = \sum_{\iota=1}^{n_{\mathcal{R}}} c_\iota \mathcal{R}_\iota(x,y), \tag{5.22}$$

where $n_{\mathcal{R}}$ denote the number of removed functions and c_ι are the corresponding scalar coefficients. We known that the new function $\psi_i(x,y)$ can be express in term of $\psi_i^{(\ell)}(x,y)$, and we would like to impose the continuity condition through the graph surface of $\psi_i^{(\ell)}(x,y)$ which is given by:

$$F_i^{(\ell)}(\xi,\eta) = \left(G^{(\ell)}(\xi,\eta), \hat{\psi}_i^{(\ell)}(\xi,\eta)\right)^T, \tag{5.23}$$

where

$$\hat{\psi}_i^{(\ell)}(\xi,\eta) = \sum_{\iota\in I_\ell} c_\iota \hat{\mathcal{R}}_\iota^{(\ell)}(\xi,\eta) \tag{5.24}$$

is the corresponding functions of $\psi_i^{(\ell)}(x,y)$ in the parametric space. In (5.24), I_ℓ is the set of remove functions index, and $\hat{\mathcal{R}}_i^{(\ell)}(\xi,\eta)$ are the removed functions in patch ℓ.

Similar with the Bézier ordinate approach, uniformly distributed points along τ are taken as collocation points, and each of them is substituted into (5.18) to obtain a constraint. The set of constraints can be represented as a homogeneous linear system:

$$Mc = 0,\, c = (c_i)_{i=\{1,2,\dots,n_{\mathcal{R}}\}}. \tag{5.25}$$

Therefore, any basis of the null space of M defines a set of values for **c** which combined with the corresponding $\mathcal{R}_i$ yield $\psi_i(x,y) \in V$.

5.3.2.3 Remarks on continuity condition

Be aware that the continuity condition given in (5.18) holds for the case when τ is parallel to the η-direction. If the common edge is parallel to ξ-direction, i.e. $\tau = G^{(1)}(\xi,0) = G^{(2)}(\xi,1)$, the condition has to be changed accordingly as:

$$\det\left(\frac{\partial F^{(1)}(\xi,1)}{\partial \xi}, \frac{\partial F^{(1)}(\xi,1)}{\partial \eta}, \frac{\partial F^{(2)}(0,\xi)}{\partial \eta}\right) = 0. \tag{5.26}$$

The cofactor expansion of (5.26) is given by:

$$\alpha(\xi,1)\frac{\partial \varphi^{(1)}(\xi,1)}{\partial \xi} + \beta(\xi,1)\frac{\partial \varphi^{(1)}(\xi,1)}{\partial \eta} + \gamma(\xi,0)\frac{\partial \varphi^{(2)}(\xi,0)}{\partial \eta} = 0, \tag{5.27}$$

where

$$\alpha(\xi,1) = \det\begin{bmatrix} \frac{\partial G_1^{(1)}(\xi,1)}{\partial \eta} & \frac{\partial G_1^{(2)}(\xi,0)}{\partial \eta} \\ \frac{\partial G_2^{(1)}(\xi,1)}{\partial \eta} & \frac{\partial G_2^{(2)}(\xi,0)}{\partial \eta} \end{bmatrix},$$

$$\beta(\xi,1) = -\det\begin{bmatrix} \frac{\partial G_1^{(1)}(\xi,1)}{\partial \xi} & \frac{\partial G_1^{(2)}(\xi,0)}{\partial \eta} \\ \frac{\partial G_2^{(1)}(\xi,1)}{\partial \xi} & \frac{\partial G_2^{(2)}(\xi,0)}{\partial \eta} \end{bmatrix},$$

and

$$\gamma(\xi,0) = \det\begin{bmatrix} \frac{\partial G_1^{(1)}(\xi,1)}{\partial \xi} & \frac{\partial G_1^{(1)}(\xi,1)}{\partial \eta} \\ \frac{\partial G_2^{(1)}(\xi,1)}{\partial \xi} & \frac{\partial G_2^{(1)}(\xi,1)}{\partial \eta} \end{bmatrix}.$$

5.3.3 C^1 coupling of finer meshes

The coupling methods can be generalized to finer meshes directly. In this section, we will point out the important aspects of the Bézier ordinates and Spline approaches when dealing with finer mesh. In addition, we will discuss the different advantages of both approaches.

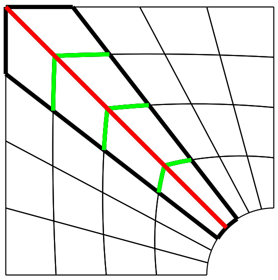
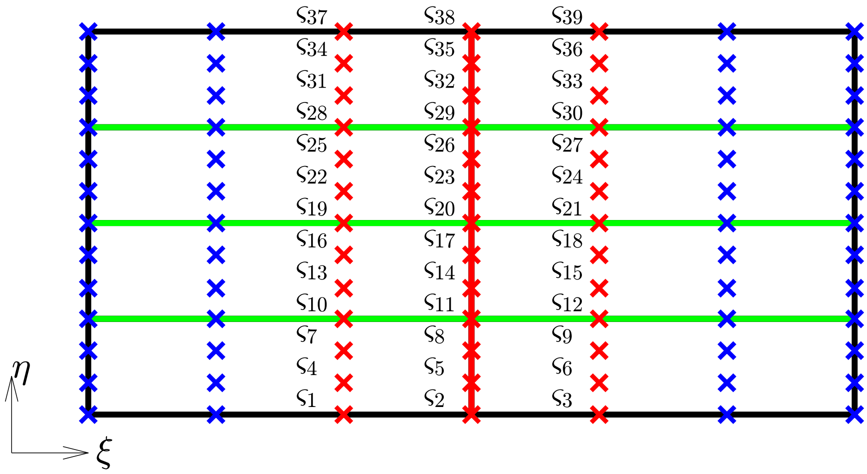


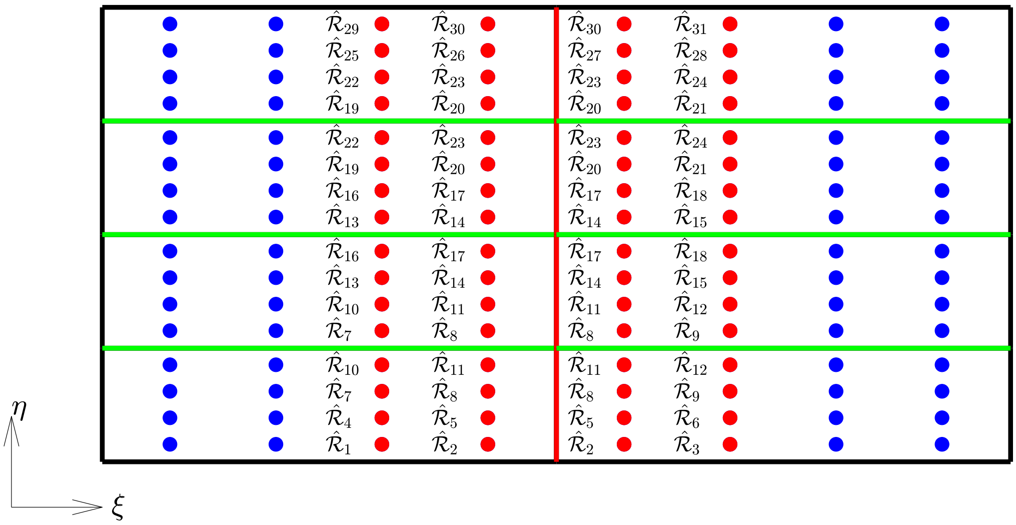
Figure 5.13: Plate with circular hole formed by finer mesh.

Suppose each patch in Figure 5.8 is formed by multiple elements as shown in the Figure 5.13. In the figure, the red line denotes the common boundary, and bold dark lines surround the element associated with it. The coupling task is to join these elements such that they are C^1 continuous across the common boundary. The relevant coefficients and the removed functions involve in the C^1 coupling can be determined using the similar concepts described in the previous section. The red crosses ans dots in Figures 5.14(a) and 5.15(b) represent the relevant coefficients and removed functions respectively. They are labeled accordingly in the figures.



(a)

Figure 5.14: Plate with circular hole example (a) relevant coefficients.


$\hat{\mathcal{R}}_{29}$ $\hat{\mathcal{R}}_{30}$ $\hat{\mathcal{R}}_{30}$ $\hat{\mathcal{R}}_{31}$
$\hat{\mathcal{R}}_{25}$ $\hat{\mathcal{R}}_{26}$ $\hat{\mathcal{R}}_{27}$ $\hat{\mathcal{R}}_{28}$
$\hat{\mathcal{R}}_{22}$ $\hat{\mathcal{R}}_{23}$ $\hat{\mathcal{R}}_{23}$ $\hat{\mathcal{R}}_{24}$
$\hat{\mathcal{R}}_{19}$ $\hat{\mathcal{R}}_{20}$ $\hat{\mathcal{R}}_{20}$ $\hat{\mathcal{R}}_{21}$
$\hat{\mathcal{R}}_{22}$ $\hat{\mathcal{R}}_{23}$ $\hat{\mathcal{R}}_{23}$ $\hat{\mathcal{R}}_{24}$
$\hat{\mathcal{R}}_{19}$ $\hat{\mathcal{R}}_{20}$ $\hat{\mathcal{R}}_{20}$ $\hat{\mathcal{R}}_{21}$
$\hat{\mathcal{R}}_{16}$ $\hat{\mathcal{R}}_{17}$ $\hat{\mathcal{R}}_{17}$ $\hat{\mathcal{R}}_{18}$
$\hat{\mathcal{R}}_{13}$ $\hat{\mathcal{R}}_{14}$ $\hat{\mathcal{R}}_{14}$ $\hat{\mathcal{R}}_{15}$
$\hat{\mathcal{R}}_{16}$ $\hat{\mathcal{R}}_{17}$ $\hat{\mathcal{R}}_{17}$ $\hat{\mathcal{R}}_{18}$
$\hat{\mathcal{R}}_{13}$ $\hat{\mathcal{R}}_{14}$ $\hat{\mathcal{R}}_{14}$ $\hat{\mathcal{R}}_{15}$
$\hat{\mathcal{R}}_{10}$ $\hat{\mathcal{R}}_{11}$ $\hat{\mathcal{R}}_{11}$ $\hat{\mathcal{R}}_{12}$
$\hat{\mathcal{R}}_{7}$ $\hat{\mathcal{R}}_{8}$ $\hat{\mathcal{R}}_{8}$ $\hat{\mathcal{R}}_{9}$
$\hat{\mathcal{R}}_{10}$ $\hat{\mathcal{R}}_{11}$ $\hat{\mathcal{R}}_{11}$ $\hat{\mathcal{R}}_{12}$
$\hat{\mathcal{R}}_{7}$ $\hat{\mathcal{R}}_{8}$ $\hat{\mathcal{R}}_{8}$ $\hat{\mathcal{R}}_{9}$
$\hat{\mathcal{R}}_{4}$ $\hat{\mathcal{R}}_{5}$ $\hat{\mathcal{R}}_{5}$ $\hat{\mathcal{R}}_{6}$
$\hat{\mathcal{R}}_{1}$ $\hat{\mathcal{R}}_{2}$ $\hat{\mathcal{R}}_{2}$ $\hat{\mathcal{R}}_{3}$
η
ξ

(b)

Figure 5.14: Plate with circular hole example (Cont.) (b) removed functions.

For the Bézier ordinates approach, all the edges shared by two elements and connected to the common boundary have to be taken into account when imposing the continuity constraints. This is because the relevant coefficients would influence the derivatives at these edges too. In this study, we call these edges as *relevant edges*. For the example given in Figure 5.13, there are a total of 10 relevant edges. Four of them are represented by the red lines which form the common boundary, and the green lines denote the other six. The continuity conditions have to be applied simultaneously at all these edges to obtain the new functions $\psi_i(x,y) \in V$.

Recall that for the spline approach, we defined the new function as a linear combination of the removed functions. The advantage of this approach is that the continuity between elements within the same patch will be inherited in the new functions. Therefore, we only have to impose C^1 continuity condition along the common boundary (the four red edges in Figure 5.13). Thus, the coupling procedure is much more simplified than the Bézier ordinates approach.

Note that we limit the entire solution space to be C^1 continuous. For 2D, the adjacent elements in the same patch will share two rows or columns of basis functions at the common edge (see Figure 5.15(b)). As a result, the spline approach will have fewer unknown parameters and have a lower computation cost. Though the Bézier ordinate approach has to deal with more unknown parameters and requires higher computation cost, it has more DOFs as a return.

As discussed above, both coupling approaches have their own advantages. Selection of a suitable approach can be made is based on the geometric model. For a higher dimension with simple geometry, the spline approach which yields a smaller homogenous linear system will be a good option. Meanwhile, for the domain with complex geometry, the Bézier ordinate approach which gives more DOFs will be the proper choice.

5.3.3.1 Remarks on collocation points

In order to avoid the underdetermined system, for the Bézier ordinate approach which involves several edges, the selection of uniformly distributed points has to fulfill the following condition:

$$\sum_{l=1}^{n_e} |\rho_l| \geq n_{\varsigma}. \tag{5.28}$$

In (5.28), n_e and n_{ς} represent the number of relevant edges and number of coefficients respectively. ρ_l is the set of uniformly distributed points on the l^{th} edge, while $|\rho_l|$ denotes the cardinality of ρ_l.

For the spline approach, condition (5.28) will be changed such that the number of collocation points is determined based on the number of removed functions:

$$\sum_{l=1}^{n_e} |\rho_l| \geq n_{\mathcal{R}}. \tag{5.29}$$

Here, n_e represents the number of edges on the common boundary, $|\rho_j|$ is the cardinality of the set of uniformly distributed points on the j^{th} edge, and $n_{\mathcal{R}}$ denotes the number of removed functions.

5.4 Multi-patch coupling

The main concepts for multi-patch coupling are similar to two patch coupling. Both Bézier ordinates and spline approaches are compatible for multi-patch C^1 coupling involving extraordinary point(s). Here, we will give the general algorithms of each approach, as well as show some of the example involving extraordinary vertices.

For multi-patch coupling which involves n_ℓ patches, we need to construct the new functions:

$$\psi_i(x,y) = \sum_{i=1}^{n_\ell} \psi_i^{(\ell)}(x,y), \tag{5.30}$$

such that $\psi_i(x,y) \in V$, where V is the C^1 space given by:

$$V = \{\psi_i \in C^1(\Omega) : \psi|_{\Omega^{(\ell)}} \in \hat{\Omega}^{(\ell)} \circ (G^{(\ell)})^{-1}, \ell \in \{1,2,\ldots,n_\ell\}\}. \tag{5.31}$$

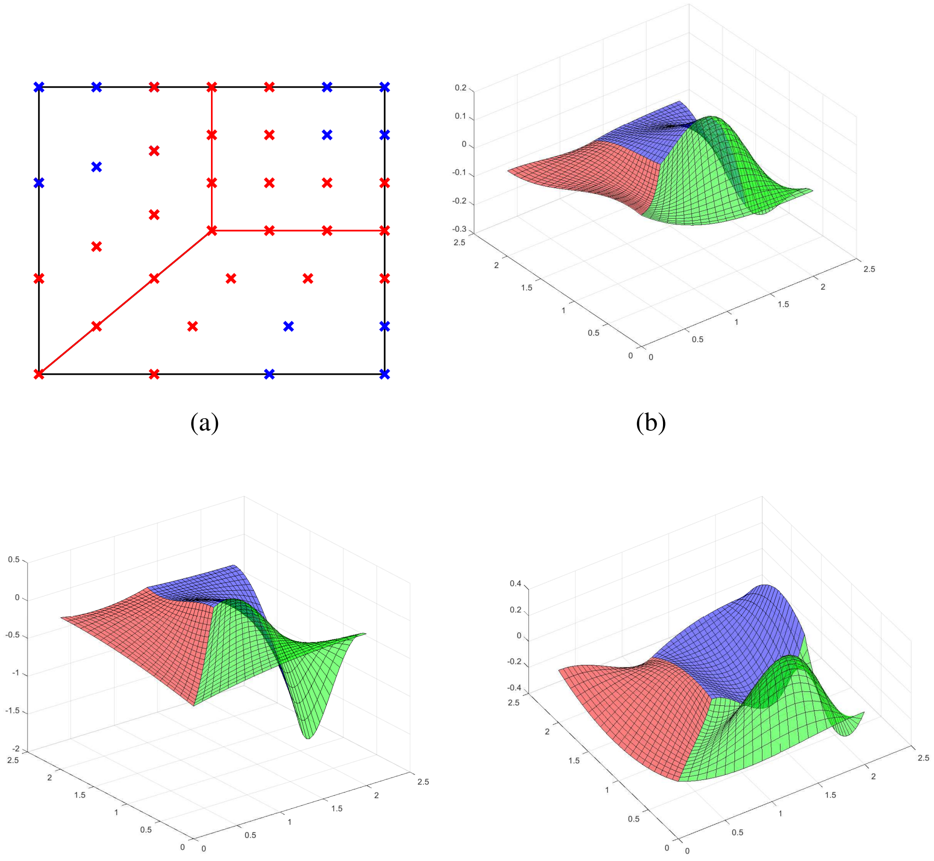
5.4.0.1 Bézier ordinate approach

As usual, the procedure start by determining the relevant edges and relevant coefficients. Followed by select sufficient collocation points such that condition (5.28) is fulfilled. Next, each collocation point is substituted into the continuity condition to obtain a continuity constraint. Note that the condition have to be defined based on the tangent plans of the graph surfaces of $\psi_i^{(\ell)}(x,y)$ along the edge, where $\hat{\psi}_i^{(\ell)}(\xi,\eta)$ and $F_i^{(\ell)}(\xi,\eta)$ are defined as (5.20) and (5.21) respectively. Each constraint is then used to form a row of the homogeneous linear system:

$$M\varsigma^i = 0,\ \varsigma^i = (\varsigma_\iota)_{\iota=1,\ldots,n_\varsigma},$$

Next, we seek for the null space of M, and each resulting basis vector corresponds to a set of values of ς^i which substitute into (5.30) to obtain a new isogeometric function $\psi_i(x,y)$.

An example of C^1 coupling for a geometry formed by three patches which meet at an extraordinary vertex is shown in Figure 5.15. For this example, we consider the simple case where only one element in each patch. The meshes are shown in Figure 5.15(a). The red lines in the figure are the common boundary, and the crosses represent the coefficients. In particular, the red crosses denote the relevant coefficients. An example of globally C^1 isogeometric function is shown in Figure 5.15(b), and the corresponding x and y derivatives are given in Figures 5.15(c) and 5.15(d) respectively.

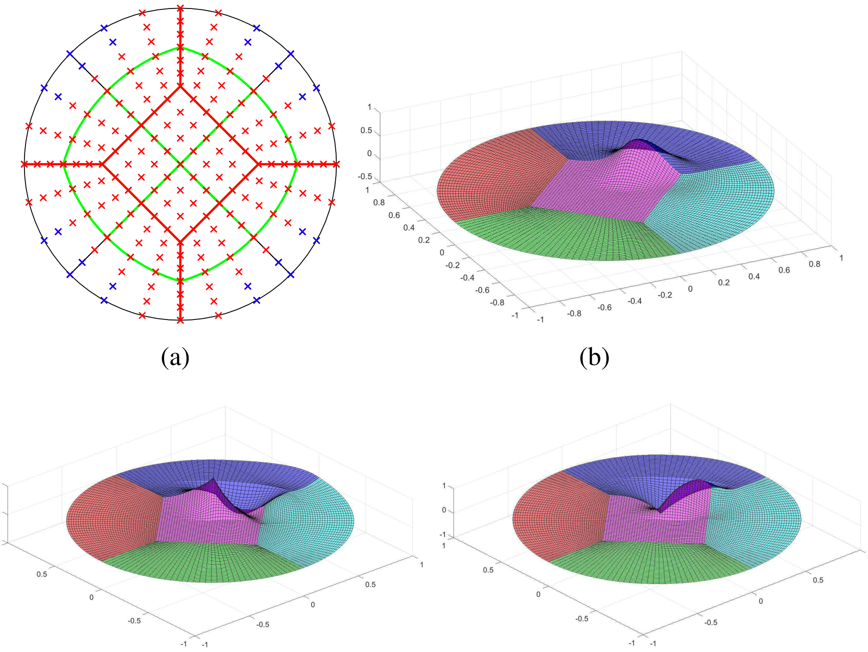


(a) (b)

(c) (d)

Figure 5.15: Three patch example: (a) A three-patch domain where the Bézier coefficients relevant to the common boundary are represented by red dots, (b) A Type 2 basis function, (c) and (d) x and y derivatives of (b).

Another example involves a quadratic NURBS geometry mapping is given in Figure 5.16; the physical domain is a circle formed by five patches, and each patch has four elements. The mesh is shown in Figure 5.16(a), where there are four extraordinary vertices which are the meet points of three patches. In the figure, the red lines represent the common boundaries between two patches, while the green lines are the edges shared by two elements and connected to the common boundaries. All these edges have to be taken into account when imposing the continuity constraints. The relevant coefficients influencing the common boundaries as well as the green edges are marked by red dots, while the blue dots represent the other coefficients. An examples of the globally C^1 basis function and its x and y derivatives are given in Figures 5.16(b), 5.16(c), and 5.16(d) respectively.



(a) (b)

(c) (d)

Figure 5.16: Fibe patch circle example: (a) A five-patch circle, (b) a Type 2 basis function, (c) and (d) x and y derivatives of (b).

The C^1 coupling using Bézier ordinate approach is summarized in Algorithm 6. The algorithm presented is suitable for coupling of $\ell \geq 2$ patches on Ω, involving an arbitrary number of elements.

5.4.0.2 Spline approach

For the spline approach, the multi-patch coupling starts with determining the relevant edges (edges on the common boundaries) and removed functions. Then we determine the number of collocation points such that condition (5.29) is fulfilled. Each collocation point is used to obtain a continuity constraint by substituting it into a carefully determined continuity condition. The resulting set of constraints is then used to develop a homogeneous linear system:

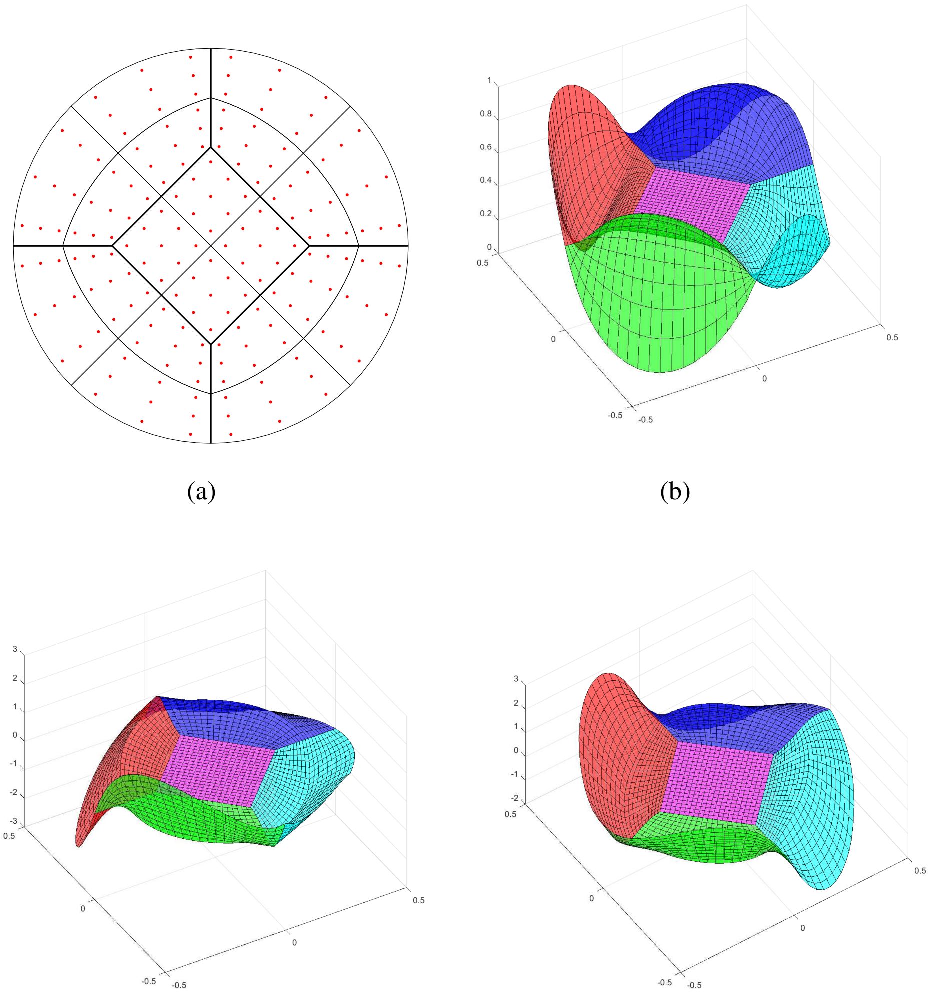
$$Mc^i = 0, c^i = (c_i)_{i=1,\ldots,n_{\mathcal{R}}}.$$

As usual, the globally C^1 continuous new functions can be obtained by finding the null space of M, where each basis of the null space determines a new function.

Algorithm 6: Algorithm for C^1 coupling using Bézier ordinate approach

```
Input  : $\Omega$
Output: $\psi_i(x,y)$
Select boundary elements associated to the common boundaries
  $\tau_j, \quad j = 1, 2, \ldots n_\tau$, where $n_\tau$ is the number of common boundaries.
Determine relevant coefficients $\varsigma_k$, $k = 1, \ldots, n_\varsigma$ from the boundary elements.
Based on $\varsigma_k$, $k = 1, \ldots, n_\varsigma$, select the relevant edge $e_l, l = 1, \ldots, n_e$.
Based on $n_\varsigma$ and $n_e$, determine number of uniform collocation points $|\rho_l|$ to be
  selected on each $e_l$ such that fulfill (5.28).
For each $e_l$:
    determine the continuity condition by using (5.21),
    select uniformly distributed collocation points on $e_l$, and
    substitute each collocation points into the continuity condition to obtain a
      continuity constraint.
Written the constrains as a homogeneous linear system $M\varsigma^i = 0$.
Find the null space of $M$.
A new function $\psi_i^{(\ell)}(x,y) = \hat{\psi}_i^{(\ell)} \circ {G^{(\ell)}}^{-1}(x,y)$, where
  $\hat{\psi}_i^{(\ell)}(\xi,\eta) = \sum_{\iota=1}^{n_\beta} \varsigma_\iota^{(\ell)} \beta_\iota(\xi,\eta)$ is obtained by assigning $\varsigma^i = (\varsigma_\iota)_{\iota=1,\ldots,n_\varsigma}$ with a
  basis vector of the null space.
```

Here, we discuss the influence of the distribution of the removed functions on the new basis functions. Figures 5.17(a), 5.18(a) and 5.19(a) shows a circle domain composed of five patches; the meet point of every three patches is a extraordinary vertex. In the figures, the bold dark lines are the common boundaries, and the dots denote the quadratic isogeometric functions. Meanwhile, the red dots denote the removed functions, and the blue dots are the original functions.

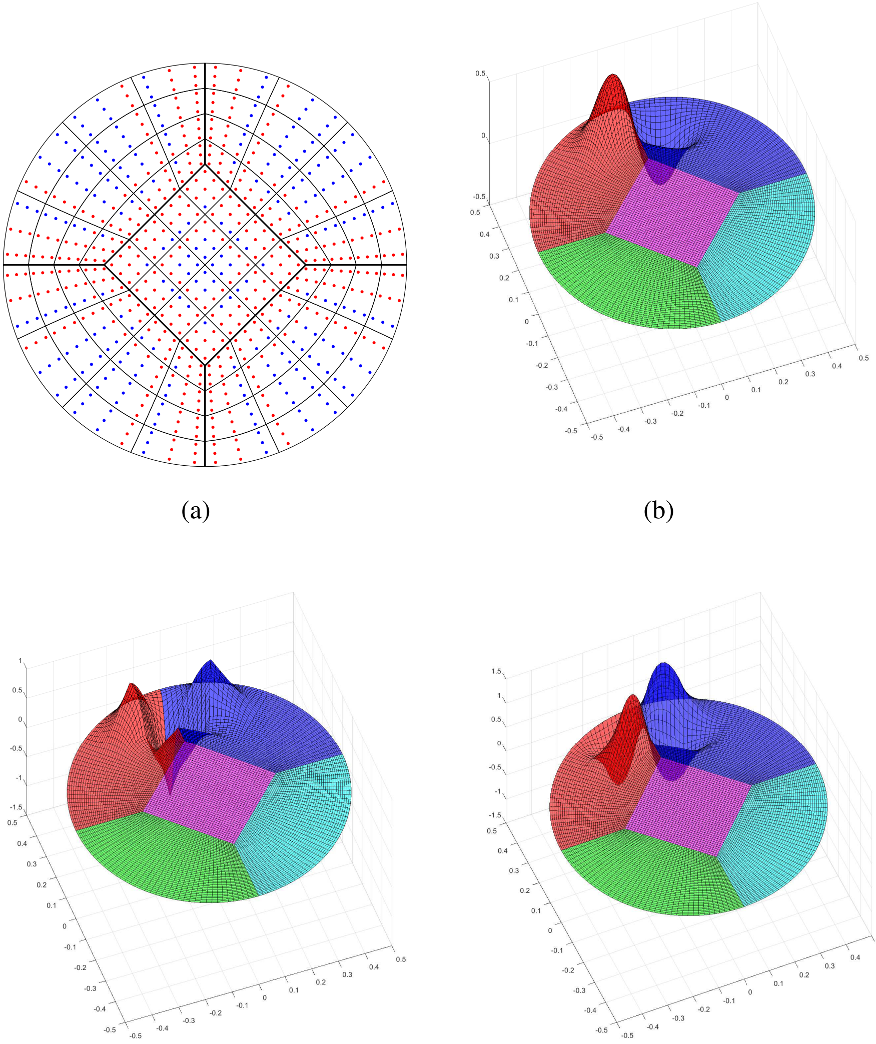


(a) (b)

(c) (d)

Figure 5.17: Circle example: (a) course mesh and distribution of quadratic isogeometric functions (b) quadratic new function, (c) and (d) x and y derivatives of (b).

An important aspect for the spline approach is that the support area of the new functions highly depends on the removed functions. As shown in Figures 5.17(a), the coarse mesh is dominated by removed functions. Thus, the resulting new functions will have support on the entire domain too. An example of C^1 continuous new function is given in Figure 5.17(b), and its corresponding x and y derivatives are shown in Figures 5.17(c) and 5.17(d) respectively. We can see that the function has support on the entire domain and it fluctuate throughout the domain.

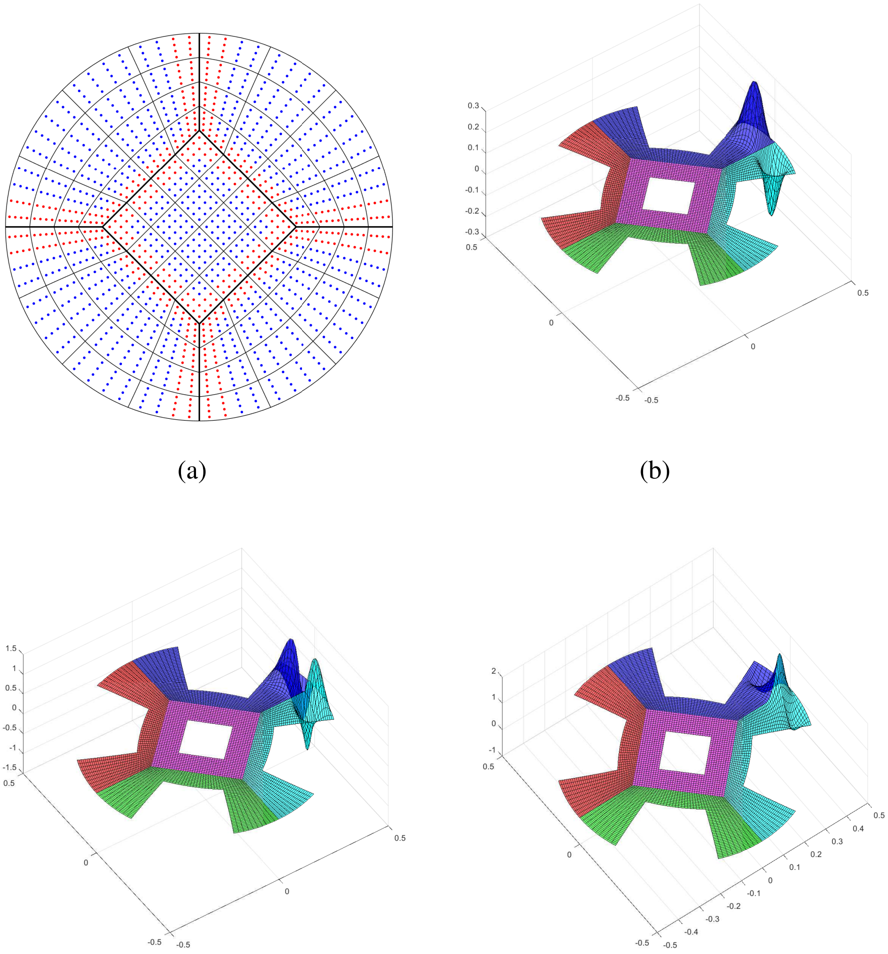


(a) (b)

(c) (d)

Figure 5.18: Circle example: (a) fine mesh and distribution of quadratic isogeometric functions (b) quadratic new function, (c) and (d) x and y derivatives of (b).

Despite the mesh size, the new functions computed with lower degree removed functions will have a larger support area. The reason for this is that the low degree removed functions have supported not only the elements associated with the common boundary but also the neighboring elements (see Figure 5.18(a)). An example of a C^1 new functions is shown in Figure 5.18(b), and its corresponding x and y derivatives are given in Figures 5.18(c) and 5.18(d) respectively. Though this function has support on the entire domain, it is rather obvious that it is more localized in the finer mesh. It has higher support around a particular common boundary between two patches.



(a) (b)

(c) (d)

Figure 5.19: Circle example:(a) fine mesh and distribution of cubic isogeometirc functions, (b) C^1 continuous new function, (c) and (d) x and y derivatives of (b).

Nevertheless, higher degree removed functions can be used for the construction of more localized new functions. Figure 5.19(a) shows the solution space formed by cubic isogeometric functions. The distribution of the removed functions is concentrated on around the common boundaries. Therefore, the new functions will have support on the neighborhood around the common boundaries. An example of cubic C^1 continuous function and its derivatives is given in Figures 5.19(b), 5.19(c) and 5.19(d) respectively.The new function has higher support in a relatively small area compared to the two previous examples, which is to say that it has better localization properties.

The algorithm for the spline approach is given in Algorithm 7. It is a general algorithm for multi-patch coupling and arbitrary mesh size.

Algorithm 7: Algorithm for C^1 coupling using spline approach

```
Input  : $\Omega$
Output: $\psi_i(x,y)$
Select boundary elements associated to the common boundaries
  $\tau_j, \quad j = 1,2,\dots n_\tau$, where $n_\tau$ is the number of common boundaries.
Determine removed functions $\mathcal{R}_\iota(x,y)$, $\iota = 1,\dots,n_\mathcal{R}$.
Determine edges $e_l$, $l = 1,\dots,n_e$which form the common boundaries.
Based on $n_\mathcal{R}$ and $n_e$, determine number of uniform collocation points $|\rho_l|$ to be
  selected on each edge such that fulfill (5.29).
For each $e_l$:
determine the continuity condition by using (5.23) and (5.24).
select uniformly distributed collocation points on $e_l$, and
substitute each collocation points into the continuity condition to obtain a
  continuity constraint.
Write the constrains as a homogeneous linear system $Mc^i = 0$.
Find the null space of $M$.
A new function $\psi_i(x,y) = \sum_{\iota=1}^{n_\mathcal{R}} c_\iota \mathcal{R}_\iota(x,y)$ is obtained by assigning
  $c^i = (c_\iota)_{\iota=1,\dots,n_\mathcal{R}}$ with a basis vector of the null space.
```

5.4.1 Localization using MDS

The standard orthogonal null space that considers all the coefficients along the common boundaries would result in isogeometric functions which have supported over all the elements associated with the boundaries. For a domain with complex geometries or fine discretizations, this will leads to ill-condition and non-banded stiffness matrices formed by non-zeros entries. A solution to this issue is to obtain the sparse nullspace, such that the resulting C^1 isogeometric functions are localized.

Referring to[92], the finding of an optimal (sparsest) basis is considered as an NP-hard problem. Nevertheless, we can compute a localized basis by determining the minimal determining set (MDS) of the relevant coefficients. The MDS represents a set of coefficients which we are allowed to assign arbitrary values, and yet guarantee C^1 continuity of the new functions. A detail discussion for computing the MDS can be found in[81]. Here, we determine the MDS with a simpler approach, similar to the one in[71]. The computation steps are given in Algorithm 8. After obtaining the MDS, we carry out the algorithm 9 to obtain the C^1 functions with localization property. In the Algorithm, n_MDS and n_ς represent the number of entries in MDS and number of relevant coefficient respectively, while MDS_j denote the jth index in the MDS set.

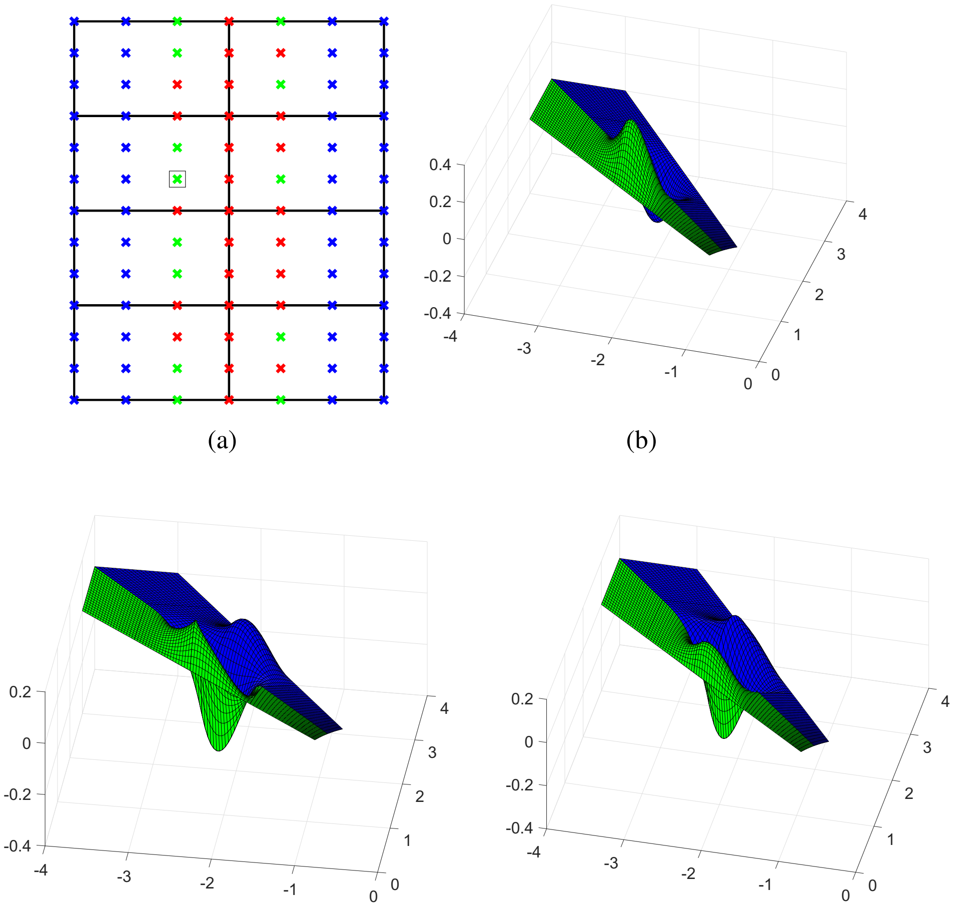
Algorithm 8: Algorithm for computing MDS

```
Input  : $n_\varsigma \times m$ matrix $T$
Output: MDS
 1: $\tilde{I} = \{\}$
 2: $\bar{I} = \{1, 2, \ldots, n_\varsigma\}$
 3: for $\hat{\imath} = 1, 2, \cdots, n_\varsigma$ do
 4:     Solve $T\varsigma^i = 0$, $\varsigma^i = (\varsigma_\imath)_{\imath=1,\ldots,n_\varsigma}$, with $\varsigma_{\imath=\hat{\imath}} = 1$ and $\varsigma_{\imath=j} = 0$ for $j \in \tilde{I} \cup (\bar{I} \backslash \{\hat{\imath}\})$,
 5:     if $\varsigma^i$ exist do
 6:         find minimal index $r$ such that $\varsigma_{\imath=r} = \max(\varsigma^i)$,
 7:         set $\tilde{I} = \tilde{I} \cup \{r\}$,
 8:     end if
 9:     set $\bar{I} = \bar{I} \backslash \{\hat{\imath}\}$.
10: end for
11: return
12: $MDS = \tilde{I}$.
```

Algorithm 9: Algorithm for localizing the new functions

```
Input  : MDS, $T$
Output: $\varsigma^i$
 1: let $\tilde{I} = \{1, \ldots, n_{\mathrm{MDS}}\}$, and $I = \{1, \ldots, n_\varsigma\}$,
 2: for $j = 1, \ldots, n_{\mathrm{MDS}}$ do
 3:     set $k = \tilde{I} \setminus j$,
 4:     set $\varsigma_{\mathrm{MDS}_j} = 1$ and $\varsigma_{\mathrm{MDS}_k} = 0$ ,
 5:     by using $\varsigma_{\mathrm{MDS}}$ obtained from step 4, solve $T\varsigma^i = 0$ for the unknown $\varsigma_\imath$ where $\imath = I \setminus \mathrm{MDS}$.
 6: end for
```

In the following, we show an example of a localized C^1 functions for the plate with a central circular hole domain (see Figure 5.13). Figure 5.20(a) shows the elements associated with the common boundary, and the crosses represent the coefficients. In particular, the green crosses are the relevant coefficients belong to the MDS (determined through Algorithm 8). An example of localized C^1 function is given in Figure 5.20(b). It is localized by setting a selected coefficient to belong to MDS (green cross marked with a square box) as 1, and the others as zeros. We seek for the values of the other relevant coefficients (denoted by the red dots) using Algorithm 9. We can observe from Figure 5.20(b) that the functions are localized around the support of the selected coefficients and its neighbors. The *x* and *y* derivatives of the function are shown in Figures 5.20(c) and 5.20(d) respectively.



(a) (b)

(c) (d)

Figure 5.20: Localization example: (a) relevant coefficients and minimal MDS, (b) localized basis function, (c)-(d) *x* and *y* derivatives of (b).

Though Algorithms 8 and 9 are written for Bézier ordinate approach, the similar algorithms can be applied for localization in the spline approach. The implication is straightforward; therefore, we will not further discuss it here.

5.5 Partial degree elevation

It is pointed out in[73] that for certain geometries, the over constraining of the solution space would lead to a suboptimal order of approximation. In this situation, the error in the approximation would not decrease even when a finer mesh is used. As a result, the convergence rate is restricted, and this circumstance is known as C^1 locking. It is known that the C^1 locking issue can be mitigated by either using a lower order continuity isogeometric functions or by increasing the degree of the functions. These approaches would restore the convergence, in spite of at a suboptimal rate.

In this study, the continuity of the solution space is restricted to be C^1. We attempt to overcome the C^1 locking by using higher degree isogeometric functions; which results in a larger approximation space. In particular, we propose to perform *partial degree elevation*, which means only degree elevate the functions near to the common boundaries. A related approach was discussed in[93] and[94] in the context of unstructured meshes using T-splines, where a hybrid of degree elevation and optimization method was used to ensure continuity and improve the approximation near extraordinary vertex.

The idea of partial degree elevation is developed from hierarchical B-splines[95], for which the hierarchical mesh is supported by a combination of basis functions defined by nested knot sequences associated with the nested domain. In partial degree elevation, the mesh is supported by the combination of p and $p+1$ degree basis functions defined on the same knot sequence. The p degree functions which introduce C^0 continuity on the common boundaries τ in the multi-patch domain will be removed and replaced with the corresponding $p+1$ degree basis functions. Note that if C^1 locking occurs, partial degree elevation will be performed on each parametric domain and then we perform the C^1 coupling again by using the degree elevated basis functions.

Consider a mesh $Q^{(\ell)}$ on the parametric domain $\hat{\Omega}^{(\ell)}$ of patch ℓ. Suppose R is set of p degree removed functions, and $\hat{\Omega}_\tau^{(\ell)} \subseteq \hat{\Omega}^{(\ell)}$ is a subdomain containing the common boundaries τ. Suppose N_p and N_{p+1} represent the p and $p+1$ degree basis functions defined by the corresponding knot sequence of $Q^{(\ell)}$. The partial degree elevated basis functions on $Q^{(\ell)}$ are defined as follows:

$$\hat{N}_p(Q^{(\ell)}) := \{N_p \in N_p : \mathrm{supp} N_p \subseteq \hat{\Omega}^{(\ell)} \wedge N_p \notin R\}, \tag{5.32}$$

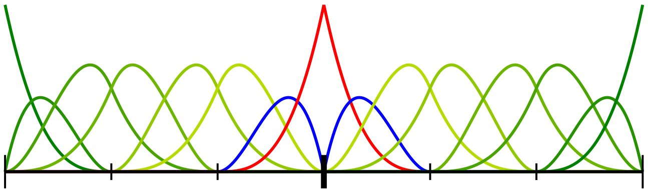
and

$$\hat{N}_{p+1}(Q^{(\ell)}) := \{N_{p+1} \in N_{p+1} : \mathrm{supp} N_{p+1} \subseteq \hat{\Omega}^{(\ell)} \wedge \mathrm{supp} N_{p+1} \subseteq \hat{\Omega}_\tau^{(\ell)}\}. \tag{5.33}$$

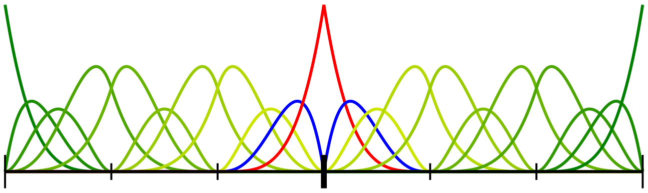
After partial degree elevation, the subdomain $\hat{\Omega}_\tau^{(\ell)}$ will be supported by $\hat{N}_p(Q^{(\ell)})$ and

$\hat{N}_{p+1}(Q^{(\ell)})$ simultaneously, while the remaining domain will be supported by $\hat{N}_p(Q^{(\ell)})$.

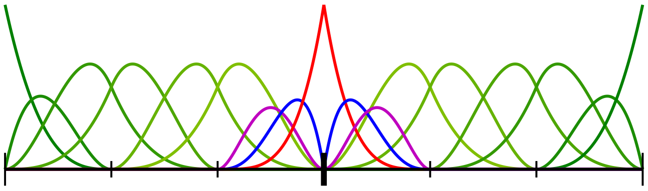
In the following, we describe the basis functions replacement by using a two patch parametric domain in 1D. The patches are jointed at a common point as shown in Figure 5.21. Figure 5.21(a) shows the cubic bases on a two patch domains, where the red and blue curves indicate the bases which possess C^0 continuity (they are the cubic removed functions). To perform partial degree elevation, we replace them with quartic basis function (the red and blue curves in Figure 5.21(b)). Meanwhile, the quartic functions adjacent to the red and blue curves in Figure 5.21(b) are added for complete support, they are selected according to (5.33).



(a)



(b)



(c)

Figure 5.21: (a) solution space formed by cubic basis functions, (b) solution space formed by quartic basis functions, and (c) solution space with combination of cubic and quartic basis functions.

Figure 5.21(c) shows the partial degree elevated solution space equipped with the three types of functions:

- the cubic functions (represented by the green curves),
- the degree elevated functions (quartic basis functions represented by blue and red curves),
- the additional quartic functions (denoted by the purple curves).

After degree elevating the functions, we continue to perform the C^1 coupling using the method as described in the previous sections. For the Bézier ordinate approach, the new $p+1$ degree C^1 continuous functions will be given by:

$$\psi_{i,p+1}(x) = \hat{\psi}_{i,p+1} \circ G^{-1}(x), \tag{5.34}$$

where

$$\hat{\psi}_{i,p+1}(\xi) = \sum_{\iota=1}^{n_\beta} \varsigma_\iota \beta_{\iota,p+1}(\xi). \tag{5.35}$$

For the spline approach, we will compute the new functions with the C^0 degree elevated functions as:

$$\psi_{i,p+1}(x) = \sum_{\iota \in I_\ell} c_\iota \mathcal{R}_{\iota,p+1}(x). \tag{5.36}$$

5.5.1 Partial degree elevation for higher dimension

Similar concept discussed above can be applied to perform partial degree elevation on higher dimensions. Figure 5.22 shows the partial degree elevation on the elements associated with the common boundary in a 2D domain. Figures 5.22(a) and 5.22(c)shows the solution space approximated by cubic basis functions and quartic basis functions respectively. The red dots in Figure 5.22(a) represent the C^0 functions which will be replaced by the associative quartic functions, which are represented by the dots in the area highlighted by red in Figure 5.22(c). The combination of cubic and quartic basis functions resulting from the partial degree elevation is shown in Figure 5.22(b), where the dots in the green color area denote the cubic functions, while the dots in the red color area represent the quartic functions. Within the red area, the red dots symbolize the C^0 functions, and the functions represented by the blue dots serve as the additional functions.

In Figure 5.23, we illustrate for the circle domain the three different types of basis functions involved in partial degree elevation. The blue and red dots represent the cubic functions and the degree elevated functions (quartic functions) respectively.

Meanwhile, the green dots denote the additional quartic functions. A comparison with a solution that is affected by C^1 locking and improved solution from partial degree elevation is given in the numerical section.

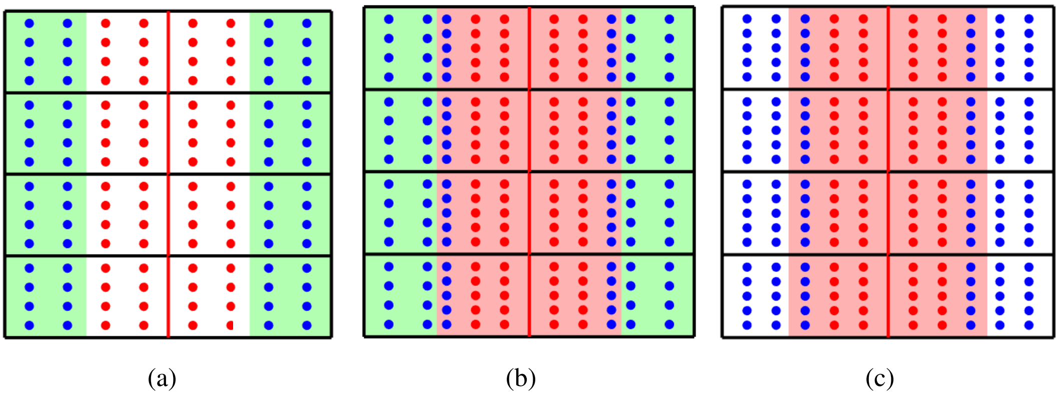

(a) (b) (c)

Figure 5.22: Basis functions in partial degree elevations: (a) cubic basis functions, (b) combined basis functions, and (c) quartic basis functions.

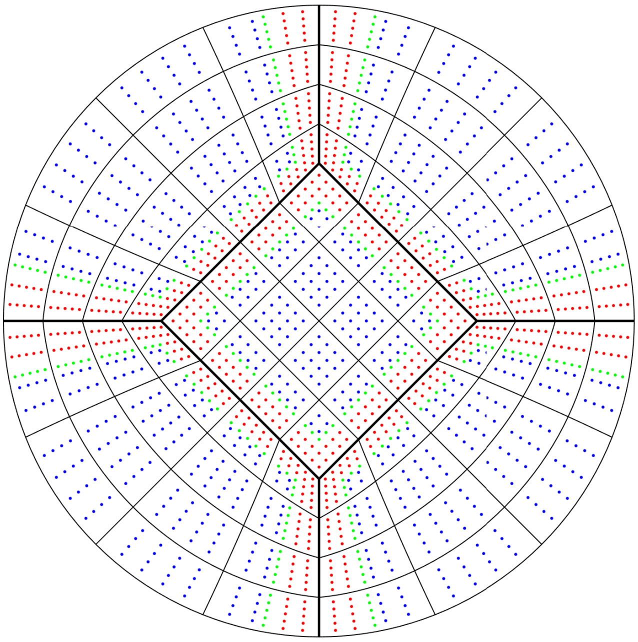


Figure 5.23: Degree elevation in 2D multi-patch domain with extraordinary vertices.

5.6 C^1 coupling for surface in space

In this section, we present the C^1 coupling for surface by using the Bézier ordinate approach. Suppose the geometric mapping in the space is defined as:

$$G^{(\ell)}(\xi,\eta) = \sum_{i=0}^{n_N} N_i(\xi,\eta) P_i^{(\ell)} = (x,y,z), \tag{5.37}$$

where $P_i^{(\ell)} \in \mathbb{R}^3$ are the corresponding control points of patch ℓ. For an arbitrary parametric point (ξ,η) on patch (ℓ), its corresponding position on the surface is given by:

$$\mathbf{X} = G^{(\ell)}(\xi,\eta) = [x(\xi,\eta), y(\xi,\eta), z(\xi,\eta)], \tag{5.38}$$

and the Jacobian has the form:

$$J = \begin{bmatrix} \frac{\partial x(\xi,\eta)}{d\xi} & \frac{\partial y(\xi,\eta)}{d\xi} & \frac{\partial z(\xi,\eta)}{d\xi} \\ \frac{\partial x(\xi,\eta)}{d\eta} & \frac{\partial y(\xi,\eta)}{d\eta} & \frac{\partial z(\xi,\eta)}{d\eta} \end{bmatrix}. \tag{5.39}$$

The geometric mapping and Jacobian are important for the computation of derivatives and Laplacian of the basis functions in the space. They are needed for solving the problem involving PDEs on surfaces. More relevant detail can be found in[96].

In the following, we describe the coupling of two patch surfaces which meet at the common boundary $\tau = G^{(1)}(1,\eta) = G^{(2)}(0,\eta)$. We seek for the new functions:

$$\psi_i(\mathbf{X}) = \psi_i^{(1)}(\mathbf{X}) + \psi_i^{(2)}(\mathbf{X}) \tag{5.40}$$

which belong to the C^1 space. We know that $\psi_i^{(1)}(\mathbf{X})$ and $\psi_i^{(2)}(\mathbf{X})$ are C^1 continuous at τ if they have the same derivatives. Therefore the continuity condition is given by:

$$\nabla \psi_i^{(1)}(\mathbf{X}) = \nabla \psi_i^{(2)}(\mathbf{X}), \quad \forall \mathbf{X} \in \tau. \tag{5.41}$$

The surface gradient is can be computed as:

$$\nabla \psi_i^{(\ell)}(\mathbf{X}) = (J \mathcal{G}^{(\ell)^{-T}} \nabla(\hat{\psi}_i^{(\ell)} \circ G^{(\ell)^{-1}}))(\mathbf{X}), \tag{5.42}$$

where $\hat{\psi}_i^{(\ell)}(\xi,\eta)$ is defined as (5.20), and the first fundamental form $\mathcal{G}^{(\ell)}$ is given by:

$$\mathcal{G}^{(\ell)} = J^T J. \tag{5.43}$$

The continuity constraints can be obtained by substituting collocation points into (5.41). Note that for each collocation point, there will be three continuity constraints which correspond to x, y, and z derivatives respectively. We then use the continuity constraints to develop the homogeneous linear system:

$$M\varsigma^i = 0.$$

Finally, we seek for the nullspace of M. The basis vector of the nullspace is taken as ς^i for construction of the C^1 isogeometric functions. The similar approach presented in Section 5.3.3 can be applied for coupling of meshes formed by finer mesh.

5.7 C^1 coupling for 3D

In this section, we describe C^1 coupling for 3D by using the spline approach. Spline approach is suitable for 3D because it involves fewer parameters for which we only have to impose continuity constraints along the common interfaces. Here, we describe the extension of the spline approach to three dimensional domains. Consider two parametric spaces $\hat{\Omega}^{(1)}$ and $\hat{\Omega}^{(2)}$ which are mapped to the physical space $\Omega = \Omega^{(1)} \cup \Omega^{(2)}$ by the geometric mapping $G^{(\ell)} : [0,1]^3 \to \mathbb{R}^3$, $\ell \in \{1,2\}$ (see Figure 5.24).

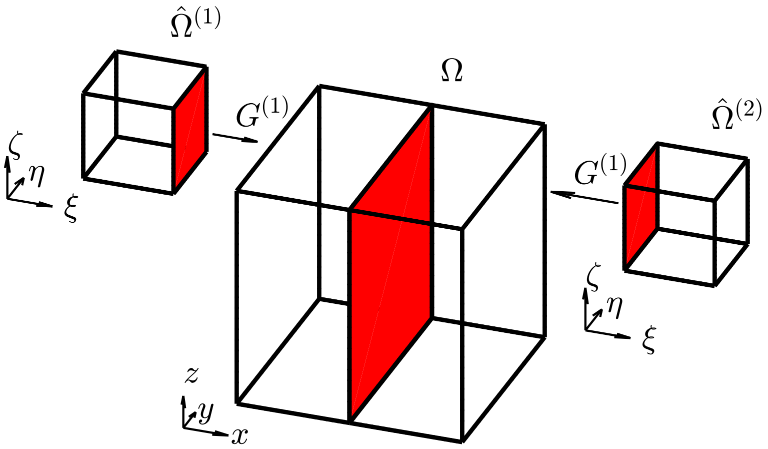


Figure 5.24: Mapping from parametric space to physical space in 3D.

In the physical space, the two patches are attached together at the common interface $\tau = G^{(1)}(1,\eta,\zeta) = G^{(2)}(0,\eta,\zeta)$ (highlighted by red in the figure). Similarly with 2D, the patches are C^0 continuous at the common interface τ, therefore, they share the isogeometric functions which have support on τ. In Figure 5.25, the dots denote the cubic isogeometric functions on Ω. The red dots represent the functions which possess C^0 continuity, hence, they are set as the removed functions.

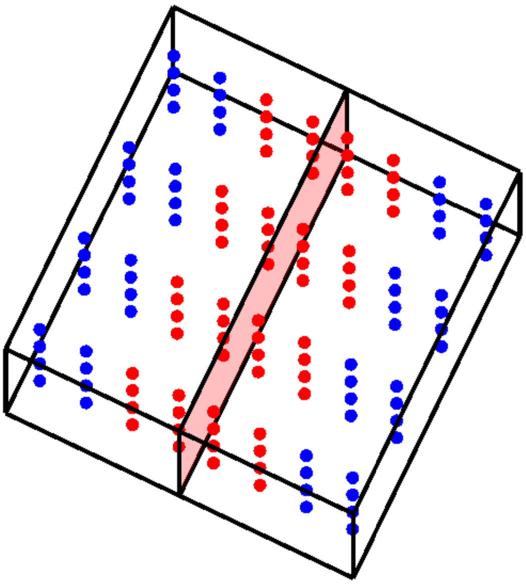


Figure 5.25: Removed basis functions in 3D.

Recall that we construct the new functions $\psi_i(x,y,z) \in V$ as a linear combination of the removed functions. Suppose that $\psi_i(x,y,z) = \psi_i^{(1)}(x,y,z) + \psi_i^{(2)}(x,y,z)$, where $\psi_i^{(\ell)}(x,y,z)$ have support on patch ℓ. The C^1 continuity condition requires the derivatives of $\psi_i^{(1)}(x,y,z)$ to be equal with $\psi_i^{(2)}(x,y,z)$:

$$\nabla \psi_i^{(1)}(x,y,z) = \nabla \psi_i^{(2)}(x,y,z). \tag{5.44}$$

The continuity condition can be imposed through the (4 dimensional) graph surface of $\psi^{(\ell)}(x,y,z)$, which is given by:

$$F^{(\ell)}(\xi,\eta,\zeta) = (G^{(\ell)}(\xi,\eta,\zeta), \hat{\psi}^{(\ell)}(\xi,\eta,\zeta))^T, \tag{5.45}$$

where $\hat{\psi}^{(\ell)}(\xi,\eta,\zeta)$ is the associate function of $\psi^{(\ell)}(x,y,z)$ defined on the parameter space. The continuity condition is given by:

$$\det\left(\frac{\partial F^{(1)}(1,\eta,\zeta)}{\partial \xi}, \frac{\partial F^{(1)}(1,\eta,\zeta)}{\partial \eta}, \frac{\partial F^{(1)}(1,\eta,\zeta)}{\partial \zeta}, \frac{\partial F^{(2)}(0,\eta,\zeta)}{\partial \xi}\right) = 0. \tag{5.46}$$

In (5.46), $\frac{\partial F^{(2)}(0,\eta,\zeta)}{\partial \eta}$ and $\frac{\partial F^{(2)}(0,\eta,\zeta)}{\partial \zeta}$ are neglected because we assume that the two patches have the same η and ζ partial derivatives on τ. Note that (5.46) is a 4×4

matrix, and its cofactor expansion is equivalent to:

$$\alpha(1,\eta,\zeta)\frac{\partial\psi^{(1)}(1,\eta,\zeta)}{\partial\xi}+\beta(1,\eta,\zeta)\frac{\partial\psi^{(1)}(1,\eta,\zeta)}{\partial v}+\gamma(1,\eta,\zeta)\frac{\partial\psi^{(1)}(1,\eta,\zeta)}{\partial\zeta}$$
$$+\delta(0,\eta,\zeta)\frac{\partial\psi^{(2)}(0,\eta,\zeta)}{\partial\zeta}=0, \quad (5.47)$$

where

$$\alpha(1,\eta,\zeta)=\det\begin{bmatrix}\frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\zeta} & \frac{\partial G_1^{(2)}(0,\eta,\zeta)}{\partial\xi}\\ \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\zeta} & \frac{\partial G_2^{(2)}(0,\eta,\zeta)}{\partial\xi}\\ \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\zeta} & \frac{\partial G_3^{(2)}(0,\eta,\zeta)}{\partial\xi}\end{bmatrix},$$

$$\beta(1,\eta,\zeta)=-\det\begin{bmatrix}\frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\zeta} & \frac{\partial G_1^{(2)}(0,\eta,\zeta)}{\partial\xi}\\ \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\zeta} & \frac{\partial G_2^{(2)}(0,\eta,\zeta)}{\partial\xi}\\ \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\zeta} & \frac{\partial G_3^{(2)}(0,\eta,\zeta)}{\partial\xi}\end{bmatrix},$$

$$\gamma(1,\eta,\zeta)=-\det\begin{bmatrix}\frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_1^{(2)}(0,\eta,\zeta)}{\partial\xi}\\ \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_2^{(2)}(0,\eta,\zeta)}{\partial\xi}\\ \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_3^{(2)}(0,\eta,\zeta)}{\partial\xi}\end{bmatrix},$$

and

$$\delta(0,\eta,\zeta)=\det\begin{bmatrix}\frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_1^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_1^{(1)}(0,\eta,\zeta)}{\partial\zeta}\\ \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_2^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_2^{(1)}(0,\eta,\zeta)}{\partial\zeta}\\ \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\xi} & \frac{\partial G_3^{(1)}(1,\eta,\zeta)}{\partial\eta} & \frac{\partial G_3^{(1)}(0,\eta,\zeta)}{\partial\zeta}\end{bmatrix}.$$

We construct the homogeneous linear system $Mc = 0$ by substituting uniformly distributed collocation points on τ into (5.47).

For interface $\tau = G^{(1)}(\xi,1,\zeta) = G^{(2)}(\xi,0,\zeta)$, the continuity condition has to be changed accordingly as follows:

$$\det\left(\frac{\partial F^{(1)}(\xi,1,\zeta)}{\partial\xi},\frac{\partial F^{(1)}(\xi,1,\zeta)}{\partial\eta},\frac{\partial F^{(1)}(\xi,1,\zeta)}{\partial\zeta},\frac{\partial F^{(2)}(\xi,0,\zeta)}{\partial\eta}\right)=0, \quad (5.48)$$

while for $\tau = G^{(1)}(\xi,\eta,1) = G^{(2)}(\xi,\eta,0)$, the condition will be given by:

$$\det\left(\frac{\partial F^{(1)}(\xi,\eta,1)}{\partial\xi},\frac{\partial F^{(1)}(\xi,\eta,1)}{\partial\eta},\frac{\partial F^{(1)}(\xi,\eta,1)}{\partial\zeta},\frac{\partial F^{(2)}(\xi,\eta,0)}{\partial\zeta}\right)=0. \quad (5.49)$$

The implementation for finer meshes is alike with that presented for 2D; therefore they will not be discussed in detail here.

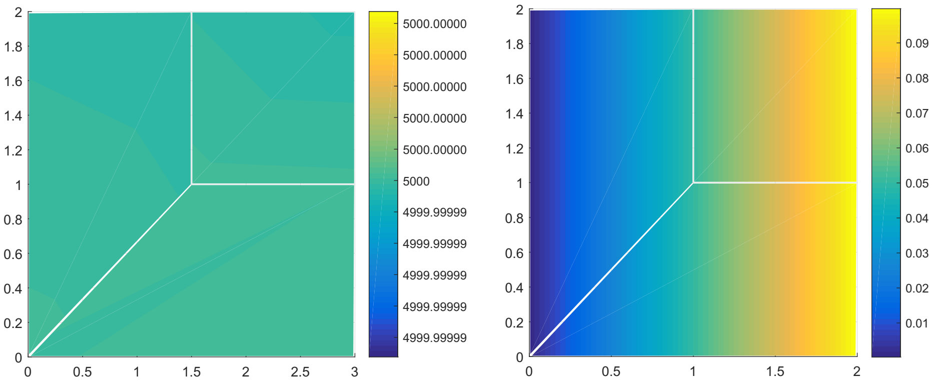
5.8 Numerical results

In this section, we demonstrate the use of the smooth solution space resulting from the proposed coupling methods for difference analysis purpose. In particular, we solve problems involving fourth order partial differential equations. We will start by showing the results for the Bézier ordinate approach; these included the Kirchhoff-Love shell models and Cahn-Hilliard phase field applications. Then, we show the results obtained from the spline approach, focusing on the Biharmonic problem.

5.8.1 Linear elasticity

5.8.1.1 Patch test

For the first example, we verify the multi-patch coupling with a simple patch test. We use a mesh as shown in Figure 5.15(a), where the square is formed by three patches which meet at an extraordinary vertex. In the patch test, we fix the left side of the square and pull right side horizontally. The resulting (constant) stress field is given in Figure 5.26(a), while Figure 5.26(b) shows the (linear) displacement in the x direction. From the figures, we can see that at least linear displacements can be represented exactly and the "patch test" is passed.



(a) (b)

Figure 5.26: Patch test of multi-patch coupling: (a) von Misses stresses, and (b) displacement in x direction.

5.8.1.2 Infinite plate with a hole

Next, we show the example of a benchmark problem of an infinite plate with hole subject to uniaxial tension at infinity. In this example, partial degree elevation is implemented to overcome C^1 locking.

The domain of the boundary elements associated with the common boundary is composed of cubic and quartic basis functions. In particular, the C^1 continuous new functions which have support at the common boundary have degree $p = 4$. The fine mesh and the stress field in the y-direction are shown in Figures 5.27(a) and 5.27(b) respectively, while the plot comparing the rates of convergence is given in Figure 5.27(c). From the graph, we can see that when cubic isogeometric functions approximate the entire solution space, the result is not converging due to C^1-locking. Improved results were obtained when the solution space is approximated by quartic isogeometric functions (see the pink line). By using partial degree elevation, the results in terms of degrees of freedom are further improved (as shown by the red dashes).

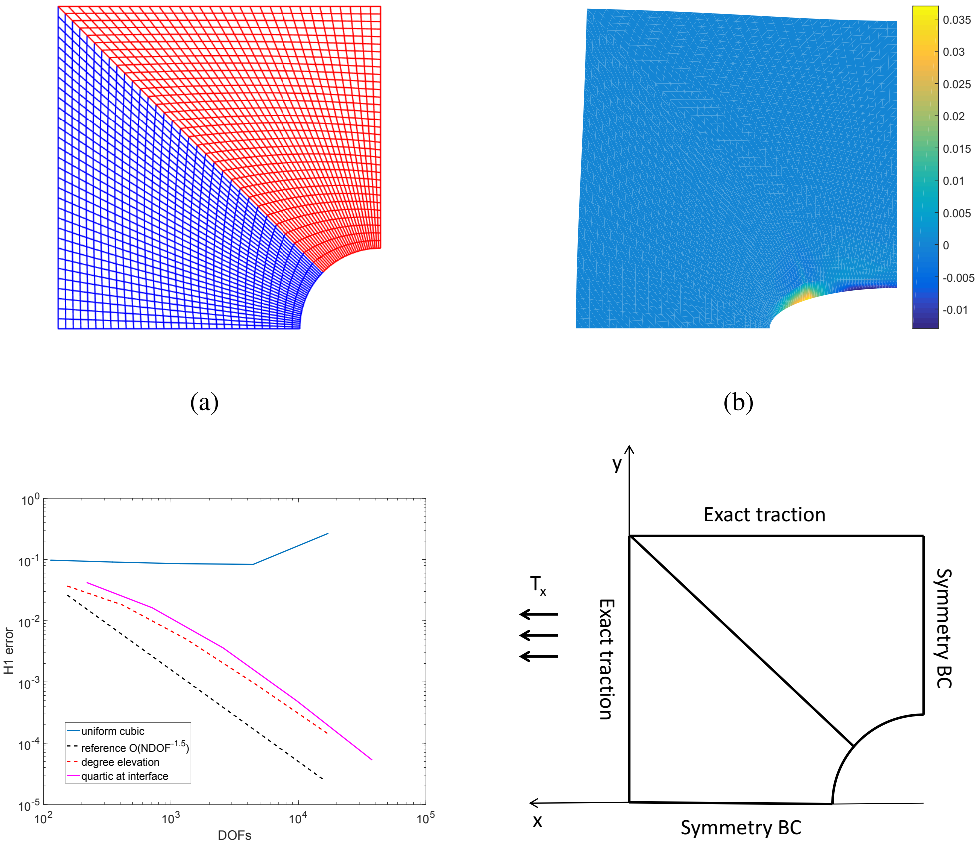
Comparison of actual condition number (CN) and scaled condition number (SCN) of the stiffness matrix are used to analyze the stability of the smooth solution space. Referring to[97,98], the SCN is obtained after applying a simple diagonal preconditioner to the stiffness matrix. The numerical results when the mesh is uniformly refined with up to five levels of refinement are given in Table 5.1. We can observe that the actual condition number is affected by the C^1 coupling and degree elevation, all the three cases have very close SCN. This indicates that the coupling approach provides a good and stable approximation of the solution space.

Table 5.1: Comparison of condition numbers

	C^0 (all cubic)			C^1 (all cubic)			C^1 (degree elevation)		
	DOF	CN	scaled CN	DOF	CN	scaled CN	DOF	CN	scaled CN
1st level	132	2.65E+03	1.5300E+02	114	2.54E+03	1.2900E+02	154	2.13E+03	4.6700E+02
2nd level	380	7.71E+04	4.8400E+02	346	9.38E+04	4.7900E+02	418	5.05E+04	5.9100E+02
3rd level	1260	1.95E+06	3.7600E+03	1194	3.59E+06	3.7700E+03	1330	1.30E+06	3.7800E+03
4th level	4556	4.06E+07	6.1000E+04	4426	1.36E+08	6.1100E+04	4690	2.68E+07	6.1000E+04
5th level	17292	7.52E+08	1.0000E+06	17034	3.86E+09	1.0000E+06	17554	4.96E+08	1.0000E+06

5.8.1.3 L-shaped bracket

In this example, we examine the coupling approach with a more complicated geometry taken from[99]. It is a linear elasticity problem on an L-shaped bracket which contains four holes and a rounded reentrant corner. The bracket is formed by 18 patches, where each patch is composed of 8 × 8 elements as shown in Figure 5.28(a). We apply traction t_N at the hole highlighted with a thick red circle and fix the other two holes marked by

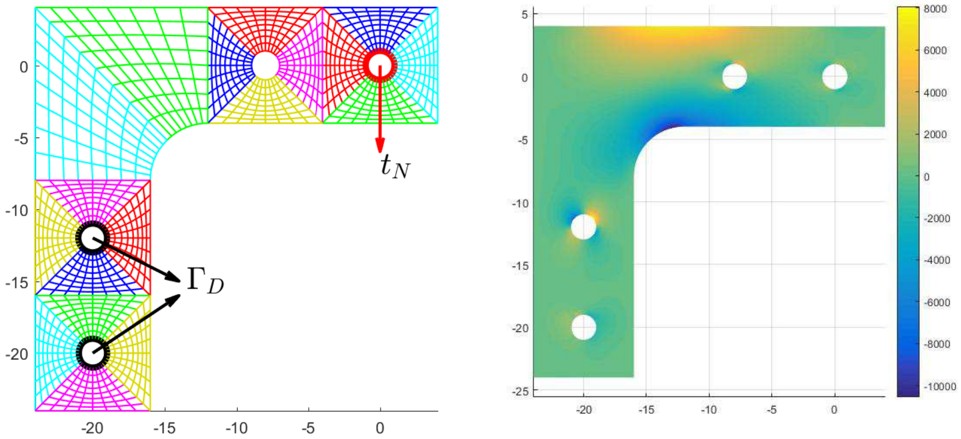


(a) (b)

(c) (d)

Figure 5.27: The infinite plate with a hole example with degree elevation: (a) Plate with hole fine mesh, (b) σ_{yy} component of the stress field, (c) convergence plot, (d) Geometry description and applied boundary conditions.

dark circles. Same material properties given in[99] are used, where Young Modulus $E = 3 \times 10^7$ and Poisson ratio $\nu = 0.3$. The resulting σ_{11} stress distribution is given in Figure 5.28(b); we obtain a very similar result with Figure 9b in[99]. The smooth stress distribution illustrated the efficiency of the C^1 coupling method on the multi-patch domain.



(a) (b)

Figure 5.28: L-shaped bracket: (a) bracket formed by 18 patches and boundary conditions, (b) stress component σ_{11} of (a).

5.8.2 Kirchhoff-Love model

5.8.2.1 Thin Plate

Here, we consider the Kirchhoff-Love Plate subject to clamped boundary conditions. The field equation for this problem is reduced to the biharmonic Equation:

$$D\nabla^4 w = D\nabla^2\nabla^2 w = q,$$

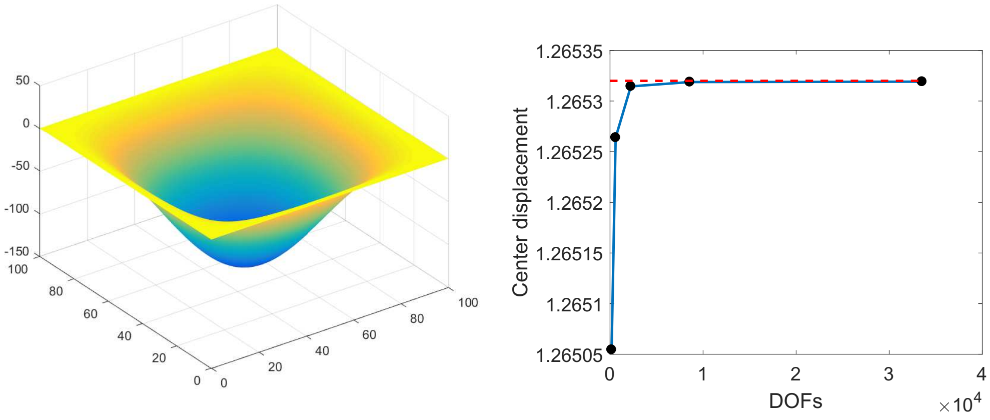
where the biharmonic operator is given by:

$$\nabla^4 = \nabla^2\nabla^2 \equiv \frac{\partial^4}{\partial x^4} + 2\frac{\partial^4}{\partial x^2 \partial y^2} + \frac{\partial^4}{\partial y^4}.$$

In the equations above, q denotes the load, D is the flexural rigidity constant, and $w(x,y)$ represents the plate deflection. In this example, we construct the plate with two patches. The material proprieties Young Modulus, Poisson ratio, and thickness are set to be $E = 10^7$, $\nu = 0.3$, and $t = 1$ respectively. Figure 5.29(a) shows the deflection of the clamped plate, and the convergence plot is given in Figure 5.29(b). In the plot, the red dashes represent the reference displacement, and the blue line shows that the result is converging with successively finer and finer meshes.

5.8.2.2 Pinched cylinder

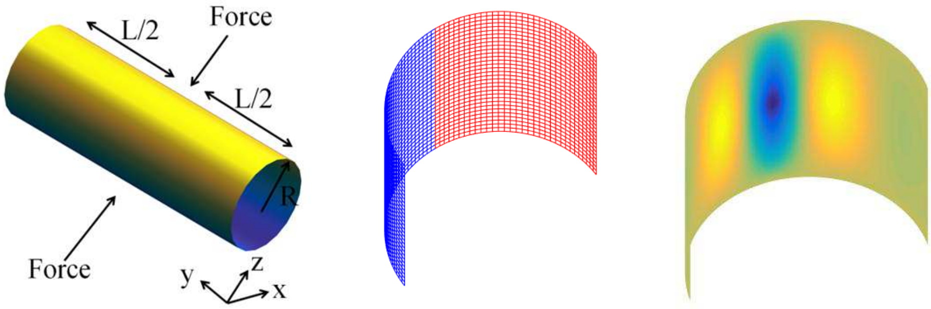
Next, we show another benchmark problem which involves solving a 4th order PDE. It is the pinched cylinder modeled with Kirchhoff-Love shells. In the following is



(a) (b)

Figure 5.29: Kirchhoff-Love plate example: (a) deflection of the plate, and (b) convergence plot.

a brief description of the problem, we refer the reader to[100,101] for more details of the relevant Kirchhoff-Love shell analysis. In this problem, the cylinder is modeled using the elastic material with Young Modulus $E = 3 \times 10^6$, Poisson ratio $\nu = 0.3$ and thickness $t = 3$. Equal forces are applied to the midspan of the cylinder from opposite sides as shown in Figure 5.30(a). We used two patches to form the half cylinder to illustrate the coupling method. The boundary conditions of the cylinder can then be interpreted easily due to the symmetry. The result is given in Figure 5.30(c), we can see that the solution space approximated by the C^1 basis functions can accurately capture the thin shell deformation.



(a) (b) (c)

Figure 5.30: Pinched cylinder example: (a) cylinder subjected to forces from opposite at the midspan, (b) half cylinder mesh in space, (c) displacement.

5.8.3 Cahn-Hilliard phase field model

5.8.3.1 Two-patch square

In[102], a discussion for solving the Cahn-Hilliard equation with basis functions which have C^{p-1} continuity within one patch. Since computation of fourth-order spatial derivatives is required, similar solutions for multi-patch domains require at least C^1 continuity. In this example, we demonstrate the Cahn-Hilliard phase field model in a two-patch domain. We consider a domain Ω which contains a binary mixture. It has a smooth boundary composed of two complementary parts ($\Gamma = \Gamma_g \bar{\cup} \Gamma_s$), and the concentration of one of its components is denoted by k. We refer to[102] for the Cahn-Hilliard equation which governs the evolution of the mixture. Let T be the absolute temperature, the problem is to find $k : \bar{\Omega} \times (0,T) \to \mathbb{R}$ such that:

$$\frac{\partial k}{\partial t} = \nabla \cdot (M_k \nabla(\mu_k - \lambda \Delta k)) \quad \text{in} \quad \Omega \times (0,T)$$

$$\begin{gathered} k = g \quad \text{on} \quad \Gamma_g \times (0,T) \\ M_k \nabla(\mu_k - \lambda \Delta k) \cdot \boldsymbol{n} \quad \text{on} \quad \Gamma_s \times (0,T) \\ M_k \lambda \nabla k \cdot \boldsymbol{n} = 0 \quad \text{on} \quad \Gamma \times (0,T) \\ k(\boldsymbol{x},t) = k_0(\boldsymbol{x}) \quad \text{in} \quad \Omega. \end{gathered}$$

In the above equations, λ is a positive constant such that $\sqrt{\lambda}$ represents the length scale of the problem, and $\boldsymbol{n}$ is the unit outward normal to Γ. Meanwhile, mobility (M_k) and chemical potential (μ_k) are defined as:

$$M_k = Dk(1-k),$$

and

$$\mu_k = \left(\frac{1}{2\theta} \log \frac{k}{1-k} + 1 - 2k \right) 3\alpha.$$

D denotes positive constant which has the dimension of diffusivity, θ is the ratio between critical temperature and absolute temperature, and α represents the dimensionless number related to the thickness of the interface.

The variational formulation for this problem can be stated as: find $k \in \mathcal{V}$ such that $\forall w \in \mathcal{V}$,

$$\left(w, \frac{\partial k}{\partial t} \right)_\Omega + (\nabla w, M_k \nabla \mu_k + \nabla M_k \Delta k)_\Omega + (\Delta w, M_k \Delta k)_\Omega = 0. \tag{5.50}$$

In 5.50, $(\cdot,\cdot)_\Omega$ is the $\mathcal{L}^2$ inner product with respect to Ω, and $\mathcal{V}$ is the space of integrable functions with square integrable first and second derivatives. Following is the solution of two patches with 16×32 elements on each patch. We used the initial condition:

$$k_0(\boldsymbol{x}) = \bar{k} + r,$$

and we set $r = 0.63$, $\alpha = 3000$, and $\theta = 1.5$. The mesh is shown in Figure 5.31(a), and the initial binary mixture is illustrated in Figure 5.31(b). Figures 5.31(c) - 5.31(h) show in ascending order the evolution of the component, and the steady state is given in Figure 5.31(i).

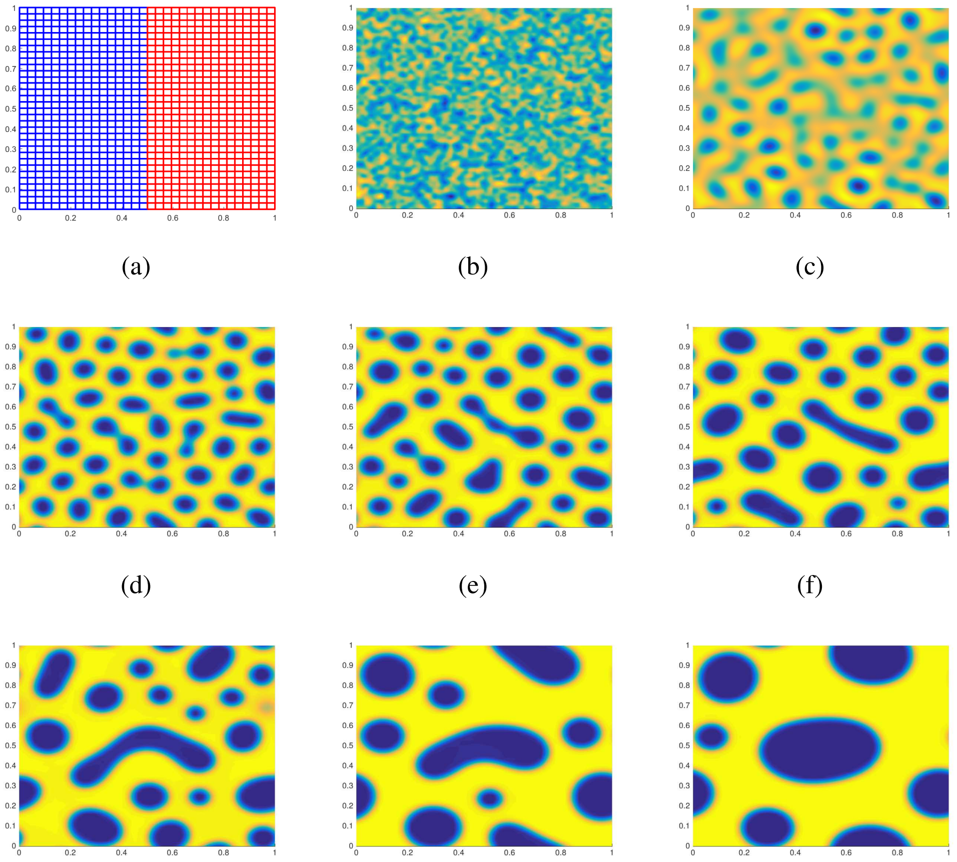


Figure 5.31: Cahn-Hilliard phase field model in a two patch planar domain: (a) two patch mesh, (b)-(i) evolution of the components.

5.8.3.2 Surface of a sphere

Next, we demonstrate the Cahn-Hilliard phase field model on a sphere surface. The sphere is composed of six patches as shown in Figure 5.32(a). The geometry mapping of the surface patches is defined by bi-quartic NURBS, where the details about the parametrization can be found in[103]. The Cahn-Hilliard evolution equation for the surface is similar with to that for 2D. However, on the surface, we use the Laplace-Beltrami operator which is defined as:

$$\Delta w = \frac{1}{\sqrt{\mathcal{G}}}\operatorname{div}\left(\sqrt{\mathcal{G}}\mathcal{G}^{-1}\nabla(W \circ G^{-1})\right). \tag{5.51}$$

The continuity between patches is enforced as described in Section 5.6. In this example, we illustrate the phase field evolution by using 24×24 elements on each patch. The initial distribution of the binary components on the sphere surface is given in Figure 5.32(b), and the evolution of components are given in Figures 5.32(c) - 5.32(f).

5.8.4 Biharmonic problem

The following are the results for the spline approach, where we focus on solving biharmonic problems. We consider the biharmonic problem on a domain Ω:

$$\begin{cases} \Delta^2 u = f & \text{on} \quad \Omega, \\ u = 0 & \text{on} \quad \partial\Omega, \\ \nabla u \cdot \mathbf{n} = 0 & \text{on} \quad \partial\Omega. \end{cases} \tag{5.52}$$

In (5.52), u is the unknown function, and $\mathbf{n}$ is the unit normal vector to the boundary $\partial\Omega$. f is the given right-hand side function. The weak form of (5.52) is given by: Find $u \in H^2(\Omega)$ such that

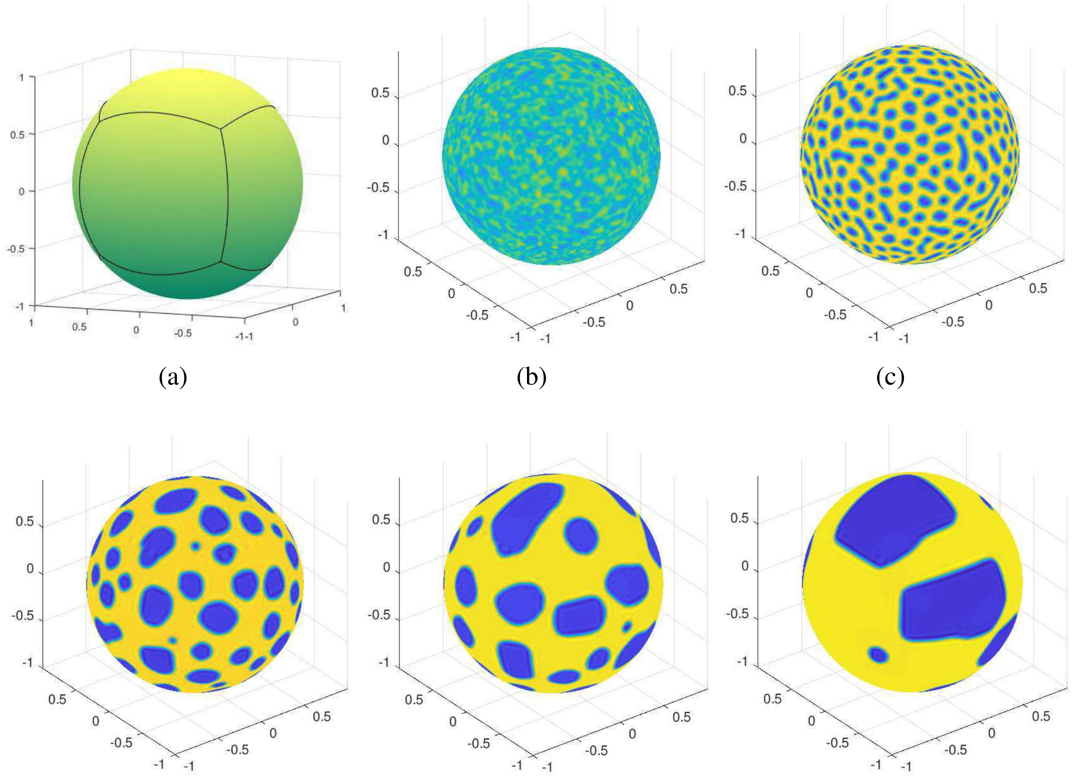
$$\int_\Omega \Delta u \Delta v \, d\Omega = \int_\Omega f v \, d\Omega, \quad \forall v \in H^2(\Omega), \tag{5.53}$$

where v is a test function.

For all the examples, we validate the numerical results by analyzing the L^2 norm convergence plots of the approximated solution. Referring to[104], a prior error estimate for fourth order PDEs is given by:

$$||u - u_h||^2_{L^2(\Omega)} \leq Ch^\beta ||u||_{H^r(\Omega)}, \quad u \in H^r(\Omega_0), \tag{5.54}$$

where $\beta = \min\{p+1, 2p-2\}$ provides convergence order estimate. In (5.54), h, C, and r represent the diameter of the elements, constant, and regularity respectively. More details of the relevant error estimates are given in[105].



(a) (b) (c)

(d) (e) (f)

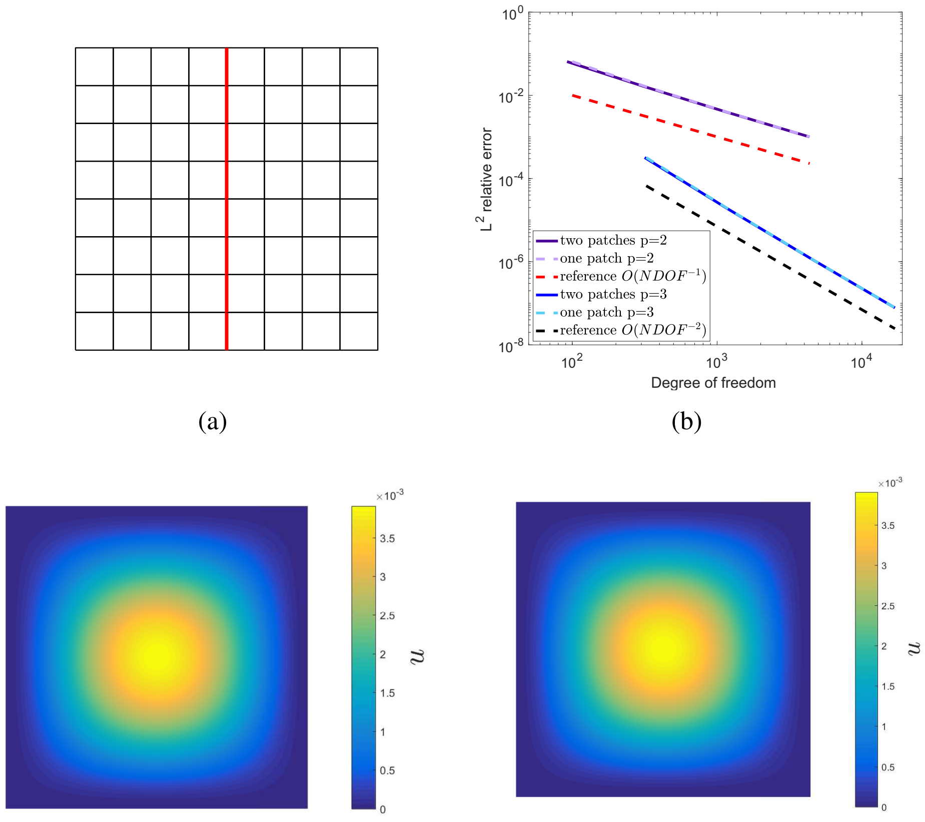
Figure 5.32: Cahn-Hilliard phase field model on the surface of a sphere: (a) sphere composed of six patches (b) initial distribution of binary components, (c)-(d) evolution of components in chronological order.

5.8.4.1 Two-patch square

In the first example, we consider the biharmonic problem on a unit square $\Omega = [0,1]^2$. We choose f such that the exact solution is given by $u_{ext} = x^2y^2(1-x)^2(1-y)^2$. Here, we examine the results between squares formed by two patches (5.33(a)) and one patch. The comparison of convergence curves results are given in Figure 5.33(b). We can observe that for both cases of degrees $p = 2$ and $p = 3$, the two patch domain with C^1 coupling and the one patch domain have very close L^2 relative errors. The almost identical solutions computed with isogeometric functions $p = 3$ for two patches and one patch are shown in Figure 5.33(c) and Figure 5.33(d) respectively.

5.8.4.2 Annulus

Next, we present the biharmonic problem on a more complicated geometry given by an annulus of inner radius 0.2 and outer radius 1. For this example, f is chosen such that



(a) (b)

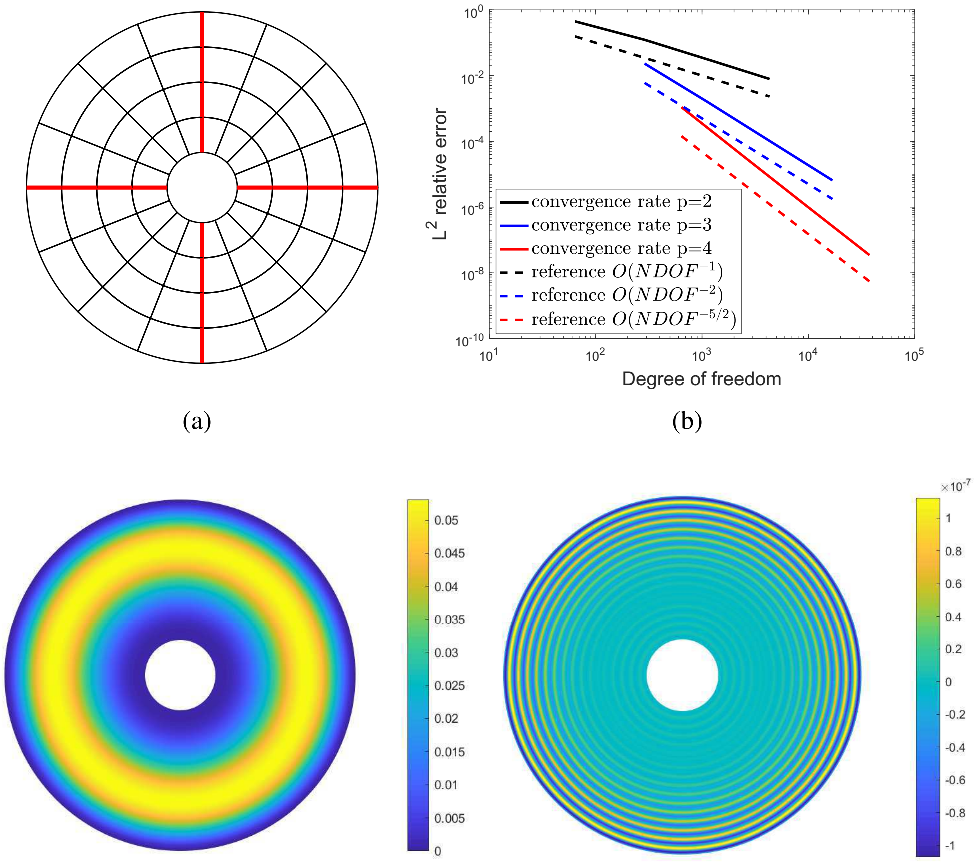
(c) (d)

Figure 5.33: Square domain:(a) square composed of two patches, (b) convergence curve, (c) solution of two patch domain, and (d) solution of one patch.

the analytic solution is given by $u_{ext} = (x^2 + y^2 - 0.04)^2 \cdot (x^2 + y^2 - 1)^2$. The annulus is composed of four patches as shown in Figure 5.34(a). In Figure 5.34(b) shows the convergence plot for solution computed with isogeometric functions of degrees $p = 2, 3$ and 4 and their corresponding reference convergence rate. The solutions for $p = 4$ and the corresponding errors are shown in Figure 5.34(c) and Figure 5.34(d) respectively.

5.8.4.3 Five-patch circle with extraordinary vertices

Here we consider the example with five bi-quadratic patches forming a circle as shown in Figure 5.35(a). In this example, f is chosen such that the exact solution is $u_{ext} = (x^2 + y^2 - 1)^2$. The convergence plots for solution given by isogeometric functions of degree $p = 3$, $p = 4$, and partial degree elevation ($p = 3$ and $p = 4$) are given in Figure 5.35. For the case $p = 3$, which is affected by C^1 locking, the L^2 relative error levels off as the degree of freedom increases (see the dark line). The other two convergence



(a) (b)

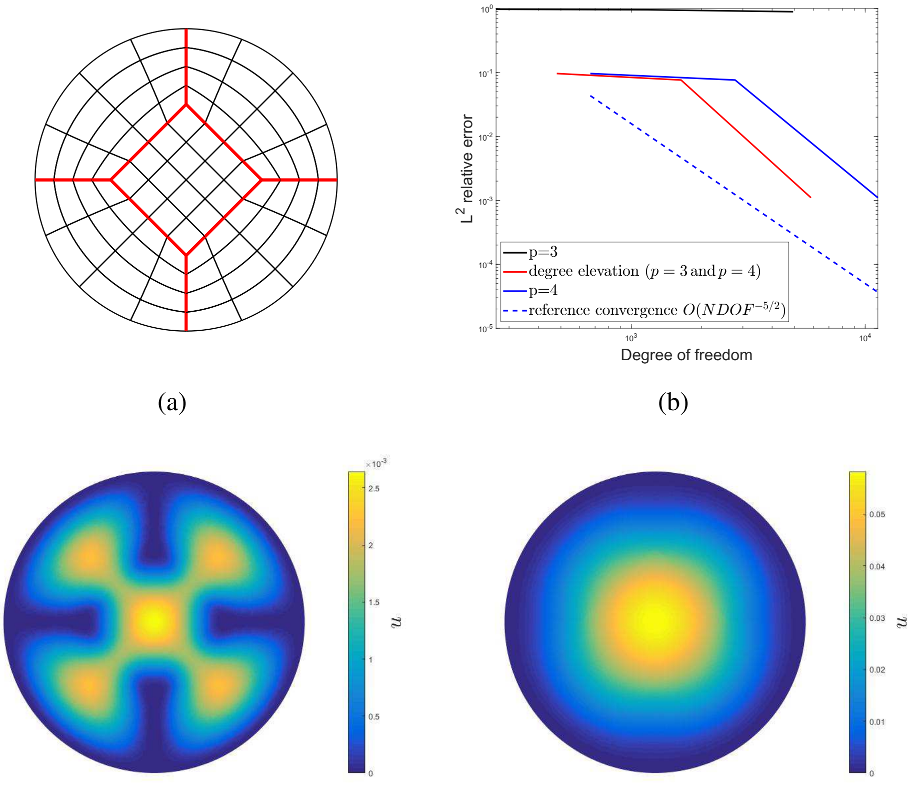
(c) (d)

Figure 5.34: Annulus example (a) annulus composed composed of four patches, (b) convergence curves, (c) solution given by $p = 4$, and (d) error

rates are the results from isogeometric functions of degree $p = 4$ (blue line) and partial degree elevation (red line). We can observe that by using partial degree elevation, the approximation converges faster. Figure 5.35(c) shows the solution affected by C^1 locking computed by using $p = 3$. The improved result given by partial degree elevation is shown in Figure 5.35(d).

5.8.4.4 Two-patch Cube

For the first 3D example, we consider the biharmonic problem on a $[0,1]^3$ cube. The function f is chosen based on the exact solution $u_{ext} = x^2y^2z^2(x-1)^2(y-1)^2(z-1)^2$. In this example, we compare the result between one patch and two patch domains. The former is known to have smooth parametric continuity in nature, while the latter contains globally C^1 basis functions obtained from the C^1 coupling method presented

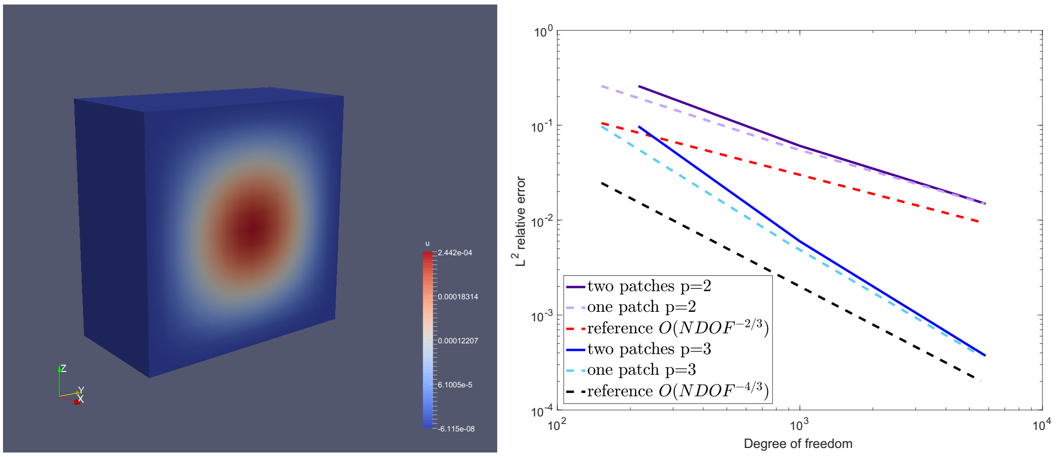


(a) (b)

(c) (d)

Figure 5.35: Circle example (a) circle composed composed of five patches, (b) convergence curves, (c) solution affected by C^1 locking, and (d) solution of partial degree elevation

in previous sections. The convergence plots for the four different settings are given in Figure 5.36(b). The purple line and light purple dashes are the results computed using functions of degree $p = 2$, while the blue line and light blue dashes are the results computed using $p = 3$. The lines represent the convergence rates of one patch and two patch domains respectively as labeled in the figure. Their reference convergence plots are denoted by the red and dark dashes respectively. We can observe that the result from the two patch domain which involves C^1 coupling are as good as the one patch domain. The solution computed on two patch domain using isogeometric functions of degree $p = 3$ is given in Figure 5.36(a).



(a) (b)

Figure 5.36: Cube example (a) solution of two patch domain computed with p=3, and (b) convergence curves.

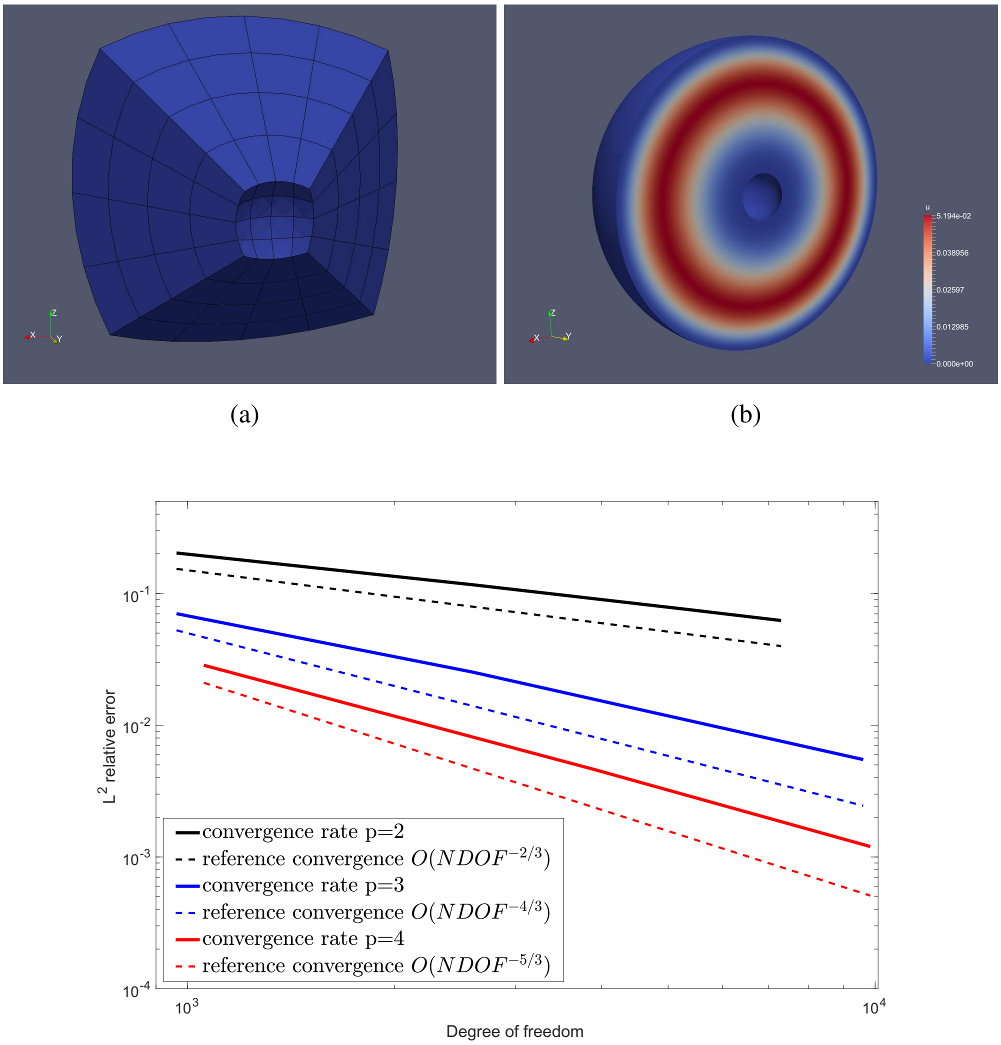
5.8.4.5 Hollow sphere with 6 patches

Finally, we consider the biharmonic problem on a hollow sphere, where the inner radius and outer radius are set as 0.2 and 1 respectively. The hollow sphere is formed by six patches and it contains eight extraordinary vertices; each of them is a common point of three patches. Figure 5.37(a) shows the geometry of one patch which is defined by quartic NURBS, and we obtain the other five patches by rotating it around the z-axis or y-axis. Relevant information about the parametrization of the patches can be found in[103]. For this example, the exact solution is given by $u = (x^2+y^2+z^2-0.04)^2 \times (x^2+y^2+z^2-1)^2$. Here, we compare the solutions computed using isogeometric functions of degree $p=2$ and $p=3$. The convergence results for each case and their corresponding reference rates are given in Figure 5.37(c). Figure 5.37(b) shows the solution computed using $p=3$.

5.9 Conclusions

In this chapter, we present two constructive approaches for C^1 coupling on the multi-patch domain: i. Bézier ordinate approach, and ii. Spline approach. The former approach offers more DOFs, while the latter approach is more efficient and simple to implement. Both approaches are applicable to the construction of C^1 basis functions of degree $p > 2$, while degree $p > 3$ is required for most of the methods discussed in the literature.

We show several examples in the numerical section. For the Bézier ordinate ap-



(a) (b)

(c)

Figure 5.37: Cube example: (a) one patch mes, (b) solution computed with p=3, and (c) convergence curves.

proach, patch test on a multi-patch domain is given to show that linear displacement can be represented exactly. While, in the infinite plate with hole example, comparison of condition numbers is given to shows the stability of the resulting solution space. Other example involving fourth order PDE included the Kirchhoff-Love shell models, Cahn-Hilliard phase field model, and biharmonic problem.

For the comparison of the isogeometric functions obtained from the proposed spline approach with the ordinary basis functions (Sections 5.8.4.1 and 5.8.4.1), we obtain identical results for both cases. The numerical results indicate optimal convergence order. In addition, the efficiency of partial degree elevation to overcome C^1 locking is illustrated in the infinite plate with a hole and circle examples.

Chapter 6

Concluding remarks

The primary objective of this thesis is to develop a self-contained method for obtaining smooth volumetric meshes. Ideally, the meshes need to geometrically approximate the desired shapes well and be suitable for analysis usage. We achieve this objective by first determining the application of B-splines for images representation and processing. We then proceed to develop a method for conversion voxel-based data into an analysis model. Finally, we propose two C^1 coupling approaches on multi-patch domains. Following are the main achievements of each part and some suggestions for future work.

6.1 Essential results

We presented a convolution method for generating the B-spline boundary representation of voxel-based images, where we adopt the filtering technique for efficient computation of B-splines coefficients. We then employ the B-spline convolution and filtering techniques to propose an image registration algorithm. The advantage of the method is that most of the computations are done by using filters which are easy to parallelize. Therefore, the computational time is reduced significantly. Another appealing aspect is that the registration process is conducted within the B-spline framework; both image representation and deformation are represented in terms of B-splines. This allows the deformation to be mapped to the image by using B-splines composition. We showed that our proposed method can produce a smooth transformation map between the images (even for large deformations). We also showed possible medical applications of our method, whereby we applied it for registration of brain images. Compared to the optical flow method, the output from our proposed method shows a better match between the registered image and the target image.

Next, we propose a self-contained method for conversion of a given voxelized image to a matching C^1 volumetric representation that is suitable for analysis. We generate the mesh by using a hierarchical basis of C^1 cubic polynomials. While the continuity of the representation is reduced compared to the highest possible for a cubic basis, the reduced overlap between the basis functions allows for more flexibility in the refinement and boundary representation. An appealing feature of the proposed method is that complex domains can be represented in a single patch framework, for which the continuity is guaranteed. Another key aspect is that we have used the concept of the osculating circle to enhance the geometry approximation. It is done by using a single circle template (sphere template for 3D). Good geometry approximation can be obtained without the need for optimization. The numerical results for benchmark problems show highly accurate results due to the improved geometry approximation. We also show that the method is suitable for complex 2D and 3D domains which contain voids or irregular shapes.

In the last part of this thesis, Bézier ordinates and splines approaches are proposed for the construction of C^1 continuous basis functions on 2D, 3D, and surfaces multi-patch domains. The methods are compatible for domains with general geometries and extraordinary vertices. They allow the modeling of the solution to fourth order PDEs on complex domains, provided that the given patches are G^1 continuous. In the Bézier ordinates approach, the C^1 basis functions are given in terms of Bézier Bernstein polynomial, while in the spline approach they are defined as a linear combination of C^0 basis functions. Linear algebra operations are used to determine the Bézier ordinates and spline coefficients for construction of the C^1 basis functions. Meanwhile, we propose to approach the C^1-locking issues by performing partial degree elevation. This approach offers more flexibility in raising the degree of freedoms (DOFs) only at desirable subdomains, hence, avoiding unnecessary computation costs. Numerical studies are conducted for problems involving fourth order PDEs. We have implemented the Bézier ordinates approaches on surfaces for Kirchhoff-Love shells and Cahn-Hilliard phase field applications. Meanwhile, we apply the splines approach for solving biharmonic problems in 2D and 3D.

6.2 Future work

The difficulty of obtaining accurate analysis or simulation of voxelized data is that some of them have multiple regions which contain different material. It will be useful to have a method that can separate the regions and represent material attributes according to the needs (e.g. finer representation for materials with higher complexity). Therefore, a possible extension of the volumetric parameterization presented in this thesis is a generalization that creates volume meshes for domain enclosed by multiple boundaries. An automatic technique for determining the initial mesh size for the different regions based on the detail of the images is also desirable. Meanwhile, using hierarchical bases that allow for higher continuity in the representation while at the same time preserving sharp features such as corners is also a good direction for improving the existing methods.

For C^1 coupling on the multi-patch domain, one useful improvement is to generate the C^1 basis functions with reduced overlap and smaller supports. This would accelerate the computation tasks, especially in the assembly of the stiffness matrix. Furthermore, constructing C^1 basis functions with properties such as non-negativity, and partition of unity would allow an interactive geometric design. Meanwhile, a related open topic is to find the space of C^1-smooth geometrically continuous isogeometric functions on general domains. A better understanding of it would resolve the "C^1-locking" issue effectively. Also, a method for explicit computation of the C^1 basis functions depending on the geometry of the physical domain would be appealing.

References

[1] N.J. Tustison, B.B. Avants, and J.C. Gee. Directly manipulated free-form deformation image registration. *Image Processing, IEEE Transactions on*, 18(3):624–635, March 2009. viii, 24, 28, 29

[2] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39):4135 – 4195, 2005. 1

[3] Y.J. Zhang. *Geometric Modeling and Mesh Generation from Scanned Images*. Chapman and Hall/CRC, 2016. 1

[4] J. Austin Cottrell, Thomas J. R. Hughes, and Yuri Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. Wiley Publishing, 1st edition, 2009. 5

[5] T. W. Sederberg, J. Zheng, A. Bakenov, and A. Nasri. T-splines and T-NURCCs. *ACM Trans. Graph.*, 22(3):477–484, July 2003. 5

[6] Y. Bazilevs, V.M. Calo, J.A. Cottrell, J.A. Evans, T.J.R. Hughes, S. Lipton, M.A. Scott, and T.W. Sederberg. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(58):229 – 263, 2010. Computational Geometry and Analysis. 5, 68

[7] W. Wang, Y. Zhang, Michael A. Scott, and Thomas J. R. Hughes. Converting an unstructured quadrilateral mesh to a standard T-spline surface. *Computational Mechanics*, 48(4):477–498, 2011. 5

[8] M. R. Dörfel, B. Jüttler, and B. Simeon. Adaptive isogeometric analysis by local h-refinement with T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(58):264 – 275, 2010. Computational Geometry and Analysis. 5, 68

[9] Jiansong Deng, Falai Chen, Xin Li, Changqi Hu, Weihua Tong, Zhouwang Yang, and Yuyu Feng. Polynomial splines over hierarchical T-meshes. *Graphical Models*, 70(4):76 – 86, 2008. 5, 10, 13, 14, 56

[10] M. J. Borden, M. A. Scott, J. A. Evans, and T. J. R. Hughes. Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, 87(1-5):15–47, 2011. 7

[11] N. Nguyen-Thanh, J. Kiendl, H. Nguyen-Xuan, R. Wchner, K.U. Bletzinger, Y. Bazilevs, and T. Rabczuk. Rotation free isogeometric thin shell analysis using PHT-splines. *Computer Methods in Applied Mechanics and Engineering*, 200(4748):3410 – 3424, 2011. 10

[12] N. Nguyen-Thanh, H. Nguyen-Xuan, S.P.A. Bordas, and T. Rabczuk. Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids. *Computer Methods in Applied Mechanics and Engineering*, 200(2122):1892 – 1908, 2011. 10

[13] P. Wang, J. Xu, J. Deng, and F. Chen. Adaptive isogeometric analysis using rational PHT-splines. *Computer-Aided Design*, 43(11):1438 – 1448, 2011. Solid and Physical Modeling 2011. 10

[14] N. Nguyen-Thanh, J. Muthu, X. Zhuang, and T. Rabczuk. An adaptive three-dimensional RHT-splines formulation in linear elasto-statics and elasto-dynamics. *Computational Mechanics*, 53(2):369–385, 2013. 10

[15] Gerald Farin. Curves and surfaces for CAGD. The Morgan Kaufmann Series in Computer Graphics. Morgan Kaufmann, San Francisco, fifth edition, 2002. 10

[16] J. Deng, F. Chen, and Y. Feng. Dimensions of spline spaces over T-meshes. *Journal of Computational and Applied Mathematics*, 194(2):267 – 283, 2006. 12

[17] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12 – 49, 1988. 23

[18] Stanley Osher and Ronald P. Fedkiw. Level set methods: An overview and some recent results. *Journal of Computational Physics*, 169(2):463 – 502, 2001. 23

[19] C. V. Verhoosel, G. J. van Zwieten, B. van Rietbergen, and R. de Borst. Image-based goal-oriented adaptive isogeometric analysis with application to the micro-mechanical modeling of trabecular bone. *Computer Methods in Applied Mechanics and Engineering*, 284:138–164, February 2015. 23, 27, 28, 34

[20] P. Cachier, E. Bardinet, D. Dormont, X. Pennec, and N. Ayache. Iconic feature based nonrigid registration: the PASHA algorithm. *Computer Vision and Image Understanding*, 89(2-3):272 – 298, 2003. Nonrigid Image Registration. 24, 30

[21] Y. Jia, Y. Zhang, and T. Rabczuk. A novel dynamic multilevel technique for image registration. *Computers and Mathematics with Applications*, 69(9):909–925, May 2015. 24

[22] Juelin Leng, Guoliang Xu, and Yongjie Zhang. Medical image interpolation based on multi-resolution registration. *Computers and Mathematics with Applications*, 66(1):1 – 18, 2013. 24

[23] X. Pennec, P. Cachier, and N. Ayache. Understanding the demons algorithm: 3D non-rigid registration by gradient descent. In Chris Taylor and Alain Colchester, editors, *Medical Image Computing and Computer-Assisted Intervention MICCAI99*, volume 1679 of *Lecture Notes in Computer Science*, pages 597–605. Springer Berlin Heidelberg, 1999. 24, 34

[24] J.-P. Thirion. Image matching as a diffusion process: an analogy with Maxwell's demons. *Medical Image Analysis*, 2(3):243 – 260, 1998. 24, 30, 31

[25] B. C. Vemuri, J. Ye, Y. Chen, and C. M. Leonard. Image registration via level-set motion: applications to atlas-based segmentation. *Medical Image Analysis*, 7(1):1 – 20, 2003. 24, 31

[26] A. Pawar, Y. Zhang, Y. Jia, X. Wei, T. Rabczuk, C. L. Chan, and C. Anitescu. Adaptive FEM-based nonrigid image registration using truncated hierarchical B-splines. *A Special Issue of FEF 2015 in Computers and Mathematics with Applications*, 2016. 24

[27] D. Rueckert, P. Aljabar, R. A. Heckemann, J. V. Hajnal, and A. Hammers. Diffeomorphic registration using B-splines. In Rasmus Larsen, Mads Nielsen, and Jon Sporring, editors, *Medical Image Computing and Computer-Assisted Intervention MICCAI 2006*, volume 4191 of *Lecture Notes in Computer Science*, pages 702–709. Springer Berlin Heidelberg, 2006. 24, 25, 33

[28] D. Rueckert, L.I. Sonoda, C. Hayes, D.L.G. Hill, M.O. Leach, and D.J. Hawkes. Nonrigid registration using free-form deformations: application to breast MR images. *Medical Imaging, IEEE Transactions on*, 18(8):712–721, Aug 1999. 24

[29] R. Szeliski and J. Coughlan. Spline-based image registration. *International Journal of Computer Vision*, 22(3):199–218, 1997. 24

[30] C. Davatzikos. Spatial transformation and registration of brain images using elastically deformable models. *Computer Vision and Image Understanding*, 66(2):207 – 222, 1997. 24

[31] J.C. Gee. On matching brain volumes. *Pattern Recognition*, 32(1):99 – 111, 1999. 24

[32] M. F. Beg, M. I. Miller, A. Trouv, and L. Younes. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *International Journal of Computer Vision*, 61(2):139–157, 2005. 24

[33] M. Bro-Nielsen and C. Gramkow. Fast fluid registration of medical images. In Karl Heinz Höhne and Ron Kikinis, editors, *Visualization in Biomedical Computing*, volume 1131 of *Lecture Notes in Computer Science*, pages 265–276. Springer Berlin Heidelberg, 1996. 24

[34] G. E. Christensen, R. D. Rabbitt, and M. I. Miller. Deformable templates using large deformation kinematics. *IEEE Transactions on Image Processing*, 5(10):1435–1447, Oct 1996. 24

[35] T. Vercauteren, X. Pennec, A. Perchant, and N. Ayache. Non-parametric diffeomorphic image registration with the demons algorithm. In N. Ayache, S. Ourselin, and A. Maeder, editors, *Medical Image Computing and Computer-Assisted Intervention MICCAI 2007*, volume 4792 of *Lecture Notes in Computer Science*, pages 319–326. Springer Berlin Heidelberg, 2007. 24, 30, 31

[36] M. Unser. Splines: a perfect fit for signal and image processing. *Signal Processing Magazine, IEEE*, 16(6):22–38, Nov 1999. 27, 28

[37] J. Ashburner. A fast diffeomorphic image registration algorithm. *NeuroImage*, 38(1):95 – 113, 2007. 31

[38] Y. Choi and S. Lee. Injectivity conditions of 2D and 3D uniform cubic B-spline functions. *Graphical Models*, 62(6):411 – 427, 2000. 34

[39] T. Vercauteren, X. Pennec, A. Perchant, and N. Ayache. Diffeomorphic demons: efficient non-parametric image registration. *NeuroImage*, 45(1, Supplement 1):S61 – S72, 2009. Mathematics in Brain Imaging. 39, 41

[40] Diffeomorphic log demons image registration. http://www.mathworks.com/matlabcentral/fileexchange/39194-diffeomorphic-log-demons-image-registration. 39

[41] Brainweb: Simulated brain database. http://brainweb.bic.mni.mcgill.ca/brainweb/. 41, 44

[42] B. Jüttler. *Isogeometric Methods for Numerical Simulation*. Springer Vienna, Vienna, 2015. 47

[43] X. Wang and X. Qian. An optimization approach for constructing trivariate B-spline solids. *Computer-Aided Design*, 46:179 – 191, 2014. 2013 SIAM Conference on Geometric and Physical Modeling. 48

[44] L. Liu, Y. Zhang, T. J. R. Hughes, M. A. Scott, and T. W. Sederberg. Volumetric T-spline construction using Boolean operations. *Engineering with Computers*, 30(4):425–439, 2013. 48

[45] R.N. Simpson, S.P.A. Bordas, J. Trevelyan, and T. Rabczuk. A two-dimensional isogeometric boundary element method for elastostatic analysis. *Computer Methods in Applied Mechanics and Engineering*, 209212:87 – 100, 2012. 48, 50, 61

[46] L. Zhang, A. Gerstenberger, Xiaodong Wang, and Wing Kam Liu. Immersed finite element method. *Computer Methods in Applied Mechanics and Engineering*, 193(2122):2051 – 2067, 2004. Flow Simulation and Modeling. 48

[47] J. Parvizian, A. Düster, and E. Rank. Finite cell method. *Computational Mechanics*, 41(1):121–133, 2007. 48

[48] T. Rabczuk, R. Gracie, J.H. Song, and T. Belytschko. Immersed particle method for fluid-structure interaction. *International Journal for Numerical Methods in Engineering*, 81(1):48–71, 2010. 48

[49] D.J. Yoo. Three-dimensional surface reconstruction of human bone using a B-spline based interpolation approach. *Computer-Aided Design*, 43(8):934 – 947, 2011. 48, 53

[50] H. Klaus, R. Ulrich, and W. Joachim. Weighted extended B-spline approximation of Dirichlet problems. *SIAM Journal on Numerical Analysis*, 39(2):442–462, 2002. 48

[51] R.A.K. Sanches, P.B. Bornemann, and F. Cirak. Immersed b-spline (i-spline) finite element method for geometrically complex domains. *Computer Methods in Applied Mechanics and Engineering*, 200(1316):1432 – 1445, 2011. 48, 53

[52] T. Rabczuk and T. Belytschko. Adaptivity for structured meshfree particle methods in 2D and 3D. *International Journal for Numerical Methods in Engineering*, 63(11):1559–1582, 2005. 48

[53] O. Marco, R. Sevilla, Y. Zhang, J. J. Ródenas, and M. Tur. Exact 3D boundary representation in finite element analysis based on cartesian grids independent of the geometry. *International Journal for Numerical Methods in Engineering*, 103(6):445–468, 2015. 48

[54] G. Xu, B. Mourrain, A. Galligo, and T. Rabczuk. High-quality construction of analysis-suitable trivariate NURBS solids by reparameterization methods. *Comput. Mech.*, 54(5):1303–1313, November 2014. 48

[55] A. Falini, J. Špeh, and B. Jüttler. Planar domain parameterization with THB-splines. *Comput. Aided Geom. Des.*, 35(C):95–108, May 2015. 49

[56] G. Farin and N. Sapidis. Curvature and the fairness of curves and surfaces. *IEEE Comput. Graph. Appl.*, 9(2):52–57, March 1989. 56

[57] A. Mantzaflaris, B. Jüttler, B. N. Khoromskij, and U. Langer. *Curves and Surfaces: 8th International Conference, Paris, France, June 12-18, 2014, Revised Selected Papers*, chapter Matrix Generation in Isogeometric Analysis by Low Rank Tensor Approximation, pages 321–340. Springer International Publishing, Cham, 2015. 61

[58] J. R. Barber. *Elasticity*, chapter Problems in Spherical Coordinates, pages 405–418. Springer Netherlands, Dordrecht, 2010. 63

[59] The stanford 3D scanning repository. http://graphics.stanford.edu/data/3Dscanrep/. 64

[60] J. Kiendl, Y. Bazilevs, M.-C. Hsu, R. Wüchner, and K.-U. Bletzinger. The bending strip method for isogeometric analysis of Kirchhoff-Love shell structures comprised of multiple patches. *Computer Methods in Applied Mechanics and Engineering*, 199(37):2403 – 2416, 2010. 67

[61] Robert Schmidt, Roland Wüchner, and Kai-Uwe Bletzinger. Isogeometric analysis of trimmed NURBS geometries. *Computer Methods in Applied Mechanics and Engineering*, 241-244(Supplement C):93 – 111, 2012. 67

[62] Ericka Brivadis, Annalisa Buffa, Barbara Wohlmuth, and Linus Wunderlich. Isogeometric mortar methods. *Computer Methods in Applied Mechanics and Engineering*, 284(Supplement C):292 – 319, 2015. Isogeometric Analysis Special Issue. 67

[63] W. Dornisch, G. Vitucci, and S. Klinkel. The weak substitution method – an application of the mortar method for patch coupling in NURBS-based isogeometric analysis. *International Journal for Numerical Methods in Engineering*, 103(3):205–234, 2015. 67

[64] Christian Hesch and Peter Betsch. Isogeometric analysis and domain decomposition methods. *Computer Methods in Applied Mechanics and Engineering*, 213-216(Supplement C):104 – 112, 2012. 67

[65] Yujie Guo and Martin Ruess. Nitsche's method for a coupling of isogeometric thin shells and blended shell structures. *Computer Methods in Applied Mechanics and Engineering*, 284(Supplement C):881 – 905, 2015. Isogeometric Analysis Special Issue. 67

[66] Andreas Apostolatos, Robert Schmidt, Roland Wüchner, and Kai-Uwe Bletzinger. A Nitsche-type formulation and comparison of the most common domain decomposition methods in isogeometric analysis. *International Journal for Numerical Methods in Engineering*, 97(7):473–504, 2014. 67

[67] V. P. Nguyen, P. Kerfriden, S. P.A. Bordas, and T. Rabczuk. Isogeometric analysis suitable trivariate NURBS representation of composite panels with a new offset algorithm. *Computer-Aided Design*, 55:49 – 63, 2014. 67

[68] N. Nguyen-Thanh, K. Zhou, X. Zhuang, P. Areias, H. Nguyen-Xuan, Y. Bazilevs, and T. Rabczuk. Isogeometric analysis of large-deformation thin shells using RHT-splines for multiple-patch coupling. *Computer Methods in Applied Mechanics and Engineering*, 316(Supplement C):1157 – 1178, 2017. Special Issue on Isogeometric Analysis: Progress and Challenges. 67

[69] Mario Kapl, Vito Vitrih, Bert Jüttler, and Katharina Birner. Isogeometric analysis with geometrically continuous functions on two-patch geometries. *Computers & Mathematics with Applications*, 70(7):1518 – 1538, 2015. High-Order Finite Element and Isogeometric Methods. 68

[70] Mario Kapl, Florian Buchegger, Michel Bercovier, and Bert Jttler. Isogeometric analysis with geometrically continuous functions on planar multi-patch geometries. *Computer Methods in Applied Mechanics and Engineering*, 316:209 – 234, 2017. Special Issue on Isogeometric Analysis: Progress and Challenges. 68

[71] Mario Kapl and Vito Vitrih. Space of C^2-smooth geometrically continuous isogeometric functions on two-patch geometries. *Computers & Mathematics with Applications*, 73(1):37 – 59, 2017. 68, 94

[72] Mario Kapl and Vito Vitrih. Space of C^2-smooth geometrically continuous isogeometric functions on planar multi-patch geometries: Dimension and numerical experiments. *Computers & Mathematics with Applications*, 73(10):2319 – 2338, 2017. 68

[73] Annabelle Collin, Giancarlo Sangalli, and Thomas Takacs. Analysis-suitable G^1 multi-patch parametrizations for C^1 isogeometric spaces. *Computer Aided Geometric Design*, 47(Supplement C):93 – 113, 2016. SI: New Developments Geometry. 68, 97

[74] Mario Kapl, Giancarlo Sangalli, and Thomas Takacs. Dimension and basis construction for analysis-suitable G^1 two-patch parameterizations. *Computer Aided Geometric Design*, 52-53(Supplement C):75 – 89, 2017. Geometric Modeling and Processing 2017. 68

[75] Thien Nguyen and Jörg Peters. Refinable C^1 spline elements for irregular quad layout. *Computer Aided Geometric Design*, 43:123 – 130, 2016. Geometric Modeling and Processing 2016. 68

[76] Deepesh Toshniwal, Hendrik Speleers, and Thomas J.R. Hughes. Smooth cubic spline spaces on unstructured quadrilateral meshes with particular emphasis on extraordinary points: Geometric design and isogeometric analysis considerations. *Computer Methods in Applied Mechanics and Engineering*, 327:411 – 458, 2017. 68

[77] Kęstutis Karčiauskas, Thien Nguyen, and Jörg Peters. Generalizing bicubic splines for modeling and IGA with irregular layout. *Computer-Aided Design*, 70:23 – 35, 2016. SPM 2015. 68

[78] Deepesh Toshniwal, Hendrik Speleers, René R. Hiemstra, and Thomas J.R. Hughes. Multi-degree smooth polar splines: A framework for geometric modeling and isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 316:1005–1061, 2017. 68

[79] Bernard Mourrain, Raimundas Vidunas, and Nelly Villamizar. Dimension and bases for geometrically continuous splines on surfaces of arbitrary topology. *Computer Aided Geometric Design*, 45:108 – 133, 2016. 68

[80] Meng Wu, Bernard Mourrain, André Galligo, and Boniface Nkonga. Hermite type spline spaces over rectangular meshes with complex topological structures. *Communications in Computational Physics*, 21:835 – 866, 2017. 68

[81] Michel Bercovier and Tanya Matskewich. *Smooth Bézier Surfaces over Unstructured Quadrilateral Meshes*. Springer International Publishing, 01 2017. 68, 69, 94

[82] Fehmi Cirak, Michael Ortiz, and Peter Schröder. Subdivision surfaces: a new paradigm for thin-shell finite-element analysis. *International Journal for Numerical Methods in Engineering*, 47(12):2039–2072, 2000. 68

[83] Xiaodong Wei, Yongjie Jessica Zhang, Thomas J.R. Hughes, and Michael A. Scott. Extended Truncated Hierarchical Catmull-Clark Subdivision. *Computer Methods in Applied Mechanics and Engineering*, 299:316 – 336, 2016. 68

[84] J.M. Escobar, J.M. Cascón, E. Rodríguez, and R. Montenegro. A new approach to solid modeling with trivariate T-splines based on mesh optimization. *Computer Methods in Applied Mechanics and Engineering*, 200(45):3210 – 3222, 2011. 68

[85] Yongjie Zhang, Wenyan Wang, and Thomas J.R. Hughes. Solid T-spline construction from boundary representations for genus-zero geometry. *Computer Methods in Applied Mechanics and Engineering*, 249-252:185 – 197, 2012. Higher Order Finite Element and Isogeometric Methods. 68

[86] Luke Engvall and John A. Evans. Isogeometric unstructured tetrahedral and mixed-element Bernstein-Bézier discretizations. *Computer Methods in Applied Mechanics and Engineering*, 319:83 – 123, 2017. 68

[87] Songtao Xia and Xiaoping Qian. Isogeometric analysis with Bézier tetrahedra. *Computer Methods in Applied Mechanics and Engineering*, 316:782 – 816, 2017. Special Issue on Isogeometric Analysis: Progress and Challenges. 68

[88] M. Majeed and F. Cirak. Isogeometric analysis using manifold-based smooth basis functions. *Computer Methods in Applied Mechanics and Engineering*, 316(Supplement C):547 – 567, 2017. Special Issue on Isogeometric Analysis: Progress and Challenges. 69

[89] Luca Dedè and Alfio Quarteroni. Isogeometric analysis for second order partial differential equations on surfaces. *Computer Methods in Applied Mechanics and Engineering*, 284(Supplement C):807 – 834, 2015. Isogeometric Analysis Special Issue. 69

[90] Thien Nguyen, Kestutis Karciauskas, and Jörg Peters. C^1 finite elements on non-tensor-product 2d and 3d manifolds. *Applied Mathematics and Computation*, 272, Part 1:148 – 158, 2016. 69

[91] Mario Kapl, Giancarlo Sangalli, and Thomas Takacs. Dimension and basis construction for analysis-suitable G^1 two-patch parameterizations. *Computer Aided Geometric Design*, 5253:75 – 89, 2017. Geometric Modeling and Processing 2017International Conference on Geometric Modeling and Processing (GMP 2017). 78

[92] Thomas F Coleman and Alex Pothen. The null space problem i. complexity. *SIAM J. Algebraic Discrete Methods*, 7(4):527–537, October 1986. 94

[93] M.A. Scott, R.N. Simpson, J.A. Evans, S. Lipton, S.P.A. Bordas, T.J.R. Hughes, and T.W. Sederberg. Isogeometric boundary element analysis using unstructured T-splines. *Computer Methods in Applied Mechanics and Engineering*, 254(Supplement C):197 – 221, 2013. 97

[94] Xiaodong Wei, Yongjie Zhang, Lei Liu, and Thomas J.R. Hughes. Truncated T-splines: Fundamentals and methods. *Computer Methods in Applied Mechanics and Engineering*, 316:349 – 372, 2017. 97

[95] Rainer Kraft. Hierarchical B-splines. *Mathematisches Institut A, Universitt Stuttgart*, 1995. 97

[96] Bert Jttler, Angelos Mantzaflaris, Ricardo Perl, and Martin Rumpf. On numerical integration in isogeometric subdivision methods for PDEs on surfaces. *Computer Methods in Applied Mechanics and Engineering*, 302:131 – 146, 2016. 101

[97] I. Babuška and U. Banerjee. Stable generalized finite element method (SGFEM). *Computer Methods in Applied Mechanics and Engineering*, 201-204:91 – 111, 2012. 106

[98] Kenan Kergrene, Ivo Babuška, and Uday Banerjee. Stable generalized finite element method and associated iterative schemes; application to interface problems. *Computer Methods in Applied Mechanics and Engineering*, 305:1 – 36, 2016. 106

[99] S. K. Kleiss, B. Jüttler, and W. Zulehner. Enhancing isogeometric analysis by a finite element-based local refinement strategy. *Computer Methods in Applied Mechanics and Engineering*, 213216:168 – 182, 2012. 106, 107

[100] Josef Kiendl, Kai-Uwe Bletzinger, J Linhard, and Roland Wüchner. Isogeometric shell analysis with Kirchhoff-Love elements. *Computer Methods in Applied Mechanics and Engineering*, 198:3902–3914, 11 2009. 109

[101] N. Nguyen-Thanh, N. Valizadeh, M.N. Nguyen, H. Nguyen-Xuan, X. Zhuang, P. Areias, G. Zi, Y. Bazilevs, L. De Lorenzis, and T. Rabczuk. An extended isogeometric thin shell analysis based on Kirchhoff-Love theory. *Computer Methods in Applied Mechanics and Engineering*, 284(Supplement C):265 – 291, 2015. Isogeometric Analysis Special Issue. 109

[102] Héctor Gómez, Victor M. Calo, Yuri Bazilevs, and Thomas J.R. Hughes. Isogeometric analysis of the Cahn-Hilliard phase-field model. *Computer Methods in Applied Mechanics and Engineering*, 197(49):4333 – 4352, 2008. 110

[103] James E. Cobb. Tiling the sphere with rational Bézier patches. Technical report, University of Utah, The College of Engineering, School of Computing, 1988. 112, 117

[104] Florian Maurin, Francesco Greco, Sander Dedoncker, and Wim Desmet. Isogeometric analysis for nonlinear planar Kirchhoff rods: Weighted residual formulation and collocation of the strong form. *Computer Methods in Applied Mechanics and Engineering*, 340:1023 – 1043, 2018. 112

[105] Anna Tagliabue, Luca Ded, and Alfio Quarteroni. Isogeometric analysis and error estimates for high order partial differential equations in fluid dynamics. *Computers and Fluids*, 102:277–303, 2014. 112

Academic Curriculum Vitae

Chan, Chiu Ling

Institute of Structural Mechanics
Bauhaus-Universität Weimar
Marienstrasse 15, 99423 Weimar, Germany
Email: chanchiuling@gmail.com

Education

- Ph.D student: Institute of Structural Mechanics, Bauhaus-Universität Weimar, Germany, 2014- 2019.
- M.Sc: Master of Mathematics (Computer Aided Geometric Design), Universiti Sains Malaysia, Penang, Malaysia, 2009-2012.
- B.Sc: Bachelor of Computational Mathematics, Universiti Malaysia Terengganu,Terengganu, Malaysia, 2006-2009.

Publications

1. Chan, C.L., Anitescu, C., and Rabczuk, T. *Strong multipatch C1-coupling for Isogeometric analysis on 2D and 3D domains*. Computer Methods in Applied Mechanics and Engineering, (357), 2019.
2. Chan, C.L., Anitescu, C., and Rabczuk, T. *Isogeometric analysis with strong multipatch C1-coupling*. Computer Aided Geometric Design, (62):294-310, 2018.
3. Chan, C.L., Anitescu, C., and Rabczuk, T. *Volumetric parametrization from a level set boundary representation with PHT-splines*. Computer-Aided Design, (82):29-41, 2017.
4. Chan, C.L., Anitescu, C., Zhang, Y., and Rabczuk, T. *Two and Three Dimensional Image Registration Based on B-Spline Composition and Level Sets*. Communications in Computational Physics, 21(2):600-622, 2017.

5. Chan, C.L. and Ali, J.M. *Low energy transition curve.* International Conference on Software Technology and Engineering, 3rd (ICSTE 2011), Othman, M, Kasim. R, ASME, NY, 2011.
6. Chan, C.L., Abbas. M. and Ali, J.M. *Approximating GCS with low energy Hermite curve.* European Journal of Scientific Research, 46(4):660-626, 2010.

Conference Presentations

1. *Strong C^1 coupling of multi-patch 3D PHT-spline,* Presented in the 14^{th} U.S. National Congress on Computational Mechanics (USNCCM14), July 17-20, Montŕeal Québec, Canada.
2. *Isogeometrix analyis with strong multi-patch c1-coupling,* Presented in the International Conference on Geometric Modeling and Processing (GMP2018), April 09-11, Aachen, Germany.
3. *2D and 3D multi-patch strong C^1 coupling,* Presented in the USACM thematic conference on Isogeometric Analysis (IGA 2018), October 10-12, Austin, Texas, U.S.