Integrating geometric modeling and structure analysis: towards the digital future of engineering

Author:
Zhang, Junqi

Publication Date:
2019

DOI:
https://doi.org/10.26190/unsworks/21348

Integrating Geometric Modeling and Structure Analysis: Towards the Digital Future of Engineering

by

Junqi Zhang

A thesis submitted in fulfilment
of the requirements for the degree of
Doctor of Philosophy

UNSW
AUSTRALIA

School of Civil and Environmental Engineering
The University of New South Wales
Sydney, Australia

April, 2019

UNSW SYDNEY | Australia's Global University

Thesis/Dissertation Sheet

Surname/Family Name	:	**ZHANG**
Given Name/s	:	**JUNQI**
Abbreviation for degree as give in the University calendar	:	**PhD**
Faculty	:	**Engineering**
School	:	**School of Civil and Environmental Engineering**
Thesis Title	:	**Integrating Geometric Modeling and Structure Analysis: Towards the Digital Future of Engineering**

Abstract 350 words maximum: (PLEASE TYPE)

Numerical method has revolutionized the engineering. With the rapid development of modern computers, the numerical method will play a more important role in the future of engineering. However, a large amount of human effort is still required in an analysis to convert a geometric model to a numerical model which can be accepted by computers. This manual process is time consuming and error prone, especially in an adaptive analysis where the mesh needs to be updated frequently. Additional challenges are imposed by new data formats used in industry, such as stereolithography (STL) models, virtual reality (VRML) models and digital image.

In this thesis, a numerical framework is developed to link geometric modeling and structure analysis automatically based on the scaled boundary finite element method, which is capable of modeling polyhedrons with arbitrary number of faces and edges by discretizing their boundaries only. This framework accepts multiple data formats as input. High quality polyhedron meshes can be generated using octree algorithm with minimum human interventions. The meshes for different parts of a model can be generated independently and merged automatically by modifying the surface meshes on their interfaces only. No interface constraints or special shape functions are required in the analysis. Composite materials with inclusions such as fibers and anchors can also be modeled. Adaptive mesh refinement can be implemented based on a posterior error estimation. The meshes can be modified locally due to the advantage of the polyhedron elements.

A virtual city analysis is performed using this framework, where a numerical model of a city block is constructed to simulate the ground settlement, the air flow between buildings and the effect of earthquakes. Large scale modeling and analysis are possible because of the automation and efficiency of the proposed method. The meshes generated for individual buildings, underground structures and terrain in the virtual city are reusable.

The numerical framework proposed in this thesis has been successfully applied by other researchers in acoustic, contact mechanics, fluid-structure interaction and many others.

FOR OFFICE USE ONLY Date of completion of requirements for Award:

ORIGINALITY STATEMENT

'I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.'

Signed …….....

Date …….....

INCLUSION OF PUBLICATIONS STATEMENT

UNSW is supportive of candidates publishing their research results during their candidature as detailed in the UNSW Thesis Examination Procedure.

Publications can be used in their thesis in lieu of a Chapter if:

- The student contributed greater than 50% of the content in the publication and is the "primary author", ie. the student was responsible primarily for the planning, execution and preparation of the work for publication
- The student has approval to include the publication in their thesis in lieu of a Chapter from their supervisor and Postgraduate Coordinator.
- The publication is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in the thesis

Please indicate whether this thesis contains published material or not.

☐ *This thesis contains no publications, either published or submitted for publication (if this box is checked, you may delete all the material on page 2)*

☒ *Some of the work described in this thesis has been published and it has been documented in the relevant Chapters with acknowledgement (if this box is checked, you may delete all the material on page 2)*

☐ *This thesis has publications (either published or submitted for publication) incorporated into it in lieu of a chapter and the details are presented below*

CANDIDATE'S DECLARATION

I declare that:

- I have complied with the Thesis Examination Procedure
- where I have used a publication in lieu of a Chapter, the listed publication(s) below meet(s) the requirements to be included in the thesis.

Name	Signature	Date (dd/mm/yy)
JUNQI ZHANG		**1/4/2019**

Postgraduate Coordinator's Declaration (to be filled in where publications are used in lieu of Chapters)

I declare that:

- the information below is accurate
- where listed publication(s) have been used in lieu of Chapter(s), their use complies with the Thesis Examination Procedure
- the minimum requirements for the format of the thesis have been met.

PGC's Name	PGC's Signature	Date (dd/mm/yy)

For each publication incorporated into the thesis in lieu of a Chapter, provide all of the requested details and signatures required

Details of publication #1: *Full title:* *Authors:* *Journal or book name:* *Volume/page numbers:* *Date accepted/ published:*						
Status	*Published*		*Accepted and In press*		*In progress (submitted)*	
The Candidate's Contribution to the Work Insert text describing how the candidate has contributed to the work						
Location of the work in the thesis and/or how the work is incorporated in the thesis: Insert text						
Primary Supervisor's Declaration I declare that: • the information above is accurate • this has been discussed with the PGC and it is agreed that this publication can be included in this thesis in lieu of a Chapter • All of the co-authors of the publication have reviewed the above information and have agreed to its veracity by signing a 'Co-Author Authorisation' form.						
Supervisor's name		*Supervisor's signature*		*Date (dd/mm/yy)*		

Details of publication #2: *Full title:* *Authors:* *Journal or book name:* *Volume/page numbers:* *Date accepted/ published:*						
Status	*Published*		*Accepted and In press*		*In progress (submitted)*	
The Candidate's Contribution to the Work Insert text describing how the candidate has contributed to the work						
Location of the work in the thesis and/or how the work is incorporated in the thesis: Insert text						
Primary Supervisor's Declaration I declare that: • the information above is accurate • this has been discussed with the PGC and it is agreed that this publication can be included in this thesis in lieu of a Chapter • All of the co-authors of the publication have reviewed the above information and have agreed to its veracity by signing a 'Co-Author Authorisation' form.						
Supervisor's name		*Supervisor's signature*		*Date (dd/mm/yy)*		

Add further boxes if required. You may delete any boxes not used

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisor, professor Chongmin Song, for his continuous advice and support during the past three and half years. His carefulness, preciseness and creativity in science have greatly inspired me to strive for excellence in research. His kindness and patience have encouraged me to move forward in the most difficult situations. I have learnt so many things from him, both in research and in my life. Without his support, guidance and help, this dissertation would not be possible.

My sincere thanks go to my co-supervisor, professor Arnaud Castel who always provides valuable advice on the research. I would like to thank Dr Yan Liu, Dr Albert Saputra and Dr Lei Liu for their continuous help and collaboration in the research. It is my great honor to work with them.

I would like to acknowledge the financial support I received from the University of New South Wales, the University International Postgraduate Award (UIPA), which allows me to undertake my PhD study.

Thank all my friends for all the memories we share.

A special thank goes to my parents for their endless encouragements and support throughout my life. All my accomplishments would not be possible without their love, care and understanding throughout the years. Words cannot express how grateful I am to them.

Abstract

Numerical method has revolutionized the engineering. With the rapid development of modern computers, the numerical method will play a more important role in the future of engineering. However, a large amount of human effort is still required in an analysis to convert a geometric model to a numerical model which can be accepted by computers. This manual process is time consuming and error prone, especially in an adaptive analysis where the mesh needs to be updated frequently. Additional challenges are imposed by new data formats used in industry, such as stereolithography (STL) models, virtual reality (VRML) models and digital image.

In this thesis, a numerical framework is developed to link geometric modeling and structure analysis automatically based on the scaled boundary finite element method, which is capable of modeling polyhedrons with arbitrary number of faces and edges by discretizing their boundaries only. This framework accepts multiple data formats as input. High quality polyhedron meshes can be generated using octree algorithm with minimum human interventions. The meshes for different parts of a model can be generated independently and merged automatically by modifying the surface meshes on their interfaces only. No interface constraints or special shape functions are required in the analysis. Composite materials with inclusions such as fibers and anchors can also be modeled. Adaptive mesh refinement can be implemented based on a posterior error estimation. The meshes can be modified locally due to the advantage of the polyhedron elements.

A virtual city analysis is performed using this framework, where a numerical model of a city block is constructed to simulate the ground settlement, the air flow between buildings and the effect of earthquakes. Large scale modeling and analysis are possible because of the automation and efficiency of the proposed method. The meshes generated for individual buildings, underground structures and terrain in the virtual city are reusable.

The numerical framework proposed in this thesis has been successfully applied by other researchers in acoustic, contact mechanics, fluid-structure interaction and many others.

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

$\ddot{u}$ acceleration

κ curvature

$[\Phi]$ eigenvector matrix

$\left[\Phi_b^{(q)}\right]$ force mode

$\left[\Phi_b^{(u)}\right]$ displacement mode

$[B_1]$, $[B_2]$ strain-displacement matrices

$[E_0]$, $[E_1]$, $[E_2]$ coefficient matrices

$[J]$ Jacobian matrix

$[J_b]$ Jacobian matrix on the boundary

$[K]$ element stiffness matrix

$[K_G]$ global stiffness matrix

$[M]$ mass matrix

$[N_V^e]$ shape function of S-element

$[Z_p]$ Hamiltonian coefficient matrix

$\langle\lambda\rangle$ eigenvalue matrix

$\{c\}$ integration constants

$\{d_G\}$ global nodal displacement vector

$\{F_G\}$ global nodal force vector

$\{q(\xi)\}$ nodal force on radial lines

$\{u(\xi)\}$ nodal displacement on radial lines

$\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ unit vectors in Cartesian coordinates

$\mathbf{r}$ position vector

ν Poisson's ratio

σ stress

ε strain

Γ surface mesh

Ω volume mesh

ξ, η, ζ scaled boundary coordinates

c cell

E Young's modulus

e edge

f face

n node

$N(\eta, \zeta)$ shape function of parametric element

o octant

u displacement

x, y, z, Cartisen coordinates

Chapter 1

Introduction

Engineering has revolutionized the world. With the rapid development of engineering in the last century, we are now able to build aircraft traveling faster than sound; We can construct buildings which can withstand the most destructive earthquakes; We can convert wind and sunlight into limitless energy; We can cure diseases to save hundreds of thousands of lives. All these achievements cannot be done without a deep understanding of the principles behind the phenomena and a reliable calculation using the proper mathematical tool. These engineering principles are derived from the fundamental laws and principles of nature, such as the conservation of mass, conservation of energy, equilibrium of force, etc. The mathematical tool has evolved from paper and pencil to modern computers.

The development of modern computers has greatly changed the engineering design process. Before the advent of computers, engineering design was mainly based on hand drafting, which is time consuming and inaccurate. The modification of the draft was tedious as engineering design is an iterative process to create a product to meet a stated objective. It was also difficult to visualize the objects in 3D space. Usually a prototype was produced for realistic evaluation before the design is finalized. With the help of modern computers, engineers are now able to design more complex models

with less human effort. Computer aided design (CAD) has become the mainstream in the engineering design in the past two decades. The CAD system provides a powerful tool for fast and accurate geometric modeling. The time spend on modification of the models in each iteration of engineering design is greatly reduced. The communication between different groups of engineers is also much easier in the modern CAD system.

There are new techniques which further facilitate the engineering design. The virtual reality technique provides an immersive environment for engineers to interact with the models. The 3D printing technique enables a fast and affordable prototyping of the designed model. Computed tomography (CT) provides an approach for non-destructive modeling of the interior structures. These techniques are developed together with new data formats, such as stereolethography model, virtual reality model and digital image. It can be foreseen that dramatic changes will happen in engineering in the digital age.

1.1 Structure analysis in engineering

One of the most important topics in the engineering design is the structure analysis. Structural analysis is the determination of the effects of loads on physical structures and their components, such as displacement, stress, velocity, etc. In engineering design it requires these effects to satisfy certain objectives, e.g. the maximum displacement is less than a tolerance. Otherwise, the design needs to be modified until the objectives are satisfied. The structure analysis has a wide range of applications in civil, mechanical, aerospace and marine engineering. The problems in structural analysis are usually represented by ordinary differential equations (ODE) and partial differential equations (PDE).

Consider an infinitesimal cube surrounding a point within an isotropic elastic material (Moaveni, 2011) (Fig. 1.1).

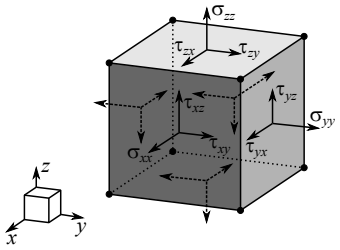


Figure 1.1: The stress components at a point

The general state of stress at the point is defined by

$$[\boldsymbol{\sigma}] = \begin{bmatrix} \sigma_{xx} & \sigma_{yy} & \sigma_{zz} & \tau_{yz} & \tau_{zx} & \tau_{xy} \end{bmatrix}^T, \tag{1.1}$$

where σ_{xx}, σ_{yy} and σ_{zz} are the normal stresses and τ_{yz}, τ_{zx} and τ_{xy} are the shear stresses. The state of strain at the point is defined by

$$[\boldsymbol{\varepsilon}] = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{yy} & \varepsilon_{zz} & \gamma_{yz} & \gamma_{zx} & \gamma_{xy} \end{bmatrix}^T, \tag{1.2}$$

where ε_{xx}, ε_{yy} and ε_{zz} are the normal strains and γ_{yz}, γ_{zx} and γ_{xy} are the shear strains. The relationships between strain components and the displacement components are

$$\varepsilon_{xx} = \frac{\partial u_x}{\partial x}, \tag{1.3a}$$

$$\varepsilon_{yy} = \frac{\partial u_y}{\partial y}, \tag{1.3b}$$

$$\varepsilon_{zz} = \frac{\partial u_z}{\partial z}, \tag{1.3c}$$

$$\gamma_{yz} = \frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y}, \tag{1.3d}$$

$$\gamma_{zx} = \frac{\partial u_z}{\partial x} + \frac{\partial u_x}{\partial z}, \tag{1.3e}$$

$$\gamma_{xy} = \frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}, \tag{1.3f}$$

where u_x, u_y and u_z are the displacement components in x, y and z directions. According

to generalized Hooke's law, the relationships between the stresses and strains are

$$\varepsilon_{xx} = \frac{1}{E}\left[\sigma_{xx} - \nu\left(\sigma_{yy} + \sigma_{zz}\right)\right], \tag{1.4a}$$

$$\varepsilon_{yy} = \frac{1}{E}\left[\sigma_{yy} - \nu\left(\sigma_{xx} + \sigma_{zz}\right)\right], \tag{1.4b}$$

$$\varepsilon_{zz} = \frac{1}{E}\left[\sigma_{zz} - \nu\left(\sigma_{xx} + \sigma_{yy}\right)\right], \tag{1.4c}$$

$$\gamma_{yz} = \frac{1}{G}\tau_{yz}, \tag{1.4d}$$

$$\gamma_{zx} = \frac{1}{G}\tau_{zx}, \tag{1.4e}$$

$$\gamma_{xy} = \frac{1}{G}\tau_{xy}, \tag{1.4f}$$

where E is Young's modulus, ν is Poisson's ratio and G is the shear modulus. The equilibrium equations in 3D are

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} + f_x = 0, \tag{1.5a}$$

$$\frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} + f_y = 0, \tag{1.5b}$$

$$\frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} + f_z = 0, \tag{1.5c}$$

where f_x, f_y and f_z are the body loads. Substituting Eq. (1.3) and Eq. (1.4) into Eq. (1.5), the governing differential equations in 3D elasticity are obtained as

$$\frac{E}{2\left(1+\nu\right)}\left(\frac{1}{1-2\nu}\frac{\partial \theta}{\partial x} + \nabla^2 u_x\right) + f_x = 0, \tag{1.6a}$$

$$\frac{E}{2\left(1+\nu\right)}\left(\frac{1}{1-2\nu}\frac{\partial \theta}{\partial y} + \nabla^2 u_y\right) + f_y = 0, \tag{1.6b}$$

$$\frac{E}{2\left(1+\nu\right)}\left(\frac{1}{1-2\nu}\frac{\partial \theta}{\partial z} + \nabla^2 u_z\right) + f_z = 0, \tag{1.6c}$$

where θ is the volumetric strain

$$\theta = \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}, \tag{1.7}$$

and ∇^2 is the Laplace operator in 3D

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}. \tag{1.8}$$

Similarly, the governing differential equations in 2D plane stress problems are (Gould and Feng, 1994)

$$\frac{E}{1-\nu^2}\left(\frac{\partial^2 u_x}{\partial x^2} + \frac{1-\nu}{2}\frac{\partial^2 u_x}{\partial y^2} + \frac{1+\nu}{2}\frac{\partial^2 u_y}{\partial x \partial y}\right) + f_x = 0, \tag{1.9a}$$

$$\frac{E}{1-\nu^2}\left(\frac{\partial^2 u_y}{\partial y^2} + \frac{1-\nu}{2}\frac{\partial^2 u_y}{\partial x^2} + \frac{1+\nu}{2}\frac{\partial^2 u_x}{\partial x \partial y}\right) + f_y = 0. \tag{1.9b}$$

The governing differential equation in 1D problem is

$$E\frac{\mathrm{d}^2 u_x}{\mathrm{d}x^2} + f_x = 0. \tag{1.10}$$

However, the integration of PDEs mentioned above is usually difficult, especially in 2D and 3D problems. The difficulty is mainly caused by the complexity of the geometry and the boundary conditions. Sometimes the material is anisotropic and heterogeneous, which further increases the difficulty. Therefore, analytical solution is usually not available in engineering practice. Numerical method can be employed to provide an approximate solution.

1.2 Numerical method in structure analysis

Numerical method has been widely used in engineering as the analytical solutions of PDEs are usually difficult to obtain. A large number of numerical methods have been developed. In this section, a brief overview is provided on the procedure to construct an approximate solution. A review of the state-of-the-art of the numerical methods will be discussed in Section 2.

The general form of a PDE is expressed as (Finlayson and Scriven, 1966; Finlayson, 2013)

$$\mathcal{L}u = g, \tag{1.11}$$

where $\mathcal{L}$ is a differential operator, u is the analytical solution and g is another function. As the analytical solution is usually not available, an approximate solution $\bar{u}$ is introduced. $\bar{u}$ can be constructed as a linear combination of n basis functions

$$\bar{u} = u_1\phi_1 + u_2\phi_2 + \ldots + u_n\phi_n, \tag{1.12}$$

where $\phi_1, \phi_2, \ldots, \phi_n$ are the basis functions and $u_1, u_2, \ldots, u_n$ are the corresponding coefficients. There are many different approaches to construct basis functions (Babuska and Melenk, 1997), for example, a set of piecewise linear functions. The approximate solution in Eq. (1.12) cannot satisfy Eq. (1.11) exactly in the whole problem domain. An error function, or residual, is introduced

$$\mathcal{L}\bar{u} = g + R. \tag{1.13}$$

The goal is to determine the coefficients $u_1, u_2, \ldots, u_n$ to minimize the residual $R = \mathcal{L}\bar{u} - g$. In the weighted residual method, the residual R is multiplied by a weighting function w, and the integration of the product over the whole problem domain is set to zero

$$\int (\mathcal{L}\bar{u} - g)\, w = 0, \tag{1.14}$$

to formulated a system of equations to determine the coefficients $u_1, u_2, \ldots, u_n$.

Different selections of basis functions and weighting functions will form different numerical methods. Only some important methods are introduced in the following of this section. A detailed comparison of different weighting functions can be found

in Finlayson and Scriven (1966).

1.2.1 Collocation method

In the collocation method, the weighting functions are a set of displaced Dirac delta functions at selected points (Malik et al., 1985; Babuska et al., 2007). These points are usually, but not necessarily, evenly distributed in the problem domain.

$$w\left(x_i\right) = \delta\left(x - x_i\right) \qquad \text{for } i = 1, 2, \ldots, n, \tag{1.15}$$

where n is the number of unknowns in the approximate solution $\bar{u}$. The value of a Dirac delta function is infinity at a specific point and zero at all the other locations. The integration in Eq. (1.14) becomes the value of function at some specific points

$$\mathcal{L}u\left(\bar{x}_i\right) - g = 0 \qquad \text{for } i = 1, 2, \ldots, n. \tag{1.16}$$

Therefore, the problem of solving of PDEs is converted to solving a set of linear algebra equations.

In this method, as the differential operator is applied on the approximate solution, it is required that the basis function be differentiable. The finite difference method, which will be discussed in Section 2, is closely related to the collocation method.

1.2.2 Subdomain method

Another well-known type of the weighted residual method is the subdomain method. The problem domain is divided into several smaller subdomains. The weighting function is set to be a constant over each subdomain. The integration over a subdomain is

zero (Canu and Ray, 1991)

$$\int_{x_i}^{x_{i+1}} (\mathcal{L}\bar{u} - g) = 0 \qquad \text{for } i = 1, 2, \ldots, n. \tag{1.17}$$

The number of subdomains n is equal to the number of unknowns in the approximate solution $\bar{u}$.

1.2.3 Galerkin method

In the Galerkin method, the weighting function w is chosen as the basis function itself (Fletcher, 1984; Yu and Heinrich, 1987)

$$w_i = \phi_i \qquad \text{for } i = 1, 2, \ldots, n. \tag{1.18}$$

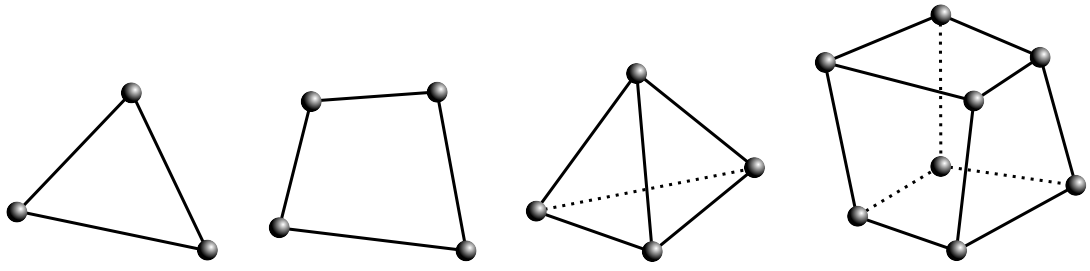
The basis functions are also called the shape functions. The integration in Eq. (1.14) becomes

$$\int \phi_i \, (\mathcal{L}\bar{u} - g) = 0 \qquad \text{for } i = 1, 2, \ldots, n. \tag{1.19}$$

Using the same functions for the weighting and approximating functions usually produces symmetric coefficients (Brebbia et al., 2012). The finite element method can be classified as Galerkin method (Segerlind, 1976; Thomee, 1984).

1.3 A brief introduction to finite element method

The finite element method (FEM) is one of the most important numerical methods used in structure analysis. The details of FEM will be presented in Section 2.1. In FEM, a geometric model is discretized into numerous small parts with simple geometry. This discretization process is called mesh generation and the small parts are called

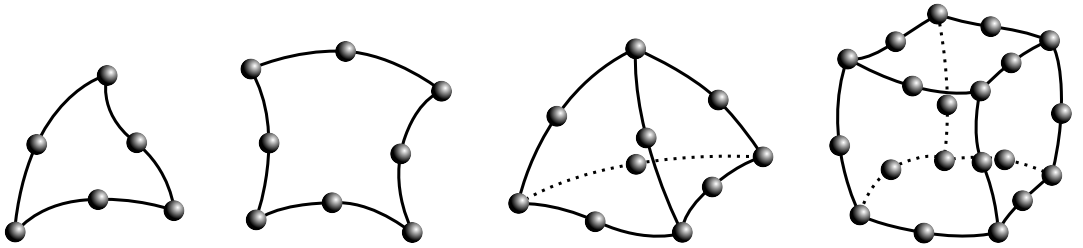


(a) Linear triangle element

(b) Linear quadrilateral element

(c) Linear tetrahedron element

(d) Linear hexahedron element



(e) Quadratic triangle element

(f) Quadratic quadrilateral element

(g) Quadratic tetrahedron element

(h) Quadratic hexahedron element

Figure 1.2: Typical elements in FEM

Figure removed due to copyright restriction (available in master copy)

(a) CAD model of a propeller

Figure removed due to copyright restriction (available in master copy)

(b) Volume mesh of the propeller

Figure 1.3: CAD model of a propeller and its volume mesh

elements (Thompson et al., 1985). The typical elements in a finite element analysis include triangle (Fig. 1.2a and Fig. 1.2e) and quadrilateral (Fig. 1.2b and Fig. 1.2f) in 2D, tetrahedron (Fig. 1.2c and Fig. 1.2g) and hexahedron (Fig. 1.2d and Fig. 1.2h) in 3D. A structure and its finite element mesh are shown in Fig. 1.3 (Geuzaine and Remacle, 2009). An approximate solution is constructed by interpolating node values in the mesh. The continuous displacement field in Eq. (1.6) is approximated by a discrete displacement field. The governing equations are converted to solving a set of linear algebra equations, which is suitable for modern computers to process.

The FEM offers greater flexibility in modeling of structures with complex geometries.

The boundary conditions can be applied easily. A large amount of commercial software based on the FEM has been developed, such as ANSYS, ABAQUS, NASTRAN and many others.

1.4 Challenges in finite element method for integrating geometry with analysis

Although the numerical methods have largely reduced the difficulty in solving PDEs, there are still significant challenges faced in engineering practice. In engineering practice the geometries of the structures are usually complex and irregular. The analysis often needs to be performed iteratively in a design process. The modifications of the models are frequently required. These challenges impede the integrating of geometry and analysis, which causes tedious human intervention to convert geometric models to numerical models. Therefore, the design process based on the current FEM is not compatible with the fast development of the modern computers and the emerging engineering techniques. Some of the main difficulties in the numerical methods, especially the FEM, are addressed in this section.

1.4.1 Finite element mesh generation

Mesh generation is a nontrivial task in the current FEM (Guersoy, 1996; Demargne et al., 2014; Ghisi et al., 2014; Bols et al., 2016; Botella et al., 2016). Quadrilateral and hexahedron elements are usually preferable to triangle and tetrahedron elements as they produce more accurate results in analysis. However, no automatic approach is available for all hexahedron mesh generation presently. Therefore, the geometric model needs to be manually partitioned into several parts with simpler geometries before these parts can be meshed. This task takes a large amount of time in the whole analysis procedure. Extensive experience and skill are required to successfully generate a high

quality mesh (Hughes et al., 2005). What's more, it is inevitable for human to make mistakes, which may jeopardize the reliability of the result of analysis. For example, inappropriate partition can result in degenerated or invalid elements. A detailed review on the mesh generation techniques will be presented in Section 2.2.

The difficulty in the mesh generation is exacerbated by several other factors, which will be discussed in the following of this section.

1.4.1.1 Different data formats of geometric models

The geometric models can be represented by the volume or the boundary. In a boundary representation, the analytical or discrete form of the boundary of the model is expressed using an explicit or implicit function. In a volume representation, the space is partitioned into a uniform grid of points. The information of whether a point is inside the model is stored. Conventional finite element analysis is based on non-uniform rational B-spline (NURBS) models, which is the standard format in the computer aided design (CAD). NURBS is able to represent the geometry accurately. However, high quality NURBS model is not always available or suitable to the purpose. New techniques in engineering are emerging with new data formats, such as stereolethography (STL) models in 3D printing, Virtual reality modeling language (VRML and X3D) in computer vision and gaming, digital image in CT scan, just to name a few. These new data formats are different from the conventional NURBS model, which imposes additional challenges for FEM mesh generation.

NURBS model NURBS is a boundary representation widely used in CAD. NURBS curve in 2D is defined by a set of weighted control points $\{\boldsymbol{P}\}$ and a knot vector Ξ. The knot vector Ξ in is a set of coordinates in the parametric space, which can be written as $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\}$, where ξ_i is the i-th knot. p is the polynomial order and n

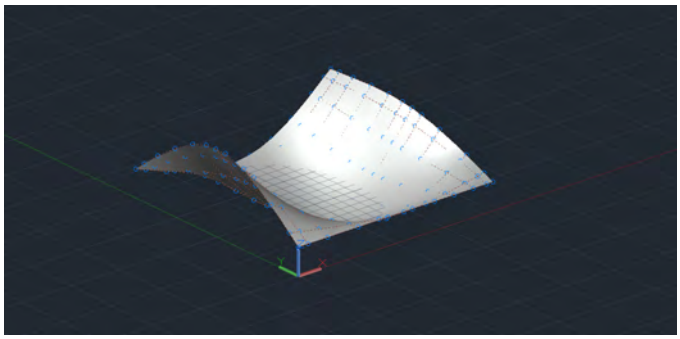


Figure 1.4: Example of NURBS surface

is the number of basis functions. A NURBS curve $C(\xi)$ can be written as

$$C(\xi) = \frac{\sum_{i=1}^{k} N_{i,n} w_i \boldsymbol{P}_i}{\sum_{i=1}^{k} N_{i,n} w_i}, \tag{1.20}$$

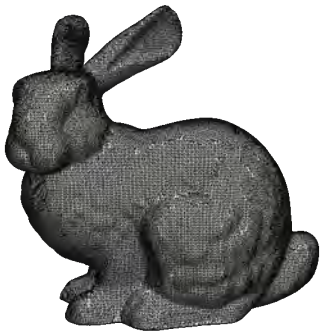
where ξ is a dimensionless coordinate in the parametric space, k is the number of control points, $\boldsymbol{P}_i$ is the coordinate of the i-th control point and w_i is the corresponding weight. The n-th order basis function $N_{i,n}$ is constructed as

$$N_{i,n} = \frac{\xi - \xi_i}{\xi_{i+n} - \xi_i} N_{i,n-1} + \frac{\xi_{i+n} - \xi}{\xi_{i+n} - \xi_i} N_{i+1,n-1}, \tag{1.21}$$

where the 0-th order basis functions are piecewise constant functions

$$N_{i,0} = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases}$$

NURBS surfaces in 3D can be constructed similarly using a matrix of control points and two dimensional basis functions. An example of NURBS surface and its control points are shown in Fig. 1.4.



(a) STL surface mesh

face normal: n_x, n_y, n_z
  outer loop
    vertex: x_1, y_1, z_1
    vertex: x_2, y_2, z_2
    vertex: x_3, y_3, z_3
  endloop
endfacet

(b) STL data format

Figure 1.5: STL model of a rabbit

Figure removed due to
copyright restriction
(available in master copy)

Figure 1.6: Example of VRML model

STL model STL is a commonly used data format in 3D printing. A sample of STL model is shown in Fig. 1.5a (`http://graphics.stanford.edu/data/3Dscanrep/`). An STL model is simple to read and write as it only contains a group of individual triangles. Only the node coordinates and normal vectors of each triangle are stored in the model, which is indicated in Fig. 1.5b. The qualities of the triangles in an STL model can be extremely poor from the view point of numerical analysis. The STL model may not be manifold, i.e. there may be self-intersection and holes in the model. Therefore, an STL model is usually not suitable for direct use in numerical modeling (Bechet et al., 2002).

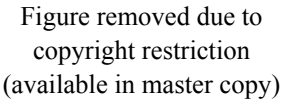
VRML model VRML (Virtual Reality Modeling Language) is usually used in computer science to render models in virtual reality environment. It is similar to STL model but it contains more types of built-in objects such as polygon, cylinder and sphere. There are also some other features like lighting and shading information. VRML and STL models can be easily converted to each other. An example of VRML model is shown in Fig. 1.6 (`https://steemit.com/`).

Figure 1.7: Digital image obtained from X-ray scan of human brain

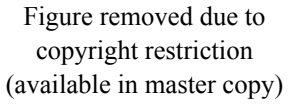
Digital image Digital image is another important data format, which is a volume representation of the model. Digital image is usually obtained from computed tomography (CT) scan using X-ray (Du Plessis et al., 2016) or neutron radiography (Dewanckele et al., 2014). The digital image is represented by pixels in 2D and voxels in 3D. A 2D black and white image is shown in Fig. 1.7 (`https://www.radiologyinfo.org/`). Each pixel/voxel is assigned with a brightness value ranging from 0 to 255. Different materials are represented by different brightness values. Digital image is especially useful when the model is irregular and not suitable for NURBS modeling, such as structures with irregular cracks, inclusions and voids. Some image segregation and processing techniques have been developed to reconstruct 3D surfaces (Avizo, 2015).

1.4.1.2 Modeling of moving boundary problems

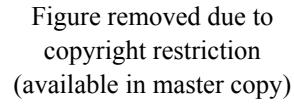
The modeling of moving boundary problems imposes additional difficulty in finite element mesh generation. Moving boundary problems are ubiquitous in engineering, for example, crack propagation (Fig. 1.8a) (`http://firmalaar.com/Altan-Usta-Kocaeli`), projectile penetration (Fig. 1.8b) (`https://www.carolina.com/teacher-resources/Interactive`) and phase transition (Fig. 1.8c) (`https://bigvinnysd.wordpress.com/tag/science/`). In these problems, the geometry of the domain is changing constantly, which makes it difficult to generate finite element meshes if manual operation is required. Therefore, a highly automatic mesh generation algorithm is necessary for the investigation of moving boundary problems.



(a) Crack propagation

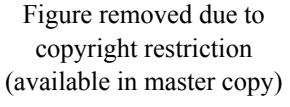


(b) Projectile penetration

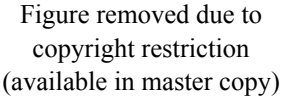


(c) Phase transition

Figure 1.8: Examples of moving boundary problems



(a) Fiber reinforced concrete



(b) Soil nail reinforced slope

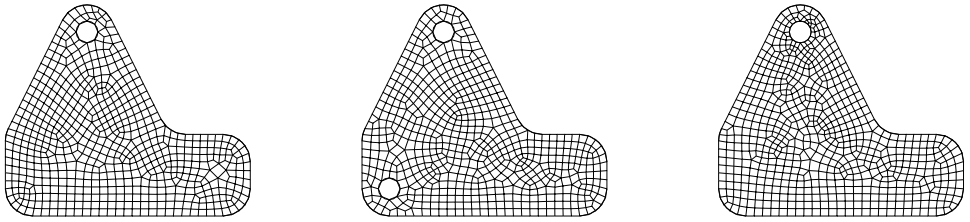
Figure 1.9: Examples of fiber reinforced materials

1.4.1.3 Mesh generation for fiber reinforced materials

Fiber reinforced materials have been widely used in engineering due to their excellent mechanical and thermal properties. Some commonly used fibers include steel fiber (Fig. 1.9a) (`https://www.indiamart.com/`), glass fiber, carbon fiber. Anchors and soil nails in geotechnical engineering can also be considered as fibers in terms of mechanical behavior (Fig. 1.9b) (`https://www.roads.maryland.gov/`). In FEM the fibers are generally modeled using 1D tension bar elements or beam elements. The matrix material is usually modeled using solid element in 2D and 3D. The mesh of the matrix usually needs to be aligned with the fibers in the conventional FEM. However, this requirement increases the difficulty in the mesh generation because the distribution and orientation of the fibers are generally random in engineering practice. In some cases only triangle/tetrahedron meshes can be generated for the matrix, not to mention the quality of these meshes is usually poor.

1.4.2 Local modification of finite element mesh

A finite element mesh needs to maintain a valid connectivity so that the requirement of displacement continuity is satisfied. In conventional FEM it is usually difficult to



(a) Quadrilateral mesh of a mechanical part

(b) Modification of the geometry

(c) Modification of the mesh size

Figure 1.10: Remeshing caused by local modification

modify the mesh locally without remeshing the whole structure. Even the modification of a small region may affects the whole mesh if quadrilateral or hexahedron elements are used. A quadrilateral mesh is generated for a mechanical part in Fig. 1.10a. The geometry is modified by adding another hole near the lower left corner in Fig. 1.10b. It is observed that the whole mesh is modified. Another case is shown in Fig. 1.10c, where the mesh in Fig. 1.10a is refined near the hole. This modification also affects the whole mesh.

In engineering design, sometimes several iterations are required to modify the geometric model based on the result of numerical analysis. Most of the times the modification is only in a small part compared with the whole model. Therefore, it is not efficient to remesh the whole model in every iteration. It is preferable that the mesh can be modified locally without affecting the other regions, while the element connectivity can be maintained.

1.4.3 Treatment of non-matching meshes

In the numerical analysis of complex and large scale models, it is desirable to divide the problem domain into several subdomains with simpler geometries. The difficulty in mesh generation can be reduced by meshing the subdomains individually. Parallel processing can be employed for the mesh generation of the subdomains to further accelerate the whole analysis procedure. It is also preferable to model the subdomains

individually in a variety of other applications such as contact mechanics, fluid-structure interaction, multi-scale analysis.

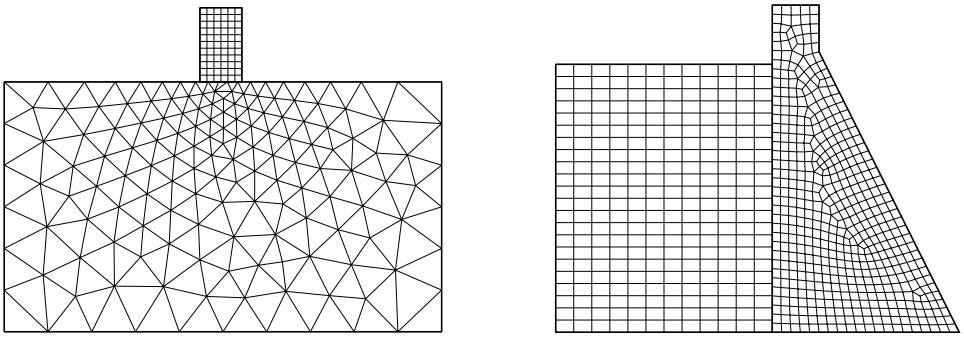
The meshes of those subdomains are only connected by their interfaces. The interface meshes are matching if the mesh topologies and element shape functions are the same on both sides of the interface, otherwise they are non-matching. Examples of non-matching meshes in 2D are shown in Fig. 1.11. Obviously, the use of non-matching meshes offer greater flexibility in practice as they enable users to generate meshes for different subdomains independently. These independent meshes need to be coupled (or "glued") together when a numerical analysis is performed.

However, the coupling of non-matching meshes is a challenging task. Some coupling techniques rely on manual modification of the meshes (Staten et al., 2010b), which is time consuming and not suitable for large scale problems. The nearest neighbor interpolation (de Boer et al., 2007) is automatic but it often leads to deterioration of accuracy. In mortar element method, a set of mortar elements are inserted between the two non-matching meshes (Maday et al., 1988). The displacements in the two meshes and the mortar elements need to minimize the total potential energy of the system. However, the solving of this problem usually introduces additional degrees of freedom or involves parameters which are difficult to choose. Alternatively, non-conventional elements can be employed to connect the non-matching meshes (Lim et al., 2007). The details of these techniques will be summarized in Section 2.3.

1.5 Objectives

The objective of this research is to develop a numerical framework to link geometric modeling and structure analysis. This numerical framework will have the following features:

1. Multiple data formats can be accepted. The proposed numerical framework will



(a) Non-matching meshes of building and soil

(b) Non-matching meshes of dam and reservoir

Figure 1.11: Non-matching meshes

accept multiple data formats, including both boundary representation (such as NURBS, STL and VRML model) and volume representation (such as digital image). It is preferable that even the models containing small defects (such as self-intersections and holes) can be accepted without manual repairing operation.

2. The mesh generation is fully automatic. It is well-known that human intervention required in the mesh generation is time consuming and error prone. The goal of this research is to develop a mesh generator which only requires minimum human effort, such as defining the maximum and minimum element size of the mesh. The rest of the task will be handled by the computer automatically. The mesh quality obtained should be good enough for numerical analysis. The mesh generator should be able to check the quality of the elements and perform optimization when necessary.

3. The mesh can be modified locally. In conventional FEM, the modification of a part of the mesh often affects other regions, causing remeshing of the whole model. In the proposed framework, the modification will be limited to the local region only. The vast majority of the elements in the original mesh do not need to be changed. Therefore the time and effort can be spent effectively on the regions where necessary. The modifications include changing the local geometry

and modifying element size and shape.

4. Non-matching meshes can be coupled in an analysis. A complex model can be spatially partitioned into several parts with simpler geometries. The meshes for those parts can be generated independently without considering the compatibility on the interface. The difficulty associated with mesh generation can be further reduced. The non-matching meshes will be converted to matching meshes automatically using polytope elements formulated in Section 3. It is preferable that the mesh quality on the interface can be maintained and smooth mesh size transition can be obtained.

5. Adaptive analysis procedure can be implemented. A posterior error indicator will be developed. An efficient mesh refinement technique will be established. The analysis performed with minimum remeshing starts from a coarse mesh. The mesh is refined iteratively in the regions with high estimated errors. The remeshing of the whole structure is not required. The computational resources can be used efficiently.

6. Numerical simulation of virtual reality models can be performed. This numerical framework will be applied to large scale virtual reality models, e.g. a city block. Meshes of different parts of the city can be generated independently and these meshes are reusable when local modifications are performed.

7. Fiber reinforced materials can be modeled considering random fiber distribution. A high quality mesh of the matrix will be generated automatically. The fibers are modeled discretely, which enables the study of stress distribution along the fibers. In the future this approach should be able to extend to fracture analysis considering the slippage of fibers.

1.6 Organization of the thesis

The remainder of this thesis is organized as follows. In Section 2, background knowledge and the state-of-the-art of each individual topics are reviewed. In Section 3, the formulations of the scaled boundary finite element method in 3D are derived. In Section 4, an automatic polyhedron mesh generation method based on octree algorithm is presented. A coupling method for non-matching meshes using polyhedron element formulated in the scaled boundary finite element method is developed in Section 5. Its application in domain decomposition is studied. In Section 6, an adaptive analysis procedure is developed including error estimation and local mesh refinement. In Section 7, numerical simulation using virtual reality models is investigated. A discrete modeling method of fiber reinforced material is presented in Section 8. Conclusions and recommendations for the future work are stated in Section 9.

1.7 List of publications

1.7.1 Journal papers

1. Zhang, J., and Song, C. (2019). A polytree based coupling method for non-matching meshes in 3D. Computer Methods in Applied Mechanics and Engineering, 349, 743-773.
   https://doi.org/10.1016/j.cma.2019.02.038
2. Xing, W., Zhang, J., Song, C. and Tin-Loi, F. (2019). A node-to-node scheme for three-dimensional contact problems using the scaled boundary finite element method. Computer Methods in Applied Mechanics and Engineering, 347, 928-956.
   https://doi.org/10.1016/j.cma.2019.01.015
3. Liu, L., Zhang, J., Song, C., Birk, C., and Gao, W. (2019). An automatic approach for the acoustic analysis of three-dimensional bounded and unbounded domains by scaled boundary finite element method. International Journal of Mechanical Sciences, 151, 563-581.
   https://doi.org/10.1016/j.ijmecsci.2018.12.018
4. Liu, L., Zhang, J., Song, C., Birk, C., Saputra, A. and Gao, W. Automatic three-dimensional acoustic-structure interaction analysis using the scaled boundary finite element method. Journal of Computational Physics, 395, 432-460.
   https://doi.org/10.1016/j.jcp.2019.06.033

5 S. Natarajan, P. A. Dharmadhikari, R. K. Annabattula, Zhang, J., Ean Tat Ooi, and Song, C. A novel method to treat implicitly defined interfaces without enrichment. Computers and Structures (under review).

1.7.2 Conference proceedings

1. Zhang, J., and Song, C. (2018). An automatic discrete method for modeling reinforced material using the scaled boundary finite element method. ACMSM25. Brisbane, Australia
2. Zhang, J., and Song, C. (2018). Automatic polyhedral mesh generation for scaled boundary finite element analysis. WCCM13. New York City, US
3. Zhang, J., and Song, C. (2018). Virtual city analysis using scaled boundary finite element method. ACCM3. Melbourne, Australia

Chapter 2

Literature review

Structure analysis is an important task in engineering. The problems in structure analysis are usually represented in the form of differential equations. Numerical methods have been developed to solve these differential equations as analytical solutions are usually not available (Section 1.2). Finite element method (FEM) is the most widely used numerical method presently. However, there are challenges in the conventional FEM, some of which have been stated in Section 1.4. In the FEM, the geometric model needs to be discretized into elements with simple geometries (Fig. 1.2), which is referred to as mesh generation. Human intervention is frequently required in the mesh generation process, which is time consuming and error prone (Section 1.4.1). This disadvantage limits the application of the FEM, especially in moving boundary problems where the mesh needs to be updated frequently (Section 1.4.1.2). The emerging techniques in engineering imposes additional challenges on the mesh generation, such as the using of non-conventional data formats for the geometry (Section 1.4.1.1). Therefore, it is the aim of this research to develop a numerical framework to link geometric modeling and structure analysis automatically, which is suitable for the rapidly development of engineering in the digital age.

In this chapter, a literature review on the topics included in this research is pre-

sented. In Section 2.1, some numerical methods developed for structure analysis are summarized. In Section 2.2, mesh generation algorithms commonly used in the numerical methods are presented. The techniques to handle non-matching meshes are reviewed in Section 2.3. The numerical simulation of virtual reality models is introduced in Section 2.4. In Section 2.5 the application of adaptive analysis is briefly discussed. The numerical methods for the modeling of fiber reinforced material are summarized in Section 2.6.

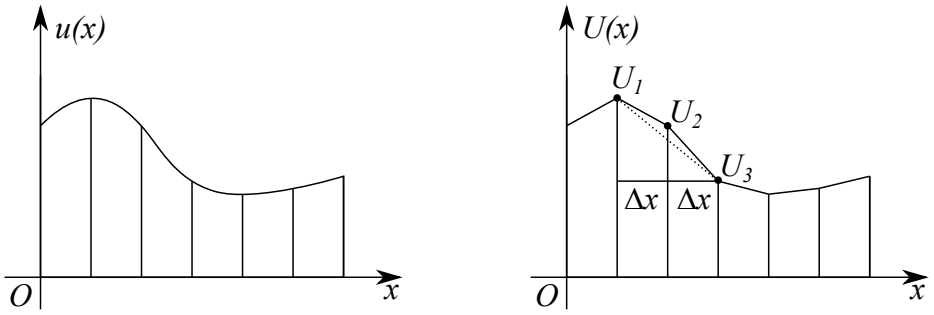
2.1 Numerical methods

A large number of numerical methods have been developed for structure analysis in the last century. In this section, some of the important numerical methods are reviewed. The advantages and disadvantages of these methods are discussed and compared.

2.1.1 Finite difference method

The finite difference method (FDM) was among the earliest developed numerical methods (Richardson Lewis and Glazebrook Richard, 1911; Courant et al., 1928). A short history of the development of FDM was presented by Thomee (2001). In FDM a problem domain is discretized into a structured (usually uniform spaced) grid. An approximate solution is constructed considering the governing differential equation and boundary conditions. The governing differential equation is converted into solving a set of linear algebra equations, which can be easily processed by the computers.

An example of finite difference method in 1D is shown in Fig. 2.1a. Consider a linear



(a) Analytical solution $u(x)$ (b) Approximate solution $U(x)$

Figure 2.1: Finite difference method in 1D

second order boundary value problem (Forsythe and Wasow, 1960)

$$a_2(x)\,u'' + a_1(x)\,u' + a_0(x)\,u = f(x) \qquad \text{in } x_l < x < x_r, \tag{2.1a}$$

$$b_{l1}u' + b_{l0}u = A_l \qquad \text{for } x = x_l, \tag{2.1b}$$

$$b_{r1}u' + b_{r0}u = A_r \qquad \text{for } x = x_r, \tag{2.1c}$$

where $[x_l, x_r]$ is the problem domain, $f(x)$, $a_0(x)$, $a_1(x)$ and $a_2(x)$ are functions of x, b_{l1}, b_{l0}, b_{r1}, b_{r0}, A_l and A_r are constants.

The problem domain is divided into M segments uniformly. In this example, $M = 6$. The step size is

$$\triangle x = \frac{(x_r - x_l)}{M}. \tag{2.2}$$

There are $M + 1$ grid points in the system after the discretization. The coordinate of m-th grid point is

$$x_m = x_l + m\triangle x \qquad \text{in } 0 \le m \le M. \tag{2.3}$$

Using Taylor series expansion, the first and second derivatives of the function u at

a specific grid point x can be approximated by the central difference

$$u'(x) = \frac{u(x+\triangle x) - u(x-\triangle x)}{2\triangle x} + O\left(\triangle x^2\right), \tag{2.4a}$$

$$u''(x) = \frac{u(x+\triangle x) - 2u(x) + u(x-\triangle x)}{\triangle x^2} + O\left(\triangle x^2\right). \tag{2.4b}$$

A discrete function U is constructed to approximate the exact solution (Fig. 2.1b). The function value at x_m is denoted as U_m. Therefore, Eq. (2.1a) can be written as a discrete function at point x_m

$$a_2(x_m)\frac{U_{m+1} - 2U_m + U_{m-1}}{\triangle x^2} + a_1(x_m)\frac{U_{m+1} - U_{m-1}}{2\triangle x} + a_0(x_m)u = f(x_m). \tag{2.5}$$

The boundary conditions can be applied by introducing two fictitious points x_{-1} and x_{M+1}. The values of the approximate solution at the fictitious points are expressed as U_{-1} and U_{M+1}. The boundary conditions in Eq. (2.1b) and Eq. (2.1c) are written as

$$b_{l1}\frac{U_1 - U_{-1}}{2\triangle x} + b_{l0}U_0 = A_l, \tag{2.6a}$$

$$b_{r1}\frac{U_{M+1} - U_{M-1}}{2\triangle x} + b_{r0}U_M = A_r. \tag{2.6b}$$

Formulating Eq. (2.5) at all internal grid points ($0 \leq m \leq M$) and considering the boundary conditions in Eq. (2.6), a system of $M+3$ linear algebraic equations are obtained. The equations can be written in a matrix form as

$$\boldsymbol{KU} = \boldsymbol{F}, \tag{2.7}$$

where $\boldsymbol{U}$ is the array of unknown variables

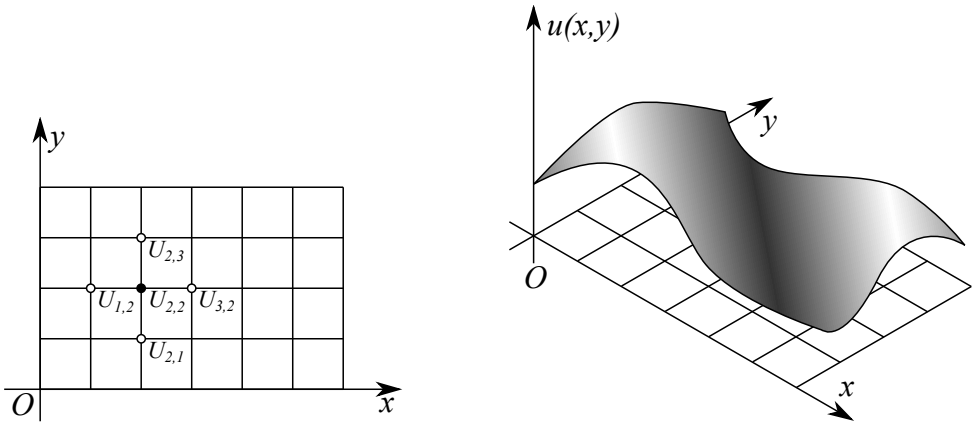
$$\boldsymbol{U} = \begin{bmatrix} U_{-1} \\ U_0 \\ U_1 \\ \cdots \\ U_M \\ U_{M+1} \end{bmatrix}, \tag{2.8}$$

and $\boldsymbol{F}$ is the array of values on the right hand side of Eq. (2.5) and Eq. (2.6)

$$\boldsymbol{F} = \begin{bmatrix} A_l \\ f(x_0) \\ f(x_1) \\ \cdots \\ f(x_M) \\ A_r \end{bmatrix}. \tag{2.9}$$

The stiffness matrix $\boldsymbol{K}$ in finite difference method is

$$\boldsymbol{K} = \begin{bmatrix} -\tilde{b}_{l1} & b_{l0} & \tilde{b}_{l1} & & & \\ \beta_0^- & \alpha_0 & \beta_0^+ & & & \\ & \beta_1^- & \alpha_1 & \beta_1^+ & & \\ & & \cdots & \cdots & \cdots & \\ & & & \beta_M^- & \alpha_M & \beta_M^+ \\ & & & -\tilde{b}_{r1} & b_{r0} & \tilde{b}_{r1} \end{bmatrix}, \tag{2.10}$$



(a) Uniform grid in $x-y$ plane (b) Analytical solution $u\left(x,y\right)$

Figure 2.2: Finite difference method in 2D

where the entries in the stiffness matrix are

$$\alpha_m = -\frac{2a_2\left(x_m\right)}{\triangle x^2} + a_0\left(x_m\right), \tag{2.11a}$$

$$\beta_m^{\pm} = \frac{a_2\left(x_m\right)}{\triangle x^2} \pm \frac{a_1\left(x_m\right)}{2\triangle x}, \tag{2.11b}$$

$$\tilde{b}_{l1} = \frac{b_{l1}}{2\triangle x}, \tag{2.11c}$$

$$\tilde{b}_{r1} = \frac{b_{r1}}{2\triangle x}. \tag{2.11d}$$

All the unknown values at the grid points, including the fictitious points, (U_{-1}, U_0, ..., U_{M+1}) can be solved.

Similarly, the partial derivatives in 2D can be approximated using the 4 grid points surrounding the point under consideration (Fig. 2.2). The assembly and solving procedure are similar to those in 1D problem.

The FDM can be classified as a collocation method (Fries et al., 2004). It requires that the governing differential equation be satisfied at specific grid points, which can be achieved by introducing a set of displaced Dirac delta functions $\delta\left(x-x_i\right)$ as weighting functions . The stiffness matrix $\boldsymbol{K}$ in the FDM is banded and usually well-conditioned. However, it is difficult for the FDM to be extended to irregular mesh, which strongly

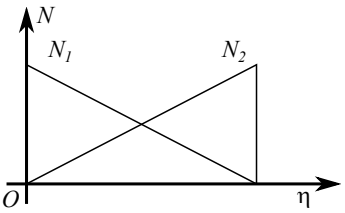
limited its application. The boundary conditions can be difficult to apply if the geometry of the problem domain is complex. In Richardson Lewis and Glazebrook Richard (1911) the FDM was applied to the stress analysis of a masonry dam in 2D. The geometry of the dam was simplified so that only horizontal, vertical and 45° inclined lines were used to represent the geometry. FDM was applied to study fluid flow in porous media in 3D by Narasimhan and Witherspoon (1976). Liszka and Orkisz (1980) explored the application of FDM in arbitrary irregular grid in 2D. It was reported that mesh can be refined locally and boundary conditions can be applied on domains with arbitrary shapes. Some recent developments on FDM and combinations with other numerical methods were reported in Sousa and Li (2015); Pandey (2015); E. Griffith and Luo (2017).

2.1.2 Finite element method

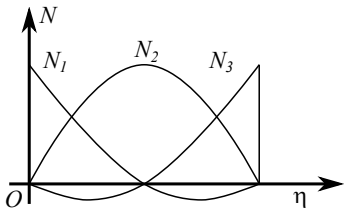
The finite element method (FEM) is presently the most popular numerical method used in engineering. There is a sizable literature on the development of the FEM (Oden, 1990; Gupta and Meek, 1996; Thomee, 2001). Some pioneer work can be found in Argyris (1955); Turner (1956); Clough (1960). The discretization of the problem domain in the FEM is similar to the FDM. A piecewise continuous function is constructed to approximate the exact displacement field. In the FEM, the problem domain can be discretized using several different types of elements (see Fig 1.2). Both structured and unstructured mesh can be used in the FEM (the details will be presented in Section 2.2). Polynomial shape function is utilized in FEM to interpolate the node values. Some examples of shape functions are plotted in Fig 2.3. The 1D linear shape functions in Fig 2.3a are expressed as

$$N_1(\eta) = 1 - \eta, \tag{2.12a}$$

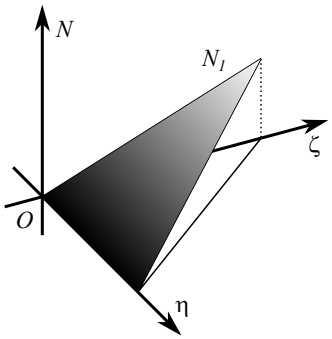
$$N_2(\eta) = \eta, \tag{2.12b}$$



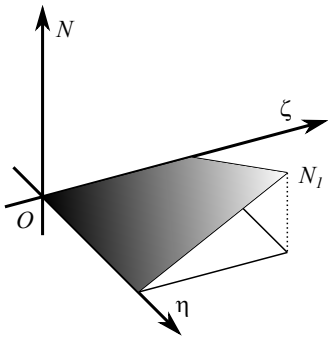
(a) Linear shape function in 1D



(b) Quadratic shape function in 1D



(c) Linear triangle shape function in 2D



(d) Linear quadrilateral shape function in 2D

Figure 2.3: Examples of finite element shape functions

where η is the coordinate of the isoparametric element. The shape functions can be written in matrix form as

$$[N] = \begin{bmatrix} N_1(\eta) & N_2(\eta) \end{bmatrix}. \tag{2.13}$$

The shape functions satisfy the Kronecker delta property and they form a partition of unity inside each element.

The displacement field in an element can be interpolated as (Zienkiewicz et al., 1977)

$$\{u\} = [N]\{u_d\}, \tag{2.14}$$

where $[N]$ is the shape function matrix and $\{u_d\}$ is the nodal value vector. The strain field is

$$\{\varepsilon\} = [L]\{u\} = [B]\{u_d\}, \tag{2.15}$$

where $[L]$ is the differential operator. In 3D problems, $[L]$ is defined as

$$[L] = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \end{bmatrix}. \tag{2.16}$$

The strain matrix $[B]$ is defined as

$$[B] = [L]\,[N]\,. \tag{2.17}$$

Considering the constitutive model, the stress is calculated as

$$\{\sigma\} = [D]\,\{\varepsilon\} = [D]\,[B]\,\{u_d\}\,, \tag{2.18}$$

where $[D]$ is the elasticity matrix. For an isotropic material in 3D, it is expressed as

$$[D] = \frac{E}{(1+\nu)\,(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}, \tag{2.19}$$

in which E is Young's modulus and ν is Poisson's ratio. The element stiffness matrix in FEM can be derived using virtual work principle (Zienkiewicz and Cheung, 1964). Alternatively, it can be derived by Galerkin method using element shape function as

the weighting function.

$$[K] = \int [B]^T [D] [B] \, dV. \tag{2.20}$$

The global stiffness matrix $[K_G]$ is then assembled using node connectivity. The global equation is written as

$$[K_G] \{U_G\} = \{F_G\}, \tag{2.21}$$

where $\{U_G\}$ is the global nodal displacement vector and $\{F_G\}$ is the global nodal force vector.

The FEM attracted extensive attention immediately after it was proposed. It was successfully applied to a large amount of engineering topics, such as isotropic and orthotropic plates (Zienkiewicz and Cheung, 1964), anisotropic flow problems (Zienkiewicz et al., 1966), elasto-plastic problems (Zienkiewicz et al., 1966), fracture mechanics (Chan et al., 1970) contact (Hughes et al., 1976) and many others. A summary of the active research groups during the development of FEM can be found in Gupta and Meek (1996). The FEM offers greater flexibility in modeling of structures with complex geometry. Both essential and natural boundary conditions can be applied easily. The stiffness matrix in the FEM is also banded. However, a disadvantage of FEM is the difficulty in mesh generation. Human intervention is frequently required to generate a high quality mesh, which is time consuming and error prone. In moving boundary problems, the mesh updating becomes a major challenge. The shape function in the FEM is polynomial, which is not adequate to represent stress singularity in fracture problems. Nowadays FEM is a well-developed numerical method. Some variants will be introduced in the following sections.

2.1.3 Boundary element method

The boundary element method (BEM) was also known as "boundary integral equation methods" or "boundary integral solutions" in the early literature (Brebbia and

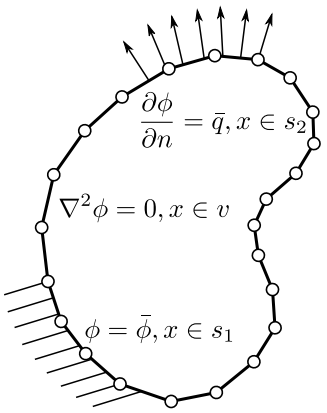


Figure 2.4: Boundary element method

Dominguez, 1977). The governing equation of the BEM can be derived using the weighted residual technique. In BEM, the weighting function is chosen as the fundamental solution of the equation representing a concentrated potential at a point i

$$\nabla^2 \psi + \Delta_i = 0, \tag{2.22}$$

where Δ_i is the Dirac delta function. A well-known solution of this function in 3D is

$$\psi = \frac{1}{4\pi r}, \tag{2.23}$$

where r is the distance from point i to the point under consideration. The volume integral of the governing equation can be converted to a surface integral (Cheng and Cheng, 2005). Only the boundary of the problem domain needs to be discretized (Fig 2.4).

In BEM no volume discretization is required, therefore the mesh generation can be largely simplified and the numbers of degree of freedom of the system can be reduced. Moving boundary problems such as crack propagation can be modeled easily (Mi and Aliabadi, 1994). It is suitable to model unbounded domain (Gaul and Schanz, 1999). However, there are some disadvantages in the BEM. A fundamental solution needs to be known before the analysis, which is not always practical in engineering. BEM usually produces fully populated matrices, which is less efficient compared with the

banded stiffness matrix in FDM and FEM (Ahmad and Banerjee, 1986). BEM has been applied to nonlinear analysis of water waves (Grilli et al., 1989), sloshing (Nakayama and Washizu, 1981) and others (Ruotsalainen and Wendland, 1988; Katsikadelis and Nerantzaki, 1999). However, in the nonlinear stress analysis the volume of the problem domain still needs to be discretized (Telles and Carrer, 1991; Carrer and Telles, 1992; Telles and Carrer, 1994), which removes one of the most important advantages of the BEM. Summaries of the development of the BEM can be found in (Liu et al., 2011; Brebbia and Walker, 2016). Some recent advances were reported in (Zhang and Liu, 2015; Nguyen et al., 2016; Li and Popov, 2018)

2.1.4 Extended finite element method

The extended finite element method (XFEM) was developed to overcome the difficulties associated with modeling cracks in FEM (Belytschko and Black, 1999). In XFEM, the mesh doesn't need to conform to the geometry around the crack tip (Fig. 2.5). The discontinuous displacement field around the crack tip is modeled by introducing enrichment functions (Abdelaziz and Hamouine, 2008)

$$\left\{F_j\left(r,\theta\right)\right\}_{j=1}^{4}=\left\{\sqrt{r}\cos\left(\frac{\theta}{2}\right),\sqrt{r}\sin\left(\frac{\theta}{2}\right),\sqrt{r}\sin\left(\frac{\theta}{2}\right)\sin\left(\theta\right),\sqrt{r}\cos\left(\frac{\theta}{2}\right)\sin\left(\theta\right)\right\}, \tag{2.24}$$

where r and θ are the polar coordinate with origin at the crack tip. The nodes around the crack tip have more degrees of freedom. The approximation takes the form of an extrinsic enrichment and can be written as

$$\boldsymbol{u}^h\left(x\right)=\sum_{i=1}^{n}N_i\left(x\right)\left(u_i+\sum_{j=1}^{ne(i)}a_{ji}F_j\left(r,\theta\right)\right), \tag{2.25a}$$

$$\boldsymbol{v}^h\left(x\right)=\sum_{i=1}^{n}N_i\left(x\right)\left(v_i+\sum_{j=1}^{ne(i)}b_{ji}F_j\left(r,\theta\right)\right), \tag{2.25b}$$

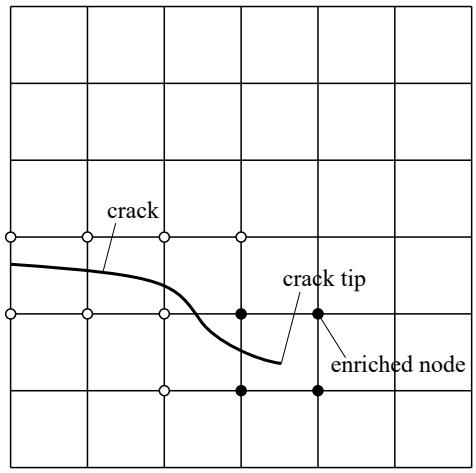


Figure 2.5: Extended finite element method

where $N_i(x)$are the standard FEM shape function. The coefficients a_{ji} and b_{ji} are associated with nodes and $ne(i)$ is the number of coefficients for i-th node. $ne(i)$ is equal to 4 for the nodes around the crack tip and zero for all the other nodes.

XFEM has been applied to solve many problems involving weak discontinuity. The modeling of dislocations in systems with multiple arbitrary material interfaces was presented in Belytschko and Gracie (2007). It was used to evaluated stress intensify factor on bimaterial interface by Liu et al. (2004). Liu et al. (2016) presented a multi-material topology optimization strategy considering the cohesive constitutive relationship of the interface. XFEM was utilized to model the material interface of fiber reinforced polymer in Kastner et al. (2011).

A comparison between weak, strong and singular enrichment in the XFEM were presented in Bouhala et al. (2013). Crack propagation in 2D was modeled using XFEM without frequent remeshing (Dolbow et al., 2000). Cracks in 3D problems were studied in Loehnert et al. (2011); Agathos et al. (2016). More recently, the modeling of non-planar crack was reported in Agathos et al. (2018). The application of XFEM in polycrystalline microstructure was explored in Beese et al. (2018). Reviews of the XFEM can be found in Karihaloo and Xiao (2003); Fries and Belytschko (2010)

2.1.5 Isogeometric analysis

Isogeometric analysis (IGA) is a numerical method developed by Hughes et al. (2005). A salient advantage of IGA is that it doesn't require the data conversion from a NURBS based geometric model to a discrete finite element model. A brief introduction to NURBS has been presented in Section 1.4.1.1. The shape function in IGA is constructed using NURBS directly. An example of quadratic basis functions for open, non-uniform knot vector $\Xi = \{0, 0, 0, 1, 2, 3, 4, 4, 5, 5, 5\}$ is shown in Fig. 2.6 (Hughes et al., 2005). It has been applied to many problems such as contact (De Lorenzis et al., 2014), vibration (Cottrell et al., 2006), fluid-structure interaction (Bazilevs et al., 2006, 2008), multiscale turbulence (Bazilevs et al., 2007) and many others (Auricchio et al., 2007). A new refinement strategy, the k-refinement was introduced to the IGA. It was claimed that the k-refinement performs better than the traditional p-refinement (Cottrell et al., 2007).

Usually a model cannot be represented by one single NURBS patch. Instead, several NURBS patches need to be trimmed and connected, which may cause discrepancy in the model, such as gaps and overlapping. A review of the trimming procedure in IGA was presented in Marussig and Hughes (2018). A new IGA using T-spline as basis function was proposed in Bazilevs et al. (2010). Rows of control points in T-spline can be incomplete therefore local refinement and coarsening of control points are allowed. It was shown that watertight models can be generated easily using multiple T-spline patches. T-spline has been adopted by 3D modeling software Rhino and Maya (Bin et al., 2008).

IGA provides a powerful alternative to the conventional FEM. The interoperability between CAD and CAE systems can be greatly improved using NURBS. However, if the model is represented by other formats, such as STL or digital image mentioned in Section 1.4.1.1, it will be difficult for IGA to handle.

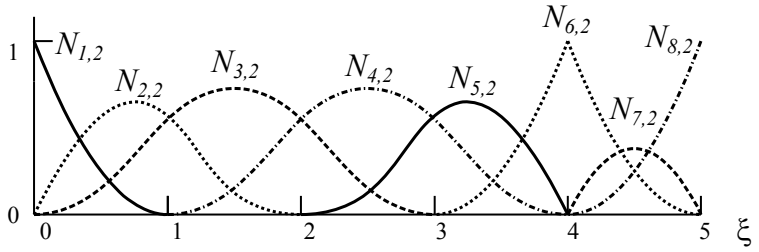


Figure 2.6: Quadratic basis functions in NURBS

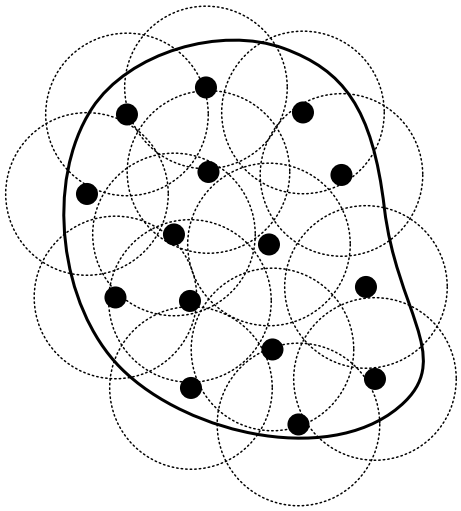


Figure 2.7: Meshfree method

2.1.6 Meshfree method

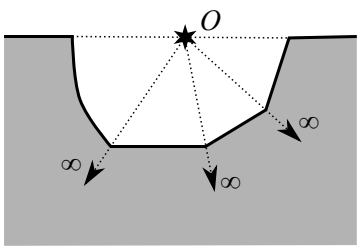
Meshfree method (MM) is a group of numerical methods which do not require a conventional mesh before the analysis. Only a group of scattered points are generated in the problem domain (Fig. 2.7) and the connectivity is established during the analysis. The first and simplest MM, smoothed particle hydrodynamics, was introduced by Lucy (1977) to test the fission hypothesis. It was further investigated using kernel estimation in Monaghan (1982). Some variations of smoothed particle hydrodynamics were developed such as corrected smoothed particle hydrodynamics (Bonet and Kulasegaram, 2000) and moving least-squares particle hydrodynamics (Dilts, 1999). Other MMs include diffuse element method (Muravin and Turkel, 2003), element free Galerkin (Belytschko et al., 1994) and *hp*-clouds (Liszka et al., 1996). Reviews of the development and classification of MM can be found in (Li and Liu, 2002; Fries et al., 2004).

MM greatly reduces the difficulty related to mesh generation as only a group of scattered points are required. Problems with complex geometry, large deformation, moving boundaries can be studied without tedious remeshing process. Both h and p refinement can be implemented in MM. However these methods usually do not construct an exact partition of unity (Fries et al., 2004). A searching algorithm is required to construct connectivity during the analysis, which can be expensive in terms of computational cost. The boundary conditions are difficult to apply.

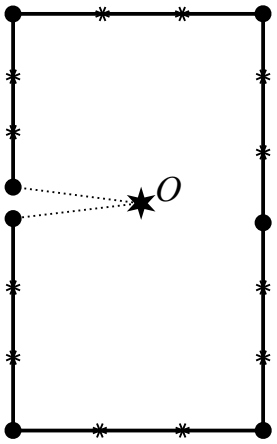
2.1.7 Scaled boundary finite element method

The scaled boundary finite element method (SBFEM) is a semi-analytical method which was first proposed by Song and Wolf (1997). The method was initially developed for modeling wave propagation in unbounded domains (Fig. 2.8a) and later explored to analyze problems with stress singularities. Deeks and Wolf (2002b) demonstrated that the scaled boundary finite element method outperforms the finite element method in situations involving unbounded domains or stress concentrations (Fig. 2.8b). During the last two decades, researchers have endeavored to apply this method to solve problems in wave propagation (Bazyar and Song, 2008), acoustics (Lehmann et al., 2006; Liu et al., 2019), fracture (Song, 2005; Li et al., 2016), stress intensity factor evaluation (Saputra et al., 2015), anisotropic soil (Lu et al., 2016), contact (Xing et al., 2018, 2019), sloshing (Ye et al., 2018), seepage (Liu et al., 2018), among many others (Lin et al., 2011, 2012, 2013, 2018). A review of the SBFEM in linear elastic fracture can be found in Song et al. (2017). The fundamental theory of the scaled boundary finite element is presented in Song (2018) accompanied by a computer program written in MATLAB.

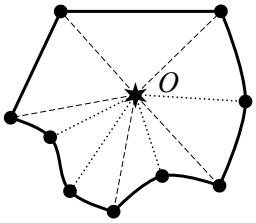
Recently, the SBFEM is extended to develop polygon elements in 2D (Chiong et al., 2014) and polyhedron elements in 3D (Talebi et al., 2016; Liu et al., 2017). The polytope (polygon/polyhedron) elements constructed by the SBFEM significantly reduce the



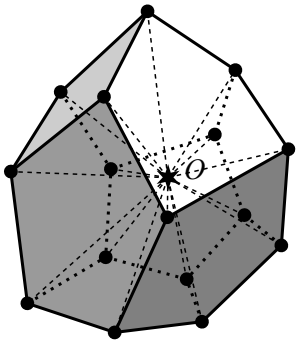
(a) Modeling of unbounded domain in 2D



(b) Modeling of edge cracked plate in 2D



(c) Polygon element in 2D



(d) Polyhedron element in 3D

Figure 2.8: Some examples in the scaled boundary finite element method

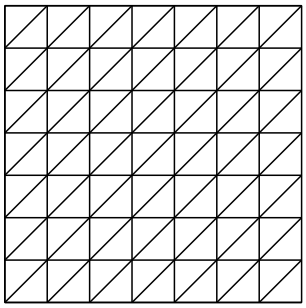
meshing burden of the standard finite elements. The geometry of a polytope element needs to meet only the scaling requirement, i.e. a point from which the whole boundary is directly visible can be identified (points denoted with O in Fig. 2.8). As only the boundary of a polytope element needs to be discretized, polytope elements of arbitrary number of faces, edges and vertices can be constructed. Examples of the polytope elements formulated in the SBFEM are shown in Fig. 2.8c and Fig. 2.8d. The polygon elements have been applied to study contact problems on the crack face (Zhang et al., 2018), hydraulic failure at the dam-foundation interface (Zhong et al., 2018) and non-linear analysis of slop stability (Lak and Bazyar, 2019).

The arbitrary polytope elements formulated by the SBFEM provide a higher degree of flexibility in automatic mesh generation. The simple quadtree/octree algorithm can be employed for mesh generation and it is highly complementary to the SBFEM. The details of quadtree/octree based mesh generation will be presented in Section 2.2.4. The advantages of the combination of the the SBFEM and quadtree/octree mesh are demonstrated by Ooi et al. (2015) in 2D and Saputra et al. (2017) in 3D. The thermal conductivity of heterogeneous materials is investigated using quadtree mesh in He et al. (2019). A combined quadtree-polygon based approach is applied to mesoscale modeling of concrete in Guo et al. (2019). Non-local damage problems is analyzed using quadtree mesh in Zhang et al. (2019). An automatic polyhedral mesh generator based on octree algorithm is developed in Liu et al. (2017). More recently, the combination of the SBFEM and octree based mesh is applied to geotechnical structures (Chen et al., 2018), saturated soil (Zou et al., 2018), porous media (Zou et al., 2019) and many others (Talebi et al., 2016).

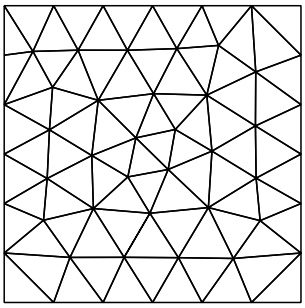
In this research, the SBFEM is used for structure analysis. The equations of the SBFEM will be derived in Chapter 3.

2.2 Mesh generation

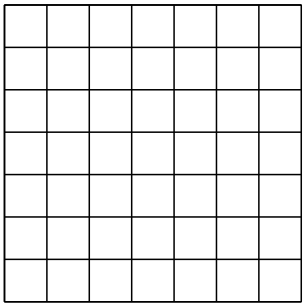
Mesh generation is an important step for many numerical analysis (Park et al., 2017). The quality of mesh has a significant impact on the accuracy of the result of analysis. However, the generation of a high quality mesh generation is a nontrivial task, which has been stated in Section 1.4.1. A large number of methods have been developed for the mesh generation (Thompson et al., 1985; Ho-Le, 1988; Geuzaine and Remacle, 2009). There are different classifications of these methods. For instance, they can be classified based on the type of elements (Fig. 1.2), e.g. triangles, quadrilaterals, polygons, tetrahedra, hexahedra and polyhedra. They can also be classified as structured and unstructured mesh generation methods. In structured mesh generation, meshes are constructed in such a way that the points can be regarded as the intersection points of curvilinear co-ordinate curves in 2D or surfaces in 3D (Farrashkhalvat and Miles, 2003). All the inner nodes have the same number of elements around them. Otherwise, the mesh is regarded as unstructured. Usually a mesh is still considered structured if it is composed of several structured meshes for individual parts of a model. Some examples of structured and unstructured meshes are shown in Fig. 2.9. Alternatively, they can be classified as direct and indirect methods (Botella et al., 2016). In an indirect method, a mesh of different element type is generated first and then converted to the desired element type. For example, a tetrahedron can be divided into 4 hexahedra (Fig. 2.10). The common conversions are subdivision, merging, primal/dual approach. In this section, the mesh generation methods will be introduced based on the algorithms, including mapping and sweeping, Delaunay triangulation, advancing front and grid based method.



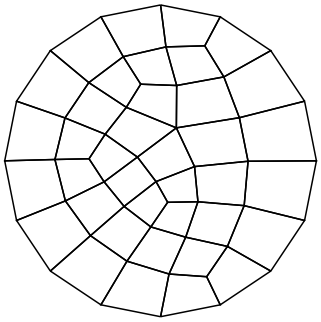
(a) Structured triangle mesh



(b) Unstructured triangle mesh

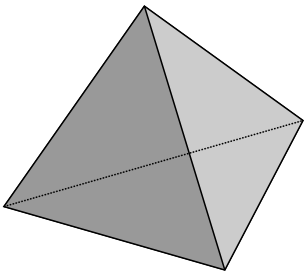


(c) Structured quadrilateral mesh

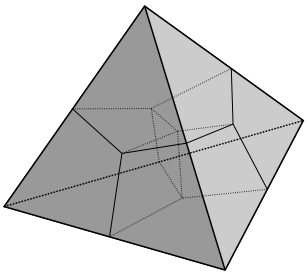


(d) Unstructured quadrilateral mesh

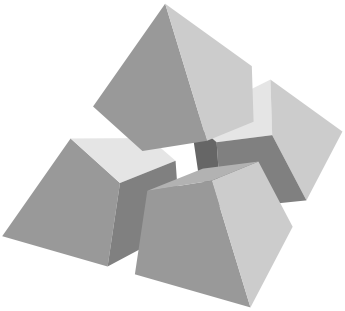
Figure 2.9: Examples of structured and unstructured meshes



(a) A tetrahedron element



(b) Subdivision of the tetrahedron element



(c) 4 hexahedron elements after the subdivision

Figure 2.10: Example of indirect mesh generation

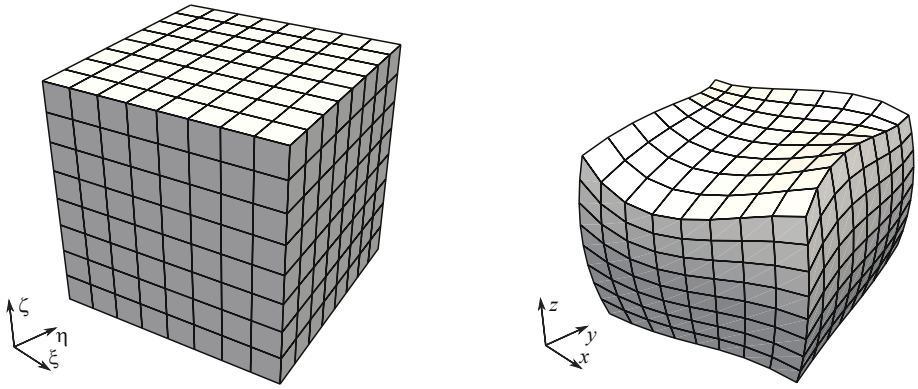
2.2.1 Mapping and sweeping

Although the ideas are relatively simple, mapping and sweeping are still widely used methods in the generation of hexahedron meshes in conventional FEM. They are considered to be reliable in terms of element quality. They usually require manual preprocessing on the geometric models.

2.2.1.1 Mapping

Mapping is usually utilized to generate structured hexahedron mesh. A parent hexahedron mesh is generated in a parametric space (Fig. 2.11a). It is then mapped into the actual geometry of the model (Fig. 2.11b). The mapping algorithm using bivariable blending function interpolation was introduced in Gordon and Hall (1973). Sometimes a model needs to be decomposed into several simpler parts (Fig. 2.12b) and each part is mapped individually (Fig. 2.12c). A decomposition method and transfinite mapping procedure was reported in Liu and Gadh (1996). The decomposition process usually involves laborious manual process. Researchers have been endeavoring to automate the decomposition process. A novel "virtual decomposition" method was developed in White et al. (1995). A feature recognition method was presented in Tautges et al. (1997) for mapping individual parts.

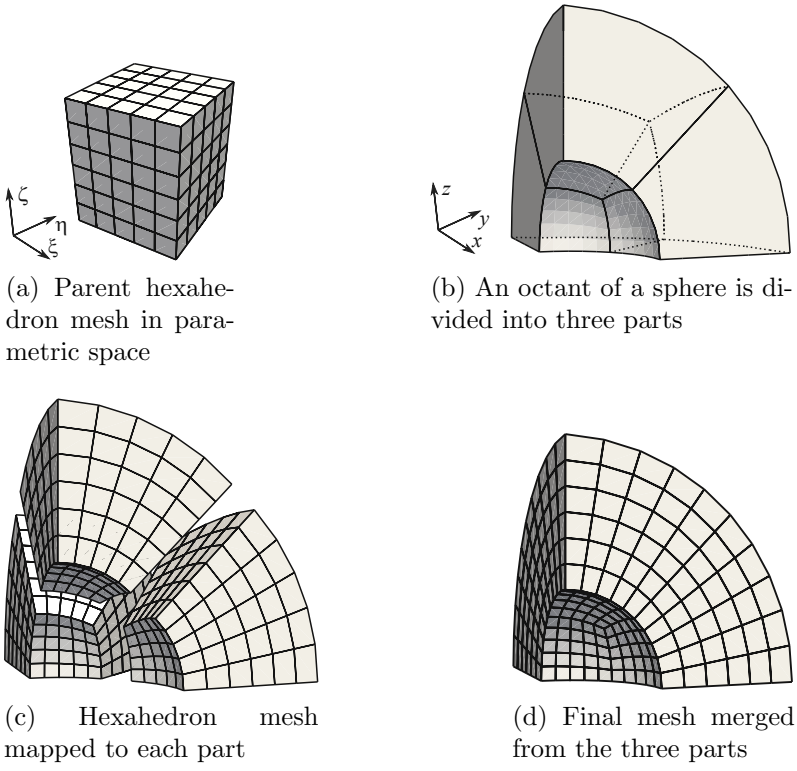
The advantage of mapping is that it is simple to implement. The topology of the structured hexahedron mesh is fixed, which is highly efficient in computational storage. However, there are significant limitations and drawbacks of this method. The geometry of each part has to be equivalent to a cube in topology, i.e. 6 faces and 12 edges. The manual decomposition is tedious and time-consuming. The quality of mesh heavily depends on the shape of the geometry, i.e. the mapped elements can be severely distorted if the geometry contains curved surface.



(a) Parent hexahedron mesh in parametric space

(b) Mapped hexahedron mesh

Figure 2.11: Mesh generation of a curvilinear cube using mapping



(a) Parent hexahedron mesh in parametric space

(b) An octant of a sphere is divided into three parts

(c) Hexahedron mesh mapped to each part

(d) Final mesh merged from the three parts

Figure 2.12: Mesh generation of an octant of hollow sphere using decomposition and mapping

(a) Quadrilateral mesh generated on the source surface

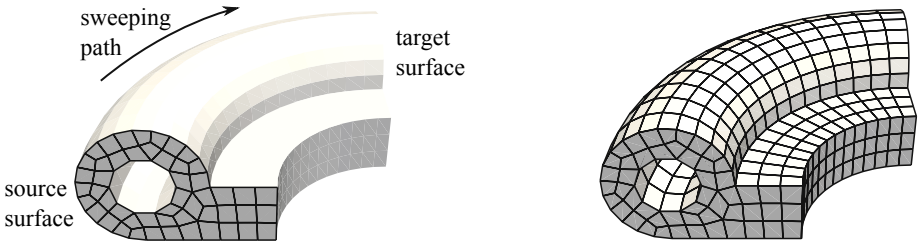
(b) Hexahedron mesh after sweeping

Figure 2.13: Mesh generation of a cylinder using sweeping

2.2.1.2 Sweeping

Sweeping is an algorithm to generate 3D meshes from 2D surface meshes. It is also referred to as $2^1/_2$D meshing. The geometry of the model contains a source surface, a target surface and a sweeping path. The shape of the source surface is usually simple so that a quadrilateral or triangle mesh can be generated for the surface (see Fig. 2.13a and Fig. 2.14a). The target surface is topologically equivalent to the source. A structured quadrilateral mesh is generated for the side face. Then the mesh of the source is offset along the sweeping path. If the source surface is meshed with quadrilateral element, the volume mesh will be a hexahedron mesh. If a triangle mesh is generated for the source surface, prism volume mesh will be obtained. Different algorithms to determine the interior node locations were reported in Mingwu et al. (1996). A complex model can be decomposed into several parts which can be swept. A "cooper tool" was presented in Blacker (1996) which was able to enforce compatibility on the interface of different parts using imprinting and shifting algorithms. Sweeping using irregular, twisted and curved sweeping paths was presented in Staten et al. (1999). A sweeping algorithm based on least-square projection was proposed in Roca and Sarrate (2010).

The quality of the mesh generated by sweeping is usually good as long as the curvature of the sweeping path is reasonably small. The method is also robust if an initial surface mesh can be generated. However, the application of this method is also limited by the requirement on the geometry, i.e. a sweeping path needs to be identified and



(a) Quadrilateral mesh generated on the source surface

(b) Hexahedron mesh after sweeping

Figure 2.14: Mesh generation of a tube using sweeping

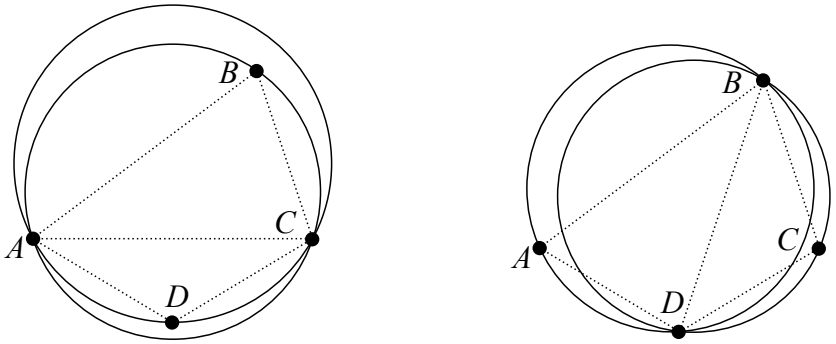
the source and target surfaces have to be similar. Manual operation is necessary to decompose the complex geometry into simpler ones which are suitable for sweeping. It is the manual decomposition that takes most of the time. Recently, a user-guided semi-automatic decomposition tool was reported in Lu et al. (2017). A fuzzy clustering algorithm was developed in Wu et al. (2018) for automatic decomposition. The capability of the algorithm was demonstrated using several complex models. Mesh generated using sweeping has been used in the optimization of aircraft engine in Shen et al. (2018).

2.2.2 Delaunay triangulation

Delaunay triangulation is widely used to generate unstructured triangle/tetrahedron meshes. The idea was first invented by Delaunay et al. (1934). The Delaunay criterion, also called the "empty sphere" property, states that any node must not be contained within the circumcircle/circumsphere of any triangle/tetrahedron within the mesh (Owen, 1998).

In 2D a convex quadrilateral can be divided into 2 triangles using 2 different schemes (Fig 2.15). In Fig 2.15a, the circumcircle of triangle ABC contains point D and triangle ACD contains point B, too, therefore the Delaunay criterion is not satisfied. The diagonal is flipped in Fig 2.15b. Now the circumcircles of triangles ABD and BCD do not contain any other points, therefore this is a valid Delaunay triangulation.

In 3D the cases are more complicated. There are 5 possible configurations of 5



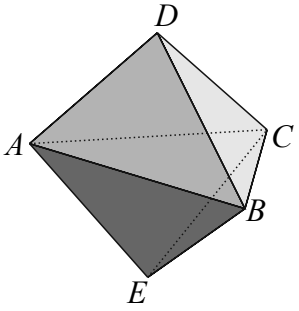
(a) Triangulation not satisfying Delaunay criterion

(b) Triangulation satisfying Delaunay criterion

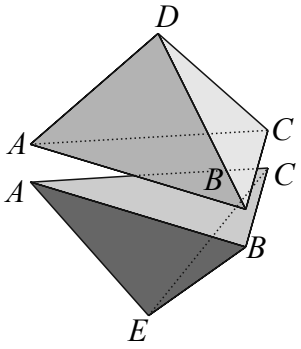
Figure 2.15: The 2D Delaunay triangulation of 4 non-collinear points

non-planar points, which were enumerated in Joe (1991). Here only the simplest case is considered in Fig 2.16a. Any 4 of the 5 points are non-planar, and all 5 points are located on the convex hull (the definition of convex hull can be found in Seidel (1986)). All faces are also located on the convex hull. There are two possible schemes to tetrahedronlize the polyhedron. In the first scheme, it is subdivided by plane ABC so that two tetrahedra $ABCE$ and $ABCE$ are generated (Fig 2.16b). In the second scheme, A new interior edge DE is constructed (Fig 2.16c), and the polyhedron is divided into 3 new tetrahedrons $ABDE$, $BCDE$ and $ACDE$ (Fig 2.16d). The circumspheres of $ABCE$ and $ABCE$ are shown in Fig 2.16e. It can be observed that the two spheres are both empty (not containing any other points). Therefore this is a valid tetrahedronlization satisfying Delaunay criterion. On the other hand, all the 3 circumspheres in Fig 2.16f contain all the 5 points, therefore this scheme doesn't satisfy the Delaunay criterion. The transformation between the 2 schemes is called 3-2 flipping in 3D (Shewchuk, 2002), similar to the edge flipping in Fig 2.15. A tetrahedronlization can always be achieved by doing this transformation iteratively.

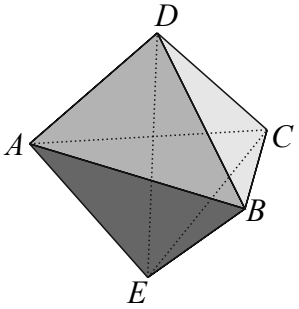
The Delaunay criterion itself is not a mesh generation method but merely a scheme to connect a set of existing points. It has to be combined with other point generation or insertion algorithms to generate meshes. Several automatic point creation schemes



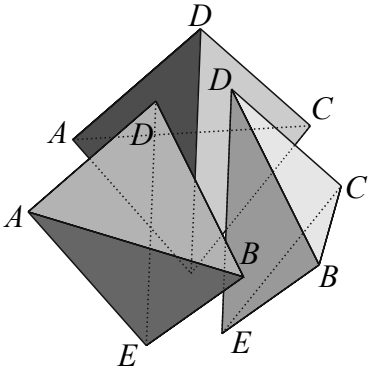
(a) A polyhedron with 5 points (any 4 of them are non-planar)



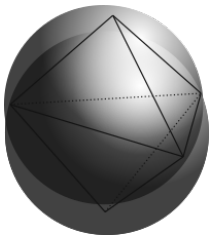
(b) Subdivision into 2 tetrahedrons



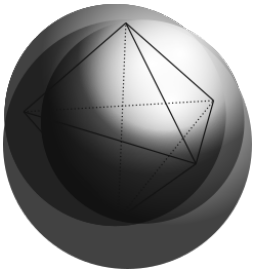
(c) New edge DE inserted



(d) Subdivision into 3 tetrahedrons



(e) Tetrahedronlization satisfying Delaunay criterion

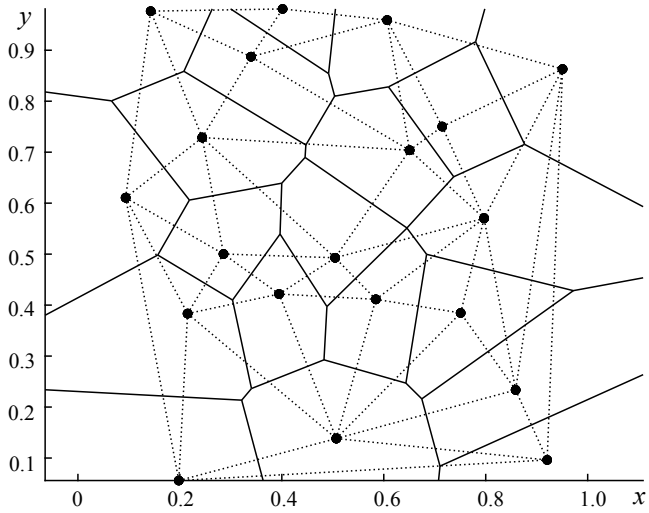


(f) Tetrahedronlization not satisfying Delaunay criterion

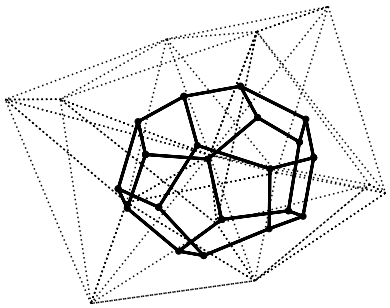
Figure 2.16: The 3D Delaunay tetrahedralization of 5 non-planar points

were compared in Weatherill and Hassan (1994). A constrained Delaunay triangulation scheme was proposed in Baker (1989) to generate tetrahedron mesh for objects of essentially arbitrary complexity. The treatment of boundary layers in Delaunay triangulation was discussed in George et al. (1991). The dual of a Delaunay triangulation is called a Voronoi diagram (Watson, 1981). It is a useful tool in the generation of polygon/polyhedron mesh (Yan et al., 2013). An example of 2D Delaunay triangulation and its dual Voronoi diagram is shown in Fig 2.17a. The dash lines represent the Delaunay triangulation of a randomly generated point set. The centers of the circumcircles of the triangles are defined as vertices in the Voronoi diagram. If two triangles share an edge, the two corresponding vertices in the Voronoi diagram will be connected. Each polygon in the Voronoi diagram corresponds to a node in the Delaunay triangulation. A 3D example of Delaunay tetrahedrons and a Voronoi polyhedron is shown in Fig 2.17b.

Delaunay triangulation is well-known for its robustness. However, a set of input points have to be defined using other algorithms. The quality of the obtained tetrahedron mesh depends on the distribution of the input points. Another difficulty associated with Delaunay triangulation is the boundary recovery. The "empty sphere" property may be violated locally in order to make the mesh conforming to the geometry. It can also be used to generate hexahedron mesh by subdividing each tetrahedron, but the obtained element quality is usually poor. Recently, a new tetrahedral mesh generator, "TetGen", was reported in Si (2015). The algorithm presented in Su et al. (2016) was designed for parallel processing of large scale point cloud. Delaunay triangulation using curved Bezier element was presented in Feng et al. (2018). An efficient serial implementation of the incremental Delaunay insertion algorithm was reported in Marot et al. (2019).

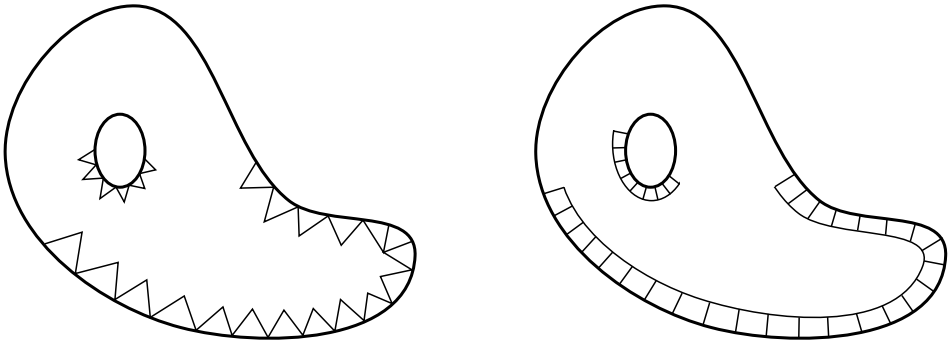


(a) Delaunay and Voronoi diagram in 2D



(b) Delaunay and Voronoi diagram in 3D

Figure 2.17: Delaunay and Voronoi diagram



(a) Traingle mesh generation by advancing front

(b) Quadrilateral mesh generation by advancing front

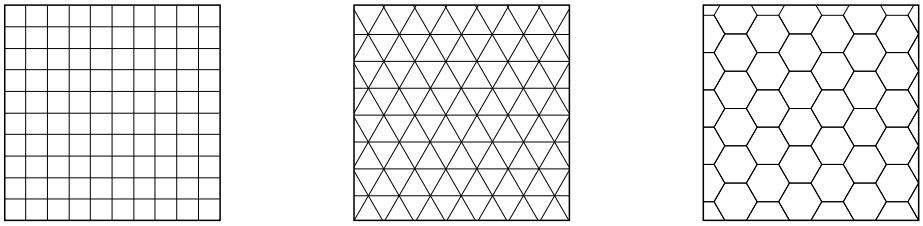
Figure 2.18: Mesh generation by advancing front

2.2.3 Advancing front

Advancing front is another widely used method in mesh generation. Sometimes it is called paving for quadrilateral mesh generation in 2D and plastering for hexahedron element in 3D (Owen, 1998). Some early development of the advancing front can be found in Lo (1985); Peraire et al. (1990); Lohner (1996). The generation of initial boundary layer and interior of the tetrahedron mesh were discussed in Lo (1991a,b). This method was applied to quadrilateral elements in Zhu et al. (1991). Paving algorithm was introduced by Blacker and Stephenson (1991). The extension to hexahedron mesh generation in 3D was reported in Blacker and Meyers (1993).

The advancing front algorithm starts from a surface mesh on the boundary (line segments in 2D and triangule/quadrilateral mesh in 3D). A layer of new elements are constructed by marching the front inward the model (Fig 2.18) and a new front is generated. This operation is done iteratively until the whole model is filled.

Advancing front is robust in triangle/tetrahedron mesh generation. However the closure of the front is not guaranteed when extending to quadrilateral and hexahedron meshes, especially when the model contains multiple holes. The mesh quality near the boundary is usually good, while sometimes distorted elements will appear in the interior part. This method is time consuming as a new front needs to be calculated



(a) Quadrilateral background grid

(b) Triangle background grid

(c) Hexagon background grid

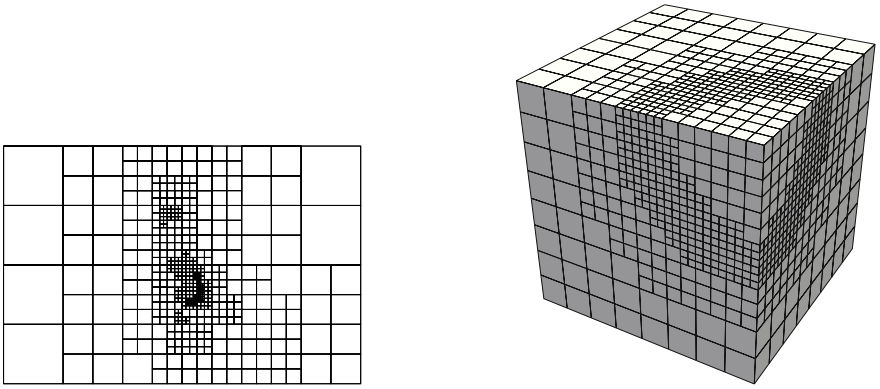
Figure 2.19: Uniform background grid

in each step. Advancing front is well-developed for triangle/tetrahedron meshes, while the application in quadrilateral/hexahedron mesh generation is still an open research topic (Armstrong et al., 2015; Zhou et al., 2016).

2.2.4 Grid based method

Grid based method is a large group of methods using a background grid to discretize the problem domain. The background grid can be composed of triangle, quadrilateral, hexagon and polygon in 2D, tetrahedron, hexahedron and polyhedron in 3D. The background grid can be used in analysis directly by selecting the elements inside the model (Giovannelli et al., 2017). Alternatively, the background grid can be fitted into the geometry by trimming or projection (Karman, 2004). Some typical uniform background grids are shown in Fig 2.19. An algorithm based on uniform triangle grid was presented in Thacker et al. (1980) to study the storm surge. A grid based hexahedron mesh generation algorithm was developed in Schneiders (1996).

Hierarchical background grids are more popular in FEM as local mesh refinement can be implemented. Quadtree and octree background grids are shown in Fig 2.20. A square/cube is recursively subdivided into 4/8 smaller squares/cubes, until certain resolution is reached. Mesh generation based on quadtree algorithm was developed in Yerry and Shephard (1983). The extension to 3D problems was presented in Yerry and Shephard (1984); Shephard and Georges (1991). Several templates were proposed



(a) Quadtree background grid (b) Octree background grid

Figure 2.20: Hierarchical background grid

to make the meshes conformal to the geometry.

There are many advantages of the grid based method. As the initial background grid is independent of the model, theoretically it is robust for arbitrary complex geometry. The mesh size can be controlled easily if hierarchical background grid is used. The quadtree/octree data structure is efficient in data retrieval and suitable for parallel processing (Samet, 1984). However, the trimming of the background grid on the boundary is usually difficult to implement (Karman and Betro, 2008). Not to mention that the element quality on the boundary is usually poor. The handling of sharp edges and corners in grid based method is another difficulty (Marechal, 2009). In conventional FEM, shape function for arbitrary polytope element is not available, therefore triangulation/tetrahedralization is usually required after the trimming of the background grid (Ghisi et al., 2014). Recently, an octree mesh generation method for multiple materials was reported in Fujita et al. (2016). A conforming mesh generation scheme based on octree algorithm was proposed in Kudela et al. (2016). It was claimed that the algorithm works for complex geometry with presence of sharp edges. Octree mesh was used in adaptive mesh refinement in Hasbestan and Senocak (2018). The application in biomedical science was discussed in Bols et al. (2016); Katsushima et al. (2018).

Octree based algorithm is applied in this research. The implementation will be

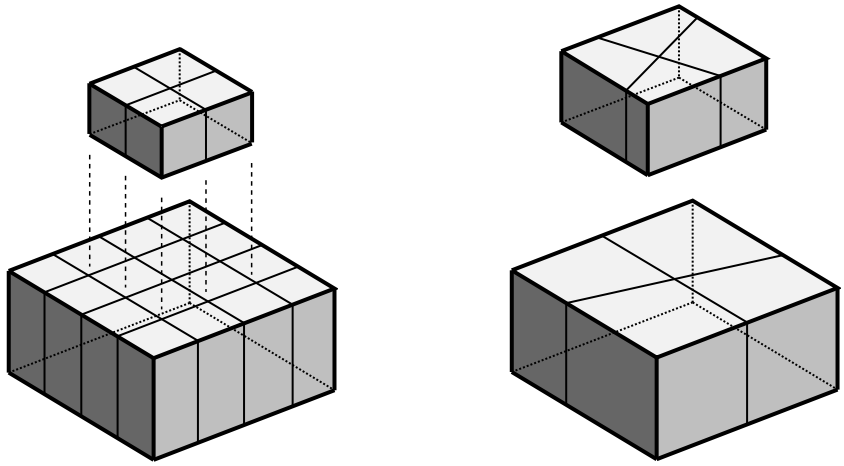
presented in Chapter 4.

2.2.5 Combination of methods

In engineering practice, the methods introduced above are usually combined based on the features of the geometry. For example, grid based method can be used to generate initial point set for Delaunay triangulation (Schroeder and Shephard, 1990). The mesh density can be controlled easily. Delaunay triangulation can be combined with the advancing front method (Mavriplis, 1995). The boundary layers are generated by advancing front and the interior nodes are connected using Delaunay triangulation. There is no need for boundary recovery and the efficiency is improved. Alternatively, the grid based method can be combined with the advancing front method (McMorris and Kallinderis, 1997). While a few layers of high quality elements can be generated near the boundary, the background grid can fill the interior of the model fast and robust (Demargne et al., 2014). Octree based polyhedron mesh can be mapped into a different geometry (Hu and Zhang, 2016). A complex geometry can be partitioned using medial axis/surface (Owen, 1998), and each part with simple geometry can be meshed using different methods.

2.3 Coupling of non-matching meshes

In the numerical analysis of complex and large scale structures, it is desirable to spatially divide the problem domain into several simpler subdomains. The meshes of these simpler subdomains are usually easier to generated using the methods introduced in Section 2.2. These subdomains are only connected by their interfaces. The interface meshes are matching if the mesh topologies and element shape functions are the same on both sides of the interface (Fig. 2.21a), otherwise they are non-matching (Fig. 2.21b). Obviously, the use of non-matching meshes offer greater flexibility in practice as they



(a) Matching meshes (b) Non-matching meshes

Figure 2.21: Matching and non-matching meshes in 3D

enable users to generate meshes for different subdomains independently. This advantage facilitates integrating geometric modeling and structure analysis by reducing the human effort required in the mesh generation for complex models. The mesh of one subdomain can be modified without affecting the meshes of the other subdomains. Parallel processing can be employed to generate meshes for the subdomains. Therefore the structure analysis in engineering design process can be largely simplified and accelerated.

These independent meshes need to be coupled (or "glued") together when a numerical analysis is performed. A large number of techniques have been developed to couple non-matching meshes. They have been applied to a variety of problems such as domain decomposition (Becker et al., 2003), contact mechanics (Laursen et al., 2012), fluid-structure interaction (Bazilevs et al., 2012), multi-scale analysis (Unger and Eckardt, 2011), etc.

2.3.1 Early developments

A variety of methods have been developed for the coupling of non-matching meshes. The nearest neighbor interpolation (de Boer et al., 2007) is among the earliest methods. It searches for the nearest node pairs from the two meshes. The displacements of a node

pair are set to be equal. Some modifications based on the nearest neighbor interpolation were developed using Gauss interpolation and radial basis functions. In cases where the nodes in a coarser mesh coincide with the nodes in a finer mesh, the shape function of elements in the coarser mesh can be used to calculate displacement of nodes in the finer mesh (Demkowicz et al., 1985). These methods, although simple to implement, often lead to deterioration of accuracy.

2.3.2 Mortar element method

Mortar element method was developed by Maday et al. (1988). A set of mortar elements, which are lines in 2D and surfaces in 3D, are inserted between the two non-matching meshes. The displacements in the two meshes and the mortar elements need to minimize the total potential energy of the system (Pantano and Averill, 2002). It can be solved by introducing Lagrange multipliers as interaction forces (Puso, 2004; Flemisch et al., 2005). Some variants were also developed using dual space (Wohlmuth, 2000) and primal hybrid formulation (Belgacem, 1999). Methods based on the Lagrange multiplier technique enjoy the advantages of satisfying the continuity conditions in a variational sense and the exact displacement constraints. On the other hand, this method introduces additional degrees of freedom and must be subject to the *inf-sup* or Ladyzhenskaya-Babuška-Brezzi condition (Brezzi and Bathe, 1990). The stiffness matrix obtained by this method is not positive-definite. Penalty method (Pantano and Averill, 2002, 2007) introduces no extra variables into the equation system and is simple to implement. However, the penalty parameters are difficult to choose. This method only applies the displacement constraints approximately. A combination of penalty and the Lagrange multipliers methods was developed in Simo et al. (1985). It is demonstrated that this method has higher accuracy than the penalty method.

2.3.3 Nitsche's method

Nitsche's method was developed by Nitsche (1971) for imposing essential boundary conditions weakly in the FEM. A variant of Nitsches method was proposed in Hansbo and Hansbo (2002) for elliptic interface problems. It was claimed that optimal convergence rate was obtained. The application in domain decomposition was investigated by Becker et al. (2003). This method was utilized for coupling fluid and structure meshes in vibration analysis by Hansbo and Hermansson (2003) and Burman and Fernandez (2014). A summary of Nitsche's method in a variety of problems was presented in Hansbo (2005). Nitsche's method was applied in Apostolatos et al. (2014) to handle non-matching meshes in 2D isogeometric analysis. It was extended to couple non-uniform rational B-spline patches in 3D by Nguyen et al. (2014). The application of Nitsche's method to contact problems was investigated in Hu et al. (2018).

2.3.4 Non-conventional elements

Another class of coupling strategies consists of methods based on non-conventional types of elements (Lim et al., 2007; Kim, 2008; Sohn and Jin, 2015). These methods modify the element formulation in order to convert the non-matching interface into a matching one. The stiffness matrix is positive-definite and banded. The compatibility of displacement along the interface is also satisfied precisely. Coupling finite element method was presented by Bitencourt Jr. et al. (2015) to build new interface element using standard finite element shape function as a reference. It inserts one additional node each time to the original element to construct a new interface element. In the interface element method (Kim, 2002), polygon and polyhedron elements of arbitrary shapes are constructed on the surfaces of non-matching meshes as interface elements. This method was first utilized to connect 3D non-matching meshes which share a flat interface (Lim et al., 2007). The capability of the interface element method was improved in handling curved interface by employing the moving least square approximation (Kim,

2008). Besides, polyhedron elements (Sohn and Jin, 2015) are developed to deal with non-matching interfaces based on the cell-based smoothed finite element method (Liu et al., 2007). Such a coupling strategy can potentially be employed with other formulations of polytope elements (Beirao da Veiga et al., 2014; Natarajan et al., 2015, 2014, 2017; Francis et al., 2017; Chen et al., 2001, to name a few).

2.4 Numerical simulation of virtual reality models

Virtual reality technique has been widely used in industrial design (Jayaram et al., 1997; Purschke et al., 1998; Kan et al., 2001). With the rapid development of modern computers, it is possible to generate virtual reality models for a whole city (Ennis and Lindsay, 2000). Virtual reality was employed to model the transportation network in Sun et al. (2002). The concept of "virtual city" for 3D modeling and visualization was presented in Dollner et al. (2006). A case study of virtual city reconstruction of ancient Rome was reported in Dylla et al. (2008). Different modern techniques to construct virtual city model were discussed in Singh et al. (2013). However, until now, the virtual city technique is mainly used for geometric modeling rather than numerical analysis. Such numerical analysis on urban scale models will have significant impact on the study of earthquake (Taborda and Bielak, 2011b,a), aerodynamics (Kanda et al., 2013), fluid dynamics (Blocken, 2015; Maronga et al., 2015; Hasbestan and Senocak, 2017) and many others (Ichimura et al., 2015; Hereth et al., 2017).

2.5 Adaptive analysis

Adaptive analysis is a technique to refine a mesh based on the error estimation in a numerical analysis (Zienkiewicz and Zhu, 1987; Zhu and Zienkiewicz, 1988). The mesh is only refined in regions where estimated errors are higher than a threshold, therefore the convergence rate is expected to be faster than a uniform refinement. A review of

the development of adaptive techniques can be found in Li and Bettess (1997). A large number of error indicators have been developed (Ainsworth and Oden, 1993; Babuska et al., 1994). Superconvergent patch was studied in adaptive analysis by Zienkiewicz and Zhu (1992a,b). Adaptive technique was later applied to study linear (Zienkiewicz et al., 1999), elasto-plastic (Boroomand and Zienkiewicz, 1999), acoustic (Bouillard and Ihlenburg, 1999) and many other problems in the FEM (Ortiz and Quigley Iv, 1991). Adaptive analysis has been applied to other numerical methods as well. In Deeks and Wolf (2002a) a stress recovery and error estimation method was developed in the SBFEM. Adaptive technique was investigated in meshfree particle method by Rabczuk and Belytschko (2005). Adaptivity in isogeometric analysis was studied in Dorfel et al. (2010); Vuong et al. (2011). A coupled FEM-BEM was used in adaptive analysis in Aurada et al. (2013). Recently, a novel error indicator without stress recovery was proposed in Song et al. (2018) using the SBFEM.

In conventional FEM usually the whole mesh needs to be regenerated in each step, even if the refinement is only in a local region. An adaptive refinement using quadtree mesh was developed in Tabarraei and Sukumar (2005). A graph-based implementation of quadtree meshes for adaptive analysis was reported in Burgarelli et al. (2006). It was shown that the storage requirements and computational cost were low compared with other methods. Unstructured polygon mesh was refined using a polytree algorithm in Spring et al. (2014). A novel polytree based polygonal finite element method was proposed in Nguyen-Xuan (2017); Nguyen-Xuan et al. (2017a,b); Chau et al. (2018). High order element was employed in adaptive analysis by Grayver and Kolev (2015).

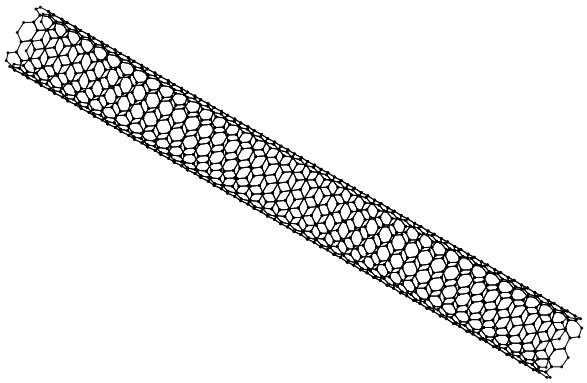


Figure 2.22: Carbon nanotube

2.6 Mesh generation of fiber reinforced materials and structures

Fiber reinforced materials have been widely used in engineering. It is reported that the mechanical property of a matrix material can be significantly improved by adding only a small amount of fibers (Spanos and Kontsos, 2008; Figiel and Buckley, 2009; Djebara et al., 2016). A new class of fiber, carbon nanotube (CNT), has attracted increasingly attention in the last decade. The geometry of a CNT can be considered as a wrapped hexagonal mesh in nanoscale (Shokrieh and Rafiee, 2010) (see Fig 2.22). It has shown exceptional mechanical properties (elastic modulus of 1TPa and tensile strength of 150GPa (Tserpes et al., 2008; Tserpes and Chanteli, 2013)).

The evaluation of the material property of fiber reinforced materials becomes an important task. Traditionally the material properties are evaluated via experiments, Experiments can only be performed after specimens have been manufactured. The process is time-consuming and expensive. An analytical method to evaluate of the elastic properties of composite materials can be found in Hill (1963). Halpin-Tsai equation (Halpin, 1969; Affdl and Kardos, 1976) and Mori-Tanaka equation (Mori and Tanaka, 1973; Benveniste, 1987) were later proposed to study fiber reinforced materials.

In recent years, FEM has been widely used as an alternative to the experiment and

the analytical methods. In FEM the fibers are generally modeled as 1D tension bars or beams. FEM has a variety of advantages, such as the problems from macro to micro even nano scale can be studied within the same framework (Sheng et al., 2004). Both aligned (Sheng et al., 2004) and randomly distributed (Odegard et al., 2005; Hbaieb et al., 2007) fibers can be modeled. Fibers with different shapes and aspect ratios can be considered (long cylinder, sphere and thin discs (Mortazavi et al., 2013)). The interphase effect between fiber and matrix can also be investigated (Odegard et al., 2005; Vo et al., 2018). Parametric study can be easily performed without excessive cost (Vo et al., 2018). In FEM a CNT can be simplified to an equivalent beam in micro scale considering the tension, bending and torsion stiffness of the structure in nano scale (Ayatollahi et al., 2011). The simplified beam is then added to the matrix to evaluate the overall materiel properties. Alternatively, the hexagonal mesh of the CNT can be coupled with the matrix directly, which has been reported in Banerjee et al. (2016). However there are some difficulties in the modeling of fiber reinforced materials in FEM. The mesh generation is difficult because the mesh usually needs to be aligned with the fibers. Quadrilateral meshes generated for 2D problem are usually distorted, not to mention that only tetrahedron meshes are possible in 3D (Hbaieb et al., 2007; Vo et al., 2018). Sometimes the locations of the fibers are prescribed to be non-overlapping (Silani et al., 2014), which may not reflect the real situation.

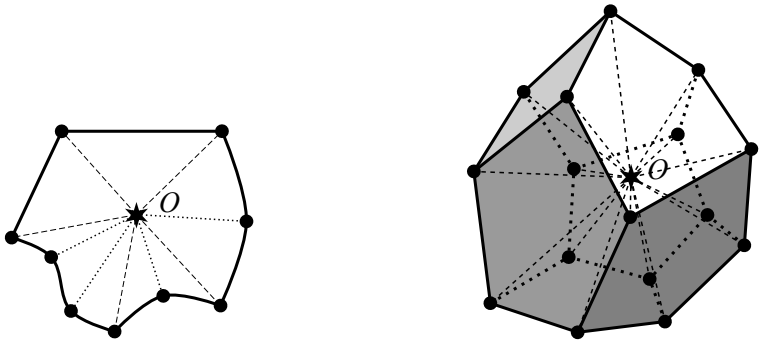
Chapter 3

Scaled boundary finite element method

In this chapter, the scaled boundary finite element method (SBFEM) is introduced. The SBFEM is a semi-analytical method recently developed in the field of computational mechanics. In this method, only the boundary of the domain needs to be discretized. Polygons and polyhedrons with arbitrary number of nodes, edges and faces can be used in an analysis, therefore this method offers greater flexibility in mesh generation and domain decomposition. The fundamental theory of the scaled boundary finite element is presented in Song (2018) accompanied by a computer program written in MATLAB.

3.1 Basic formulations of the scaled boundary finite element method

In this section, the basic formulations of the scaled boundary finite element method in 3D are derived using the virtual work principle. The formulations in 2D problems and unified 2D/3D formulations can be found in Song (2018).



(a) S-domain in 2D (b) S-domain in 3D

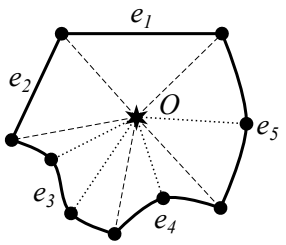
Figure 3.1: S-domains in 2D and 3D

3.1.1 Modeling of geometry in scaled boundary coordinates

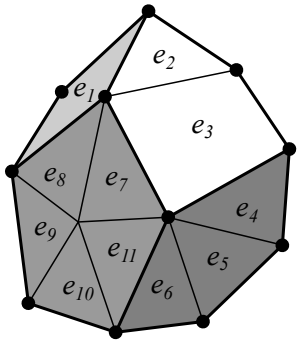
3.1.1.1 S-domains and S-elements

In the SBFEM, polytope elements are utilized to model the problem domain. There is no restriction on the number of nodes, edges and faces in the polytope elements. The only requirement is that the whole boundary of the polytope element must be directly visible from at least one point inside. Such polytope element is defined as an S-domain. The point is defined as scaling center (point O in Fig. 3.1a and Fig. 3.1b). The visibility angle is the minimum angle formed by the boundary and the line connecting the scaling center to the boundary (dash lines in Fig. 3.1a and Fig. 3.1b). The visibility angle should not be too small otherwise the Jacobian matrix of coordinate transformation in Eq. (3.13) becomes ill-conditioned. It is discussed in Song (2018) that an S-domain is closely related to a star polytope in mathematics. The scaling requirement can always be satisfied by subdivision of the polytope elements.

Only the boundary of an S-domain needs to be discretized. The elements on the boundary are standard isoparametric elements in the FEM, which are denoted as e. In 2D the boundary of a polygon is discretized using 1D line elements ($e_1, e_2, \ldots, e_5$ in Fig. 3.2a), while in 3D the boundary of a polyhedron is discretized using triangle and quadrilateral elements ($e_1, e_2, \ldots$ in Fig. 3.2b). The S-domain and its boundary



(a) 2D S-domain discretized using line elements



(b) 3D S-domain discretized using triangle and quadrilateral elements

Figure 3.2: Boundary discretizations of the S-domains

discretization define an S-element.

The coordinates of a point on the boundary can be interpolated using finite element shape function in 2D (Fig. 3.3).

$$x_b(\eta,\zeta) = \sum_{i=1}^{ne} N_i(\eta,\zeta)\, x_i = [N(\eta,\zeta)]\{x\}, \tag{3.1a}$$

$$y_b(\eta,\zeta) = \sum_{i=1}^{ne} N_i(\eta,\zeta)\, y_i = [N(\eta,\zeta)]\{y\}, \tag{3.1b}$$

$$z_b(\eta,\zeta) = \sum_{i=1}^{ne} N_i(\eta,\zeta)\, z_i = [N(\eta,\zeta)]\{z\}, \tag{3.1c}$$

where $\{x\}$, $\{y\}$ and $\{z\}$ are the coordinates of nodes of the element on the boundary. ne is the number of nodes of the element. $N_i(\eta,\zeta)$ is the shape function related to node i. $[N(\eta,\zeta)]$ is the shape functions in matrix form

$$[N(\eta,\zeta)] = \begin{bmatrix} N_1(\eta,\zeta) & N_2(\eta,\zeta) & \dots \end{bmatrix}. \tag{3.2}$$

(a) Isoparametric linear triangle element

(b) Isoparametric linear quadrilateral element

Figure 3.3: Examples of isoparametric elements

The shape functions of an isoparametric linear triangle element are (Fig. 3.3a)

$$N_1(\eta,\zeta) = 1 - \eta - \zeta, \tag{3.3a}$$

$$N_2(\eta,\zeta) = \eta, \tag{3.3b}$$

$$N_3(\eta,\zeta) = \zeta. \tag{3.3c}$$

The shape functions of an isoparametric linear quadrilateral element are (Fig. 3.3b)

$$N_1(\eta,\zeta) = \frac{1}{4}(1-\eta)(1-\zeta), \tag{3.4a}$$

$$N_2(\eta,\zeta) = \frac{1}{4}(1+\eta)(1-\zeta), \tag{3.4b}$$

$$N_3(\eta,\zeta) = \frac{1}{4}(1-\eta)(1+\zeta), \tag{3.4c}$$

$$N_4(\eta,\zeta) = \frac{1}{4}(1+\eta)(1+\zeta). \tag{3.4d}$$

3.1.1.2 Scaled boundary transformation

In this section, the scaled boundary coordinate transformation of an S-domain in 3D is presented.

The position vector of an arbitrary point is

$$\mathbf{r} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}, \tag{3.5}$$

where **i**, **j** and **k** are unit vectors in Cartesian coordinates. The spherical coordinates of the point are

$$r = \sqrt{x^2 + y^2 + z^2}, \tag{3.6a}$$

$$\theta = \arctan \frac{y}{x}, \tag{3.6b}$$

$$\phi = \arccos \frac{z}{r}. \tag{3.6c}$$

The position vector of a point on the boundary is written as

$$\mathbf{r}_b = x_b\mathbf{i} + y_b\mathbf{j} + z_b\mathbf{k}. \tag{3.7}$$

where x_b, y_b and z_b are the coordinates of the point on the boundary. Suppose the position vector of the scaling center is $\mathbf{r}_0$, an arbitrary point inside the S-domain is represented by

$$\mathbf{r} = \xi\left(\mathbf{r}_b - \mathbf{r}_0\right) + \mathbf{r}_0, \tag{3.8}$$

where ξ is a value between $[0, 1]$. Since the whole boundary is directly visible from the scaling center, each point $\mathbf{r}_b$ on the boundary is connected to $\mathbf{r}_0$ by a unique line. Therefore, each point can be uniquely defined by a pair of $\mathbf{r}_b$ and ξ . It can be written as x, y and z components in Cartesian coordinates

$$x = \xi\left(x_b - x_0\right) + x_0, \tag{3.9a}$$

$$y = \xi\left(y_b - y_0\right) + y_0, \tag{3.9b}$$

$$z = \xi\left(z_b - z_0\right) + z_0. \tag{3.9c}$$

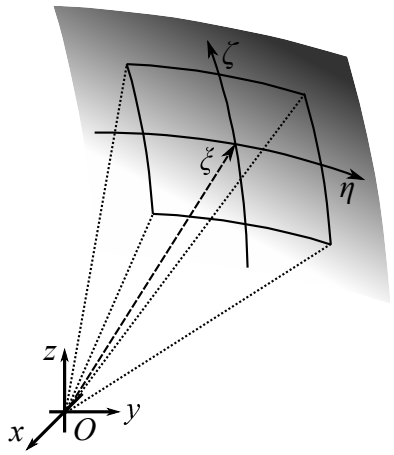


Figure 3.4: Scaled boundary transformation of a surface element

The scaled boundary transformation of a linear quadrilateral element is shown in Fig. 3.4. Substituting Eq. 3.1 into Eq. 3.9, a point inside an S-domain can be represented using scaled boundary coordinates ξ, η and ζ

$$x = \xi\left(\left[N\left(\eta,\zeta\right)\right]\left\{x\right\} - x_0\right) + x_0, \tag{3.10a}$$

$$y = \xi\left(\left[N\left(\eta,\zeta\right)\right]\left\{y\right\} - y_0\right) + y_0, \tag{3.10b}$$

$$z = \xi\left(\left[N\left(\eta,\zeta\right)\right]\left\{z\right\} - z_0\right) + z_0. \tag{3.10c}$$

Applying the chain rule, the partial differential operators with respect to the scaled boundary coordinates are expressed as

$$\frac{\partial}{\partial\xi} = \frac{\partial}{\partial x}\frac{\partial x}{\partial\xi} + \frac{\partial}{\partial y}\frac{\partial y}{\partial\xi} + \frac{\partial}{\partial z}\frac{\partial z}{\partial\xi}, \tag{3.11a}$$

$$\frac{\partial}{\partial\eta} = \frac{\partial}{\partial x}\frac{\partial x}{\partial\eta} + \frac{\partial}{\partial y}\frac{\partial y}{\partial\eta} + \frac{\partial}{\partial z}\frac{\partial z}{\partial\eta}, \tag{3.11b}$$

$$\frac{\partial}{\partial\zeta} = \frac{\partial}{\partial x}\frac{\partial x}{\partial\zeta} + \frac{\partial}{\partial y}\frac{\partial y}{\partial\zeta} + \frac{\partial}{\partial z}\frac{\partial z}{\partial\zeta}. \tag{3.11c}$$

It can be written in matrix form as

$$\left\{ \begin{array}{c} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{array} \right\} = [J] \left\{ \begin{array}{c} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{array} \right\}, \tag{3.12}$$

where $[J]$ is the Jacobian matrix

$$[J] = \left[\begin{array}{ccc} x_{,\xi} & y_{,\xi} & z_{,\xi} \\ x_{,\eta} & y_{,\eta} & z_{,\eta} \\ x_{,\zeta} & y_{,\zeta} & z_{,\zeta} \end{array} \right]. \tag{3.13}$$

Inverting Eq. 3.12, the partial derivatives of with respect to Cartesian coordinates can be written as

$$\left\{ \begin{array}{c} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{array} \right\} = [J]^{-1} \left\{ \begin{array}{c} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{array} \right\}, \tag{3.14}$$

In Cartesian coordinates, an infinitesimal increase in the displacement can be written as

$$\mathrm{d}u = \frac{\partial u}{\partial x}\mathrm{d}x + \frac{\partial \mathrm{u}}{\partial \mathrm{y}}\mathrm{d}y + \frac{\partial \mathrm{u}}{\partial \mathrm{z}}\mathrm{d}z = \left[\begin{array}{ccc} \mathrm{d}x & \mathrm{d}y & \mathrm{d}z \end{array} \right] \left\{ \begin{array}{c} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial u}{\partial z} \end{array} \right\}. \tag{3.15}$$

Similarly, in scaled boundary coordinates, $\mathrm{d}u$ can be expressed as

$$\mathrm{d}u = \frac{\partial u}{\partial \xi}\mathrm{d}\xi + \frac{\partial \mathrm{u}}{\partial \eta}\mathrm{d}\eta + \frac{\partial \mathrm{u}}{\partial \zeta}\mathrm{d}\zeta = \left[\begin{array}{ccc} \mathrm{d}\xi & \mathrm{d}\eta & \mathrm{d}\zeta \end{array} \right] \left\{ \begin{array}{c} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial u}{\partial \zeta} \end{array} \right\}. \tag{3.16}$$

Comparing Eq. 3.15 and Eq. 3.16,

$$\left[\begin{array}{ccc} \mathrm{d}x & \mathrm{d}y & \mathrm{d}z \end{array}\right] \left\{\begin{array}{c} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{array}\right\} = \left[\begin{array}{ccc} \mathrm{d}\xi & \mathrm{d}\eta & \mathrm{d}\zeta \end{array}\right] \left\{\begin{array}{c} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{array}\right\}. \tag{3.17}$$

Considering Eq. 3.12,

$$\left[\begin{array}{ccc} \mathrm{d}x & \mathrm{d}y & \mathrm{d}z \end{array}\right] = \left[\begin{array}{ccc} \mathrm{d}\xi & \mathrm{d}\eta & \mathrm{d}\zeta \end{array}\right] [J]\,. \tag{3.18}$$

The partial derivatives in the Jacobian matrix are obtained from Eq. (3.9) and Eq. (3.1)

$$x_{,\xi} = x_b - x_0 \qquad y_{,\xi} = y_b - y_0 \qquad z_{,\xi} = z_b - z_0, \tag{3.19a}$$

$$x_{,\eta} = \xi x_{b,\eta} \qquad y_{,\eta} = \xi y_{b,\eta} \qquad z_{,\eta} = \xi z_{b,\eta}\,, \tag{3.19b}$$

$$x_{,\zeta} = \xi x_{b,\zeta} \qquad y_{,\zeta} = \xi y_{b,\zeta} \qquad z_{,\zeta} = \xi z_{b,\zeta}\,. \tag{3.19c}$$

The Jacobian matrix can be written as a product of two matrices

$$[J] = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & \xi & 0 \\ 0 & 0 & \xi \end{array}\right] \left[\begin{array}{ccc} x_b - x_0 & y_b - y_0 & z_b - z_0 \\ x_{b,\eta} & y_{b,\eta} & z_{b,\eta} \\ x_{b,\zeta} & y_{b,\zeta} & z_{b,\zeta} \end{array}\right]. \tag{3.20}$$

Substitute Eq. (3.20) into Eq. (3.14),

$$\left\{\begin{array}{c} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{array}\right\} = [J_b]^{-1} \left\{\begin{array}{c} \frac{\partial}{\partial \xi} \\ \frac{1}{\xi}\frac{\partial}{\partial \eta} \\ \frac{1}{\xi}\frac{\partial}{\partial \zeta} \end{array}\right\}, \tag{3.21}$$

where $[J_b] = [J(\xi = 1)]$ is the Jacobian matrix on the boundary

$$[J_b] = \begin{bmatrix} x_b - x_0 & y_b - y_0 & z_b - z_0 \\ x_{b,\eta} & y_{b,\eta} & z_{b,\eta} \\ x_{b,\zeta} & y_{b,\zeta} & z_{b,\zeta} \end{bmatrix}. \tag{3.22}$$

The entries of the Jacobian matrix on the boundary $[J_b]$ are obtained from Eq. (3.1) as

$$x_b - x_0 = [N]\{x\} \qquad y_b - y_0 = [N]\{y\} \qquad z_b - z_0 = [N]\{z\}, \tag{3.23a}$$

$$x_{b,\eta} = [N_{,\eta}]\{x\} \qquad y_{b,\eta} = [N_{,\eta}]\{y\} \qquad z_{b,\eta} = [N_{,\eta}]\{z\}, \tag{3.23b}$$

$$x_{b,\zeta} = [N_{,\zeta}]\{x\} \qquad y_{b,\zeta} = [N_{,\zeta}]\{y\} \qquad z_{b,\zeta} = [N_{,\zeta}]\{z\}. \tag{3.23c}$$

The determinant of the matrix $[J_b]$ is

$$\begin{aligned} |J_b| = {} & (x_b - x_0)(y_{b,\eta}\, z_{b,\zeta} - z_{b,\eta}\, y_{b,\zeta}) \\ & + (y_b - y_0)(z_{b,\eta}\, x_{b,\zeta} - x_{b,\eta}\, z_{b,\zeta}) \\ & + (z_b - z_0)(x_{b,\eta}\, y_{b,\zeta} - y_{b,\eta}\, x_{b,\zeta}). \end{aligned} \tag{3.24}$$

The inverse of the Jacobian matrix on the boundary $[J_b]^{-1}$ can be written as

$$[J_b]^{-1} = \begin{bmatrix} j_{11} & j_{12} & j_{13} \\ j_{21} & j_{22} & j_{23} \\ j_{31} & j_{32} & j_{33} \end{bmatrix}, \tag{3.25}$$

where the entries in $[J_b]^{-1}$ are

$$j_{11} = \frac{1}{|J_b|}\left(y_{b,\eta}\, z_{b,\zeta} - z_{b,\eta}\, y_{b,\zeta}\right), \tag{3.26a}$$

$$j_{21} = \frac{1}{|J_b|}\left(z_{b,\eta}\, x_{b,\zeta} - x_{b,\eta}\, z_{b,\zeta}\right), \tag{3.26b}$$

$$j_{31} = \frac{1}{|J_b|}\left(x_{b,\eta}\, y_{b,\zeta} - y_{b,\eta}\, x_{b,\zeta}\right), \tag{3.26c}$$

$$j_{12} = \frac{1}{|J_b|}\left(\left(z_b - z_0\right) y_{b,\zeta} - \left(y_b - y_0\right) z_{b,\zeta}\right), \tag{3.26d}$$

$$j_{22} = \frac{1}{|J_b|}\left(\left(x_b - x_0\right) z_{b,\zeta} - \left(z_b - z_0\right) x_{b,\zeta}\right), \tag{3.26e}$$

$$j_{32} = \frac{1}{|J_b|}\left(\left(y_b - y_0\right) x_{b,\zeta} - \left(x_b - x_0\right) y_{b,\zeta}\right), \tag{3.26f}$$

$$j_{13} = \frac{1}{|J_b|}\left(\left(y_b - y_0\right) z_{b,\eta} - \left(z_b - z_0\right) y_{b,\eta}\right), \tag{3.26g}$$

$$j_{23} = \frac{1}{|J_b|}\left(\left(z_b - z_0\right) x_{b,\eta} - \left(x_b - x_0\right) z_{b,\eta}\right), \tag{3.26h}$$

$$j_{33} = \frac{1}{|J_b|}\left(\left(x_b - x_0\right) y_{b,\eta} - \left(y_b - y_0\right) x_{b,\eta}\right). \tag{3.26i}$$

The partial derivatives in Eq. (3.21) can be rearranged as

$$\begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{Bmatrix} = \begin{Bmatrix} j_{11} \\ j_{21} \\ j_{31} \end{Bmatrix} \frac{\partial}{\partial \xi} + \begin{Bmatrix} j_{12} \\ j_{22} \\ j_{32} \end{Bmatrix} \frac{1}{\xi}\frac{\partial}{\partial \eta} + \begin{Bmatrix} j_{13} \\ j_{23} \\ j_{33} \end{Bmatrix} \frac{1}{\xi}\frac{\partial}{\partial \zeta}. \tag{3.27}$$

3.1.1.3 Geometrical properties in scaled boundary coordinates

The geometrical properties of a 3D S-domain in the scaled boundary coordinates are discussed in this section.

A sector in an S-domain is shown in Fig. 3.5. The tangential vectors of a surface on the boundary can be obtained from the partial derivatives of the position vector $\mathbf{r}_b$

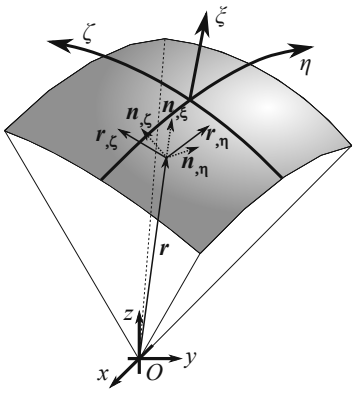


Figure 3.5: Geometrical properties of an S-domain in the scaled boundary coordinates with respect to η and ζ.

$$\mathbf{r}_{b,\eta} = x_{b,\eta}\mathbf{i} + y_{b,\eta}\mathbf{j} + z_{b,\eta}\mathbf{k}, \tag{3.28a}$$

$$\mathbf{r}_{b,\zeta} = x_{b,\zeta}\mathbf{i} + y_{b,\zeta}\mathbf{j} + z_{b,\zeta}\mathbf{k}. \tag{3.28b}$$

The outward normal vector of the surface is calculated by

$$\begin{aligned}
\mathbf{g}^{\xi} &= \mathbf{r}_{b,\eta} \times \mathbf{r}_{b,\zeta} \\
&= \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ x_{b,\eta} & y_{b,\eta} & z_{b,\eta} \\ x_{b,\zeta} & y_{b,\zeta} & z_{b,\zeta} \end{vmatrix} \\
&= (y_{b,\eta}z_{b,\zeta} - z_{b,\eta}y_{b,\zeta})\,\mathbf{i} + (z_{b,\eta}x_{b,\zeta} - x_{b,\eta}z_{b,\zeta})\,\mathbf{j} + (x_{b,\eta}y_{b,\zeta} - y_{b,\eta}x_{b,\zeta})\,\mathbf{k}.
\end{aligned} \tag{3.29}$$

It can be normalized as

$$\mathbf{n}^{\xi} = n_x^{\xi}\mathbf{i} + n_y^{\xi}\mathbf{j} + n_z^{\xi}\mathbf{k} = \frac{\mathbf{g}^{\xi}}{|\mathrm{g}^{\xi}|}, \tag{3.30}$$

where $\left|\mathrm{g}^{\xi}\right|$ is the length of the vector $\mathbf{g}^{\xi}$. The normal vector is written as matrix form

$$\left\{n^{\xi}\right\} = \left\{\begin{array}{c} n_x^{\xi} \\ n_y^{\xi} \\ n_z^{\xi} \end{array}\right\} = \frac{1}{\left|\mathrm{g}^{\xi}\right|} \left\{\begin{array}{c} y_{b,\eta} z_{b,\zeta} - z_{b,\eta} y_{b,\zeta} \\ z_{b,\eta} x_{b,\zeta} - x_{b,\eta} z_{b,\zeta} \\ x_{b,\eta} y_{b,\zeta} - y_{b,\eta} x_{b,\zeta} \end{array}\right\}. \tag{3.31}$$

Similarly, the tangential vectors of an arbitrary surface with a constant ξ is

$$\mathbf{r}_{,\eta} = \xi \mathbf{r}_{b,\eta} = \xi x_{b,\eta}\mathbf{i} + \xi y_{b,\eta}\mathbf{j} + \xi z_{b,\eta}\mathbf{k}, \tag{3.32a}$$

$$\mathbf{r}_{,\zeta} = \xi \mathbf{r}_{b,\zeta} = \xi x_{b,\zeta}\mathbf{i} + \xi y_{b,\zeta}\mathbf{j} + \xi z_{b,\zeta}\mathbf{k}. \tag{3.32b}$$

An infinitesimal area on the surface element is equal to

$$\begin{aligned} \mathrm{d}S^{\xi} &= \left|\mathbf{r}_{,\eta} \times \mathbf{r}_{,\zeta}\right| \mathrm{d}\eta\mathrm{d}\zeta \\ &= \left|\xi\mathbf{r}_{b,\eta} \times \xi\mathbf{r}_{b,\zeta}\right| \mathrm{d}\eta\mathrm{d}\zeta \\ &= \xi^2 \left|\mathrm{g}^{\xi}\right| \mathrm{d}\eta\mathrm{d}\zeta. \end{aligned} \tag{3.33}$$

The tangential vector to the radial direction on the side faces with constant η or ζ is

$$\mathbf{r}_{,\xi} = x_{,\xi}\mathbf{i} + y_{,\xi}\mathbf{j} + z_{,\xi}\mathbf{k}. \tag{3.34}$$

Considering Eq. (3.9), the tangential vector can be written as

$$\mathbf{r}_{,\xi} = \mathbf{r}_b - \mathbf{r}_0 = (x_b - x_0)\,\mathbf{i} + (y_b - y_0)\,\mathbf{j} + (z_b - z_0)\,\mathbf{k}. \tag{3.35}$$

The normal vectors on the side faces are

$$\begin{aligned}\mathbf{g}^{\eta} &= \mathbf{r}_{b,\zeta} \times \mathbf{r}_{,\xi} \\ &= \left((z_b - z_0)\, y_{b,\zeta} - (y_b - y_0)\, z_{b,\zeta}\right) \mathbf{i} + \left((x_b - x_0)\, z_{b,\zeta} - (z_b - z_0)\, x_{b,\zeta}\right) \mathbf{j} \\ &+ \left((y_b - y_0)\, x_{b,\zeta} - (x_b - x_0)\, y_{b,\zeta}\right) \mathbf{k}, \end{aligned} \tag{3.36a}$$

$$\begin{aligned}\mathbf{g}^{\zeta} &= \mathbf{r}_{,\xi} \times \mathbf{r}_{b,\eta} \\ &= \left((y_b - y_0)\, z_{b,\eta} - (z_b - z_0)\, y_{b,\eta}\right) \mathbf{i} + \left((z_b - z_0)\, x_{b,\eta} - (x_b - x_0)\, z_{b,\eta}\right) \mathbf{j} \\ &+ \left((x_b - x_0)\, y_{b,\eta} - (y_b - y_0)\, x_{b,\eta}\right) \mathbf{k}. \end{aligned} \tag{3.36b}$$

The unit normal vectors are calculated as

$$\mathbf{n}^{\eta} = n_x^{\eta}\mathbf{i} + n_y^{\eta}\mathbf{j} + n_z^{\eta}\mathbf{k} = \frac{\mathbf{g}^{\eta}}{|\mathrm{g}^{\eta}|}, \tag{3.37a}$$

$$\mathbf{n}^{\zeta} = n_x^{\zeta}\mathbf{i} + n_y^{\zeta}\mathbf{j} + n_z^{\zeta}\mathbf{k} = \frac{\mathbf{g}^{\zeta}}{|\mathrm{g}^{\zeta}|}. \tag{3.37b}$$

They are written in matrix form as

$$\{n^{\eta}\} = \left\{ \begin{array}{c} n_x^{\eta} \\ n_y^{\eta} \\ n_z^{\eta} \end{array} \right\} = \frac{1}{|\mathrm{g}^{\eta}|} \left\{ \begin{array}{c} (z_b - z_0)\, y_{b,\zeta} - (y_b - y_0)\, z_{b,\zeta} \\ (x_b - x_0)\, z_{b,\zeta} - (z_b - z_0)\, x_{b,\zeta} \\ (y_b - y_0)\, x_{b,\zeta} - (x_b - x_0)\, y_{b,\zeta} \end{array} \right\}, \tag{3.38a}$$

$$\left\{n^{\zeta}\right\} = \left\{ \begin{array}{c} n_x^{\zeta} \\ n_y^{\zeta} \\ n_z^{\zeta} \end{array} \right\} = \frac{1}{|\mathrm{g}^{\zeta}|} \left\{ \begin{array}{c} (y_b - y_0)\, z_{b,\eta} - (z_b - z_0)\, y_{b,\eta} \\ (z_b - z_0)\, x_{b,\eta} - (x_b - x_0)\, z_{b,\eta} \\ (x_b - x_0)\, y_{b,\eta} - (y_b - y_0)\, x_{b,\eta} \end{array} \right\}. \tag{3.38b}$$

The infinitesimal areas on the side faces are determined as

$$\mathrm{d}S^{\eta} = \left|\mathbf{r}_{,\zeta} \times \mathbf{r}_{,\xi}\right| \mathrm{d}\zeta\mathrm{d}\xi = \left|\xi\mathbf{r}_{b,\zeta} \times \mathbf{r}_{,\xi}\right| \mathrm{d}\zeta\mathrm{d}\xi = \xi \left|\mathrm{g}^{\eta}\right| \mathrm{d}\zeta\mathrm{d}\xi, \tag{3.39a}$$

$$\mathrm{d}S^{\zeta} = \left|\mathbf{r}_{,\xi} \times \mathbf{r}_{,\eta}\right| \mathrm{d}\xi\mathrm{d}\eta = \left|\mathbf{r}_{,\xi} \times \xi\mathbf{r}_{b,\eta}\right| \mathrm{d}\xi\mathrm{d}\eta = \xi \left|\mathrm{g}^{\zeta}\right| \mathrm{d}\xi\mathrm{d}\eta. \tag{3.39b}$$

An infinitesimal volume is calculated as

$$\mathrm{d}V = \mathbf{r}_{,\xi} \cdot \left(\mathbf{r}_{,\eta} \times \mathbf{r}_{,\zeta}\right) \mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta = \xi^2 \left(\mathbf{r}_b - \mathbf{r}_0\right) \cdot \left(\mathbf{r}_{b,\eta} \times \mathbf{r}_{b,\zeta}\right) \mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta. \tag{3.40}$$

Considering Eq. (3.28) and Eq. (3.35), the vector product is equal to the determinant of Jacobian matrix on the boundary

$$\left(\mathbf{r}_b - \mathbf{r}_0\right) \cdot \left(\mathbf{r}_{b,\eta} \times \mathbf{r}_{b,\zeta}\right) = \begin{vmatrix} x_b - x_0 & y_b - y_0 & z_b - z_0 \\ x_{b,\eta} & y_{b,\eta} & z_{b,\eta} \\ x_{b,\zeta} & y_{b,\zeta} & z_{b,\zeta} \end{vmatrix} = \left|J_b\right|. \tag{3.41}$$

Therefore, the infinitesimal volume is

$$\mathrm{d}V = \xi^2 \left|J_b\right| \mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\zeta. \tag{3.42}$$

Comparing with Eq. (3.26), the entries in the inverse of Jacobian matrix can be written as

$$\begin{Bmatrix} j_{11} \\ j_{21} \\ j_{31} \end{Bmatrix} = \frac{\left|\mathrm{g}^{\xi}\right|}{\left|J_b\right|} \begin{Bmatrix} n_x^{\xi} \\ n_y^{\xi} \\ n_z^{\xi} \end{Bmatrix}, \tag{3.43a}$$

$$\left\{ \begin{array}{c} j_{12} \\ j_{22} \\ j_{32} \end{array} \right\} = \frac{|\mathrm{g}^{\eta}|}{|J_b|} \left\{ \begin{array}{c} n_x^{\eta} \\ n_y^{\eta} \\ n_z^{\eta} \end{array} \right\}, \tag{3.43b}$$

$$\left\{ \begin{array}{c} j_{13} \\ j_{23} \\ j_{33} \end{array} \right\} = \frac{|\mathrm{g}^{\zeta}|}{|J_b|} \left\{ \begin{array}{c} n_x^{\zeta} \\ n_y^{\zeta} \\ n_z^{\zeta} \end{array} \right\}. \tag{3.43c}$$

The inverse of Jacobian matrix is

$$[J_b]^{-1} = \frac{1}{|J_b|} \left[\begin{array}{ccc} |\mathrm{g}^{\xi}|\, n_x^{\xi} & |\mathrm{g}^{\eta}|\, n_x^{\eta} & |\mathrm{g}^{\zeta}|\, n_x^{\zeta} \\ |\mathrm{g}^{\xi}|\, n_y^{\xi} & |\mathrm{g}^{\eta}|\, n_y^{\eta} & |\mathrm{g}^{\zeta}|\, n_y^{\zeta} \\ |\mathrm{g}^{\xi}|\, n_z^{\xi} & |\mathrm{g}^{\eta}|\, n_z^{\eta} & |\mathrm{g}^{\zeta}|\, n_z^{\zeta} \end{array} \right]. \tag{3.44}$$

Therefore, the partial differential operator in Eq. (3.27) is expressed as

$$\left\{ \begin{array}{c} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{array} \right\} = \frac{|\mathrm{g}^{\xi}|}{|J_b|} \left\{ n^{\xi} \right\} \frac{\partial}{\partial \xi} + \frac{1}{\xi} \left(\frac{|\mathrm{g}^{\eta}|}{|J_b|} \left\{ n^{\eta} \right\} \frac{\partial}{\partial \eta} + \frac{|\mathrm{g}^{\zeta}|}{|J_b|} \left\{ n^{\zeta} \right\} \frac{\partial}{\partial \zeta} \right). \tag{3.45}$$

3.1.2 Governing equations of linear elasticity in scaled boundary coordinates

In 3D elasticity theory, the strain vector $\{\varepsilon\}$ at an arbitrary point is calculated from displacement field $\{u\}$ as

$$\{\varepsilon\} = [L]\{u\}, \tag{3.46}$$

where $[L]$ is the differential operator in 3D

$$[L] = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \end{bmatrix}. \tag{3.47}$$

Substitute the rows in Eq. (3.27) into Eq. (3.47), the differential operator in scaled boundary coordinates is expressed as

$$[L] = [b_1] \frac{\partial}{\partial \xi} + \frac{1}{\xi} \left([b_2] \frac{\partial}{\partial \eta} + [b_3] \frac{\partial}{\partial \zeta} \right), \tag{3.48}$$

where the matrices $[b_1]$, $[b_2]$ and $[b_3]$ are

$$[b_1] = \begin{bmatrix} j_{11} & 0 & 0 \\ 0 & j_{21} & 0 \\ 0 & 0 & j_{31} \\ 0 & j_{31} & j_{21} \\ j_{31} & 0 & j_{11} \\ j_{21} & j_{11} & 0 \end{bmatrix}, \tag{3.49a}$$

$$[b_2] = \begin{bmatrix} j_{12} & 0 & 0 \\ 0 & j_{22} & 0 \\ 0 & 0 & j_{32} \\ 0 & j_{32} & j_{22} \\ j_{32} & 0 & j_{12} \\ j_{22} & j_{12} & 0 \end{bmatrix}, \tag{3.49b}$$

$$[b_3] = \begin{bmatrix} j_{13} & 0 & 0 \\ 0 & j_{23} & 0 \\ 0 & 0 & j_{33} \\ 0 & j_{33} & j_{23} \\ j_{33} & 0 & j_{13} \\ j_{23} & j_{13} & 0 \end{bmatrix}. \tag{3.49c}$$

The strain vector $\{\varepsilon\}$ is expressed as

$$\{\varepsilon\} = [b_1] \frac{\partial \{u\}}{\partial \xi} + \frac{1}{\xi} \left([b_2] \frac{\partial \{u\}}{\partial \eta} + [b_3] \frac{\partial \{u\}}{\partial \zeta} \right). \tag{3.50}$$

The stress vector is obtained considering constitutive model

$$\{\sigma\} = [D] \{\varepsilon\}, \tag{3.51}$$

where $[D]$ is the elasticity matrix. It is expressed for an isotropic material as

$$[D] = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}, \tag{3.52}$$

where E is Young's modulus and ν is Poisson's ratio. On the surfaces with constant ξ, η and ζ, the surface tractions are

$$\left\{t^{\xi}\right\} = \frac{|J_b|}{|\mathrm{g}^{\xi}|} \left[b_1\right]^T \{\sigma\}, \tag{3.53a}$$

$$\left\{t^{\eta}\right\} = \frac{|J_b|}{|\mathrm{g}^{\eta}|} \left[b_2\right]^T \{\sigma\}, \tag{3.53b}$$

$$\left\{t^{\zeta}\right\} = \frac{|J_b|}{|\mathrm{g}^{\zeta}|} \left[b_3\right]^T \{\sigma\}. \tag{3.53c}$$

3.1.3 Semi-analytical representation of displacement and strain fields

An unknown displacement vector $\{u(\xi)\}$ along the radial lines connecting the scaling center and the nodes on the boundary is constructed as

$$\{u(\xi)\} = \begin{bmatrix} u_{1x}(\xi) & u_{1y}(\xi) & u_{1z}(\xi) & u_{2x}(\xi) & u_{2x}(\xi) & u_{2x}(\xi) & \dots \end{bmatrix}^T. \tag{3.54}$$

On the boundary, nodal displacement functions are defined as

$$\{d\} = \{u(\xi = 1)\}. \tag{3.55}$$

The nodal displacement function can be assembled using element connectivity

$$\{u(\xi)\} = \sum_{e} \{u^e(\xi)\}, \tag{3.56}$$

where $\{u^e(\xi)\}$ represents the nodal displacements along the radial lines in a sector corresponding to a surface element e. The symbol $\sum_{e}$ indicates the standard finite element assembly process. In the scaled boundary finite element method, it is assumed that the displacement field at a constant value of ξ can be interpolated using the same shape function as the boundary. Therefore the displacement at an arbitrary point is evaluated in a sector as

$$u_x(\xi,\eta,\zeta) = \sum_{i=1}^{ne} N_i(\eta,\zeta)\, u_{ix}^e(\xi) = [N(\eta,\zeta)]\{u_x^e(\xi)\}, \tag{3.57a}$$

$$u_y(\xi,\eta,\zeta) = \sum_{i=1}^{ne} N_i(\eta,\zeta)\, u_{iy}^e(\xi) = [N(\eta,\zeta)]\left\{u_y^e(\xi)\right\}, \tag{3.57b}$$

$$u_z(\xi,\eta,\zeta) = \sum_{i=1}^{ne} N_i(\eta,\zeta)\, u_{iz}^e(\xi) = [N(\eta,\zeta)]\{u_z^e(\xi)\}, \tag{3.57c}$$

where ne is the number of nodes of element e. It can be written in matrix form as

$$\{u\} = [N_u]\{u^e(\xi)\}, \tag{3.58}$$

where the shape function matrix is

$$[N_u] = \begin{bmatrix} N_1[I] & N_2[I] & \dots \end{bmatrix} = \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & \dots \\ 0 & N_1 & 0 & 0 & N_2 & 0 & \dots \\ 0 & 0 & N_1 & 0 & 0 & N_2 & \dots \end{bmatrix}, \tag{3.59}$$

where $[I]$ is a 3×3 identity matrix. As the displacement at a constant value of ξ is only interpolated by the shape functions of the isoparametric element, the partial derivatives

of the displacement can be written as

$$\frac{\partial \{u\}}{\partial \eta} = [N_u]_{,\eta} \{u^e(\xi)\}, \tag{3.60a}$$

$$\frac{\partial \{u\}}{\partial \zeta} = [N_u]_{,\zeta} \{u^e(\xi)\}. \tag{3.60b}$$

The strain vector is expressed using $\{u^e(\xi)\}$

$$\{\varepsilon\} = [b_1][N_u]\{u^e(\xi)\}_{,\xi} + \frac{1}{\xi}\left([b_2][N_u]_{,\eta} + [b_3][N_u]_{,\zeta}\right)\{u^e(\xi)\}. \tag{3.61}$$

It can be written as

$$\{\varepsilon\} = [B_1]\{u^e(\xi)\}_{,\xi} + \frac{1}{\xi}[B_2]\{u^e(\xi)\}, \tag{3.62}$$

where $[B_1]$ and $[B_2]$ are defined as

$$[B_1] = [b_1][N_u], \tag{3.63a}$$

$$[B_2] = [b_2][N_u]_{,\eta} + [b_3][N_u]_{,\zeta}. \tag{3.63b}$$

Substituting Eq. (3.62) into Eq. (3.51), the stress vector is written as

$$\{\sigma\} = [D]\left([B_1]\{u^e(\xi)\}_{,\xi} + \frac{1}{\xi}[B_2]\{u^e(\xi)\}\right). \tag{3.64}$$

3.1.4 Derivation of the scaled boundary finite element equation by the virtual work principle

In this section, the scaled boundary finite element equation is derived using the virtual work principle (Deeks and Wolf, 2002b). Alternatively, the same equation can be obtained using the minimum potential energy principle or Galerkin's method (Wolf and Song, 2000).

3.1.4.1 Virtual displacement and strain fields

The virtual displacement on the boundary where $\xi = 1$ is written as

$$\{\delta d\} = \{\delta u\,(\xi = 1)\}\,. \tag{3.65}$$

The virtual displacement field at an arbitrary point is

$$\{\delta u\} = [N_u]\,\{\delta u^e\,(\xi)\}\,. \tag{3.66}$$

The virtual strain field produced by the virtual displacement is expressed as

$$\{\delta \varepsilon\} = [B_1]\,\{\delta u^e\,(\xi)\}\,_{,\xi} + \frac{1}{\xi}\,[B_2]\,\{\delta u^e\,(\xi)\}\,. \tag{3.67}$$

3.1.4.2 Nodal force functions

The nodal force functions are introduced on the radial lines connecting the scaling center and the nodes on the boundary. In a sector corresponding to a surface element, $\{q^e\,(\xi)\}$ are statically equivalent to the surface traction on a surface with constant ξ. Considering the virtual work principle,

$$\{\delta u^e\,(\xi)\}^T\,\{q^e\,(\xi)\} = \int_{S^\xi} \{\delta u\}^T\,\left\{t^\xi\right\}\,\mathrm{d}S^\xi. \tag{3.68}$$

Substitute Eq. (3.66) into Eq. (3.68),

$$\{\delta u^e\,(\xi)\}^T\,\{q^e\,(\xi)\} = \int_{S^\xi} \{\delta u^e\,(\xi)\}^T\,[N_u]^T\,\left\{t^\xi\right\}\,\mathrm{d}S^\xi. \tag{3.69}$$

Considering the arbitrariness of the virtual displacement function, the nodal force functions are written as

$$\{q^e(\xi)\} = \int_{S^\xi} [N_u]^T \left\{t^\xi\right\} \mathrm{d}S^\xi. \tag{3.70}$$

Substituting the infinitesimal area in Eq. (3.33) into Eq. (3.70),

$$\{q^e(\xi)\} = \int_{-1}^{+1}\int_{-1}^{+1} [N_u]^T \left\{t^\xi\right\} \xi^2 \left|\mathrm{g}^\xi\right| \mathrm{d}\eta\mathrm{d}\zeta. \tag{3.71}$$

Considering Eq. (3.53a), Eq. (3.63a) and Eq. (3.64), $\{q^e(\xi)\}$ can be written as

$$\begin{aligned}
\{q^e(\xi)\} &= \xi^2 \int_{-1}^{+1}\int_{-1}^{+1} [N_u]^T \left[b^1\right]^T \{\sigma\} |J_b| \,\mathrm{d}\eta\mathrm{d}\zeta \\
&= \xi^2 \int_{-1}^{+1}\int_{-1}^{+1} [B_1]^T \{\sigma\} |J_b| \,\mathrm{d}\eta\mathrm{d}\zeta \\
&= \xi^2 \int_{-1}^{+1}\int_{-1}^{+1} [B_1]^T [D] \left([B_1] \{u(\xi)\}_{,\xi} + \frac{1}{\xi} [B_2] \{u(\xi)\} \right) |J_b| \,\mathrm{d}\eta\mathrm{d}\zeta.
\end{aligned} \tag{3.72}$$

The nodal force vector of a S-domain is assembled using the nodal force vectors of all the surface elements

$$\{q(\xi)\} = \sum_e \{q^e(\xi)\}. \tag{3.73}$$

3.1.4.3 The scaled boundary finite element equation

The virtual work done by the external force is equal to the variation of the strain energy

$$\int_V \{\delta\varepsilon\}^T \{\sigma\} \,\mathrm{d}V = \{\delta d\}^T \{F\}. \tag{3.74}$$

The strain energy is

$$U_{\varepsilon} = \int_{V} \{\delta\varepsilon\}^{T} \{\sigma\} \, \mathrm{d}V. \tag{3.75}$$

The integration is evaluated sector by sector in the S-domain. The volume of a sector corresponding to a surface element e is denoted as V^e.

$$U_{\varepsilon} = \sum \int_{V^e} \{\delta u\,(\xi)\}_{,\xi}^{T} [B_1]^{T} \{\sigma\} \, \mathrm{d}V + \sum \int_{V^e} \{\delta u\,(\xi)\}^{T} [B_2]^{T} \{\sigma\} \frac{1}{\xi} \mathrm{d}V. \tag{3.76}$$

The right hand side can be divided into two parts. Considering Eq. (3.42), the first part of Eq. (3.76) is expressed as

$$U_{\varepsilon 1} = \sum \int_{0}^{1} \{\delta u\,(\xi)\}_{,\xi}^{T} \int_{-1}^{+1} \int_{-1}^{+1} [B_1]^{T} \{\sigma\} \, \xi^2 \, |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta \mathrm{d}\xi. \tag{3.77}$$

The summation can be replaced by the finite element assembly

$$U_{\varepsilon 1} = \int_{0}^{1} \{\delta u\,(\xi)\}_{,\xi}^{T} \sum_{e} \xi^2 \int_{-1}^{+1} \int_{-1}^{+1} [B_1]^{T} \{\sigma\} \, |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta \mathrm{d}\xi. \tag{3.78}$$

Considering Eq. (3.72) and Eq. (3.73)

$$\begin{aligned} U_{\varepsilon 1} &= \int_{0}^{1} \{\delta u\,(\xi)\}_{,\xi}^{T} \sum_{e} \{q^{e}\,(\xi)\} \, \mathrm{d}\xi \\ &= \int_{0}^{1} \{\delta u\,(\xi)\}_{,\xi}^{T} \{q\,(\xi)\} \, \mathrm{d}\xi \\ &= \{\delta u\,(\xi)\}^{T} \{q\,(\xi)\} \, |_{0}^{1} - \int_{0}^{1} \{\delta u\,(\xi)\}^{T} \{q\,(\xi)\}_{,\xi} \, \mathrm{d}\xi \\ &= \{\delta d\}^{T} \{q\,(\xi = 1)\} - \int_{0}^{1} \{\delta u\,(\xi)\}^{T} \{q\,(\xi)\}_{,\xi} \, \mathrm{d}\xi. \end{aligned} \tag{3.79}$$

The second part of Eq. (3.76) is

$$\begin{aligned} U_{\varepsilon 2} &= \sum \int_0^1 \{\delta u(\xi)\}^T \int_{-1}^{+1} \int_{-1}^{+1} [B_2]^T \{\sigma\} \xi |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta \mathrm{d}\xi \\ &= \int_0^1 \{\delta u(\xi)\}^T \sum_e \xi \int_{-1}^{+1} \int_{-1}^{+1} [B_2]^T \{\sigma\} |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta \mathrm{d}\xi. \end{aligned} \tag{3.80}$$

Substitute Eq. (3.79) and Eq. (3.80) back to Eq. (3.76) and Eq. (3.74),

$$\begin{aligned} U_\varepsilon &= \{\delta d\}^T \{q(\xi = 1)\} \\ &\quad - \int_0^1 \{\delta u(\xi)\}^T \left(\{q(\xi)\}_{,\xi} - \sum_e \xi \int_{-1}^{+1} \int_{-1}^{+1} [B_2]^T \{\sigma\} |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta \right) \mathrm{d}\xi \\ &= \{\delta d\}^T \{F\}. \end{aligned} \tag{3.81}$$

The nodal force vector $\{q(\xi)\}$ is statically equivalent to the traction on any surface with a constant ξ. On the boundary, $\{q(\xi)\}$ are equal to the nodal forces of the S-domain.

$$\{F\} = \{q(\xi = 1)\}. \tag{3.82}$$

Eq. (3.81) holds for arbitrary selection of $\{\delta d\}$ satisfying the boundary conditions, therefore the term in the bracket is set to be zero

$$\begin{aligned} \{q(\xi)\}_{,\xi} &= \sum_e \xi \int_{-1}^{+1} \int_{-1}^{+1} [B_2]^T \{\sigma\} |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta \\ &= \sum_e \xi \int_{-1}^{+1} \int_{-1}^{+1} [B_2]^T [D] \left([B_1] \{u(\xi)\}_{,\xi} + \frac{1}{\xi} [B_2] \{u(\xi)\} \right) |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta. \end{aligned} \tag{3.83}$$

Coefficient matrices $[E_0^e]$, $[E_1^e]$ and $[E_2^e]$ are introduced for the surface elements

$$[E_0^e] = \int_{-1}^{+1}\int_{-1}^{+1} [B_1]^T [D] [B_1] |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta, \tag{3.84a}$$

$$[E_1^e] = \int_{-1}^{+1}\int_{-1}^{+1} [B_2]^T [D] [B_1] |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta, \tag{3.84b}$$

$$[E_2^e] = \int_{-1}^{+1}\int_{-1}^{+1} [B_2]^T [D] [B_2] |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta. \tag{3.84c}$$

Coefficient matrices $[E_0]$, $[E_1]$ and $[E_2]$ are assembled from $[E_0^e]$, $[E_1^e]$ and $[E_2^e]$ according to element connectivity

$$[E_0] = \sum_e [E_0^e], \tag{3.85a}$$

$$[E_1] = \sum_e [E_1^e], \tag{3.85b}$$

$$[E_2] = \sum_e [E_2^e]. \tag{3.85c}$$

Eq. (3.72) and Eq. (3.83) can be written as

$$\{q(\xi)\} = [E_0] \xi^2 \{u(\xi)\}_{,\xi} + [E_1]^T \xi \{u(\xi)\}, \tag{3.86a}$$

$$\{q(\xi)\}_{,\xi} = [E_1] \xi \{u(\xi)\}_{,\xi} + [E_2] \{u(\xi)\}. \tag{3.86b}$$

Eliminating the terms with $\{q(\xi)\}$ and $\{q(\xi)\}_{,\xi}$, the scaled boundary finite element equation in displacement is expressed as

$$\left([E_0] \xi^2 \{u(\xi)\}_{,\xi} + [E_1]^T \xi \{u(\xi)\}\right)_{,\xi} = [E_1] \xi \{u(\xi)\}_{,\xi} + [E_2] \{u(\xi)\}. \tag{3.87}$$

The derivatives on the left hand side can be written as

$$2\left[E_0\right]\xi\left\{u\left(\xi\right)\right\}_{,\xi}+\left[E_0\right]\xi^2\left\{u\left(\xi\right)\right\}_{,\xi\xi}+\left[E_1\right]^T\left\{u\left(\xi\right)\right\}+\left[E_1\right]^T\left\{u\left(\xi\right)\right\}_{,\xi} \tag{3.88}$$

$$=\left[E_1\right]\xi\left\{u\left(\xi\right)\right\}_{,\xi}+\left[E_2\right]\left\{u\left(\xi\right)\right\}. \tag{3.89}$$

Rearranging the terms, the scaled boundary finite element equation in displacement is obtained as

$$\left[E_0\right]\xi^2\left\{u\left(\xi\right)\right\}_{,\xi\xi}+\left(2\left[E_0\right]+\left[E_1\right]^T-\left[E_1\right]\right)\xi\left\{u\left(\xi\right)\right\}_{,\xi}+\left(\left[E_1\right]^T-\left[E_2\right]\right)\left\{u\left(\xi\right)\right\}=0. \tag{3.90}$$

3.2 Solution of the scaled boundary finite element equation

In this section, the solution procedure of Eq. (3.90) is presented using eigenvalue decomposition. An alternative approach using Schur decomposition can be found in Song (2018).

3.2.1 Solution procedure for the scaled boundary finite element equations in displacement

Eq. (3.90) is a second order ordinary differential equation. It can be reduced to a set of first order differential equations by eliminating $\left\{u\left(\xi\right)\right\}_{,\xi}$ in Eq. (3.86b). Rearranging Eq. (3.86a) and Eq. (3.86b),

$$\xi\left\{u\left(\xi\right)\right\}_{,\xi}=\xi^{-1}\left[E_0\right]^{-1}\left\{q\left(\xi\right)\right\}-\left[E_0\right]^{-1}\left[E_1\right]^T\left\{u\left(\xi\right)\right\}, \tag{3.91a}$$

$$\xi\left\{q\left(\xi\right)\right\}_{,\xi}=\left[E_1\right]\left[E_0\right]^{-1}\left\{q\left(\xi\right)\right\}-\xi\left(\left[E_1\right]\left[E_0\right]^{-1}\left[E_1\right]^T-\left[E_2\right]\right)\left\{u\left(\xi\right)\right\}. \tag{3.91b}$$

Introducing a pair of new variables $\{\bar{u}(\xi)\}$ and $\{\bar{q}(\xi)\}$

$$\{\bar{u}(\xi)\} = \xi^{0.5}\{u(\xi)\}, \tag{3.92a}$$

$$\{\bar{q}(\xi)\} = \xi^{-0.5}\{q(\xi)\}, \tag{3.92b}$$

the nodal displacement vector on the radial lines $\{u(\xi)\}$ and its derivative $\{u(\xi)\}_{,\xi}$ are written as

$$\{u(\xi)\} = \xi^{-0.5}\{\bar{u}(\xi)\}, \tag{3.93a}$$

$$\{u(\xi)\}_{,\xi} = -0.5\xi^{-1.5}\{\bar{u}(\xi)\} + \xi^{-0.5}\{\bar{u}(\xi)\}_{,\xi}, \tag{3.93b}$$

while nodal force vector $\{q(\xi)\}$ and its derivative $\{q(\xi)\}_{,\xi}$ are written as

$$\{q(\xi)\} = \xi^{0.5}\{\bar{q}(\xi)\}, \tag{3.94a}$$

$$\{q(\xi)\}_{,\xi} = 0.5\xi^{-0.5}\{\bar{q}(\xi)\} + \xi^{0.5}\{\bar{q}(\xi)\}_{,\xi}. \tag{3.94b}$$

Substitute Eq. (3.93) and Eq. (3.94) into Eq. (3.91)

$$\xi\{\bar{u}(\xi)\}_{,\xi} = [E_0]^{-1}\{\bar{q}(\xi)\} - \left([E_0]^{-1}[E_1]^T - 0.5[I]\right)\{\bar{u}(\xi)\}, \tag{3.95a}$$

$$\xi\{\bar{q}(\xi)\}_{,\xi} = \left([E_1][E_0]^{-1} - 0.5[I]\right)\{\bar{q}(\xi)\} - \left([E_1][E_0]^{-1}[E_1]^T - [E_2]\right)\{\bar{u}(\xi)\}. \tag{3.95b}$$

Introducing a new vector $\{X(\xi)\}$ containing both $\{\bar{u}(\xi)\}$ and $\{\bar{q}(\xi)\}$

$$\{X(\xi)\} = \begin{Bmatrix} \{\bar{u}(\xi)\} \\ \{\bar{q}(\xi)\} \end{Bmatrix} = \begin{Bmatrix} \xi^{0.5}\{u(\xi)\} \\ \xi^{-0.5}\{q(\xi)\} \end{Bmatrix}, \tag{3.96}$$

Eq. (3.95) is expressed as

$$\xi \left\{ X\left(\xi \right) \right\}_{,\xi} = \left[Z_p \right] \left\{ X\left(\xi \right) \right\}, \tag{3.97}$$

where $[Z_p]$ is a $2n \times 2n$ matrix (n is the number of degrees of freedom in the S-element)

$$\left[Z_p \right] = \begin{bmatrix} -\left[E_0 \right]^{-1} \left[E_1 \right]^T + 0.5 \left[I \right] & \left[E_0 \right]^{-1} \\ -\left[E_1 \right] \left[E_0 \right]^{-1} \left[E_1 \right]^T + \left[E_2 \right] & \left[E_1 \right] \left[E_0 \right]^{-1} - 0.5 \left[I \right] \end{bmatrix}. \tag{3.98}$$

A well-known solution to Eq. (3.97) is in the form of a power function

$$\left\{ X\left(\xi \right) \right\} = \xi^{\lambda} \left\{ \phi \right\}. \tag{3.99}$$

Eq. (3.97) becomes a eigenvalue problem

$$\left[Z_p \right] \left\{ \phi \right\} = \lambda \left\{ \phi \right\}, \tag{3.100}$$

where $\{\phi\}$ and λ are a pair of eigenvector and eigenvalue of the matrix $[Z_p]$. Concatenate all the eigenvectors and eigenvalues in matrix form,

$$\left[Z_p \right] \left[\Phi \right] = \left[\Phi \right] \left\langle \lambda \right\rangle, \tag{3.101}$$

where $[\Phi]$ is the eigenvector matrix

$$\left[\Phi \right] = \begin{bmatrix} \{\phi_1\} & \{\phi_2\} & \dots & \{\phi_{2n}\} \end{bmatrix}, \tag{3.102}$$

and $\langle \lambda \rangle$ is a diagonal matrix of the eigenvalues

$$\left\langle \lambda \right\rangle = \operatorname{diag} \begin{pmatrix} \lambda_1 & \lambda_2 & \dots & \lambda_{2n} \end{pmatrix}. \tag{3.103}$$

The solution $\{X(\xi)\}$ is expressed as

$$\begin{aligned}\{X(\xi)\} =& \sum_{i=1}^{2n} c_i \xi^{\lambda_i} \{\phi_i\} \\ &= c_1 \xi^{\lambda_1} \{\phi_1\} + c_2 \xi^{\lambda_2} \{\phi_2\} + \ldots + c_{2n} \xi^{\lambda_{2n}} \{\phi_{2n}\}, \end{aligned} \tag{3.104}$$

where $c_1, c_2, \ldots, c_{2n}$ are the integration constants. The eigenvector can be divided into two parts, corresponding to the displacement mode and nodal force mode

$$\{\phi_i\} = \begin{Bmatrix} \left\{\phi_i^{(u)}\right\} \\ \left\{\phi_i^{(q)}\right\} \end{Bmatrix}. \tag{3.105}$$

The displacement and nodal force along the radial lines are expressed as

$$\{u(\xi)\} = \sum_{i=1}^{2n} c_i \xi^{\lambda_i} \left\{\phi_i^{(u)}\right\}, \tag{3.106a}$$

$$\{q(\xi)\} = \sum_{i=1}^{2n} c_i \xi^{\lambda_i} \left\{\phi_i^{(q)}\right\}. \tag{3.106b}$$

The eigenvalues are sorted following descending order of their real parts. The last n eigenvalues are negative. As the displacement has to be finite inside the S-element as ξ approaches 0, the constants from c_{n+1} to c_{2n} have to be 0. The solutions can be expressed as

$$\{u(\xi)\} = \sum_{i=1}^{n} c_i \xi^{\lambda_i} \left\{\phi_i^{(u)}\right\}, \tag{3.107a}$$

$$\{q(\xi)\} = \sum_{i=1}^{n} c_i \xi^{\lambda_i} \left\{\phi_i^{(q)}\right\}. \tag{3.107b}$$

Introducing displacement mode $\left[\Phi_b^{(u)}\right]$ and force mode $\left[\Phi_b^{(q)}\right]$,

$$\left[\Phi_b^{(u)}\right] = \left[\begin{array}{cccc} \left\{\phi_1^{(u)}\right\} & \left\{\phi_2^{(u)}\right\} & \ldots & \left\{\phi_n^{(u)}\right\} \end{array} \right], \tag{3.108a}$$

$$\left[\Phi_b^{(q)}\right] = \left[\begin{array}{cccc} \left\{\phi_1^{(q)}\right\} & \left\{\phi_2^{(q)}\right\} & \ldots & \left\{\phi_n^{(q)}\right\} \end{array} \right]. \tag{3.108b}$$

the solutions are written in matrix form

$$\{u(\xi)\} = \left[\Phi_b^{(u)}\right] \xi^{\langle\lambda_b\rangle} \{c\}, \tag{3.109a}$$

$$\{q(\xi)\} = \left[\Phi_b^{(q)}\right] \xi^{\langle\lambda_b\rangle} \{c\}. \tag{3.109b}$$

where $\{c\}$ is the vector containing the integration constants

$$\{c\} = \left[\begin{array}{cccc} c_1 & c_2 & \ldots & c_n \end{array} \right]^T. \tag{3.110}$$

Eliminating the integration constant in Eq. (3.109a) and Eq. (3.109b),

$$\{q(\xi)\} = \left[\Phi_b^{(q)}\right] \left[\Phi_b^{(u)}\right]^{-1} \{u(\xi)\}. \tag{3.111}$$

On the boundary, $\{q(\xi = 1)\}$ and $\{u(\xi = 1)\}$ represent nodal force and node displacement respectively, comparing with

$$\{F\} = [K]\{d\}, \tag{3.112}$$

the element stiffness matrix can be obtained from displacement mode and nodal force mode directly

$$[K] = \left[\Phi_b^{(q)}\right] \left[\Phi_b^{(u)}\right]^{-1}. \tag{3.113}$$

3.2.2 Assembly of S-elements and solution of global system of equations

Eq. (3.113) is evaluated in each S-element. The global stiffness matrix is assembled in the same way as in finite element method

$$[K_G] = \sum_{Se} [K]\,, \tag{3.114}$$

where $\sum_{Se}$ denotes the assemblage of all the S-elements. The global nodal displacement vector is assembled as

$$\{d_G\} = \sum_{Se} \{d\}\,. \tag{3.115}$$

The nodal force vector is assembled as

$$\{F_G\} = \sum_{Se} \{F\}\,. \tag{3.116}$$

The global equilibrium equation is written as

$$[K_G]\,\{d_G\} = \{F_G\}\,. \tag{3.117}$$

The nodal displacement vector $\{d_G\}$ is divided into two vectors, the unknown displacement $\{d_1\}$ and the prescribed boundary condition $\{d_2\}$. Similarly, the nodal force vector is divided into $\{F_1\}$, where the nodal force is already calculated, and $\{F_2\}$, where reaction forces are to be solved. The stiffness matrix is also divided into 4 blocks.

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{Bmatrix} \{d_1\} \\ \{d_2\} \end{Bmatrix} = \begin{Bmatrix} \{F_1\} \\ \{F_2\} \end{Bmatrix}. \tag{3.118}$$

Rearrange Eq. (3.118) by moving $\{d_2\}$ to the other side,

$$\begin{bmatrix} K_{11} \\ K_{21} \end{bmatrix} \{d_1\} = \begin{Bmatrix} \{F_1\} \\ \{F_2\} \end{Bmatrix} - \begin{bmatrix} K_{12} \\ K_{22} \end{bmatrix} \{d_2\} . \tag{3.119}$$

The unknown displacement is solved from the upper part of the equation,

$$[\ K_{11}\ ]\{d_1\} = \{F_1\} - [\ K_{12}\ ]\{d_2\} . \tag{3.120}$$

The unknown reaction forces are solved after $\{d_1\}$ is obtained

$$\{F_2\} = \begin{bmatrix} K_{21} & K_{22} \end{bmatrix} \begin{Bmatrix} \{d_1\} \\ \{d_2\} \end{Bmatrix} . \tag{3.121}$$

3.2.3 Evaluation of internal displacements and stresses of an S-element

Substituting $\xi = 1$ into Eq. (3.109a), the nodal displacement on the boundary is

$$\{d\} = \left[\Phi_b^{(u)}\right] \{c\} . \tag{3.122}$$

The integration constant vector $\{c\}$ is evaluated as

$$\{c\} = \left[\Phi_b^{(u)}\right]^{-1} \{u_b\} . \tag{3.123}$$

The displacement at the scaling center is only related to the last 3 columns of the displacement mode because all the other terms ξ^{λ_i} vanish as ξ approaches zero.

$$\{u\,(\xi = 0)\} = \sum_{i=n-2}^{n} c_i \left\{\phi_i^{(u)}\right\} . \tag{3.124}$$

The derivative of $\{u(\xi)\}$ with respect to ξ is

$$\{u(\xi)\}_{,\xi} = \sum_{i=1}^{n} c_i \lambda_i \xi^{\lambda_i - 1} \left\{\phi_i^{(u)}\right\}. \tag{3.125}$$

The strain inside a sector corresponding to an element e is

$$\{\varepsilon\} = [B_1]\{u^e(\xi)\}_{,\xi} + \frac{1}{\xi}[B_2]\{u^e(\xi)\}, \tag{3.126}$$

where the nodal displacement on the radial lines of the sector is

$$\{u^e(\xi)\} = \sum_{i=1}^{n} c_i \xi^{\lambda_i} \left\{\phi_i^{(u)e}\right\}. \tag{3.127}$$

It can be written in matrix form as

$$\{u^e(\xi)\} = \left[\phi_i^{(u)e}\right] \xi^{\langle \lambda_b \rangle} \{c\}. \tag{3.128}$$

The strain is expressed using the displacement modes as

$$\{\varepsilon\} = \sum_{i=1}^{n-3} c_i \xi^{\lambda_i - 1} \left(\lambda_i [B_1] + [B_2]\right) \left\{\phi_i^{(u)e}\right\}. \tag{3.129}$$

As the last 3 modes correspond to the rigid body motions, they do not contribute to the strain. Introducing the strain mode $\left\{\phi_i^{(\varepsilon)}\right\}$

$$\left\{\phi_i^{(\varepsilon)}\right\} = \text{concat}\left[\left(\lambda_i [B_1] + [B_2]\right) \left\{\phi_i^{(u)e}\right\}\right], \tag{3.130}$$

where concat $[\bullet]$ stands for array concatenation. The strain vector is written as

$$\{\varepsilon\} = \sum_{i=1}^{n-3} c_i \xi^{\lambda_i - 1} \left\{\phi_i^{(\varepsilon)}\right\}. \tag{3.131}$$

Using the constitutive model in Eq. (3.52), the stress mode is defined as

$$\left\{\phi_i^{(\sigma)}\right\} = \text{concat}\left[[D]\left(\lambda_i\left[B_1\right] + \left[B_2\right]\right)\left\{\phi_i^{(u)e}\right\}\right]. \tag{3.132}$$

Therefore the stress at an arbitrary point inside the S-element is

$$\{\sigma\} = \sum_{i=1}^{n-3} c_i \xi^{\lambda_i - 1}\left\{\phi_i^{(\sigma)}\right\}. \tag{3.133}$$

Substituting Eq. (3.128) and Eq. (3.123) into Eq. (3.58) leads to the displacement field in a sector

$$\{u\} = [N_u]\left[\Phi_b^{(u)e}\right]\xi^{\langle\lambda_b\rangle}\left[\Phi_b^{(u)}\right]^{-1}\{d\}. \tag{3.134}$$

The shape functions of the S-element are give sector by sector as

$$[N_V^e] = [N_u]\left[\Phi_b^{(u)e}\right]\xi^{\langle\lambda_b\rangle}\left[\Phi_b^{(u)}\right]^{-1}. \tag{3.135}$$

3.2.4 Dynamics and vibration analysis

The principle of virtual work is augmented to include inertial force

$$\int_V \{\delta\varepsilon\}^T\{\sigma\}\,\mathrm{d}V + \int_V \{\delta u\}^T\rho\{\ddot{u}\}\,\mathrm{d}V = \{\delta d\}^T\{F\}. \tag{3.136}$$

The increment of the virtual work by inertial force U_ρ is evaluated sector by sector

$$U_\rho = \sum\int_{V^e}\{\delta u\}^T\rho\{\ddot{u}\}\,\mathrm{d}V. \tag{3.137}$$

The acceleration is interpolated using the same shape function as the displacement in Eq. (3.135)

$$\{\ddot{u}\} = [N_V^e]\left\{\ddot{d}\right\}. \tag{3.138}$$

Substitute Eq. (3.138) into Eq. (3.137) and replace the summation with finite element assembly

$$U_\rho = \sum_e \{\delta d^e\}^T \int_{V^e} [N_V^e]^T \rho [N_V^e] \, \mathrm{d}V \left\{ \ddot{d}^e \right\}. \tag{3.139}$$

Assembling the contribution of all sectors, Eq. 3.139 is expressed as

$$U_\rho = \{\delta d\}^T [M] \left\{ \ddot{d} \right\}, \tag{3.140}$$

where the mass matrix is

$$[M] = \sum_e \int_{V^e} [N_V^e]^T \rho [N_V^e] \, \mathrm{d}V. \tag{3.141}$$

The virtual work principle is now expressed as

$$\{\delta d\}^T [K] \{d\} + \{\delta d\}^T [M] \left\{ \ddot{d} \right\} = \{\delta d\}^T \{F\}. \tag{3.142}$$

Considering the arbitrariness of the virtual displacement,

$$[K] \{d\} + [M] \left\{ \ddot{d} \right\} = \{F\}. \tag{3.143}$$

The mass matrix is determined by substituting Eq. (3.135) into Eq. (3.141)

$$\begin{aligned} [M] = & \left[\Phi_b^{(u)} \right]^{-T} \int_0^1 \xi^{\langle \lambda_b \rangle} \sum_e \left[\Phi_b^{(u)e} \right]^T \int_{-1}^{+1} \int_{-1}^{+1} [N_u]^T \rho [N_u] \, \xi^2 \, |J_b| \\ & \times \mathrm{d}\eta \mathrm{d}\zeta \left[\Phi_b^{(u)e} \right] \xi^{\langle \lambda_b \rangle} \mathrm{d}\xi \left[\Phi_b^{(u)} \right]^{-1} \\ = & \left[\Phi_b^{(u)} \right]^{-T} \int_0^1 \xi^{\langle \lambda_b \rangle} \left[\Phi_b^{(u)e} \right]^T [M_0] \left[\Phi_b^{(u)e} \right] \xi^{\langle \lambda_b \rangle} \mathrm{d}\xi \left[\Phi_b^{(u)} \right]^{-1}. \end{aligned} \tag{3.144}$$

The coefficient matrix $[M_0]$ is introduced

$$[M_0] = \sum_e [M_0^e], \tag{3.145}$$

where the element coefficient matrix $[M_0^e]$ is defined as

$$[M_0^e] = \int_{-1}^{+1} \int_{-1}^{+1} [N_u]^T \rho [N_u] |J_b| \, \mathrm{d}\eta \mathrm{d}\zeta. \tag{3.146}$$

In order to perform the integration in Eq. (3.144) analytically, the abbreviation is introduced

$$[m_0] = \left[\Phi_b^{(u)}\right]^T [M_0] \left[\Phi_b^{(u)}\right]. \tag{3.147}$$

The integration in Eq. (3.144) is rewritten as

$$[m] = \int_0^1 \xi^{\langle \lambda_b \rangle} [m_0] \xi^{\langle \lambda_b \rangle} \xi^2 \mathrm{d}\xi. \tag{3.148}$$

The mass matrix $[M]$ is written as

$$[M] = \left[\Phi_b^{(u)}\right]^{-T} [m] \left[\Phi_b^{(u)}\right]^{-1}. \tag{3.149}$$

Each entry of the matrix $[m]$ is integrated analytically

$$\begin{aligned} m_{ij} &= \int_0^1 \xi^{\lambda_{bi}} m_{0ij} \xi^{\lambda_{bj}} \xi^2 \mathrm{d}\xi \\ &= \frac{m_{0ij}}{\lambda_{bi} + \lambda_{bj} + 3}. \end{aligned} \tag{3.150}$$

The natural frequencies and mode shapes are considered without external forces

$$[K] \{d\} + [M] \left\{\ddot{d}\right\} = 0. \tag{3.151}$$

The solution of the system describes a harmonic vibration

$$\{d\} = \{D\} \sin(\omega t), \tag{3.152a}$$

$$\left\{\ddot{d}\right\} = -\omega^2 \{D\} \sin(\omega t), \tag{3.152b}$$

where $\{D\}$ is the amplitude of the nodal displacement and ω is the circular frequency. Substitute Eq. 3.152 into Eq. 3.151 yields a generalized eigenvalue problem

$$([K] - \lambda [M]) \{D\} = 0. \tag{3.153}$$

where the natural frequencies can be calculated from the eigenvalues

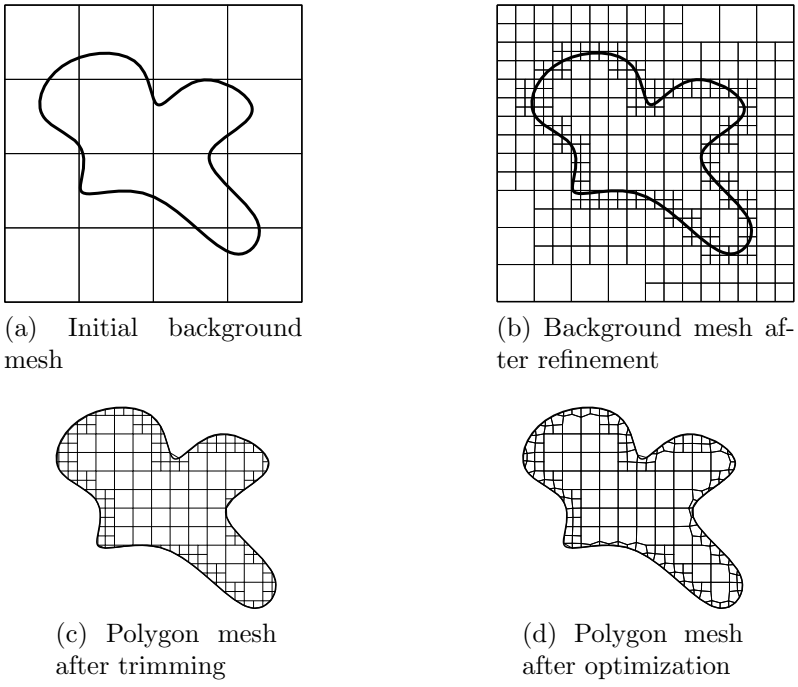
$$\omega = \sqrt{\lambda}. \tag{3.154}$$

Chapter 4

Octree based polyhedron mesh generation

Mesh generation plays an important role in the numerical analysis. In this chapter, the octree based mesh generation algorithm employed in this research is presented. Octree based method is robust for complex geometries because the initial background mesh is generated independent of the model. Fast mesh size transition can be achieved. The octree data structure is efficient in data retrieval and suitable for parallel processing. Moreover, the difficulty associated with hanging node in the conventional finite element method can be overcome by using the polyhedron shape function in the scaled boundary finite element method (SBFEM). Six numerical examples are provided to demonstrate the proposed algorithm. It is shown that the polyhedron meshes generated by octree algorithm is highly complementary to the SBFEM for incorporating geometric models with numerical analysis.

The key idea of the octree algorithm is briefly illustrated using an example in 2D (where it is called quadtree) in Fig. 4.1. A uniform background mesh is generated to cover the model (Fig. 4.1a). The background mesh is refined near the boundary of the model (Fig. 4.1b). The refined mesh is trimmed by the boundary, as shown in Fig. 4.1c.



(a) Initial background mesh

(b) Background mesh after refinement

(c) Polygon mesh after trimming

(d) Polygon mesh after optimization

Figure 4.1: Mesh generation in 2D

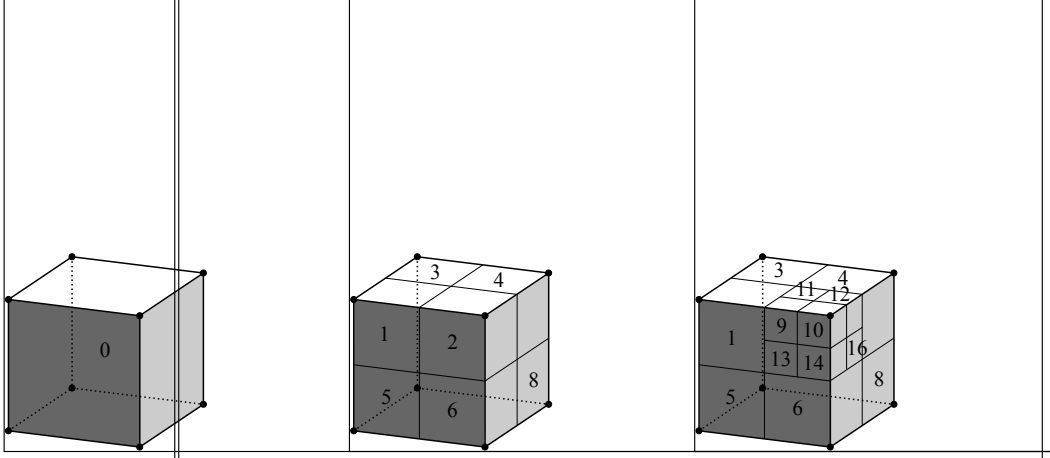
The quality of the mesh after trimming is optimized in Fig. 4.1d.

4.1 Data structure

In this section, the data structures of the mesh generation algorithm are introduced. Two different data structures are utilized in the different stages. Octree data structure is fast and robust in constructing a background mesh at the beginning. Polyhedron data structure, on the other hand, is suitable for trimming and smoothing operations. The data is converted from octree structure to polyhedron structure after the background mesh is generated and refined.

4.1.1 Octree data structure

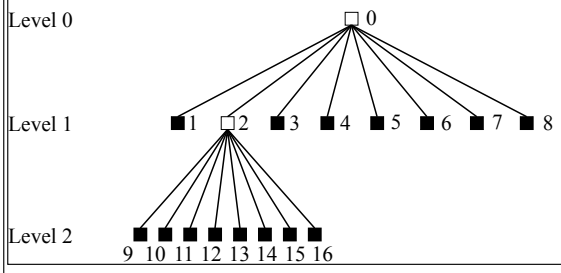
An octree is a tree data structure widely used in computer graphics, image processing, computational geometry, geographic information systems and robotics (Samet, 1984).



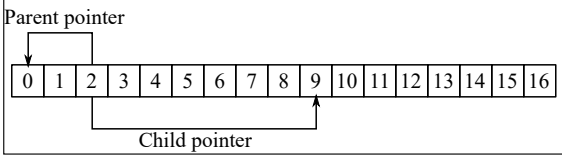
(a) The root of an octree mesh

(b) Octree mesh after first refinement

(c) Octree mesh after second refinement



(d) Hierarchical structure of the octree mesh



(e) Parent and child pointers

Figure 4.2: Octree data structure

In a numerical method, octree is employed to recursively partition of a volume in 3D into axis-aligned cubes (Eppstein, 1994). The application of octree data structure in the SBFEM was developed by Saputra et al. (2017). The terminology is presented in the following of this section using the example shown in Fig. 4.2.

Octant

Each cube in an octree structure is called an octant. The set of all octants in Fig. 4.2 is written as $O = \{o_0, o_1, o_2, \cdots, o_{16}\}$. The level of an octant is the number of times it has been subdivided from the initial cube. The size of an octant can be directly calculated from its level and the initial cube size. The location of each octant in space is also stored (usually the corner with minimum x, y and z coordinates).

Root

A root is an octant on level 0, e.g. o_0 in Fig. 4.2a. It is allowed to start with multiple roots to cover the model. Usually starting with multiple roots will occupy more storage space, but it reduces the number of subdivision required in the refinement steps. A typical case where starting with multiple roots has significant advantage is when modeling a slender or flat object. It is more efficient to generate a group of smaller roots than a large root.

Parent

When an octant is subdivided, it is called parent to the new generated octants. In the refinement step from Fig. 4.2a to Fig. 4.2b, o_0 is the parent of $o_1, o_2, \cdots, o_8$. Pointers are defined from a parent to all its children (Fig. 4.2e). The pointers are labeled according to the children's relative location to the parent. For instance, the child o_1 is located at the upper front left corner of its parent o_0, Therefore the pointer can be written as $C_{ufl}(o_0) = o_1$. Similarly, the concept of ancestor is defined. The ancestors of an octant are the octants above it on the same branch, including its parent. For example, the ancestors of o_9 include o_0 and o_2.

Child

An octant is child to another octant if it is subdivided from that octant. For example, octants $o_9, o_{10}, \cdots, o_{16}$ are the children of o_2. A pointer is also defined from each child to its parent, e.g. $P(o_1) = o_0$.

Leaf

An octant is called a leaf if it has no children. The blocks in black in Fig. 4.2d represent the leaves of the octree mesh. The leaves are the elements which will be finally used in the analysis.

Neighbor

Two octants are neighbors to each other if they are spatially connected. There are three types of neighbors in an octree structure, the neighbors sharing a face, an edge and a node, respectively. The subscripts of a neighbor pointer indicate the relative location of the neighbor, including up(u), down(d), front(f), back(b), left(l) and right(r). In Fig. 4.2c, the neighbor on top of of octant o_5 is o_1, i.e. $N_u(o_5) = o_1$. The bottom front neighbor of octant o_4 is o_6, i.e. $N_{df}(o_4) = o_6$. The top right back neighbor of octant o_{13} is o_{12}, i.e. $N_{ubr}(o_{13}) = o_{12}$. An octant may have more than one neighbors in one direction if the size of the octant is larger than its neighbors. Neighbors of an octant are not explicitly stored in the data, but it can be easily retrieved using parent and child information (Eppstein, 1994).

4.1.2 Polyhedron data structure

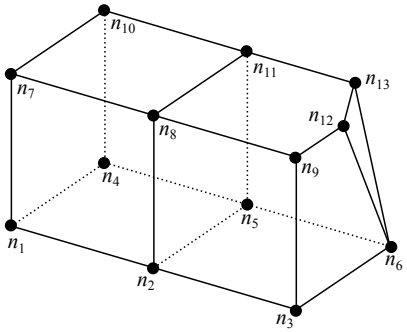
The polyhedron data structure is illustrated with a simple example in Fig. 4.3 and Table 4.1.

An arbitrary polyhedron mesh Ω is a 4-tuple (C, F, E, N) (Staten et al., 2010a), where C is a non-empty set of polyhedron cells, F is a set of polygon faces incident to one or two polyhedrons in C, E is a set of edges incident to two or more polygons faces in F, N is a set of nodes incident to two or more edges in E.

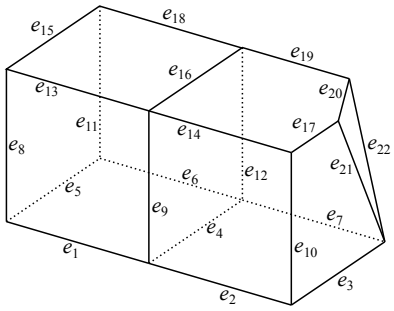
A polygon surface mesh Γ is a 3-tuple (F_S, E_S, N_S), where F_S is a non-empty set of polygon faces on the surface, E_S is a set of edges incident to one or two polygons in F_S, N_S is a set of nodes incident to two or more edges in E_S.

Node

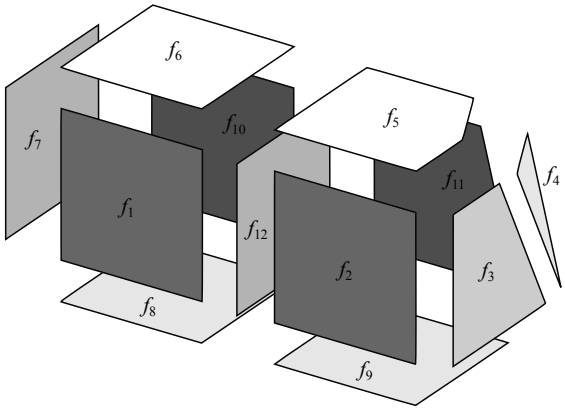
The node set N stores the coordinates of nodes (Table 4.1a). Each node is assigned with a unique nodal number (Fig. 4.3a), same as in the finite element method.



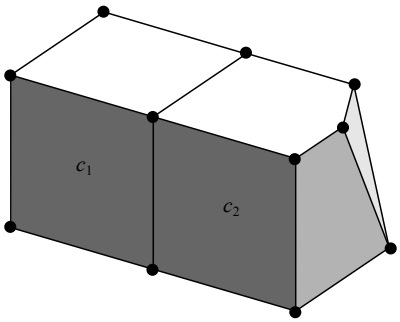
(a) Node set $N = \{n_1, n_2, n_3, \cdots\}$



(b) Edge set $E = \{e_1, e_2, e_3, \cdots\}$



(c) Face set $F = \{f_1, f_2, f_3, \cdots\}$



(d) Cell set $C = \{c_1, c_2\}$

Figure 4.3: A polyhedron mesh Ω with two cells

Table 4.1: Data structure of a polyhedron mesh Ω

(a) Node set

Node index	Coordinates
1	(0,0,0)
2	(1,0,0)
3	(2,0,0)
...	...

(b) Edge set

Edge index	Nodes
1	(n_1, n_2)
2	(n_2, n_3)
3	(n_3, n_6)
...	...

(c) Face set

Face index	Edges
1	$(e_1, e_9, -e_{13}, -e_8)$
2	$(e_2, e_{10}, -e_{14}, -e_9)$
3	$(e_3, e_{21}, -e_{17}, -e_{10})$
...	...

(d) Cell set

Cell index	Faces
1	$(f_1, f_6, f_7, f_8, f_{10}, f_{12})$
2	$(f_2, f_3, f_4, f_5, f_9, f_{11}, -f_{12})$

Edge

In the edge set E, each edge is described by the starting and ending nodes (Table 4.1b). For consistency, an edge always starts from the node with smaller ID and ends with the one with larger ID. The edges of a mesh Ω are shown in Fig. 4.3b.

Face

A face is represented by a closed loop of edges (Table 4.1c). The sign of the edge ID is positive if the edge is in the same direction with the loop, otherwise it is negative. The faces of the mesh Ω are shown in Fig. 4.3c.

Cell

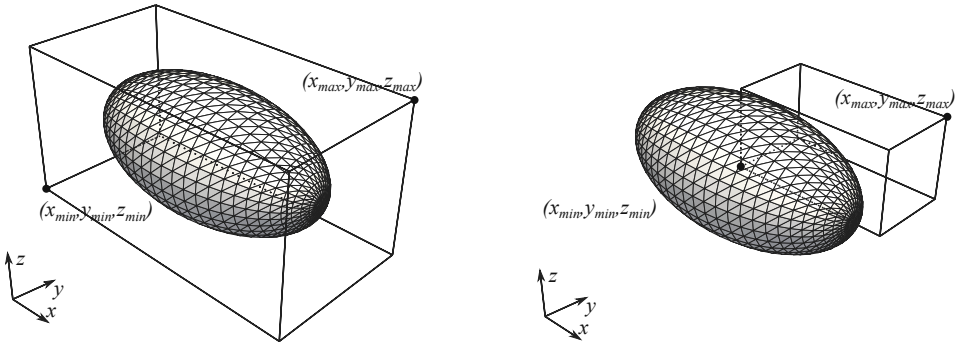
A polyhedron cell is formed by a group of faces enclosing a volume (Table 4.1d). The sign of the face ID is positive if the normal vector of the face is pointing outward of

the cell. Two polyhedron cells are shown in Fig. 4.3d. The faces of a cell have to be manifold, i.e. each edge is shared by exactly two faces, except for the cells constructed in Section 4.7.3.

4.2 Boundary representation of the geometry

The geometry of a model can be represented by the volume or the boundary. In a volume representation, the space is partitioned into a uniform grid of points. The information of whether a point is inside the model is stored. A typical example of volume representation is the digital image, which is stored as pixels in 2D and voxels in 3D. Volume representation is usually robust and Boolean operation can be easily implemented. However, it is difficult to represent models with curved surfaces accurately. The storage required by a volume representation is usually higher than a boundary representation. The boundary representation, on the other hand, defines the geometry of the model by its boundary only. It can be further classified as explicit and implicit boundary representations. In an explicit boundary representation, the analytical or discrete form of the boundary is expressed using an explicit function. An implicit representation can be defined as as a signed distance function $\varphi(x, y, z)$ which can determine the interior, exterior and boundary of a region. The points where $\varphi(x) = 0$ are known as zero level set. Boundary representation is more accurate in modeling curved and complex geometry. However it is difficult to perform Boolean operation on an explicit boundary model. The boundary model may not be manifold, which may cause difficulties in the mesh generation. The typical boundary representations including non-uniform rational B-Spline (NURBS), stereolithography (STL) and virtual reality modeling language (VRML) are already introduced in Section 1.

In this research, explicit discrete boundary representation is used as input format.



(a) A bounding box covering the whole model

(b) A user-specified bounding box covering a part of the model

Figure 4.4: Bounding box of a ellipsoid

4.3 Octree background mesh generation

In this section, the procedure to obtain an octree background mesh is presented. Different refinement criteria are discussed. A properly refined octree background mesh is crucial to the accuracy and efficiency in the analysis.

4.3.1 Bounding box of geometric model

A bounding box is generated to cover the whole or a part of the model. By default, the maximum and minimum coordinates from all the nodes in the model are calculated and a cuboid slightly larger than the model is generated so that the whole model will be covered (Fig. 4.4a). Otherwise, users can input two specific points as the corners of the bounding box (Fig. 4.4b). The dimensions of the bounding box is expressed as $L_x \times L_y \times L_z$. It is noted that the size of the bounding box may be adjusted based on the element size, which will be described in the next section.

4.3.2 Maximum and minimum element size

Two control parameters are defined by users, the maximum element size *smax* and minimum element size *smin*. In octree based algorithm, the ratio between *smax* and *smin* should be an integer power of 2. If it is not, *smax* will be adjusted to another

value $\hat{smax}$, which is calculated as

$$\hat{smax} = 2^{\text{round}\left(\log_2 \frac{smax}{smin}\right)} \times smin = s \times smin, \tag{4.1}$$

where round $(\bullet)$ represents rounding towards the nearest integer and s is defined as the maximum mesh size ratio. If the ratio between the dimensions of the bounding box and $\hat{smax}$ is not an integer, the bounding box is also adjusted. The adjusted length, width and height are expressed as

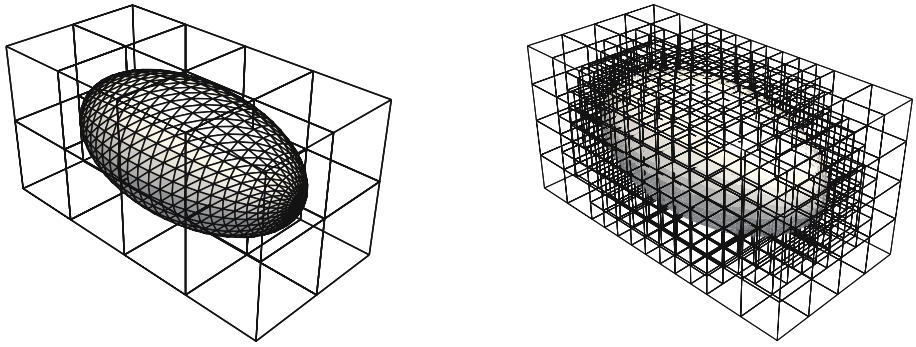
$$\hat{L_x} = \text{ceil}\left(\frac{L_x}{\hat{smax}}\right) \times \hat{smax}, \tag{4.2a}$$

$$\hat{L_y} = \text{ceil}\left(\frac{L_y}{\hat{smax}}\right) \times \hat{smax}, \tag{4.2b}$$

$$\hat{L_z} = \text{ceil}\left(\frac{L_z}{\hat{smax}}\right) \times \hat{smax}, \tag{4.2c}$$

where ceil $(\bullet)$ means rounding to the nearest integer towards infinity. It is noted that $\hat{smax}$ merely represents the maximum allowed element size in the octree mesh. The actual maximum element size depends on different factors such as the refinement criteria, mesh size transition rate and geometry of the model.

An example is shown in Fig. 4.5 to illustrate the generation of background mesh considering element size. The dimensions of the ellipsoid is 2mm $\times$ 0.99mm $\times$ 0.99mm. A default bounding box is generated by scaling the dimensions of the model by 1.1. The user input element sizes are $smax = 0.5$mm and $smin = 0.15$mm. The adjusted maximum element size is calculated using Eq. (4.1), $\hat{smax} = 0.6$mm. The dimensions of the bounding box are adjusted according to Eq. (4.2). The adjusted bounding box is 2.4mm $\times$ 1.2mm $\times$ 1.2mm. The bounding box is initially divided into 4 octants in x direction and 2 octants in y and z directions as shown in Fig. 4.5a. These octants are refined until minimum element size 0.15mm is reached (Fig. 4.5b).



(a) Initial octree background mesh

(b) Final octree background mesh after refinement

Figure 4.5: Octree background mesh

4.3.3 Refinement criteria

Usually an octree background mesh is not refined uniformly. The mesh is only refined in selected regions so that the computational resources can be used most efficiently. There are several different refinement criteria which can be selected and combined during the refinement.

4.3.3.1 Refinement based on curvature

As the shape of the surface elements on the boundary is interpolated by piece-wise polynomials, it is desirable that the mesh size is smaller at the location where the curvature is high, therefore the geometry can be better represented. A variety of method have been proposed to estimate the curvature in a discrete boundary model (Surazhsky et al., 2003; Razdan and Bae, 2005; Gatzke and Grimm, 2006). A simple approach reported in Liu et al. (2017) is employed. In an octant, the intersection points $I = \{I_1, I_2, ..., I_n\}$ between the edges and the boundary model are calculated. The point at the average coordinates of I is denoted as O. The shortest distance from O to all the triangles near the octant is calculated (the length of segment OO' in Fig. 4.6). The

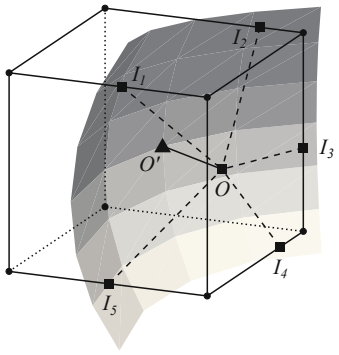


Figure 4.6: Estimation of curvature

curvature is estimated as

$$\kappa = \frac{n \left\| O - O' \right\|}{\sum_{i=1}^{n} \left\| O - I_i \right\|}. \tag{4.3}$$

If the curvature is larger than a predefined threshold κ_{tol}, the octant will be refined. The typical value of κ_{tol} is between 0.05 and 0.1.

4.3.3.2 Refinement based on surface patch

The octree mesh can be refined near a specified patch of surface. The patch can be represented in the same way as the input model. This criterion is especially useful in contact problems, i.e. the region near the contact interface will be refined to better represent the contact pressure.

4.3.3.3 Refinement based on ridge

Sharp ridges, especially concave ridges, may cause stress singularities in an analysis. The mesh size can be refined around the sharp ridges. The detailed definition and processing of the ridges will be addressed in Section 4.5.2.

4.3.3.4 Refinement based on location

Sometimes the octree mesh in a special location needs to be refined. A refinement function can be defined by the coordinates as $f(x, y, z)$. If $f = 1$ at some nodes of an

octant, it will be refined. Otherwise, if $f = 0$ for all the nodes of an octant, it will not be refined. A typical application of this refinement criterion is the elasto-plastic analysis, where the mesh in the potential plastic zone can be refined beforehand.

4.3.3.5 Combination of refinement criteria

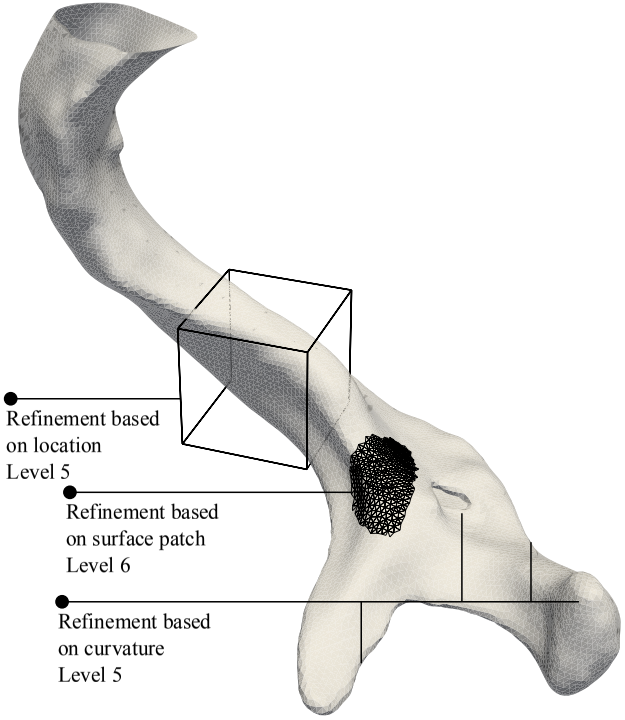
These criteria in Sections 4.3.3.1–4.3.3.4 can be combined based on the user requirement. Each of the criteria is assigned with an individual maximum refinement level. An STL model is shown in Fig. 4.7a. 3 refinement criteria are defined based on location, surface patch and curvature. The refinement levels are 5, 6 and 5, respectively. After the refinement and trimming (trimming will be presented in Section 4.4), the mesh size near the surface patch (refinement level 6) is refined to the minimum size, while the mesh size of the region in the box (refinement level 5) is twice the minimum size. In the whole model where the curvature is higher than 0.05 the mesh is refined to twice the minimum size (refinement level 5). The final mesh is shown in Fig. 4.7b.

4.3.4 Balance of octree mesh

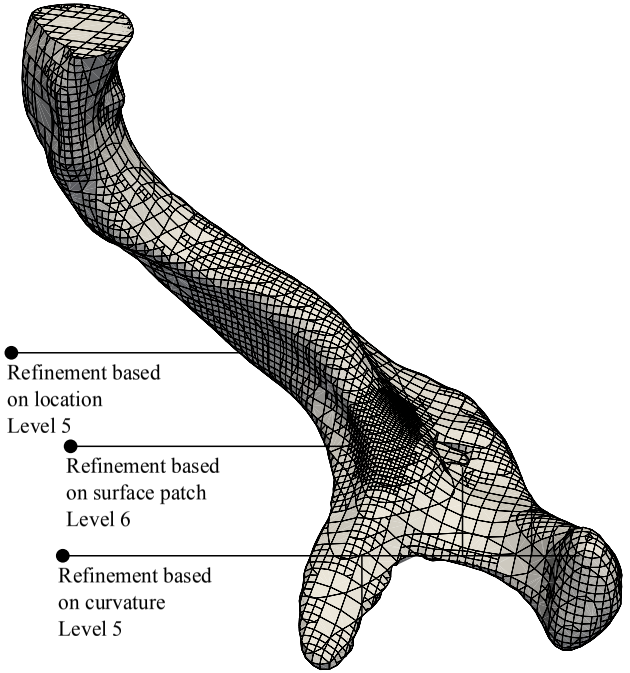
A so called "1:2 rule" is enforced on the octree mesh, which means the maximum size ratio between adjacent elements can not exceed 2. This requirement provides a smooth mesh size transition and reduces the number of possible configurations of octants. In Fig. 4.8a, the octree mesh contains octants of three different levels. There are octants from level 0 directly adjacent to those from level 2, which means the maximum size ratio between adjacent elements is 4. Those octants on level 0 are subdivided into smaller octants on level 1 to obtain the balanced octree mesh in Fig. 4.8b.

4.3.5 Control of mesh size transition

The mesh size transition based on the "1:2 rule" only may lead to more rapid change of mesh size than desired. The mesh size transition can be controlled by a mesh size

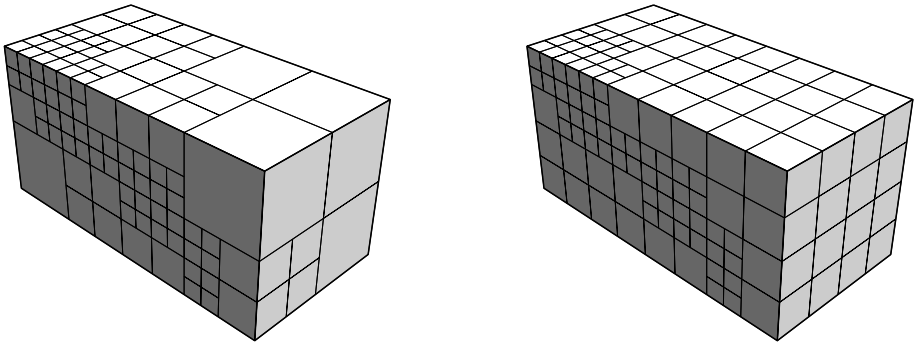


(a) An STL model with 3 refinement criteria



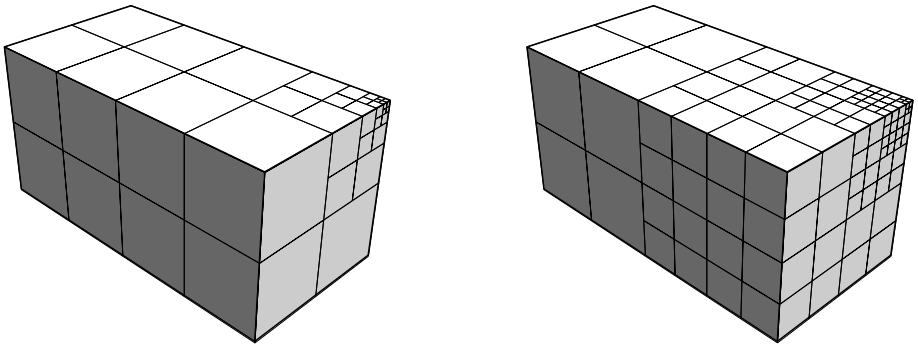
(b) Polyhedron mesh after refinement and trimming

Figure 4.7: Combination of refinement criteria



(a) Unbalanced octree mesh (b) Balanced octree mesh

Figure 4.8: Balance of the octree mesh



(a) Default mesh size transition based on "1:2 rule"

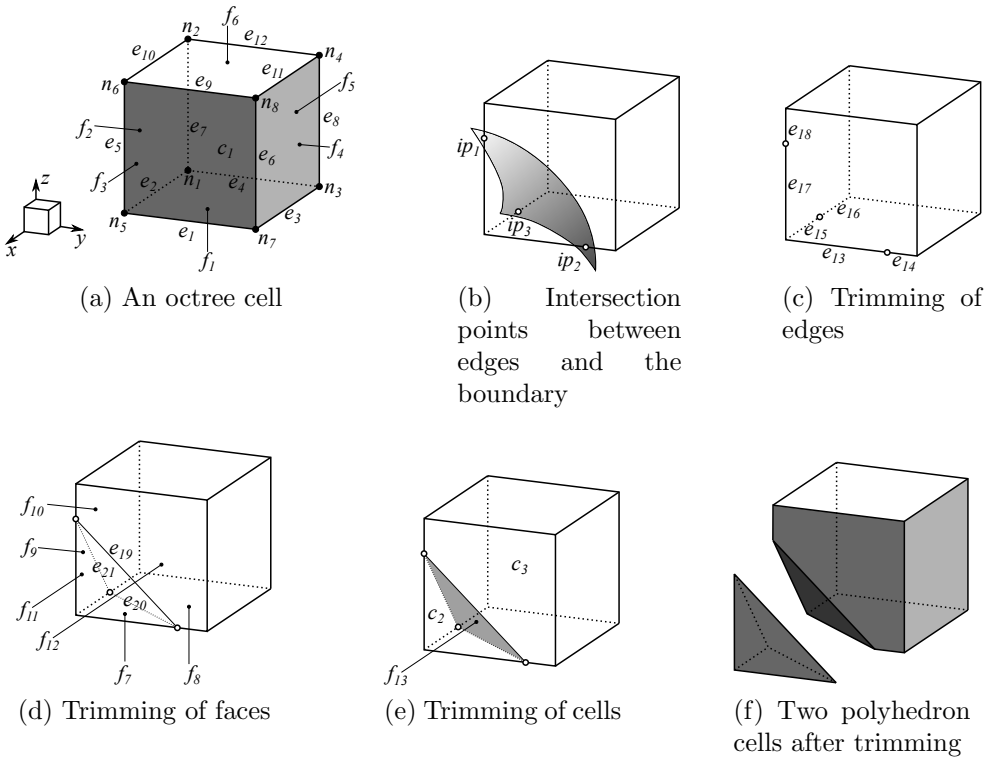
(b) Slower mesh size transition with $m = 2$

Figure 4.9: Control of mesh size transition

transition rate m. As an example, a balanced octree mesh is shown in Fig. 4.9a. If an octant satisfies the refinement criteria in Section 4.3.3 or it needs to be balanced according to Section 4.3.4, its neighbor octants within m layers will be refined as well. As a result, there will be at least m layers of octants of level n before the mesh size can increase to level $n - 1$. The transition rate is $m = 2$ in Fig. 4.9b, so on each level at least two layers of octants are generated.

4.4 Trimming of octree mesh

The octree mesh can be used in an analysis directly. However, when the boundary of the model is curved it takes a large number of cube elements to approximate the



(a) An octree cell

(b) Intersection points between edges and the boundary

(c) Trimming of edges

(d) Trimming of faces

(e) Trimming of cells

(f) Two polyhedron cells after trimming

Figure 4.10: Trimming of an octree cell

geometry to obtain an accurate solution. Although this approach is simple and robust, its computational efficiency may not be optimal. To overcome this shortcoming, the data structure of the octree mesh is converted to that of a polyhedron mesh as specified in Section 4.1.1 and Section 4.1.2. The cube elements are trimmed by the boundary of the model so that a conforming polyhedron mesh is generated. The related work can be found in Liu et al. (2017).

In this section the trimming procedure is illustrated using a simple cube shown in Fig. 4.10a. The data of the cube is shown in Table 4.2. The trimming process follows a bottom-up scheme in the sequence of edges (on the bottom level of the polyhedron data structure) as illustrated in Fig. 4.10c, faces (the middle level) in Fig. 4.10d and polyhedron cells (the top level) in Fig. 4.10e.

Table 4.2: Original data of a cube

(a) Node set

Index	Coordinates	Index	Coordinates
1	(0,0,0)	5	(1,0,0)
2	(0,0,1)	6	(1,0,1)
3	(0,1,0)	7	(1,1,0)
4	(0,1,1)	8	(1,1,1)

(b) Edge set

Index	Nodes	Index	Nodes	Index	Nodes
1	(n_5, n_7)	5	(n_5, n_6)	9	(n_6, n_8)
2	(n_1, n_5)	6	(n_7, n_8)	10	(n_2, n_6)
3	(n_3, n_7)	7	(n_1, n_2)	11	(n_4, n_8)
4	(n_1, n_3)	8	(n_3, n_4)	12	(n_2, n_4)

(c) Face set

Index	Edges	Index	Edges
1	$(-e_1, -e_2, e_4, e_3)$	4	$(-e_3, e_8, e_{11}, -e_6)$
2	$(e_5, -e_{10}, -e_7, e_2)$	5	$(-e_4, e_7, e_{12}, -e_8)$
3	$(e_1, e_6, -e_9, -e_5)$	6	$(e_9, -e_{11}, -e_{12}, e_{10})$

(d) Cell set

Index	Faces
1	$(f_1, f_2, f_3, f_4, f_5, f_6)$

4.4.1 Intersection mark and node sign

The first step of trimming is to obtain the intersection mark and determine the node sign. The edges intersected by the boundary are marked and the intersection points in those edges are stored. If an intersection point is located at one end of the edge, the node is marked as intersected. The node sign is obtained using a breadth first search (Moore, 1959; Luo et al., 2010). The sign of a node is defined as positive if the node is located outside the model, negative if it is inside the model and zero if it is on the boundary.

The breadth first search to assign node signs is shown in Algorithm 4.1. An initial node sign s_1 and a starting point n_1 are specified. By default, the starting point is located at the bottom back left corner of the initial octree grid. The default initial node sign is positive (outside). A queue q_1 is generated with the starting point added at the beginning. All the nodes connected to the starting point by edges which are not intersected by the boundary are searched and appended to the queue. The starting point is marked as "visited". Then this procedure will continue for the next node in the queue. If a node is already visited, it will be skipped and the next node will be checked. This process is repeated until all the members in the queue are visited. Those visited nodes will be assigned the initial node sign (positive by default). Then an edge with one and only one intersection is located, the end node which has not been assigned the initial sign is found. A new queue q_2 is generated starting from this new node. The searching process is conducted once more and all the nodes visited this time will be assigned the inverse of initial sign s_2 (negative by default). The remainder of the nodes, which are located on the boundary, will be assigned zero sign. The intersection points on the edges are appended to the node list (row 9 to 11 in Table 4.3a), and their signs are zero as well (points ip_1, ip_2, ip_3 in Fig. 4.10b).

Algorithm 4.1 Breadth first search

```
1: Input mesh
2: Define starting node n_1 and initial sign s_1
3: Generate empty queue q_1
4: Append n_1 to the end of q_1
5: i = 1
6: while not all nodes in q_1 are "visited" do
7:     if n_i is "visited" then
8:         Continue
9:     else
10:        Find all nodes connected to n_i by unintersected edges
11:        Append the nodes to the end of q_1
12:        Mark n_i as "visited"
13:        i = i + 1
14:    end if
15: end while
16: Assign s_1 to all the nodes in q_1
17: Locate starting node n_1 with inversed initial sign s_2
18: Generate empty queue q_2
19: Append n_1 to the end of q_2
20: i = 1
21: while not all nodes in q_2 are "visited" do
22:    if n_i is "visited" then
23:        Continue
24:    else
25:        Find all nodes connected to n_i by unintersected edges
26:        Append the nodes to the end of q_2
27:        Mark n_i as "visited"
28:        i = i + 1
29:    end if
30: end while
31: Assign s_2 to all the nodes in q_2
32: Assign zero sign to all the rest of the nodes
```

Table 4.3: Data of the trimmed cube

(a) Node set

Index	Coordinates	Index	Coordinates	Index	Coordinates
1	(0,0,0)	5	(1,0,0)	ip_1	(1,0,0.7)
2	(0,0,1)	6	(1,0,1)	ip_2	(1,0.8,0)
3	(0,1,0)	7	(1,1,0)	ip_3	(0.5,0,0)
4	(0,1,1)	8	(1,1,1)		

(b) Edge set

Index	Nodes	Index	Nodes	Index	Nodes	Index	Nodes
1(trimmed)	(n_5, n_7)	7	(n_1, n_2)	13	(n_5, ip_2)	19	(ip_1, ip_2)
2(trimmed)	(n_1, n_5)	8	(n_3, n_4)	14	(ip_2, n_7)	20	(ip_2, ip_3)
3	(n_3, n_7)	9	(n_6, n_8)	15	(ip_3, n_5)	21	(ip_1, ip_3)
4	(n_1, n_3)	10	(n_2, n_6)	16	(n_1, ip_3)		
5(trimmed)	(n_5, n_6)	11	(n_4, n_8)	17	(n_5, ip_1)		
6	(n_7, n_8)	12	(n_2, n_4)	18	(ip_1, n_6)		

(c) Face set

Index	Edges	Index	Edges
1(trimmed)	$(-e_1, -e_2, e_4, e_3)$	7	$(-e_{13}, -e_{15}, -e_{20})$
2(trimmed)	$(e_5, -e_{10}, -e_7, e_2)$	8	$(-e_{16}, e_4, e_3, -e_{14}, e_{20})$
3(trimmed)	$(e_1, e_6, -e_9, -e_5)$	9	(e_{15}, e_{17}, e_{21})
4	$(-e_3, e_8, e_{11}, -e_6)$	10	$(e_{18}, -e_{10}, -e_7, e_{16}, -e_{21})$
5	$(-e_4, e_7, e_{12}, -e_8)$	11	$(-e_{17}, e_{13}, -e_{19})$
6	$(e_9, -e_{11}, -e_{12}, e_{10})$	12	$(e_{14}, e_6, -e_9, -e_{18}, e_{19})$
		13	$(e_{19}, e_{20}, -e_{21})$

(d) Cell set

Index	Faces
1(trimmed)	$(f_1, f_2, f_3, f_4, f_5, f_6)$
2	$(f_7, f_9, f_{11}, f_{13})$
3	$(f_4, f_5, f_6, f_8, f_{10}, f_{12}, -f_{13})$

4.4.2 Trimming of edges

The trimming of edges is a straight forward procedure. The edges are partitioned into new edges at the intersection points. For example, edge e_1 is divided into two new edges $e_{13} = (n_5, ip_2)$ and $e_{14} = (ip_2, n_7)$ in Fig. 4.10c. The new edges are appended to the edge list (row 13 to 18 in Table 4.3b). The original edges are marked as "trimmed" and will be deleted later. A map from the original edges to the new edges is generated. The faces will be updated according to the map, for instance, face $f_1 = (-e_1, -e_2, e_4, e_3)$ will be updated as $f_1 = (-e_{14}, -e_{13}, -e_{15}, -e_{16}, e_4, e_3)$. It is allowed to have more than one intersection points in one edge. Those edges will be divided into more than two new edges.

4.4.3 Trimming of faces

The new edges connecting the intersection points will be appended to the edge list as well (row 19 to 21 in Table 4.3b). The faces containing the trimmed edges will be trimmed at the intersection points. The edge set is divided into several smaller segments at the intersection points, e.g. the edge set of face f_1 is divided in to two connected segments: $(-e_{13}, -e_{15})$ and $(-e_{16}, e_4, e_3, -e_{14})$. Each of the segments can form a closed loop together with the appended edge. For example, the face f_1 becomes $f_7 = (-e_{13}, -e_{15}, -e_{20})$ and $f_8 = (-e_{16}, e_4, e_3, -e_{14}, e_{20})$ as illustrated in Fig. 4.10d. The new faces are appended to the end of the face list (row 7 to 12 in Table 4.3c) and original faces are marked as "trimmed". The cells are updated according to the mapping from the original faces to the new faces. Cell $c_1 = (f_1, f_2, f_3, f_4, f_5, f_6)$ will be updated as $c_1 = (f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12})$. If there are more than two intersection points in one face, it will be divided into more than two new faces. The first step is to divide the edge set into segments at intersection points. The segments which can be closed by one appended edge will be constructed as new faces first. Then the remaining segments will be closed by all the appended edges.

4.4.4 Trimming of cells

The cells will be trimmed using a similar approach as in the trimming of faces. A new face is generated by connecting the new appended edges, e.g. $f_{13} = (e_{19}, e_{20}, -e_{21})$. The face set of the cell is divided into two groups based on the node sign. In Fig. 4.10e, faces f_7, f_9 and f_{11} contain negative and zero node sign only, therefore they are stored in a negative group. The rest of the faces are stored in a positive group. The two groups are closed by the appended face f_{13}. Therefore, cell c_1 becomes c_2, c_3 (Fig. 4.10f).

If there are more than one new faces generated from the cell, it will be divided into more new cells. More than two groups will be generated. The principle is that the faces in a group are connected by shared edges. If the non-manifold edges in a group are identical to the edges of one of the appended faces, the face is appended to the group so that the group of faces can be closed. The remaining of the groups, will be closed by appending all the new faces.

4.5 Handling sharp ridges and corners

The trimming procedure in Section 4.4 is designed for the models with smooth boundaries. When there are sharp ridges and corners in the model , they cannot be preserved in the obtained mesh. Therefore, special treatments are developed in this section to recover sharp ridges and corners from the model.

4.5.1 Ridge and corner mark

In computational geometry, a ridge is an edge connecting two boundary faces which form a sharp dihedral angle (Seidel, 1986). A corner is a node connected to two or more ridges.

The information of ridge and corner is not explicitly stored in an STL model, therefore it needs to be extracted from the model first. As the STL model contains duplicated

nodes, firstly a unique node list needs to be constructed. Then the unique edges are identified. The normal vectors of the two faces sharing an edge are calculated. If the angle between the two vectors is larger than a threshold, e.g. 45°, the edge is identified as a sharp ridge. The nodes connected to more than two sharp ridges are identified as corners.

The faces in the octree mesh intersected by the ridges of the STL model are marked. The locations of the intersection points are also stored. The octree cells containing the corners of the STL model are marked.

4.5.2 Recovery of ridges

The trimming procedure of a cell containing ridges is illustrated using an example in Fig. 4.11. The original cell is the same as the one in Fig. 4.10a. The boundary of the model near the cell is shown in the shaded area in Fig. 4.11a. It can be observed that there is a sharp ridge passing through the cell, leaving two intersection points rd_1 and rd_2. They are located in face f_2 and f_3, respectively.

The trimming of edges is the same as the process in Fig. 4.10c. 6 new edges (row 13 to 18 in Table 4.4b) are generated and appended to the edge list.

The trimming of faces intersected with ridges is different. In Fig. 4.11c, face f_3 is intersected with a ridge. Therefore, the intersection point rd_2 is appended to the node list. The intersection points in the face, ip_1 and ip_2, are connected to rd_2, respectively. Two new edges will be generated. The face is then partitioned into f_{11} and f_{12} as shown in Fig. 4.11d. When trimming cell c_1, a new edge e_{24} connecting the ridge points rd_1 and rd_2 is constructed first. This edge, together with the new appended edges (row 19 to 23 in Table 4.4b), will form two new faces f_{13} and f_{14} (see Fig. 4.11d). The new faces will be appended to the other faces to form two new cells c_2 and c_3.

Table 4.4: Data of the trimmed polyhedron mesh with a ridge

(a) Node set

Index	Coordinates	New nodes	Coordinates
1	(0,0,0)	ip_1	(1,0,0.6)
2	(0,0,1)	ip_2	(1,0.5,0)
3	(0,1,0)	ip_3	(0.5,0,0)
4	(0,1,1)	rd_1	(0.7,0,0.6)
5	(1,0,0)	rd_2	(1,0.6,0.4)
6	(1,0,1)		
7	(1,1,0)		
8	(1,1,1)		

(b) Edge set

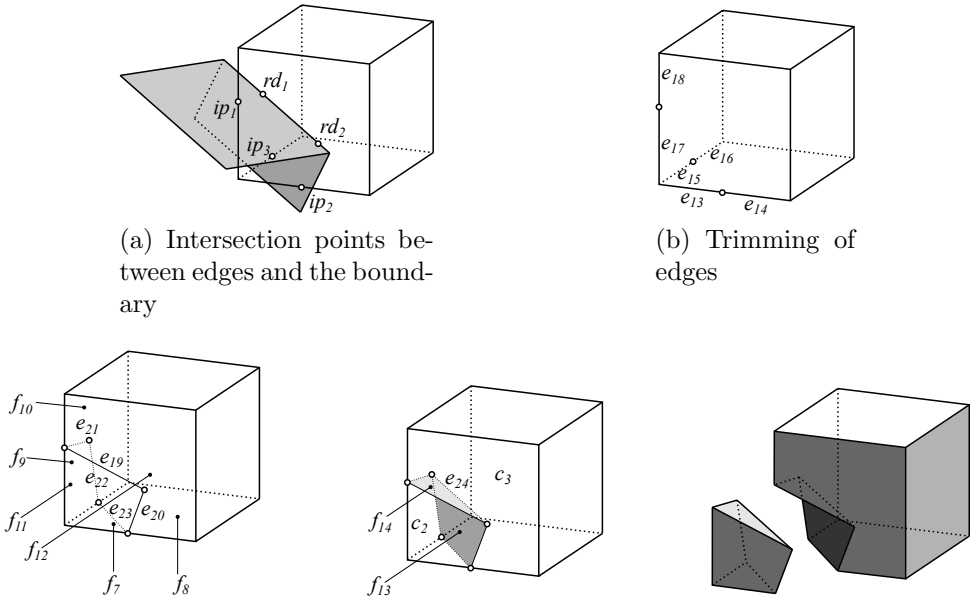
Index	Nodes	Index	Nodes	Index	Nodes	Index	Nodes
1(trimmed)	(n_5, n_7)	7	(n_1, n_2)	13	(n_5, ip_2)	19	(ip_1, rd_2)
2(trimmed)	(n_1, n_5)	8	(n_3, n_4)	14	(ip_2, n_7)	20	(ip_2, rd_2)
3	(n_3, n_7)	9	(n_6, n_8)	15	(ip_3, n_5)	21	(ip_1, rd_1)
4	(n_1, n_3)	10	(n_2, n_6)	16	(n_1, ip_3)	22	(ip_3, rd_1)
5(trimmed)	(n_5, n_6)	11	(n_4, n_8)	17	(n_5, ip_1)	23	(ip_2, ip_3)
6	(n_7, n_8)	12	(n_2, n_4)	18	(ip_1, n_6)	24	(rd_1, rd_2)

(c) Face set

Index	Edges	Index	Edges
1(trimmed)	$(-e_1, -e_2, e_4, e_3)$	7	$(-e_{13}, -e_{15}, -e_{23})$
2(trimmed)	$(e_5, -e_{10}, -e_7, e_2)$	8	$(-e_{16}, e_4, e_3, -e_{14}, e_{23})$
3(trimmed)	$(e_1, e_6, -e_9, -e_5)$	9	$(e_{15}, e_{17}, e_{21}, -e_{22})$
4	$(-e_3, e_8, e_{11}, -e_6)$	10	$(e_{18}, -e_{10}, -e_7, e_{16}, e_{22} - e_{21})$
5	$(-e_4, e_7, e_{12}, -e_8)$	11	$(-e_{17}, e_{13}, e_{20}, -e_{19})$
6	$(e_9, -e_{11}, -e_{12}, e_{10})$	12	$(e_{14}, e_6, -e_9, -e_{18}, e_{19}, -e_{20})$
		13	$(e_{19}, -e_{24}, -e_{21})$
		14	$(-e_{20}, e_{23}, e_{22}, e_{24})$

(d) Cell set

Index	Faces
1(trimmed)	$(f_1, f_2, f_3, f_4, f_5, f_6)$
2	$(f_7, f_9, f_{11}, f_{13}, f_{14})$
3	$(f_4, f_5, f_6, f_8, f_{10}, f_{12}, -f_{13}, -f_{14})$



(a) Intersection points between edges and the boundary

(b) Trimming of edges

(c) Trimming of faces

(d) Trimming of cells

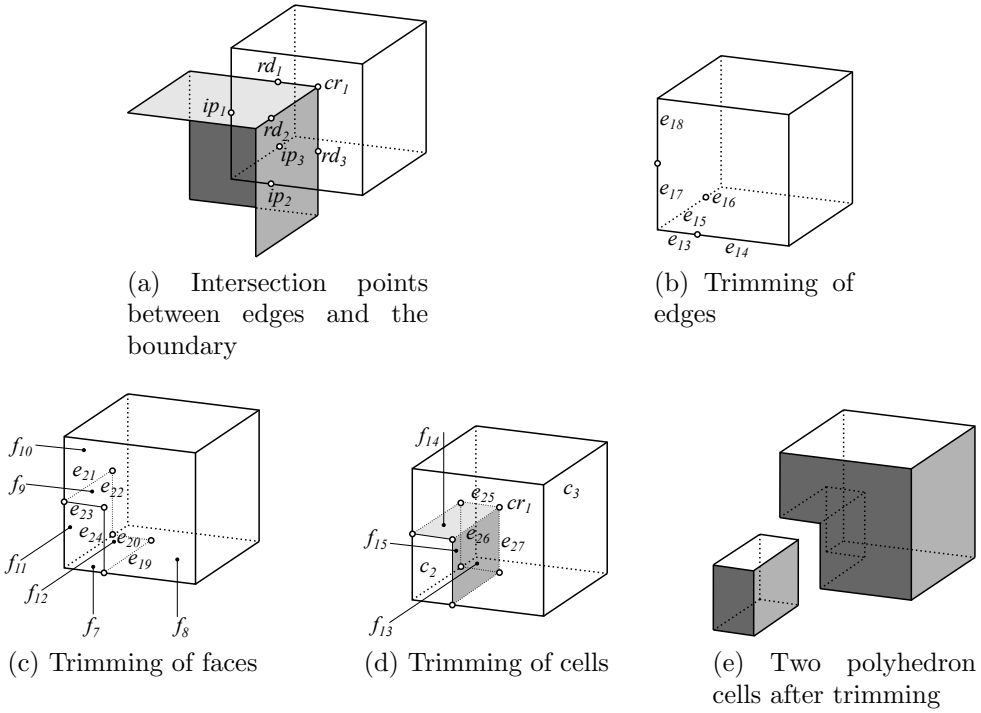
(e) Two polyhedron cells after trimming

Figure 4.11: Trimming of an octree cell with a ridge

4.5.3 Recovery of corners

The trimming of a cell with a sharp corner inside is shown in Fig. 4.12. There are 3 sharp ridges passing through the cell, leaving 3 intersection points rd_1, rd_2 and rd_3. The joint of the 3 ridges cr_1, which is a sharp corner, is located inside the cell.

The trimming procedure of edges and faces is the same as the procedure in Section 4.5.2. The difference is in the trimming of the cell (Fig. 4.12d). The sharp corner point cr_1 is appended to the node list (row 15 in Table 4.5a). 3 new edges, connecting the ridge points to the corner point, are appended to the edge list (row 25 to 27 in Table 4.5b). The appended edges on the surface of the cell, are divided in to 3 groups at the ridge points. For example, one such group consists of edge 21 and 23. These edges are combined with the appended edges inside the cell to form a closed loop (in this case, edge 25 and 26). The closed loop is constructed as a new face inside the cell (row 13 in Table 4.5c). The new faces will be appended to the trimmed faces to form 2 new cells (Fig. 4.5d).



(a) Intersection points between edges and the boundary

(b) Trimming of edges

(c) Trimming of faces

(d) Trimming of cells

(e) Two polyhedron cells after trimming

Figure 4.12: Trimming of an octree cell with a corner

Currently it is not allowed to have more than one corner point inside a cell.

4.6 Face discretization of polyhedron cells

4.6.1 Triangle quality

In the presented approach, the polygon faces of a polyhedron cell are discretized with triangular and quadrilateral elements if the polygons have more than 4 edges. Degenerated quadrilateral faces will also be triangulated. The algorithm is to enumerate all the possible triangulation schemes and select the scheme which can produce triangles with best qualities. Different measures of triangle quality can be found in (Pebay and Baker, 2003). A simple approach is followed here. Denoting the lengths of the three

Table 4.5: Data of the trimmed polyhedron mesh with a corner

(a) Node set

Index	Coordinates	Index	Coordinates
1	(0,0,0)	ip_1	(1,0,0.5)
2	(0,0,1)	ip_2	(1,0.3,0)
3	(0,1,0)	ip_3	(0.3,0,0)
4	(0,1,1)	rd_1	(0.3,0,0.5)
5	(1,0,0)	rd_2	(1,0.3,0.5)
6	(1,0,1)	rd_3	(0.3,0.3,0)
7	(1,1,0)	cr_1	(0.3,0.3,0.5)
8	(1,1,1)		

(b) Edge set

Index	Nodes	Index	Nodes	Index	Nodes	Index	Nodes
1(trimmed)	(n_5, n_7)	7	(n_1, n_2)	13	(n_5, ip_2)	19	(ip_2, rd_3)
2(trimmed)	(n_1, n_5)	8	(n_3, n_4)	14	(n_7, ip_2)	20	(ip_3, rd_3)
3	(n_3, n_7)	9	(n_6, n_8)	15	(n_5, ip_3)	21	(ip_1, rd_1)
4	(n_1, n_3)	10	(n_2, n_6)	16	(n_1, ip_3)	22	(ip_3, rd_1)
5(trimmed)	(n_5, n_6)	11	(n_4, n_8)	17	(n_5, ip_1)	23	(ip_1, rd_2)
6	(n_7, n_8)	12	(n_2, n_4)	18	(n_6, ip_1)	24	(ip_2, rd_2)
						25	(rd_1, cr_1)
						26	(rd_2, cr_1)
						27	(rd_3, cr_1)

(c) Face set

Index	Edges	Index	Edges
1(trimmed)	$(-e_1, -e_2, e_4, e_3)$	7	$(-e_{13}, e_{15}, e_{20}, -e_{19})$
2(trimmed)	$(e_5, -e_{10}, -e_7, e_2)$	8	$(-e_{16}, e_4, e_3, e_{14}, e_{19}, -e_{20})$
3(trimmed)	$(e_1, e_6, -e_9, -e_5)$	9	$(-e_{15}, e_{17}, e_{21}, -e_{22})$
4	$(-e_3, e_8, e_{11}, -e_6)$	10	$(-e_{18}, -e_{10}, -e_7, e_{16}, e_{22} - e_{21})$
5	$(-e_4, e_7, e_{12}, -e_8)$	11	$(-e_{17}, e_{13}, e_{24}, -e_{23})$
6	$(e_9, -e_{11}, -e_{12}, e_{10})$	12	$(-e_{14}, e_6, -e_9, e_{18}, e_{23}, -e_{24})$
		13	$(-e_{21}, e_{23}, e_{26}, -e_{25})$
		14	$(e_{19}, e_{27}, -e_{26}, -e_{24})$
		15	$(-e_{20}, e_{22}, e_{25}, -e_{27})$

(d) Cell set

Index	Faces
1(trimmed)	$(f_1, f_2, f_3, f_4, f_5, f_6)$
2	$(f_7, f_9, f_{11}, f_{13}, f_{14}, f_{15})$
3	$(f_4, f_5, f_6, f_8, f_{10}, f_{12}, -f_{13}, -f_{14}, -f_{15})$

edges of a triangle as l_1, l_2 and l_3, the quality q of a triangle is expressed as:

$$q = \frac{r}{\max(l_1, l_2, l_3)}, \tag{4.4}$$

with

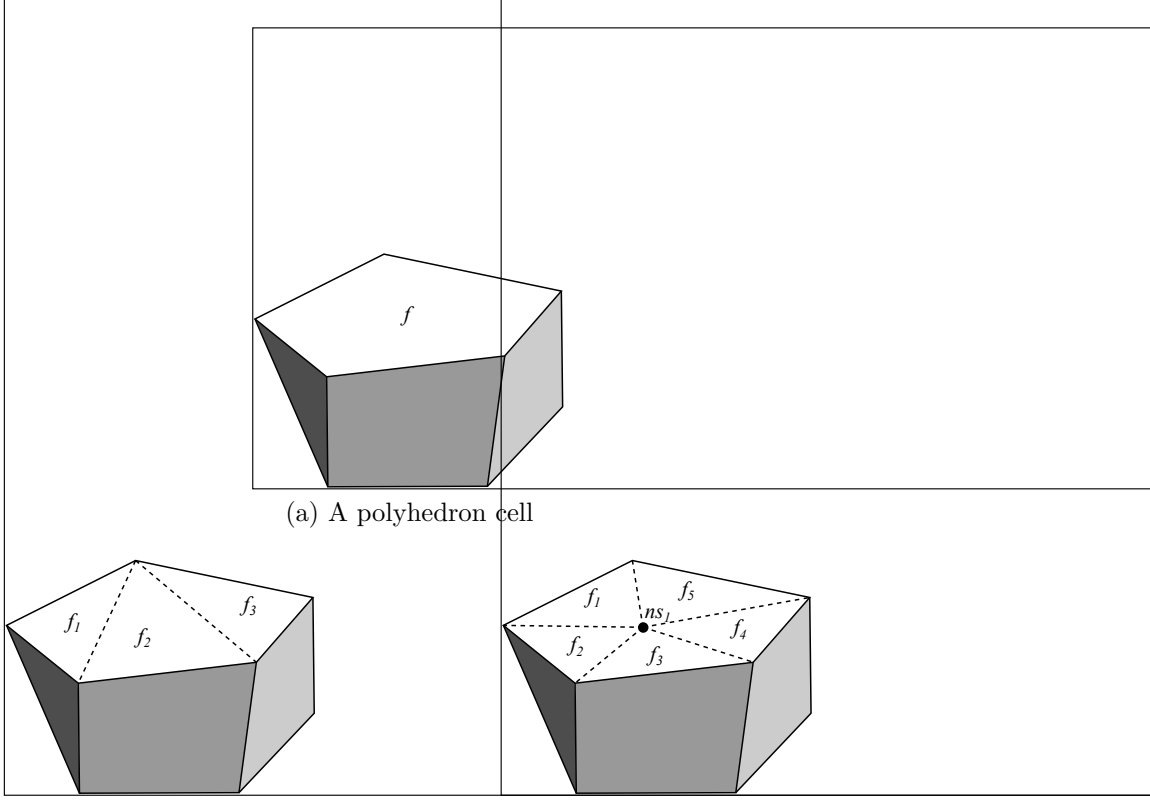
$$r = \sqrt{\frac{(s-l_1)(s-l_2)(s-l_3)}{s}}, \tag{4.5}$$

and

$$s = \frac{l_1 + l_2 + l_3}{2}. \tag{4.6}$$

4.6.2 Triangulation schemes

Two triangulation schemes are tested in the enumeration. The first scheme uses only the existing nodes of the polygon. The other scheme introduces a Steiner point (Eppstein, 1994; Berg et al., 2008), which is always inside the polygon, to triangulate the polygon. For each polygon, the worst quality of all generated triangles is recorded as the quality of the discretization of the polygon. The discretization of the highest quality is selected. Examples of the two scheme are shown in Fig. 5.8. The pentagon face f in Fig. 5.8a is divided into 3 triangles by connecting the existing nodes. A sample of such connection (triangles f_1, f_2 and f_3) is shown in Fig. 5.8b. The second scheme is inserting a Steiner point ns_1 in Fig. 5.8c. The point ns_1 is connected to all the edges of the polygon to form new triangles (from f_1 to f_5). In this case, the triangles have better quality than those obtained by direct triangulation without inserting the Steiner point. After the discretization of all the faces of the polyhedron cells, the polyhedron mesh can be used in the scaled boundary finite element analysis as formulated in Section 3.



(a) A polyhedron cell

(b) Triangulation without inserting node

(c) Triangulation with inserted Steiner point

Figure 4.13: Face discretization of a polyhedron cell

4.7 Optimization of mesh quality

A well-known disadvantage of octree based mesh is that the element quality near the boundary is usually poor, especially when the boundary of the model forms a small angle with the octree grid. Distorted surface elements and short edges will appear on the boundary which are not suitable for analysis. There are a variety of techniques developed to optimize the mesh quality. They can be classified into mainly two different groups. The first group of methods relocate the nodes without changing the connectivity, such as Laplacian smoothing, optimization based method, physically based method just to name a few (Owen, 1998). The other group of methods modify the node connectivity, such as edge flipping, edge bisection, node insertion and many others. In this research, Laplacian smoothing is utilized to improve mesh quality.

4.7.1 Laplacian smoothing

Laplacian smoothing is the simplest and most straight forward optimization technique. A node is shifted towards the average coordinate of all the nodes connected to it. The

average coordinate is expressed as

$$\bar{\boldsymbol{p}} = \frac{\sum_{i=1}^{n} \boldsymbol{p}_i}{n}, \tag{4.7}$$

where $\boldsymbol{p}_i$ is the coordinates of i-th neighbor node and n is the number of neighbor nodes to be considered. The new coordinate of the point is

$$\boldsymbol{p}_n = \lambda \left(\bar{\boldsymbol{p}} - \boldsymbol{p}_o \right) + \boldsymbol{p}_o, \tag{4.8}$$

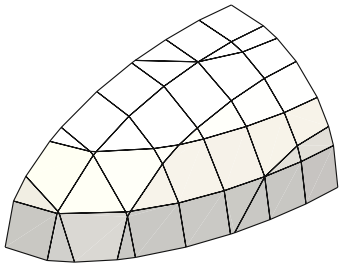
where $\boldsymbol{p}_o$ is the original coordinate and λ is a parameter to control the smoothing speed. The value of λ ranges from $(0, 1]$. A larger λ moves the nodes faster, however sometimes it will result in oscillation of the node position, which makes it difficult to converge. Usually the a value between 0.2 and 0.4 is selected. The smoothing operation iterates several times until the mesh quality is good enough. A comparison between different iteration steps is shown in Fig. 4.14.

As the element quality inside the octree mesh is usually good, the smoothing operation doesn't modify the location of the nodes inside. The smoothing operation is restricted to the boundary layer and the the nodes directly connected to the boundary layer (which is referred to as the second boundary layer). This restriction also improves the efficiency of the algorithm. In order to preserve original geometry, several rules are followed when using Eq. (4.7):

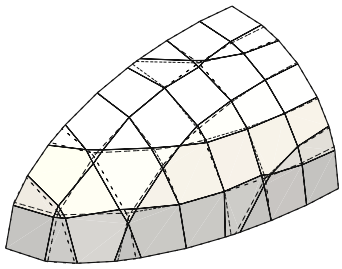
1. The node on a corner will not be smoothed;

2. The node on a ridge will only be smoothed using the neighbor nodes on the same ridge;

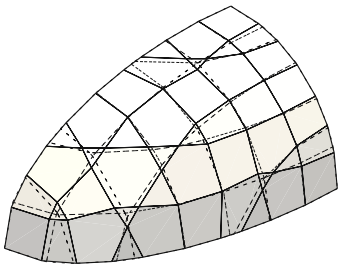
3. The node on the boundary will only be smoothed using the neighbor nodes on the boundary.



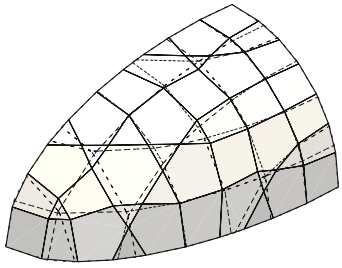
(a) Polyhedron mesh of a part of an ellipsoid before smoothing



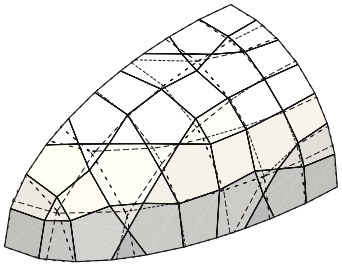
(b) Iteration step 1



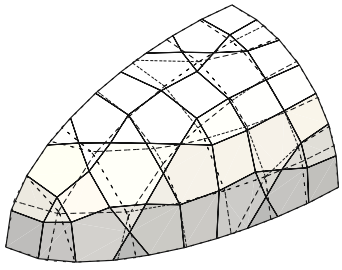
(c) Iteration step 2



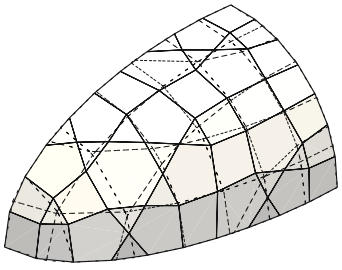
(d) Iteration step 3



(e) Iteration step 5

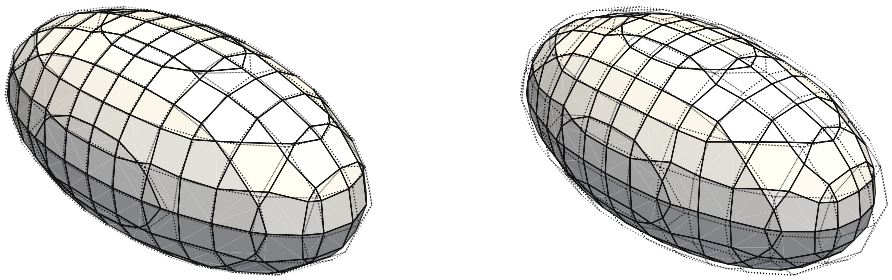


(f) Iteration step 10



(g) Iteration step 20

Figure 4.14: Smoothing at different steps using $\lambda = 0.3$



(a) $\lambda = 0.2, iter = 3$ (b) $\lambda = 0.2, iter = 10$

Figure 4.15: Shrinking of the boundary without projection

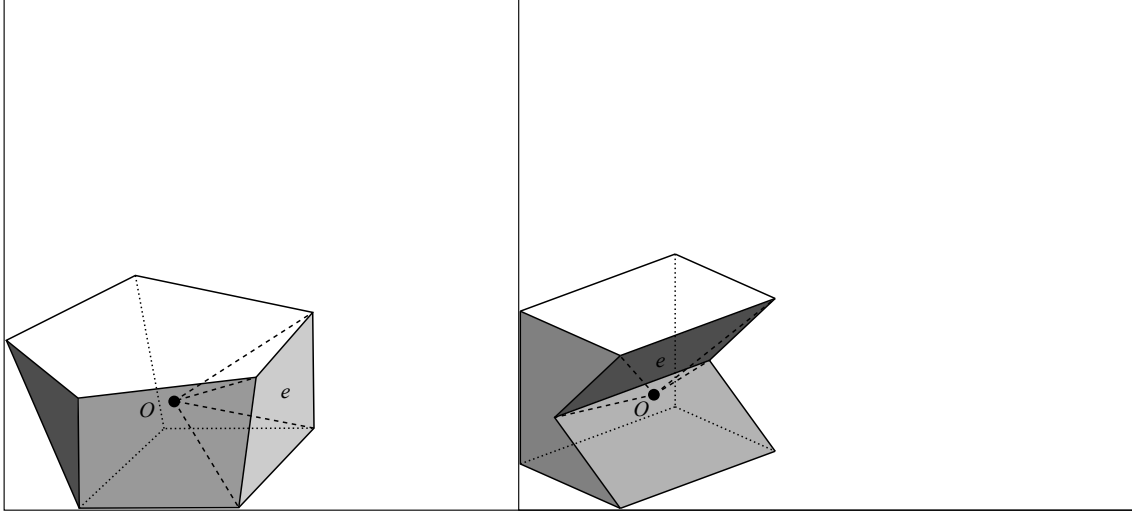
After each smoothing step, the boundary layer needs to be projected to the original geometry. Otherwise a convex geometry will shrink as shown in Fig. 4.14 (the dash line represents the original geometry).

4.7.2 Validity of S-domains

In the scaled boundary finite element method, the S-domains need to satisfy the scaling requirement, i.e. a point from which all the boundary is directly visible can be identified. The point is defined as scaling center. Therefore, it is necessary at this stage to check the scaling requirement and locate the scaling centers before the mesh is passed to the solver. The procedure is discussed in this section.

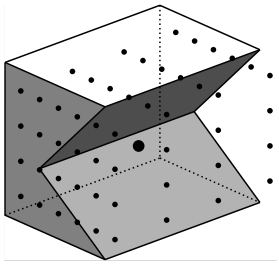
The barycenter of the all the nodes in a cell is calculated as a first trial point. It is checked that whether all the surface elements can form a positive volume with the barycenter. For example, in Fig. 4.16a, the barycenter O forms a positive volume with the surface element e, along with all the other surface elements. Therefore O is selected as the scaling center. Convex polyhedron always satisfy visibility condition and the barycenter is always a valid scaling center.

If the volume formed by barycenter and a surface element is negative or nearly zero, it means the surface element is invisible from the barycenter (see Fig. 4.16b, barycenter O is located outside the subdomain). This doesn't necessarily mean that the S-domains is invalid, the visibility might be improved by selecting another scaling center. A set

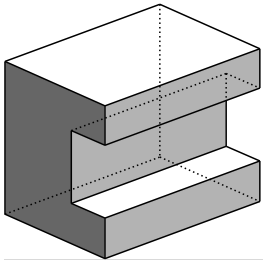


(a) Positive volume formed by the barycenter and surface element

(b) Negative volume formed by the barycenter (outside the polyhedron) and surface element



(c) Group of the trial points of the concave polyhedron



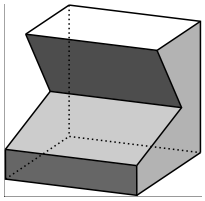
(d) Concave polyhedron not satisfying the scaling requirement

Figure 4.16: Location of the scaling centers

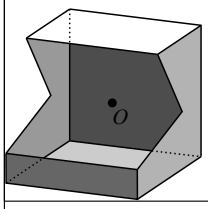
of points are generated surrounding the barycenter as new trial points (the points in Fig. 4.16c). The scaling requirement is checked for the trial points one by one. If some of them satisfy the scaling requirement, the point which has the best visibility will be selected as the final scaling center. Otherwise, if no such point can be identified (see Fig. 4.16d), the S-domains is invalid and will be discarded.

4.7.3 Open S-domains

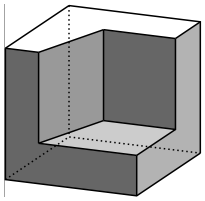
As shown in Section 4.5.2 and Section 4.5.3, the trimming of cells containing sharp ridges and corners is more complex than the trimming of other cells. Sometimes the resulting cells cannot satisfy the visibility condition or the visibility is poor if the scaling centers are placed inside the cells. Therefore it is necessary to develop a scheme which can reduce the complexity of the trimming procedure and produce cells with better



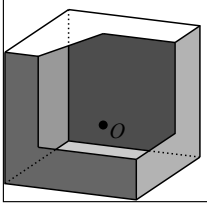
(a) A closed S-domain with a concave ridge



(b) An open S-domain with a concave ridge



(c) A closed S-domain with a concave corner



(d) An open S-domain with a concave corner

Figure 4.17: Concave S-domains

quality.

One of the advantages of the SBFEM is that the polyhedron cells don't have to be closed. The scaling center can be placed at the corner in the geometry and the boundary is scaled towards the scaling center to form a volume. This means as long as the boundary of a cell is properly trimmed, the volume is naturally represented without further operation. The side faces connecting the scaling center and the boundary are not explicitly modeled. This advantage reduces the dimension of problem by one. The stress singularity around the concave corner can be represented analytically in the SBFEM shape function.

If there is only one sharp ridge passing through a cell, the scaling center can be placed anywhere on that ridge inside the cell. However, it is usually placed in the middle of the segment (Fig. 4.17b) for better visibility. If a corner is located inside the cell, the scaling center will be place at the corner (Fig. 4.17d).

The complete procedure of polyhedron mesh generation is shown in Algorithm 4.2.

Algorithm 4.2 Polyhedron mesh generation based on octree algorithm

```
Input model
Extract sharp ridges and corners
Define background mesh parameters: bounding box, maximum and minimum element size
Define refinement criteria and mesh size transition rate
Define smoothing parameters: lambda, iteration number
Calculate bounding box
while minimum refinement level not reached do
    Octree mesh refinement
end while
Balance octree mesh
Convert octree data to polyhedron data
Obtain intersection mark
Obtain ridge and corner mark
Trim edges
Trim faces
Trim cells
while iteration step < iteration number do
    Smooth boundary layer
    Project boundary layer
    Smooth second boundary layer
end while
Boundary triangulation
Calculate scaling center
Discard invalid cells
Output mesh
```

4.8 Numerical examples

In this section, six numerical examples are presented. To verify the meshes generated by the proposed method are valid, linear elastic analysis are performed. The accuracy of the results is evaluated by the relative error norm in displacement:

$$e_u = \frac{\|\boldsymbol{u}_{num} - \boldsymbol{u}_{ref}\|}{\|\boldsymbol{u}_{ref}\|}, \tag{4.9}$$

where $\boldsymbol{u}_{num}$ is the numerical result calculated by the proposed method. Analytical solution can be used as reference when available. When analytical solution is not available, numerical result calculated from refined mesh or high order elements are used as reference solutions.

4.8.1 A pressurized hollow sphere

A pressurized hollow sphere is shown in Fig. 4.18. Only an octant of the sphere is modeled due to symmetry. The radius of the inner sphere and the outer sphere are $a = 1$m and $b = 2$m. A uniform normal pressure $P = 1$Pa is applied on the inner surface of the sphere. At the surfaces at $x = 0$, $y = 0$ and $z = 0$, the displacements perpendicular to surfaces are prescribed as zero. The Young's modulus is $E = 1$Pa and Poisson's ratio $\nu = 0.3$.

The exact solution can be found in (Bower, 2009). The displacement solution in spherical coordinates r, θ, ϕ is expressed as

$$u_r = \frac{P}{2E}\frac{a^3}{b^3 - a^3}\left(2\left(1 - 2\nu\right) + \left(1 + \nu\right)\frac{b^3}{r^2}\right), \tag{4.10a}$$

$$u_\theta = u_\phi = 0, \tag{4.10b}$$

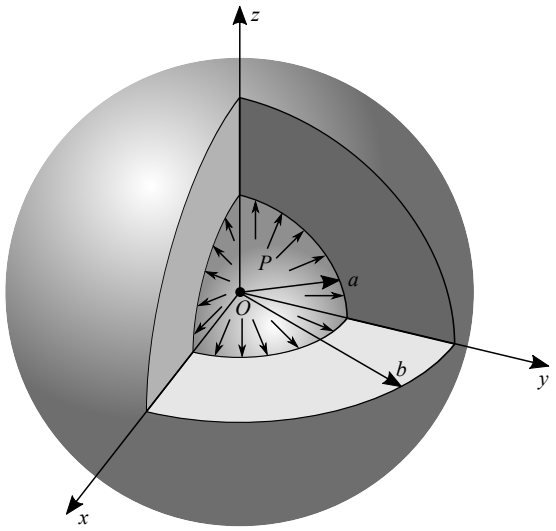


Figure 4.18: Geometry and boundary conditions of the hollow sphere

and the stress solution is

$$\sigma_r = P\frac{a^3}{b^3 - a^3}\frac{r^3 - b^3}{r^3}, \tag{4.11a}$$

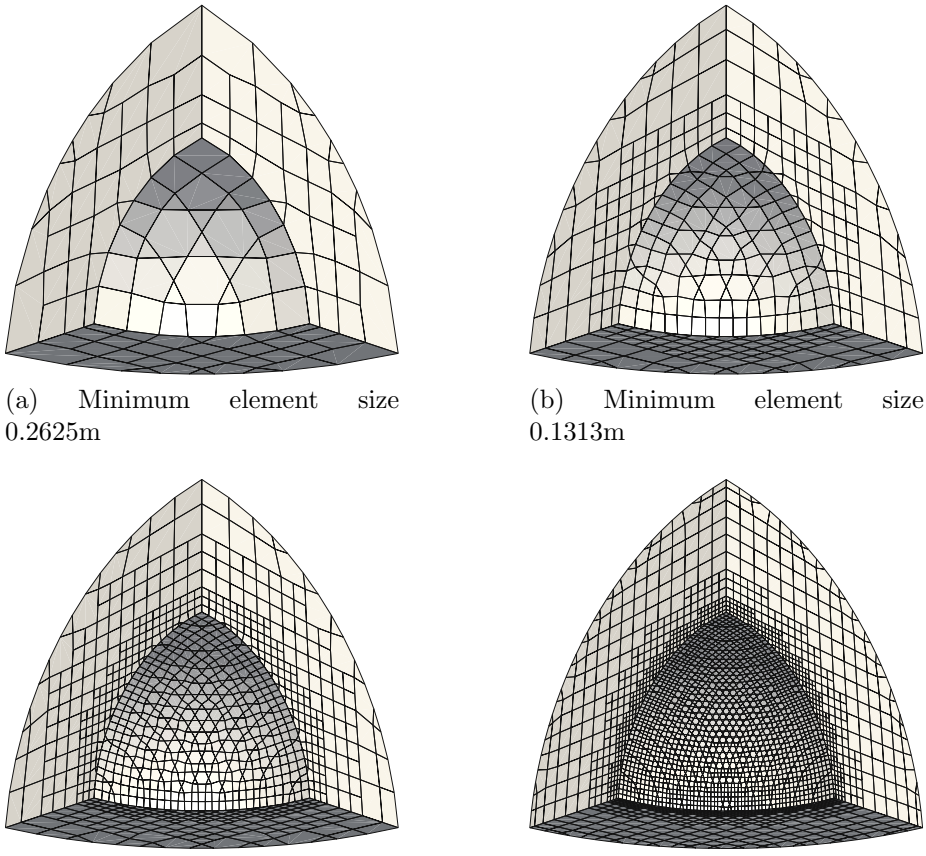
$$\sigma_\theta = \sigma_\phi = \frac{P}{2}\frac{a^3}{b^3 - a^3}\frac{2r^3 + b^3}{r^3}, \tag{4.11b}$$

$$\tau_{\theta\phi} = \tau_{\phi r} = \tau_{r\theta} = 0. \tag{4.11c}$$

Meshes of different element sizes are generated as shown in Fig. 4.19. The two corners of the bounding box are fixed, the coordinates of which are $(0.0\text{m}, 0.0\text{m}, 0.0\text{m})$ and $(2.1\text{m}, 2.1\text{m}, 2.1\text{m})$. The maximum mesh size ratio is set to be 4. The minimum element sizes are 0.2625m, 0.1313m, 0.0656m, 0.0328m, and 0.0164m respectively. The mesh is refined to the minimum element size on the inner surface. The mesh size transition rate is $m = 2$. The displacements are plotted in Fig. 4.20.

A convergence study is performed using linear and quadratic elements. The theoretical error norm of finite elements is expressed as

$$error = O\left(DOF^{-\frac{p+1}{3}}\right), \tag{4.12}$$



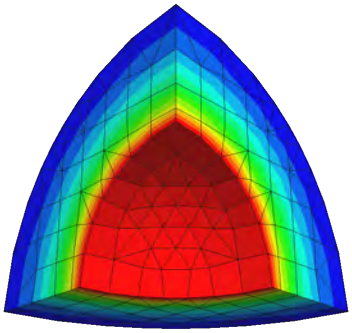
(a) Minimum element size 0.2625m

(b) Minimum element size 0.1313m

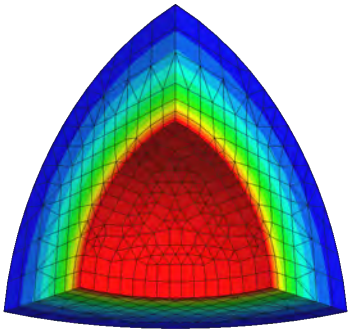
(c) Minimum element size 0.0656m

(d) Minimum element size 0.0328m

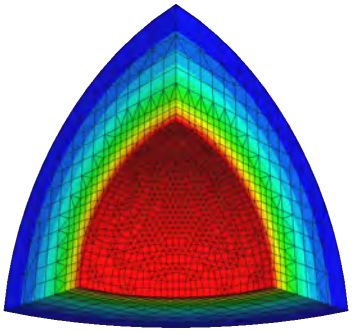
Figure 4.19: Meshes of different element sizes of the hollow sphere



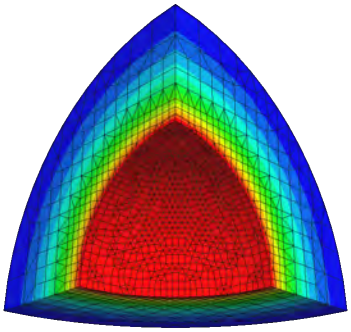
(a) Minimum element size 0.2625m



(b) Minimum element size 0.1313m



(c) Minimum element size 0.0656m



(d) Minimum element size 0.0328m

Figure 4.20: Displacement of the hollow sphere

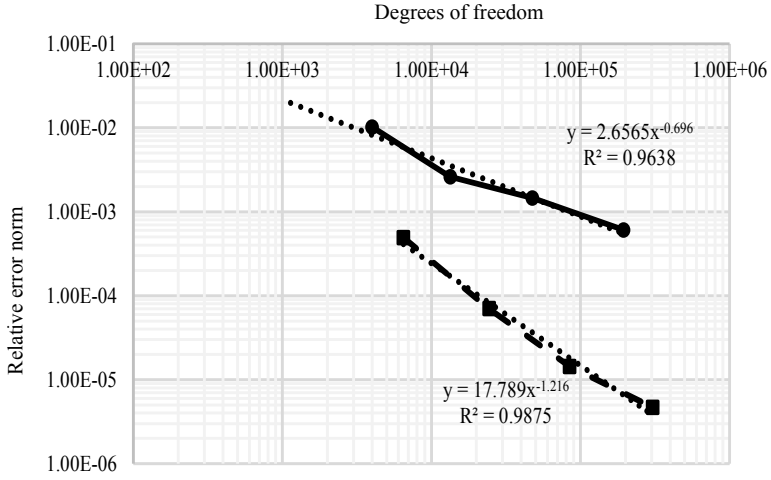


Figure 4.21: Convergence behavior of the hollow sphere in terms of error norm in displacement

where DOF is degrees of freedom and p is the order of element.

The relative error norm is plotted versus DOF in log-log scale in Fig. 4.21. The convergence rates are calculated by fitting the curves with power functions. The theoretical convergence rates are achieved for both linear and quadratic elements.

4.8.2 An ellipsoid with two holes

In this example, a ellipsoid with two holes is considered. The geometry is shown in Fig. 4.22. Only an octant of the ellipsoid is modeled because of symmetry. The longer diameter of the ellipsoid is $a = 2$m and the shorter diameter $b = 1$m. The diameters of the two spheres are both $c = 0.4$m. Only self-weight is considered. At the surfaces at $x = 0$, $y = 0$ and $z = 0$, the displacements perpendicular to the surfaces are constrained. The Young's modulus is $E = 1$Pa and Poisson's ratio is $\nu = 0.3$. The mass density of the material is $\rho = 1\text{kg/m}^3$.

Meshes of different sizes are generated as shown in Fig. 4.23. The two corners of the bounding box are $(0.0\text{m}, 0.0\text{m}, 0.0\text{m})$ and $(0.55\text{m}, 0.55\text{m}, 1.1\text{m})$. The maximum mesh size ratio is 4. The minimum element sizes are 0.0688m, 0.0344m, 0.0172m, and 0.0086m. The mesh is refined to the minimum element size on the inner and outer

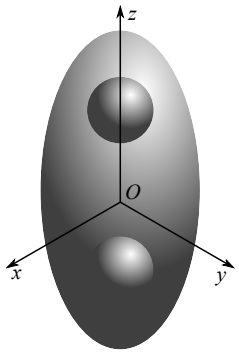
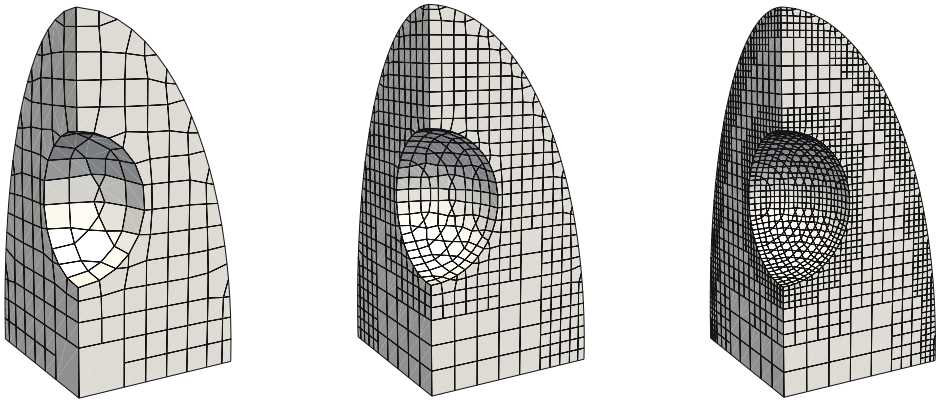


Figure 4.22: The geometry of the ellipsoid with two holes



(a) Minimum element size 0.0688m

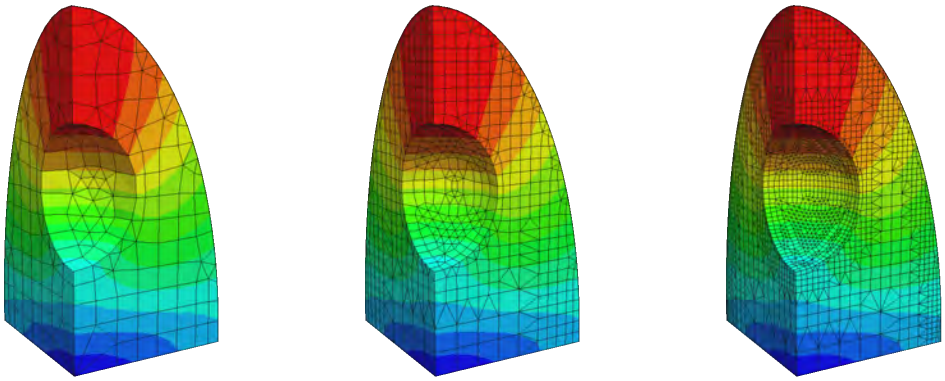
(b) Minimum element size 0.0344m

(c) Minimum element size 0.0172m

Figure 4.23: Meshes of the the ellipsoid

boundaries. The mesh size transition rate is $m = 2$. The displacements in z direction of different meshes are plotted in Fig. 4.24.

The point at the top of the ellipsoid is selected to compare the results. A convergence study using a series of tetrahedron meshes of second order finite elements (C3D10) is performed in ABAQUS to obtain the reference solution. The relative error of the displacement in z direction at the point is plotted versus the number of degrees of freedom in log-log scale in Fig. 4.25.



(a) Minimum element size 0.0688m

(b) Minimum element size 0.0344m

(c) Minimum element size 0.0172m

Figure 4.24: Displacement of the ellipsoid

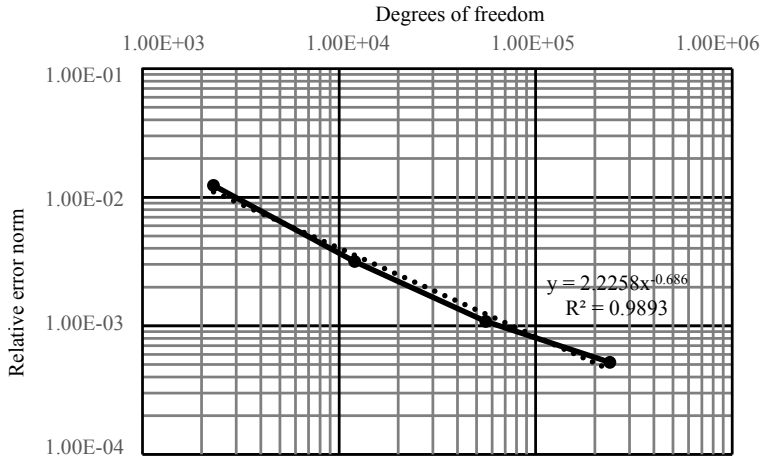


Figure 4.25: Convergence behavior of the ellipsoid in terms of error norm in displacement

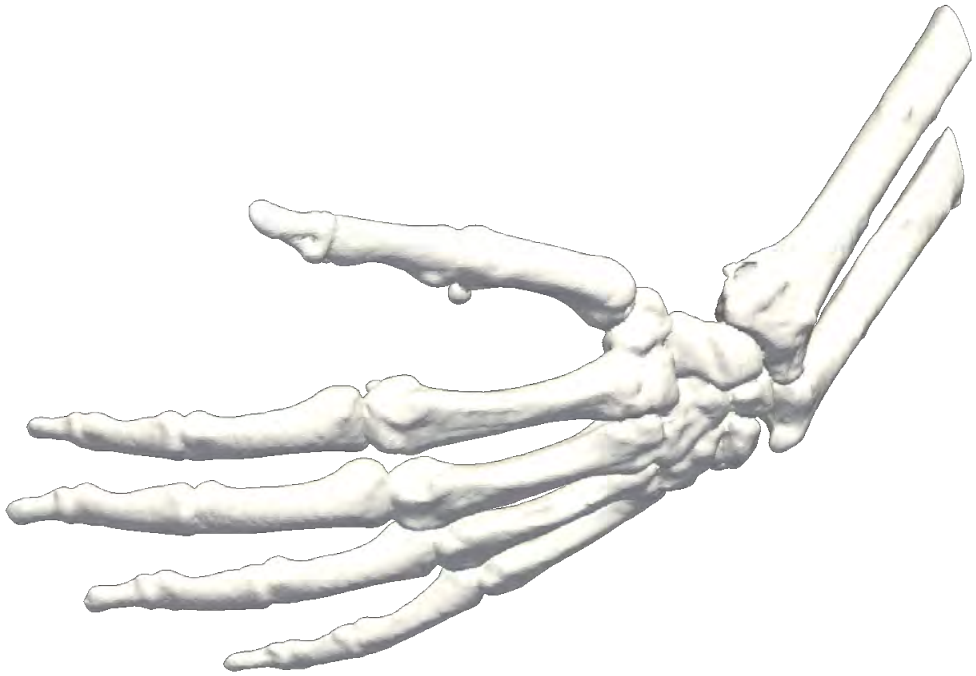


Figure 4.26: Geometry of the hand bone

4.8.3 A human hand bone

An STL model of a human hand bone is shown in Fig. 4.26 (`https://www.thingiverse.com/thing:147800`). The geometry of the model is irregular and there are a large number of distorted triangles, which makes it difficult to generate a standard finite element mesh. However, a polyhedron mesh with relative good quality can be generated using the proposed method automatically, as shown in Fig. 4.27. The material is cortical bone. The Young's modulus is $E = 1.7 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.3$. The mass density of the material is $\rho = 1.9 \times 10^3 \text{kg/m}^3$. The bounding box of this model is 53.8462mm $\times$ 76.9231mm $\times$ 100mm. The maximum element size is 0.9616mm and the minimum element size is 0.2404mm.

A modal analysis is performed without applying any constraints. The mesh is clipped using a plane perpendicular to y axis (Fig. 4.28). The magnitude of displacement of the 14th mode shape is plotted.

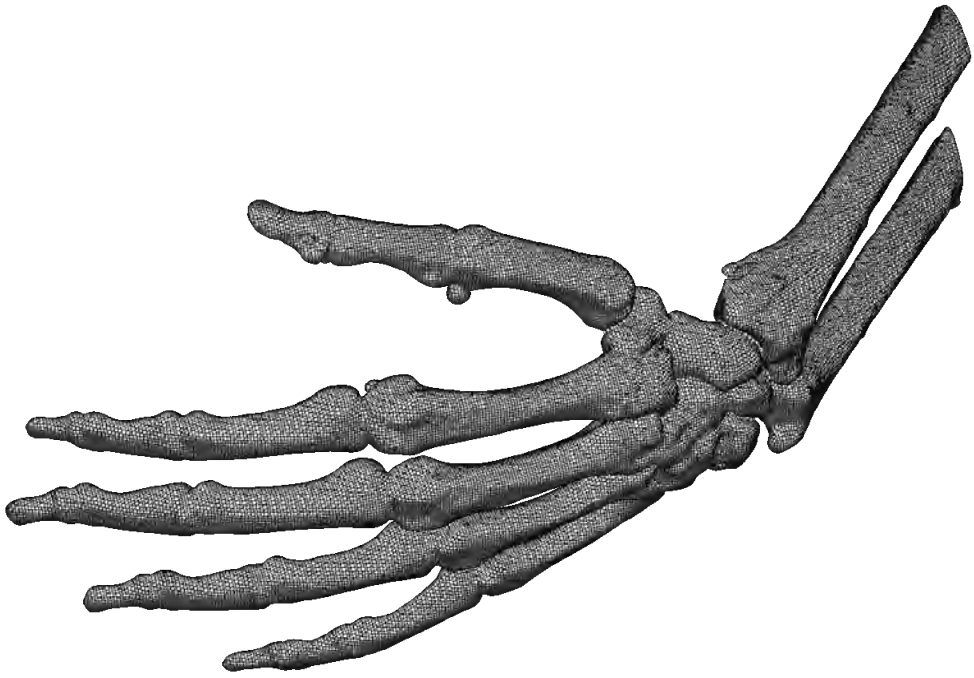


Figure 4.27: Mesh of the hand bone

4.8.4 A socket

In this example, a socket is considered. The geometry of the model is shown in Fig. 4.29a. It can be observed that the model contains sharp ridges and corners. A uniform distributed load $P = 1000$Pa is applied to the surfaces shown in Fig. 4.29b. The top surface of the model is fixed in all directions.

A polyhedron mesh is generated using the proposed method as shown in Fig. 4.30. The maximum element size is 1.5mm and the minimum element size is 0.375mm.

The Young's modulus of ABS plastic is $E = 2.6 \times 10^9$Pa and Poisson's ratio is $\nu = 0.35$. The mass density is $\rho = 1.4 \times 10^3 \text{kg/m}^3$. The magnitude of the displacement is plotted in Fig. 4.31a. A clip view of the model is shown in Fig. 4.31b.

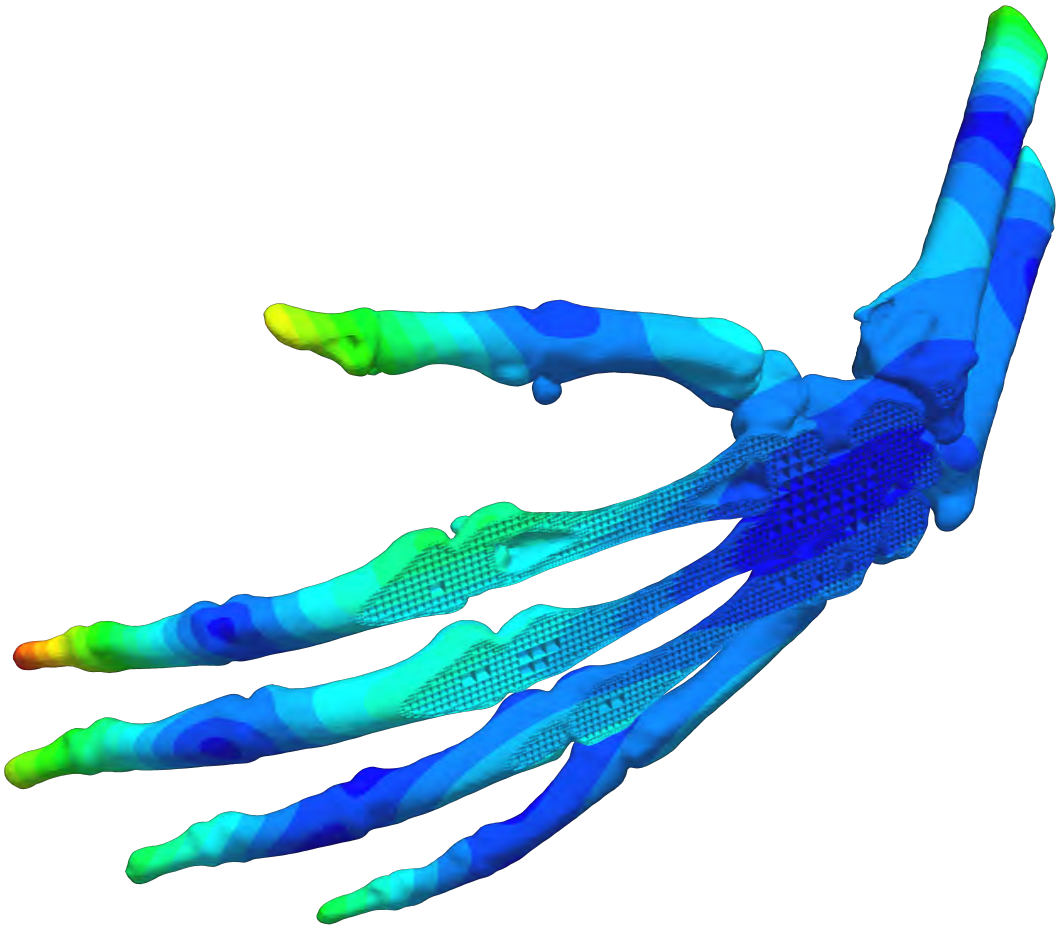
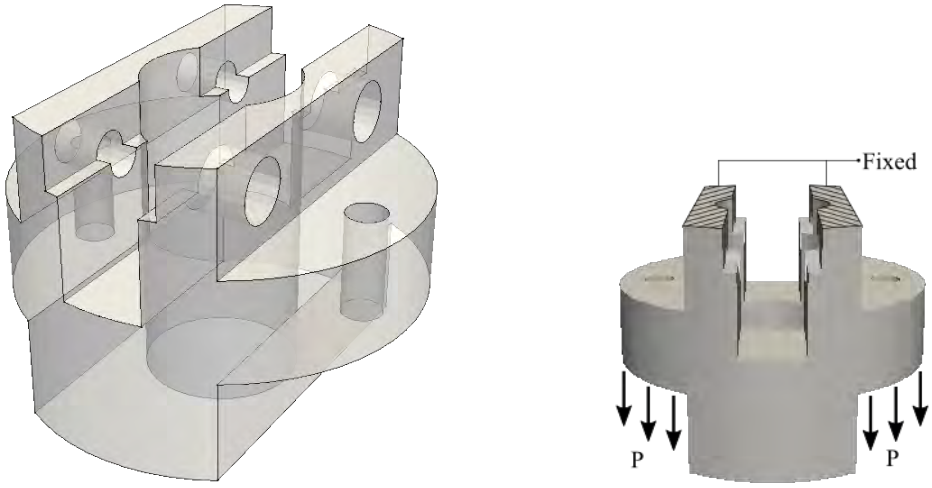


Figure 4.28: Mode shape of the hand bone



(a) Geometry of the socket

(b) Boundary conditions of the socket

Figure 4.29: Geometry and boundary conditions of the socket

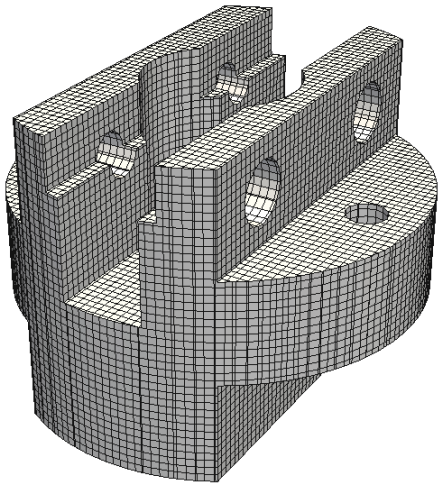
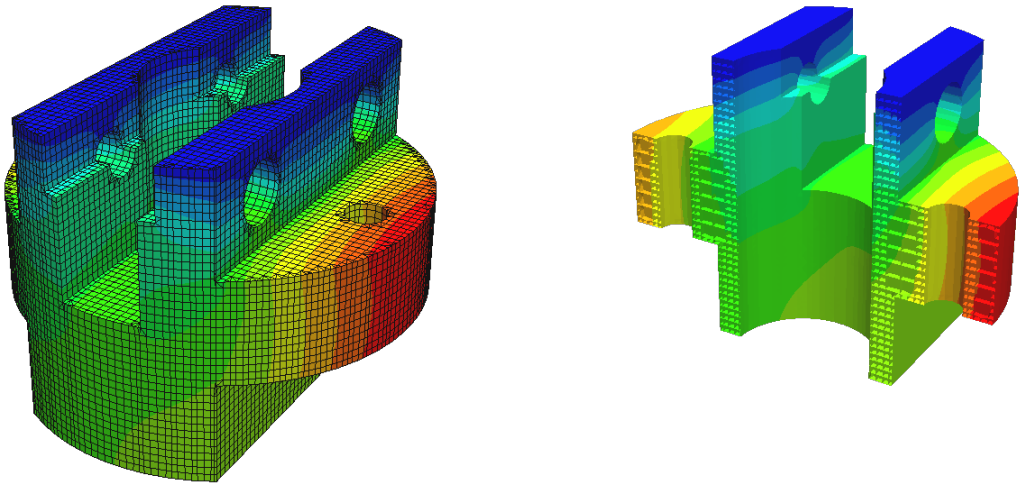


Figure 4.30: Mesh of the socket



(a) Overall view of the displacement (b) Clip view of the displacement

Figure 4.31: Displacement of the socket

4.8.5 A car engine

A car engine with 8 cylinders is considered in this example (`https://www.thingiverse.com/thing:2007841`). Only 2 cylinders on the left side are covered by the bounding box as shown in Fig. 4.32. The dimensions of the bounding box is 28.5714mm × 100mm × 71.4286mm. The maximum element size is 3.5714mm and the minimum element size is 0.4464mm. The octree mesh is shown in Fig. 4.33a. The Young's modulus of the cast iron is $E = 9 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.2$. The mass density is $\rho = 7.8 \times 10^3 \text{kg/m}^3$. The bottom of the engine is fixed and self-weight of the engine is considered. The displacement of the engine is plotted in Fig. 4.33b.

4.8.6 A Chinese guardian lion

A sculpture of a Chinese guardian lion is modeled in this example (`https://www.thingiverse.com/thing:635242`). The geometry of the STL model is shown in Fig. 4.34. A clipped view of the mesh is shown in Fig. 4.35. The size ratio between maximum element and minimum element in the mesh is 32.

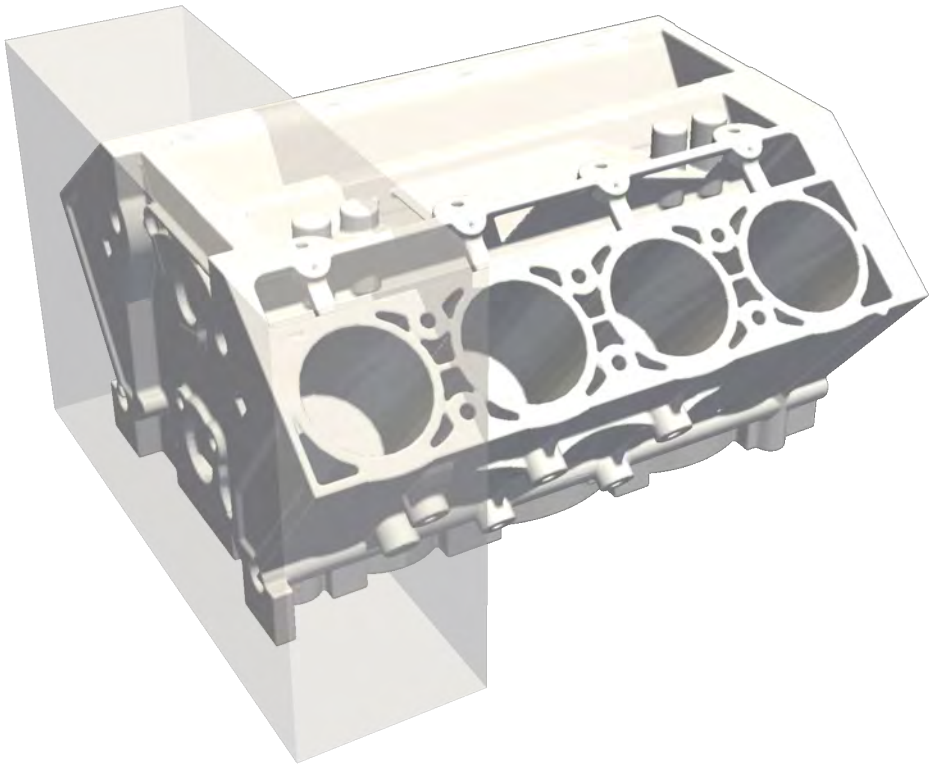
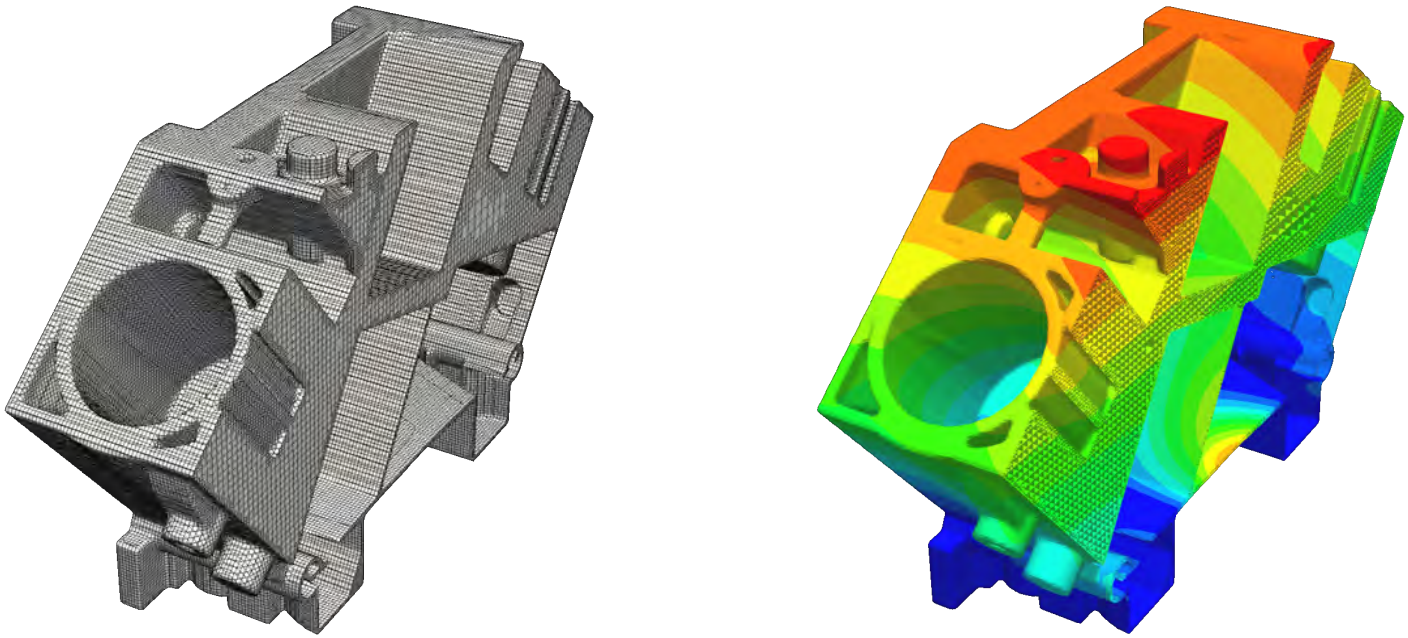


Figure 4.32: Geometry of the car engine



(a) Mesh of the engine

(b) Displacement of the engine

Figure 4.33: Mesh and displacement of the car engine

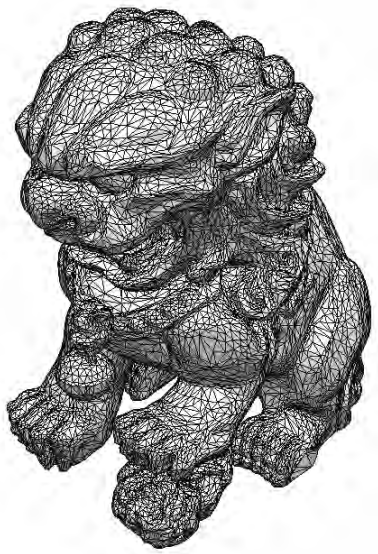


Figure 4.34: Geometry of the Chinese guardian lion

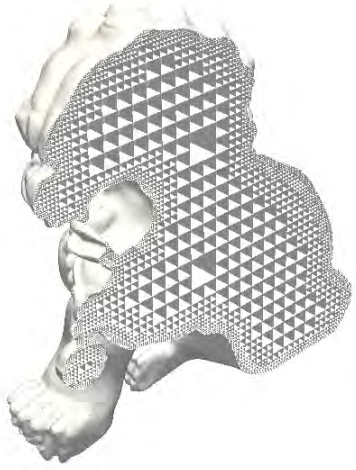


Figure 4.35: Mesh of the Chinese guardian lion

The Young's modulus of the stone is $E = 2 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.2$. The mass density is $\rho = 2.7 \times 10^{3}\text{kg/m}^{3}$. The bottom of the lion is fixed and self-weight is considered as external load. The displacement is plotted in Fig. 4.36.

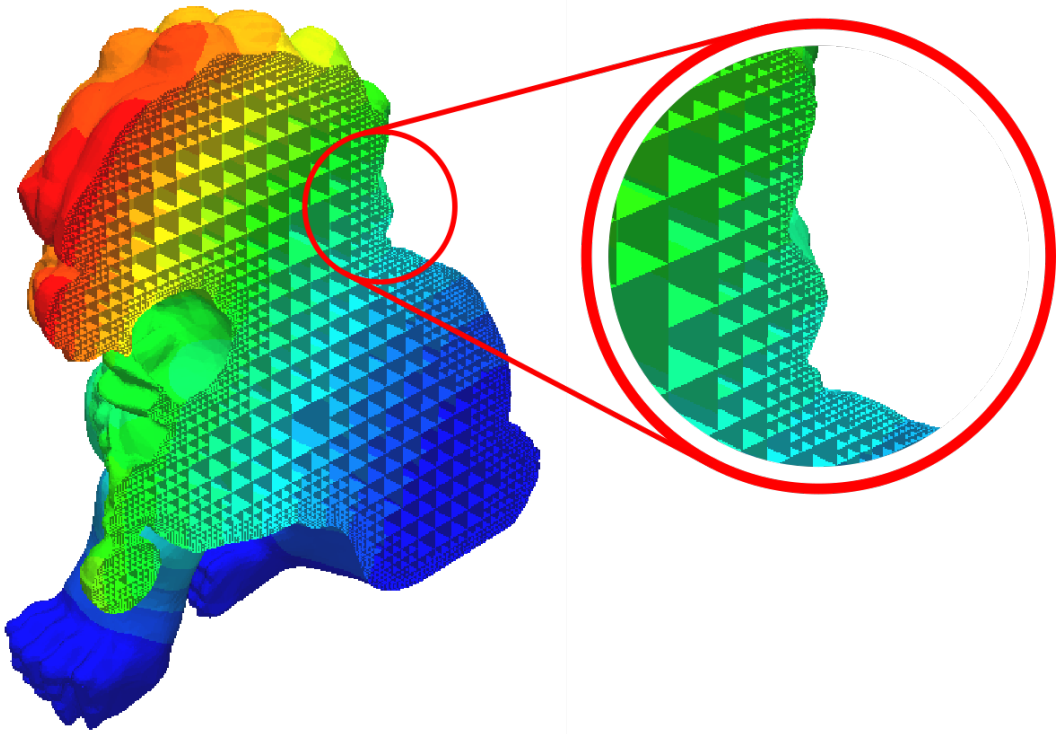


Figure 4.36: Displacement of the Chinese guardian lion

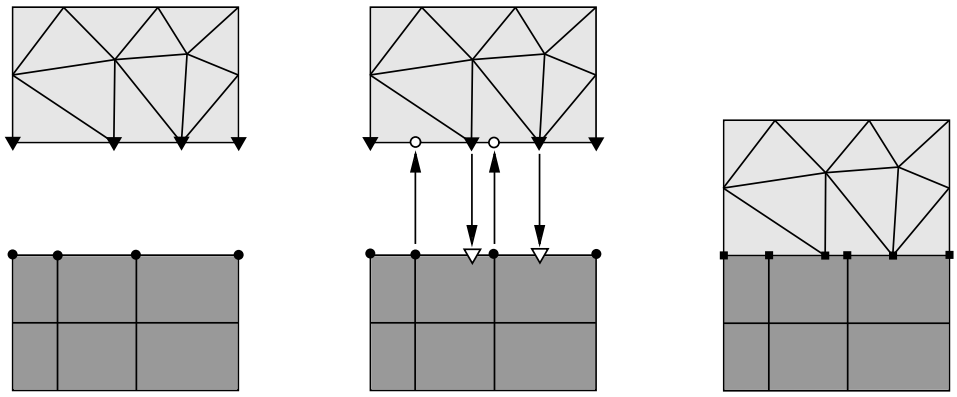
Chapter 5

Coupling of non-matching meshes

In the analysis of complex and large scale structures, it is desirable to divide the problem domain into several simpler subdomains. The meshes of the subdomains can be generated individually, which reduces the difficulty in meshing the complex models. Parallel processing can be employed for the mesh generation of the subdomains to further accelerate the whole analysis procedure. In a variety of other applications such as contact mechanics (Laursen et al., 2012), fluid-structure interaction (Bazilevs et al., 2012), multi-scale analysis (Unger and Eckardt, 2011), it is also preferable to model the subdomains individually.

The meshes of those subdomains are only connected by their interfaces. The interface meshes are matching if the mesh topologies and element shape functions are the same on both sides of the interface, otherwise they are non-matching. Obviously, the use of non-matching meshes offer greater flexibility in practice as they enable users to generate meshes for different subdomains independently. These independent meshes need to be coupled (or "glued") together when a numerical analysis is performed. Some of the techniques of coupling non-matching meshes have been summarized in Section 2.3.

In this research, a coupling method is presented which is able to handle arbitrary non-matching polygon meshes in 2D and polyhedron meshes in 3D (including standard



(a) Two non-matching meshes

(b) New nodes are inserted and edges are split

(c) Two meshes matching on the interface are created

Figure 5.1: Coupling of non-matching meshes in 2D

finite element meshes). The key idea is briefly illustrated in Fig. 5.1 using a 2D example with a straight interface. Two meshes generated independently are shown in Fig. 5.1a. Their interfaces are non-matching. The non-matching nodes of one mesh are inserted to the interface of the other mesh by splitting the edges of the elements (Fig. 5.1b). The elements affected by the node insertion are modeled as polygon elements. The boundary discretization of the polygon elements is constructed using the split edges to replace the original ones. As mentioned previously, domain discretization of the polygon elements is not needed. The two meshes become matching on the interface, as shown in Fig. 5.1c. The two meshes can be merged into one for analysis. This procedure can be applied recursively to combine multiple non-matching meshes. A MATLAB code implementing this procedure is found in Song (2018).

This approach is extended to 3D in the following of this chapter. The surface meshes on the common interface are extracted from two volume meshes. The two interface meshes at the opposite sides are merged and new polygons are constructed from intersected edges. Therefore the two volume meshes become matching on the interface (Section 5.1). A shifting procedure is designed before the merging to improve the element quality on the interface (Section 5.2). A refinement technique based on

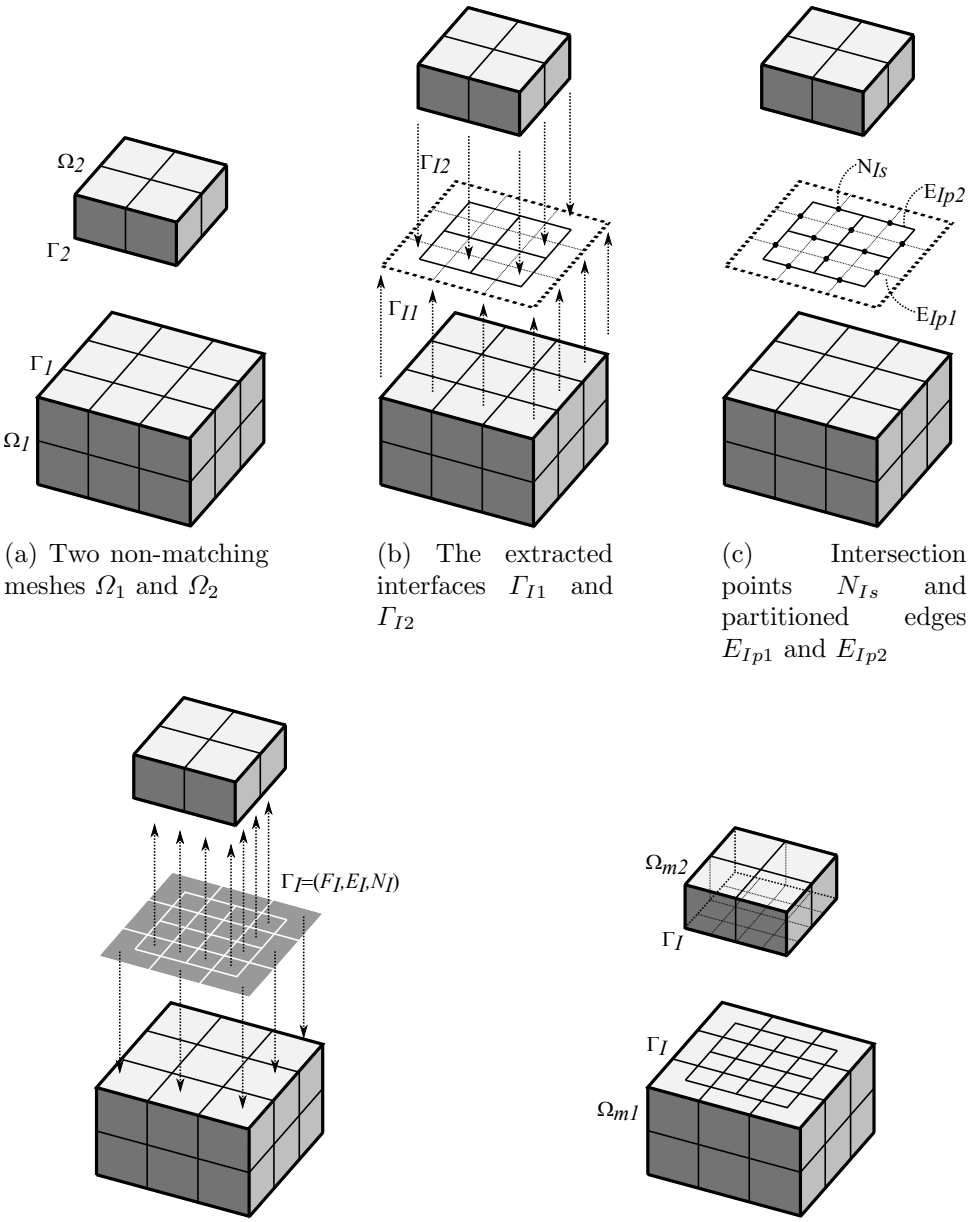
polytree algorithm is implemented in Section 5.3 to produce a smooth element size transition between the two volume meshes. As only the boundary of a polyhedron element needs to be discretized in an analysis, no volume discretization is required. The discretization scheme introduced in Section 4.6 is applied to the merged mesh. The proposed coupling method is verified using seven numerical examples. The results are compared to available analytical solutions or results from finite element analyses.

5.1 Construction of matching surface mesh

In a numerical method, a 3D mesh consists of volume elements, such as tetrahedra, hexahedra and polyhedra. The boundary of a 3D mesh is described by a surface mesh. The typical surface elements are triangles, quadrilaterals and polygons. The polyhedron element constructed using the scaled boundary finite element formulation in Chapter 3 is highly flexible in geometry. The element may have arbitrary number of faces, edges and nodes. Therefore, the polyhedron elements are utilized in this approach to couple the non-matching meshes. The data structure of polyhedron mesh has been presented in Section 4.1.2.

The interface meshes of the two non-matching meshes are extracted based on the geometry of the two subdomains (Section 5.1.1). On the interface, a new surface mesh of polygon elements is constructed from the intersected edges (Section 5.1.2). The polyhedron elements connected to the interface are modified by replacing their faces on the interface with the new polygon elements, leading to matching discretization on the interface (Section 5.1.3). The modified polyhedron elements are modeled by the scaled boundary finite element method and no further volume discretization is required.

The coupling of the two non-matching polyhedron meshes Ω_1 and Ω_2 shown in Fig. 5.2a is used as an example to illustrate the proposed approach. The boundaries are expressed as Γ_1 and Γ_2.



(a) Two non-matching meshes Ω_1 and Ω_2

(b) The extracted interfaces Γ_{I1} and Γ_{I2}

(c) Intersection points N_{Is} and partitioned edges E_{Ip1} and E_{Ip2}

(d) Common interface mesh Γ_I is assembled back to Ω_1 and Ω_2

(e) The matching meshes Ω_{m1} and Ω_{m2}

Figure 5.2: Construction of matching surface mesh

5.1.1 Extraction of interface

The interfaces of the two meshes, Γ_{I1} and Γ_{I2} are the patches in Γ_1 and Γ_2 which are overlapping in geometry (see Fig. 5.2b). Generally, the two patches are not the same, i.e. $\Gamma_{I1} \neq \Gamma_{I2}$. A face f_i in the boundary face set F_{S1} will be extracted and included in the interface if:

1. For all the nodes of the face, $\exists n \in f_i$, the distance between n and any $f_j \in F_{S2}$ is less than a tolerance ε_d; and

2. The normal vector $\boldsymbol{v}_i$ of f_i is opposite to at least one normal vector $\boldsymbol{v}_j$ of a nearby face $f_j \in F_{S2}$, i.e. $\frac{\boldsymbol{v}_i \cdot \boldsymbol{v}_j}{\|\boldsymbol{v}_i\| \times \|\boldsymbol{v}_i\|} < \cos(\varepsilon_\theta)$, where ε_θ is a parameter related to the curvature of the interface.

The two parameters ε_d and ε_θ are used to control the initial gap and penetration due to different discretizations. ε_d is related to the mesh size, for example, can be chosen as 0.1 times the minimum edge length. ε_θ should be smaller than the minimum angle formed by the normal vectors of the nearby faces on two sides of the interface. Typically a value between 150° and 170° is selected. Usually the higher the curvature of the interface is, the smaller ε_θ should be. However, ε_θ should always be larger than the dihedral angle formed by the interface and the side faces of the model.

The first requirement guarantees that all the faces, including those only partially overlapping with the other mesh, can be extracted. The second requirement, on the other hand, excludes all the side faces which are not on the interface.

5.1.2 Insertion of non-matching nodes and edges

The two interfaces Γ_{I1} and Γ_{I2} are matched by inserting non-matching nodes and edges from one interface to the other as in the 2D case shown in Fig. 5.1. A common interface Γ_I is constructed as $\Gamma_I = (F_I, E_I, N_I)$. The set of new nodes of the common interface is $N_I = N_{I1} \cup N_{I2} \cup N_{Is}$. N_{I1} and N_{I2} are the nodes copied from Γ_{I1} and Γ_{I2}. N_{Is} is the

set of intersection points of E_{I1} and E_{I2} (see Fig. 5.2c). E_{I1} and E_{I2} are partitioned at those intersection points. $E_I = E_{Ip1} \cup E_{Ip2}$, where E_{Ip1} and E_{Ip2} are the set of new edges partitioned from E_{I1} and E_{I2}. F_I is the set of new faces enclosed by E_I.

5.1.3 Assembly of matching surface mesh

The matching surface mesh Γ_I is then connected to the volume meshes Ω_1 and Ω_2, as shown in Fig. 5.2d. If a face f_i of a polyhedron cell c_j is on the interface, it is replaced by the corresponding set of new faces. The set of faces in the cell c_j is updated accordingly.

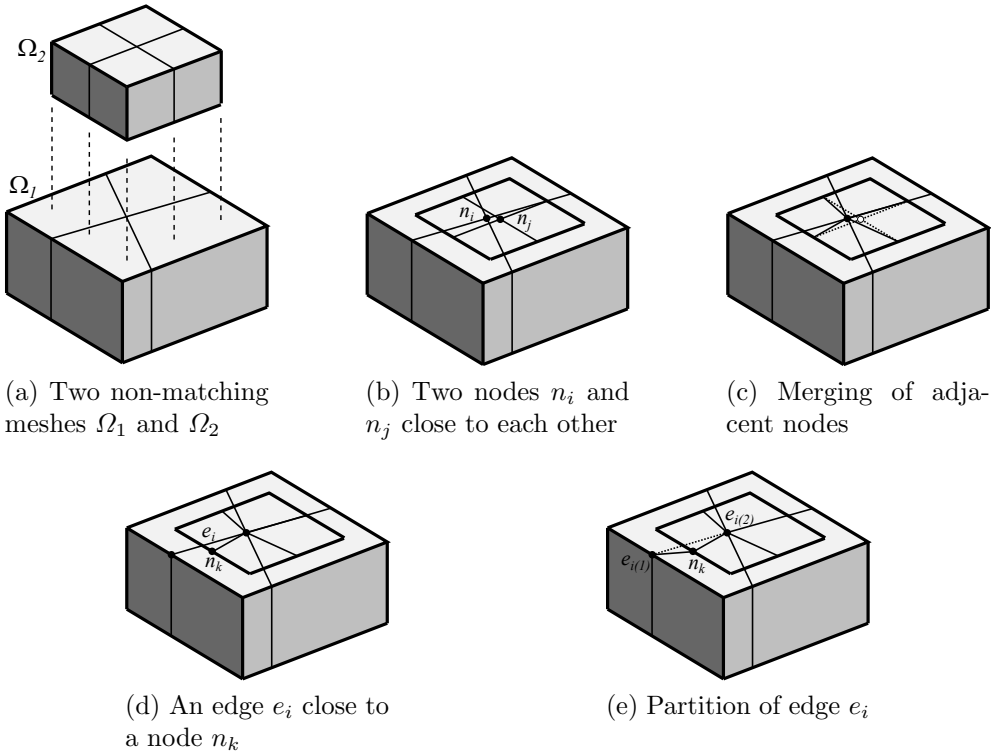
The two new volume meshes Ω_{m1} and Ω_{m2} share the same surface mesh Γ_I on the interface. Therefore, the matching meshes are constructed (Fig. 5.2e). Standard procedures of the finite element method for conforming meshes are directly applicable.

5.2 Optimization of surface mesh quality

On the interface of the two non-matching meshes, some nodes and edges on one surface mesh may be close to those on the other mesh. Coupling the meshes directly can result in short edges and distorted surface elements. These distorted elements not only reduce the robustness of the merging procedure, but also affect the accuracy of a stress analysis. A shifting operation is designed to eliminate the potential short edges and distorted elements. This operation is performed before the intersection calculation mentioned in Section 5.1. This operation includes two steps, with the first step merging adjacent nodes and the second step handling the edges. An illustrative example is shown in Fig. 5.3.

5.2.1 Merging of adjacent nodes

If a node $n_i \in N_{I1}$ is close to a node $n_j \in N_{I2}$, the two nodes will be merged into one (Fig. 5.3c). In order to ensure the validity of the surface mesh, the merging tolerance



(a) Two non-matching meshes Ω_1 and Ω_2

(b) Two nodes n_i and n_j close to each other

(c) Merging of adjacent nodes

(d) An edge e_i close to a node n_k

(e) Partition of edge e_i

Figure 5.3: Optimization of surface mesh quality

is set to be related to the element size near each node.

The merging tolerance at the node n_i is defined as $tol = \varepsilon_n \times l(n_i)$, where $l(n_i) = \min(l(e_{k1}), l(e_{k2}), \cdots, l(e_{kp}))$. $e_{k1}, e_{k2}, \cdots, e_{kp} \in E_{I1}$ are the edges connected to n_i. $l(e)$ represents the length of en edge e. ε_n is a predefined relative tolerance. Typically, a value between 0.2 and 0.3 is selected.

If the distance between the two nodes $d(n_i, n_j) < \varepsilon_n \times \min(l(n_i), l(n_j))$, the two nodes will be merged. The merged node is usually located at the average coordinate of the two original nodes, except for the cases included in Section 5.2.3.

5.2.2 Partition of edges close to nodes

Only merging the nodes may not eliminate all the short edges. The edges which are close to nodes in the other mesh need to be processed as well.

If an edge $e_i \in E_{I1}$ is close to a node $n_k \in N_{I2}$ within a specified tolerance (Fig. 5.3d),

two new edges, $(e_i(1), n_k)$ and $(n_k, e_i(2))$, will be generated to replace the original e_i (Fig. 5.3e). $e_i(1)$ and $e_i(2)$ represent the two end nodes of the original edge e_i.

The tolerance is $tol = \varepsilon_e \times \min(l(e_i), l(n_k))$. The relative tolerance ε_e is a value smaller than ε_n, and is typically chosen as $0.1 \sim 0.2$.

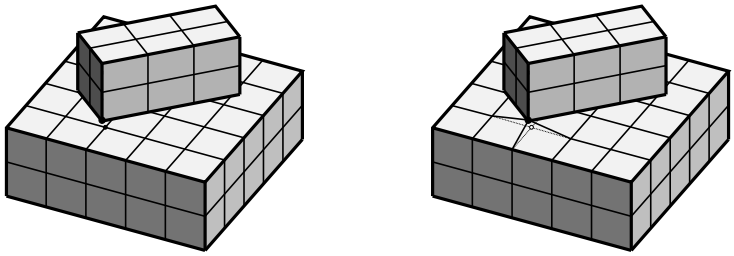
5.2.3 Preserving corners and ridges of shapes

In computational geometry, a ridge is an edge connecting two boundary faces which form a sharp dihedral angle (Seidel, 1986). A corner is a node connected to two or more ridges. The merging and partition operations above may shift the ridges and corners from their exact locations in the original geometry. To preserve the geometry, special treatments are applied on the ridges and corners on the interface. The rules are stated as follows:

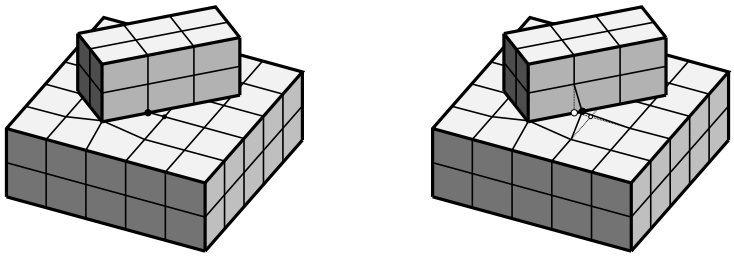
1. The node on a corner will not be moved (see Fig. 5.4a).
2. The node on a ridge will only be moved along the same ridge (see Fig. 5.4b).
3. The edge on a ridge will only be partitioned into new edges on the same ridge (see Fig. 5.4c).

5.3 Transition of mesh size

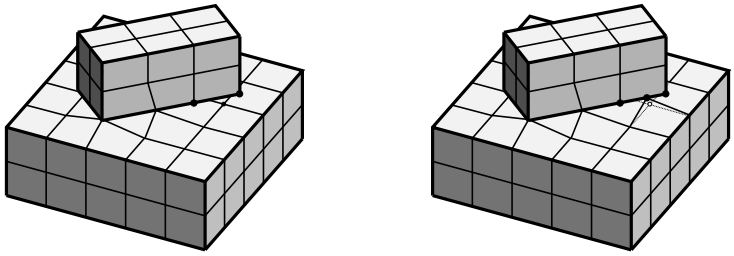
Meshes with significantly different element sizes usually cause difficulties in coupling as well as in analysis. Degenerated polygons will appear on the merged interface if the lengths of its edges are significantly different. In the present approach, the mesh with the larger element size will be refined near the interface until the difference between the sizes of the two meshes is acceptable. The refinement using polytree algorithm (Spring et al., 2014; Nguyen-Xuan et al., 2017b) involves the treatment of surfaces (2D) and volumes (3D). Surface refinement can be performed as a standalone operation, it can



(a) The node on a corner will not be moved



(b) The node on a ridge will only be moved along the same ridge



(c) The edge on a ridge will only be partitioned into new edges on the same ridge

Figure 5.4: Preserving corners and ridges of shapes during shifting

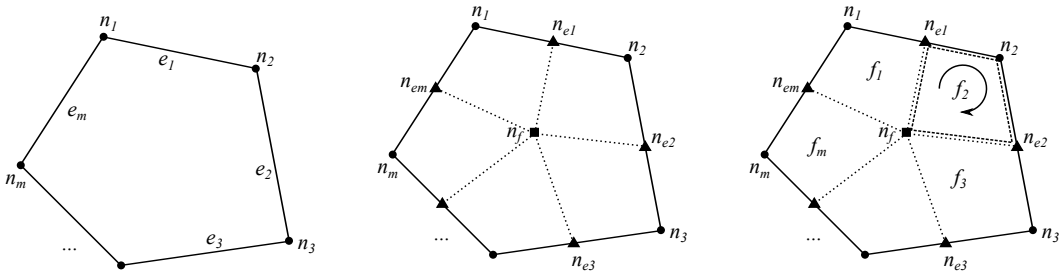
also be utilized as the first step of volume refinement, which is described in Section 5.3.2. In the following of this chapter, "surface refinement" refers to the standalone surface refinement operation only. Volume refinement is performed before surface refinement. Two parameters r_v and r_s are defined, typically, r_v is chosen as $1.5 \sim 2.0$ and r_s is $1.2 \sim 1.5$. If the size ratio between volume elements is larger than r_v, the volume mesh with the larger element size will be refined recursively until it is less than r_v. If the size ratio between surface elements is still larger than r_s, the surface mesh will be further refined until it is less than r_s. The sequence of the refinement is illustrated in Algorithm 5.1. Since the polyhedron elements based on the scaled boundary finite element method can have arbitrary number of edges and nodes, the modification is limited to the local region near the interface.

5.3.1 Polytree refinement in 2D

A polygon face in 2D is expressed as $f = (E, N)$, where $E = \{e_1, e_2, \cdots, e_m\}$ and $N = \{n_1, n_2, \cdots, n_m\}$. m is the number of nodes and edges of the face (see Fig. 5.5a).

A polygon face is partitioned by bisecting all of its edges. The mid-points of the edges $N_e = \{n_{e1}, n_{e2}, \cdots, n_{em}\}$ are inserted as new nodes, as shown in Fig. 5.5b. The edges are split into $E_p = \{(n_1, n_{e1}), (n_{e1}, n_2), (n_2, n_{e2}), \cdots, (n_{em}, n_1)\}$. The centroid of the polygon, n_f, is also inserted as a new node. The new edges inside the face are constructed by connecting n_f and the mid-points in N_e, $E_f = \{(n_{e1}, n_f), (n_{e2}, n_f), \cdots, (n_{em}, n_f)\}$. The new node list becomes $N = N \cup N_e \cup \{n_f\}$ and the new edge list is $E = E_p \cup E_f$.

The original polygon face of m edges is divided into m new polygons (quadrilaterals) by connecting one original node from N with two mid-points of the two adjacent edges from N_e and the center node n_f. For example, in Fig. 5.5c the nodes of a new face f_2 are collected following the clockwise order $\{n_{e1}, n_2, n_{e2}, n_f\}$. The m new faces $F = \{f_1, f_2, \cdots, f_m\}$ will replace the original face.



(a) Arbitrary star convex polygon in 2D

(b) Inserting mid-points of edges and centroid of face

(c) Construction of new faces

Figure 5.5: Polytree refinement of a polygon in 2D

5.3.2 Polytree refinement in 3D

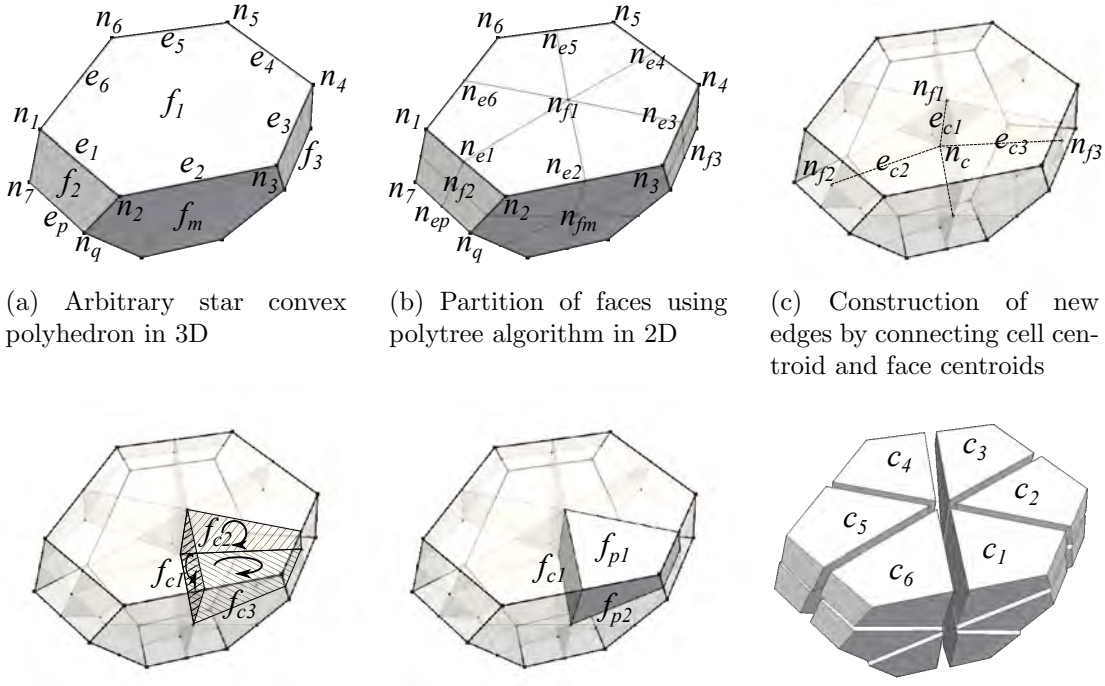
A polyhedron cell (see Fig. 5.6a) to be subdivided is expressed as $c = (F, E, N)$, where $F = \{f_1, f_2, \cdots, f_m\}$, $E = \{e_1, e_2, \cdots, e_p\}$ and $N = \{n_1, n_2, \cdots, n_q\}$. m is the number of faces, p is the number of edges and q is the number of nodes.

To partition a polyhedron, all the faces are partitioned first using the procedure in the previous section as illustrated in Fig. 5.6b. N_e and N_f denote the set of mid-points of the edges and the set of centroids of the faces, respectively. A new node n_c is inserted at the centroid of the polyhedron. The new node list is expressed as $N = N \cup N_e \cup N_f \cup \{n_c\}$.

New edges $E_c = \{(n_{f1}, n_c), (n_{f2}, n_c), \cdots, (n_{fm}, n_c)\}$ are formed by connecting the centroid of the polyhedron n_c to the centroids N_f of all the original faces in F, as shown in Fig. 5.6c. The new edge list is $E = E_p \cup E_f \cup E_c$.

New interior faces (quadrilaterals) F_c are constructed by connecting one mid-points from N_e with the centroids of two adjacent faces from N_f and the cell centroid n_c (see Fig. 5.6d). The new face list is $F = F_p \cup F_c$.

The original polyhedron of q nodes will be divided into q new polyhedron cells. For node n_i, a new polyhedron c_i is enclosed by all the new faces connected to the node and the interior faces connected to those faces, as shown in Fig. 5.6e. The q new cells $C = \{c_1, c_2, \cdots, c_q\}$ in Fig. 5.6f will replace the original cell c.



(a) Arbitrary star convex polyhedron in 3D

(b) Partition of faces using polytree algorithm in 2D

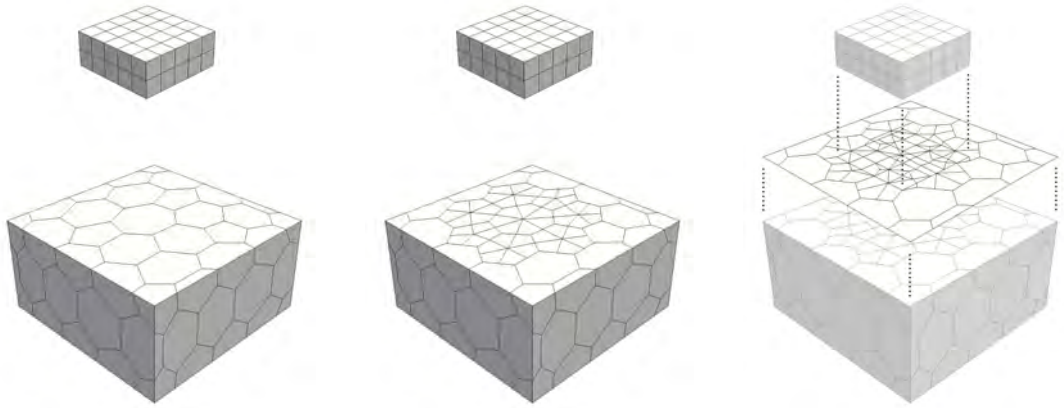
(c) Construction of new edges by connecting cell centroid and face centroids

(d) Construction of new faces inside the cell

(e) Construction of new cells by enclosing a set of new faces

(f) A set of new cells after polytree refinement

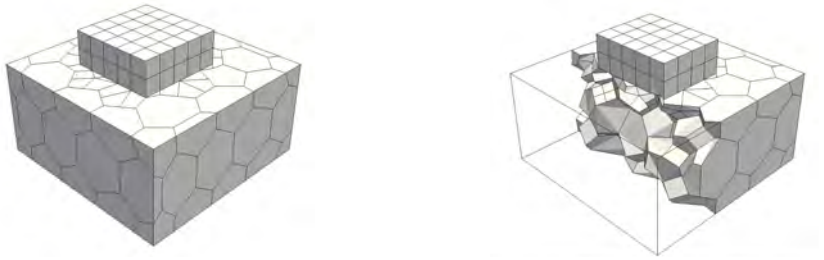
Figure 5.6: Polytree refinement of a polyhedron in 3D



(a) Two cuboids meshed with different element types and sizes

(b) Polytree refinement on the lower part

(c) Construction of matching interface



(d) Merged volume mesh

(e) Interior view of the merged volume mesh

Figure 5.7: Merging of non-matching meshes of two cuboids use polytree refinement

The coupling procedure using polytree refinement is illustrated with an example in Fig. 5.7. The upper cuboid is discretized with hexahedron elements and the lower cuboid with polyhedron elements (Fig. 5.7a). The size ratio between the two meshes is 1.7. The thresholds in this example are $r_v = 1.5$ and $r_s = 1.2$. The volume mesh of the lower part connected to the interface is refined (Fig. 5.7b). The size ratio becomes 1.1 after the refinement, so both the volume and surface refinement stop. A matching interface is then constructed leading to matching meshes with good quality (Fig. 5.7c).

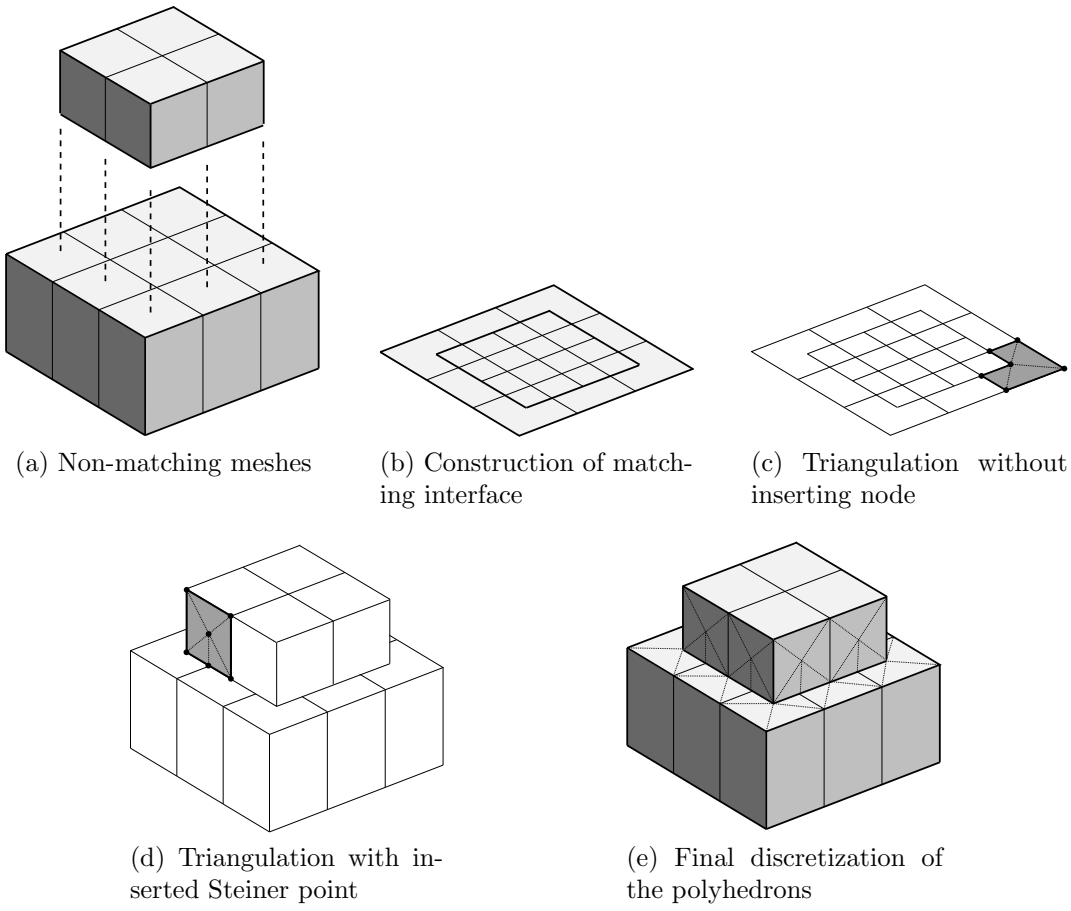
5.4 Face discretization of polyhedron cells

In the presented approach, the polygon faces of a polyhedron cell are discretized with triangular and quadrilateral elements if the polygons have more than 4 edges. Degenerated quadrilateral faces will also be triangulated. The measurement of triangle qualities and triangulation schemes are already stated in Section 4.6. In this section only two examples are presented to illustrate the two triangulation schemes. An example of first scheme is shown is Fig. 5.8c. The polygon in shaded area is divided into 4 triangles by connecting the existing nodes. The second scheme is illustrated by the shaded polygon in Fig. 5.8d. The new inserted node is connected to all the edges of the polygon to form new triangles. In this case, the triangles have better quality than those obtained by direct triangulation without inserting the Steiner point. After the discretization of all the faces of the polyhedron cells, the polyhedron mesh (Fig. 5.8e) can be used in the scaled boundary finite element analysis as formulated in Section 3.

The complete procedure of coupling non-matching meshes is shown in Algorithm 5.1.

5.5 Numerical examples

In this section, seven numerical examples are presented to verify the proposed method. The first example is a pure bending patch test of a cube modeled with two non-matching meshes. It is shown that the proposed method produces results with machine accuracy. The second example is a cantilever beam subject to an end-shear load. The mesh is refined and the convergence rate is examined. The third example is a mechanical handle with a ball joint. The non-matching meshes of three parts are merged. The fourth example is a fist holding a stick. The mesh of the stick is rotated to produce different non-matching interfaces and the results are consistent. The fifth example is an arch dam with a foundation. The different meshes of the foundation (polyhedron, tetrahedron and hexahedron elements) are merged with mesh of the dam. The next



(a) Non-matching meshes

(b) Construction of matching interface

(c) Triangulation without inserting node

(d) Triangulation with inserted Steiner point

(e) Final discretization of the polyhedrons

Figure 5.8: Face discretization of polyhedron cells

Algorithm 5.1 Coupling of non-matching meshes

```
Input meshes
Define parameters: ε_d, ε_θ, ε_n, ε_e, r_v, r_s
Calculate mesh size ratio r
while r >= r_v do
    Polytree volume refinement
    Update mesh size ratio r
end while
while r >= r_s do
    Polytree surface refinement
    Update mesh size ratio r
end while
Extract interfaces (based on ε_d and ε_θ)
while not converged do
    Merge adjacent nodes (based on ε_n)
    Partition edges close to nodes (based on ε_e)
end while
Insert non-matching nodes and edges
Assemble matching surface mesh
Discretize boundary of polyheron cells
Output matching meshes
```

example is a mechanical part composed of three components. The meshes are refined on the two interfaces using polytree algorithm in 3D. The last example is a propeller blade attached to a shaft. The surface mesh of the shaft is refined using polytree algorithm in 2D.

The relative error norm in displacement is calculated using Eq. (4.9). Similarly, the relative error norm in stress is calculated using Eq. (5.1).

$$e_s = \frac{\|\boldsymbol{\sigma}_{num} - \boldsymbol{\sigma}_{ref}\|}{\|\boldsymbol{\sigma}_{ref}\|}, \tag{5.1}$$

where $\boldsymbol{u}_{num}$ and $\boldsymbol{\sigma}_{num}$ are the numerical results calculated by the proposed method. When available, an analytical solution is used as the reference solution. When an analytical solution is not available, the numerical result obtained from a convergence study using the finite element method is selected as the reference solution.

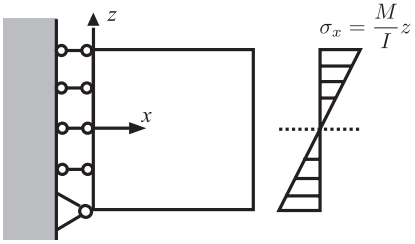


Figure 5.9: The boundary conditions of pure bending patch test

5.5.1 Patch test

This example is a cube subject to a pure bending load. The side view of the cube and the boundary conditions are shown in Fig. 5.9. The dimension of the cube is $10 \times 10 \times 10$m ($h = 10$m). Young's modules and Poisson's ratio are $E = 1 \times 10^6$Pa and $\nu = 0.3$. On one surface of the cube a linearly distributed load is applied. On the opposite surface the displacements perpendicular to that surface are prescribed as zero. The boundary conditions are shown in Fig. 5.9.

The only non-zero stress component of pure bending is given by Eq. (5.2) as

$$\sigma_x = \frac{M}{I}z, \tag{5.2}$$

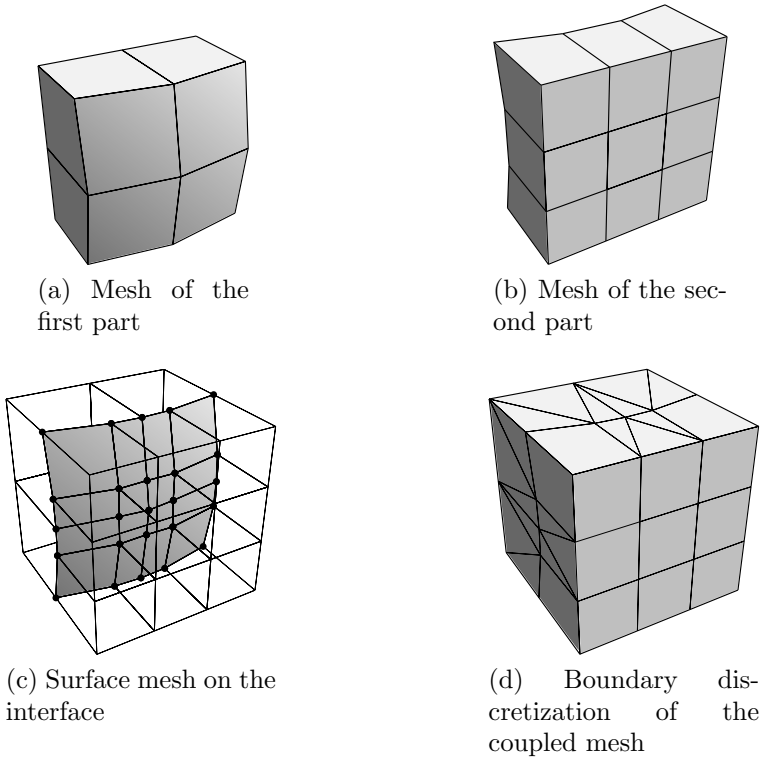
where $M = 6\times10^5\text{N}\cdot\text{m}$ is the bending moment applied on the surface and $I = 833.33\text{m}^4$ is the second moment of area. The displacement field can be calculated by

$$u_x = \frac{M}{EI}xz, \tag{5.3a}$$

$$u_y = -\nu\frac{M}{EI}\left(yz - \frac{h^2}{4}\right), \tag{5.3b}$$

$$u_z = -\frac{M}{2EI}\left(x^2 + \nu\left(z^2 - y^2\right)\right). \tag{5.3c}$$

The cube is divided into two parts by a curved surface. The two parts are meshed independently using 4 and 9 hexahedron elements (Fig. 5.10a and Fig. 5.10b). On the interface, the ratio between the element sizes of the meshes of the two parts is



(a) Mesh of the first part

(b) Mesh of the second part

(c) Surface mesh on the interface

(d) Boundary discretization of the coupled mesh

Figure 5.10: Coupling of non-matching meshes of the two parts of a cube connected by a curved interface

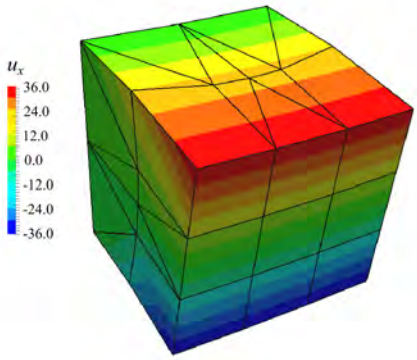
Table 5.1: Error of SBFEM results compared with analytical solution

Relative error norm in displacement	1.8256e-12
Relative error norm in stress	1.4747e-13

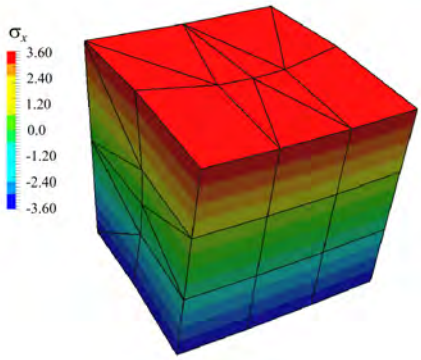
approximately 2:3 (Fig. 5.10c). Only the surface mesh needs to be triangulated. Second order triangle and quadrilateral elements are used in the analysis.

The results in displacement and stress of the proposed method are shown in Fig. 5.11. The errors of numerical results compared to the analytical solution are calculated using Eq. (4.9) and Eq. (5.1). Machine accuracy is obtained in the solution as shown in Table 5.1.

FEM analysis is performed using ABAQUS for comparison. The two non-matching meshes are connected using tie constraint (Fig. 5.12). The geometry and boundary conditions are the same as those in Fig. 5.9. The element type is C3D20. The errors are shown in Table 5.2. It is shown that ABAQUS fails to pass the patch test.

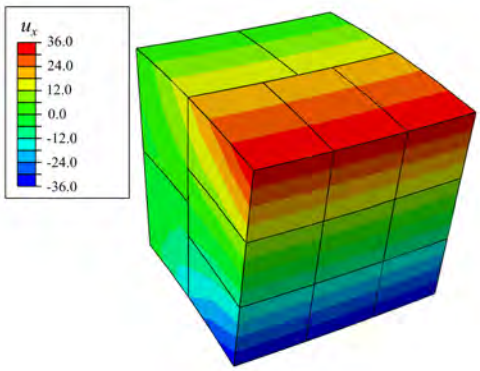


(a) The displacement u_x of non-matching meshes. Unit: mm

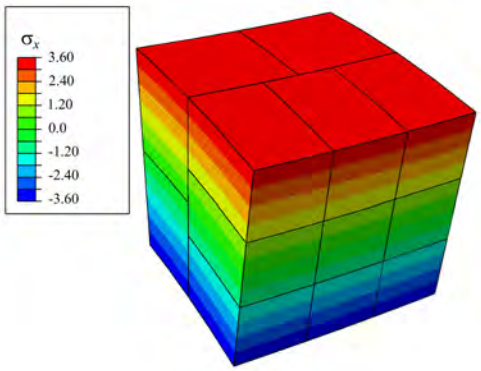


(b) The stress σ_x of non-matching meshes. Unit: kPa

Figure 5.11: The displacement and stress distribution of the pure bending patch test in the SBFEM



(a) The displacement u_x of non-matching meshes. Unit: mm



(b) The stress σ_x of non-matching meshes. Unit: kPa

Figure 5.12: The displacement and stress distribution of the pure bending patch test in the FEM

Table 5.2: Error of FEM results compared with analytical solution

Relative error norm in displacement	9.1545e-3
Relative error norm in stress	1.7633e-2

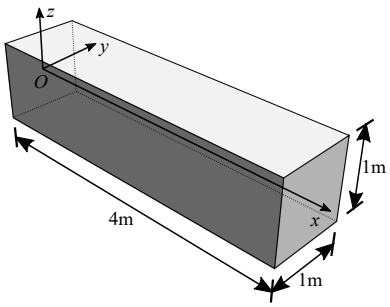


Figure 5.13: Geometry of the cantilever beam

5.5.2 A cantilever beam subject to an end-shear load

A cantilever beam subject to an end-shear force is shown in Fig. 5.13. The total length of the beam is $L = 4\text{m}$, the width is $b = 1\text{m}$ and the height is $h = 1\text{m}$. The Young's modulus of the beam is $E = 1 \times 10^9\text{Pa}$, and Poisson's ratio is $\nu = 0$. The shear modulus is $G = 5 \times 10^8\text{Pa}$. The origin of the coordinate system is located at the center of the section at one end of the beam. The beam is fixed in weak form on the plane $x = 0$ by applying the analytical solution of displacement as the boundary condition. At the free end of the beam where $x = 4\text{m}$, parabolic distributed shear force is applied. The total magnitude of the shear force is $P = 1.67 \times 10^3\text{N}$.

The stress field is described by quadratic shear stress and linear normal stress (Liu et al., 2008)

$$\sigma_x = \frac{P\,(L-x)\,z}{I}, \tag{5.4a}$$

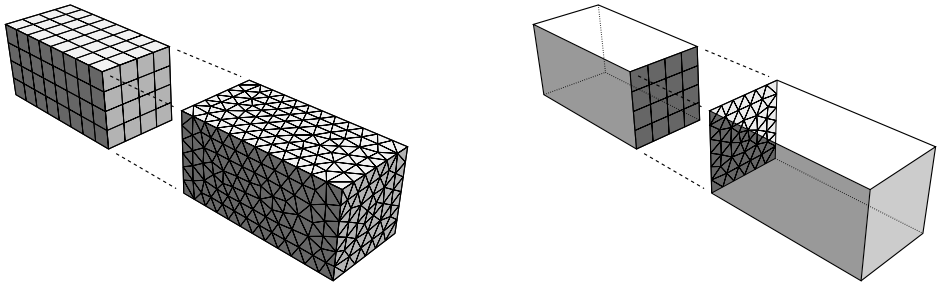
$$\sigma_y = 0, \tag{5.4b}$$

$$\sigma_z = 0, \tag{5.4c}$$

$$\tau_{yz} = 0, \tag{5.4d}$$

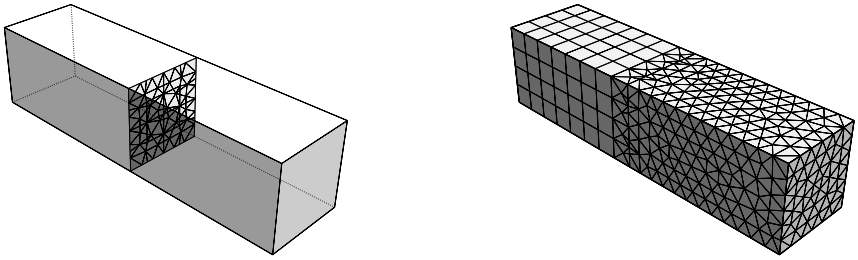
$$\tau_{xz} = -\frac{P}{2I}\left(\left(\frac{h}{2}\right)^2 - z^2\right), \tag{5.4e}$$

$$\tau_{xy} = 0. \tag{5.4f}$$



(a) Non-matching meshes for the two beam segments

(b) The interface meshes are extracted



(c) Merged interface

(d) Merged volume mesh

Figure 5.14: Coupling of meshes of two beam segments

where I is the second moment of area of the beam. At point $(0, -0.5\text{m}, -0.5\text{m})$, $u_x = u_y = u_z = 0$. Rigid body translation and rotation are constrained. Considering the constitutive equation and the boundary conditions, analytical solution in displacement can be derived by integration

$$u_x = \frac{P}{EI}\left(Lx - \frac{x^2}{2}\right)z - \frac{P}{6GI}\left(\left(\frac{h}{2}\right)^2 z - z^3\right), \tag{5.5a}$$

$$u_y = 0, \tag{5.5b}$$

$$u_z = -\frac{P}{EI}\left(\frac{L}{2}x^2 - \frac{x^3}{6}\right) - \frac{P}{3GI}\left(\frac{h}{2}\right)^2 x. \tag{5.5c}$$

The beam is divided into two segments, the dimensions of both are $2 \times 1 \times 1$m, as shown in Fig. 5.14a. The first segment is discretized with hexahedron elements and the second with tetrahedron elements. The interface meshes are extracted (Fig. 5.14b) and merged (Fig. 5.14c). The merged surface mesh is assembled to the volume meshes to replace the original faces.

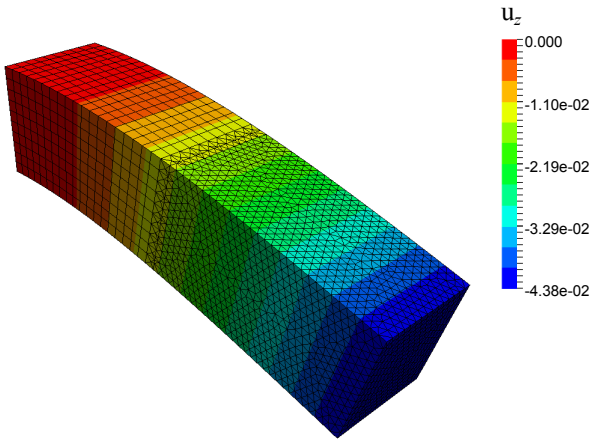


Figure 5.15: The deformation of the beam. Unit: mm. The deformation is scaled by 2×10^6.

The errors of numerical results compared to the analytical solution are calculated using Eq. (4.9). A high-order patch test is performed with third order elements. Solution of machine accuracy is obtained.

Convergence study is performed using linear and quadratic elements. The meshes of the two segments are refined respectively, while 2:3 mesh size ratio is enforced. The theoretical error norm of finite elements is in Eq. (4.12).The relative error norm is plotted versus DOF in log-log scale. The convergence rates are calculated by fitting the curves with power functions. The theoretical convergence rates are achieved for both linear and quadratic elements.

5.5.3 A handle with a ball joint

In this example a mechanical handle consisting of three components is considered. The geometries of the three components are a box, a sphere and a cylinder. The sphere is carved out of the box and the cylinder by Boolean operations, leaving two concave sphere interfaces. The dimension of the box is $400 \times 400 \times 150$mm. The diameter of the sphere is 200mm. The diameter and length of the cylinder are 100mm and 300mm respectively. The three parts are meshed separately using hexahedron elements, as

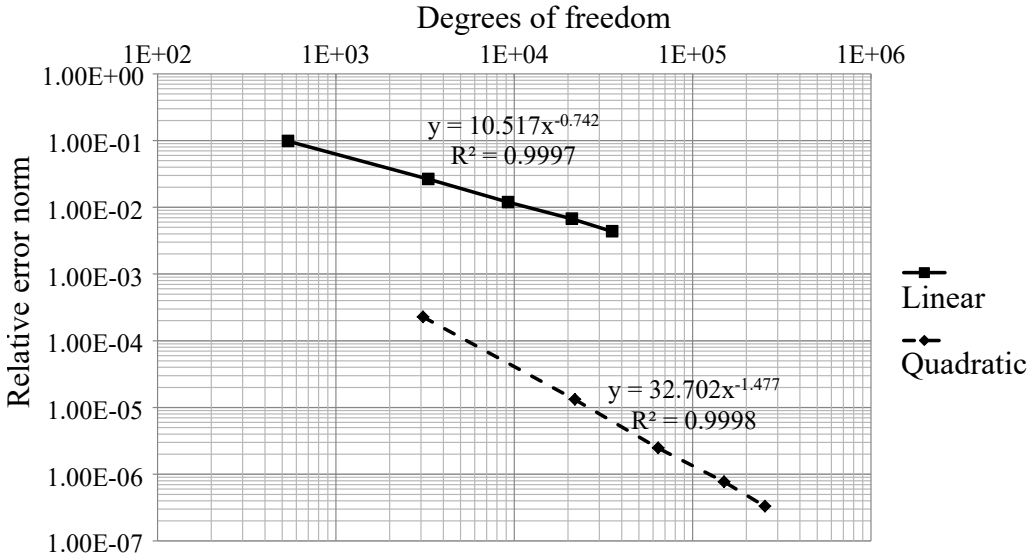
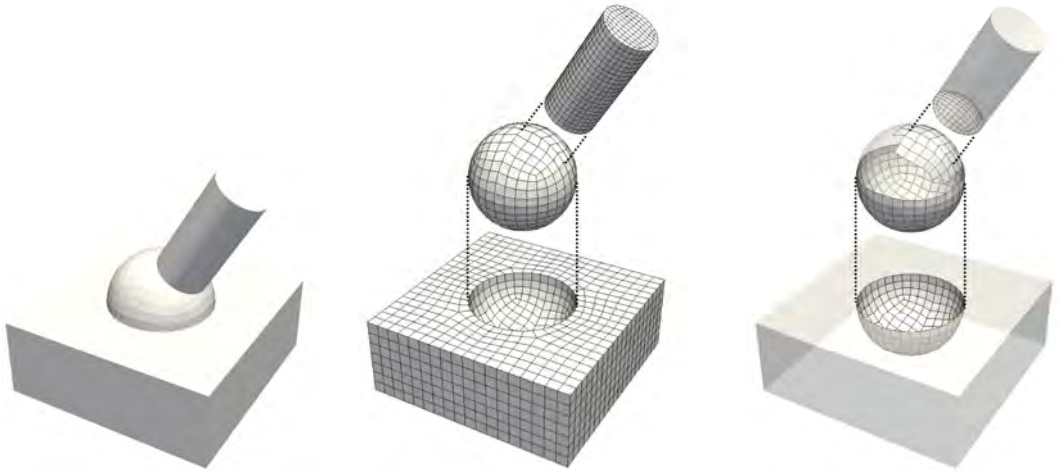


Figure 5.16: Convergence behavior of the cantilever beam in terms of error norm in displacement

shown in Fig. 5.17b. The mesh sizes of the box and sphere are both 20mm. The mesh size of cylinder is 10mm. The three meshes are merged into one mesh in Fig. 5.17e.

The model is fixed on the bottom of the box. A uniform pressure of the magnitude $p = 1 \times 10^6$Pa is applied to the top surface of the cylinder. The direction of the pressure is perpendicular to the surface and towards the center of sphere, as shown in Fig. 5.18a. The Young's modulus of the material is $E = 2.1 \times 10^{11}$Pa, and Poisson's ratio is $\nu = 0.3$. The deformation is plotted in Fig. 5.18b.

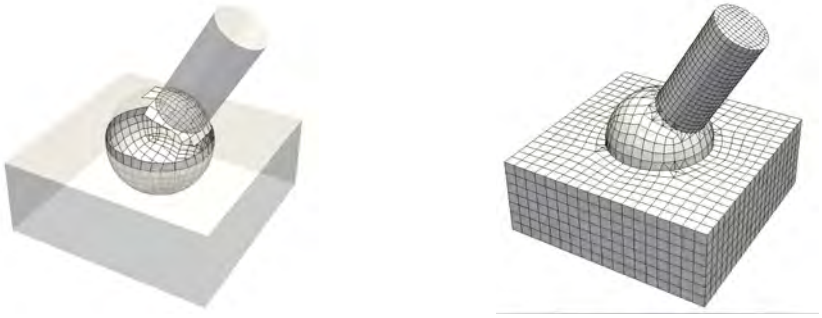
A convergence study is performed using a group of meshes of different sizes. The meshes of the three parts are refined at the same time while maintaining the size ratio of the elements. A specific point A (Fig. 5.18a) is selected to compare the results. The reference solution is obtained using ABAQUS. A convergence study using a series of tetrahedron meshes of second order finite elements (C3D10) is performed. The relative error of the displacement in z direction at point A is plotted versus the number of degrees of freedom in log-log scale in Fig. 5.19.



(a) The geometry of the handle

(b) Non-matching meshes of the handle

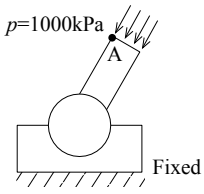
(c) Interface meshes extracted



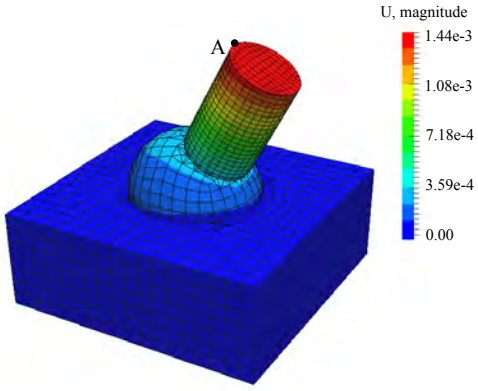
(d) Merged interface mesh

(e) Merged volume mesh

Figure 5.17: The geometry and mesh of the handle



(a) The boundary conditions of the handle



(b) The deformation of the handle. Unit: mm.

Figure 5.18: The boundary conditions and deformation of handle

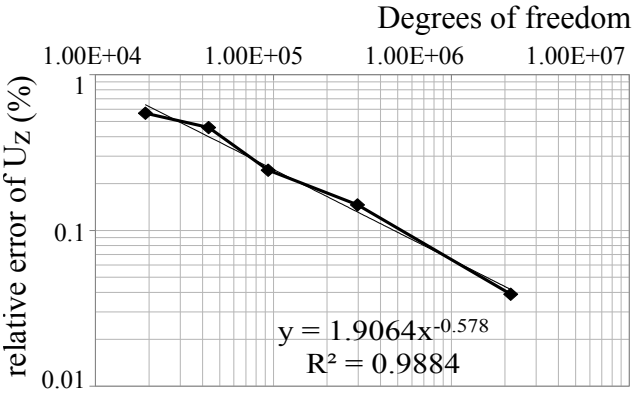


Figure 5.19: The convergence curve of displacement in z direction at point A of the handle

5.5.4 A fist holding a stick

In this example, a sculpture of a human fist holding a rigid stick is considered. The model of the fist is given in STL format (`https://www.thingiverse.com/thing:12629`). The STL model contains a large number of degenerated triangles (Fig. 5.20), which makes it difficult to generate a standard finite element mesh. A polyhedron mesh with relative high quality can be generated from the STL model directly without human intervention using the algorithm present in Liu et al. (2017). In addition to the STL boundary, a cylinder passing through the model is considered as an additional boundary. This cylindrical hole will be filled by the stick meshed with hexahedron elements. The polyhedron mesh of the fist and the hexahedron mesh of the stick are coupled automatically using the present approach.

To test the robustness of the present approach, different interfaces are generated by rotating the stick around its central axis as shown in Fig. 5.21a. Four different rotation angles from 0° to 11.25° are considered. The rotated meshes of the stick are merged with the mesh of the fist (Fig. 5.21b).

The material properties of the fist are $E = 1 \times 10^5$Pa and $\nu = 0.3$. The stick is considered as rigid by choosing $E = 1 \times 10^{10}$Pa and $\nu = 0$. The boundary conditions are shown in Fig. 5.22a. A uniform pressure along the direction of the stick is applied

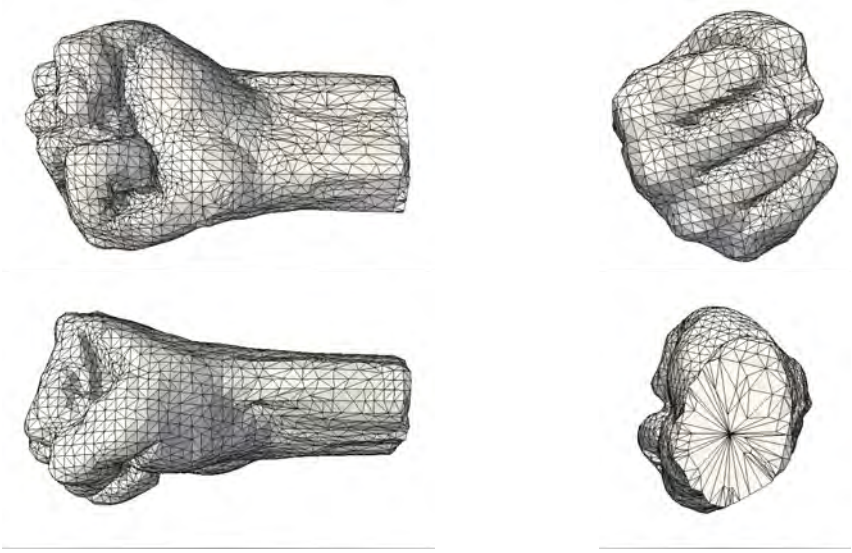
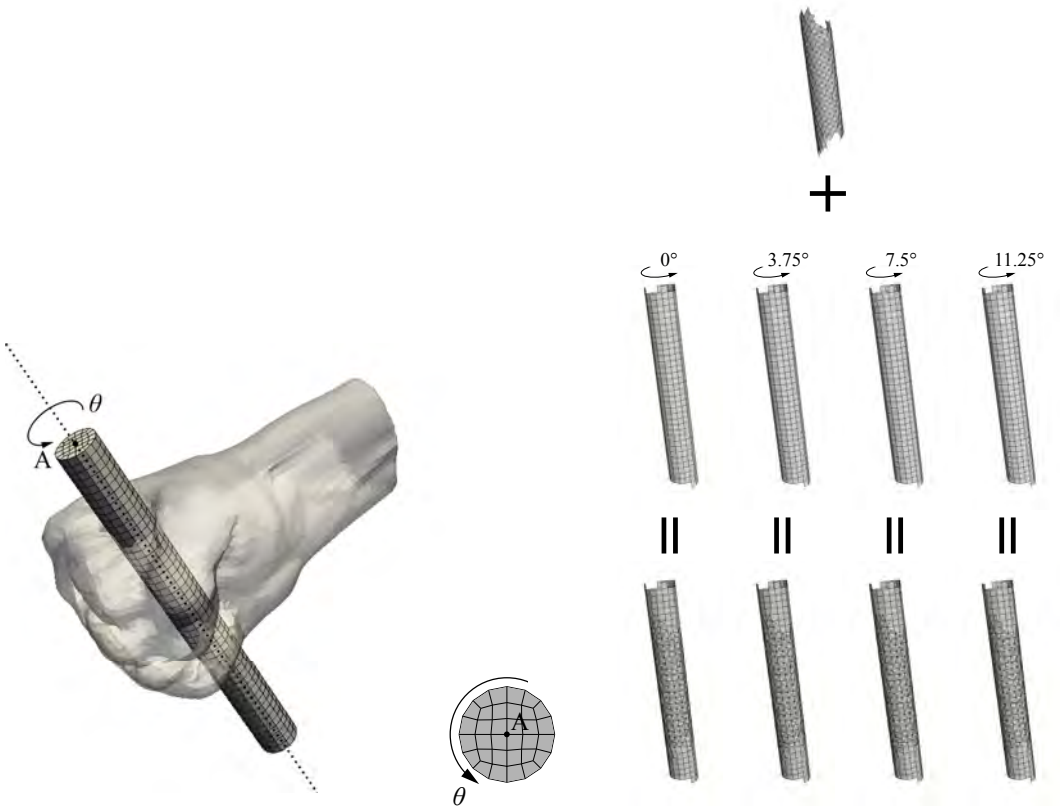


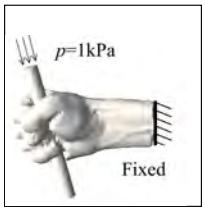
Figure 5.20: The STL model of the fist



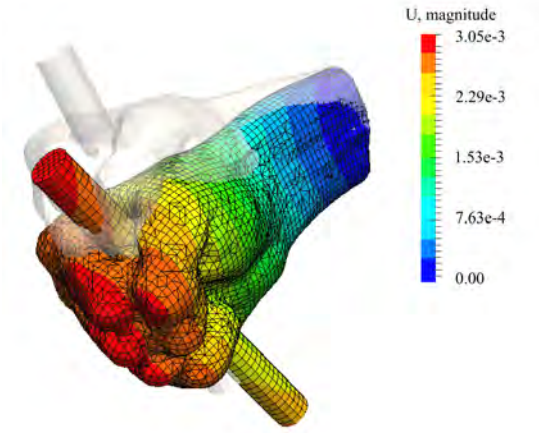
(a) Rotation of the stick

(b) Different interfaces extracted from rotated mesh

Figure 5.21: Rotation of mesh of the stick



(a) The boundary conditions of the fist and stick



(b) The deformation of the fist and stick. Unit: mm.

Figure 5.22: The boundary conditions and deformation of the fist and stick

Table 5.3: Comparison of results from different meshes of the stick

Rotation angle θ (degree)	0	3.75	7.5	11.25
Number of nodes	19135	19158	19142	19144
Displacement u_z of point A (mm)	-1.25620	-1.25790	-1.25814	-1.25842
Relative difference	-	0.14%	0.15%	0.18%

on the top surface. The plane near the wrist of the fist is fixed in all directions. The deformation of the model is plotted in Fig. 5.22b. The vertical displacements at the central point A of the top surface of the stick (Fig. 5.21a) is shown in Table 5.3 for the four meshes at different rotation angles. The differences caused by the rotation are negligible.

5.5.5 An arch dam

To demonstrate the capability of the proposed method in engineering practice, an arch dam with a foundation is considered in this example (Fig. 5.23). A hexahedron mesh is generated for the dam. Three meshes are generated for the foundation using polyhedron, tetrahedron and hexahedron elements, respectively. Each of the three meshes of the foundation is merged with the mesh of the dam as shown in Fig. 5.24 with the

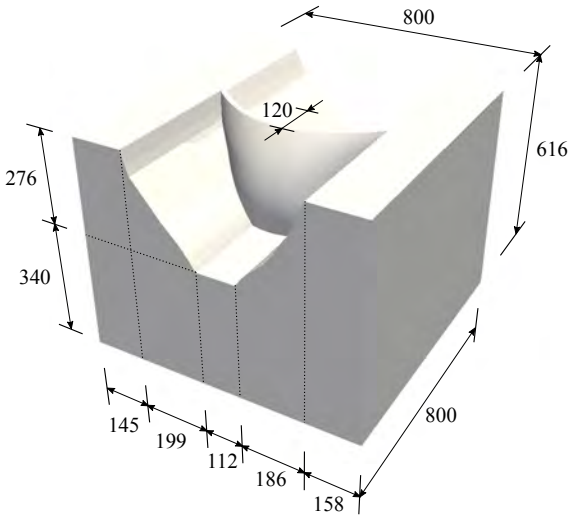


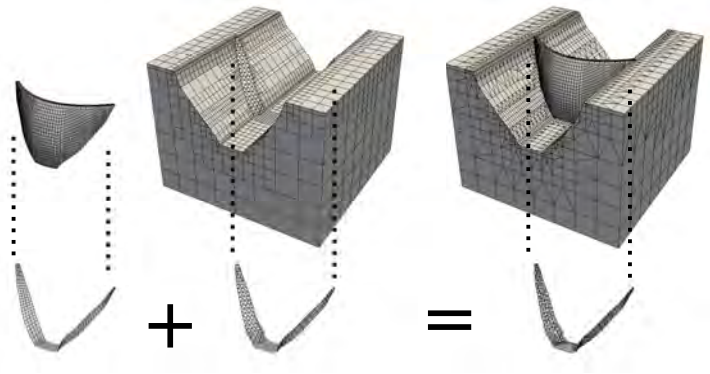
Figure 5.23: Geometry of the arch dam. Unit: m

corresponding interface mesh.

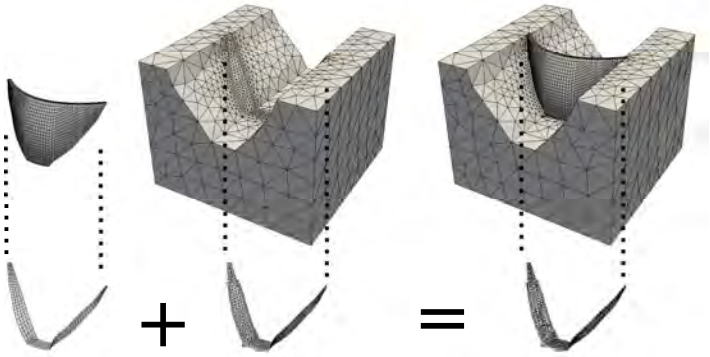
The material of the dam is concrete. The Young's modulus is $E = 17 \times 10^9$Pa and Poisson's ratio $\nu = 0.3$. The mass density of the concrete is $\rho = 2.4 \times 10^3$kg/m^3. The self weight of the foundation is not considered. At the vertical boundaries and the bottom of the foundation, the displacements perpendicular to the surfaces are constrained. The hydro-static pressure $p = \rho_w g h$ is applied on the surface of the dam, where $\rho_w = 1 \times 10^3$kg/m^3 is the mass density of water, $g = 9.81$N/kg and h is the vertical distance of a point to the water surface.

A perspective view and the top view of the deformation of the dam and foundation obtained with the polyhedron mesh in Fig. 5.24a is shown in Fig. 5.25. To compare the results, a point A is selected at the middle of the top surface of the dam. The horizontal displacements along the downstream direction calculated from different meshes are shown in Table 5.4. It is observed that relative difference between polyhedron mesh and hexahedron mesh is small. The difference between polyhedron mesh and tetrahedron mesh is larger because of the fewer number of nodes of the tetrahedron mesh.

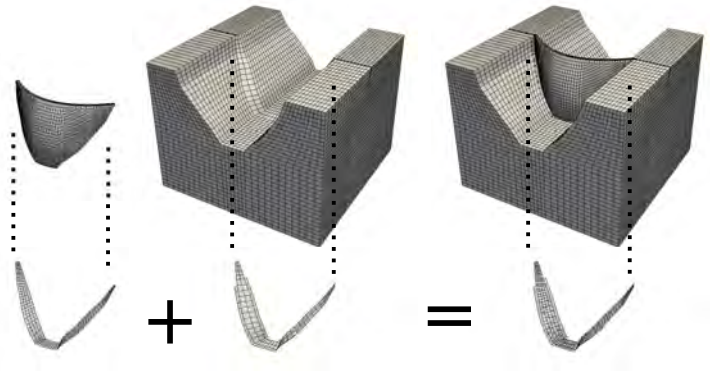
It is worth mentioning that the polyhedron mesh of the foundation is easy to generate using the octree algorithm and provides a smooth transition of element size. However,



(a) Polyhedron mesh for the foundation

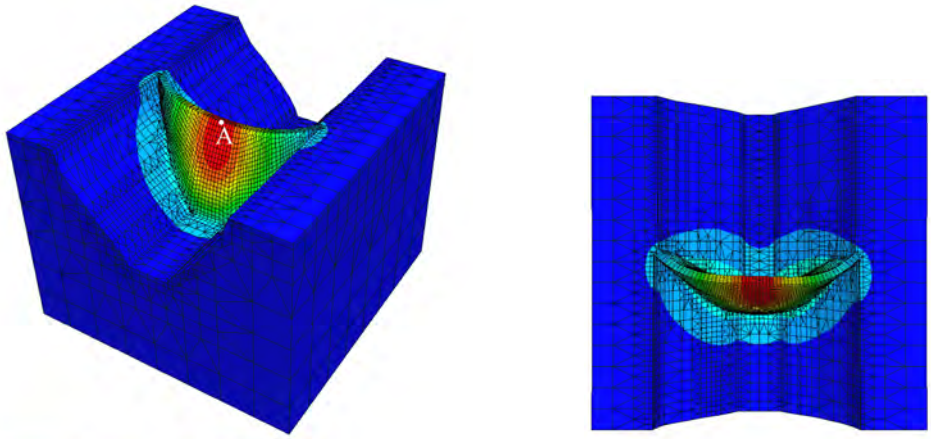


(b) Tetrahedron mesh for the foundation



(c) Hexahedron mesh for the foundation

Figure 5.24: Coupling of non-matching meshes of the arch dam and foundation. Different meshes are generated for the foundation.



(a) Perspective view (b) Top view

Figure 5.25: The deformation of the dam and foundation obtained with polyhedron mesh. Maximum displacement occurs in the middle of dam. The deformation is scaled by 1000. Unit: mm

Table 5.4: Comparison of results from different meshes of the arch dam

Element type of the foundation	polyhedron	tetrahedron	hexahedron
Number of elements	8871	15005	36100
Number of nodes	13743	7970	41349
Displacement u_z (mm)	52.424	51.771	52.146
Relative difference	-	1.36%	0.53%

the other two meshes requires tedious manual operations, such as partition, merging and repairing, which makes the mesh generation process time consuming. The element sizes also need to be controlled carefully to successfully generate the mesh. Therefore, the proposed coupling method and octree algorithm provide a powerful approach in automated analysis of models with complex geometry.

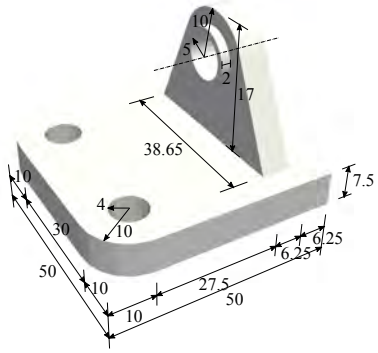
5.5.6 A mechanical part

A mechanical part is shown in Fig. 5.26a. The geometry consists of three components: a base plate with two cylindrical holes, a vertical plate with a hole and a ring. The three parts are meshed independently (Fig. 5.26b). Polyhedron meshes are generated for the base plate and the vertical plate using octree algorithm. Structured hexahedron mesh is generated for the ring. The sizes of the three meshes are different on the two interfaces. On the first interface, where the ring and the vertical plate are connected, the mesh of the plate is refined using polytree algorithm. Similarly, on the interface between the vertical plate and the base plate, the mesh of the base plate is refined, as shown in Fig. 5.26c. After refinement, the three meshes are connected (Fig. 5.26d). It can be observed that near the two interfaces the sizes of volume elements are similar (Fig. 5.26f), which contributes to maintaining the mesh quality after merging.

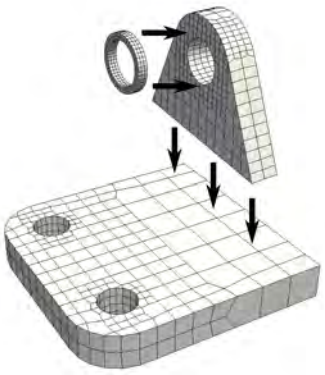
The material of the mechanical part is steel. Young's modulus is $E = 2.1 \times 10^{11}$Pa and Poisson's ratio $\nu = 0.3$. The two holes on the base plate are fixed. Uniform surface traction $t = 1 \times 10^{6}$Pa is applied on the inner surface of the cylindrical hole in the vertical plate (See Fig. 5.27a). The deformation is shown in Fig. 5.27b.

5.5.7 A propeller blade attached to a shaft

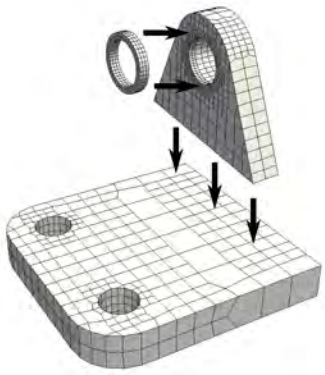
The STL model of a propeller is considered (`https://www.thingiverse.com/thing:2137615`). Only one blade and the shaft are modeled. The dimension of the propeller is shown in Fig. 5.28b and Fig. 5.28c. Hexahedron meshes are generated for the two



(a) The geometry of the mechanical part

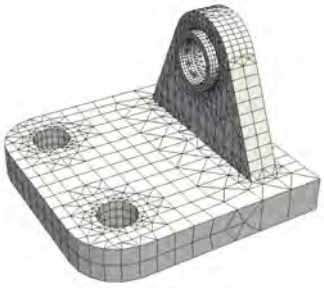


(b) Non-matching meshes of the three components

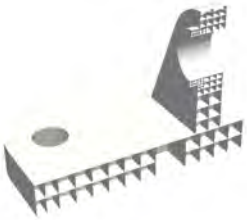


(c) The meshes are refined on the two interfaces

(d) The interface meshes are extracted

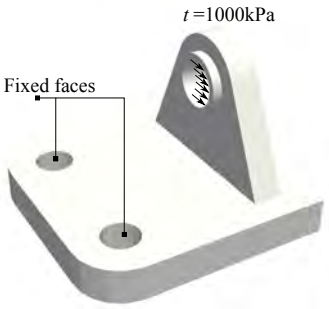


(e) The merged volume mesh

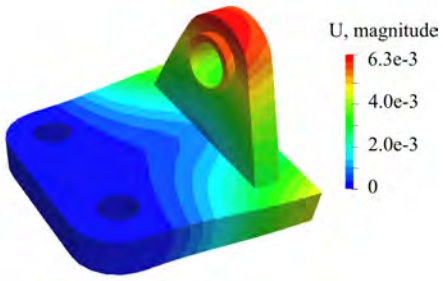


(f) Clip view of the merged mesh

Figure 5.26: The geometry and mesh of the mechanical part



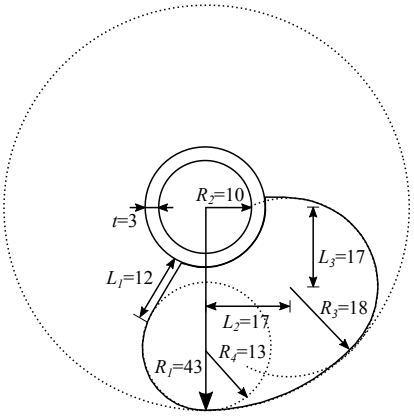
(a) The boundary conditions of the mechanical part



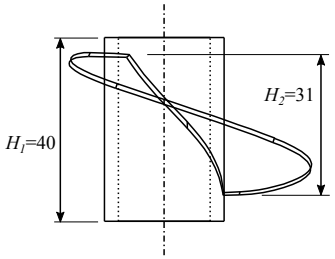
(b) The deformation of the mechanical part. Unit: mm

Figure 5.27: The boundary conditions and deformation of the mechanical part

(a) The blade and shaft of the propeller



(b) Top view of the propeller. Unit: cm.

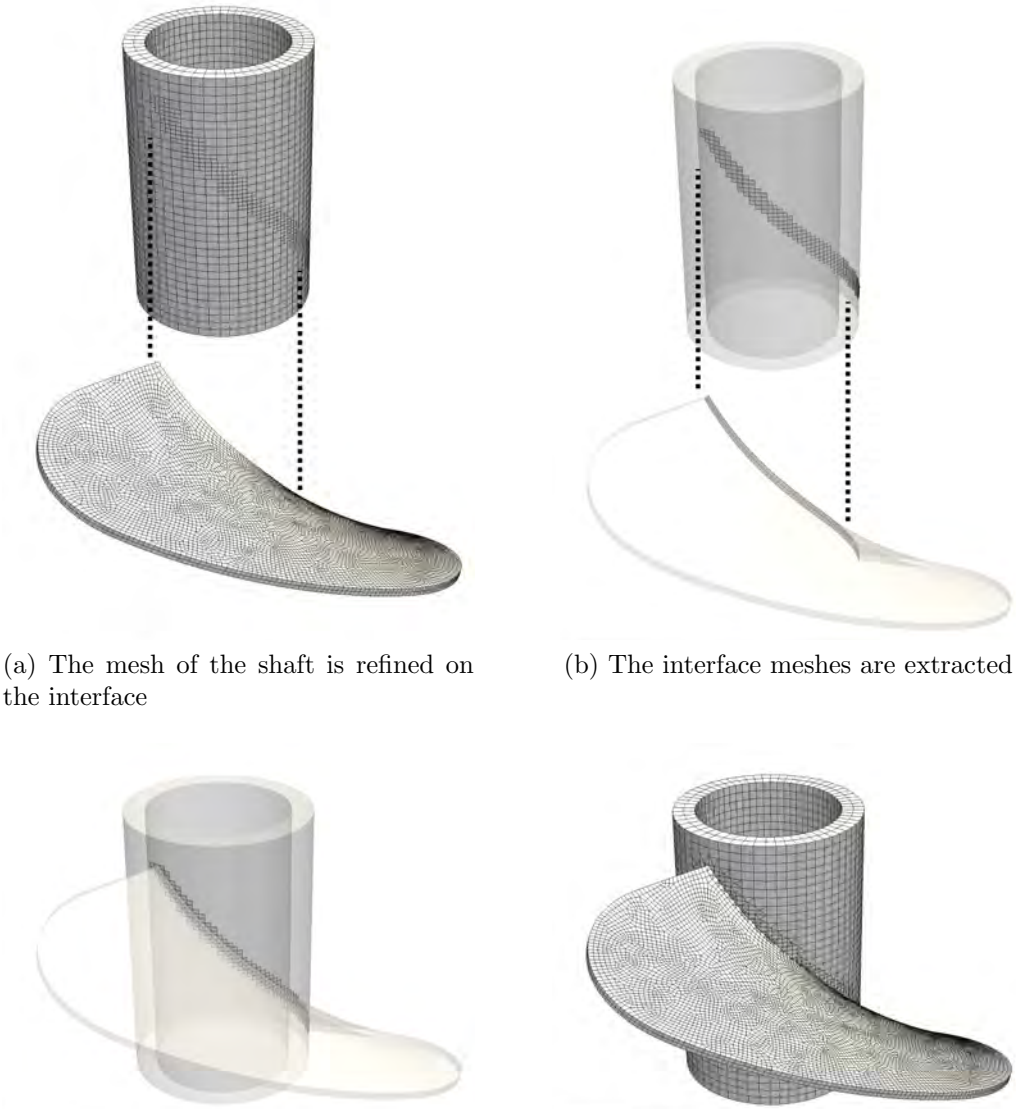


(c) Side view of the propeller. Unit: cm.

Figure 5.28: The geometry of the propeller

parts. The surface mesh of the shaft is refined on the interface, as shown in Fig. 5.29a.

The bottom of the shaft is fixed. Young's modulus is $E = 2.1 \times 10^{11}$Pa and Poisson's ratio is $\nu = 0.3$. A uniform downward surface traction $t = 1 \times 10^6$Pa is applied on the top surface of the blade (Fig. 5.30a). The deformation is shown in Fig. 5.30b.



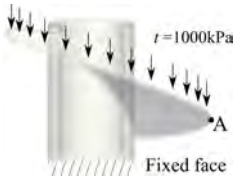
(a) The mesh of the shaft is refined on the interface

(b) The interface meshes are extracted

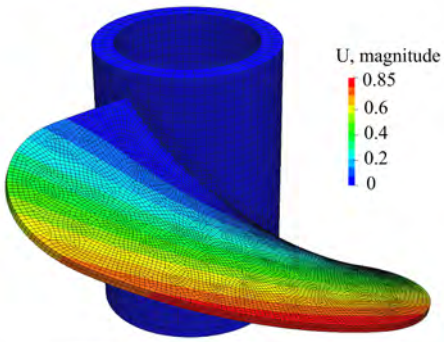
(c) The merged interface mesh

(d) The merged volume mesh

Figure 5.29: The geometry and mesh of the propeller



(a) The boundary conditions of the propeller



(b) The deformation of the propeller. Unit: mm

Figure 5.30: The boundary conditions and deformation of the propeller

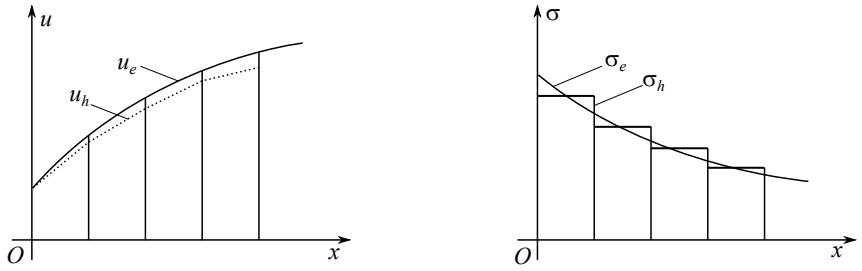
Chapter 6

Adaptive mesh refinement

Adaptive analysis is an important technique in the finite element analysis. The error of an analysis is estimated using a posterior indicator. The mesh in a region with high estimated error is refined therefore the discretization error is expected to reduce in the region. As a result, the mesh is only refined where necessary and the computational resource can be used efficiently. The convergence rate of an adaptive refinement is usually higher than the uniform refinement. In this chapter, an adaptive mesh refinement method is proposed. A simple error indicator is implemented together with a refinement technique based on polytree algorithm. Numerical examples are provided to validate the proposed method.

6.1 Error indicator

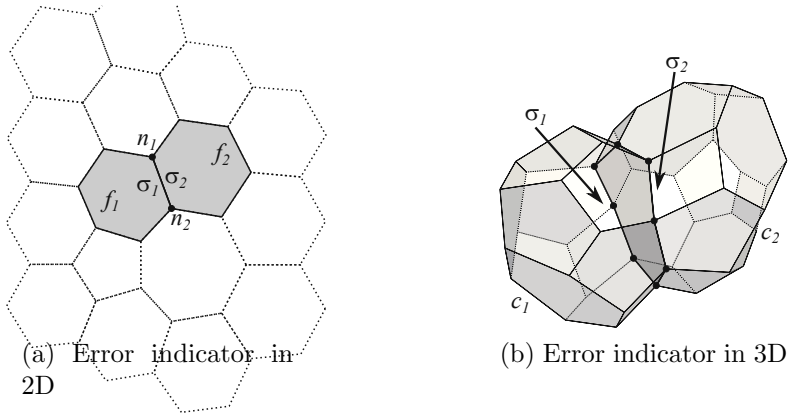
A large amount of error indicators have been developed in the finite element analysis. In this research, a simple approach is followed using the discontinuity of the stress field. If linear element is utilized in the finite element analysis, a piecewise linear displacement field u_h is calculated to approximate the exact solution u_e (Fig. 6.1a). The numerical solution of stress σ_h is represented by the derivative of u_h, which is a step function as shown in Fig. 6.1b. The stress at an internal node can be calculated from the



(a) Actual displacement and numerical solution

(b) Actual stress and numerical solution

Figure 6.1: Discontinuity in the stress field in 1D



(a) Error indicator in 2D

(b) Error indicator in 3D

Figure 6.2: Error indicator based on stress discontinuity

two elements connected to the node. Those two values are usually different, and the difference can be used as an error indicator.

In the scaled boundary finite element analysis, polygon elements in 2D are constructed from 1D line elements. The stress of a line element can be evaluated from the two polygons on both sides. In Fig. 6.2a, the line element $n_1 n_2$ is shared by two polygon faces f_1 and f_2. The results are usually different as the displacement field only satisfies C^o continuity on the boundary. Therefore, the difference between the stresses can be used as an error indicator. In 3D the stress on a face can be calculated from the two polyhedrons containing the face. In Fig. 6.2b, the polyhedrons c_1 and c_2 share the same polygon face (it will be triangulated before the analysis). The stresses calculated from the two polyhedrons are compared and the difference is used as the error indicator.

The stress components on the faces can be calculated following Eq. 3.133 in Sec-

tion 3. The von Mises stress σ_v is calculated as

$$\sigma_v = \sqrt{\frac{(\sigma_1 - \sigma_2)^2 + (\sigma_2 - \sigma_3)^2 + (\sigma_3 - \sigma_1)^2}{2}}, \tag{6.1}$$

where σ_1, σ_2 and σ_3 are the principle stresses. The average von Mises stress at a face i is

$$\sigma_{vi} = \frac{\sigma_{vi1} + \sigma_{vi2}}{2}, \tag{6.2}$$

where σ_{vi1} and σ_{vi2} are the stresses calculated from the two cells connected to the face. The absolute value of the difference between the stresses is

$$\Delta\sigma_{vi} = |\sigma_{vi1} - \sigma_{vi2}|. \tag{6.3}$$

The maximum von Mises stress of all the faces in the whole mesh is found as

$$\sigma_{max} = \max(\sigma_{vi}) \qquad \text{for } i = 1, 2, \ldots, n, \tag{6.4}$$

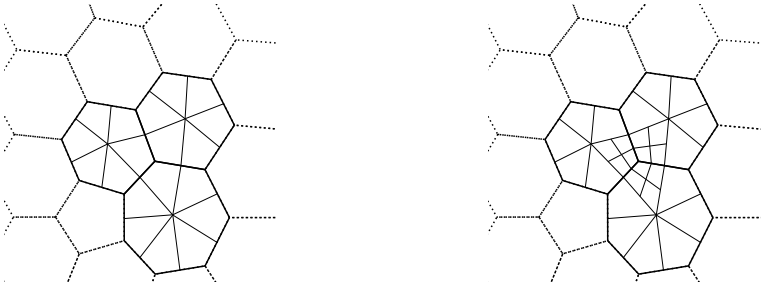
where n is the total number of faces in the mesh. The local relative error at face i is estimated as

$$e_i = \frac{\Delta\sigma_{vi}}{\sigma_{max}} \times 100\%. \tag{6.5}$$

The faces on the boundary are not considered in this process as they are connected to one cell only. A tolerance e_{tol} is defined. The typical range of the value is $1\% \sim 5\%$. All the faces satisfying

$$e_i \geq e_{tol}, \tag{6.6}$$

will be selected. All the cells containing these faces will be refined in the next iteration.



(a) First refinement (b) Second refinement

Figure 6.3: Polytree mesh refinement in 2D

6.2 Mesh refinement technique

A complete adaptive remeshing includes refinement and coarsening of the mesh. In this research, only refinement is implemented as coarsening is usually less frequently required, especially when the initial mesh size is reasonably coarse.

A polytree algorithm is employed for mesh refinement. The details of the implementation can be found in Section 5.3. A polygon with n nodes will be divided into n new polygons. The new polygons are always quadrilaterals in 2D because the nodes are connected by two edges in each polygon. In 3D the number of new polyhedrons is equal to the number of nodes in the original polyhedron. The shape of the new polyhedrons depend on the number of faces connected to each node. If a node is shared by 3 faces in the polyhedron, the obtained polyhedron around that node will be a hexahedron. If a node is shared by 4 faces in the polyhedron, the obtained polyhedron around that node will be an octahedron. In all the cases the faces of the new polyhedrons are always quadrilaterals.

All the faces and cells are assigned with a level number. In the initial mesh, all the faces and cells are on level 0. In the refinement iterations, each time a cell or a face is refined, the level number is increased by 1. The mesh is balanced after the refinement (maximum level difference between adjacent cells can not exceed 1). In other words, the difference between the level of a cell and the maximum level of its faces can not exceed

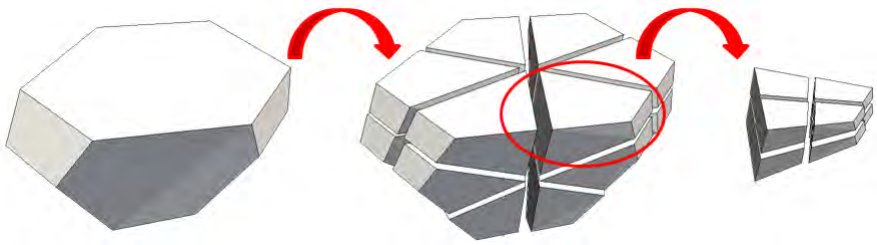
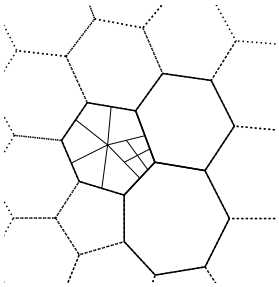
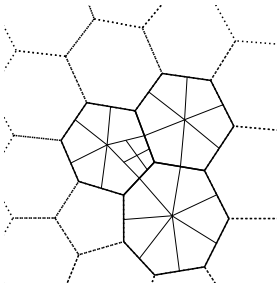


Figure 6.4: Polytree mesh refinement in 3D



(a) Unbalanced polytree mesh



(b) Balanced polytree mesh

Figure 6.5: Balance of polygon mesh after polytree refinement

1. The idea of is similar to the balancing in octree mesh presented in Section 4.3.4. In Fig. 6.5a, the quadrilaterals on level 2 are adjacent with 2 polygons on level 0. Those polygons contain edges on level 0, 1 and 2. Therefore they are refined in Fig. 6.5b.

The complete procedure of mesh refinement in 3D is shown in the flow chart in Fig. 6.6. An initial polyhedron mesh is imported at the first step. The surface meshes of the polyhedrons are triangulated. A copy of the initial polyhedron mesh is still stored in the program for later use. These two meshes share the same cell index. The analysis of the cells is performed using their triangulated surface meshes and the errors are estimated based on the error indicator in Section 6.1. If all the faces satisfy the requirement in Eq. (6.6), the program stops. Otherwise, a list containing the index of cells which need to be refined is generated. A polytree refinement is performed on the polyhedron mesh (before triangulation) based on this list. The refined polyhedron mesh is used as input for the next iteration. This process is repeated until the requirement

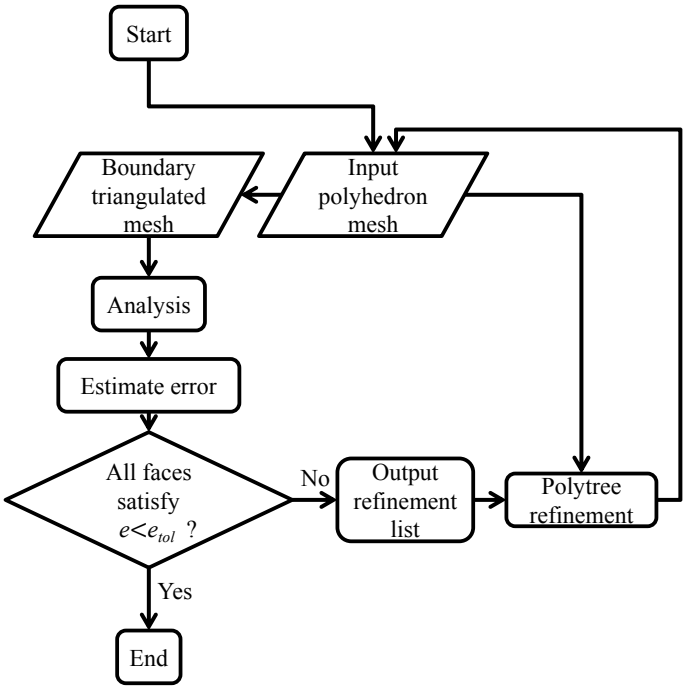


Figure 6.6: Adaptive analysis procedure

in Eq. (6.6) is satisfied on all of the faces or a predefined maximum iteration number is reached.

6.3 Numerical examples

In this section, five numerical examples are presented to validate the adaptive refinement method. The convergence behavior is examined.

6.3.1 A pressurized hollow sphere

A pressurized hollow sphere is shown in Fig. 6.7. Only an octant of the sphere is modeled due to symmetry. The radius of the inner surface is $a = 20$m while the outer surface $b = 50$m. A uniform normal pressure $P = 1$Pa is applied on the inner surface of the sphere. At the surfaces at $x = 0$, $y = 0$ and $z = 0$, the displacements perpendicular to surfaces are constrained. The Young's modulus is $E = 1000$Pa and Poisson's ratio $\nu = 0.3$.

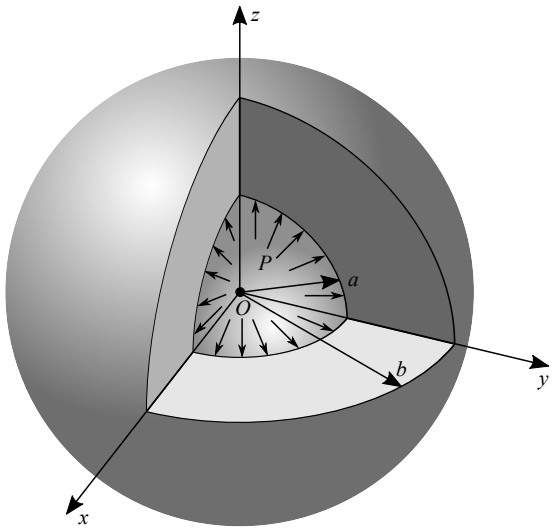


Figure 6.7: Geometry and boundary conditions of the hollow sphere

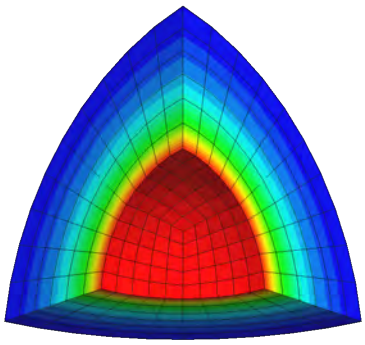
The refinement criteria is $e_{tol} = 1\%$. In the first two iterations almost all the cells are refined. From the third iteration, only the region near the inner sphere where the stress is higher than the outer sphere is refined. The contour of displacement on different meshes is shown in Fig. 6.8.

The actual error of this problem can be calculated by comparing with analytical solution Eq. (4.10). The convergence curve is plotted in Fig. 6.9. It is observed that the result converges faster than the convergence rate from a uniform refinement in Section 4.

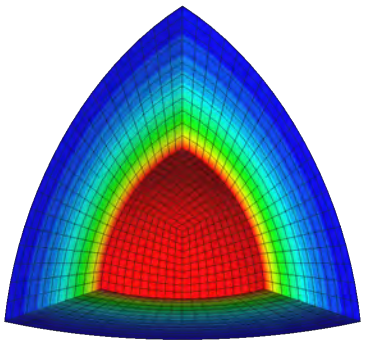
6.3.2 A short cantilever beam

In this example, a short cantilever beam is considered. A similar problem in 2D can be found in Song et al. (2018). The dimension of the beam is $20\text{m} \times 10\text{m} \times 10\text{m}$ (Fig. 6.10a). The beam is fixed at one end where $x = 0$. A uniform pressure $P = 1\text{Pa}$ is applied on the top surface of the beam as shown in Fig. 6.10b. The Young's modulus is $E = 1000\text{Pa}$ and Poisson's ratio $\nu = 0.3$.

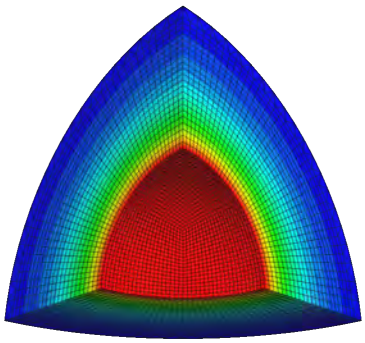
The initial mesh is a uniform hexahedron grid. The refinement criteria is $e_{tol} = 1\%$. The von Mises stress is plotted on each of the meshes in Fig. 6.11. In the first two iterations almost all the cells are refined. From the third iteration, only the region



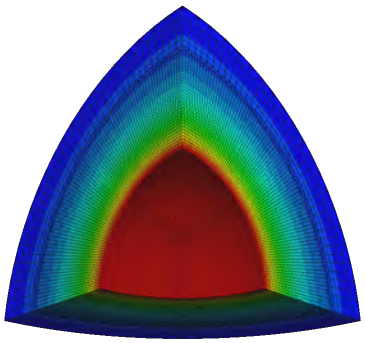
(a) Initial mesh



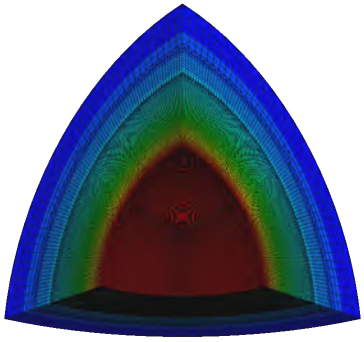
(b) First refinement



(c) Second refinement



(d) Third refinement



(e) Fourth refinement

Figure 6.8: Displacement of the hollow sphere after adaptive refinement

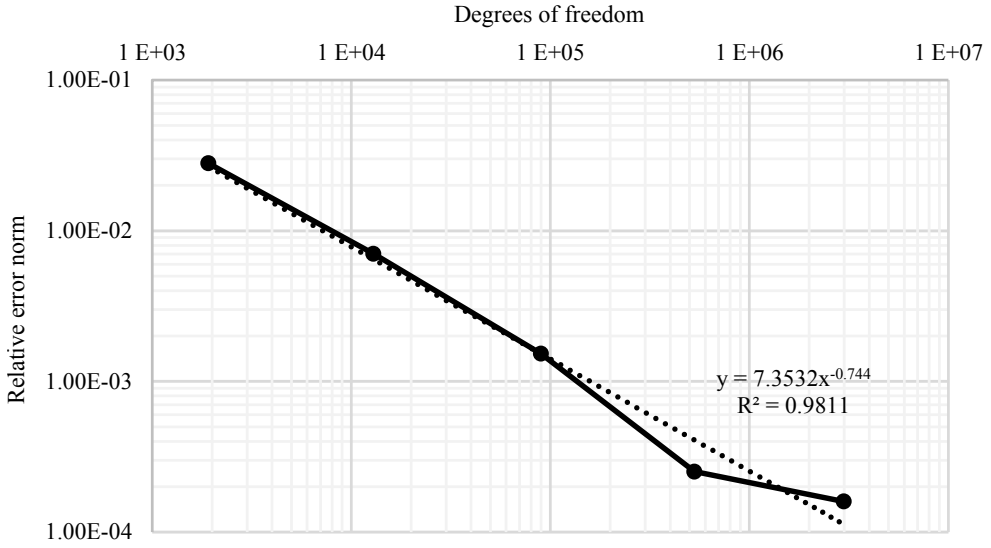
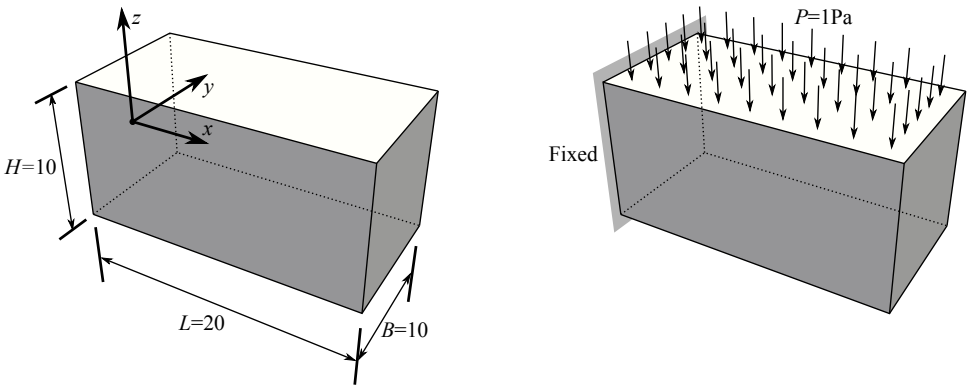


Figure 6.9: Convergence behavior of the hollow sphere in terms of error norm in displacement



(a) Geometry of the beam. Unit: m

(b) Boundary conditions of the beam

Figure 6.10: Geometry and boundary conditions of the beam

near the fixed end is refined. It agrees with the expectation because of the high stress concentrations near the fixed end.

6.3.3 A hook

In this example, a hook is considered. A similar problem in 2D can be found in Chau et al. (2018). The geometry of the hook is shown in Fig. 6.12a. The hook is fixed around the sphere on the top as shown in Fig. 6.12b. A uniform downward pressure $P = 1000$Pa is applied at the lower part of the hook. The Young's modulus is $E = 2.1 \times 10^{11}$Pa and Poisson's ratio $\nu = 0.3$.

The initial mesh composes of three parts, the two spherical shapes on the two ends and the segment in the middle. Structured hexahedron meshes are generated for those parts individually and merged together. The refinement criteria is $e_{tol} = 3\%$. The von Mises stress is plotted on each of the meshes in Fig. 6.13. It is observed that only the region near the middle of the curve is refined.

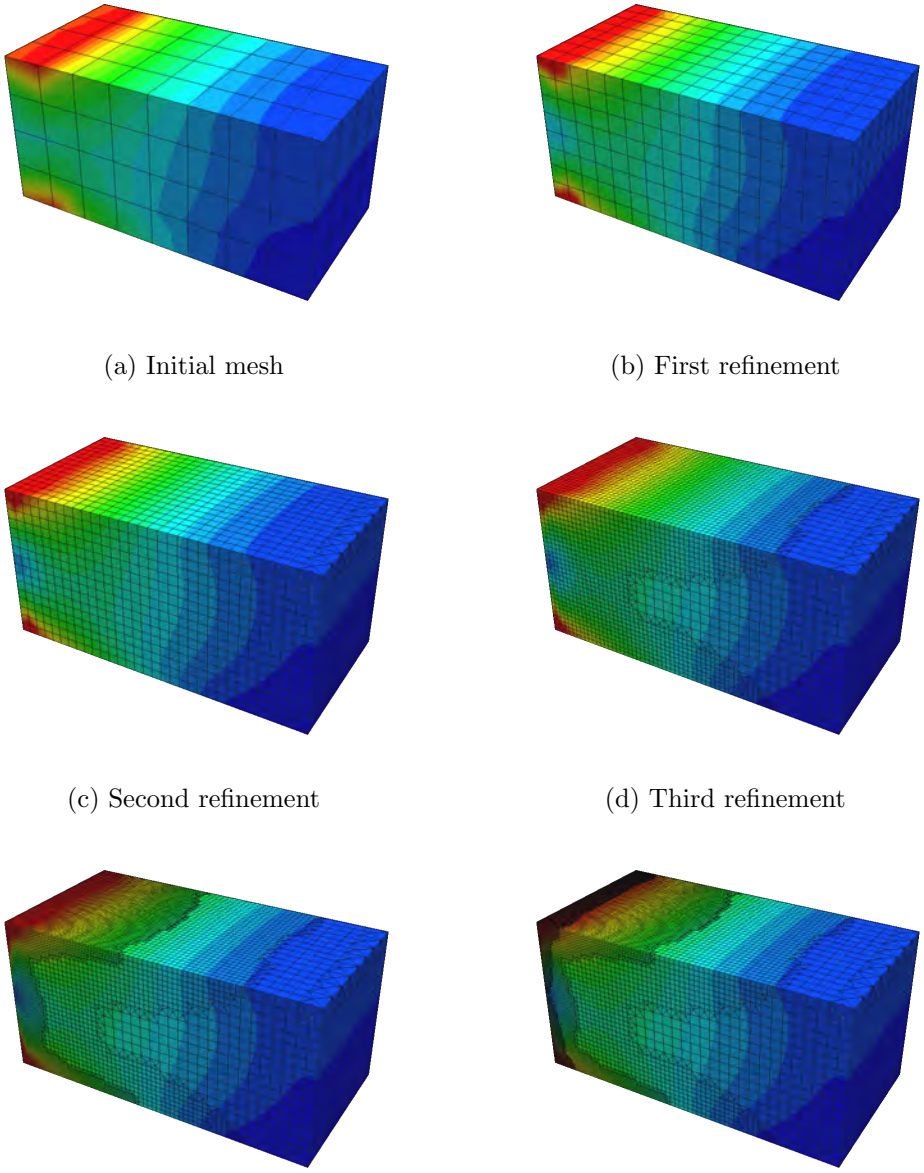
A detailed view of the mesh after the third refinement step is shown in Fig. 6.14.

6.3.4 A connecting rod

A connecting rod is shown in Fig. 6.15a. A similar problem can be found in Nguyen et al. (2014). The rod is fixed at the left end as shown in Fig. 6.12. A uniform downward pressure $P = 1000$Pa is applied at the inner surface of the ring at the right end. The Young's modulus is $E = 2.1 \times 10^{11}$Pa and Poisson's ratio $\nu = 0.3$.

The initial mesh composes several parts of structured hexahedron meshes. The refinement criteria is $e_{tol} = 5\%$. The von Mises stress is plotted on each of the meshes in Fig. 6.16. It is observed that the region in the middle of the rod, where large bending moment is located, is refined. The joint between the rod and the larger ring on the left is also refined.

A plane view of the mesh after the third refinement step is shown in Fig. 6.17.



(a) Initial mesh

(b) First refinement

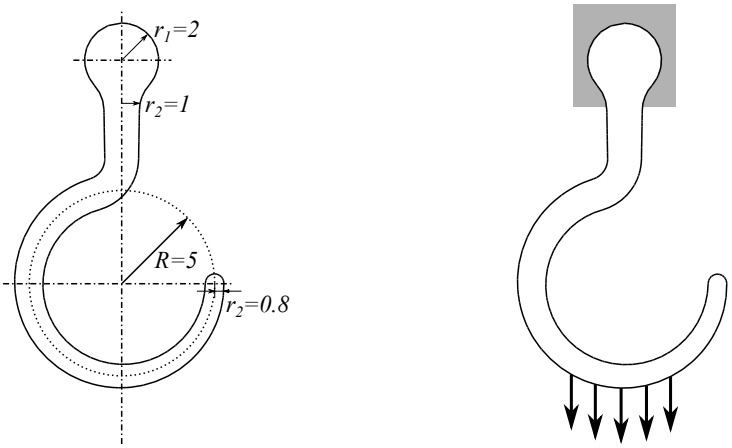
(c) Second refinement

(d) Third refinement

(e) Fourth mesh

(f) Fifth refinement

Figure 6.11: Von Mises stress of the beam after mesh refinement



(a) Geometry of the hook. Unit: mm

(b) Boundary conditions of the hook

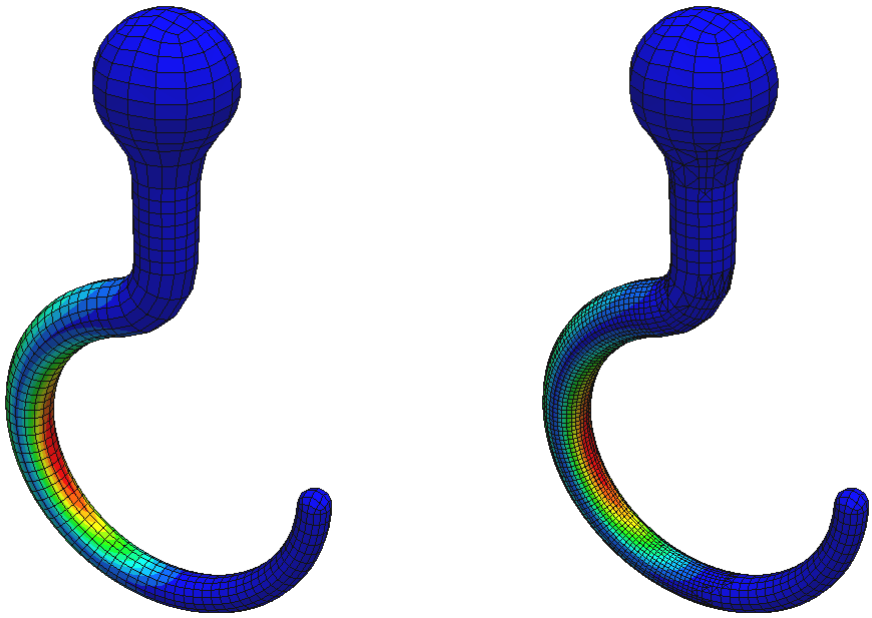
Figure 6.12: Geometry and boundary conditions of the hook

6.3.5 A sculpture

In this example, a sculpture of a lion is considered. The geometry of the model is given in STL format and quite complex (`https://www.thingiverse.com/thing:1393345`). A polyhedron mesh is generated using octree based algorithm. The bottom of the model is fixed, as shown in Fig. 6.18. Only the self-weight of the sculpture is considered. The material is granite. The Young's modulus is $E = 7 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.3$. The density of the material is $\rho = 2.8 \times 10^{3}$kg/m^3.

The von Mises stress is plotted on the initial model (Fig. 6.19). It is observed that on the supporting legs the stress is high.

The refinement tolerance is $e_{tol} = 5\%$. The mesh of the sculpture is refined twice as shown in Fig. 6.20.



(a) Initial mesh

(b) First refinement

(c) Second refinement

(d) Third refinement

Figure 6.13: Von Mises stress of the hook with each refinement step

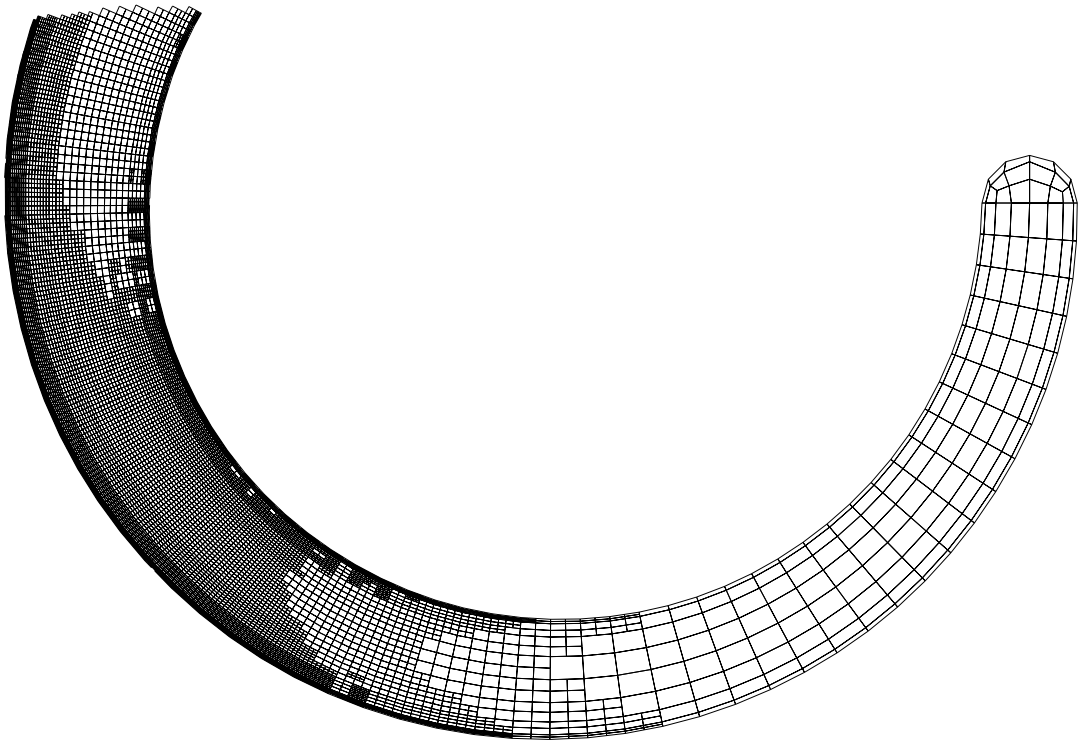
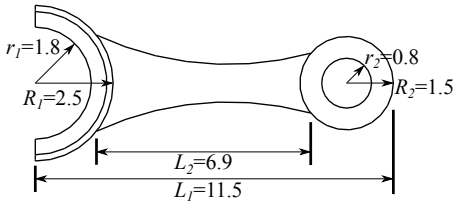
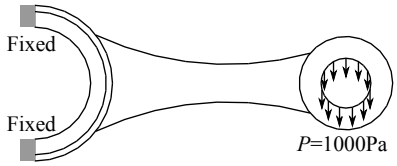


Figure 6.14: Detailed view of the mesh of the hook after the third refinement

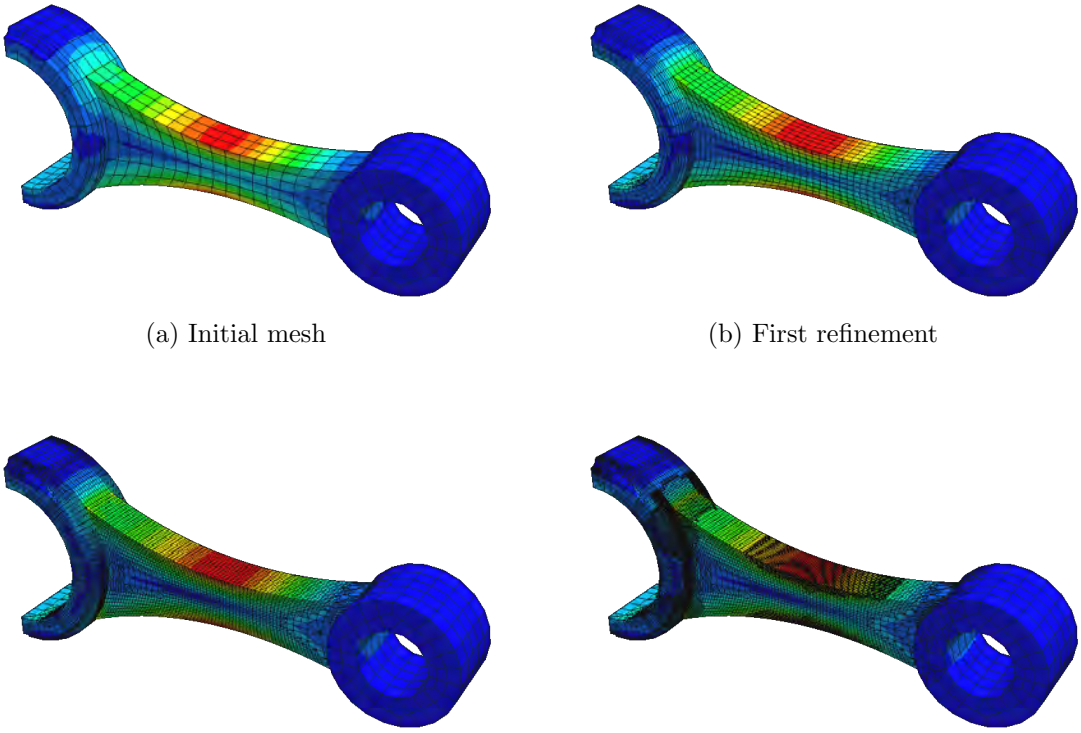


(a) Geometry of the connecting rod



(b) Boundary conditions of the connecting rod

Figure 6.15: Geometry and boundary conditions of the connecting rod



(a) Initial mesh

(b) First refinement

(c) Second refinement

(d) Third refinement

Figure 6.16: Von Mises stress of the connecting rod after each refinement step

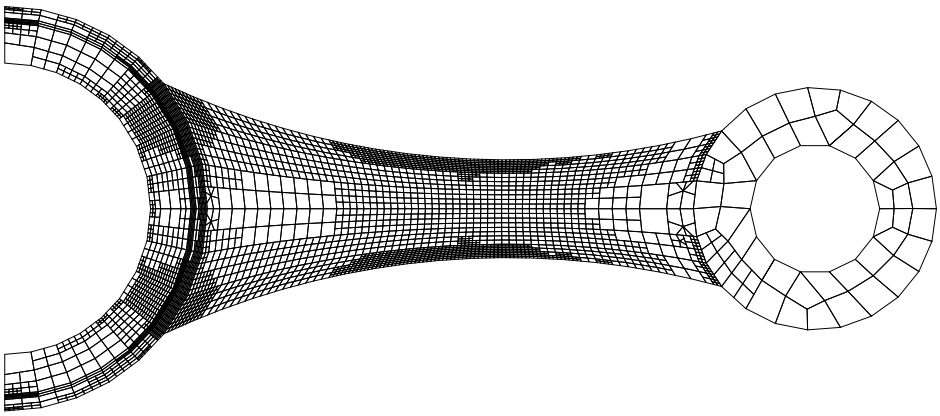


Figure 6.17: Mesh of the connecting rod after the third refinement

Figure 6.18: Boundary conditions of the sculpture

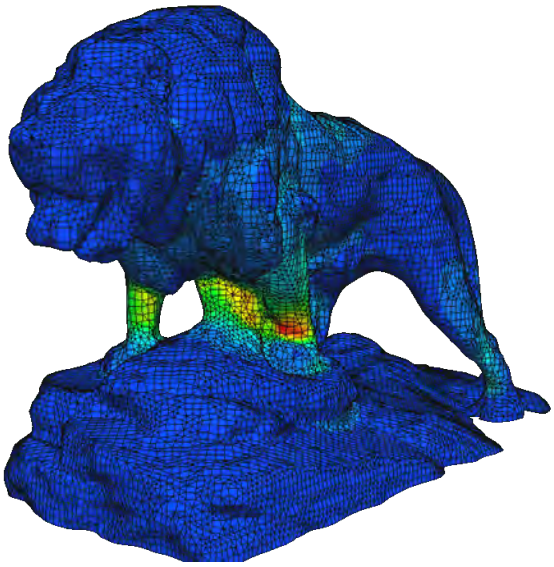
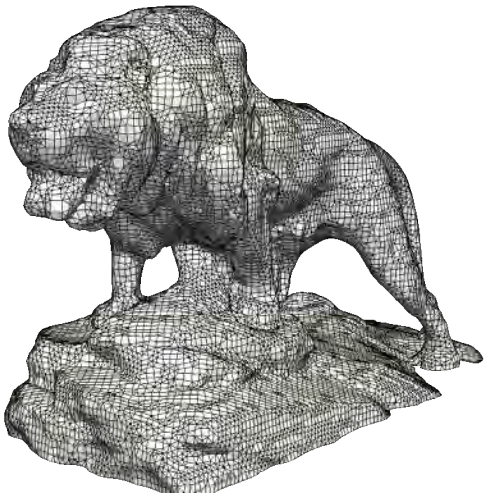
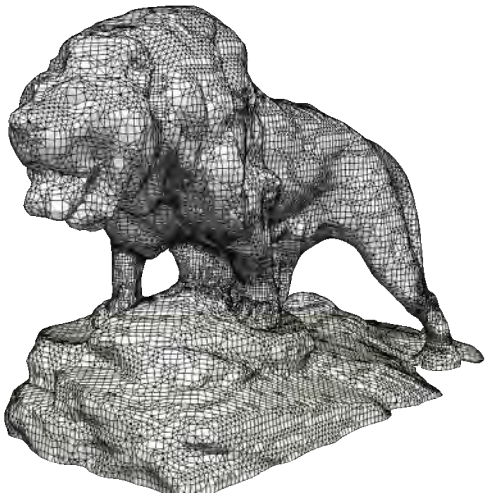


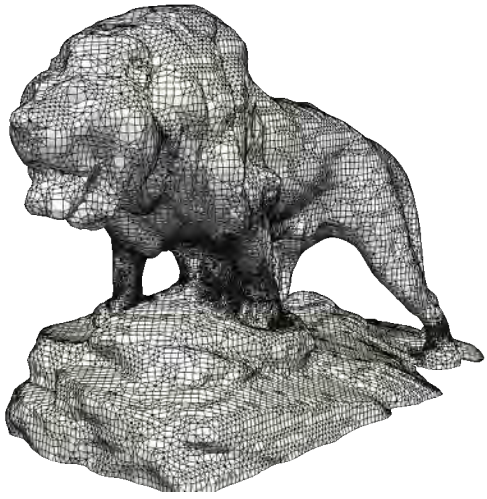
Figure 6.19: Von Mises stress of the sculpture



(a) Initial mesh



(b) First refinement



(c) Second refinement

Figure 6.20: Meshes of the sculpture

Chapter 7

Towards engineering analysis of virtual city

Virtual reality technique creates an immersive environment for engineers to interact with the models. It can help engineers with a variety of tasks such as design, maintenance and modification of models. With the rapid development of modern computers, modeling of a city in the virtual reality environment becomes possible, which can be referred to as a virtual city. Virtual city has been explored in the fields such as architecture (Dollner et al., 2006), transportation (Sun et al., 2002), archaeology (Dylla et al., 2008) and many others. Combining virtual city modeling and structural analysis has great potential in numerous engineering applications, such as urban planning, hazard simulation, structural health monitoring, etc. However, the progress in this field is still lacking. The key impedance is the difficulty in mesh generation from the geometric model. The geometric model of a virtual city can include conventional CAD model, image based model, STL model and virtual reality model. The virtual reality model is described in a data format different from the traditional CAD models. Furthermore, the models may contain defects which are not suitable for numerical modeling. Manual operations are frequently required, which are time consuming and error prone. Combin-

ing these meshes generated from those different data formats in an analysis is another difficulty as they are usually non-matching on the interface. In this chapter, a new approach to combine virtual city modeling with structural analysis is presented using the techniques developed in Chapter 3, Chapter 4 and Chapter 5.

7.1 Geometric modeling of virtual city

This section is a summary of the emerging techniques to construct virtual reality models for the buildings in a city. Three commonly used techniques, photogrammetry, LiDAR (light detection and ranging) and computed tomography are reviewed.

7.1.1 Photogrammetry

Photogrammetry is a technique to construct 3D models from 2D photos captured from different view angles. It is illustrated using an example in Fig. 7.1 (Modernist House #3, `https://3dwarehouse.sketchup.com`). This method is highly automatic and efficient, therefore it has been widely used in large scale modeling of buildings (Kobayashi, 2006). However, as photos only contain 2D information, the reconstruction of the 3D objects is a challenging task. Usually several simple shapes are predefined as templates for the buildings (Zlatanova et al., 1998; Hammoudi and Dornaika, 2011). The buildings in the photos are identified based on the templates. The photos are mapped to those templates as texture. A novel photo based modeling method without templates can be found in Xiao et al. (2009). Machine learning can be used in image recognition and classification. Aerial images can be used to construct large scale 3D models, such as terrain.

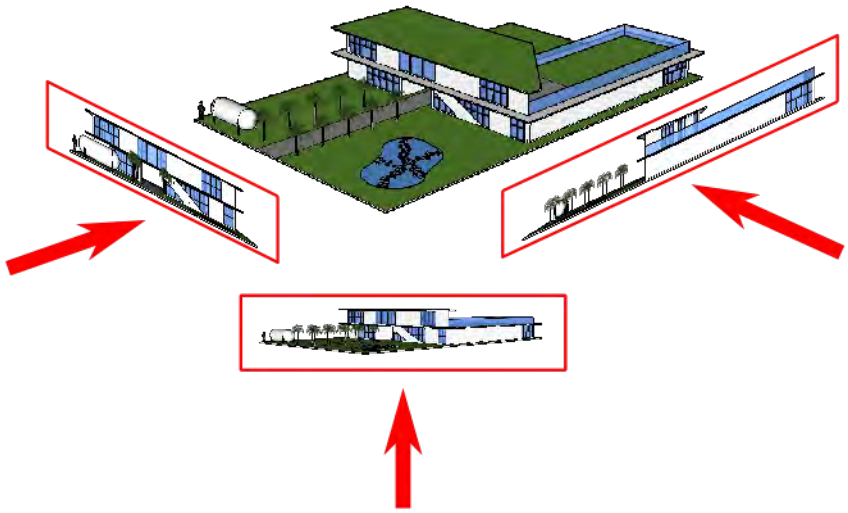


Figure 7.1: Geometric modeling of a building using photos captured from different angles

7.1.2 LiDAR

LiDAR is another commonly used approach in 3D city modeling. The LiDAR device transmits a light beam and receives the reflection signal. The distance between the LiDAR device and the object can be calculated. A 3D point cloud is constructed (Fig. 7.2), which can be later connected as a surface model. The advantage of LiDAR is that the 3D geometric information is obtained directly form the data. No predefined template is required. Buildings with complex and irregular geometries can be modeled using this technique. However, this approach is usually more time-consuming and requires more human effort when obtaining the raw data. Point reduction is often required due to the huge amounts of LiDAR data normally involved, which often fail to load in 3D modeling software (Heo et al., 2013).

7.1.3 Computed tomography

Computed tomography (CT) scan is a different method to construct 3D models using irradiation to produce internal and external representations. Different techniques based on X-ray (Du Plessis et al., 2016) and neutron radiography (Dewanckele et al., 2014)

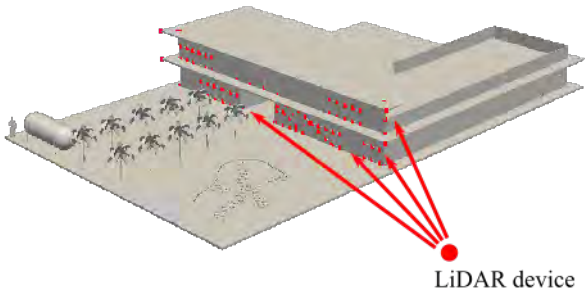


Figure 7.2: Geometric modeling of a building using LiDAR

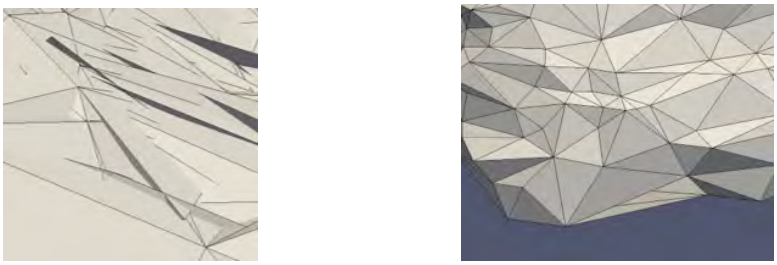
have been developed. The advantage of CT scan is that it is able to model the interior structure, such as cracks, inclusions and voids inside the model, while photogrammetry and LiDAR only provide information of the boundary. However, the application is usually limited by the size of the model. High resolution 3D digital images obtained from CT scan usually require more storage space compared with photos and point clouds.

7.2 Numerical modeling of virtual city

After the geometric model is constructed, it will be converted to a numerical model for structure analysis. The process is described in the following of this section and several issues are discussed.

7.2.1 Automatic mesh generation

The virtual reality models are usually designed for visualization of the geometry rather than numerical analysis. Therefore there may be defects in the geometric models, such as self-intersection (Fig. 7.3a) and holes (Fig. 7.3b). These defects are difficult to handle



(a) Self-intersection (b) Hole

Figure 7.3: Defects in the geometric model

for mainstream mesh generation methods such as mapping and advancing front, as they require manual repair and reconstruction. In large scale numerical modeling, manual operation is not preferable. Therefore, octree based method is utilized to generate polyhedron mesh automatically for the models. In the octree based method, the self-intersection can be tolerated as long as the intersection points between the model and the octree background mesh can be identified. The holes smaller than the element size can also be accepted if there is no edge in the octree background mesh passing through the holes.

Fast mesh size transition can be achieved using octree based algorithm. Each time the mesh is refined, the element size is halved. In virtual city analysis the mesh is only refined in the interested region, while in other regions element size grows fast. Therefore in large scale problems computational resources can be used effectively.

7.2.2 Local mesh modification

Another impedance in virtual city analysis using the conventional FEM is the difficulty in local modification of the mesh. For instance, the modification of a building will affect the mesh of the soil under the building, which may result in remeshing of the whole city. Apparently this is not preferable in an analysis. In the present approach, different parts of a city can be meshed independently by different groups of engineers. The mesh of the terrain can be generated by geotechnical engineers using surface model obtained by

photogrammetry and the drilling data of the soil layers. The buildings can be modeled by individual design companies. These meshes are generated independently, therefore they are usually non-matching on the interface. The coupling method proposed in Chapter 5 is able to merge those meshes automatically. Only a small number of elements near the interface need to be modified.

When the design of a building is modified, only the mesh of that building needs to be regenerated. The original mesh of the soil can be coupled with the new mesh of the building. If a part of the model needs to be removed, such as ground evacuation and tunnel drilling, some elements in the original octree mesh are removed first, leaving an empty space. Then a new mesh is inserted between the remaining elements and the evacuated boundary. These modifications enable the same meshes to be reused in different problems, which saves a large amount of human effort. The details will be demonstrated in the numerical examples in Section 7.3.3 and Section 7.3.4.

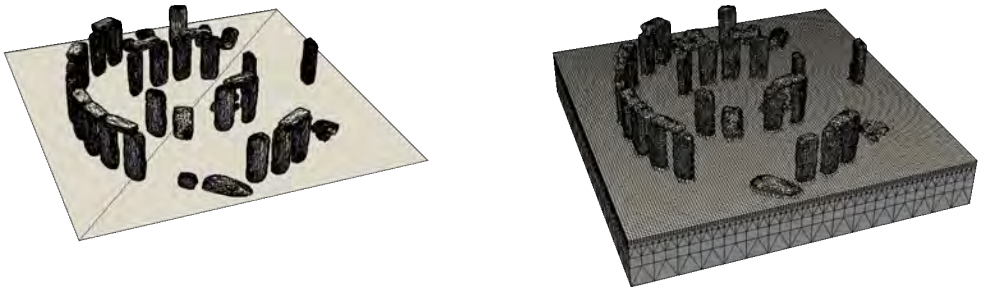
7.3 Numerical examples

In this section, four numerical examples are presented to demonstrate the idea of virtual city analysis.

7.3.1 Stonehenge

The Stonehenge is considered in this example to test the robustness of the proposed mesh generation method (`https://www.thingiverse.com/thing:2250774`). The geometry is shown in Fig. 7.4a. It can be observed that the geometric model contains distorted triangles and self-intersections, which makes it difficult to generate a conventional FEM mesh. A polyhedron mesh is generated in Fig. 7.4b automatically using the octree algorithm.

At the vertical boundaries and the bottom of the soil, the displacements perpendic-



(a) Geometry of the Stonehenge (b) Mesh of the Stonehenge

Figure 7.4: Geometry and mesh of the Stonehenge

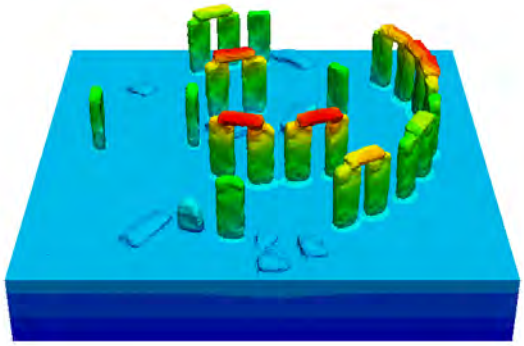


Figure 7.5: Displacement of the Stonehenge

ular to the surfaces are constrained. The material is granite. The Young's modulus is $E = 5 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.2$. The mass density is $\rho = 2.7 \times 10^3$kg/m^3. The displacement under self-weight is shown in Fig. 7.5.

7.3.2 Virtual repair of a column

In this example, the combination of image based model and virtual reality model is explored. A damaged column in an ancient temple is shown in Fig. 7.6 (`https://www.thingiverse.com/thing:56740`). There is a notch in the middle part of the column.

The geometry of the column is stored as 3D digital images (Fig. 7.7a). Mesh of

Figure 7.6: A damaged column

the column is generated based on the images using the technique developed in Saputra et al. (2017). A replacement part is produced using 3D printing techniques. The STL model of the replacement part is shown in Fig. 7.7b. Mesh of the replacement part is generated from the STL model without trimming (Fig. 7.7c). The two meshes are merged in Fig. 7.7d.

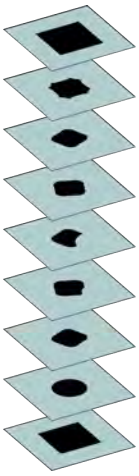
The material of the column is limestone. The Young's modulus is $E = 7 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.2$. The mass density of is $\rho = 2.7 \times 10^3$kg/m^3. It is assumed that the material properties of the 3D printing material is the same as the material of the damaged column. The deformations of the column under self-weight before and after the repair are shown in Fig. 7.8a and Fig. 7.8b.

7.3.3 Modeling of city and foundation

In this example, several buildings in a city are modeled together with several layers of soil representing the foundation. The different stages of the modeling process are presented and the local modification techniques are applied.

7.3.3.1 Modeling of foundation

The soil in this example is divided into three different layers according to their different material properties. The different materials are represented by different colors in Fig. 7.9. The bottom layer is well graded gravel (Fig. 7.9a). The Young's modulus is $E = 7 \times 10^7$Pa and Poisson's ratio $\nu = 0.15$. The middle layer is clay (Fig. 7.9b). The Young's modulus is $E = 3 \times 10^7$Pa and Poisson's ratio $\nu = 0.2$. The top layer is assumed to be silty sand (Fig. 7.9c). The Young's modulus is $E = 1 \times 10^7$Pa and Poisson's ratio $\nu = 0.3$. The meshes are generated using the same bounding box and element size. The three layers are merged together for later analysis.



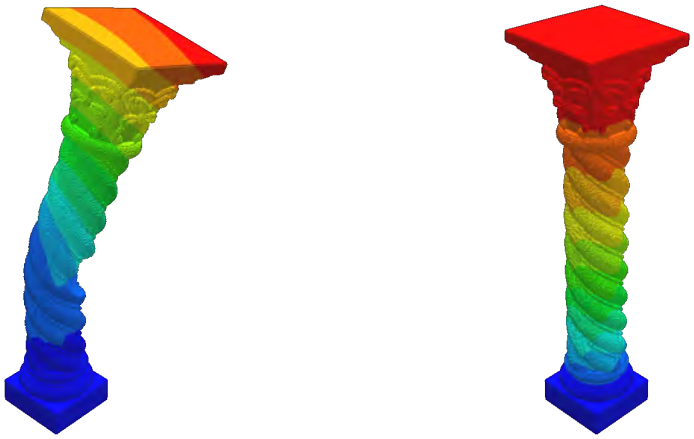
(a) Images of the damaged column

(b) STL model of a replacement part

(c) Merging of the meshes

(d) Mesh of the repaired column

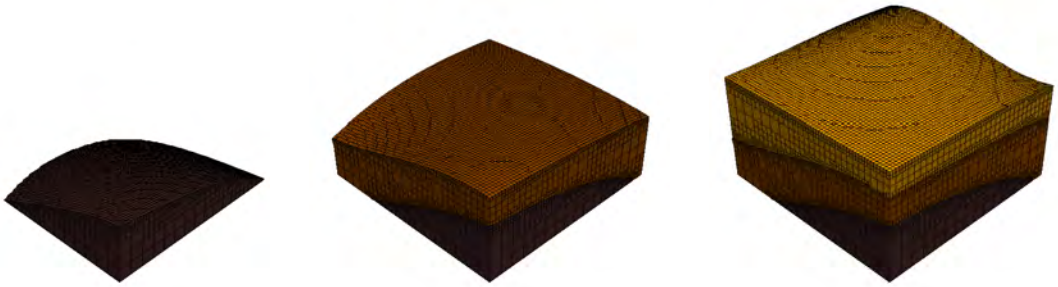
Figure 7.7: Repair of the column



(a) Deformation of the column before repair

(b) Deformation of the column after repair

Figure 7.8: Deformation of the column

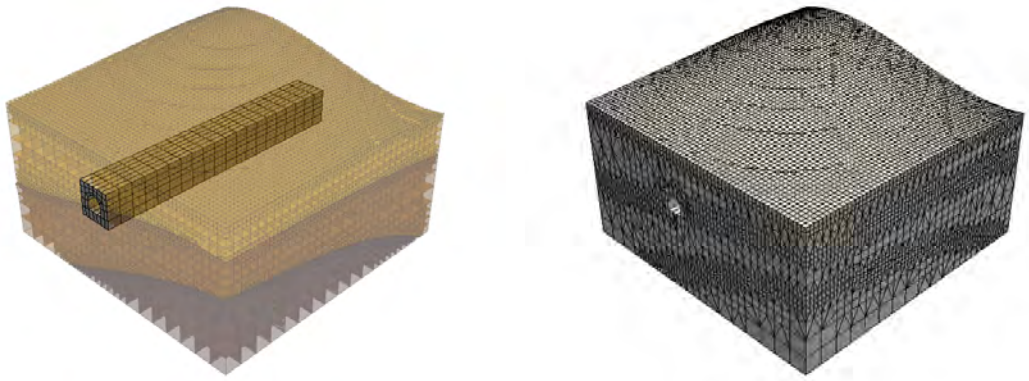


(a) Lower layer

(b) Middle layer

(c) Top layer

Figure 7.9: Different layers of soil



(a) Mesh of the tunnel

(b) Mesh of the tunnel merged with the foundation

Figure 7.10: Construction of a tunnel

7.3.3.2 Modeling of tunnel

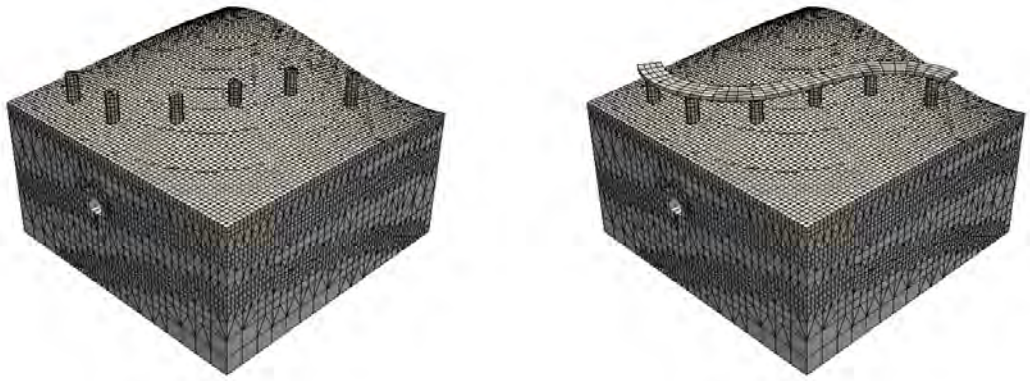
A tunnel passing through the interface between the top and middle soil layer is constructed. The elements in the merged mesh of the foundation are removed around the location of the tunnel, leaving an empty space slightly larger than the tunnel. A hexahedron mesh is generated using sweeping method, filling the gap between the tunnel and the remaining mesh of the foundation (Fig. 7.10a). The two non-matching meshes are connected by merging their interface (Fig. 7.10b).

7.3.3.3 Modeling of bridge

A bridge is constructed on ground. Structured hexahedron meshes are generated for the columns and slab of the bridge. The columns are merged with the foundation (Fig. 7.11a) and then the slab is merged with the columns (Fig. 7.11b).

7.3.3.4 Modeling of buildings

Several buildings are constructed on both sides of the bridge. The buildings are represented in STL format. Polyhedron meshes are generated for the buildings based on oc-



(a) Meshes of the columns of the bridge (b) Mesh of the slab of the bridge

Figure 7.11: Construction of a bridge

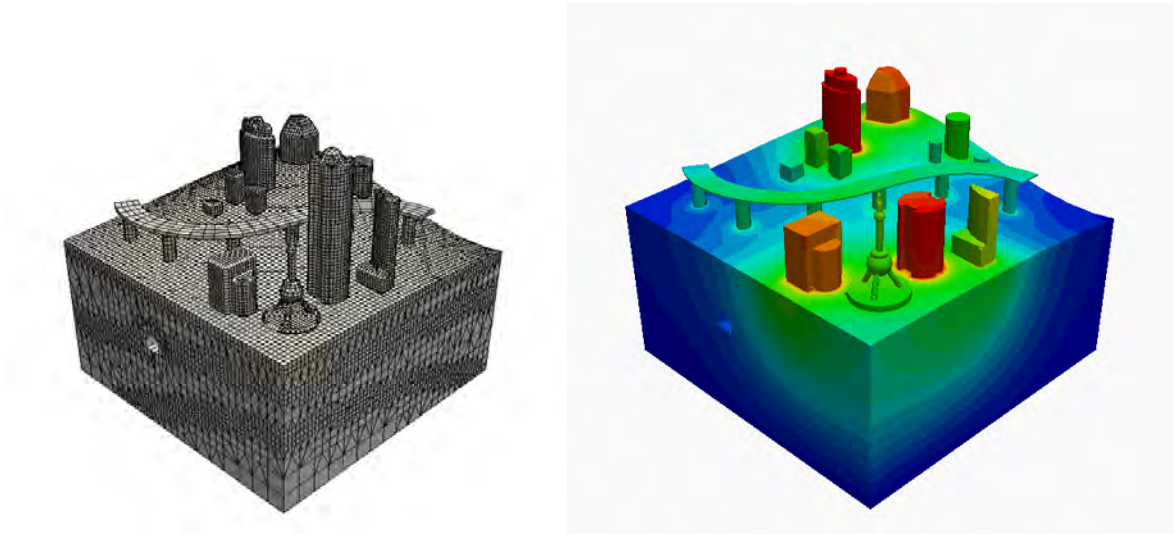
tree algorithm. The meshes of the buildings are merged with the foundation (Fig. 7.12a). At the vertical boundaries and the bottom of the foundation, the displacements perpendicular to the surfaces are constrained. The material of the buildings and the bridge is assumed to be concrete. The Young's modulus is $E = 1.7 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.3$. The mass density of the concrete is $\rho = 2.4 \times 10^3$kg/m^3. The self-weight of the soil is not considered. The displacement under the self-weight of the bridge and buildings is shown in Fig. 7.12b.

7.3.4 Modeling of city and terrain

In this example, the combination of the city and the terrain is explored.

7.3.4.1 Modeling of terrain

A part of the terrain near Mount Crichton in New Zealand on Google map is shown in Fig. 7.13. The size of the area is 12km × 6km (red rectangle in Fig. 7.14a). The surface model of the terrain is saved in STL format (Fig. 7.14c). Octree mesh can be generated using the surface model. The size of the bounding box is 8km × 5km × 3km. The ratio between maximum element size and minimum element size is 16 (Fig. 7.14d).



(a) Merged mesh of the buildings and foundation (b) Displacement of the model

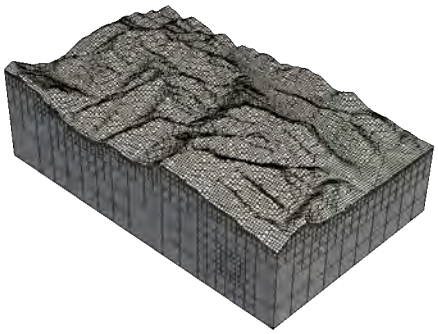
Figure 7.12: Construction of buildings

Figure 7.13: Mount Crichton in Google map

(a) 3D terrain in Google map

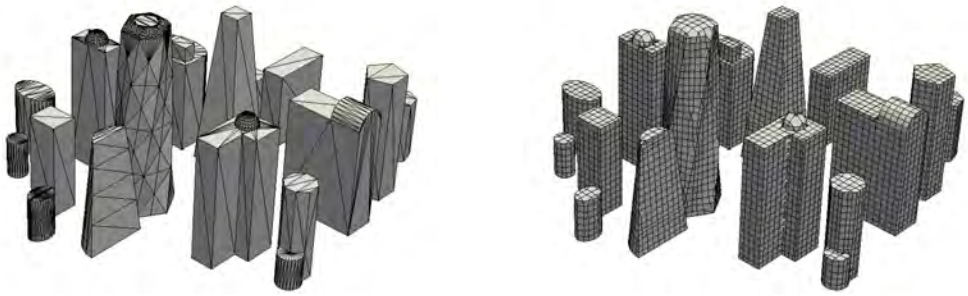
(b) STL surface overlapping Google map

(c) Extracted STL surface



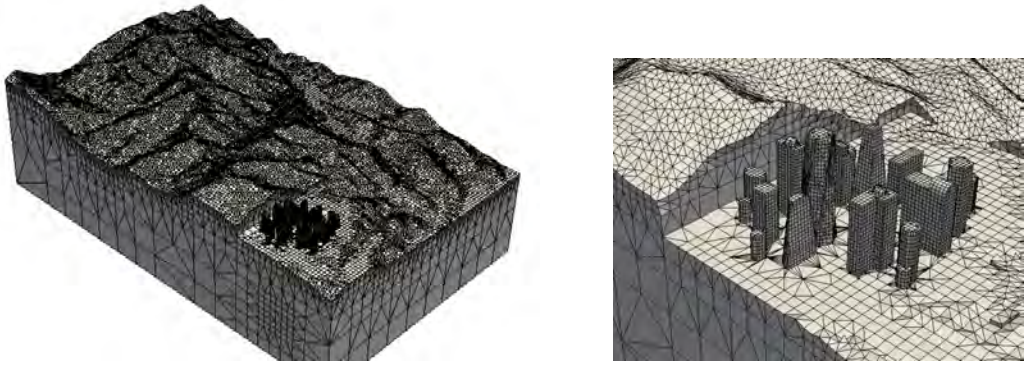
(d) Octree mesh of the terrain

Figure 7.14: Numerical modeling of the terrain



(a) STL model of the buildings (b) Octree mesh of the buildings

Figure 7.15: Numerical modeling of the buildings



(a) Overall view of the terrain and buildings (b) Detailed view of the terrain and buildings

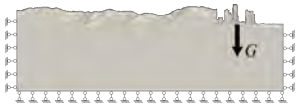
Figure 7.16: Coupling of the meshes of the terrain and buildings

7.3.4.2 Coupling of terrain and city

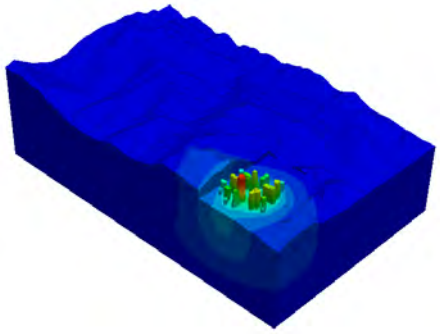
Several buildings representing a city block are shown in Fig. 7.15a. The buildings are also meshed using the octree algorithm (Fig. 7.15b).

The meshes of the buildings are connected to the mesh of the terrain (Fig. 7.16). To simplify the problem, a flat interface is constructed by removing some elements from the mesh of the terrain.

The displacements perpendicular to the surfaces are constrained at the vertical boundaries and the bottom of the terrain. Only the self-weight of the buildings is applied as external load. The material of the ground is assumed to be rock and homogeneous.

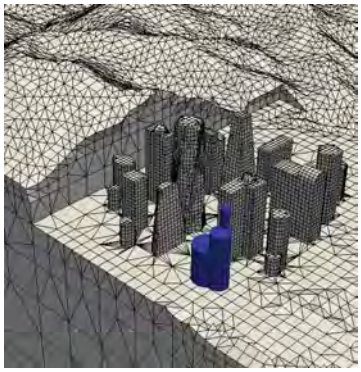


(a) Boundary conditions of the model



(b) Displacement of the merged terrain and buildings

Figure 7.17: Boundary conditions and displacement of the merged terrain and buildings



(a) Location of the new building

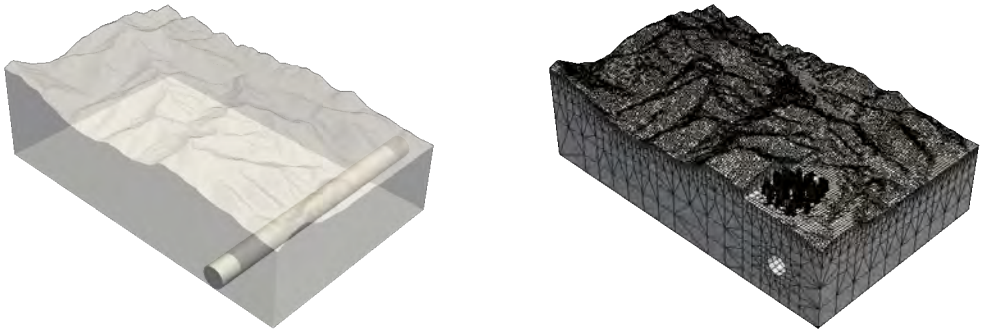
(b) Mesh of the new building merged with existing mesh

Figure 7.18: Construction of a new building

The Young's modulus is $E = 5 \times 10^9$Pa and Poisson's ratio $\nu = 0.15$. The material of the buildings is concrete. The Young's modulus is $E = 1.7 \times 10^{10}$Pa and Poisson's ratio $\nu = 0.3$. The mass density of the concrete is $\rho = 2.4 \times 10^3 \text{kg/m}^3$. The boundary conditions are shown in Fig. 7.17a. The displacement of the model is plotted in Fig. 7.17b.

7.3.4.3 Interaction between new and existing structures

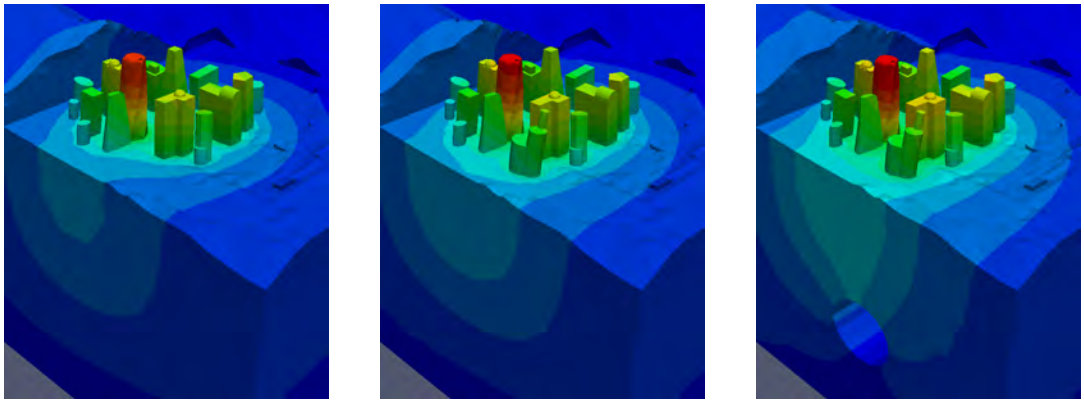
A new building is constructed next to the site (Fig. 7.18a). The meshes of the terrain and existing buildings can be reused. Only the elements on the interface near the new building needs to be modified. The meshes become matchingon the interface, which is shown in Fig. 7.18b.



(a) STL model of the tunnel

(b) Octree mesh after the tunnel is subtracted

Figure 7.19: Construction of a tunnel



(a) Displacement before construction of the new building

(b) Displacement after construction of the new building

(c) Displacement after construction of the tunnel

Figure 7.20: Displacement of the model at different stages

7.3.4.4 Construction of a tunnel

The geometry of a tunnel is represented by a cylinder (Fig. 7.19a). The mesh of the terrain is trimmed by the tunnel. The trimming follows the same procedure in Section 4.4. The elements located inside the tunnel are removed after trimming (Fig. 7.19b). No remeshing is required for the whole model.

The displacements of the model in different stages are shown in Fig. 7.20. A comparison of the deformations in a selected section is shown in Fig. 7.21.

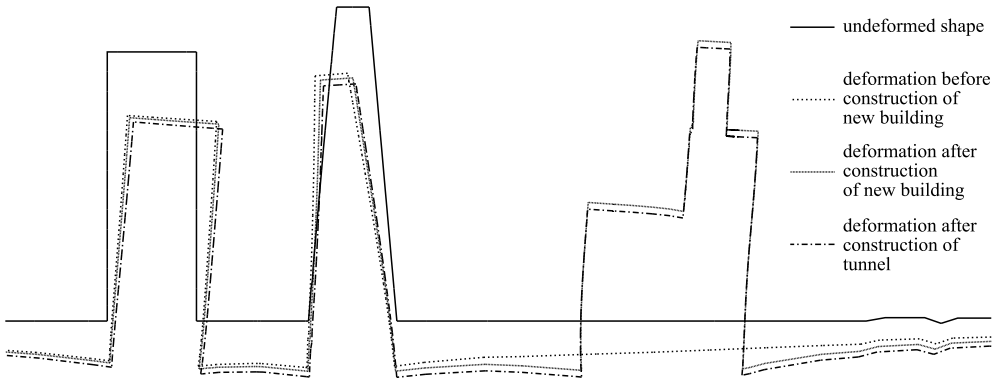


Figure 7.21: Slice view of the deformation of the city. The deformation is scaled by 1000.

Chapter 8

Discrete modeling of fiber reinforced material

Fiber reinforced materials have been widely used in engineering due to their excellent mechanical and thermal properties. A large number of fiber reinforced materials have been designed and manufactured to fit different purposes in modern engineering. The mechanical property of the matrix (base) material can be significantly improved by adding only a small number of fibers. It is important to evaluate the mechanical property of the fiber reinforced material. A traditional evaluation method is experiment. However, experiments are usually time consuming and expensive. The experiments can only be carried out after the specimen has been produced. Several empirical formulae were proposed as alternatives (Halpin, 1969; Mori and Tanaka, 1973). The drawback of this approach is that it is difficult to represent complex fiber distribution and orientation. Numerical method offers better flexibility and feasibility in the modeling of fiber reinforced materials. The numerical methods to model fiber reinforced materials can be divided into two categories, the smeared method and the discrete method (Radtke et al., 2010; Kang et al., 2014). In the smeared method, the effect of fibers is distributed in the matrix (Owen et al., 1983; Vecchio, 1989; Barzegar and Maddipudi,

1997). The material properties of the matrix in certain directions are modified. The mesh generation is simple in a smeared method. It is efficient as no additional degree of freedom is introduced to the system. However, this method can not describe the stress distribution along the fibers. The discrete method can be further classified into two approaches. The first approach requires that the nodes of the fibers have to be coincident with some nodes of the matrix (Hbaieb et al., 2007; Kang et al., 2014; Silani et al., 2014). Conventional finite element solver can be utilized to solve the equations. The cost is that the mesh generation is difficult, especially when large amount of fibers are considered (Radtke et al., 2010). In the other approach, the mesh of the matrix can be generated independently, and special interface constraints are employed to connect the fibers and the matrix (Vanalli et al., 2008; Bolander et al., 2008). The difficulty in mesh generation can be greatly reduced. However, additional degrees of freedom need to be added to the system, which may result in non positive-definite matrix.

In this chapter, a discrete method for the modeling of fiber reinforced material based on the scaled boundary finite element method (SBFEM) is presented. The meshes of the matrix and the fibers can be generated independently. Polygon elements are used to model the matrix. The fibers are modeled with truss elements. A novel embedment method is developed to combine the meshes of the matrix and the fibers. The edges in the matrix intersected with the fibers are partitioned and the intersection points are inserted to the mesh. The fiber tips located inside the elements of matrix are embedded by subdividing those elements into several smaller elements using polytree algorithm. The remaining of the elements in the matrix don't need subdivision as polygons with arbitrary number of edges and nodes can be analyzed in SBFEM directly. All the nodes in the fibers are overlapping with some nodes of the matrix. Therefore, the two meshes become matching along the fibers. This work focuses on the mesh generation. The slippage of fibers is not considered. The stiffness matrix of the fibers is superposed to the matrix material directly. No interface elements are introduced to the system.

Realistic, random fiber distributions can be analyzed in the model.

8.1 Embedment of fiber to matrix

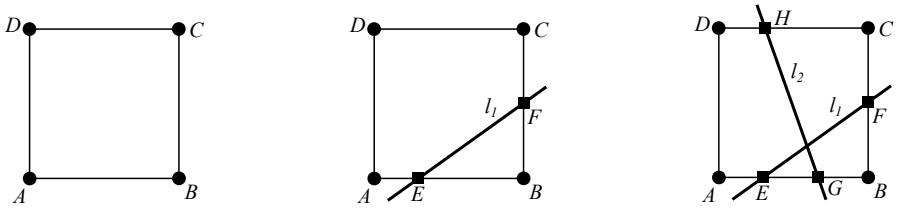
In the finite element method, the fibers are usually modeled as truss or beam elements. The fibers usually need to be matching with the mesh of the matrix, i.e. for each point on the fiber, there must be a point in the matrix overlapping with it. Due to the advantage of arbitrary polytope element formulated in SBFEM, a simple algorithm can be implemented to make the mesh of the fiber and the matrix compatible.

8.1.1 Embedment of fiber in 2D

The procedure of embedding fiber to matrix is illustrated in Fig. 8.1 and Fig. 8.2 in 2D. A quadrilateral element $ABCD$ is shown in Fig. 8.2a. A fiber l_1 is passing through the solid element with a random orientation. The intersection points are marked as E and F, as shown in Fig. 8.1b. The two intersected edges of the quadrilateral element, AB and BC, are partitioned into new edges AE, EB and BF, FC, respectively. The quadrilateral element becomes a polygon with 6 edges. If there are more than one fibers passing through the element (Fig. 8.1c), all the intersection points are inserted and the edges are partitioned. The stiffness matrix of the fibers is formulated using

$$K_{bar} = \frac{EA}{L}\begin{bmatrix} C_x^2 & C_xC_y & -C_x^2 & -C_xC_y \\ C_xC_y & C_y^2 & -C_xC_y & -C_y^2 \\ -C_x^2 & -C_xC_y & C_x^2 & C_xC_y \\ -C_xC_y & -C_y^2 & C_xC_y & C_y^2 \end{bmatrix}, \tag{8.1}$$

and superposed into the stiffness of the polygon element based on node connectivity. E is the Young's modulus of the fiber and A is the cross section area. L is the length of



(a) An element in the mesh of the matrix

(b) A fiber intersected by the element

(c) Two fibers intersected by the element

Figure 8.1: Intersection between the fiber and the matrix in 2D

the segment. C_x and C_y are defined as

$$C_x = \frac{x_2 - x_1}{L}, \tag{8.2a}$$

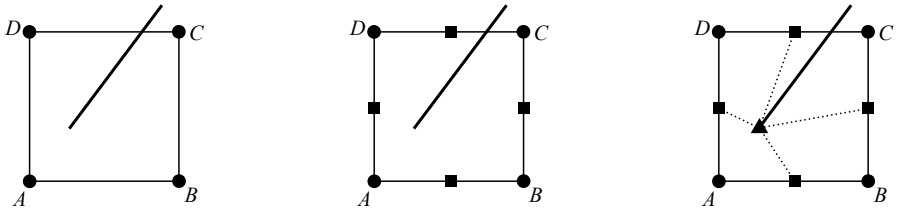
$$C_y = \frac{y_2 - y_1}{L}, \tag{8.2b}$$

where (x_1, y_1) and (x_2, y_2) are the coordinates of starting point and end point of the bar segment.

If a fiber tip is located inside a polygon element as shown in Fig. 8.2a, the element will be subdivided into several smaller elements using polytree algorithm introduced in Section 5.3.1. A new node is inserted at the fiber tip. The polygon is partitioned by bisecting all of its edges. The mid-points of the edges are inserted as new nodes (Fig. 8.2b). The fiber tip is connected to the mid-points, which is shown in Fig. 8.2c. In 2D the new elements are always quadrilaterals.

8.1.2 Embedment of fiber in 3D

In 3D the procedure of embedment is similar. If a fiber intersects with a solid element as shown in Fig. 8.3a, the intersection points between the fiber and the faces of the solid element are inserted as new nodes. The faces intersected by the fiber are subdivided into several new faces using the polytree refinement in 2D. No volume discretization is required. In Fig. 8.3b, the hexahedron becomes a polyhedron with 12 faces after the



(a) A fiber tip located inside an element

(b) Bisecting of the edges of the element

(c) The new elements generated by connecting the fiber tip and the middle points

Figure 8.2: Embedment of fiber tip in 2D

embedment. The stiffness matrix of the bar in 3D is expressed as

$$K_{bar} = \frac{EA}{L}\begin{bmatrix} C_x^2 & C_xC_y & C_xC_z & -C_x^2 & -C_xC_y & -C_xC_z \\ C_xC_y & C_y^2 & C_yC_z & -C_xC_y & -C_y^2 & -C_yC_z \\ C_xC_z & C_yC_z & C_z^2 & -C_xC_z & -C_yC_z & -C_z^2 \\ -C_x^2 & -C_xC_y & -C_xC_z & C_x^2 & C_xC_y & C_xC_z \\ -C_xC_y & -C_y^2 & -C_yC_z & C_xC_y & C_y^2 & C_yC_z \\ -C_xC_z & -C_yC_z & -C_z^2 & C_xC_z & C_yC_z & C_z^2 \end{bmatrix}, \tag{8.3}$$

where C_x, C_y and C_z are defined as

$$C_x = \frac{x_2 - x_1}{L}, \tag{8.4a}$$

$$C_y = \frac{y_2 - y_1}{L}, \tag{8.4b}$$

$$C_z = \frac{z_2 - z_1}{L}, \tag{8.4c}$$

where (x_1, y_1, z_1) and (x_2, y_2, z_2) are the coordinates in 3D of starting point and end point of the segment.

If a fiber tip is located inside a solid element in 3D (Fig. 8.4a), the solid element will be divided into several new polyhedrons using polytree refinement in 3D, which is

(a) A hexahedron element in the mesh of the matrix

(b) A fiber intersected by the element

Figure 8.3: Intersection between the fiber and the matrix in 3D

described in Section 5.3.2. The fiber tip is then connected to the new inserted node inside the solid element as shown in Fig. 8.4c.

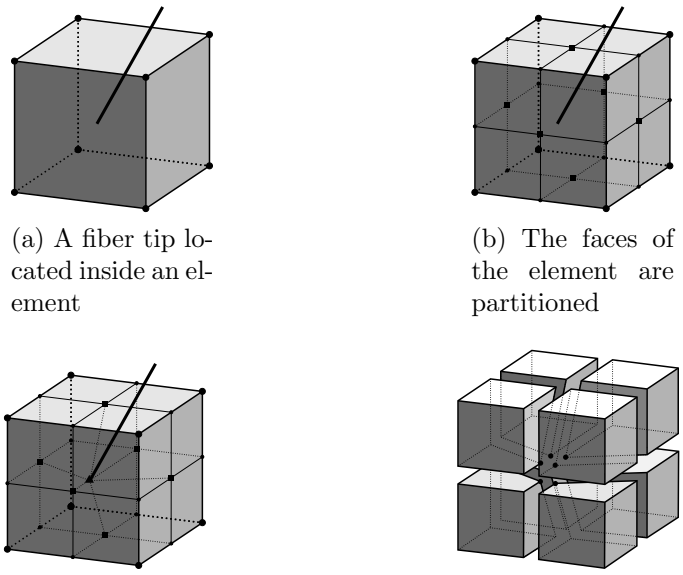
8.2 Optimization of matrix mesh quality

The mesh quality of the matrix has significant influence on the accuracy of the result. In this section, a shifting scheme is developed to improve mesh quality while embedding the fiber into the matrix.

8.2.1 Shifting to fiber tips

If a node in the mesh of matrix is close a fiber tip, it will result in distorted polygons. In Fig. 8.5a the fiber tip O is close the the node A in element $ABCD$. If the element is subdivided using the method in Section 8.1 directly, there will be a concave quadrilateral $AEOH$ which should be avoided in an analysis. Therefore node A is shifted to fiber tip O in Fig. 8.5b. No subdivision is required in this case, and the mesh quality of is maintained. The shifting tolerance at a node is $tol = \varepsilon_n \times l_n$, where ε_n is a predefined relative tolerance (typically, a value between 0.2 and 0.3) and l_n is the minimum length of edges connected to the node.

A similar case in shown in Fig. 8.5c, where the fiber tip O is close to an edge AD. The obtained quadrilateral $DHOG$ is also distorted by direct subdivision. In order



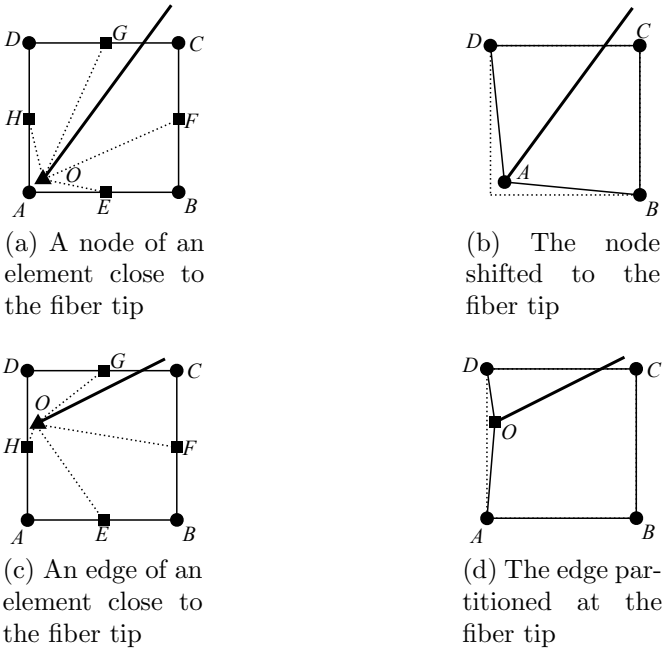
(a) A fiber tip located inside an element

(b) The faces of the element are partitioned

(c) The element is divided into new elements connecting the fiber tip and the centers of original faces

(d) The hexahedron elements after subdivision

Figure 8.4: Embedment of fiber tip in 3D



(a) A node of an element close to the fiber tip

(b) The node shifted to the fiber tip

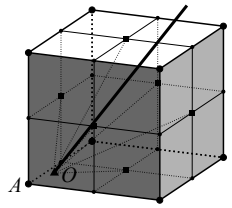
(c) An edge of an element close to the fiber tip

(d) The edge partitioned at the fiber tip

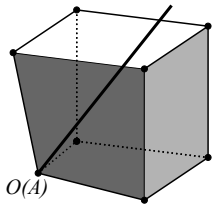
Figure 8.5: Shifting of nodes and partitioning of edges near a fiber tip in 2D

to improve the mesh quality, edge AD is partitioned into AO and OD and no further subdivision is required for this element. The tolerance at an edge is $tol = \varepsilon_e \times l_e$, where ε_e is a another relative tolerance smaller than ε_n (usually between 0.1 and 0.2) and l_e is the length of the edge.

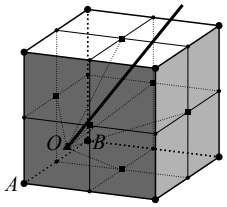
In 3D more cases need to be considered in the shifting procedure. The shifting of nodes (Fig. 8.6a) and partition of edges (Fig. 8.6c) are the similar to the process in 2D. In addition, if there is a fiber tip located near a face, e.g. face $ABCD$ in Fig. 8.6e, the polyhedrons after subdivision will be distorted around the face. Therefore, the face is subdivided into several new faces at the fiber tip as shown in Fig. 8.6f. The volume of the polyhedron is not subdivided anymore, and no distorted polyhedron will be generated. The shifting tolerance at a face is $tol = \varepsilon_f \times l_f$, where ε_f is a relative tolerance typically between 0.05 and 0.1) and l_f is the average length of the edges in the face.



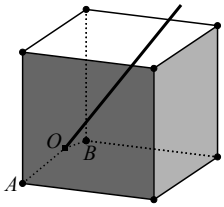
(a) A node of an element close to the fiber tip



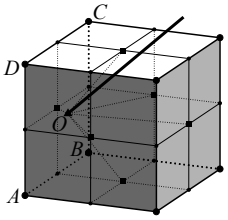
(b) The node shifted to the fiber tip



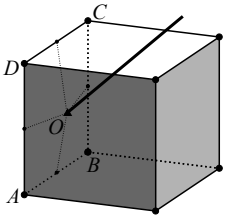
(c) An edge of an element close to the fiber tip



(d) The edge partitioned at the fiber tip



(e) A face of an element close to the fiber tip



(f) The face partitioned at the fiber tip

Figure 8.6: Shifting of nodes and partitioning of edges and faces near a fiber tip in 3D

(a) A node of an element close to the fiber segment

(b) The node shifted to the fiber

Figure 8.7: Shifting of nodes near the fiber in 2D

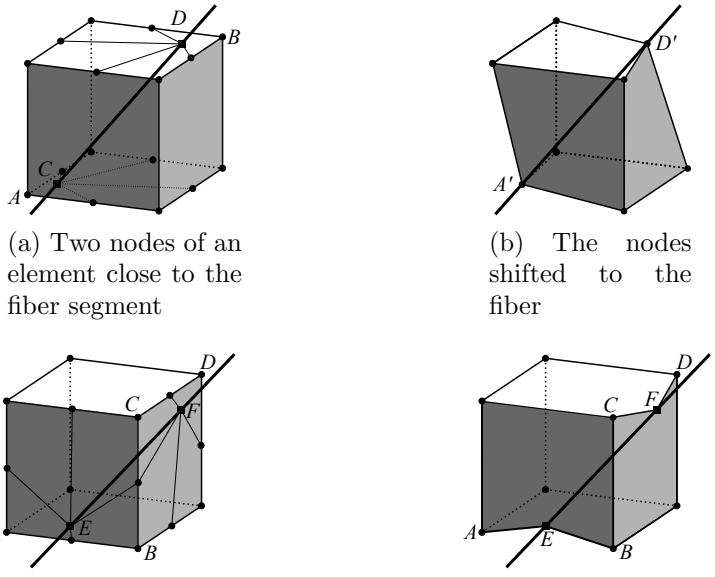
8.2.2 Shifting to fiber segments

The shifting scheme in Section 8.2.1 eliminates the distortion around the fiber tips. However, the nodes which are close to the middle of the fibers may cause distortion as well. In Fig. 8.7a a node C is close to the fiber (but not the fiber tip). After partition of the edges, there will be a short edge CF in the polygon. In order to avoid such cases, the node is shifted to the nearest point to the fiber. The same shifting tolerance $tol = \varepsilon_n \times l_n$ can be used.

In 3D the nodes close to the fibers are shifted using the same approach as shown in Fig. 8.8a and Fig. 8.8b. The new case is when some edges of the solid element are close the fiber (Fig. 8.8c). The quality of the faces on the boundary are distorted. A point is located on the fiber which is closest to the edge, e.g. point E on the fiber in Fig. 8.8d. The edge AB is partitioned into AE and EB. The faces containing the original edge AB will be updated accordingly. As a result, the faces on the boundary after discretization will have better quality.

8.3 Numerical examples

In this section, six numerical examples are presented to validate the proposed method. A variety of different applications are considered.



(a) Two nodes of an element close to the fiber segment

(b) The nodes shifted to the fiber

(c) Two edges of an element close to the fiber segment

(d) The edges partitioned at the nearest points to the fiber

Figure 8.8: Shifting of nodes and partitioning of edges near fiber in 3D

8.3.1 Patch test in 2D

This example is a square reinforced with two bars subject to a uniaxial tension load. Plane stress condition is considered. The dimension of the square is 1×1m. The length of the bars is 1m and the radius is 0.01m (Fig. 8.9). The rigid body motion is constrained. On the top and bottom surfaces of the matrix, uniform surface tractions t are applied. Point loads F are applied to the ends of the bars. Young's modules and Poisson's ratio of the matrix are $E_m = 1 \times 10^3$Pa and $\nu = 0.3$. The Young's modules of the bars is $E_f = 1 \times 10^6$Pa.

The normal strain in vertical direction in both the matrix and the bar is prescribed as a constant $\varepsilon_y = 0.001$. Therefore, the exact displacement field is linearly distributed

$$u_x = -\nu\varepsilon_y x = -0.0003x, \tag{8.5a}$$

$$u_y = \varepsilon_y y = 0.001y, \tag{8.5b}$$

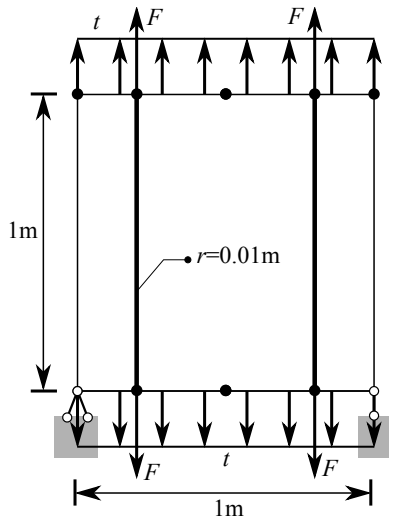


Figure 8.9: Geometry and boundary conditions of the patch test

where x and y are the coordinates of a point. The only non-zero stress component in the matrix is

$$\sigma_{my} = E_m \varepsilon_y = 1\text{Pa}. \tag{8.6}$$

The magnitude of the tractions applied on the top and bottom surfaces is $t = 1$Pa. The stress in the bars is

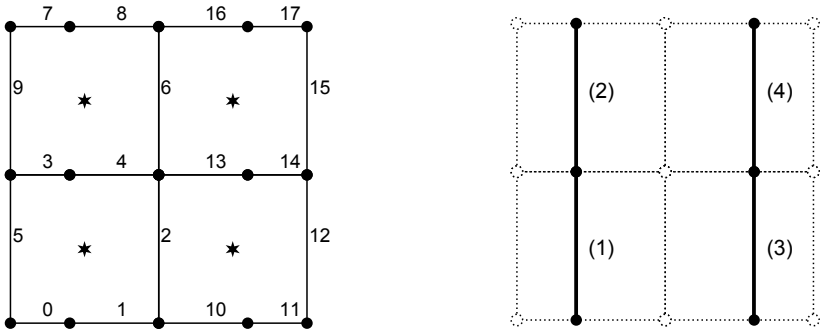
$$\sigma_{fy} = E_f \varepsilon_y = 1000\text{Pa}. \tag{8.7}$$

The axial forces in the bars are

$$F = \pi r^2 \sigma_{fy} = 0.1\pi\text{N}. \tag{8.8}$$

The square is discretized into 4 subdomains (Fig. 8.10a). Each subdomain contains 6 edges. Each of the bars is discretized into 2 segments (Fig. 8.10b).

The deformation of the matrix and the bars is shown in Fig. 8.11. The vertical displacement of the matrix is plotted in color. It is observed that the displacement



(a) Mesh of the matrix (b) Mesh of the bars

Figure 8.10: The meshes of the matrix and the reinforcement bars

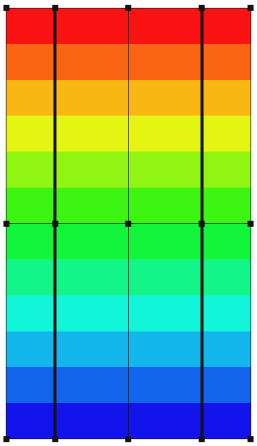
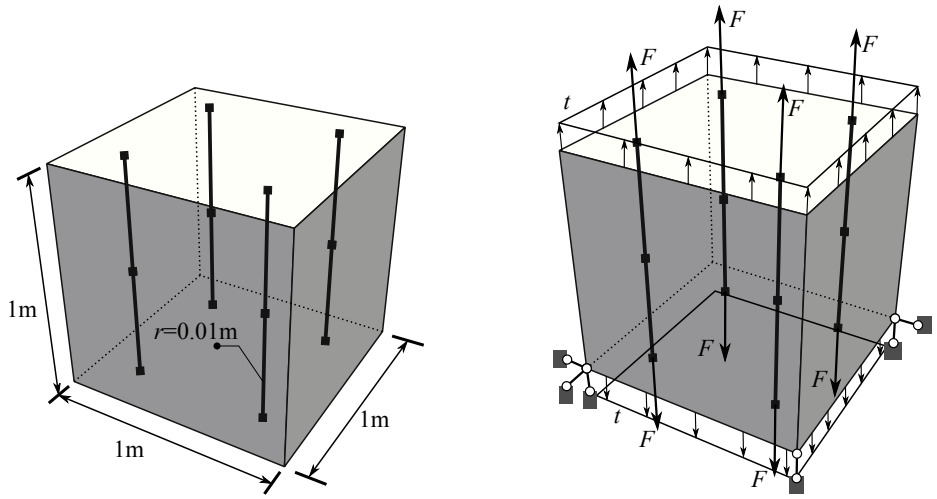


Figure 8.11: The vertical displacement of the matrix. The deformation is scaled by 500.

is linear along the the vertical direction. As first order element is accurate enough to represent linear displacement field, the error norm is 1.6588×10^{-14}.

8.3.2 Patch test in 3D

A 3D version of the patch test is considered. This example is a cube reinforced with 4 bars subject to a uniaxial tension load. The geometry of the patch test is shown in Fig. 8.12a. The dimension of the cube is $1 \times 1 \times 1$m. The length of the bars is 1m and the radius is 0.01m. The rigid body motion is constrained. On the top and bottom surfaces of the cube, uniform tractions of magnitude t are applied. In addition, point loads F are applied to the ends of the bars (Fig. 8.12b). Young's modules and Poisson's



(a) Geometry of the patch test

(b) Boundary conditions of the patch test

Figure 8.12: Geometry and boundary conditions of the patch test

ratio of the matrix are $E_m = 1 \times 10^3$Pa and $\nu = 0.3$. The Young's modules of the bars is $E_f = 1 \times 10^6$Pa.

The strain in vertical direction in both the matrix and the bar is prescribed as a constant $\varepsilon_z = 0.001$. The displacement field is expressed as

$$u_x = -\nu\varepsilon_z x = -0.0003x, \tag{8.9a}$$

$$u_z = -\nu\varepsilon_z y = -0.0003y, \tag{8.9b}$$

$$u_z = \varepsilon_z z = 0.001z, \tag{8.9c}$$

where x, y and z are the coordinates of a point. Only the normal stress in z direction is non-zero in the matrix

$$\sigma_{mz} = E_m\varepsilon_z = 1\text{Pa}. \tag{8.10}$$

The magnitude of the tractions on the top and bottom surfaces is $t = 1$Pa. The stress

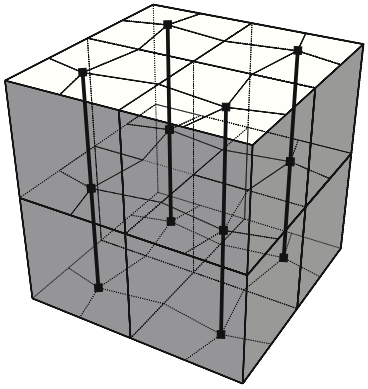


Figure 8.13: The meshes of the matrix and the bars

in the bars is

$$\sigma_{fz} = E_f \varepsilon_z = 1000\text{Pa}. \tag{8.11}$$

The axial forces in the bars are

$$F = \pi r^2 \sigma_{fz} = 0.1\pi\text{N}. \tag{8.12}$$

The square is discretized into 8 hexahedron subdomains. Each of the bars is discretized into 2 segments. Each bar is intersected by 3 faces in the mesh of the matrix. Those faces are divided into new faces and the bars are embedded into the matrix. The meshes of the matrix and the bars are shown in Fig. 8.13.

The deformation of the matrix and the bars is shown in Fig. 8.14. The vertical displacement of the matrix is plotted in color with transparency. The displacement of the bars is plotted in solid color. It is observed that the displacement is linearly distributed in the model. The relative error norm is 7.4534×10^{-14}.

8.3.3 A parametric study

In this example a parametric study is performed to investigate the influence of fiber distribution on the overall elastic properties of a fiber reinforced material. Three pa-

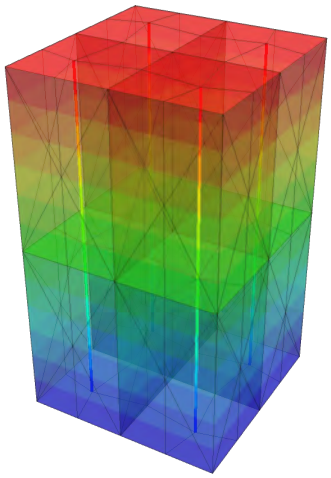


Figure 8.14: The vertical displacement of the matrix and the bars. The deformation is scaled by 500.

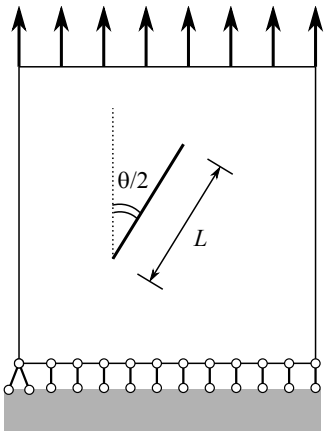
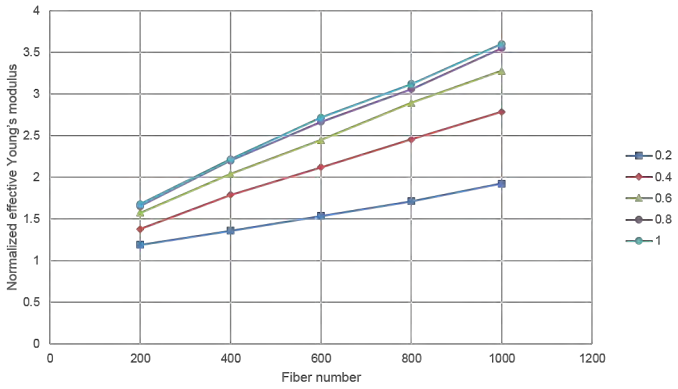


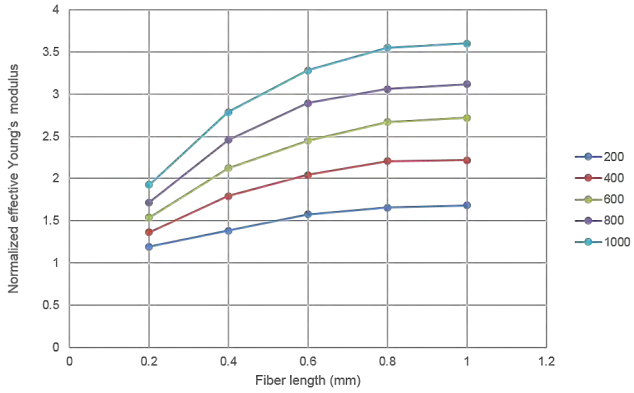
Figure 8.15: Geometry of the fiber reinforced material

rameters are considered in this example, the fiber number N, the fiber length L and the fiber distribution angle θ (Fig. 8.15). The size of the sample is 2.5×2.5mm. The range of fiber number N is between 200 and 1000. The length L varies between 0.2mm and 1mm. The fibers are randomly distributed. The minimum distribution angle is 30° while the maximum angle is 180°.

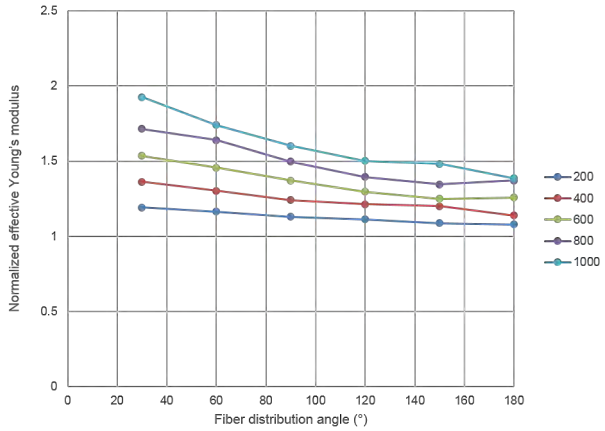
The numerical result of the effective Young's modulus is compared with Halpin-Tsai equation. The effective Young's modulus in the longitudinal direction E_{11} is calculated



(a) Relation between effective Young's modulus and fiber number



(b) Relation between effective Young's modulus and fiber length



(c) Relation between effective Young's modulus and fiber distribution angle

Figure 8.16: Homogenization of fiber reinforced material with different fiber distributions

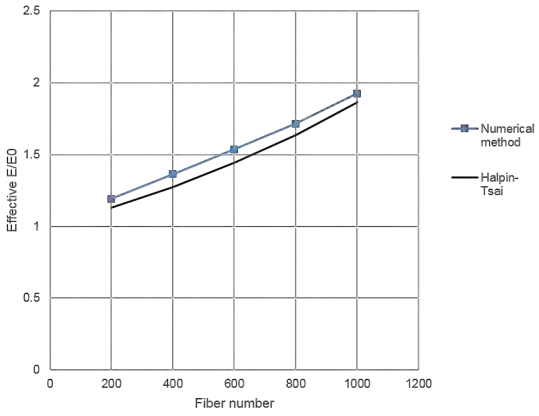


Figure 8.17: Comparison between numerical result of 180° fiber distribution angle and Halpin-Tsai equation

as

$$\frac{E_{11}}{E_m} = \frac{1 + \eta v_f}{1 - \eta v_f}, \tag{8.13}$$

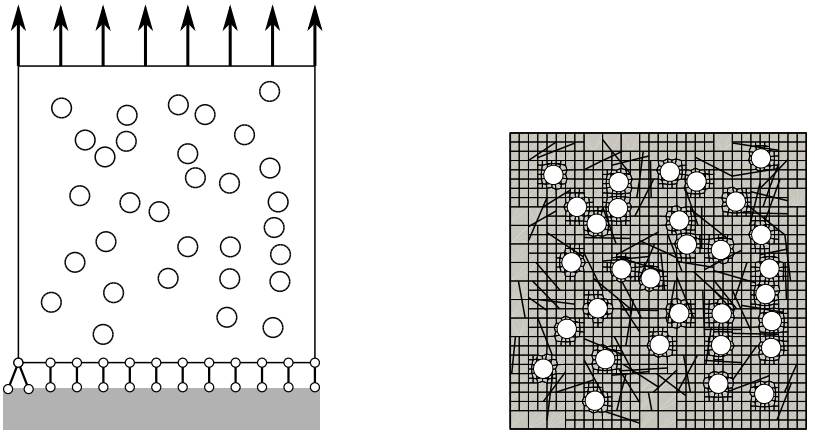
where E_m is the Young's modulus of the matrix, v_f is the volume fraction of the fiber, and η is calculated as

$$\eta = \frac{\frac{E_f}{E_m} - 1}{\frac{E_f}{E_m} + \frac{2L}{d}}, \tag{8.14}$$

where E_f is the Young's modulus of the fiber, L and d are the length and diameter of the fiber. As Halpin-Tsai equation only considers 180° fiber distribution angle, one curve in the numerical result is selected for comparison in Fig. 8.17. It is observed that the numerical result agrees with the prediction of Halpin-Tsai equation well.

8.3.4 A fiber reinforced porous material

In this example, a piece of porous material reinforced by fibers is considered (Fig. 8.18a). Plane stress condition is considered. The dimension of the matrix is 60×60mm. The diameter of the pores is 4mm. There are 31 pores in the matrix. The porosity of the material is $\phi = 10.8\%$. 87 fibers are randomly distributed in the porous matrix. The length of the fibers is $L = 10$mm and diameter $d = 1$mm. A quadtree mesh is



(a) The geometry and boundary conditions of the porous material

(b) The mesh of the porous material

Figure 8.18: The geometry and mesh of the porous material

generated for the matrix with the region near the pores refined (Fig. 8.18b). The fibers are embedded into the matrix. A uniform tension load is applied on the top surface of the matrix. Young's modules and Poisson's ratio of the concrete are $E = 1.7 \times 10^{10}$Pa and $\nu = 0.3$. The Young's modules of the fibers is $E = 2.1 \times 10^{11}$Pa.

The vertical displacement of the porous material is plotted in the deformed mesh in Fig. 8.19.

8.3.5 A fiber reinforced concrete

In this example, a fiber reinforced concrete specimen is considered. The model is represented by a digital image of the concrete (Fig. 8.20a). The dimension of the matrix is 256×256mm. There are two materials in the concrete, the mortar (grey) and the aggregate (black). Fibers are randomly distributed in the mortar. The length of the fibers is $L = 40$mm and diameter $d = 1$mm. A quadtree mesh is generated based on the image (Fig. 8.20b). The mesh is refined on the interface between the mortar and the aggregate. The fibers are embedded into the matrix. A uniform tension load is applied on the top surface of the matrix. Young's modules and Poisson's ratio of the

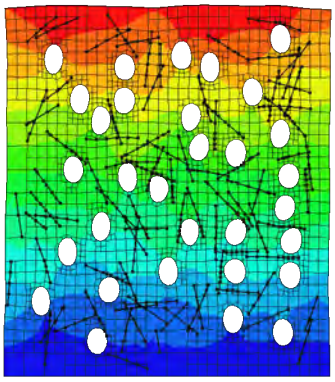
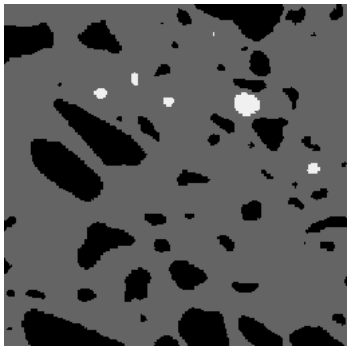
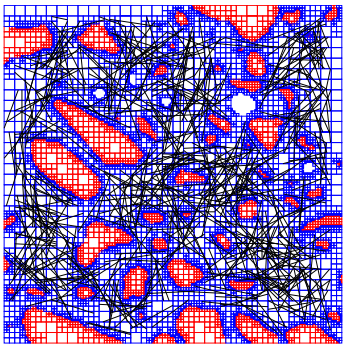


Figure 8.19: The displacement of the porous material



(a) The geometry and boundary conditions of the concrete



(b) The mesh of the concrete

Figure 8.20: The geometry and mesh of the concrete

aggregate are $E = 5.5 \times 10^{10}$Pa and $\nu = 0.25$. Young's modules and Poisson's ratio of the mortar are $E = 1 \times 10^{10}$Pa and $\nu = 0.2$. The Young's modules of the fibers is $E = 2.1 \times 10^{11}$Pa.

The deformation of the concrete is shown in Fig. 8.19.

8.3.6 A carbon nanotube (CNT) reinforced polymer

In this example, a CNT reinforced polymer is considered. A similar problem can be found in Guo et al. (2018) and Banerjee et al. (2016). The CNTs are modeled as groups of hexagons in Fig. 8.22a. The dimension of the matrix is 30×30nm. The edge length

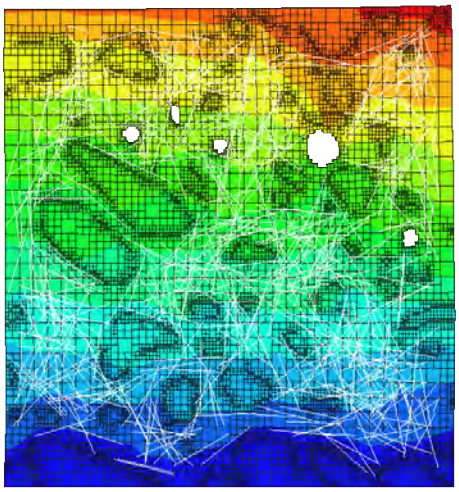
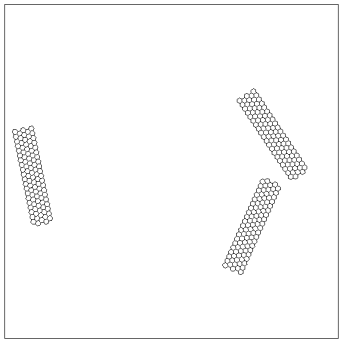
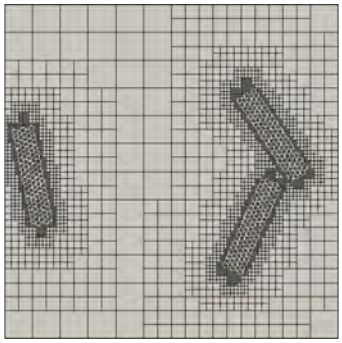


Figure 8.21: The vertical displacement of the concrete



(a) The geometry and boundary conditions of the CNT reinforced polymer



(b) The mesh of the CNT reinforced polymer

Figure 8.22: The geometry and mesh of the CNT reinforced polymer

of the hexagons in the CNTs is $a = 0.246$nm and diameter $d = 0.1$nm. A quadtree mesh is generated with a $1:16$ mesh size ratio (Fig. 8.22b). The CNTs are embedded into the matrix. A uniform tension load is applied on the top surface of the matrix. Young's modules and Poisson's ratio of the polymer are $E = 3.4 \times 10^9$Pa and $\nu = 0.3$. The Young's modules of the CNT is $E = 1 \times 10^{12}$Pa.

The vertical displacement of the CNT reinforced polymer is plotted on the deformed mesh in Fig. 8.23.

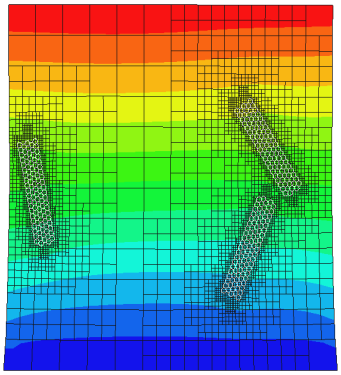


Figure 8.23: The vertical displacement of the CNT reinforced polymer

Chapter 9

Conclusions and recommendations

In this chapter, conclusions of the previous chapters and recommendations for the future work are presented.

9.1 Conclusions

In this thesis, a numerical framework was developed to link geometric modeling and structure analysis. The objective of this research was to automate the process converting geometric models to numerical models for analysis, which took significant amount of time in the engineering design. This framework was based on the SBFEM, which is a semi-analytical method requiring the discretization of the boundary only. In the SBFEM, polygons/polyhedrons with arbitrary number of nodes, edges and faces can be utilized as elements, which offers greater flexibility in mesh generation and domain decomposition.

The following developments were completed: 1) An octree based mesh generation method was proposed, which is highly complementary to the SBFEM. The geometric models represented by a variety of data formats can be converted to numerical models seamlessly with minimum human intervention. 2) A technique to handle non-matching meshes was developed. Complex and large scale models can be divided into smaller

models with simpler shapes and meshed independently, which further reduces the difficulty in mesh generation. These individual meshes can be merged using arbitrary polyhedron elements formulated in the SBFEM. 3) An adaptive analysis procedure, including error estimation and local mesh refinement, was established. The analysis performed with minimum remeshing starts from a coarse mesh. The mesh is refined iteratively in the regions with high estimated errors. The computational resources can be used efficiently. 4) Numerical simulation using virtual reality models was investigated. It can be applied to large scale models to facilitate different tasks such as urban planning, hazard simulation, structural health monitoring, etc. The numerical model is reusable when local modifications of the virtual reality model are performed. 5) A discrete modeling method for fiber reinforced material was proposed. It is a simple but effective approach to evaluate the mechanical properties of fiber reinforced materials with complex fiber distributions. The summaries of each chapter will be presented in the following of this section.

The proposed method based on the SBFEM utilizes polygon/polyhedron element in the analysis, therefore reduces the difficulty in mesh generation in conventional FEM. It offers more flexibility than FDM when the geometry of the model is complex. Comparing with BEM, no fundamental solution is required. The proposed method doesn't need node enrichment as in XFEM. The singular stress field near the crack tip can be represented by the shape function directly. IGA is mostly limited to NURBS based models. The proposed approach is more versatile in the analysis of models represented by STL format and digital image. The shape function in SBFEM constructs an exact partition of unity, therefore patch tests can be successfully passed while the meshfree method usually fails.

9.1.1 Octree based polyhedron mesh generation

An automatic polyhedron mesh generation method was proposed based on the octree algorithm in Section 4. The octree background mesh was refined considering several different refinement criteria. Users were able to specify different mesh size transition rate. Octree and polyhedron data structures were utilized to store and process the mesh data in different stages. A trimming procedure following the bottom-up scheme was implemented for the polyhedron cells on the boundary. Special treatments were designed to trim the cells near the sharp ridges and corners using the information extracted from the boundary model. An optimization algorithm using Laplacian smoothing was designed to improve the element quality near the boundary. Only the outmost two layers of the nodes were relocated by smoothing, therefore the octree mesh structure inside the model was maintained. The smoothed polyhedron cells can be used in analysis with only their boundaries discretized. Two different discretization schemes were developed to obtain best element quality. One of the schemes inserted Steiner points on the boundary and another did not. The scaling centers were located by generating a set of test points and checking their visibility angles.

The versatility and efficiency of the method were demonstrated using six numerical examples. It was shown that high quality meshes can be automatically generated for models with complex geometries. The theoretical convergence rates for linear and quadratic elements were achieved.

9.1.2 Coupling of non-matching meshes

In Section 5 a coupling method of non-matching meshes was presented using arbitrary polyhedron elements based on the SBFEM. On the interface, a surface mesh of polygon elements was constructed by merging the non-matching meshes. The volume elements on the interface were updated by replacing their faces on the interface with the new constructed polygon elements, leading to a matching discretization on the interface. The

updated polyhedron elements were formulated by the SBFEM. No interface constraints or special shape functions were required in the analysis. A polytree refinement algorithm was applied to produce a smooth transition of element size. A shifting procedure was designed to prevent short edges and distorted elements. The method was simple, efficient and versatile in handling flat and curved interface meshes of different types of elements.

The proposed method was illustrated using seven numerical examples. High order patch tests of non-matching meshes ware successfully passed due to the compatibility on the interface. Convergence studies and examples with complex geometries were presented.

9.1.3 Adaptive analysis

3D adaptive analysis using the SBFEM was developed in Section 6. A simple error indicator based on discontinuity of the stress field was proposed. The stresses on an interior face were calculated from the two polyhedron cells connected to the face. The difference between the two stresses was used to indicate the error. If the difference was larger than a predefined threshold, the two polyhedron cells connected to the face would be refined. The error was not estimated for the faces on the boundary. A polytree based mesh refinement technique was developed for arbitrary star-convex polyhedron elements. The refinement was performed locally in the interested region. Remeshing of the whole model was not required.

Five numerical examples were presented using this technique. In each example several adaptive iterations were performed. It was observed that the convergence rate of adaptive refinement was faster than the uniform refinement.

9.1.4 Towards engineering analysis of virtual city

In Section 7 a numerical simulation scheme based on virtual reality models of urban scale was presented, which was referred to as virtual city analysis. Techniques for the geometric modeling of buildings in the city using photogrammetry, LiDAR and computed tomography were reviewed. Meshes of different parts were generated independently, such as buildings, soil layers, tunnels, bridges, etc. The meshes were combined in an analysis by modifying their interfaces only. The reusability of the meshes was discussed. The applications of the proposed method were demonstrated using four numerical examples.

9.1.5 Discrete modeling of fiber reinforced material

A discrete modeling method for fiber reinforced material was proposed in Section 8. The matrix material was modeled using polygon elements in 2D and polyhedron elements in 3D. The fibers were modeled using bar elements. The mesh of the matrix was generated without considering the location of fibers. The fibers were attached to the matrix by adding the intersection points to the mesh of the matrix. The elements of the matrix were still polygons/polyhedrons after adding the fibers. Only the boundaries of the polygon/polyhedron elements were discretized, except for the elements near the fiber tips where the volume was discretized to embed the fiber tips. A shifting procedure was designed to improve the mesh quality of the matrix near the fiber tips and the fiber segments. The shifting procedure was implemented for both 2D and 3D meshes.

The method was tested using six numerical examples. The patch tests for linear elements were successfully passed. A parametric study was performed to evaluated the overall material property considering different fiber length, number and distribution angle. The applications in porous material, concrete and nanocomposite were presented.

9.2 Future work

9.2.1 Mesh generation using high order elements

In the present work, the mesh is generated using linear elements. A large number of elements are required in order to represent a curved boundary accurately. Mesh generation using high order elements has attracted significant attention recently (Moxey et al., 2015; Fortunato and Persson, 2016; Fries, 2018). Using high order elements not only reduces the number of elements required to approximate the geometry, but also produces faster convergence rate. However, the research on high order mesh generation based on octree algorithm is still lacking. Therefore, developing a mesh generation method using high order elements is useful in the numerical analysis.

9.2.2 Mesh generation for crack propagation in 3D

In conventional FEM, the modeling of crack is difficult due to the stress singularity on the crack front. The mesh needs to be refined near the crack front to represent the singular stress field. The propagation of the crack will result in frequent remeshing, which is especially challenging in 3D problems. A new mesh generation method for crack propagation can be developed based on the proposed numerical framework. In 3D the crack face is represented by a surface and the crack front is modeled with a line (Fig. 9.1). The octree background mesh will be cut by the crack face. The elements intersected by the crack front will be modeled using open subdomains mentioned in Section 4.7.3. The crack face will propagate based on the result of the analysis, and the background mesh will be cut by the new crack face again. In each step the background mesh can be refined near the crack front.

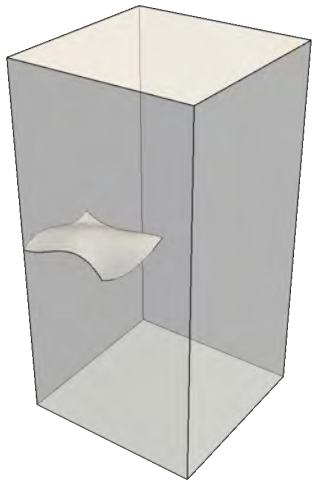


Figure 9.1: Modeling of a crack in 3D

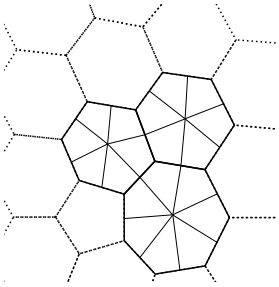
9.2.3 Alternate mesh refinement techniques

In the present polytree mesh refinement method, a polygon with n nodes will be divided into n quadrilaterals (Fig. 9.2a). The quality of the quadrilaterals will degenerate with the increase of n, e.g. greater than 7. A new refinement technique was proposed in Nguyen-Xuan et al. (2017a), which divides a polygon into $n+1$ new polygons, among which are n pentagons on the boundary and one n sided polygon inside (Fig. 9.2b). This scheme usually produces polygons with better quality.

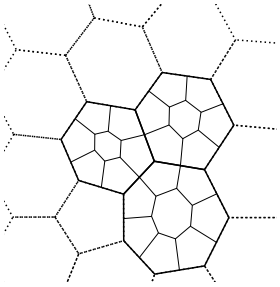
However, this idea has not been extended to 3D. In the present approach, a polyhedron with n nodes is divided into n new polyhedrons. In the future, the polyhedron can be divided into $n+1$ new polyhedrons. The additional polyhedron will be located inside the original polyhedron. The number of faces of the new polyhedron will be equal to the number of nodes of the original polyhedron.

9.2.4 Modeling of curved fibers

Curved fibers have been widely used in engineering (Silani et al., 2014; Khudari Bek et al., 2018). The present approach can be easily extended to the modeling of curved fibers by modifying the intersection calculation and shifting procedure. The fibers will



(a) Polygons divided into n quadrilaterals



(b) Polygons divided into $n+1$ polygons

Figure 9.2: Comparison between different polytree refinement techniques

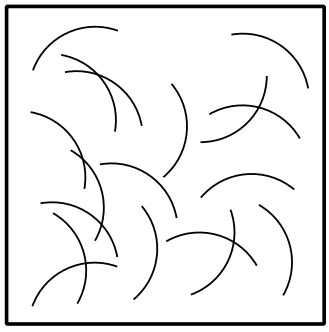


Figure 9.3: Curved fibers

be represented by arcs (Fig. 9.3). The intersection points between the arcs and the background mesh will be calculated and inserted to the background mesh. The nodes in the background mesh will be shifted to the arcs to prevent short edges and distorted elements.

9.2.5 Fracture analysis of fiber reinforced material

In the present work it is assumed that the fibers and the matrix are perfect bonded. In engineering practice, the slippage of the fibers often needs to be considered if the matrix is cracked, as shown in Fig. 9.4. The present discrete modeling method can be applied to fracture analysis of fiber reinforced material by introducing bond-slip model between the fibers and the matrix.

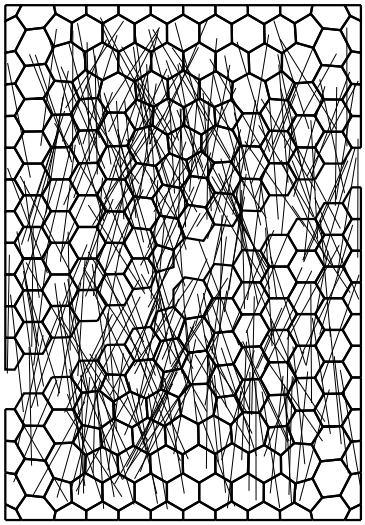


Figure 9.4: Fiber reinforced material with crack

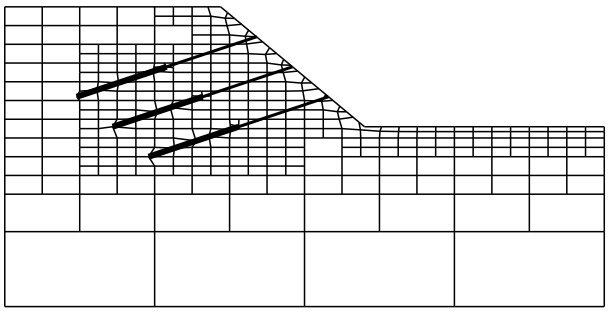


Figure 9.5: Anchor reinforced slope

9.2.6 Elasto-plastic analysis of anchor reinforced slope

The anchor reinforced slope can also be considered as a special fiber reinforced material. The treatment of the fibers can be applied to the anchors with a simple modification. The anchors will be divided into the free segments and the grouted segments (Fig. 9.5). The grouted segment is embedded to the mesh of the soil. As for the free segment, only the two end points are connected to the soil. The stability of the slope can be studied considering the plasticity of the soil.

Bibliography

Abdelaziz, Y., Hamouine, A., 2008. A survey of the extended finite element. Computers & Structures 86, 1141–1151.

Affdl, J. H., Kardos, J., 1976. The Halpin-Tsai equations: a review. Polymer Engineering & Science 16, 344–352.

Agathos, K., Chatzi, E., Bordas, S. P., Talaslidis, D., 2016. A well-conditioned and optimally convergent XFEM for 3D linear elastic fracture. International Journal for Numerical Methods in Engineering 105, 643–677.

Agathos, K., Ventura, G., Chatzi, E., Bordas, S. P., 2018. Stable 3D XFEM/vector level sets for non-planar 3D crack propagation and comparison of enrichment schemes. International Journal for Numerical Methods in Engineering 113, 252–276.

Ahmad, S., Banerjee, P. K., 1986. Free vibration analysis by BEM using particular integrals. Journal of Engineering Mechanics 112, 682–695.

Ainsworth, M., Oden, J. T., 1993. A unified approach to a posteriori error estimation using element residual methods. Numerische Mathematik 65, 23–50.

Apostolatos, A., Schmidt, R., Wuchner, R., Bletzinger, K.-U., 2014. A Nitsche-type formulation and comparison of the most common domain decomposition methods in isogeometric analysis. International Journal for Numerical Methods in Engineering 97, 473–504.

Argyris, J. H., 1955. Energy theorems and structural analysis: a generalized discourse with applications on energy principles of structural analysis including the effects of temperature and non-linear stress-strain relations Part I. general theory. Aircraft Engineering and Aerospace Technology 27, 42–58.

Armstrong, C. G., Fogg, H. J., Tierney, C. M., Robinson, T. T., 2015. Common themes in multi-block structured quad/hex mesh generation. Procedia Engineering 124, 70–82.

Aurada, M., Feischl, M., Fuhrer, T., Karkulik, M., Melenk, J. M., Praetorius, D., 2013. Classical FEM-BEM coupling methods: nonlinearities, well-posedness, and adaptivity. Computational Mechanics 51, 399–419.

Auricchio, F., da Veiga, L. B., Buffa, A., Lovadina, C., Reali, A., Sangalli, G., 2007. A fully "locking-free" isogeometric approach for plane linear elasticity problems: a stream function formulation. Computer Methods in Applied Mechanics and Engineering 197, 160–172.

Avizo, 2015. Avizo User's Guide. 2nd Edition.

Ayatollahi, M. R., Shadlou, S., Shokrieh, M. M., 2011. Multiscale modeling for mechanical properties of carbon nanotube reinforced nanocomposites subjected to different types of loading. Composite Structures 93, 2250–2259.

Babuska, I., Melenk, J. M., 1997. The partition of unity method. International Journal for Numerical Methods in Engineering 40, 727–758.

Babuska, I., Nobile, F., Tempone, R., 2007. A stochastic collocation method for elliptic partial differential equations with random input data. SIAM Journal on Numerical Analysis 45, 1005–1034.

Babuska, I., Strouboulis, T., Upadhyay, C., Gangaraj, S., Copps, K., 1994. Validation of a posteriori error estimators by numerical approach. International Journal for Numerical Methods in Engineering 37, 1073–1123.

Baker, T. J., 1989. Automatic mesh generation for complex three-dimensional regions using a constrained Delaunay triangulation. Engineering with Computers 5, 161–175.

Banerjee, D., Nguyen, T., Chuang, T.-J., 2016. Mechanical properties of single-walled carbon nanotube reinforced polymer composites with varied interphase's modulus and thickness: a finite element analysis study. Computational Materials Science 114, 209–218.

Barzegar, F., Maddipudi, S., 1997. Three-dimensional modeling of concrete structures. II: reinforced concrete. Journal of Structural Engineering 123, 1347–1356.

Bazilevs, Y., Calo, V., Cottrell, J., Hughes, T., Reali, A., Scovazzi, G., 2007. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Computer Methods in Applied Mechanics and Engineering 197, 173–201.

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

Bazilevs, Y., Calo, V. M., Hughes, T. J., Zhang, Y., 2008. Isogeometric fluid-structure interaction: theory, algorithms, and computations. Computational Mechanics 43, 3–37.

Bazilevs, Y., Calo, V. M., Zhang, Y., Hughes, T. J., 2006. Isogeometric fluid–structure interaction analysis with applications to arterial blood flow. Computational Mechanics 38, 310–322.

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

Bazyar, M. H., Song, C., 2008. A continued-fraction-based high-order transmitting boundary for wave propagation in unbounded domains of arbitrary geometry. International Journal for Numerical Methods in Engineering 74, 209–237.

Bechet, E., Cuilliere, J.-C., Trochu, F., 2002. Generation of a finite element mesh from stereolithography (STL) files. Computer-Aided Design 34, 1–17.

Becker, R., Hansbo, P., Stenberg, R., 2003. A finite element method for domain decomposition with non-matching grids. ESAIM: Mathematical Modelling and Numerical Analysis 37, 209–225.

Beese, S., Loehnert, S., Wriggers, P., 2018. 3D ductile crack propagation within a polycrystalline microstructure using XFEM. Computational Mechanics 61, 71–88.

Beirao da Veiga, L., Brezzi, F., Marini, L. D., Russo, A., 2014. The hitchhiker's guide to the virtual element method. Mathematical Models and Methods in Applied Sciences 24, 1541–1573.

Belgacem, F. B., 1999. The mortar finite element method with Lagrange multipliers. Numerische Mathematik 84, 173–197.

Belytschko, T., Black, T., 1999. Elastic crack growth in finite elements with minimal remeshing. International Journal for Numerical Methods in Engineering 45, 601–620.

Belytschko, T., Gracie, R., 2007. On XFEM applications to dislocations and interfaces. International Journal of Plasticity 23, 1721–1738.

Belytschko, T., Lu, Y. Y., Gu, L., 1994. Element-free Galerkin methods. International Journal for Numerical Methods in Engineering 37, 229–256.

Benveniste, Y., 1987. A new approach to the application of Mori-Tanaka's theory in composite materials. Mechanics of Materials 6, 147–157.

Berg, M. D., Cheong, O., Kreveld, M. V., Overmars, M., 2008. Computational Geometry: Algorithms and Applications. Springer-Verlag TELOS.

Bin, W., Ri-jing, P., Zhi-qiang, Y., Shan-chao, Y., Xiao-qing, F., Zhi-geng, P., 2008. Watermarking T-spline surfaces. In: 2008 11th IEEE International Conference on Communication Technology. IEEE, pp. 773–776.

Bitencourt Jr., L. A. G., Manzoli, O. L., Prazeres, P. G., Rodrigues, E. A., Bittencourt, T. N., 2015. A coupling technique for non-matching finite element meshes. Computer Methods in Applied Mechanics and Engineering 290, 19–44.

Blacker, T., 1996. The cooper tool. In: 5th International Meshing Roundtable, SAND 95-2130, Sandia National Laboratories. Citeseer.

Blacker, T. D., Meyers, R. J., 1993. Seams and wedges in plastering: a 3-D hexahedral mesh generation algorithm. Engineering with Computers 9, 83–93.

Blacker, T. D., Stephenson, M. B., 1991. Paving: A new approach to automated quadrilateral mesh generation. International Journal for Numerical Methods in Engineering 32, 811–847.

Blocken, B., 2015. Computational fluid dynamics for urban physics: importance, scales, possibilities, limitations and ten tips and tricks towards accurate and reliable simulations. Building and Environment 91, 219–245.

Bolander, J. E., Choi, S., Duddukuri, S. R., 2008. Fracture of fiber-reinforced cement composites: effects of fiber dispersion. International Journal of Fracture 154, 73–86.

Bols, J., Taelman, L., De Santis, G., Degroote, J., Verhegghe, B., Segers, P., Vierendeels, J., 2016. Unstructured hexahedral mesh generation of complex vascular trees

using a multi-block grid-based approach. Computer Methods in Biomechanics and Biomedical Engineering 19, 663–672.

Bonet, J., Kulasegaram, S., 2000. Correction and stabilization of smooth particle hydrodynamics methods with applications in metal forming simulations. International Journal for Numerical Methods in Engineering 47, 1189–1214.

Boroomand, B., Zienkiewicz, O. C., 1999. Recovery procedures in error estimation and adaptivity. Part II: adaptivity in nonlinear problems of elasto-plasticity behaviour. Computer Methods in Applied Mechanics and Engineering 176, 127–146.

Botella, A., Levy, B., Caumon, G., 2016. Indirect unstructured hex-dominant mesh generation using tetrahedra recombination. Computational Geosciences 20, 437–451.

Bouhala, L., Shao, Q., Koutsawa, Y., Younes, A., Nunez, P., Makradi, A., Belouettar, S., 2013. An XFEM crack-tip enrichment for a crack terminating at a bi-material interface. Engineering Fracture Mechanics 102, 51–64.

Bouillard, P., Ihlenburg, F., 1999. Error estimation and adaptivity for the finite element method in acoustics: 2D and 3D applications. Computer Methods in Applied Mechanics and Engineering 176, 147–163.

Bower, A. F., 2009. Applied Mechanics of Solids. CRC press.

Brebbia, C., Dominguez, J., 1977. Boundary element methods for potential problems. Applied Mathematical Modelling 1, 372–378.

Brebbia, C. A., Telles, J. C. F., Wrobel, L. C., 2012. Boundary Element Techniques: Theory and Applications in Engineering. Springer Science & Business Media.

Brebbia, C. A., Walker, S., 2016. Boundary Element Techniques in Engineering. Elsevier.

Brezzi, F., Bathe, K. J., 1990. A discourse on the stability conditions for mixed finite element formulations. Computer Methods in Applied Mechanics and Engineering 82, 27–57.

Burgarelli, D., Kischinhevsky, M., Biezuner, R. J., 2006. A new adaptive mesh refinement strategy for numerically solving evolutionary PDE's. Journal of Computational and Applied Mathematics 196, 115–131.

Burman, E., Fernandez, M. A., 2014. An unfitted nitsche method for incompressible fluid-structure interaction using overlapping meshes. Computer Methods in Applied Mechanics and Engineering 279, 497–514.

Canu, P., Ray, W., 1991. Discrete weighted residual methods applied to polymerization reactions. Computers & Chemical Engineering 15, 549–564.

Carrer, J., Telles, J., 1992. A boundary element formulation to solve transient dynamic elastoplastic problems. Computers & Structures 45, 707–713.

Chan, S., Tuba, I., Wilson, W., 1970. On the finite element method in linear fracture mechanics. Engineering Fracture Mechanics 2, 1–17.

Chau, K. N., Chau, K. N., Ngo, T., Hackl, K., Nguyen-Xuan, H., 2018. A polytree-based adaptive polygonal finite element method for multi-material topology optimization. Computer Methods in Applied Mechanics and Engineering 332, 712–739.

Chen, J.-S., Wu, C.-T., Yoon, S., You, Y., 2001. A stabilized conforming nodal integration for Galerkin mesh-free methods. International Journal for Numerical Methods in Engineering 50, 435–466.

Chen, K., Zou, D., Kong, X., Yu, X., 2018. An efficient nonlinear octree SBFEM and its application to complicated geotechnical structures. Computers and Geotechnics 96, 226–245.

Cheng, A. H.-D., Cheng, D. T., 2005. Heritage and early history of the boundary element method. Engineering Analysis with Boundary Elements 29, 268–302.

Chiong, I., Ooi, E. T., Song, C., Tin-Loi, F., 2014. Scaled boundary polygons with application to fracture analysis of functionally graded materials. International Journal for Numerical Methods in Engineering 98, 562–589.

Clough, R. W., 1960. The finite element method in plane stress analysis. In: Proceedings of 2nd ASCE Conference on Electronic Computation, Pittsburgh Pa., Sept. 8 and 9, 1960.

Cottrell, J., Hughes, T., Reali, A., 2007. Studies of refinement and continuity in isogeometric structural analysis. Computer Methods in Applied Mechanics and Engineering 196, 4160–4183.

Cottrell, J. A., Reali, A., Bazilevs, Y., Hughes, T. J., 2006. Isogeometric analysis of structural vibrations. Computer Methods in Applied Mechanics and Engineering 195, 5257–5296.

Courant, R., Friedrichs, K., Lewy, H., 1928. On the partial difference equations of mathematical physics. Mathematische Annalen 100, 32–74.

de Boer, A., Van Zuijlen, A., Bijl, H., 2007. Review of coupling methods for non-matching meshes. Computer Methods in Applied Mechanics and Engineering 196, 1515–1525.

De Lorenzis, L., Wriggers, P., Hughes, T. J., 2014. Isogeometric contact: a review. GAMM-Mitteilungen 37, 85–123.

Deeks, A. J., Wolf, J. P., 2002a. Stress recovery and error estimation for the scaled boundary finite-element method. International Journal for Numerical Methods in Engineering 54, 557–583.

Deeks, A. J., Wolf, J. P., 2002b. A virtual work derivation of the scaled boundary finite-element method for elastostatics. Computational Mechanics 28, 489–504.

Delaunay, B., et al., 1934. Sur la sphere vide. Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk 7, 1–2.

Demargne, A. A., Evans, R., Tiller, P., Dawes, W. N., 2014. Practical and reliable mesh generation for complex, real-world geometries. In: 52nd Aerospace Sciences Meeting. p. 0119.

Demkowicz, L., Devloo, P., Oden, J. T., 1985. On an h-type mesh-refinement strategy based on minimization of interpolation errors. Computer Methods in Applied Mechanics and Engineering 53, 67–89.

Dewanckele, J., De Kock, T., Fronteau, G., Derluyn, H., Vontobel, P., Dierick, M., Van Hoorebeke, L., Jacobs, P., Cnudde, V., 2014. Neutron radiography and X-ray computed tomography for quantifying weathering and water uptake processes inside porous limestone used as building material. Materials Characterization 88, 86–99.

Dilts, G. A., 1999. Moving-least-squares-particle hydrodynamics-I. consistency and stability. International Journal for Numerical Methods in Engineering 44, 1115–1155.

Djebara, Y., El Moumen, A., Kanit, T., Madani, S., Imad, A., 2016. Modeling of the effect of particles size, particles distribution and particles number on mechanical properties of polymer-clay nano-composites: Numerical homogenization versus experimental results. Composites Part B: Engineering 86, 135–142.

Dolbow, J., Moes, N., Belytschko, T., 2000. Discontinuous enrichment in finite elements with a partition of unity method. Finite Elements in Analysis and Design 36, 235–260.

Dollner, J., Baumann, K., Buchholz, H., 2006. Virtual 3D city models as foundation of complex urban information spaces. In: 11th International Conference on Urban Planning and Spatial Development for the Information Society. pp. 107–112.

Dorfel, M. R., Juttler, B., Simeon, B., 2010. Adaptive isogeometric analysis by local h-refinement with T-splines. Computer Methods in Applied Mechanics and Engineering 199, 264–275.

Du Plessis, A., Olawuyi, B. J., Boshoff, W. P., Le Roux, S. G., 2016. Simple and fast porosity analysis of concrete using X-ray computed tomography. Materials and Structures 49, 553–562.

Dylla, K., Frischer, B., Muller, P., Ulmer, A., Haegler, S., 2008. Rome reborn 2.0: A case study of virtual city reconstruction using procedural modeling techniques. Computer Graphics World 16.

E. Griffith, B., Luo, X., 2017. Hybrid finite difference/finite element immersed boundary method. International Journal for Numerical Methods in Biomedical Engineering 33, e2888.

Ennis, G., Lindsay, M., 2000. VRML possibilities: the evolution of the Glasgow model. IEEE MultiMedia 7, 48–51.

Eppstein, D., 1994. Approximating the minimum weight Steiner triangulation. Discrete & Computational Geometry 11, 163–191.

Farrashkhalvat, M., Miles, J., 2003. Basic Structured Grid Generation: With an Introduction to Unstructured Grid Generation. Elsevier.

Feng, L., Alliez, P., Buse, L., Delingette, H., Desbrun, M., 2018. Curved optimal Delaunay triangulation. ACM Transactions on Graphics (TOG) 37, 61.

Figiel, L., Buckley, C. P., 2009. Elastic constants for an intercalated layered-silicate/polymer nanocomposite using the effective particle concept - a parametric study using numerical and analytical continuum approaches. Computational Materials Science 44, 1332–1343.

Finlayson, B., Scriven, L., 1966. The method of weighted residuals-a review. Appl. Mech. Rev 19, 735–748.

Finlayson, B. A., 2013. The Method of Weighted Residuals and Variational Principles. Vol. 73. SIAM.

Flemisch, B., Puso, M. A., Wohlmuth, B. I., 2005. A new dual mortar method for curved interfaces: 2D elasticity. International Journal for Numerical Methods in Engineering 63, 813–832.

Fletcher, C. A., 1984. Computational Galerkin methods. In: Computational Galerkin Methods. Springer, pp. 72–85.

Forsythe, G., Wasow, W. R., 1960. Finite-Difference Methods for Partial Differential Equations, Applied Mathematical Series. Wiley, New York.

Fortunato, M., Persson, P.-O., 2016. High-order unstructured curved mesh generation using the Winslow equations. Journal of Computational Physics 307, 1–14.

Francis, A., Ortiz-Bernardin, A., Bordas, S. P., Natarajan, S., 2017. Linear smoothed polygonal and polyhedral finite elements. International Journal for Numerical Methods in Engineering 109, 1263–1288.

Fries, T., 2018. Higher-order conformal decomposition FEM (CDFEM). Computer Methods in Applied Mechanics and Engineering 328, 75–98.

Fries, T.-P., Belytschko, T., 2010. The extended/generalized finite element method: an overview of the method and its applications. International Journal for Numerical Methods in Engineering 84, 253–304.

Fries, T.-P., Matthies, H., et al., 2004. Classification and overview of meshfree methods. Technical report informatikbericht-nr.2003-03, Institute of Scientific Computing, Technical University Braunschweig, Braunschweig, Germany.

Fujita, K., Katsushima, K., Ichimura, T., Hori, M., Maddegedara, L., 2016. Octree-based multiple-material parallel unstructured mesh generation method for seismic response analysis of soil-structure systems. Procedia Computer Science 80, 1624–1634.

Gatzke, T. D., Grimm, C. M., 2006. Estimating curvature on triangular meshes. International Journal of Shape Modeling 12, 1–28.

Gaul, L., Schanz, M., 1999. A comparative study of three boundary element approaches to calculate the transient response of viscoelastic solids with unbounded domains. Computer Methods in Applied Mechanics and Engineering 179, 111–123.

George, P. L., Hecht, F., Saltel, E., 1991. Automatic mesh generator with specified boundary. Computer Methods in Applied Mechanics and Engineering 92, 269–288.

Geuzaine, C., Remacle, J.-F., 2009. Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. International Journal for Numerical Methods in Engineering 79, 1309–1331.

Ghisi, I. T., Camata, J. J., Coutinho, A. L., 2014. Impact of tetrahedralization on parallel conforming octree mesh generation. International Journal for Numerical Methods in Fluids 75, 800–814.

Giovannelli, L., Rodenas, J., Navarro-Jimenez, J., Tur, M., 2017. Direct medical image-based finite element modelling for patient-specific simulation of future implants. Finite Elements in Analysis and Design 136, 37–57.

Gordon, W. J., Hall, C. A., 1973. Construction of curvilinear co-ordinate systems and applications to mesh generation. International Journal for Numerical Methods in Engineering 7, 461–477.

Gould, P. L., Feng, Y., 1994. Introduction to Linear Elasticity. Springer.

Grayver, A. V., Kolev, T. V., 2015. Large-scale 3D geoelectromagnetic modeling using parallel adaptive high-order finite element method. Geophysics 80, E277–E291.

Grilli, S. T., Skourup, J., Svendsen, I. A., 1989. An efficient boundary element method for nonlinear water waves. Engineering Analysis with Boundary Elements 6, 97–107.

Guersoy, H. N., 1996. Tetrahedral finite element mesh generation from NURBS solid models. Engineering with Computers 12, 211–223.

Guo, H., Ooi, E., Saputra, A., Yang, Z., Natarajan, S., Ooi, E., Song, C., 2019. A quadtree-polygon-based scaled boundary finite element method for image-based mesoscale fracture modelling in concrete. Engineering Fracture Mechanics.

Guo, Z., Song, L., Chai, G. B., Li, Z., Li, Y., Wang, Z., 2018. Multiscale finite element analyses on mechanical properties of graphene-reinforced composites. Mechanics of Advanced Materials and Structures, 1–8.

Gupta, K., Meek, J., 1996. A brief history of the beginning of the finite element method. International Journal for Numerical Methods in Engineering 39, 3761–3774.

Halpin, J., 1969. Stiffness and expansion estimates for oriented short fiber composites. Journal of Composite Materials 3, 732–734.

Hammoudi, K., Dornaika, F., 2011. A featureless approach to 3D polyhedral building modeling from aerial images. Sensors 11, 228–259.

Hansbo, A., Hansbo, P., 2002. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. Computer Methods in Applied Mechanics and Engineering 191, 5537–5552.

Hansbo, P., 2005. Nitsche's method for interface problems in computational mechanics. GAMM-Mitteilungen 28, 183–206.

Hansbo, P., Hermansson, J., 2003. Nitsche's method for coupling non-matching meshes in fluid-structure vibration problems. Computational Mechanics 32, 134–139.

Hasbestan, J. J., Senocak, I., 2017. A parallel adaptive mesh refinement software for complex geometry flow simulations. In: 23rd AIAA Computational Fluid Dynamics Conference. p. 3301.

Hasbestan, J. J., Senocak, I., 2018. Binarized-octree generation for cartesian adaptive mesh refinement around immersed geometries. Journal of Computational Physics 368, 179–195.

Hbaieb, K., Wang, Q. X., Chia, Y. H. J., Cotterell, B., 2007. Modelling stiffness of polymer/clay nanocomposites. Polymer 48, 901–909.

He, Y., Guo, J., Yang, H., 2019. Image-based numerical prediction for effective thermal conductivity of heterogeneous materials: a quadtree based scaled boundary finite element method. International Journal of Heat and Mass Transfer 128, 335–343.

Heo, J., Jeong, S., Park, H.-K., Jung, J., Han, S., Hong, S., Sohn, H.-G., 2013. Productive high-complexity 3D city modeling with point clouds collected from terrestrial LiDAR. Computers, Environment and Urban Systems 41, 26–38.

Hereth, E. A., Sreenivas, K., Taylor, L. K., Nichols, D. S., 2017. An automatic parallel octree grid generation software with an extensible solver framework and a focus on urban simulation. In: 55th AIAA Aerospace Sciences Meeting. p. 0587.

Hill, R., 1963. Elastic properties of reinforced solids: Some theoretical principles. Journal of the Mechanics and Physics of Solids 11, 357–372.

Ho-Le, K., 1988. Finite element mesh generation methods: a review and classification. Computer-Aided Design 20, 27–38.

Hu, K., Zhang, Y. J., 2016. Centroidal Voronoi tessellation based polycube construction for adaptive all-hexahedral mesh generation. Computer Methods in Applied Mechanics and Engineering 305, 405–421.

Hu, Q., Chouly, F., Hu, P., Cheng, G., Bordas, S. P., 2018. Skew-symmetric Nitsche's formulation in isogeometric analysis: Dirichlet and symmetry conditions, patch coupling and frictionless contact. Computer Methods in Applied Mechanics and Engineering 341, 188–220.

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

Hughes, T. J., Taylor, R. L., Sackman, J. L., Curnier, A., Kanoknukulchai, W., 1976. A finite element method for a class of contact-impact problems. Computer Methods in Applied Mechanics and Engineering 8, 249–276.

Ichimura, T., Fujita, K., Quinay, P. E. B., Maddegedara, L., Hori, M., Tanaka, S., Shizawa, Y., Kobayashi, H., Minami, K., 2015. Implicit nonlinear wave simulation with 1.08T DOF and 0.270T unstructured finite elements to enhance comprehensive earthquake simulation. In: High Performance Computing, Networking, Storage and Analysis, 2015 SC-International Conference for. IEEE, pp. 1–12.

Jayaram, S., Connacher, H. I., Lyons, K. W., 1997. Virtual assembly using virtual reality techniques. Computer-Aided Design 29, 575–584.

Joe, B., 1991. Construction of three-dimensional Delaunay triangulations using local transformations. Computer Aided Geometric Design 8, 123–142.

Kan, H., Duffy, V. G., Su, C.-J., 2001. An internet virtual reality collaborative environment for effective product design. Computers in Industry 45, 197–213.

Kanda, M., Inagaki, A., Miyamoto, T., Gryschka, M., Raasch, S., 2013. A new aerodynamic parametrization for real urban surfaces. Boundary-Layer Meteorology 148, 357–377.

Kang, J., Kim, K., Lim, Y. M., Bolander, J. E., 2014. Modeling of fiber-reinforced cement composites: Discrete representation of fiber pullout. International Journal of Solids and Structures 51, 1970–1979.

Karihaloo, B. L., Xiao, Q., 2003. Modelling of stationary and growing cracks in FE framework without remeshing: a state-of-the-art review. Computers & Structures 81, 119–129.

Karman, S., 2004. Hierarchical unstructured mesh generation. In: 42nd AIAA Aerospace Sciences Meeting and Exhibit. p. 613.

Karman, S., Betro, V., 2008. Parallel hierarchical unstructured mesh generation with general cutting. In: 46th AIAA Aerospace Sciences Meeting and Exhibit. p. 918.

Kastner, M., Haasemann, G., Ulbricht, V., 2011. Multiscale XFEM-modelling and simulation of the inelastic material behaviour of textile-reinforced polymers. International Journal for Numerical Methods in Engineering 86, 477–498.

Katsikadelis, J., Nerantzaki, M., 1999. The boundary element method for nonlinear problems. Engineering Analysis with Boundary Elements 23, 365–373.

Katsushima, K., Fujita, K., Ichimura, T., Hori, M., Maddegedara, L., 2018. Development of octree-based high-quality mesh generation method for biomedical simulation. In: International Conference on Computational Science. Springer, pp. 354–367.

Khudari Bek, Y., Hamdia, K. M., Rabczuk, T., Koenke, C., 2018. Micromechanical model for polymeric nano-composites material based on SBFEM. Composite Structures 194, 516–526.

Kim, H. G., 2002. Interface element method (IEM) for a partitioned system with non-matching interfaces. Computer Methods in Applied Mechanics and Engineering 191, 3165–3194.

Kim, H. G., 2008. Development of three-dimensional interface elements for coupling of non-matching hexahedral meshes. Computer Methods in Applied Mechanics and Engineering 197, 3870–3882.

Kobayashi, Y., 2006. Photogrammetry and 3D city modeling. Digital Architecture and Construction 90, 209.

Kudela, L., Zander, N., Kollmannsberger, S., Rank, E., 2016. Smart octrees: accurately integrating discontinuous functions in 3D. Computer Methods in Applied Mechanics and Engineering 306, 406–426.

Lak, M. R. J., Bazyar, M. H., 2019. A non-linear scaled boundary finite-element analysis applied to geotechnical problems. Geotechnical and Geological Engineering 37, 501–515.

Laursen, T. A., Puso, M. A., Sanders, J., 2012. Mortar contact formulations for deformable-deformable contact: past contributions and new extensions for enriched and embedded interface formulations. Computer Methods in Applied Mechanics and Engineering 205-208, 3–15.

Lehmann, L., Langer, S., Clasen, D., 2006. Scaled boundary finite element method for acoustics. Journal of Computational Acoustics 14, 489–506.

Li, J., Fu, X., Chen, B., Wu, C., Lin, G., 2016. Modeling crack propagation with the extended scaled boundary finite element method based on the level set method. Computers & Structures 167, 50–68.

Li, L.-y., Bettess, P., 1997. Adaptive finite element methods: a review. Applied Mechanics Reviews 50, 581–591.

Li, Q., Popov, V. L., 2018. Boundary element method for normal non-adhesive and adhesive contacts of power-law graded elastic materials. Computational Mechanics 61, 319–329.

Li, S., Liu, W. K., 2002. Meshfree and particle methods and their applications. Applied Mechanics Reviews 55, 1–34.

Lim, J. H., Im, S., Cho, Y. S., 2007. MLS (moving least square)-based finite elements for three-dimensional nonmatching meshes and adaptive mesh refinement. Computer Methods in Applied Mechanics and Engineering 196, 2216–2228.

Lin, G., Han, Z., Li, J., 2013. An efficient approach for dynamic impedance of surface footing on layered half-space. Soil Dynamics and Earthquake Engineering 49, 39–51.

Lin, G., Liu, J., Li, J., Fang, H., September 2011. Scaled boundary finite element approach for waveguide eigenvalue problem. IET Microwaves, Antennas Propagation 5, 1508–1515.

Lin, G., Wang, Y., Hu, Z., 2012. An efficient approach for frequency-domain and time-domain hydrodynamic analysis of dam-reservoir systems. Earthquake Engineering & Structural Dynamics 41, 1725–1749.

Lin, G., Zhang, P., Liu, J., Li, J., 2018. Analysis of laminated composite and sandwich plates based on the scaled boundary finite element method. Composite Structures 187, 579–592.

Liszka, T., Duarte, C., Tworzydlo, W., 1996. hp-meshless cloud method. Computer Methods in Applied Mechanics and Engineering 139, 263–288.

Liszka, T., Orkisz, J., 1980. The finite difference method at arbitrary irregular grids and its application in applied mechanics. Computers & Structures 11, 83–95.

Liu, G., Zhang, J., Y. Lam, K., Li, H., Xu, G., H. Zhong, Z., Y. Li, G., Han, X., 2008. A gradient smoothing method (GSM) with directional correction for solid mechanics problems. Computational Mechanics 41, 457–472.

Liu, G. R., Dai, K. Y., Nguyen, T. T., 2007. A smoothed finite element method for mechanics problems. Computational Mechanics 39, 859–877.

Liu, J., Li, J., Li, P., Lin, G., Xu, T., Chen, L., 2018. New application of the isogeometric boundary representations methodology with SBFEM to seepage problems in complex domains. Computers & Fluids 174, 241–255.

Liu, L., Zhang, J., Song, C., Birk, C., Gao, W., 2019. An automatic approach for the acoustic analysis of three-dimensional bounded and unbounded domains by scaled boundary finite element method. International Journal of Mechanical Sciences 151, 563–581.

Liu, P., Luo, Y., Kang, Z., 2016. Multi-material topology optimization considering interface behavior via XFEM and level set method. Computer Methods in Applied Mechanics and Engineering 308, 113–133.

Liu, S.-S., Gadh, R., 1996. Basic logical bulk shapes (BLOBs) for finite element hexahedral mesh generation. In: 5th International Meshing Roundtable. Citeseer, pp. 291–306.

Liu, X., Xiao, Q., Karihaloo, B. L., 2004. XFEM for direct evaluation of mixed mode sifs in homogeneous and bi-materials. International Journal for Numerical Methods in Engineering 59, 1103–1118.

Liu, Y., Mukherjee, S., Nishimura, N., Schanz, M., Ye, W., Sutradhar, A., Pan, E., Dumont, N., Frangi, A., Saez, A., 2011. Recent advances and emerging applications of the boundary element method. Applied Mechanics Reviews 64, 030802.

Liu, Y., Saputra, A. A., Wang, J., Tin-Loi, F., Song, C., 2017. Automatic polyhedral mesh generation and scaled boundary finite element analysis of STL models. Computer Methods in Applied Mechanics and Engineering 313, 106–132.

Lo, S., 1985. A new mesh generation scheme for arbitrary planar domains. International Journal for Numerical Methods in Engineering 21, 1403–1426.

Lo, S., 1991a. Volume discretization into tetrahedra-I. verification and orientation of boundary surfaces. Computers & Structures 39, 493–500.

Lo, S. H., 1991b. Volume discretization into tetrahedra-II. 3D triangulation by advancing front approach. Computers & Structures 39, 501–511.

Loehnert, S., Mueller-Hoeppe, D., Wriggers, P., 2011. 3D corrected XFEM approach and extension to finite deformation theory. International Journal for Numerical Methods in Engineering 86, 431–452.

Lohner, R., 1996. Progress in grid generation via the advancing front technique. Engineering with Computers 12, 186–210.

Lu, J. H.-C., Quadros, W. R., Shimada, K., 2017. Evaluation of user-guided semi-automatic decomposition tool for hexahedral mesh generation. Journal of Computational Design and Engineering 4, 330–338.

Lu, S., Liu, J., Lin, G., Wang, W., 2016. A new analysis of the complex two-dimensional multilayered anisotropic soil in time domain. Acta Mechanica 227, 2125–2155.

Lucy, L. B., 1977. A numerical approach to the testing of the fission hypothesis. The Astronomical Journal 82, 1013–1024.

Luo, L., Wong, M., Hwu, W.-m., 2010. An effective GPU implementation of breadth-first search. In: Proceedings of the 47th Design Automation Conference. ACM, pp. 52–55.

Maday, Y., Mavriplis, C., Patera, A., 1988. Nonconforming mortar element methods: application to spectral discretizations. Tech. rep., NASA Langley Research Center; Hampton, VA, United States.

Malik, M. R., Zang, T. A., Hussaini, M. Y., 1985. A spectral collocation method for the Navier-Stokes equations. Journal of Computational Physics 61, 64–88.

Marechal, L., 2009. Advances in octree-based all-hexahedral mesh generation: handling sharp features. In: Proceedings of the 18th International Meshing Roundtable. Springer, pp. 65–84.

Maronga, B., Gryschka, M., Heinze, R., Hoffmann, F., Kanani-Suhring, F., Keck, M., Ketelsen, K., Letzel, M. O., Suhring, M., Raasch, S., 2015. The parallelized large-eddy simulation model (PALM) version 4.0 for atmospheric and oceanic flows: model formulation, recent developments, and future perspectives. Geoscientific Model Development 8, 2515.

Marot, C., Pellerin, J., Remacle, J.-F., 2019. One machine, one minute, three billion tetrahedra. International Journal for Numerical Methods in Engineering 117, 967–990.

Marussig, B., Hughes, T. J., 2018. A review of trimming in isogeometric analysis: challenges, data exchange and simulation aspects. Archives of Computational Methods in Engineering 25, 1059–1127.

Mavriplis, D. J., 1995. An advancing front Delaunay triangulation algorithm designed for robustness. Journal of Computational Physics 117, 90–101.

McMorris, H., Kallinderis, Y., 1997. Octree-advancing front method for generation of unstructured surface and volume meshes. AIAA journal 35, 976–984.

Mi, Y., Aliabadi, M., 1994. Three-dimensional crack growth simulation using BEM. Computers & Structures 52, 871–878.

Mingwu, L., Benzley, S. E., Sjaardema, G., Tautges, T., 1996. A multiple source and target sweeping method for generating all-hexahedral finite element meshes. In: Proceedings, 5th International Meshing Roundtable. Vol. 96. Citeseer, pp. 217–225.

Moaveni, S., 2011. Finite Element Analysis Theory and Application with ANSYS, 3/e. Pearson Education India.

Monaghan, J. J., 1982. Why particle methods work. SIAM Journal on Scientific and Statistical Computing 3, 422–433.

Moore, E. F., 1959. The shortest path through a maze. In: Proc. Int. Symp. Switching Theory, 1959. pp. 285–292.

Mori, T., Tanaka, K., 1973. Average stress in matrix and average elastic energy of materials with misfitting inclusions. Acta Metallurgica 21, 571–574.

Mortazavi, B., Bardon, J., Ahzi, S., 2013. Interphase effect on the elastic and thermal conductivity response of polymer nanocomposite materials: 3D finite element study. Computational Materials Science 69, 100–106.

Moxey, D., Green, M., Sherwin, S., Peiro, J., 2015. An isoparametric approach to high-order curvilinear boundary-layer meshing. Computer Methods in Applied Mechanics and Engineering 283, 636–650.

Muravin, B., Turkel, E., 2003. Advance diffraction method as a tool for solution of complex non-convex boundary problems-implementation and practical application. In: Meshfree Methods for Partial Differential Equations. Springer, pp. 307–317.

Nakayama, T., Washizu, K., 1981. The boundary element method applied to the analysis of two-dimensional nonlinear sloshing problems. International Journal for Numerical Methods in Engineering 17, 1631–1646.

Narasimhan, T., Witherspoon, P., 1976. An integrated finite difference method for analyzing fluid flow in porous media. Water Resources Research 12, 57–64.

Natarajan, S., Bordas, S. P., Ooi, E. T., 2015. Virtual and smoothed finite elements: a connection and its application to polygonal/polyhedral finite element methods. International Journal for Numerical Methods in Engineering 104, 1173–1199.

Natarajan, S., Francis, A., Atroshchenko, E., Bordas, S., 2017. A new one point quadrature rule over arbitrary star convex polygon/polyhedron. arXiv preprint arXiv:1707.00399.

Natarajan, S., Ooi, E. T., Chiong, I., Song, C., 2014. Convergence and accuracy of displacement based finite element formulations over arbitrary polygons: Laplace interpolants, strain smoothing and scaled boundary polygon formulation. Finite Elements in Analysis and Design 85, 101–122.

Nguyen, B., Tran, H., Anitescu, C., Zhuang, X., Rabczuk, T., 2016. An isogeometric symmetric Galerkin boundary element method for two-dimensional crack problems. Computer Methods in Applied Mechanics and Engineering 306, 252–275.

Nguyen, V. P., Kerfriden, P., Brino, M., Bordas, S. P., Bonisoli, E., 2014. Nitsche's method for two and three dimensional NURBS patch coupling. Computational Mechanics 53, 1163–1182.

Nguyen-Xuan, H., 2017. A polytree-based adaptive polygonal finite element method for topology optimization. International Journal for Numerical Methods in Engineering 110, 972–1000.

Nguyen-Xuan, H., Nguyen, S. H., Kim, H.-G., Hackl, K., 2017a. An efficient adaptive polygonal finite element method for plastic collapse analysis of solids. Computer Methods in Applied Mechanics and Engineering 313, 1006–1039.

Nguyen-Xuan, H., Nguyen-Hoang, S., Rabczuk, T., Hackl, K., 2017b. A polytree-based adaptive approach to limit analysis of cracked structures. Computer Methods in Applied Mechanics and Engineering 313, 1006–1039.

Nitsche, J., 1971. About a variational principle for solving Dirichlet problems when using non-constrained subdomains. In: Essays from the Mathematical Seminar of the University of Hamburg. Vol. 36. Springer, pp. 9–15.

Odegard, G. M., Clancy, T. C., Gates, T. S., 2005. Modeling of the mechanical properties of nanoparticle/polymer composites. Polymer 46, 553–562.

Oden, J. T., 1990. Historical comments on finite elements. In: A history of scientific computing. ACM, pp. 152–166.

Ooi, E. T., Man, H., Natarajan, S., Song, C., 2015. Adaptation of quadtree meshes in the scaled boundary finite element method for crack propagation modelling. Engineering Fracture Mechanics 144, 101–117.

Ortiz, M., Quigley Iv, J., 1991. Adaptive mesh refinement in strain localization problems. Computer Methods in Applied Mechanics and Engineering 90, 781–804.

Owen, D., Figueiras, J., Damjanic, F., 1983. Finite element analysis of reinforced and prestressed concrete structures including thermal loading. Computer Methods in Applied Mechanics and Engineering 41, 323–366.

Owen, S. J., 1998. A survey of unstructured mesh generation technology. In: IMR. pp. 239–267.

Pandey, P. K., 2015. Non-standard finite difference method for numerical solution of second order linear fredholm integro-differential equations. International Journal of Mathematical Modelling & Computations 5, 259–266.

Pantano, A., Averill, R. C., 2002. A penalty-based finite element interface technology. Computers & Structures 80, 1725–1748.

Pantano, A., Averill, R. C., 2007. A penalty-based interface technology for coupling independently modeled 3D finite element meshes. Finite Elements in Analysis and Design 43, 271–286.

Park, B. Y., Roberts, B. L., Sobolik, S. R., 2017. Construction of hexahedral finite element mesh capturing realistic geometries of a petroleum reserve. Finite Elements in Analysis and Design 135, 68–78.

Pebay, P., Baker, T., 2003. Analysis of triangle quality measures. Mathematics of Computation 72, 1817–1839.

Peraire, J., Morgan, K., Peiro, J., 1990. Unstructured finite element mesh generation and adaptive procedures for CFD. Application of Mesh Generation to complex.

Purschke, F., Schulze, M., Zimmermann, P., 1998. Virtual reality-new methods for improving and accelerating the development process in vehicle styling and design. In: Computer Graphics International, 1998. Proceedings. IEEE, pp. 789–797.

Puso, M. A., 2004. A 3D mortar method for solid mechanics. International Journal for Numerical Methods in Engineering 59, 315–336.

Rabczuk, T., Belytschko, T., 2005. Adaptivity for structured meshfree particle methods in 2D and 3D. International Journal for Numerical Methods in Engineering 63, 1559–1582.

Radtke, F., Simone, A., Sluys, L., 2010. A computational model for failure analysis of fibre reinforced concrete with discrete treatment of fibres. Engineering Fracture Mechanics 77, 597–620.

Razdan, A., Bae, M., 2005. Curvature estimation scheme for triangle meshes using biquadratic bezier patches. Computer-Aided Design 37, 1481–1491.

Richardson Lewis, F., Glazebrook Richard, T., 1911. The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 210, 307–357.

Roca, X., Sarrate, J., 2010. An automatic and general least-squares projection procedure for sweep meshing. Engineering with Computers 26, 391–406.

Ruotsalainen, K., Wendland, W., 1988. On the boundary element method for some nonlinear boundary value problems. Numerische Mathematik 53, 299–314.

Samet, H., 1984. The quadtree and related hierarchical data structures. ACM Comput. Surv. 16, 187–260.

Saputra, A. A., Birk, C., Song, C., 2015. Computation of three-dimensional fracture parameters at interface cracks and notches by the scaled boundary finite element method. Engineering Fracture Mechanics 148, 213–242.

Saputra, A. A., Talebi, H., Tran, D., Birk, C., Song, C., 2017. Automatic image-based stress analysis by the scaled boundary finite element method. International Journal for Numerical Methods in Engineering 109, 697–738.

Schneiders, R., 1996. A grid-based algorithm for the generation of hexahedral element meshes. Engineering with Computers 12, 168–177.

Schroeder, W. J., Shephard, M. S., 1990. A combined octree/Delaunay method for fully automatic 3-D mesh generation. International Journal for Numerical Methods in Engineering 29, 37–55.

Segerlind, L. J., 1976. Applied Finite Element Analysis. Vol. 316. Wiley New York.

Seidel, R., 1986. Constructing higher-dimensional convex hulls at logarithmic cost per face. In: Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing. ACM, pp. 404–413.

Shen, X., Hu, W., Fan, J., 2018. Automatic blade blend modeling and hexahedral mesh regeneration for aircraft engine optimization. Structural and Multidisciplinary Optimization 57, 1345–1355.

Sheng, N., Boyce, M. C., Parks, D. M., Rutledge, G. C., Abes, J. I., Cohen, R. E., 2004. Multiscale micromechanical modeling of polymer/clay nanocomposites and the effective clay particle. Polymer 45, 487–506.

Shephard, M. S., Georges, M. K., 1991. Automatic three-dimensional mesh generation by the finite octree technique. International Journal for Numerical methods in engineering 32, 709–749.

Shewchuk, J. R., 2002. Two discrete optimization algorithms for the topological improvement of tetrahedral meshes. Unpublished manuscript 65.

Shokrieh, M. M., Rafiee, R., 2010. Stochastic multi-scale modeling of CNT/polymer composites. Computational Materials Science 50, 437–446.

Si, H., 2015. TetGen, a Delaunay-based quality tetrahedral mesh generator. ACM Transactions on Mathematical Software (TOMS) 41, 11.

Silani, M., Talebi, H., Ziaei-Rad, S., Kerfriden, P., Bordas, S. P. A., Rabczuk, T., 2014. Stochastic modelling of clay/epoxy nanocomposites. Composite Structures 118, 241–249.

Simo, J. C., Wriggers, P., Taylor, R. L., 1985. A perturbed Lagrangian formulation

for the finite element solution of contact problems. Computer Methods in Applied Mechanics and Engineering 50, 163–180.

Singh, S. P., Jain, K., Mandla, V. R., 2013. Virtual 3D city modeling: techniques and applications. ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 73–91.

Sohn, D., Jin, S., 2015. Polyhedral elements with strain smoothing for coupling hexahedral meshes at arbitrary nonmatching interfaces. Computer Methods in Applied Mechanics and Engineering 293, 92–113.

Song, C., 2005. Evaluation of power-logarithmic singularities, T-stresses and higher order terms of in-plane singular stress fields at cracks and multi-material corners. Engineering Fracture Mechanics 72, 1498–1530.

Song, C., 2018. The Scaled Boundary Finite Element Method: Introduction to Theory and Implementation. John Wiley & Sons.

Song, C., Ooi, E. T., Natarajan, S., 2017. A review of the scaled boundary finite element method for two-dimensional linear elastic fracture mechanics. Engineering Fracture Mechanics.

Song, C., Ooi, E. T., Pramod, A. L., Natarajan, S., 2018. A novel error indicator and an adaptive refinement technique using the scaled boundary finite element method. Engineering Analysis with Boundary Elements 94, 10–24.

Song, C., Wolf, J. P., 1997. The scaled boundary finite-element method-alias consistent infinitesimal finite-element cell method-for elastodynamics. Computer Methods in Applied Mechanics and Engineering 147, 329–355.

Sousa, E., Li, C., 2015. A weighted finite difference method for the fractional diffusion equation based on the Riemann–Liouville derivative. Applied Numerical Mathematics 90, 22–37.

Spanos, P. D., Kontsos, A., 2008. A multiscale monte carlo finite element method for determining mechanical properties of polymer nanocomposites. Probabilistic Engineering Mechanics 23, 456–470.

Spring, D. W., Leon, S. E., Paulino, G. H., 2014. Unstructured polygonal meshes with adaptive refinement for the numerical simulation of dynamic cohesive fracture. International Journal of Fracture 189, 33–57.

Staten, M. L., Canann, S. A., Owen, S. J., 1999. BMSweep: locating interior nodes during sweeping. Engineering with Computers 15, 212–218.

Staten, M. L., Kerr, R. A., Owen, S. J., Blacker, T. D., Stupazzini, M., Shimada, K., 2010a. Unconstrained plastering-hexahedral mesh generation via advancing-front geometry decomposition. International Journal for Numerical Methods in Engineering 81, 135–171.

Staten, M. L., Shepherd, J. F., Ledoux, F., Shimada, K., 2010b. Hexahedral mesh matching: converting non-conforming hexahedral-to-hexahedral interfaces into conforming interfaces. International Journal for Numerical Methods in Engineering 82, 1475–1509.

Su, T., Wang, W., Lv, Z., Wu, W., Li, X., 2016. Rapid Delaunay triangulation for randomly distributed point cloud data using adaptive Hilbert curve. Computers & Graphics 54, 65–74.

Sun, J., Yu, X., Baciu, G., Green, M., 2002. Template-based generation of road networks for virtual city modeling. In: Proceedings of the ACM Symposium on Virtual Reality Software and Technology. ACM, pp. 33–40.

Surazhsky, T., Magid, E., Soldea, O., Elber, G., Rivlin, E., 2003. A comparison of Gaussian and mean curvatures estimation methods on triangular meshes. In: 2003

IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422). Vol. 1. IEEE, pp. 1021–1026.

Tabarraei, A., Sukumar, N., 2005. Adaptive computations on conforming quadtree meshes. Finite Elements in Analysis and Design 41, 686–702.

Taborda, R., Bielak, J., 2011a. Full 3D integration of site-city effects in regional scale earthquake simulations. In: Proc. 8th Int. Conf. Struct. Dyn. EURODYN. pp. 511–518.

Taborda, R., Bielak, J., 2011b. Large-scale earthquake simulation: computational seismology and complex engineering systems. Computing in Science & Engineering 13, 14–27.

Talebi, H., Saputra, A. A., Song, C., 2016. Stress analysis of 3D complex geometries using the scaled boundary polyhedral finite elements. Computational Mechanics 58, 697–715.

Tautges, T. J., Liu, S., Lu, Y., Kraftcheck, J., Gadh, R., 1997. Feature recognition applications in mesh generation. Tech. rep., Sandia National Labs., Albuquerque, NM (United States).

Telles, J., Carrer, J., 1991. Implicit procedures for the solution of elastoplastic problems by the boundary element method. Mathematical and Computer Modelling 15, 303–311.

Telles, J., Carrer, J., 1994. Static and transient dynamic nonlinear stress analysis by the boundary element method with implicit techniques. Engineering Analysis with Boundary Elements 14, 65–74.

Thacker, W., Gonzalez, A., Putland, G., 1980. A method for automating the construction of irregular computational grids for storm surge forecast models. Journal of Computational Physics 37, 371–387.

Thomee, V., 1984. Galerkin Finite Element Methods for Parabolic Problems. Vol. 1054. Springer.

Thomee, V., 2001. From finite differences to finite elements a short history of numerical analysis of partial differential equations. In: Numerical Analysis: Historical Developments in the 20th Century. Elsevier, pp. 361–414.

Thompson, J. F., Warsi, Z. U., Mastin, C. W., 1985. Numerical Grid Generation: Foundations and Applications. Vol. 45. North-holland Amsterdam.

Tserpes, K. I., Chanteli, A., 2013. Parametric numerical evaluation of the effective elastic properties of carbon nanotube-reinforced polymers. Composite Structures 99, 366–374.

Tserpes, K. I., Papanikos, P., Labeas, G., Pantelakis, S. G., 2008. Multi-scale modeling of tensile behavior of carbon nanotube-reinforced composites. Theoretical and Applied Fracture Mechanics 49, 51–60.

Turner, M., 1956. Stiffness and deflection analysis of complex structures. Journal of the Aeronautical Sciences 23, 805–823.

Unger, J. F., Eckardt, S., 2011. Multiscale modeling of concrete. Archives of Computational Methods in Engineering 18, 341.

Vanalli, L., Paccola, R., Coda, H., 2008. A simple way to introduce fibers into FEM models. International Journal for Numerical Methods in Biomedical Engineering 24, 585–603.

Vecchio, F. J., 1989. Nonlinear finite element analysis of reinforced concrete membranes. ACI Structural Journal 86, 26–35.

Vo, V. S., Nguyen, V.-H., Mahouche-Chergui, S., Carbonnier, B., Naili, S., 2018. Estimation of effective elastic properties of polymer/clay nanocomposites: a parametric study. Composites Part B: Engineering 152, 139–150.

Vuong, A.-V., Giannelli, C., Jüttler, B., Simeon, B., 2011. A hierarchical approach to adaptive local refinement in isogeometric analysis. Computer Methods in Applied Mechanics and Engineering 200, 3554–3567.

Watson, D. F., 1981. Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes. The Computer Journal 24, 167–172.

Weatherill, N. P., Hassan, O., 1994. Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints. International Journal for Numerical Methods in Engineering 37, 2005–2039.

White, D. R., Mingwu, L., Benzley, S. E., Sjaardema, G. D., 1995. Automated hexahedral mesh generation by virtual decomposition. In: Proceedings of the 4th International Meshing Roundtable, Sandia National Laboratories, Albuquerque, USA. pp. 165–176.

Wohlmuth, B. I., 2000. A mortar finite element method using dual spaces for the lagrange multiplier. SIAM Journal on Numerical Analysis 38, 989–1012.

Wolf, J. P., Song, C., 2000. The scaled boundary finite-element method-a primer: derivations. Computers & Structures 78, 191–210.

Wu, H., Gao, S., Wang, R., Chen, J., 2018. Fuzzy clustering based pseudo-swept volume decomposition for hexahedral meshing. Computer-Aided Design 96, 42–58.

Xiao, J., Fang, T., Zhao, P., Lhuillier, M., Quan, L., 2009. Image-based street-side city modeling. In: ACM Transactions on Graphics (TOG). Vol. 28. ACM, p. 114.

Xing, W., Song, C., Tin-Loi, F., 2018. A scaled boundary finite element based node-to-node scheme for 2D frictional contact problems. Computer Methods in Applied Mechanics and Engineering 333, 114–146.

Xing, W., Zhang, J., Song, C., Tin-Loi, F., 2019. A node-to-node scheme for three-dimensional contact problems using the scaled boundary finite element method. Computer Methods in Applied Mechanics and Engineering 347, 928–956.

Yan, D.-M., Wang, W., Levy, B., Liu, Y., 2013. Efficient computation of clipped Voronoi diagram for mesh generation. Computer-Aided Design 45, 843–852.

Ye, W., Liu, J., Lin, G., Xu, B., Yu, L., 2018. Application of scaled boundary finite element analysis for sloshing characteristics in an annular cylindrical container with porous structures. Engineering Analysis with Boundary Elements 97, 94–113.

Yerry, M. A., Shephard, M. S., 1983. A modified quadtree approach to finite element mesh generation. IEEE Computer Graphics and Applications 3, 39–46.

Yerry, M. A., Shephard, M. S., 1984. Automatic three-dimensional mesh generation by the modified-octree technique. International Journal for Numerical Methods in Engineering 20, 1965–1990.

Yu, C.-C., Heinrich, J. C., 1987. Petrov-Galerkin method for multidimensional, time-dependent, convective-diffusion equations. International Journal for Numerical Methods in Engineering 24, 2201–2215.

Zhang, A., Liu, Y., 2015. Improved three-dimensional bubble dynamics model based on boundary element method. Journal of Computational Physics 294, 208–223.

Zhang, P., Du, C., Tian, X., Jiang, S., 2018. A scaled boundary finite element method for modelling crack face contact problems. Computer Methods in Applied Mechanics and Engineering 328, 431–451.

Zhang, Z., Liu, Y., Dissanayake, D. D., Saputra, A. A., Song, C., 2019. Nonlocal damage modelling by the scaled boundary finite element method. Engineering Analysis with Boundary Elements 99, 29–45.

Zhong, H., Li, H., Ooi, E. T., Song, C., 2018. Hydraulic fracture at the dam-foundation interface using the scaled boundary finite element method coupled with the cohesive crack model. Engineering Analysis with Boundary Elements 88, 41–53.

Zhou, X., Sutulo, S., Soares, C. G., 2016. A paving algorithm for dynamic generation of quadrilateral meshes for online numerical simulations of ship manoeuvring in shallow water. Ocean Engineering 122, 10–21.

Zhu, J., Zienkiewicz, O., 1988. Adaptive techniques in the finite element method. International Journal for Numerical Methods in Biomedical Engineering 4, 197–204.

Zhu, J., Zienkiewicz, O., Hinton, E., Wu, J., 1991. A new approach to the development of automatic quadrilateral mesh generation. International Journal for Numerical Methods in Engineering 32, 849–866.

Zienkiewicz, O. C., Boroomand, B., Zhu, J. Z., 1999. Recovery procedures in error estimation and adaptivity part I: Adaptivity in linear problems. Computer Methods in Applied Mechanics and Engineering 176, 111–125.

Zienkiewicz, O. C., Cheung, Y. K., 1964. The finite element method for analysis of elastic isotropic and orthotropic slabs. Proceedings of the Institution of Civil Engineers 28, 471–488.

Zienkiewicz, O. C., Mayer, P., Cheung, Y. K., 1966. Solution of anisotropic seepage by finite elements. Journal of the Engineering Mechanics Division 92, 111–120.

Zienkiewicz, O. C., Taylor, R. L., Nithiarasu, P., Zhu, J., 1977. The Finite Element Method. Vol. 3. McGraw-hill London.

Zienkiewicz, O. C., Zhu, J. Z., 1987. A simple error estimator and adaptive procedure for practical engineerng analysis. International Journal for Numerical Methods in Engineering 24, 337–357.

Zienkiewicz, O. C., Zhu, J. Z., 1992a. The superconvergent patch recovery and a posteriori error estimates. Part 1: the recovery technique. International Journal for Numerical Methods in Engineering 33, 1331–1364.

Zienkiewicz, O. C., Zhu, J. Z., 1992b. The superconvergent patch recovery and a posteriori error estimates. Part 2: error estimates and adaptivity. International Journal for Numerical Methods in Engineering 33, 1365–1382.

Zlatanova, S., Paintsil, J., Tempfli, K., 1998. 3D object reconstruction from aerial stereo images. In: the 6th International Conference in Central Europe on Computer Graphics and Visualization.

Zou, D., Teng, X., Chen, K., Liu, J., 2019. A polyhedral scaled boundary finite element method for three-dimensional dynamic analysis of saturated porous media. Engineering Analysis with Boundary Elements 101, 343–359.

Zou, D., Teng, X., Chen, K., Yu, X., 2018. An extended polygon scaled boundary finite element method for the nonlinear dynamic analysis of saturated soil. Engineering Analysis with Boundary Elements 91, 150–161.