Académie Wallonie Europe
Université de Liège
Faculté des Sciences Appliquées
Mécanique Numérique Non Linéaire

From medical imaging to finite element simulations: a contribution to mesh generation and locking-free formulations for tetrahedra.

Vinciane d'OTREPPE de BOUVETTE
Ingénieur civil électromécanicien (aérospatiale)
Aspirante F.R.S.-FNRS

Thèse présentée en vue de l'obtention
du grade légal de Docteur en Sciences de l'Ingénieur

Octobre 2012

Abstract

Patient-specific finite element (FE) modelling is gaining more and more attention over the years because of its potential to improve clinical treatment and surgical outcomes. Thanks to patient-specific modelling, the design of individualised implants and prostheses, surgical pre-operative planning and simulation, and the computation of stresses and strains in a patient's organ for diagnostic purposes will become a reality in the future. This work investigates two of the most challenging tasks of patient-specific modelling: the creation of image-based finite element meshes and the development of a low-order locking-free tetrahedral element.

First, a general meshing strategy for tetrahedral mesh generation from segmented 3D images is proposed. The originality of the approach is the addition of surface reconstruction algorithm to the traditional image-to-mesh pipeline. The main advantages for this are: the generation of smooth boundaries, robustness to segmentation noise, a user-defined mesh resolution and a good fidelity of the mesh boundaries with respect to the underlying image. Also, the proposed meshing strategy is capable of generating meshes of heterogeneous structures, containing several interconnected types of tissues. Applications demonstrate that the interfaces between distinct material regions are topologically correct, i.e. the connections are edge-on-edge and node-on-node. Specific mesh decimation and mesh smoothing algorithms were designed for this multi-material tetrahedral mesh generator. In a last chapter, patient-specific hexahedral meshes are created by combining the proposed surface reconstruction algorithm with a classical voxel-conversion algorithm.

Second, a low-order tetrahedral element for the solution of solid mechanics problems involving nearly incompressible materials is developed. The formulation is based on F-bar methodologies and nodal-based formulations. As in nodal based formulations, nodal Jacobians are defined. These nodal quantities are then averaged over the element to define a modified elemental Jacobian, which is used to define a modified deformation gradient, F-bar, for the element. Both 2D triangular and 3D tetrahedral are proposed and they can be used for both implicit and explicit analysis. The exact stiffness terms for the tangent stiffness matrix are derived so that a quadratic convergence rate in ensured for the Newton-Raphson equilibrium iterations. Most importantly, the new element can be used regardless the material model. Benchmarking 2D and 3D numerical tests using several constitutive models indicate a substantial removing of both the volumetric and the shear locking tendency of the standard linear triangle and tetrahedron, as well as an accurate distribution of strain, stress and pressure fields.

The potential of the resulting image - to - FE model procedure is demonstrated in the last part of this work, through patient-specific finite element analyses of actual biomechanical research topics.

Acknowledgements

Je tiens à remercier toutes les personnes qui, de près ou de loin, ont contribué à la concrétisation de ce travail de thèse de doctorat.

Je remercie tout d'abord les membres du jury d'avoir accepté de lire et de juger mon travail.

Je tiens ensuite à remercier mon promoteur, le professeur Jean-Philippe Ponthot. Merci d'avoir cru en moi alors que je n'étais qu'une étudiante de Master et de m'avoir encouragée à postuler au FNRS. Merci d'avoir suivi et encadré mon doctorat. Tu m'as aidée lorsque j'en avais besoin, tout en me laissant l'autonomie nécessaire afin que je puisse emprunter les directions de recherches que je trouvaient pertinentes.

Merci à l'équipe du service LTAS-MN^{2}L au sein duquel cette recherche s'est déroulée dans une bonne ambiance. Plus particulièrement, merci à Romain Boman et à Luc Papeleux pour les longues discussions sur Metafor. Merci également de passer tant de temps et d'énergie à améliorer et rendre accessible ce logiciel éléments finis. Grâce à cela les développements de cette thèse ne seront pas perdus et pourront être ré-utilisés facilement. Merci aussi à Marlène Mengoni pour sa motivation à faire de la biomécanique numérique un domaine de recherches reconnu à l'Université de Liège; et d'avoir été la première à utiliser mon mailleur et mes maillages sur des cas concrets de biomécanique.

Je tiens à remercier également le Professeur Marc Balligand, Béatrice Böhme et Nestor Nsiampa d'avoir cru en mes capacités de modélisation et de m'avoir fourni des applications réelles sur lesquelles tester ce travail de thèse. Ce fut une réelle motivation pour moi de voir que ce travail puisse profiter à d'autres chercheurs.

This thesis was mainly written in Australia. I would like to thank all the members of the Paediatric Spine Research Group, Brisbane, who considered me as part their team. I say thank you to Professors Clayton Adam and Mark Pearcy for sharing their expertise, insights and passion and a thanks to Dr. Paige Little for supervising my work in Brisbane.

Je tiens également à remercier le Fonds National pour la Recherche Scientifique, F.R.S.-FNRS, pour m'avoir accordée une bourse d'Aspirant et pour m'avoir permis de participer à plusieurs congrès et séjours de recherches; ainsi que l'Université de Liège pour les bourses accordées.

Enfin merci à ma famille et amis. Merci à mes parents de m'avoir soutenue tout au long de mes études. Et un énorme merci à Axel, pour m'avoir supportée sans relache pendant ces quatre années de doctorat, et pour m'avoir accompagné jusqu'en Australie. Merci de t'être intéressé et d'avoir fait l'effort de comprendre mes recherches.

Contents

Abstract i

Acknowledgements iii

Table of contents v

Introduction v

1 Goals and introduction of the research 3
- 1.1 Computational biomechanics . . . 3
- 1.2 Image-to-mesh pipeline . . . 5
- 1.3 The finite element method and volumetric locking . . . 7
- 1.4 Biomechanical applications . . . 7
- 1.5 Contributions of this thesis . . . 8

I Mesh generation from medical datasets 11

2 Challenges and compromises of patient-specific mesh generation 13
- 2.1 Challenges . . . 14
- 2.2 Defining objectives and outcomes . . . 14
- 2.3 Different ways to represent medical data . . . 16
- 2.4 Image-to-mesh pathways . . . 20
- 2.5 Recommended image-to-mesh pathways . . . 21
- 2.6 Conclusions . . . 26

3 From segmented 3D images to implicitly defined analytical surfaces 29
- 3.1 Motivation . . . 29
- 3.2 Literature review . . . 31
- 3.3 The multi-level Partition of Unity surface reconstruction method . . . 36
- 3.4 Our contributions to the multi-level Partition of Unity method . . . 43

3.5 Implicit representation of multi-label datasets 49
3.6 Geometric approximation accuracy evaluation 51
3.7 Applications and results 53
3.8 Conclusions 61

4 Multi-domain tetrahedral mesh generation and adaptation 63
4.1 Literature review 63
4.2 Classical marching tetrahedra algorithm 65
4.3 Multi-material marching tetrahedra algorithm 66
4.4 Decimation of multi-material surface meshes 69
4.5 Mesh adaptation 71
4.6 Applications and results 72
4.7 Conclusions 84

5 Multi-domain hexahedral mesh generation and adaptation 85
5.1 Motivation 85
5.2 Literature Review 86
5.3 Proposed approach 87
5.4 Applications and results 90
5.5 Conclusions 94

II Unlocking the linear tetrahedron 97

Notations 99

6 Background 103
6.1 The Finite Element Method 103
6.2 Continuum mechanics 104
6.3 Total and updated Lagrangian formulations 109
6.4 Consistent Linearisation and Tangent Stiffness Matrix 113
6.5 Time integration 121
6.6 The standard linear tetrahedron 123
6.7 Conclusions 129

7 Towards a locking-free formulation for the linear tetrahedron 131
7.1 Nodal-based formulations 133
7.2 F-bar and F-bar-patched methods 142
7.3 Contribution 1: a patch volume change ratio linear tetrahedron 150
7.4 Contribution 2: an Average Elemental Jacobian (AEJ) tetrahedral element . . 153

7.5 Conclusions 158

8 Numerical applications **161**
8.1 Cook's membrane 161
8.2 Thick-walled cylinder under internal pressure 179
8.3 Taylor bar impact 185
8.4 Concluding Remarks 190

III Biomechanical Applications **193**

9 Introduction **195**

10 Influence of meshing strategy on FE analysis of cellular structures **197**
10.1 Building of the finite element model 197
10.2 Finite element simulations 201
10.3 Conclusions 201

11 Finite element modelling of brain shift deformation **205**
11.1 Context 205
11.2 Building of patient-specific biomechanical model 207
11.3 Finite element simulations and results 208
11.4 Conclusions 210

12 Modelling of canine humeral condylar fractures **213**
12.1 Context 214
12.2 Dataset preparation 214
12.3 FE study of the influence of elbow configuration on fracture type 215
12.4 Finite element simulation of a condylar fractures 220
12.5 Conclusions and future work 224

Conclusions **vii**

13 Conclusions **229**
13.1 General conclusions 229
13.2 Patient-specific finite element mesh generation 230
13.3 Locking-free formulations for the linear tetrahedron under 233
13.4 Applications of this work and Perspectives 235

Appendix **vii**

A Tensor Analysis **239**
A.1 Tensor Product 239

B Directional derivatives **241**
B.1 Directional derivative of a function 241
B.2 Directional derivative and partial derivatives 242
B.3 Directional derivative of the determinant of a matrix 242
B.4 Linearisation of the deformation gradient 243
B.5 Linearisation of the Jacobian 243

C F-bar methodology **245**
C.1 Virtual internal work linearisation for the F-bar Hexahedron 245

Bibliography **249**

Introduction

Chapter 1

Goals and introduction of the research

1.1 Computational biomechanics

Biomechanics is broadly defined as the scientific discipline that investigates the effects of forces acting on and within biological structures. **Computational Biomechanics** uses mathematical modelling and computer simulation with the aim of

- providing a better understanding of human and animal bodies at all scales; important research fields being the characterisation and modelling of *Tissue properties* [83] and *Bone properties* [46], *Injury Mechanics analysis* [11, 72, 125] and *Multi-scale modelling* [14, 65, 87, 174];
- providing the medical doctors with the possibility to simulate and plan a surgical procedure pre-operatively [30, 44, 141, 166];
- enhancing visualisation techniques during surgery, mainly via *Image-guided Surgery* techniques [77, 119, 176, 183];
- designing better implants and prostheses [70, 89, 90, 149, 171, 179].

Computational mechanics has led to technological developments in all traditional engineering disciplines, as, for example, in the fields of aerospace, automotive, civil and structural engineering. The challenge for researchers in *Computational Biomechanics* is to extend this success to biomedical sciences and medicine. The major issue certainly is that biological structures are rather complex so that creating computer models, ideally based on medical scans, defining material properties, applying initial and boundary conditions and validation are often problematical.

In an attempt to simplify the approach, many investigations in biomechanics employ generic finite element meshes based on generic patient geometries. In most studies however, the variation in human anatomical structures in geometrical shape and tissue properties must be taken into account. **Patient-specific modelling** is the development of computational models of human patho-physiology that are individualized to patient-specific data. Patient-specific modelling is gaining more and more attention over the years, as demonstrated by the increasing numbers of submitted articles and funding, because of its potential to improve clinical treatment and surgical outcomes. Thanks to patient-specific modelling, the design of individualised implants and prostheses, surgical pre-operative planning and simulation, and the computation of stresses and strains in a patient's organ for diagnostic purposes will become a reality in the future.

Popular research topics in *patient-specific modelling* are:

- The investigation of the effects of cardio-vascular devices, a stent for example, on blood circulation as well as the prediction of outcomes of therapies for individual patients [164]. In general, meshes based on medical images (MRI, CT or US) are generated to create the needed *computational fluid dynamics* models.
- The prediction of the likelihood of vascular aneurysm by computing the stresses, caused by blood circulation, in the vessel walls. [36, 37, 168]. These models fall into the category of *fluid-structure interaction* modelling.
- The modelling of heart contraction mechanics and linked electrical activation using *coupled* electrical and finite element models [127, 153]. A thermoelectric coupling due to the presence of a pacemaker could also be included [43].
- The simulation of brain deformation due to craniotomy and surgery, with the goal of improving image-guided visualisation tools [119, 176, 183], which can also be done by *finite element analysis*.
- The estimation of the needed scoliosis deformity correction for a particular treatment or surgery, using a *finite element* model of the patient's spine [1, 76, 106].

The above examples illustrate the variety of research areas in patient-specific modelling: variety in the structures that are considered (blood vessels, heart, brain, bones), variety in the scale of interest (tissue-level, organ-level, system level), variety in the numerical method used (computational fluid dynamics, fluid-structure interaction, finite element method, coupled methods, ...).

The scope of this dissertation is patient-specific modelling of solid biological structures. The finite element method will be used to solve the equations of continuum mechanics.

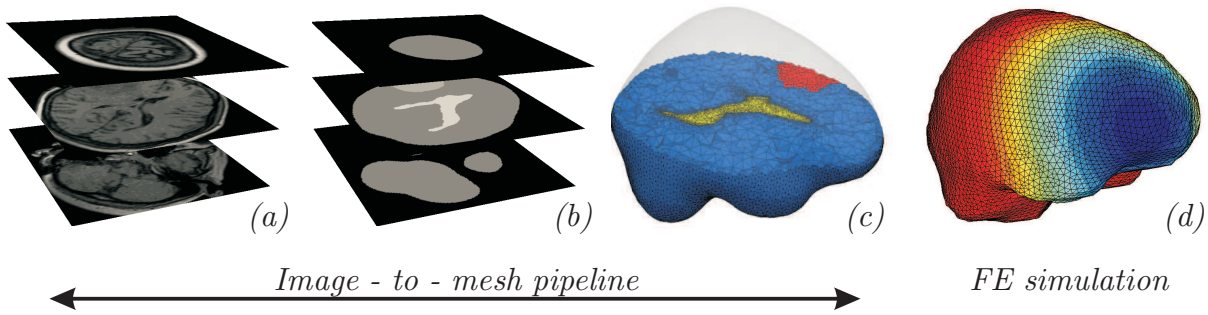


FIGURE 1.1: ***Patient-specific modelling pipeline.*** *(a) Images are taken from the patient. (b) The structures of interest are extracted from theses images. (c) A computer model is built. (d) Advanced numerical methods are employed to calculate the behaviour of the structure under specific loads and boundary conditions.*

Instead of studying one part of the body in detail, this thesis investigates generic algorithms that can be used to model any solid biological structure (bone, soft tissue, ...). The image-to-mesh pipeline, considered as the bottleneck of patient-specific modelling, is examined in detail and issues regarding the finite element modelling of incompressible media are solved. These two topics constitute *Part One* and *Part Two* of this manuscript and are introduced in the following two sections.

1.2 Image-to-mesh pipeline

In patient-specific modelling, images are taken from the patient, the structures of interest are extracted from these images, a computer model is built and advanced numerical methods are employed to calculate the behaviour of the structure under specific loads and boundary conditions. The process of generating a computational mesh from a medical image is called the *image-to-mesh pipeline* and is summarised in Figure 1.1.

The image-to-mesh pipeline requires many competences as it is located at the junction of different research fields: image processing, geometry, surfacing, meshing and numerical modelling. Too often researchers in these fields consider their problem as an independent setting and not as part of a pipeline. Instead, it is the author's belief that the image-to-mesh pipeline should be considered as a whole and be designed to meet the final goals of improving surgical outcomes by building patient-specific computer models. In this perspective robustness, computation time and automatisation are particularly important.

In addition to the variety of research topics, the variety of biomechanical applications, makes it even more difficult for researchers to distinguish the nuances between the many algorithms proposed in literature. It is an objective of this thesis to give a better insight into

available methods, their usability within the image-to-mesh pipeline, their limitations and their range of application.

Let us now review the different steps involved in the image-to-mesh pipeline, illustrated in Figure 1.1. Patient-specific mesh generation always starts with data acquisition. Most often computed tomography (CT) or magnetic resonance imaging (MRI) is used to obtain a three-dimensional image, that may be viewed as a set of cross-section slices (Figure 1.1,(a)). The value of each voxel[1] is linked to the tissue in which it is located.

The second and often also manual step within the image-to-mesh pipeline is the extraction of the structures of interests in the three-dimensional medical images. When no a priori knowledge of the image is taken into account, this step consists either in segmentation or in landmark detection. On the one hand, segmentation consists in dividing the greyscale voxels into groups, each group corresponding either to a tissue or the background (Figure 1.1, (b)). On the other hand, landmarks detection refers to the identification of specific reference points in the image. Unfortunately, both approaches involve manual steps. These manual steps are the main raison why patient-specific modelling is still not a reality in the hospital. In fact, these manual steps could be removed by studying a specific part of the body and targeting a specific medical application. But, in all cases, there is trade-off between the generality of the approach, the time needed to generate the computer models and the geometric accuracy of these models. Chapter 2 explains how different applications can lead to different choices in regard to the method that should be adopted.

The next step in the image-to-mesh pipeline is the building of a computational grid, that is, in the framework of the finite element method, a mesh (Figure 1.1, (c)). This mesh consists in a series of nodes connected together in a specific way so that the object is subdivided in a series of elemental volumes. The latter usually are tetrahedrons or hexahedrons, even though other topologies exist. The choice of the finite element type is important. Tetrahedrons are easy to generate: any volume can be subdivided into tetrahedrons in an automated manner and they can approximate well curved objects boundaries. This makes the tetrahedron the element of choice for complex geometries. Hexahedrons are more popular in finite element simulations because they behave well (generally much better than tetrahedrons) in numerical computation, while tetrahedrons may produce inaccurate results under certain conditions. This makes the hexahedron the element of choice for advanced finite element simulations. This trade-off will also be investigated in this work: both patient-specific tetrahedral and hexahedral meshes will be generated and used in finite element simulations, so that we will be able to compare both approaches knowingly and draw conclusions.

[1] A voxel, *volumetric pixel*, is a volume element representing a value on a regular grid in three dimensional space, i.e. in a 3D-image.

1.3 The finite element method and volumetric locking

It is well-known that low-order finite elements exhibit an overstiff behaviour under near incompressibility constraints. This spurious stiffness, called *volumetric locking*, is observed during the finite element analysis of incompressible linear elastic and hyperelastic materials, but also for J2 elasto-plastic materials.

The problem can easily be solved by using higher-order elements. But, due to their simplicity and robustness, elements with linear shape functions are often preferred for non-linear problems, particularly when these involve high strains, frictional contact or material fracture.

Efficient unlocking solutions for the linear quadrilateral and hexahedron have been proposed. But, because of the complexity of the geometries involved, tetrahedral meshes are often more practical in computational biomechanics.

The quest for a low-order locking-free tetrahedral element is the object of Part 2 of this dissertation. Ideally, the element should be suitable for both explicit and implicit analysis. Also, it should not impose particular restrictions on the employed constitutive model. A third requirement for this work is that it has to be easily implementable into the in-house finite element software Metafor.

In Chapter 7, a new low-order non-locking tetrahedral element that meets the above requirements is developed. Benchmarking numerical tests are performed in Chapter 8 in order to compare its performance with other popular finite elements of the literature.

1.4 Biomechanical applications

The last part of this thesis demonstrate the applicability of this work to solve real-life biomedical problems. All three applications are the result of collaborative projects that could benefit from the meshing algorithms and/or the non-locking tetrahedral element developed in this thesis.

The first application is the finite element analysis of the compression of a deer antler cancellous bone. Several types of meshing methods, hexahedral and tetrahedral, are studied and their influence on the results of the finite element simulation is assessed.

The second application is the finite element modelling of intra-operative brain shift deformation, based on pre-operative and intra-operative scan-data. We use both our mesh-

ing algorithm and our non-locking tetrahedral element to improve a previously proposed biomechanical model of the brain [175].

The third application is the finite element study of dog humeral fractures. The developments of this thesis are used to create a patient-specific model of a dog elbow. The influence of the skeletal development (young versus adult dog), the elbow configuration (flexion-extension and exo-endoration angles) and the load direction on stress distribution within the humerus is analysed; and possible fracture types are deduced.

The purpose of these applications is to illustrate the possibilities and application range of the algorithms proposed in this work, and not, in the framework of this thesis, to solve specific problems of biomechanics.

1.5 Contributions of this thesis

1.5.1 Major contributions

1. A patient-specific multi-material tetrahedral mesh generator

A new strategy to create the meshes required for finite element analysis from segmented medical scans is presented. The originalities of the proposed mesh generator are detailed below.

- It creates topologically correct multi-material meshes in an integrated manner.
- It removes the characteristic stair-case irregularities appearing in meshes created from segmented scans, without jeopardising the fidelity of the model with respect to the underlying image.
- It is associated with specific mesh decimation and mesh adaptation algorithms that preserve the geometric accuracy and the multi-material nature of the model.

2. A robust, geometry-preserving, smoothing method for voxel-based meshes

A novel strategy to smooth hexahedral voxel-based meshes based on the use of a multi-level partition of unity (MPU) surface reconstruction method is proposed. The resulting hexahedral mesh generator has the following specificities.

- It generates patient-specific hexahedral meshes with relatively smooth boundaries.

- The element distortion due to smoothing is limited by propagating the mesh deformation through the volume.
- It is very well adapted for the generation of multi-material meshes. In that case, inner and outer boundaries are smoothed.
- It is very robust and effective in generating hexahedral meshes of truss-like structures, like foams or trabecular bone, for which most algorithms fail.

3. A new locking-free formulation for the low-order tetrahedron that is suitable for the large strain analysis of nearly incompressible solids

A low-order finite element suitable for the large strain analysis of nearly incompressible solids is proposed. The unlocking formulation is proposed for both the two-dimensional triangle and the three-dimensional tetrahedron. Compared to other formulations, the main advantages of the proposed formulation are:

- It is suitable for both explicit and implicit analysis.
- It preserves the displacement-based structure of the finite element equations.
- It can be used with any, strain-driven, constitutive model.
- The exact expression of the stiffness terms in the tangent stiffness matrix are proposed. Therefore, the Newton-Raphson algorithm can be used to solve the global equilibrium equations and quadratic convergence rates are ensured.
- It can be used for heterogeneous solids constituted of several materials.

The performance of the proposed element is thoroughly assessed by means of popular 2D and 3D benchmarking tests. Results indicate that the proposed element substantially reduces both the volumetric and the shear locking of the standard linear tetrahedron. It also allows a good evaluation of the deformed shapes as well as stress, strain and pressure fields within the solid.

1.5.2 Other contributions and novelties

Other contributions of this work are:

- on realistic biomedical applications, a demonstration of the whole process of patient-specific finite element modelling: image processing, mesh generation, model definition, finite element analysis;
- the comparison of different meshing approaches on finite element results of cancellous bone compression;
- the use of the proposed non-locking tetrahedra for the modelling of intra-operative brain deformation;
- the finite element modelling of canine humeral fractures;
- the extension of the multi-level partition of unity implicit surface reconstruction method for the representation of heterogeneous objects;
- the presentation of the mathematical developments for the calculation of the exact stiffness terms of popular and proposed unlocking formulation, rarely provided in literature;
- for the new researcher in the field: a global view of the image-to-mesh pipeline, the popular meshing strategies and practical guidelines with respect to the targeted application;

Finally, the image-to-mesh approach and the unlocking formulation for the linear tetrahedral element were implemented and are available in the large strains finite element code Metafor, developed at the non-linear computational mechanics laboratory of the University of Liege. Also, a graphical user interface was developed to simplify image visualisation and pre-processing as well as mesh generation, visualisation and adaptation. Therefore this thesis is a first but significant step in making the finite element software Metafor suitable for patient-specific modelling.

Part I

Mesh generation from medical datasets

Chapter 2

Challenges and compromises of patient-specific mesh generation

In this chapter, patient-specific mesh generation is broadly introduced. The discretisation of the domain of interest, through the generation a mesh, is one of the first steps of finite element modelling. In patient-specific biomechanical modelling, the finite element model is generated on basis of the patient's medical scans, we speak of an image-to-mesh procedure. The wide range of possible biomechanical applications, and the variety of available modelling software, results in numerous available image-to-mesh strategies in literature. As a result, it is often difficult for the out-of-the-domain researcher to select the most appropriate procedure for a particular application.

In Section 2.1, we present the challenges of patient-specific mesh generation. Section 2.2 gives the key questions that a researcher should consider in order to define an appropriate meshing strategy. Mesh generation actually consists in the transformation of one type of data representation to another, more adapted to finite element analysis. Section 2.3 reviews the different ways to represent geometries, from medical data, to analytic functions and finite element meshes. From this prerequisite, Section 2.2 presents popular pathways to transform medical data into a finite element mesh. Taking account of the objectives defined in Section 2.2, Section 2.5 gives practical recommendations on which meshing procedure to take.

2.1 Challenges

Despite numerous papers on mesh generation methods available in literature, extracting finite element meshes from medical data is still a challenge. Main reasons for this are that, in patient-specific mesh generation:

- The geometry is not described analytically but must be extracted from three-dimensional medical images, as opposed to CAD geometries[1] for example.
- Discerning the tissues in medical images is rarely straightforward.
- The geometry of medical tissues is usually far more complex than in traditional engineering fields.

2.2 Defining objectives and outcomes

The possibilities of obtaining a finite element mesh from an image are numerous and the path to be chosen will depend on the targeted application. In order to determine which procedure to take, the researcher should answer and prioritize the following questions.

Which level of geometric accuracy is required? Patient-specificity is not always a necessity; sometimes the mesh may be built from generic, averaged, geometries. When the application requires a patient-specific mesh, it is useful to define what level of accuracy is needed. The geometric accuracy of the mesh will generally be measured by calculating the Hausdorff distance[2] between the mesh and the initial dataset [88]. For the model to correspond to the real geometry, the medical datasets must also be acquired with a sufficiently fine resolution. High geometric accuracy will generally require precise image segmentation, tetrahedral meshing and large numbers of degrees of freedom.

What are the targeted simulations times? When real-time simulations are needed, the method should produce as few elements as possible. An efficient solution to limit the overall number of elements whilst still producing accurate results is to use adaptive meshes. In adaptive meshes, the mesh scale is allowed to vary spatially so that small elements are generated along the region of interest while larger elements are

[1] Computer-aided design (CAD) is the use of computer systems to assist in the creation, modification, analysis, or optimization of a design [126].

[2] The Hausdorff distance will be defined in Section 3.6.2.

produced further away. However, in the field of patient-specific modelling, only a few mesh generation procedures produce adaptive meshes and the latter are generally created via subsequent consecutive mesh decimation and adaptation algorithms [67, 68].

Which type of finite elements should be generated? Quadrilateral and hexahedral elements are better than triangles and tetrahedrons for the convergence of the finite element method. This is even more true in the biomechanical field because biological tissues are often incompressible, leading to volumetric locking of the standard linear triangle and tetrahedron. However, generating hexahedral meshes from medical data results in a trade-off between geometric accuracy and automaticity : no meshing method exists that produces a three-dimensional hexahedral mesh of good geometric accuracy and mesh quality without any user input.

Should the model include shell, spring or beam elements? Some tissues will be better modelled as collections of shell elements, like the ribs, or with membrane elements, like the cranial dura matter. Others will be better modelled by connectors or spring elements, like the ligaments. Applying a volumetric mesh generation algorithm to these structures will generate a needlessly large number of tetrahedra and hexahedra. When volumetric and non-volumetric elements should be generated from the same medical dataset two procedures may be taken. The most common solution is to use a volumetric patient-specific mesh generation algorithm and add the 1D and 2D elements afterwards, manually. When the location of these elements should absolutely be automatically extracted from the patient's datasets, specific meshing approaches may be required.

Which level of complexity should the model include? Should the model include several structures, with different material properties, and are these tissues connected/attached to each other? If yes, how should the interface between these material domains be modelled? Distinct tissues composing the medical structure should correspond to distinct material domains. When these tissues are allowed to move separately, or slide along each other, distinct meshes should be generated and contact properties should be defined. When, however, these tissues are intimately linked, the whole structure should be modelled as one unique mesh, made of one outer boundary and one or several inner boundaries: the outer boundary separates the structure with the exterior and the inner boundaries separate two distinct tissues. Ideally, the inner boundary meshes should then join the outer boundary mesh in a consistent way, that is to say, without T-junctions nor gaps. This type of meshing will be referred to as *multi-material meshing* in this work.

What is the level of automation required ? Will the whole meshing procedure be repeated for several patients ? Some mesh generation procedures may be time-consuming the first time but almost automatic afterwards. Also, some meshing approaches may utilize the first mesh to create the subsequent, not so different, meshes.

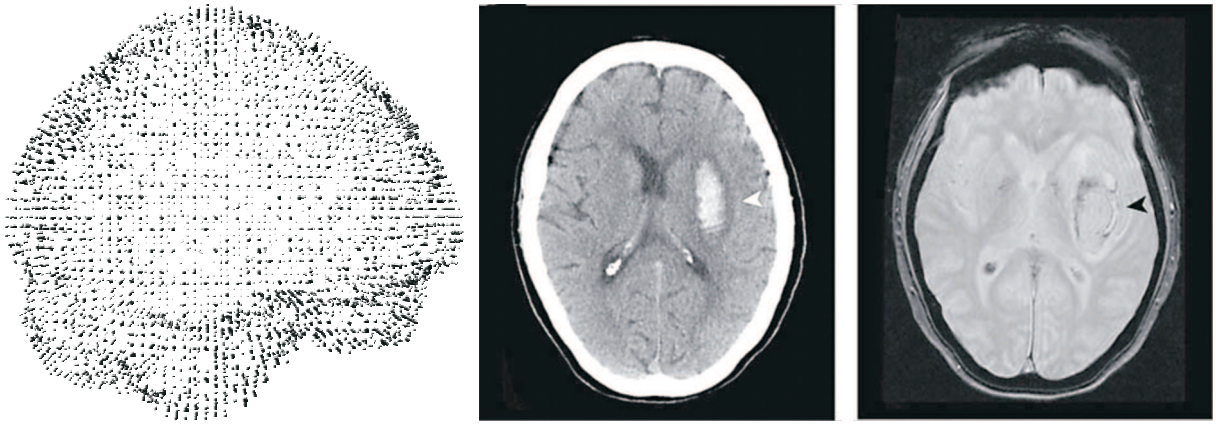
Finally, for the sake of rapid convergence of the non-linear finite element iterative solution algorithm, the quality of the generated elements, hexahedral or tetrahedral, should always be acceptable in the sense of the finite element method. And, needless to say, the generated meshes should be topologically valid, also in the sense of the finite element method, meaning that connections should be node-on-node, edge-on-edge and face-on-face.

2.3 Different ways to represent medical data

Finite element model creation consists in transforming medical data into a form that is better adapted to computer modelling. There exists indeed many ways to represent a geometry, and distinct data acquisition techniques and mesh generation methods will produce and utilize one or the other form. This section reviews the different ways to represent a three-dimensional object, from the acquisition of the dataset to the volume mesh used in finite element simulation.

Data acquisition techniques will produce different outputs. Laser scanners, extensively used in Computer Graphics, only acquire the outer surface of the object, and generate a point set in the three-dimensional space (Figure 2.1, Left). On the other hand, Computed Tomography (CT) and Magnetic Resonance Imaging (MRI), popularly used by physicians, acquire the volume of an object. Computer Tomography is an imaging tool that combines X-rays with computer technology to produce a series of, generally parallel, two dimensional scans (Figure 2.1, Middle). Magnetic resonance imaging uses magnetic fields to acquire the volume and produces a three-dimensional image, viewed as a regular, but usually anisotropic, grid of voxels (Figure 2.1, Right). In each case, the pixel value of the two-dimensional cross-section or the voxel value of the three-dimensional image provide information on the internal structure of the scanned object. It is important to note that the set of two-dimensional scans may be seen as a three-dimensional image, if we stack all 2D scans. Finally the voxels may be seen as a set of points in 3D with specific values, each point being the centre of the hexahedral voxel in the image.

Medical images (2D or 3D) are not suited for computational analysis and must be preprocessed. They contain too much information, and researchers will need to extract the surface or the volume that they need to model, here-after called the object of interest. Seg-



Set of points *CT-scan* *MRI slice*

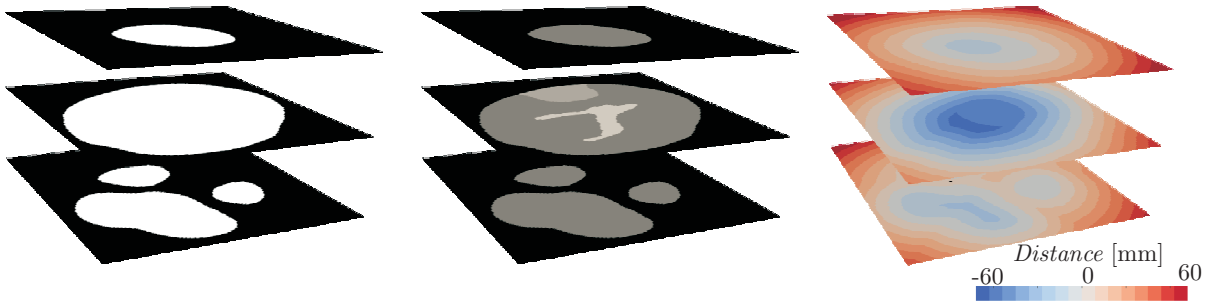
FIGURE 2.1: ***Different ways to represent medical data.*** *Classical outputs of data acquisition techniques. Left: Point set produced by laser scanners [95]. Middle: Two-dimensional scans of the brain produced by Computer Tomography imaging. Right: Slice of a three-dimensional image of the brain produced by Magnetic Resonance Imaging [95].*

mentation consists in colouring the pixels of the successive 2D image slices, or the voxels of the 3D image, according to whether they are located inside (value 1) or outside (value 0) the object of interest; leading to binary 2D slices or a binary volume image respectively (Figure 2.2, Left). More values may be used when multiple structures (e.g. containing several materials) must be extracted, leading to multi-label images (Figure 2.2, Middle). Some image-to-mesh procedures may require the application of 2D or 3D distances filters[3] on these segmented datasets, in order to have an information on the distance to the boundary of the object of interest (Figure 2.2, Right).

An alternative method to *extract* an object from a medical image is the identification of anatomical landmarks, distinct geometrically recognizable features, or reference points, in the medical data (Figure 2.3, Left). Interpolating lines and curved are then manually defined in order to delineate the boundary of the object (Figure 2.3, Middle), which results in a CAD model (Figure 2.3, Right).

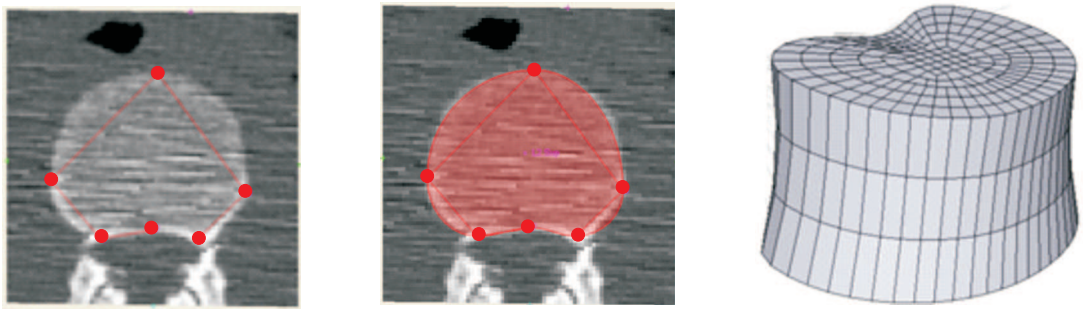
An analytic representation of the image may also be employed to guide mesh generation: the surface of the object is represented by an implicit equation of the form $f(\boldsymbol{x}) = c$ (Figure 2.4). This function enables us to determine if a particular point of the three-dimensional space is inside the object $f(\boldsymbol{x}) > 0$, on the object boundary $f(\boldsymbol{x}) = 0$ or outside the object $f(\boldsymbol{x}) < 0$. Approaches to obtain implicit surfaces form medical images or from a set of points are reviewed in Chapter 3.

[3]Applied on a binary image, a distance filter will output a grey-scale image in which each pixel (2D image) or voxel (3D image) is a measure of the distance to the segmented region in the input image.



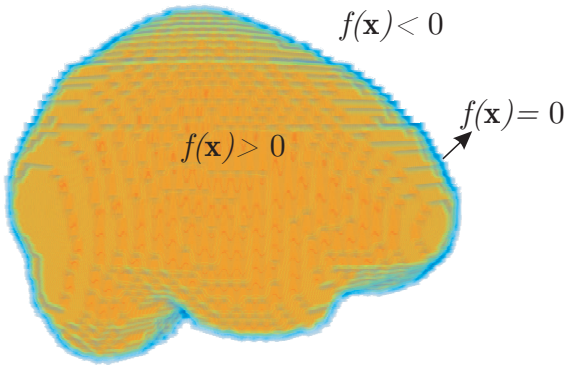
binary 3D image *multi-label 3D image* *3D Distance field*

FIGURE 2.2: ***Different ways to represent medical data.*** *Outputs of image processing. Left: Binary three-dimensional image produced by image segmentation. Middle: Multi-label three-dimensional image produced by image segmentation or region labelling. Right: Distance three-dimensional image produced by applying a distance filter to a binary image.*



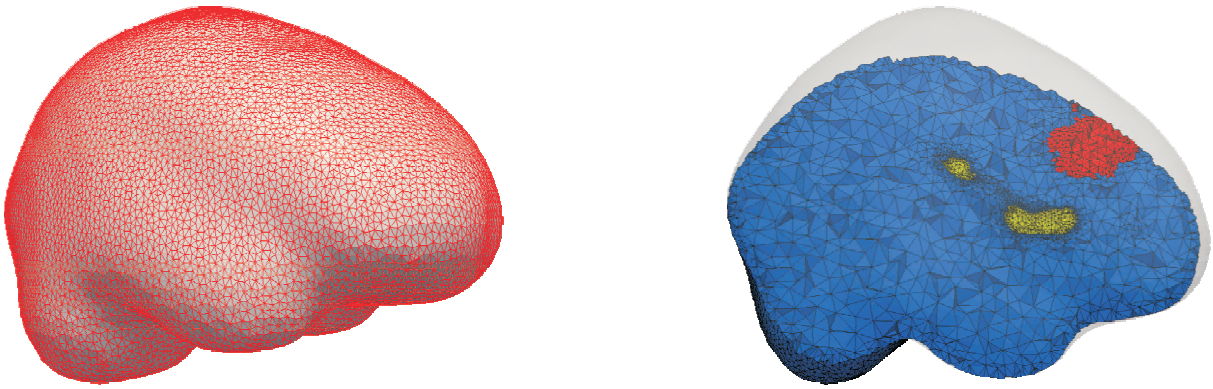
Anatomical Landmarks *Analytical curve* *CAD model*

FIGURE 2.3: ***Different ways to represent medical data.*** *Feature points and CAD model. Left: Anatomical landmarks or reference points, ordered in a pre-defined way as shown by the red line. Middle: Analytical curves and lines. The closed analytical curve is oriented so that its interior indicates the interior of the structure, as highlighted in red. Right: Computer-aided design and drafting model built from the anatomical reference points, analytical curves and surfaces. Source: Paediatric Spine Research Group, Queensland University of Technology [107].*



Implicit function

FIGURE 2.4: ***Different ways to represent medical data.*** *Implicit function. Interior:* $f(\boldsymbol{x}) > 0$. *Boundary surface:* $f(\boldsymbol{x}) = 0$. *Exterior:* $f(\boldsymbol{x}) < 0$.



surface mesh *volume mesh*

FIGURE 2.5: ***Different ways to represent medical data.*** *Surface and volume meshes. Left: Triangular surface mesh of the brain boundary. Right: Tetrahedral volume mesh of the brain.*

Numerical computation requires the data to be further simplified in the form of a mesh (Figure 2.5). In a volume mesh, the volume of the object is represented as a set of elementary volumes connected together in a *valid* manner. Even though more complex meshes may exist (hybrid, prism, ...), this dissertation focuses on meshes that are composed of tetrahedra or hexahedra only. Whereas a hexahedral mesh is generally generated throughout the volume directly, the creation of a tetrahedral mesh of a volume will often need the intermediate creation of a triangular mesh of its boundary. The different ways to obtain a polygonal mesh from medical images or point sets are reviewed here-after, in Section 2.4.

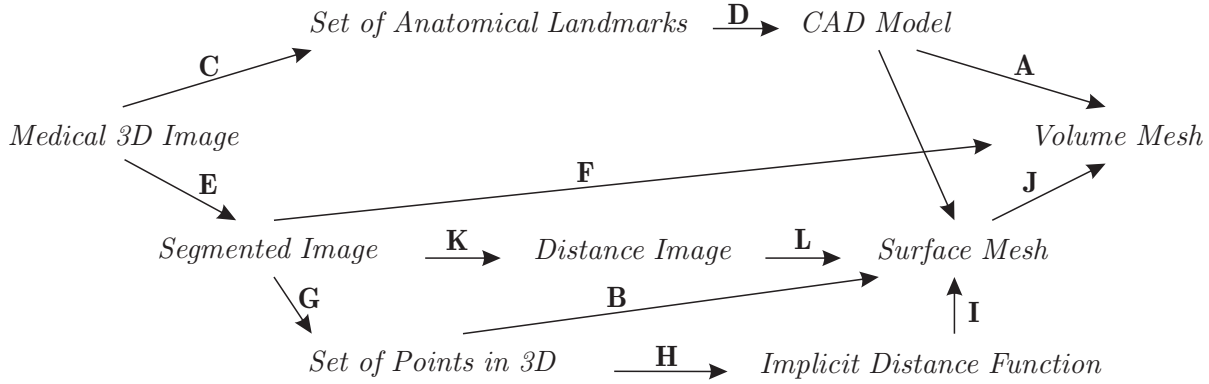


FIGURE 2.6: ***Image-to-mesh pathways.***

2.4 Image-to-mesh pathways

All the geometric representations described above may be a starting point to mesh generation techniques. Volume or surface meshes may be directly generated from segmented 2D scans or from a segmented 3D image. The points in a 3D point set may be connected to form a polygonisation. An implicit surface representation may also be extracted from the 3D point set in order to facilitate further surface mesh generation. A volume mesh may be obtained from an implicit surface representation or from a surface mesh of the boundary.

The different paths to generate a volume mesh from medical datasets are illustrated in Figure 2.6.

Path A Mesh generation procedures were originally developed in the Mechanical Engineering community to generate meshes from digital models, created using Computer Aided Design software.

Path B The creation of polygonal meshes from a set of points belonging to the exterior surface of an object, acquired from laser scanners, has also been extensively investigated, mainly for computer visualisation purposes.

More recently, with the advances in the medical field, researches are striving to generate finite element meshes from a three-dimensional image, or equivalently, from two-dimensional scans. Two main approaches, with many variants, are taken.

On the one hand, in an attempt to extend **path A** (CAD model to Volume Mesh),

Path C-D-A a CAD model is constructed by first extracting relevant anatomical landmarks from the medical image (**C**) and second joining these reference points, using CAD software or in-house code, to create a digital model of the object's surface (**D**). Usually, when commercial software is used, a stereolithography (STL) file will generally

by generated. This STL file is a representation of the boundary surface of the object in the form of a triangulation. Because STL files are mainly used for visualisation purposed, the quality of the triangulation is far from satisfying the requirements for a finite element mesh. Therefore specific techniques must be used in a subsequent step to create a finite element surface and/or volume mesh [13].

On the other hand, **segmentation techniques** are used in a first pre-processing step to delineate the object of interest in the 3D medical image (**E**). From a segmented image, many options are possible:

path E-F a hexahedral volume mesh can be easily generated by turning each image voxel into a hexahedral finite element, or,

path E-K-L-J a triangular surface mesh may be computed by using popular surface triangulation techniques. Surface triangulation algorithms generally need to compute the distance to the surface to be meshed, at a finite number of points in the three-dimensional space. In general, this information is provided by applying a distance filter to the segmented image (**K**); the surface mesh may then be obtained more easily (**L**).

path E-G-H-I-J The alternative path proposed in this dissertation is to first extract a set of points belonging to the surface of the object (**G**), second define a continuous implicit analytical function $f(\boldsymbol{x})$ that gives, at every point of the three-dimensional space $\boldsymbol{x}$, the distance to the object's surface, that is to say to the set of points, (**H**), and then use this continuously defined distance field in place of the discrete distance image to generate the surface mesh (**I**). A tetrahedral volume mesh from a surface triangulation is then obtained using well-documented algorithms or available software (**J**).

2.5 Recommended image-to-mesh pathways

A striving force during these four years of doctoral studies has been to create a meshing algorithm that would solve all issues of patient-specific mesh generation and this, without restricting the algorithm to specific applications. However, the author also believes that, when specific applications are targeted, this generality may not be needed and tailor-made meshing algorithms may then be more efficient, for example by requiring less user input.

The next sections will give the researchers some practical recommendations on which mesh generation procedure to use, in regard to the application targeted, and as a result of his responses to the questions of Section 2.2. Let us recall that these questions were related

to the desired level of geometric accuracy, the targeted simulation times, the type of finite elements to be generated and the level of complexity required. There is indeed a trade-off between these topics, which will lead to different meshing options. To solve the different types of problems, we recommend to use one of the following three meshing options: a general meshing strategy, a meshing method that is tailored to specific applications and a third procedure that will be efficient when several patient-specific meshes of the same structure must be generated.

2.5.1 Proposed general image-to-mesh solution

This first approach results form this thesis work and will be presented throughout Chapter 3 and Chapter 4. It has the advantage of being applicable for every biological structure, of any shape and size. It will always produce a valid and visually appealing output that is suitable for further finite element simulations, provided that the parameters of the method are correctly set. The drawbacks of the approach is that it produces tetrahedral meshes only and that it requires prior segmentation of the patient's medical images.

The approach consists in taking path E-G-H-I-J in Figure 2.8, or, equivalently path (a)-(b)-(c)-(d)-(e)-(f) in Figure 2.7. The input data comes from such sources as Magnetic Resonance Imaging (MRI) or Computed Tomography (CT) of biological tissues and consists in a usually anisotropic regular 3D lattice of voxels (Figure 2.7 (a)).

In a first segmentation step the desired structures are delineated in the medical image (Figure 2.7 (b)). Single-material images are segmented into two regions by assigning a zero-value $v = 0$ to background voxels, located outside the structure, and a positive value $v = 1$ to foreground voxels, located in the tissue. On the other hand, biological structures containing $M > 1$ tissues are segmented into $M + 1$ regions: each foreground voxel is assigned a unique label $v = 1, \ldots, M$ according to the tissue in which it is located. As a result, image segmentation provides a *single-label* or *binary* image in the single-material case and a *multi-label* image in the multi-material case.

The points located on the boundary between two segmented regions are then extracted and a local surface orientation is computed at each of these points (Figure 2.7 (c)). For multi-label images the points are distributed into different sets of points according to the material boundary from which they were extracted. These points are called *boundary points* or *extracted points* and constitute a compact description of the segmented data. For the remainder of the algorithm the segmented image is removed from memory and replaced by the extracted points and their associated normals.

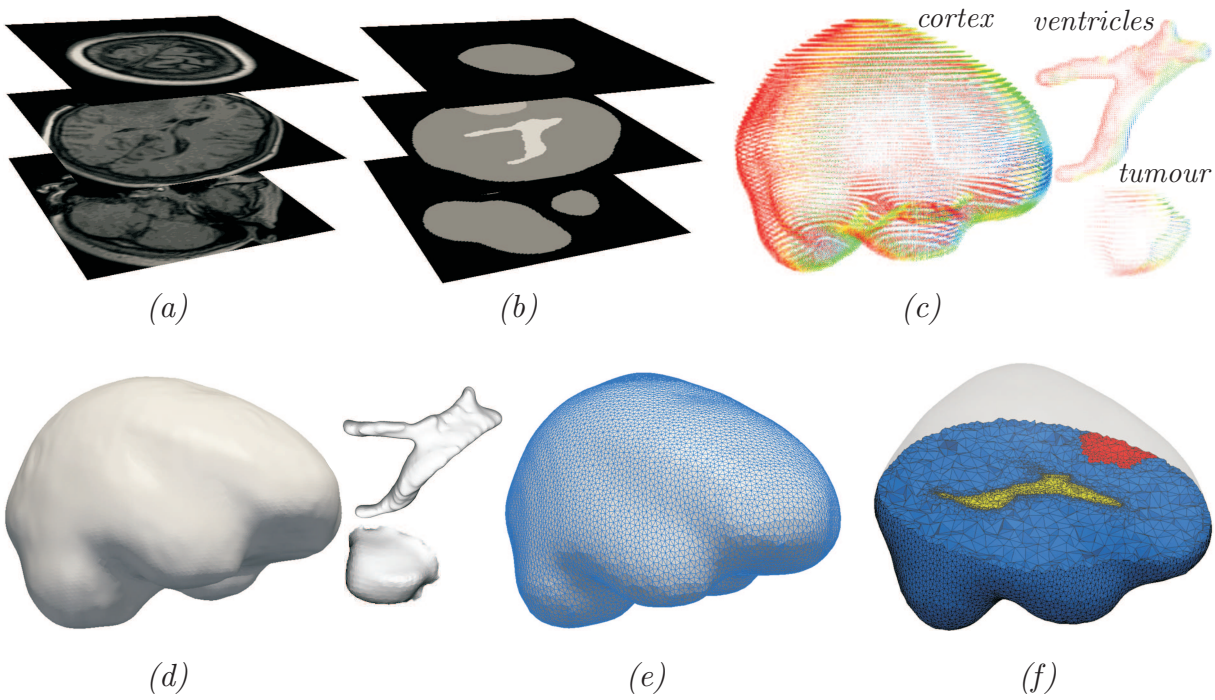


FIGURE 2.7: ***Proposed general image-to-mesh solution.*** *The medical scans (a) are first segmented using a unique label for each tissue composing the structure. In (b), the brain is segmented into healthy brain tissue, tumour and ventricles. For each material region, boundary points and corresponding outward normals are extracted (c). An implicit model is fit to each set of extracted points (d). A multi-material surface mesh is then generated using a multi-material marching tetrahedra method followed by decimation and smoothing algorithms (e). This multi-region surface mesh serves as input to classical volume mesh generators (f).*

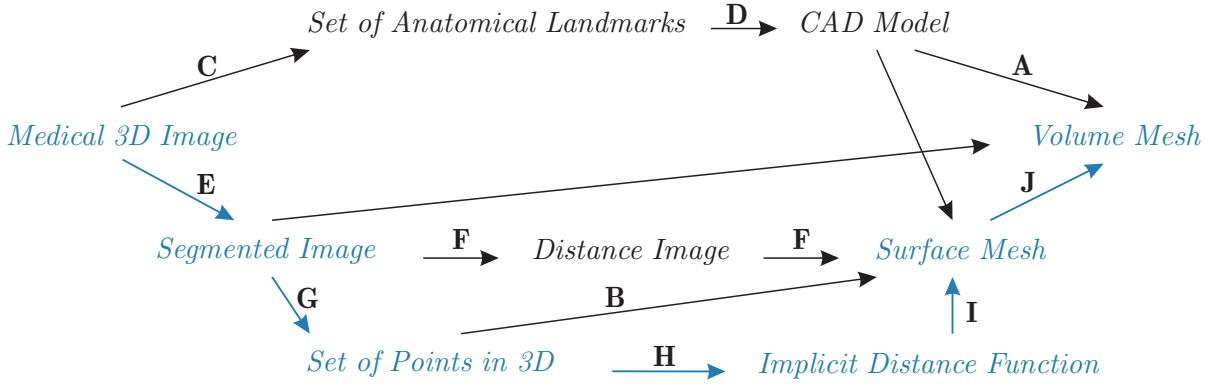


FIGURE 2.8: ***Proposed general image-to-mesh solution.***

From each set of extracted points the proposed surface reconstruction method defines an implicit function $f(\boldsymbol{x})$ that approximates the signed distance to the corresponding boundary surface (Figure 2.7 (d)).

These surfaces are then polygonised using a multi-material marching tetrahedra algorithm. The latter is an extension of the well-known marching tetrahedra algorithm to multiple materials. This means that the algorithm is capable of generating meshes that are constituted of several inner regions, and that these regions join each other in a proper way in the sense of the finite element method, that is to say node on node and edge on edge. As for the traditional marching tetrahedra, our algorithm requires a sampling grid to be defined. In the proposed algorithm however the resolution of this sampling grid may be chosen independently of the original image resolution (Figure 2.7 (e)). The latter is an important feature of our algorithm: the user may define the desired mesh size according to the need of the targeted application rather than the resolution of the medical scans.

Finally, enhanced decimation and smoothing procedures are applied to optimise the triangle count and their quality.

This procedure creates high-quality multi-region surface meshes from which tetrahedral volume meshes may be obtained using Tetgen [158] or Gmsh [73] for example (Figure 2.7 (f)).

2.5.2 Recommended tailor-made image-to-mesh solution

With low quality medical datasets, for atypical patients or for particular tissues, segmentation of the scans can be particularly difficult and time-consuming. In these cases, segmentation may be circumvented by extracting only a set of anatomical reference points from the input data. With the help of these reference points, the user may then build an analytical model of the structure by defining analytical lines and curves and (interpolated) surfaces to recreate the object's boundaries. In the meantime, a way of subdividing the volume into hexahedra may be defined. Truly, the user constructs the model by hand in this procedure, but in the end, a patient-specific hexahedral mesh can be obtained without image segmentation.

However, because only a few points of the model are really extracted from the patient's data, the obtained CAD model is only an approximation of the patient's actual geometry. The quality of the reconstruction is highly dependent on the user's ability and efforts to recreate an anatomically representative structure. Also, the procedure may lack repeatability when the landmarks are not well defined.

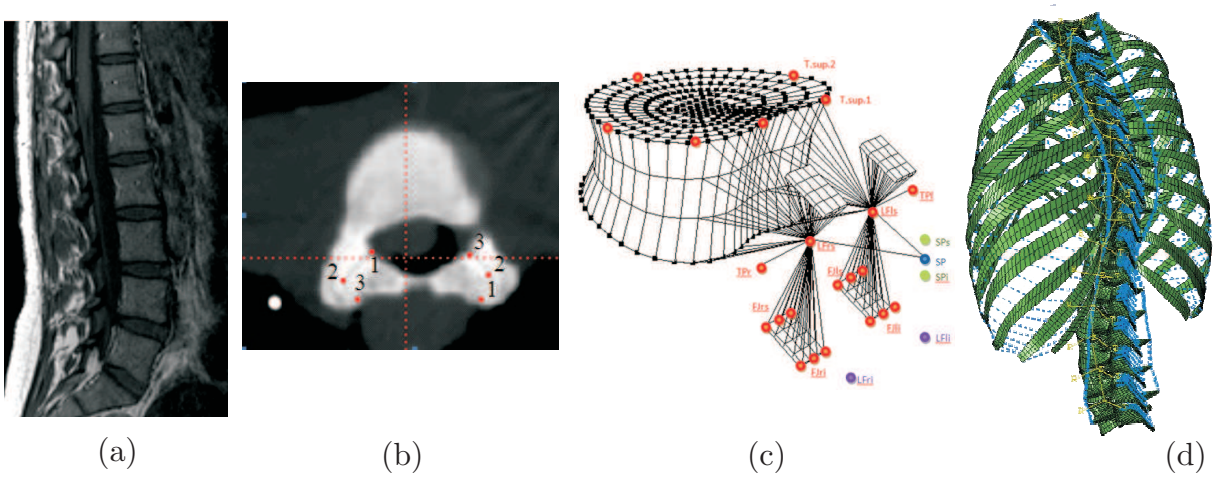


FIGURE 2.9: ***Recommended tailor-made image-to-mesh solution.*** *(a) Magnetic resonance 3D image of the spine. (b) Anatomical landmark selection in the medical dataset. (c) Construction of the geometry and the finite elements, based on the landmarks. (d) Obtained model. Procedure developed by, and images taken from, the Paediatric Spine Research Group, Queensland University of Technology [107].*

This approach may look far more complicated to implement than image segmentation. Indeed, it is. But, as soon as the parametric CAD geometry has been defined, repeating the procedure for other patient's only requires the selection of the new anatomical landmarks. Selecting landmarks will usually be less time-consuming than image segmentation, depending on the geometry, the quality and resolution of the medical scans and the software used.

Finally this approach may be preferred when different types of finite elements should be used to model the geometry, shell or beam elements along with hexahedral elements for example. Because the whole mesh is explicitly defined, adding 1D and 2D elements requires less additional efforts.

This procedure was used successfully by the author within the Paediatric Spine Research Group, Queensland University of Technology, Australia, to construct patient-specific models of the spine. As illustrated in Figure 2.9, vertebrae are discretised using a hexahedral finite elements and beam elements, shell elements are used to model the ribcage and ligaments are modelled with axial connectors. An additional advantage of applying this image-to-mesh technique to the spine is that the same parametric definition can used for all the vertebrae composing the spine, so that the work to create the model is reduced, even for the first model.

2.5.3 Recommended model-based image-to-mesh solution, for repetitive model creation

When the same anatomical structure will be modelled several times, for different patients for example, I would recommend the reader to use mesh morphing techniques. The idea is schematised in Figure 2.10. For the first patient, a reference model is constructed by one or the other above procedures. When creating this reference mesh, a set of anatomical reference points is defined and their position within the final mesh is recorded. For the subsequent patients, these anatomical reference points must be, manually or automatically, selected in the current patient's dataset. A correspondence between the reference geometry and the current, *source*, geometry may then be established and the reference mesh may be translated, scaled and deformed in accordance.

The efficiency of this morphing strategy highly depends on the location of the landmarks and their number. Good locations will ensure the repeatability of the approach. The number of landmarks will determine its accuracy and efficiency. High numbers of landmarks require more user inputs but will generate a more complex morphing deformation.

The author has implemented and used the algorithm presented by Lerios et al. [103] with good results and would recommend using the same algorithm [59].

2.6 Conclusions

This chapter presented several meshing options, with their advantages and disadvantages, and gave some practical recommendations on the meshing strategy to adopt in regard to the targeted application.

The scope of this dissertation is the creation of patient-specific meshes from segmented images. Therefore, it falls into the first of the three above recommended mesh generation strategies. Reasons for this is that patient-specific biomechanical modelling was non-existent at the University of Liège before this thesis. Hence, it was more appropriate to implement a general mesh generation procedure which will, and already has, motivate future research possibilities and collaborations within and outside the University.

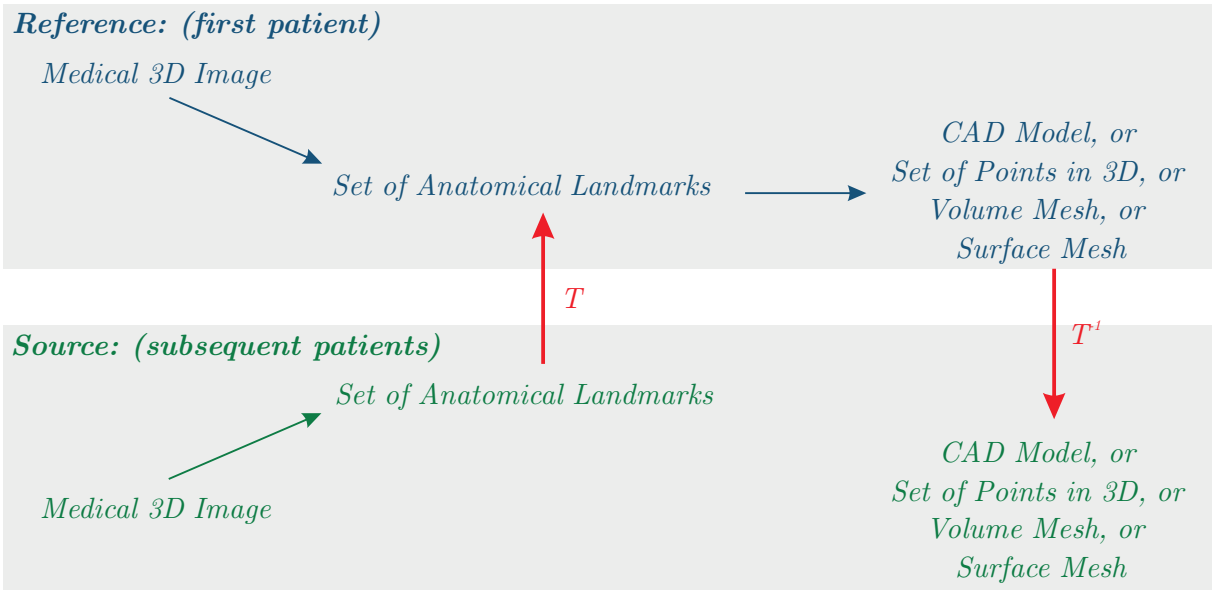


FIGURE 2.10: ***Recommended model-based image-to-mesh solution, for repetitive model creation.*** *Morphing of an existing mesh to create multiple patient-specific meshes. Let us consider that we have a first model of the desired structure and that we have extracted, for this first patient, a set of anatomical landmarks. In order to obtain a finite element model for the subsequent patients, only this set of anatomical reference points must be selected, or automatically detected, in the patient's medical scans. The transformation* $\mathbf{T}$ *that maps this new set of anatomical points into the reference landmarks can be defined. Applying the inverse transformation* $\mathbf{T}^{-1}$ *to the reference finite element model outputs a finite element for the current patient.*

Chapter 3

From segmented 3D images to implicitly defined analytical surfaces

The foundation of our approach to mesh generation from segmented medical images, and its originality, is the addition of a surface reconstruction algorithm to the traditional image-to-mesh pipeline. The advantages of this addition are detailed in Section 3.1. Literature on surface reconstruction methods is reviewed in Section 3.2. From this literature review, the multi-level partition of unity surface reconstruction method, detailed in Section 3.3, was selected and adapted to patient-specific meshing, in Section 3.4, and to multi-material meshing in Section 3.5. Section 3.6 introduces the Taubin and the Hausdorff distance, needed to evaluate the fidelity of the reconstructed surface in relation to the initial dataset. Finally, in Section 3.7, the proposed surface reconstruction method is applied on several types of input datasets and the quality of the obtained analytic surfaces is assessed.

3.1 Motivation

The idea of this chapter is to construct a smooth representation of the object from the discrete, jagged, segmented image and this, prior to mesh generation (surface triangulation). The main advantages for this are (1) aliasing or staircase artefacts are alleviated, (2) the result is more robust to segmentation noise, (3) the user may define the mesh resolution freely, independently of the image resolution, (4) geometric accuracy is ensured to remain unchanged during possible subsequent mesh adaptation steps. These four features are detailed in the subsequent sections.

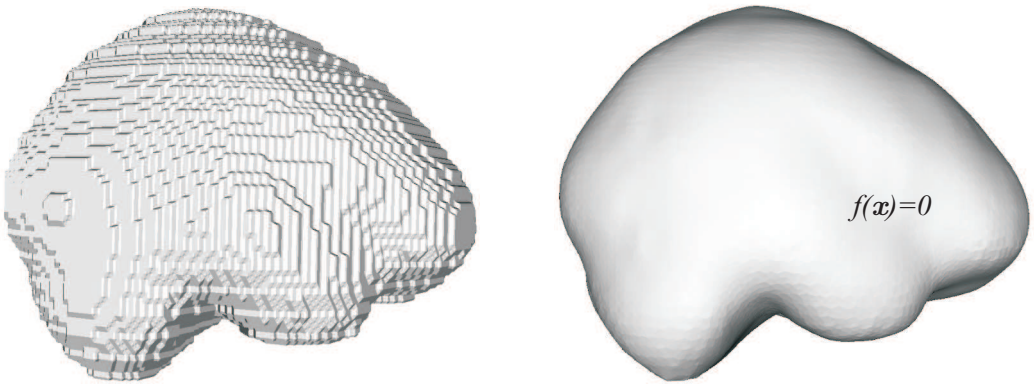


FIGURE 3.1: ***Circumventing aliasing artefacts****. (a) Illustration of the staircase artefacts appearing in meshes generated from segmented data. (b) In this dissertation, these artefacts are avoided by constructing a smooth representation of the segmented volume in the form of an implicit analytic distance function $f(\boldsymbol{x})$ prior to mesh generation.*

3.1.1 Circumventing aliasing artefacts

A common issue in mesh generation from segmented data is the appearance of aliasing artefacts, also known as stair-stepped artefacts. These jagged edges, illustrated in Figure 3.1 (a), are caused by image voxelisation. They are visually unappealing and are unsuited for finite element simulations. A more natural smoothness, qualifying the majority of biological structures, may be recovered either by smoothing the extracted mesh or by filtering the input image. However, mesh smoothing approaches that do not take into account the underlying data cannot ensure an accurate representation of the original volume. Consequently, approaches to smooth the meshes whilst restricting the mesh nodes to remain near the boundary surface defined in the segmented data have been proposed [74, 133]. But, due to this latter constraint, the artefacts are not entirely removed. The second popular strategy is to filter the segmented data typically with a low-pass filter [178]; yet, depending on the used kernel, these methods are either inefficient or they blur away relevant details.

Obtaining a $\mathcal{C}_1$-smooth representation, illustrated in Figure 3.1 (b), of the segmented data prior to mesh generation is our solution to stair-stepped artefacts.

3.1.2 Robustness to segmentation noise

Segmentation is never perfect and often entangled with noise, particularly when this segmentation is done automatically. Of course, filters may be applied in order to smooth the

segmentation and remove outliers, but, as these techniques rarely take into account the underlying images, interesting features may be lost.

3.1.3 Free control of mesh resolution

The addition of an implicit surface extraction algorithm to the image-to-mesh pipeline comes with the replacement of the segmented image with an analytical function $f(\boldsymbol{x})$. Contrarily to a segmented image, where the values of the image voxels can only computed at discrete points, the implicit function $f(\boldsymbol{x})$ is continuously defined. Hence, the distance to the surface to be meshed can be evaluated at each point of the three-dimensional space. This means that the sampling grid used in surface triangulation algorithms can be decoupled from the image spacing. As a result, the user may choose the resolution and level of details of the output model.

3.1.4 Volume-preserving mesh adaptation

Classical smoothing algorithms lead to volume shrinkage and mesh deformation [137]. Because, in our approach, the object's boundary is defined by a continuously differentiable function $f(\boldsymbol{x})$, that may be evaluated at every position in $\Re^3$, a Newton-Raphson iteration scheme may be used at any moment to easily project the mesh nodes on the object's boundary. Thanks to this, a volume-preserving smoothing procedure may be obtained by first applying traditional mesh smoothing algorithms, like the Laplacian smoothing, and second projecting the mesh nodes back on $f(\boldsymbol{x}) = 0$ (Section 4.5).

3.2 Literature review

As introduced above, the topic investigated in this chapter is the addition of an effective surface reconstruction algorithm in the classical image-to-mesh pipeline, after image segmentation and prior to surface triangulation. Research on surface reconstruction approaches originated with the appearance of 3D range scanners and the need to transform the large sets of points generated by these scanners in a computer-friendly format. The latter may consist of a polygonal, often triangular, surface mesh or in an implicit function. Surface reconstruction methods that directly create a surface mesh from the set of points are classified as *explicit*, those that output an implicit function are classified as *implicit*.

3.2.1 Explicit surface reconstruction

Explicit surface reconstruction methods from a set of points, sampled from a 3D surface, output a polygonal mesh from this set of points. The resulting mesh is usually a triangular mesh, representing the initial sampled, or scanned, surface. The two main approaches to generate a polygonal mesh from a set of points are *Delaunay triangulation* and *region-growing algorithms*.

On the one hand, *Delaunay-based* surface reconstruction constructs a Voronoi diagram, or its dual Delaunay triangulation, from the initial cloud of points (Please refer to [20, 139, 146] for more information on Delaunay triangulation and Voronoi diagrams). This produces a partition of the convex hull of the sample points, i.e. the volume of the object of interest, into tetrahedra. A representation of the outer boundary may then be obtained in the form of a triangular surface by identifying the nodes that belong to the surface. The first Delaunay-based reconstruction method was proposed by Boissonnat [21]. Since then many methods have been proposed and some of the most popular are the Power Crust of Amenta et al. [4] and the Robust Cocone of Dey and Goswami [53].

On the other hand, *region-growing* algorithms start from an initial triangle and iterate by attaching new triangles to the region boundary until all points have been processed. The ball-pivoting algorithm of Bernardini et al. [16] is a popular example of region-growing algorithms.

At first glance, these explicit surface reconstruction methods are the solution of choice as they directly produce the triangular mesh we are looking for. However, the major drawback of explicit methods is that they need a dense input set of points with little noise. Hence, they are not adapted to the set of points extracted from images that were segmented using threshold-based segmentation algorithms. Moreover, explicit methods are time-consuming as each input point needs to be considered in turn.

3.2.2 Implicit surface reconstruction

Implicit surface reconstruction methods from a set of points are more and more used because of their robustness with respect to noise and their low computational cost. These methods give a continuous representation of the surface in the form of an implicit function, which is defined as a function $f(\boldsymbol{x})$ that associates for each $\boldsymbol{x} \in \Re^3$ its distance to the surface. Obviously, the surface itself is defined by the zero-level of the function.

3.2.2.1 Global or local ?

Implicit surface reconstruction methods may be classified as *global* or *local*. *Global* methods aim to construct a single function that interpolates or approximates the set of points. In *local* methods, the global function results from the blending of local shape functions, each one of which interpolates or approximates a subset of the input set of points. Global methods can, theoretically, better handle low quality data than local methods. Local methods can generate accurate surface reconstructions in less time than global methods. Hence, there is a trade-off between noise robustness and geometric accuracy. In general, local methods will be preferred for large datasets and a global approach will be used for smaller and sparse sets of points.

3.2.2.2 Interpolating or approximating ?

Implicit surface reconstruction methods may also be classified as *interpolating* or *approximating*, depending on whether the global or local function is constrained to pass through the points, in this case we speak of interpolation, or only in the neighbourhood of the points, in that case we speak of approximation. It is assumed that the geometric approximation error will be less important for interpolating methods than for approximating methods, assuming that the initial set of points is located precisely on the original surface. And, approximation methods should produce a smoother result and be more robust to noise in the input set of points.

3.2.2.3 Popular approaches from literature

Finally, implicit surface reconstruction methods may be classified based on the approach they originate from: approaches based on signed distance functions, approaches based on moving least squares, approaches using Radial Basis Functions, approaches based on the multi-level Partition of Unity and approaches based on deformable models. These five classes of methods are described hereinafter.

Approaches based on signed distance functions One of the first implicit surface reconstruction method has been proposed by Hoppe et al. [84]. Their idea is to locally estimate the signed distance function to the surface by measuring the distance to the tangent plane of the nearest input point. Later on, Curless and Levoy [48] introduced a volumetric method to reconstruct the surface of scanned objects, by first computing the signed distance function of each two-dimensional scan and second averaging the signed distance

functions obtained from each 2D image. A third popular method based on the computation of signed distance functions has been proposed by Boissonnat and Cazals [22]. The level set method [140] is a popular method for calculating the signed distance function, often used for modelling time-varying objects (leading to the approaches described in the last paragraph of this section).

Approaches based on moving least squares The use of moving least squares to interpolate or approximate scattered data was first introduced by Lancaster and Salkauskas [100]. This method essentially consists of a least squares approximating method with local shape control. The algorithm proposed by Shen et al. [154] has been proved to give a good approximation of the signed distance function to the surface [98]. An alternative approach is the projection moving least squares of Levin [104]. The latter has been extensively used by Alexa et al. [3] for point-based modelling and rendering.

Approaches based on Radial Basis Functions The third family of surface reconstruction methods is based on Radial Basis Functions. Again, these Radial Basis functions may have a global [34, 150, 169] or a local [180] support. However, globally supported Radial Basis Functions lead to a dense linear system so that the method becomes prohibitive for large datasets. The use of locally supported Radial Basis Functions within a Partition of Unity framework, a popular method to patch together locally defined functions, has been investigated by Tobor et al. [167] and Ohtake et al. [136]. The latter gives rise to the so called multi-level Partition of Unity approach, explained hereafter.

Approaches based on the multi-level Partition of Unity In the Multi-level Partition of Unity (MPU) approach, a local fit of the surface is obtained by using an octree-based subdivision scheme and approximating the points in each subdivision cell using locally supported quadratic functions [136, 138]. A global solution function is obtained by blending the local solutions functions using smooth local weights that sum up to one everywhere on the domain (see Shepard's blending method [155]). Advantages of the MPU approach are a fast computation, the ability to handle large datasets and a user-controlled geometric approximation error. The main drawback of the method is noise-sensitivity.

Approaches based on deformable models In this last category, an initial surface is defined around the object of interest, represented by the input set of points. This surface is then iteratively deformed towards the set of points until a good approximation of the set of points is obtained. The algorithm is driven by the minimization of an energy func-

tional. Surface reconstruction approaches based on deformable models often use level-set functions [140, 192].

3.2.3 Discussion

In all cases, in order to use one of the above surface reconstruction methods within our image-to-mesh pipeline, a set of points belonging to the boundary of the segmented region will have to be defined. An advantage of replacing the segmented image by a set of 3D points is that the 3D image may then be removed from memory, releasing on average 90% of the computer's memory, depending on the image complexity.

Explicit surface triangulation has the advantage of producing the required result directly. However, each input point will be considered in turn and the number of mesh nodes is directly related to the size of the input set of points. Therefore, if the method used to extract the boundary points from the segmented scan is such that one boundary voxel gives one input point, the time needed for surface triangulation will be directly related to the resolution of the image dataset. The mesh resolution should be determined according to the object's complexity rather than on the parameters of acquisition of the medical images. Therefore points up- and under-sampling may often be required to increase or reduce the density of the input dataset.

Implicit surface reconstruction methods do not feature the above problem: the mesh resolution can be user-chosen. Implicit methods have the additional advantage of, usually, being more robust to noise in the input set of points, and thus, to segmentation. Moreover, implicit surface reconstruction approaches, and local implicit surface reconstruction in particular, is a lot more adapted to large datasets because it is less time- and memory-consuming.

For these reasons, an implicit surface reconstruction method is used in our patient-specific mesh generation algorithm. Amongst the implicit surface reconstruction approaches presented above, the most suited for accurate reconstruction of large datasets is said to be the multi-level Partition of Unity approach [136, 138]. This approach is detailed in the following section.

3.3 The multi-level Partition of Unity surface reconstruction method

In this section the multi-level Partition of Unity (MPU) approach as initially proposed by Ohtake et al. [138] is described in detail. Our contributions to the approach are explained in the next section.

The main idea of the multi-level Partition of Unity approach is, starting from a set of points sampled from a surface, to construct an implicit function $f(\boldsymbol{x})$ that gives an approximated distance to this set of points. This function is constructed by using a recursive, octree-based, subdivision of a cube containing the input points, and approximating the points in each subdivision cell using local quadratic approximating functions. The use of a Partition of Unity approach enables to combine the locally defined approximating functions, using local weights, and obtain a global surface definition $f(\boldsymbol{x}) = 0$.

In other words, the multi-level Partition of Unity surface reconstruction algorithm may be summarised by the following four key points:

1. The use of a Partition of Unity to combine locally defined approximating functions, explained in Section 3.3.1.
2. A recursive octree-based subdivision of space, explained in Section 3.3.2.
3. The definition of smooth local weights, that sum up to one everywhere on the domain, explained in Section 3.3.3.
4. A local approximation of the surface in each subdivision cell, explained in Section 3.3.4.

3.3.1 Partition of unity

Given a set of points $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_n\}$ sampled from a surface and equipped with unit normals $\mathcal{N} = \{\boldsymbol{n}_1, \boldsymbol{n}_2, ..., \boldsymbol{n}_n\}$, the multi-level Partition of Unity surface reconstruction method defines an implicit function $f(\boldsymbol{x})$, that is an approximation of the signed distance from $\mathcal{P}$. This function divides the space into the interior $f(\boldsymbol{x}) > 0$ and the exterior $f(\boldsymbol{x}) < 0$ of the object. The boundary surface thus corresponds to the zero-level of the distance function: $f(\boldsymbol{x}) = 0$.

Globally, a Partition of Unity function is composed of overlapping local approximation functions $Q_i(\boldsymbol{x})$ that are blended together using non-negative compactly supported functions $\phi_i(\boldsymbol{x})$ that sum up to 1 everywhere on a bounded Euclidean domain Ω [9, 66, 156].

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \phi_i(\boldsymbol{x}) Q_i(\boldsymbol{x}) \tag{3.1}$$

where each $\phi_i(\boldsymbol{x})$ is computed as:

$$\phi_i(\boldsymbol{x}) = \frac{w_i(\boldsymbol{x})}{\sum_{j=1}^{N} w_j(\boldsymbol{x})} \quad \Rightarrow \quad \sum_i \phi_i(\boldsymbol{x}) = 1 \qquad \forall \boldsymbol{x} \in \Omega \tag{3.2}$$

where $w_i(\boldsymbol{x})$ is an associated local weight, as will be further explained in Section 3.3.3.

3.3.2 Octree-based subdivision

An octree-based subdivision of space if obtained as follows. First a cube is defined around the object of interest. This cube is then subdivided recursively in a series of eight cells, by subdividing each edge into two, at mid-distance. This recursive subdivision ends locally when a pre-defined criterion is met. In the case of the MPU implicit surface reconstruction method, the subdivision process ends when a sufficiently good geometric approximation of the points located in this subdivision cell has been found. If this approximation error is noted ε, recursive subdivision stops when

$$\varepsilon \leq \varepsilon_{\max} \tag{3.3}$$

In the proposed algorithm, the maximum tolerated approximation error within a subdivision cell, denoted $\varepsilon_{\max}$, is user-defined.

3.3.3 Weight functions

In a Partition of Unity approach, the global implicit function $f(\boldsymbol{x})$ consists in a weighted average of local approximating functions $Q_i(\boldsymbol{x})$. Introducing (3.2) into (3.1), we have

$$f(\boldsymbol{x}) = \frac{\sum_i w_i(\boldsymbol{x}) Q_i(\boldsymbol{x})}{\sum_i w_i(\boldsymbol{x})} \tag{3.4}$$

which shows that the weight $w_i(\boldsymbol{x})$ determines how much the local function $Q_i(\boldsymbol{x})$ influences the global function $f(\boldsymbol{x})$.

In the original implementation of the multi-level Partition of Unity surface reconstruction algorithm, each weight function $w_i(\boldsymbol{x})$ is a quadratic B-spline $b\left(t_i(\boldsymbol{x})\right)$ centred at $\boldsymbol{c}_i$ and having a spherical support, or approximation balls, of radius R_i[1].

$$w_i(\boldsymbol{x}) = b\left(t_i(\boldsymbol{x})\right) \quad \text{where} \quad t_i(\boldsymbol{x}) = \frac{3\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|}{2R_i} \tag{3.5}$$

[1]The notation $\|\boldsymbol{x}\|$ refers to the norm of vector $\boldsymbol{x}$.

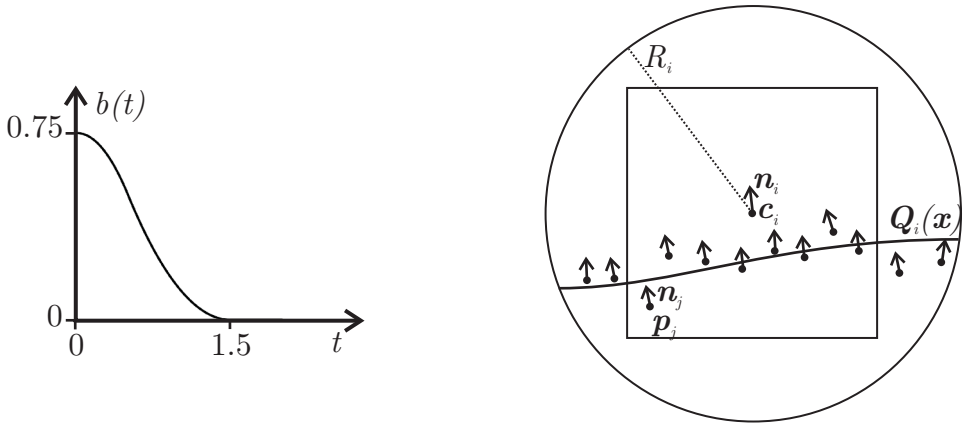


FIGURE 3.2: ***The multi-level Partition of Unity surface reconstruction method****. (a) The quadratic B-spline $b(t)$, defined by* (3.6)*, used as weight function in the MPU method to combine the local approximation functions. (b) Local approximation $Q_i(\boldsymbol{x})$ of a set of points $\boldsymbol{p}_j$ in a subdivision cell of radius R_i centred at $\mathbf{c}_i$.*

where the quadratic B-spline $b(t)$, drawn in two-dimensions in Figure 3.2 (a), is defined by:

$$b(t) = \begin{cases} \frac{3}{4} - |t|^2 & \text{for } |t| \leq \frac{1}{2} \\ \frac{1}{2}\left(|t| - \frac{3}{2}\right)^2 & \text{for } \frac{1}{2} \leq |t| \leq \frac{3}{2} \\ 0 & \text{for } |t| > \frac{3}{2} \end{cases} \tag{3.6}$$

3.3.4 Quadratic local approximation

The MPU approach uses an octree-based subdivision process to adapt to the local geometry. The algorithm starts by rescaling the point cloud $\mathcal{P}$ so that it fits into an axis-aligned bounding cube with a main diagonal of unit length. Then, this bounding cube, or approximation cube, is iteratively divided into $2\times2\times2 = 8$ cells, using the octree-based subdivision scheme explained in Section 3.3.2. Each generated cell i is characterised by its centre $\mathbf{c}_i$ and the length of its main diagonal d_i. Moreover, the support radius R_i of the weight function $w_i(\boldsymbol{x})$, defined in (3.5), is defined as a multiple of the diagonal d_i:

$$R_i = \alpha\, d_i \tag{3.7}$$

where α is user-defined, but comprised between 0.75 and 0.9, which permits an overlapping of adjacent spherical supports (approximation balls) [138]. Figure 3.2(b) illustrates a subdivision cell with its associated approximation ball.

The set of points located in the sphere of radius R_i and centred at $\boldsymbol{c}_i$:

$$\mathcal{P}_i = \left\{ \boldsymbol{p}_j : p_j \in \mathcal{P}, \left\| \boldsymbol{p}_i - \boldsymbol{c}_i \right\| < R_i \right\} \tag{3.8}$$

is used to define a local shape function $Q_i(\boldsymbol{x})$, which, in the original paper, is a quadratic function [138]. The latter is built so as to approximate the set of points $\mathcal{P}_i$ in a least squares sense. This process is illustrated in Figure 3.2(b).

The number of points enclosed in the approximation cube should be greater than the number of coefficients of the quadratic function, which is 10 for a general three-dimensional quadratic function (see Section 3.3.4.2). More points will produce a smoother result. Indeed, when the quadratic function is defined to approximate a greater set of points, the initial surface sampled with the set of input points is approximated, i.e. simplified, to a greater extent, and thus smoother. In the algorithm, the minimum number of points required in a subdivision cell, denoted $N_{\min}$, is user-defined. When the approximation cube does not contain enough points, the radius of corresponding approximation sphere is increased until this minimum is obtained, and the cell will not be further subdivided.

Ohtake et al. [138] advise using one of the three following local approximation functions, depending on the distribution of the points $\mathcal{P}_i$ in the subdivision cell:

1. a general three-dimensional quadric,
2. a bivariate quadric polynomial in local coordinates,
3. a piecewise quadric surface that fits an edge or a corner.

Given that we aim at representing biological structures, which do generally not present sharp edges, only the general quadric and the bivariate quadric polynomials will be used in our implementation, and presented here.

The choice between both functions is determined by the distribution of the point normals in the cell. The latter must be provided as an input of the algorithm: each input point should be associated with a normal $\boldsymbol{n}_j$. An average normal $\bar{\boldsymbol{n}}_i$ for the subdivision cell i is first computed:

$$\bar{\boldsymbol{n}}_i = \frac{1}{\left|\mathcal{P}_i\right|} \sum_{\boldsymbol{p}_j \in \mathcal{P}_i} \boldsymbol{n}_j \tag{3.9}$$

where $\left|\mathcal{P}_i\right|$ denotes the size of the point set $\mathcal{P}_i$, defined in (3.8).

When the maximum deviation of normals to the average normal direction in the cell $\bar{\boldsymbol{n}}_i$, defined as the angle between the averaged normal $\bar{\boldsymbol{n}}_i$ and the point normals $\boldsymbol{n}_j$, is more than $\pi/2$, a general 3D quadric is used. Otherwise, a bivariate quadratic polynomial employed.

How to, from a set of input points, define these two types of quadratic functions, least squares approximation of the input points, is presented below.

3.3.4.1 Fitting of a bivariate quadric

Let us consider the following general expression of a bivariate quadric:

$$Q_i(r,s,t) = t - (a_1 r^2 + 2a_2 rs + a_3 s^2 + a_4 r + a_5 s + a_6) \tag{3.10}$$

where (r,s,t) are the local coordinates of subdivision cell i centred at $\boldsymbol{c}_i$ and such that t is the direction indicated by the normal of the subdivision cell $\bar{\boldsymbol{n}}_i$. To facilitate further developments we rewrite the bivariate quadric as:

$$Q_i(r,s,t) = t - \sum_{k=1}^{K} a_k \pi_k \tag{3.11}$$

with $K = 6$ and where we have defined

- the vector of basis functions $\boldsymbol{\pi} = [r^2, rs, s^2, r, s, 1]$
- the vector of coefficients $\boldsymbol{a} = [a_1, a_2, a_3, a_4, a_5, a_6]$

Similarly to the global function $f(\boldsymbol{x})$, the local functions $Q_i(\boldsymbol{x})$ give, at each point of the three-dimensional space, an approximation to the input set of points. Indeed, $Q_i(\boldsymbol{x})$ is a *distance function*, that gives the distance to the local set of points $\mathcal{P}_i$. In the specific case where the approximation is exact we speak of interpolation and Q_i evaluated at an input point $\boldsymbol{p}_j \in \mathcal{P}_i$ is zero, i.e. $Q_i(\boldsymbol{p}_j) = 0$. Therefore, the value of function Q_i at the extracted point, gives a measure of the approximation error. In a least squares fitting procedure, the coefficients a_k, in (3.10) and (3.11), are determined so as to minimise the weighted sum of the squares of the errors $Q_i(\boldsymbol{p}_j)$ at the points $\boldsymbol{p}_j$. Putting this in equation, the following objective function must be minimized[2]:

$$f_{\text{obj},i} = \sum_{p_j \in \mathcal{P}_i} w_i(\boldsymbol{p}_j)\, Q_i(\boldsymbol{p}_j)^2 \tag{3.12}$$

where the indice i corresponds to the i-th subdivision cell, $\mathcal{P}_i$ is the set of points located in subdivision cell i, the weights w_i are defined by (3.5) and Q_i is given by (3.11).

The minimum of the objective function in local coordinates is found by setting the corresponding gradient to zero.

$$\frac{\partial f_{\text{obj},i}}{\partial a_k} = 0, \quad k = 1, \ldots K \tag{3.13}$$

This minimisation results in K linear equations to determine the K parameters a_k:

$$2 \sum_{\boldsymbol{p}_j \in \mathcal{P}_i} w_i(\boldsymbol{p}_j)\, Q_i(\boldsymbol{p}_j)\, \frac{\partial Q_i}{\partial a_k}(\boldsymbol{p}_j) = 0, \quad k = 1, \ldots K \tag{3.14}$$

[2]Please distinguish the notation f_{obj} referring to an objective function and the MPU implicit function $f(\boldsymbol{x})$ defined in Equation (3.1).

or, replacing $Q_i(\boldsymbol{p}_j)$ by its expression (3.11) and since $\partial Q_i/\partial a_k = \pi_k(\boldsymbol{p}_j)$,

$$\sum_{\boldsymbol{p}_j \in \mathcal{P}_i} w_i(\boldsymbol{p}_j) \left(t(\boldsymbol{p}_j) - \sum_{l}^{K} a_l \pi_l(\boldsymbol{p}_j) \right) \pi_k(\boldsymbol{p}_j) = 0, \quad k = 1, \dots K \tag{3.15}$$

$$\sum_{\boldsymbol{p}_j \in \mathcal{P}_i} w_i(\boldsymbol{p}_j) \left(\sum_{l}^{K} a_l \pi_l(\boldsymbol{p}_j) \right) \pi_k(\boldsymbol{p}_j) = \sum_{p_j \in \mathcal{P}_i} w_i(\boldsymbol{p}_j) t(\boldsymbol{p}_j) \pi_k(\boldsymbol{p}_j), \quad k = 1, \dots K \tag{3.16}$$

with $K = 6$ for a bivariate quadric.

Writing this in matrix form, we have

$$\left[\left(\boldsymbol{w}_i \boldsymbol{\Pi} \right)^T \boldsymbol{\Pi} \right] \boldsymbol{a} = \left(\boldsymbol{w}_i \boldsymbol{\Pi} \right)^T \boldsymbol{t} \tag{3.17}$$

where

- $\boldsymbol{t} = \left\{ t(\boldsymbol{p}_j) \right\}$ is the vector containing the t-coordinate of the input points in local coordinates;
- $\boldsymbol{\Pi} = \left\{ \pi_{jk} \right\}$ with $\pi_{jk} = \pi_k(\boldsymbol{p}_j)$, $k = 1, \dots K, K = 6$ and $\boldsymbol{p}_j \in \mathcal{P}_i$ contains the value of the basis functions evaluated at each point;
- $\boldsymbol{w}_i = w_i(\boldsymbol{p}_j)$ is the vector containing the weights (3.12) of the weighted minimization procedure at each point .

Finally, the unknowns $\boldsymbol{a}$ are found by solving the following system of equations

$$\boldsymbol{a} = \left[\left(\boldsymbol{w}_i \boldsymbol{\Pi} \right)^T \boldsymbol{\Pi} \right]^{-1} \left(\boldsymbol{w}_i \boldsymbol{\Pi} \right)^T \boldsymbol{t} \tag{3.18}$$

3.3.4.2 Fitting of a general quadric

A general quadratic function in the three-dimensional space can be expressed as

$$Q_i(x, y, z) = a_1 x^2 + a_2 y^2 + a_3 z^2 + a_4 xy + a_5 xz + a_6 yz + a_7 x + a_8 y + a_9 z + a_{10} \tag{3.19}$$

Similarly to the above procedure, this equation may be rewritten in the form:

$$Q_i(x, y, z) = \sum_{k=1}^{K} a_k \pi_k \tag{3.20}$$

where we have defined

- the vector of basis functions $\boldsymbol{\pi} = [x^2, y^2, z^2, xy, xz, yz, x, y, z, 1]$

- the vector of unknowns coefficients $\boldsymbol{a} = [a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}]$

Ohtake et al. [138] suggest to use a set of auxiliary points $\{\boldsymbol{q}_l\}$ to help orient the local function. With this aim in mind, reliable estimates are selected amongst the points $\boldsymbol{q}_l$ located on the corners and at the centre of the subdivision cell. First, the six nearest neighbours $\boldsymbol{p}_m$ of $\boldsymbol{q}_l$ in $\mathcal{P}_i$ are taken. Then, the six scalar products between the vector connecting $\boldsymbol{p}_m$ and $\boldsymbol{q}_l$ and the normal $\boldsymbol{n}_m$ at $\boldsymbol{p}_m$ are computed:

$$s_m(\boldsymbol{q}_l) = \boldsymbol{n}_m \cdot (\boldsymbol{p}_m - \boldsymbol{q}_l), \quad m = 1, \ldots, 6 \tag{3.21}$$

The point $\boldsymbol{q}_l$ is added to the set of reliable estimates $\{\boldsymbol{q}_l\}$ if and only if all its six scalar products $s_m(\boldsymbol{q}_l)$ have the same sign, meaning that all six neighbours $\boldsymbol{p}_m$ have a normal $\boldsymbol{n}_m$ that points towards the direction of $\boldsymbol{q}_l$. If the latter is not satisfied the auxiliary point $\boldsymbol{q}_l$ is not used.

The objective function (3.12) becomes:

$$f_{\text{obj},i} = \frac{1}{\sum_{\boldsymbol{p}_j \in \mathcal{P}_i} w_i(\boldsymbol{p}_j)} \sum_{\boldsymbol{p}_j \in \mathcal{P}_i} w_i(\boldsymbol{p}_j) Q_i(\boldsymbol{p}_j)^2 + \frac{1}{L} \sum_{\boldsymbol{q}_l \in \{\boldsymbol{q}_l\}}^{L} \left(Q_i(\boldsymbol{q}_l) - d_l(\boldsymbol{q}_l)\right) \tag{3.22}$$

where $L \leq 9$ is the size of the set of reliable auxiliary points $\{\boldsymbol{q}_l\}$ and $d_l(\boldsymbol{q}_l)$ is the average of the computed scalar products between the auxiliary point l and its six neighbours:

$$d_l(\boldsymbol{q}_l) = \frac{1}{6} \sum_{m=1}^{6} s_m(\boldsymbol{q}_l) \tag{3.23}$$

As above, the unknown parameters $\boldsymbol{a}$ in (3.19) and (3.20) are determined by setting the objective function $f_{\text{obj},i}$ with respect to the K unknowns a_k to zero, which gives the $K = 10$ gradient equations:

$$\frac{\partial f_{\text{obj},i}}{\partial a_k} = 0, \quad k = 1, \ldots K \tag{3.24}$$

3.3.5 Parameters

In the original implementation of MPU surface reconstruction three parameters are user-defined:

1. the minimum number of points required in a subdivision cell $N_{\min}$,
2. the approximation error tolerance $\varepsilon_{\max}$,
3. the ratio of approximation ball radius to subdivision cell diagonal α.

TABLE 3.1: ***Multi-level Partition of Unity approach****. Bounds of the* lion-dog *statue dataset.*

$x_{\min}$	$x_{\max}$	$y_{\min}$	$y_{\max}$	$z_{\min}$	$z_{\max}$
-9.18104 mm	9.24407 mm	-9.97299 mm	9.96796 mm	1.48438 mm	18.5167 mm

The influence of these parameters on the obtained surface are illustrated on a set of 99977 points sampled from a *lion-dog* statue (Figures 3.3 and Figure 3.4). The point cloud is courtesy of Dr. A. Belyaev of the Max-Planck-Institut für Informatik in Germany [189], but the reconstructions were done with our implementation of the algorithm. The dimensions of the *lion-dog* statue are indicated in Table 3.1.

Parameter $\varepsilon_{\max}$ gives the acceptable distance between the reconstruction surface and the sampled boundary points (Section 3.6). In the algorithm, a subdivision cell is divided until the local shape function $Q_i(\boldsymbol{x})$ approximates the local boundary points better than the defined tolerance: $\varepsilon < \varepsilon_{max}$. Figure 3.3 shows the computed results for $\varepsilon_{max} = 0.8$ mm and $\varepsilon_{max} = 8$ mm, and with $N_{\min} = 30$ and $\alpha = 0.75$. These values should be viewed in regard of the datasets dimensions (Table 3.1) and the number of input points (99977). The illustrations show that increasing this error tolerance relaxes the surface shape, but increases the approximation error.

Parameter $N_{\min}$ is the minimum number of points required in a subdivision cell. Figure 3.4 presents the obtained results for three values of $N_{\min}$ and for $\varepsilon_{\max} = 0.08$ and $\alpha = 0.75$. As illustrated, increasing $N_{\min}$ has the effect of smoothing the surface representation.

Finally, parameter $\alpha \in [0.75, 0.9]$ defines how much adjacent approximation balls are overlapping each other (see Equation (3.7) and Figure 3.2(b)). Increasing this parameter increases the computation times and gives a smoother result. This parameter will always be fixed to 0.75 in our algorithm.

3.4 Our contributions to the multi-level Partition of Unity surface reconstruction method

Our original contributions to the Partition of Unity approach for implicit surface reconstruction are fourfold. First, an efficient way to extract the input sets of points and normals from segmented medical data is presented. Second, weight function are analysed and the pos-

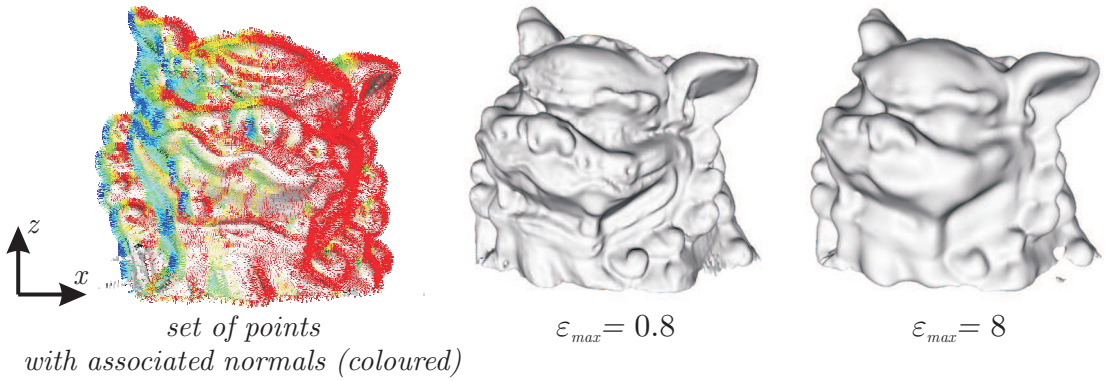


FIGURE 3.3: ***Multi-level Partition of Unity surface reconstruction of the lion-dog statue.*** *Influence of parameter ε_{max}, with $N_{min} = 30$ and $\alpha = 0.75$.*

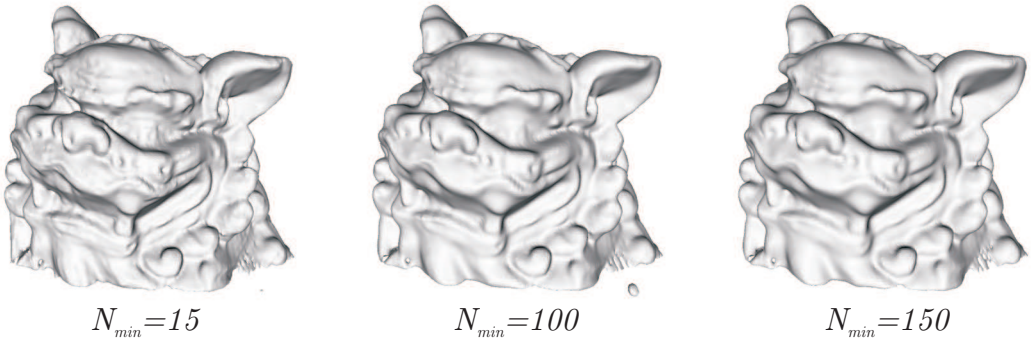


FIGURE 3.4: ***Multi-level Partition of Unity surface reconstruction of the lion-dog statue.*** *Influence of parameter N_{min}, with $\varepsilon_{max} = 0.08$ and $\alpha = 0.75$.*

sibility to interpolate rather than approximate the points is added. Third, linear instead of quadratic local approximating functions are used to increase the robustness of the method to noise. Finally, we propose an efficient strategy to represent objects containing multiple domains with a set of implicit functions.

3.4.1 Extraction of boundary points and normals from segmented data

The multi-level Partition of Unity approach requires a set of points $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_n\}$ sampled from a surface in 3D and corresponding normals $\mathcal{N} = \{\boldsymbol{n}_1, \boldsymbol{n}_2, ..., \boldsymbol{n}_n\}$ to be defined. The method was initially developed in the field of Computer Graphics, where the set of points $\mathcal{P}$ is obtained by laser scanners, mechanical touch probes and computer vision techniques such as depth from stereo [138]. In our case, the input data is a segmented 3D volume and a procedure for automatically extracting the boundary points and normals from this

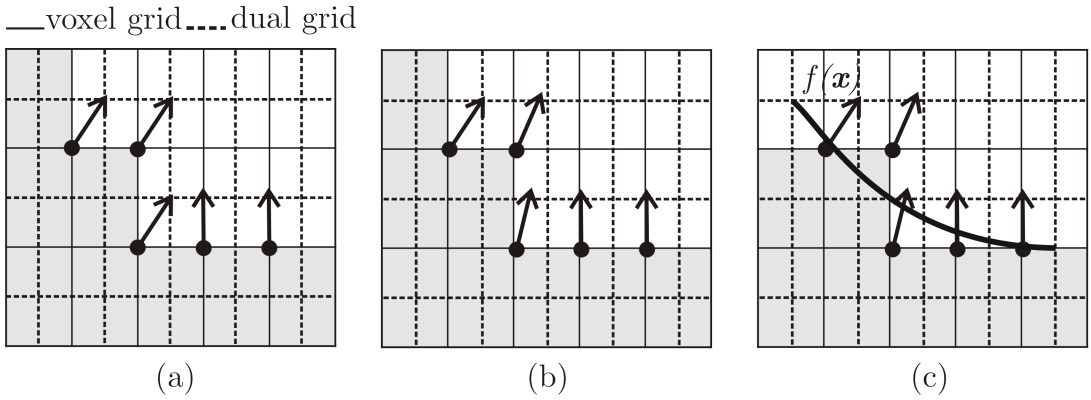


FIGURE 3.5: ***Extraction of boundary points and normals from segmented data.*** *The segmented material region is represented in grey, whereas the background is white. (a) Points are created on the dual grid, represented with dashed lines, at the centre of the grid cells that are located in both foreground and background. A surface orientation vector is computed for each point. (b) Normals are subsequently smoothed to increase the quality of the reconstructed surface. (c) Based on these boundary points and normals, an MPU implicit function $f(\boldsymbol{x})$ is defined.*

image must be developed [29]. The adopted strategy is first explained in the case of a single-material image. The general case of a multi-label image is considered in Section 3.5.

A smoother result is obtained by considering the dual grid rather than the image voxel grid to extract the required boundary points [75]. As illustrated by the dotted lines in Figure 3.5, this dual grid is obtained by shifting the image hexahedral grid by half a voxel spacing in each direction. The grid is processed by taking each grid cell in turn. If the eight voxel values evaluated at the dual cell corners are identical, the cell is declared to be located inside or outside the object. However, if one or more values are different from their neighbours, the cell crosses the object's boundary. In that case, a boundary point is created at the centre of the dual grid cell, indicating that the object surface lies near this point (Figure 3.5 (a)). The associated boundary orientation is evaluated as the sum of the directions to the background voxels (detected as voxels with an associated value of 0) in the cell.

$$\boldsymbol{n}_{i_0,j_0,k_0} = \sum_{i=i_0}^{i_0+1}\sum_{j=j_0}^{j_0+1}\sum_{k=k_0}^{k_0+1} \begin{bmatrix} i-i_0-0.5 \\ j-j_0-0.5 \\ k-k_0-0.5 \end{bmatrix}_{v_{i,j,k}=0} \tag{3.25}$$

As medical images are generally anisotropic, this normal must be divided by the voxel width in each direction $(\Delta x, \Delta y, \Delta z)$. The latter is then normalised because we are only

interested in its orientation.

$$\boldsymbol{n} = \begin{bmatrix} n_x/\Delta x \\ n_y/\Delta y \\ n_z/\Delta z \end{bmatrix}, \quad \bar{\boldsymbol{n}} = \frac{\boldsymbol{n}}{\|\boldsymbol{n}\|} \tag{3.26}$$

Finally, the normals are iteratively smoothed to augment the quality of the reconstructed surface (Figure 3.5 (b)).

$$\boldsymbol{n}_{i,k+1} = \bar{\boldsymbol{n}}_{i,k} + \frac{1}{N} \sum_{\substack{j \\ |\boldsymbol{p}_j - \boldsymbol{p}_i| < r}}^{N} (\bar{\boldsymbol{n}}_{j,k} - \bar{\boldsymbol{n}}_{i,k}) \quad ; \quad \bar{\boldsymbol{n}}_{k+1} = \frac{\boldsymbol{n}_{k+1}}{\left\|\boldsymbol{n}_{k+1}\right\|} \tag{3.27}$$

The new normal orientation $\bar{\boldsymbol{n}}_{k+1}$ is related to the present normal orientation $\bar{\boldsymbol{n}}_k$ and an average orientation of adjacent normals. We take $r = 1.1\ \Delta z$ so that the nearest points of the upper and the lower slice are taken into account even for highly anisotropic grids[3]. In practice, one iteration is sufficient to obtain good results for the surface reconstruction algorithm.

3.4.2 Interpolating weight functions

The weight function proposed by Ohtake et al. [138] is the quadratic b-spline, centred at the centre of the subdivision cell $\boldsymbol{c}_i$ and having a spherical support R_i. As already presented in Section 3.3.3, the weight functions are computed by

$$w_i(\boldsymbol{x}) = \begin{cases} \frac{3}{4} - \left| \frac{3\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|}{2R_i} \right|^2 & \text{for } \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\| \leq \frac{R_i}{3} \\ \frac{1}{2} \left(\left| \frac{3\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|}{2R_i} \right| - \frac{3}{2} \right)^2 & \text{for } \frac{R_i}{3} < \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\| \leq R_i \\ 0 & \text{for } \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\| > R_i \end{cases} \tag{3.28}$$

Other options are possible for the weight functions w_i. The choice between these options is important, as it determines the quality and the smoothness of the global function. And, by choosing adequate weight functions, the global implicit function may either behave as an interpolating function or as an approximating function. An interpolation of the input data points may be obtained by replacing the above weight functions (3.28) with the inverse-distance singular weights proposed by Franke and Nielson [66]:

$$w_i(\boldsymbol{x}) = \begin{cases} \left(\frac{R_i - \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|}{R_i \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|} \right)^2 & \text{for } \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\| \leq R_i \\ 0 & \text{for } \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\| > R_i \end{cases} \tag{3.29}$$

[3]The z-direction is defined as the scanning direction.

The effects of using interpolating weight functions rather than approximating ones will be illustrated at the end of this chapter, Section 3.7. It is assumed that interpolating weight functions will give a geometrically more accurate result than approximating weights, at least for small sets of points $\mathcal{P}_i$.

3.4.3 Linear local approximation

The pioneers of the MPU approach, Ohtake et al. [138], propose to approximate the local set of points with quadratic functions. But, the main criticism of their approach is its sensitivity to noise. In response to this observation, we investigated whether noise robustness could be achieved using linear instead of quadratic local approximating functions:

$$L(x,y,z) = a_1 x + a_2 y + a_3 z + a_4 \tag{3.30}$$

The idea is the following. The linear shape function[4] $L(\boldsymbol{x})$should be a good approximation of the Euclidean distance field $D(\boldsymbol{x})$ near the input points $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_n\}$. Remembering that these input points are equipped with normals $\mathcal{N} = \{\boldsymbol{n}_1, \boldsymbol{n}_2, ..., \boldsymbol{n}_n\}$, the distance field near a sample point $\boldsymbol{p}_j$ may be approximated by:

$$D_{\boldsymbol{p}_j}(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{p}_j)^T \cdot \boldsymbol{n}_{\boldsymbol{p}_j} \tag{3.31}$$

Using a least squares error metric, the error between the linear shape function $L(\boldsymbol{x})$ and the distance field $D(\boldsymbol{x})$ is expressed as:

$$E_{LS} = \sum_{\boldsymbol{p}_j \in P} \Big(L(\boldsymbol{x}) - D_{\boldsymbol{p}_j}(\boldsymbol{x}) \Big)^2 \tag{3.32}$$

In the weighted least squares approach, weights $w(\boldsymbol{p}_j)$ are added to the error measurement (3.32) to give a greater importance is given to the points located near the evaluation point $\boldsymbol{x}$:

$$\sum_{\boldsymbol{p}_j \in P} E_{LS} = \Big(L(\boldsymbol{x}) - D_{\boldsymbol{p}_j}(\boldsymbol{x}) \Big)^2 w(\boldsymbol{p}_j) \tag{3.33}$$

where the weights $w(\boldsymbol{p}_j)$ will still be computed by (3.5) and (3.6) or, equivalently by (3.28), if approximating is desired and by (3.29) if interpolation of the data is the goal.

An expression for the coefficients of a general linear function (3.30) is found by first considering:

$$L(\boldsymbol{x}) = c_0 \tag{3.34}$$

[4]The index i referring to the current subdivision cell is omitted in this section to simplify the notations. Therefore one should read $L_i(\boldsymbol{x})$ for the linear function as well as $w_i(\boldsymbol{p}_j)$ for the local weights.

Introducing (3.34) into (3.33), we obtain the following minimization problem:

$$\min \sum_{\boldsymbol{p}_j \in P} \left(c_0 - D_{\boldsymbol{p}_j}(\boldsymbol{x}) \right)^2 w(\boldsymbol{p}_j) \tag{3.35}$$

This problem is solved by setting the gradient of the objective function with respect to c_0 to zero:

$$\begin{aligned} 2 \sum_{\boldsymbol{p}_j \in P} \left(c_0 - D_{\boldsymbol{p}_j}(\boldsymbol{x}) \right) w(\boldsymbol{p}_j) = 0 \\ c_0 \sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j) - \sum_{\boldsymbol{p}_j \in P} D_{\boldsymbol{p}_j}(\boldsymbol{x}) w(\boldsymbol{p}_j) = 0 \end{aligned} \tag{3.36}$$

Re-arranging the terms and taking account of (3.34), we obtain the following expression for the local shape function

$$L(\boldsymbol{x}) = c_0 = \frac{\sum_{\boldsymbol{p}_j \in P} D_{\boldsymbol{p}_j}(\boldsymbol{x}) w(\boldsymbol{p}_j)}{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j)} \tag{3.37}$$

replacing function $D_{\boldsymbol{p}_j}(\boldsymbol{x})$, that gives the Euclidean distance from a sample point $\boldsymbol{p}_j$, by its expression (3.31), we have:

$$L(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{p}_j \in P} (\boldsymbol{x} - \boldsymbol{p}_j)^T \cdot \boldsymbol{n}_{\boldsymbol{p}_j} w(\boldsymbol{p}_j)}{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j)} \tag{3.38}$$

which may be rewritten in the polynomial form:

$$L(x, y, z) = a_1 x + a_2 y + a_3 z + a_4 \tag{3.39}$$

with the following expression for the coefficients:

$$a_1 = \frac{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j) n_{x,j}}{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j)} \tag{3.40}$$

$$a_2 = \frac{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j) n_{y,j}}{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j)} \tag{3.41}$$

$$a_3 = \frac{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j) n_{z,j}}{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j)} \tag{3.42}$$

$$a_4 = -\frac{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j)\, \boldsymbol{p}_j \cdot \boldsymbol{n}_j}{\sum_{\boldsymbol{p}_j \in P} w(\boldsymbol{p}_j)} \tag{3.43}$$

Therefore, compared to the quadratic functions proposed in the initial MPU implementation (Section 3.3.4), there is no need for solving a linear system of the type $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$. This

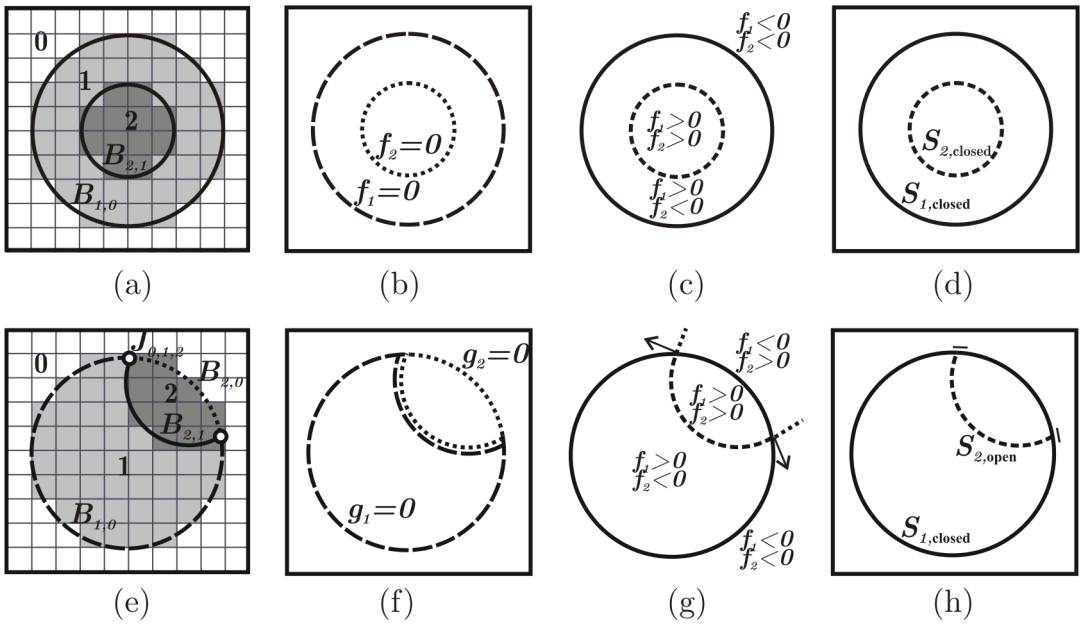


FIGURE 3.6: ***Implicit representation of multi-label datasets****. (a) and (e) Segmented images. Regions 1 and 2 are disjoint in (a) whereas a material junction $J_{0,1,2}$ exists in (e). (b) and (f) Result obtained when a closed distance function is defined separately for each tissue. This approach is not adequate for images containing junction-lines. Indeed boundary $B_{2,1}$ is defined twice in (f), which may create inconsistent meshes. (c) and (g) In this work each tissue boundary is represented by the zero level of a unique distance function $f(\boldsymbol{x})$. (d) and (h) This requires the definition of* closed *and* open *surfaces.*

leads to a gain in computer time as compared to the use of quadratic functions. Also, as will be illustrated in Section 3.7, the resulting approach will be more robust and less sensitive to noise than the original algorithm. For these reasons, linear functions will be preferred for large noisy input datasets.

3.5 Implicit representation of multi-label datasets

The above procedure cannot be easily extended to the multi-material case. Indeed, when the labelled dataset contains more than two distinct labels, i.e. more than one foreground label, a single implicit function $f(\boldsymbol{x})$ is not sufficient to describe the whole multi-region system.

When the tissues labelled in the segmented dataset have *separate boundaries*, like in Figure 3.6 (a), the extension to the multi-material case is straightforward: the surface extraction scheme is simply repeated for each new material domain [177]. In this way,

a first implicit function $f_1(\boldsymbol{x})$ is created to represent the material 1. This function takes positive values in the region with label 1 and negative values outside. A second function $f_2(\boldsymbol{x})$ is then defined to represent the second material region, labelled 2. As a result, the boundaries $B_{1,0}$ and $B_{2,1}$ are represented by the zero-levels of the MPU functions $f_1(\boldsymbol{x}) = 0$ and $f_2(\boldsymbol{x}) = 0$ respectively. The classical single-material marching tetrahedra algorithm is extended to be able to triangulate both boundary surfaces at once: both implicit functions are evaluated at each grid vertex and a triangulation is built when one of the functions changes sign.

The above procedure is not adequate to obtain an analytical description of more general multi-material structures in which *three or more tissues join each other*. For the example shown in Figure 3.6 (e) two functions will be created with the classical procedure [177]: $g_1(\boldsymbol{x})$ represents material 1 and $g_2(\boldsymbol{x})$ represents material 2. As illustrated in Figure 3.6 (f), the inner boundary $B_{2,1}$ separating regions 2 and 1 is defined twice, by both $g_1(\boldsymbol{x}) = 0$ and $g_2(\boldsymbol{x}) = 0$. These two functions will most likely not coincide exactly, thus creating voids and overlays. Although this procedure is sufficient for surface rendering and visualisation, it is not adequate to create valid contiguous finite element meshes. This example leads us to deduce that each material boundary $B_{1,0}$, $B_{2,0}$ and $B_{2,1}$ composing the multi-region system should be defined by the zero-level of a unique implicit function. Moreover, the junctions $J_{0,1,2}$ between these boundaries should be well defined.

We solve this problem by describing a junction-line J as the intersection between an *open* and *closed* surface. In Figure 3.6 (e), the inner boundary $S_{2,\text{open}} = B_{2,1}$ is attached to the exterior boundary $S_{1,\text{closed}} = B_{1,0} \cup B_{2,0}$. Therefore, the inner boundary $S_{2,\text{open}}$, separating regions 1 and 2, is considered as *open* with respect to the *closed* outer boundary $S_{1,\text{closed}}$ in our algorithm. During the image processing step, points located on the outer boundary $S_{1,\text{closed}} = B_{1,0} \cup B_{2,0}$ and corresponding normals are extracted as in the single material case. These sets of extracted points and normals form the input sets $\mathcal{P}_1$ and $\mathcal{N}_1$. The MPU function created from these sets $f_1(\boldsymbol{x})$ take positive values inside the heterogeneous object, negative values outside and $f_1(\boldsymbol{x}) = 0$ corresponds to S_1. Points located on the inner boundary $S_{2,\text{open}} = B_{1,2}$ are extracted and added to a second point set $\mathcal{P}_2$. A second MPU function $f_2(\boldsymbol{x})$ is created to approximate the points in $\mathcal{P}_2$. Near the extracted points the MPU surface approximates the inner boundary $S_{2,\text{open}}$ accurately. Away from the extracted points the function extends the surface, in a direction perpendicular to the normals of the boundary points located at the limit of the point set (Figure 3.6 (g)). Thanks to this important property of MPU functions, the junction between the bounding surfaces is defined as the intersection between the two functions $f_1(\boldsymbol{x}) = 0$ and $f_2(\boldsymbol{x}) = 0$. Hence, the junction-lines between three different regions are well defined whatever the relative angle and position between the boundary surfaces.

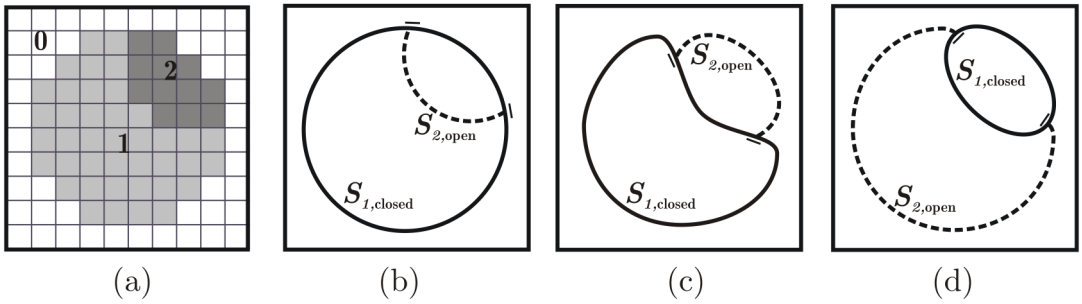


FIGURE 3.7: ***Definition of a junction between three materials using implicit functions.*** *A junction between three materials* 0*,* 1 *and* 2 *can be defined in three ways, according to the region that is considered as dominant. The junction is always defined as the intersection between an open $S_{2,open}$ and a closed $S_{1,closed}$ surface. From an image containing several segmented regions (a), our algorithm ensures that the closed surface is extracted as a smooth triangular mesh. (b) Region* 0 *is dominant, $S_{1,closed} = B_{1,0} \cup B_{2,0}$ and $S_{2,open} = B_{2,1}$. (c) Material* 1 *is dominant, $S_{1,closed} = B_{1,0} \cup B_{2,1}$ and $S_{2,open} = B_{2,0}$. (d) Material* 2 *is dominant, $S_{1,closed} = B_{2,0} \cup B_{2,1}$ and $S_{2,open} = B_{1,0}$.*

An *open* surface is always defined respectively to a *closed* surface, and may either be located *inside* or *outside* this surface. During the subsequent polygonisation process, a triangulation is built on the zero-level of the distance functions. When the considered function is classified as *open*, the sign of its corresponding *closed* surface is evaluated to check the domain of validity of the function. In Figure 3.6 a change of sign in $f_2(\boldsymbol{x})$ yields to a triangulation only when $f_1(\boldsymbol{x}) > 0$.

Figure 3.7 shows that the same segmented image may be defined in several ways according to the label that is considered as dominant. Hence, the user of the mesher must define the function $S_{1,\text{closed}}$ and $S_{2,\text{open}}$ according to the desired result. Indeed, in our algorithm an overall smoothness is always assigned to the closed surface $S_{1,\text{closed}}$, bounding the region that is considered as dominant, and an open surface $S_{2,\text{open}}$ always joins a closed surface $S_{1,\text{closed}}$ with a sharp edge.

3.6 Geometric approximation accuracy evaluation

3.6.1 Taubin distance

Parameter $\varepsilon_{\max}$ gives the acceptable distance between the reconstructed surface and the input boundary points. In the algorithm, a subdivision cell is divided until the local shape function $Q_i(\boldsymbol{x})$ approximates the local boundary points within the defined tolerance: $\varepsilon <$

$\varepsilon_{\max}$. Therefore, the distance from a point to the reconstructed surface must be computed throughout the algorithm. Determining the Euclidean distance from a point $\boldsymbol{p} \in \Re^3$ to the surface defined by $f(\boldsymbol{x}) = 0$, is equivalent to finding the minimum distance of point $\boldsymbol{p}$ to the points located on the zero-level set of f:

$$\delta_{\text{Euclidean}}(\boldsymbol{p}, f(\boldsymbol{x}) = 0) = \min\left(\left\|\boldsymbol{p} - \boldsymbol{q}\right\|, \boldsymbol{q} : f(\boldsymbol{q}) = 0\right) \quad \boldsymbol{q}, \boldsymbol{p} \in \Re^3 \tag{3.44}$$

Let $\boldsymbol{p}$ be a point that is not located on the surface, i.e. $\boldsymbol{p} : f(\boldsymbol{p}) \neq 0$ and compute the first order Taylor expansion of $f(\boldsymbol{q})$,

$$f(\boldsymbol{q}) = 0 = f(\boldsymbol{p}) + \nabla f(\boldsymbol{p})(\boldsymbol{q} - \boldsymbol{p}) + O(\left\|\boldsymbol{p} - \boldsymbol{q}\right\|) \tag{3.45}$$

Dropping the higher order terms gives,

$$f(\boldsymbol{q}) = 0 \approx f(\boldsymbol{p}) + \nabla f(\boldsymbol{p})(\boldsymbol{q} - \boldsymbol{p}) \tag{3.46}$$

which is equivalent to:

$$\left|f(\boldsymbol{q})\right| = 0 \approx \left|f(\boldsymbol{p}) + \nabla f(\boldsymbol{p})(\boldsymbol{q} - \boldsymbol{p})\right| \tag{3.47}$$

Applying the triangular inequality,

$$\left|f(\boldsymbol{q})\right| = 0 \geq \left|f(\boldsymbol{p})\right| + \left|\nabla f(\boldsymbol{p})(\boldsymbol{q} - \boldsymbol{p})\right| \tag{3.48}$$

and then Cauchy–Schwarz inequality, we obtain

$$\left|f(\boldsymbol{q})\right| = 0 \geq \left|f(\boldsymbol{p})\right| + \left\|\nabla f(\boldsymbol{p})\right\| \left\|(\boldsymbol{q} - \boldsymbol{p})\right\| \tag{3.49}$$

In the end, the first order approximation of the Euclidean distance from a point $\boldsymbol{q} : f(\boldsymbol{q}) = 0$ to a point $\boldsymbol{p} : f(\boldsymbol{p}) \neq 0$ is given by

$$\left\|(\boldsymbol{q} - \boldsymbol{p})\right\| \approx \frac{\left|f(\boldsymbol{q})\right| - \left|f(\boldsymbol{p})\right|}{\left\|\nabla f(\boldsymbol{p})\right\|} = -\frac{\left|f(\boldsymbol{p})\right|}{\left\|\nabla f(\boldsymbol{p})\right\|} \tag{3.50}$$

which is called the Taubin's distance [163].

$$\delta_{\text{Taubin}} = \frac{\left|f(\boldsymbol{q})\right| - \left|f(\boldsymbol{p})\right|}{\left\|\nabla f(\boldsymbol{p})\right\|} = -\frac{\left|f(\boldsymbol{p})\right|}{\left\|\nabla f(\boldsymbol{p})\right\|} \tag{3.51}$$

Therefore, in the MPU algorithm, the geometric accuracy ε of the local quadratic function $Q_i(\boldsymbol{x})$ approximating the set of points $\mathcal{P}_i$ in subdivision cell i is evaluated by computing the Taubin's distance of the subdivision cell:

$$\varepsilon = \max_{\boldsymbol{p}_j \in \mathcal{P}_i} \frac{\left\|Q_i(\boldsymbol{p}_j)\right\|}{\left\|\nabla Q_i(\boldsymbol{p}_j)\right\|} \tag{3.52}$$

3.6.2 Hausdorff distance

The Taubin distance presented above is used within the MPU algorithm to guide the octree-based subdivision of cells. It is a measure of the distance between a point and a surface. However, to present our results at the end of this chapter, we need to introduce the Hausdorff distance [79]. The Hausdorff distance is extensively used in literature to compute distance between two sets of points.

Let $\boldsymbol{p}$ be a point of the three-dimensional space $\Re^3$ and $\mathcal{S}$ be a two-dimensional surface embedded in $\Re^3$. The distance δ from $\boldsymbol{p}$ to $\mathcal{S}$ has already been defined in (3.44) as:

$$\delta(\boldsymbol{p},\mathcal{S}) = \min\left(\left\|\boldsymbol{p}-\boldsymbol{q}\right\|, \boldsymbol{q}\in\mathcal{S}\right) \tag{3.53}$$

Let now $\mathcal{S}_1$ and $\mathcal{S}_2$ be two two-dimensional surfaces embedded in $\Re^3$, and $\boldsymbol{p}_1$ a point of $\Re^3$ belonging to $\mathcal{S}_1$, the distance between $\mathcal{S}_1$ and $\mathcal{S}_2$ can be defined by extending the idea of (3.53),

$$\Delta(\mathcal{S}_1,\mathcal{S}_2) = \min\left\{\delta(\boldsymbol{p}_1,\mathcal{S}_2), \boldsymbol{p}_1\in\mathcal{S}_1\right\} \tag{3.54}$$

The distance (3.54) is set to be relative because it is not symmetrical $\Delta(\mathcal{S}_1,\mathcal{S}_2) \neq \Delta(\mathcal{S}_2,\mathcal{S}_1)$. In response to this remark, the Hausdorff distance d between two surfaces $\mathcal{S}_1$ and $\mathcal{S}_2$ is defined as the maximum of these two relative distances:

$$d(\mathcal{S}_1,\mathcal{S}_2) = \max\left\{\Delta(\mathcal{S}_1,\mathcal{S}_2), \Delta(\mathcal{S}_1,\mathcal{S}_2)\right\} \tag{3.55}$$

3.7 Applications and results

The multi-level Partition of Unity surface reconstruction method presented in Section 3.3, its extensions to binary three-dimensional images presented in Section 3.4 and to multi-domain three-dimensional images presented in Section 3.5 was successfully implemented and integrated in the finite element software Metafor. It is now actively used to re-construct geometries from medical images in view of finite element modelling. Results obtained from three types of input datasets are presented here: (1) a set of points in $\Re^3$, (2) a binary segmented dataset and (3) a multi-label segmented dataset.

3.7.1 Surface reconstruction from a set points

The set of points sampled from a *lion-dog* statue, already illustrated in Figures 3.3 and 3.4 is used here to illustrate the influence of the newly proposed parameters on the reconstructed geometries, as compared to the original MPU surface reconstruction method.

Figure 3.8 illustrates the results obtained using five different sets of parameters in our reconstruction algorithm. The first model on the left was obtained by using the method proposed by Hoppe et al. [84], which is: the distance to a set of points is the distance to the nearest surface point, computed along the normal associated to this point. Hoppe et al. [84] use the normal to the local tangent plane to compute this distance. However, in the present case, we use the normal associated with the input point, because this normal is available in the input dataset. The remaining four models, on the right of Figure 3.8, were computed using a multi-level Partition of Unity approach. Parameter N_{min}, the minimum number of points in a subdivision cell, is set to $N_{\text{min}} = 60$, which is to be compared with the size of the point cloud of 99977 points. Parameter α, giving the overlapping ratio between subdivision cells, is defined as $\alpha = 0.75$, which is the value proposed in the original algorithm and the default value in our implementation. The geometric error-tolerance parameter ε_{max} was set to one hundredth of the model cross sectional size, taking account of the dataset dimensions reported in Table 3.1, $\varepsilon_{\text{max}} = 0.01\Delta\boldsymbol{x} = 0.018$mm.

The upper models of Figure 3.8 illustrate the smoothness of the reconstructed surfaces. The implicit distance definition of Hoppe et al. [84] certainly reconstructs the most geometric details. Most details are also reconstructed with quadratic interpolating and approximating functions. Linear functions, with identical N_{min}, α and ε_{max} parameters, output a smoother result.

The lower models illustrate the obtained geometric approximation error, measured by constructing fine meshes on the reconstructed surfaces and, for each node, computing its distance with the set of input points. The distance indicated under the models are the computed Hausdorff distances, defined in Section 3.6.2, between the reconstructed surface and the input set of points (Section 3.6.2). First, looking at the distance fields, quadratic functions and Hoppe et al. [84] give more distributed results with more points located near the extrema of the color scale. This is a direct result of the smoothness of the reconstructed models. Now looking at the computed Hausdorff distances, the best overall geometric approximation is given by the linear interpolating function. In both linear and quadratic, the use of interpolating weight functions gives a lower Hausdorff distance between input points and reconstructed surface which is in accordance with our predictions of Section 3.4.3. The model of Hoppe et al. [84] extracts the fine details of the geometry but also generates many outliers, resulting in a high Hausdorff distance.

Table 3.2 summarises the computed Hausdorff distances between input point set and reconstructed surface for the five models presented in Figure 3.8 and an additional two models, computed with quadratic interpolating and approximating functions with a higher value for parameter N_{min}, meaning that the subdivision procedure will stop earlier. Results

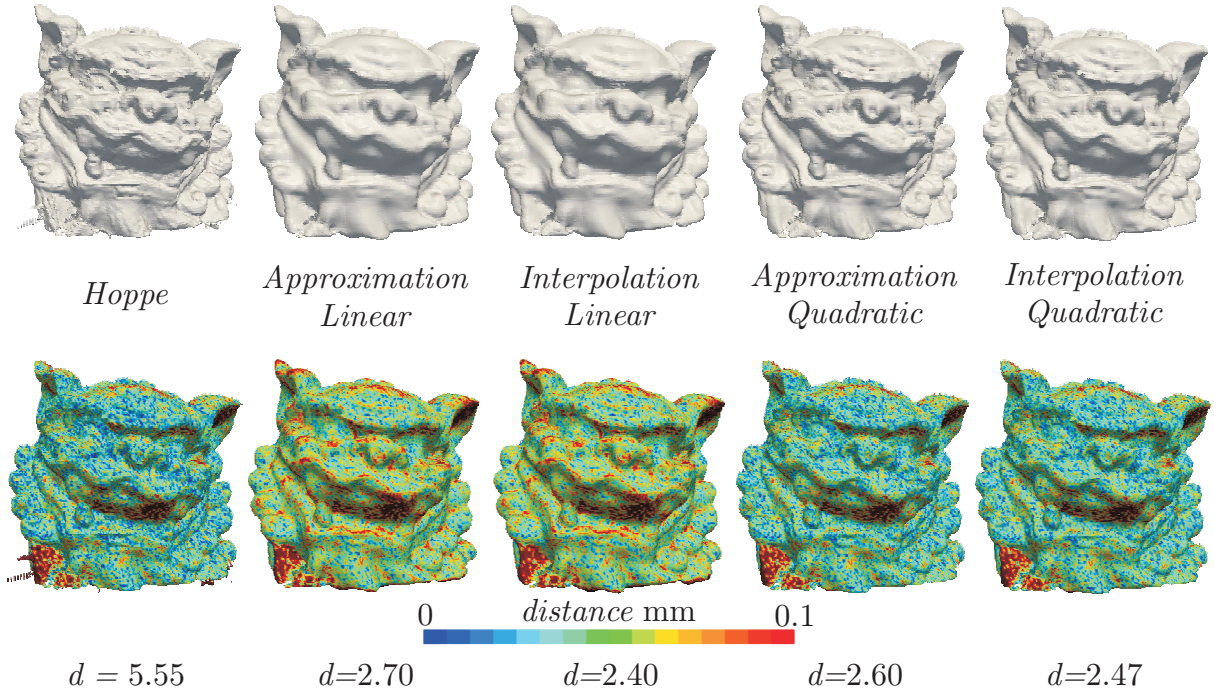


FIGURE 3.8: ***Evaluation of several reconstruction approaches on the set of points sampled from a* lion-dog *statue.*** *The first model on the left was reconstructed using our implementation of the method proposed by [84]. The remaining four models were computed using a multi-level Partition of Unity approach with parameters $N_{min} = 60$ and $\alpha = 60$ and $\varepsilon_{max} = 0.18$ mm. The upper models illustrate the smoothness of the reconstructed surfaces. The lower models illustrate the obtained geometric approximation error, measured by constructing fine meshes on the reconstructed surfaces and, for each node, computing its distance with the set of input points. The distance indicated under the models are the computed Hausdorff distances.*

indicate that with higher numbers of N_{min} (100 instead of 60) the geometric fidelity is not improved by using interpolating rather than approximating weight functions.

3.7.2 Surface reconstruction from a binary image

The first dataset that is considered to evaluate our extension of the MPU approach to segmented datasets is a three-dimensional binary image obtained by sampling a sphere of diameter 40 mm with a spacing of 2 mm. Extracting the boundary points from this input image, using the procedure of Section 3.4.1, resulted into a set of 1904 input points with associated normals. This set was then used as input of our MPU surface reconstruction algorithm with $\varepsilon_{\text{max}} = 1$ voxel, $\alpha = 0.75$ and various values of N_{min}. Again linear and quadratic, interpolating and approximating functions are considered.

TABLE 3.2: ***Evaluation of several reconstruction approaches on the set of points sampled from a* lion-dog *statue.*** *The parameters used for MPU reconstructions are* $\alpha = 60$ *and* $\varepsilon_{max} = 0.18$ *mm.*

Surface reconstruction method and parameters	**Hausdorff distance**
Hoppe et al. [84]	5.55
Linear function, Approximating weights, $N_{\min} = 60$	2.70
Linear function, Interpolating weights, $N_{\min} = 60$	2.40
Quadratic function, Approximating weights, $N_{\min} = 60$	2.60
Quadratic function, Interpolating weights, $N_{\min} = 60$	2.47
Quadratic function, Approximating weights, $N_{\min} = 100$	2.61
Quadratic function, Interpolating weights, $N_{\min} = 100$	2.74

Figure 3.9 presents the obtained implicit surfaces $f(\boldsymbol{x}) = 0$. Quadratic functions with a low value of $N_{\min}$ outputs an irregular surface, showing the discretisation of the initial image. In a view of removing the stair-stepped artefacts of patient-specific meshes using a MPU-based surface reconstruction method, this is exactly the result we would like to avoid. Smoother results are obtained for higher values of $N_{\min}$ in all cases. In both interpolating and approximating approaches, the use of linear functions seems to output a more robust result with respect to the $N_{\min}$ parameter. As it is not clear how to define $N_{\min}$, this robustness surely is an advantage.

The graph drawn in Figure 3.10 shows the effect of parameter $N_{\min}$ on the computed Hausdorff distance. The best match between the input points and the reconstructed model is obtained for quadratic functions with a finely tuned $N_{\min}$ parameter. Low values of $N_{\min}$ produce jagged surfaces with outliers, which should absolutely be avoided when these implicit functions are used for further mesh generation, as this will produce topologically incorrect meshes. High values of $N_{\min}$ give less accurate results. Another option is to use linear (approximation or interpolating) functions, with a low value of $N_{\min}$. Even though the result may be slightly less close to the input points, the approximation is still acceptable as the maximum distances between the model and the segmented boundaries is less than the image spacing. With linear functions, a smooth result will be generated in all cases with no risk of irregular jagged reconstruction.

The second dataset that is investigated here consists of a segmented scan of an aluminium foam, which was imaged at ETH Zürich [129]. Figure 3.11, upper, illustrates the

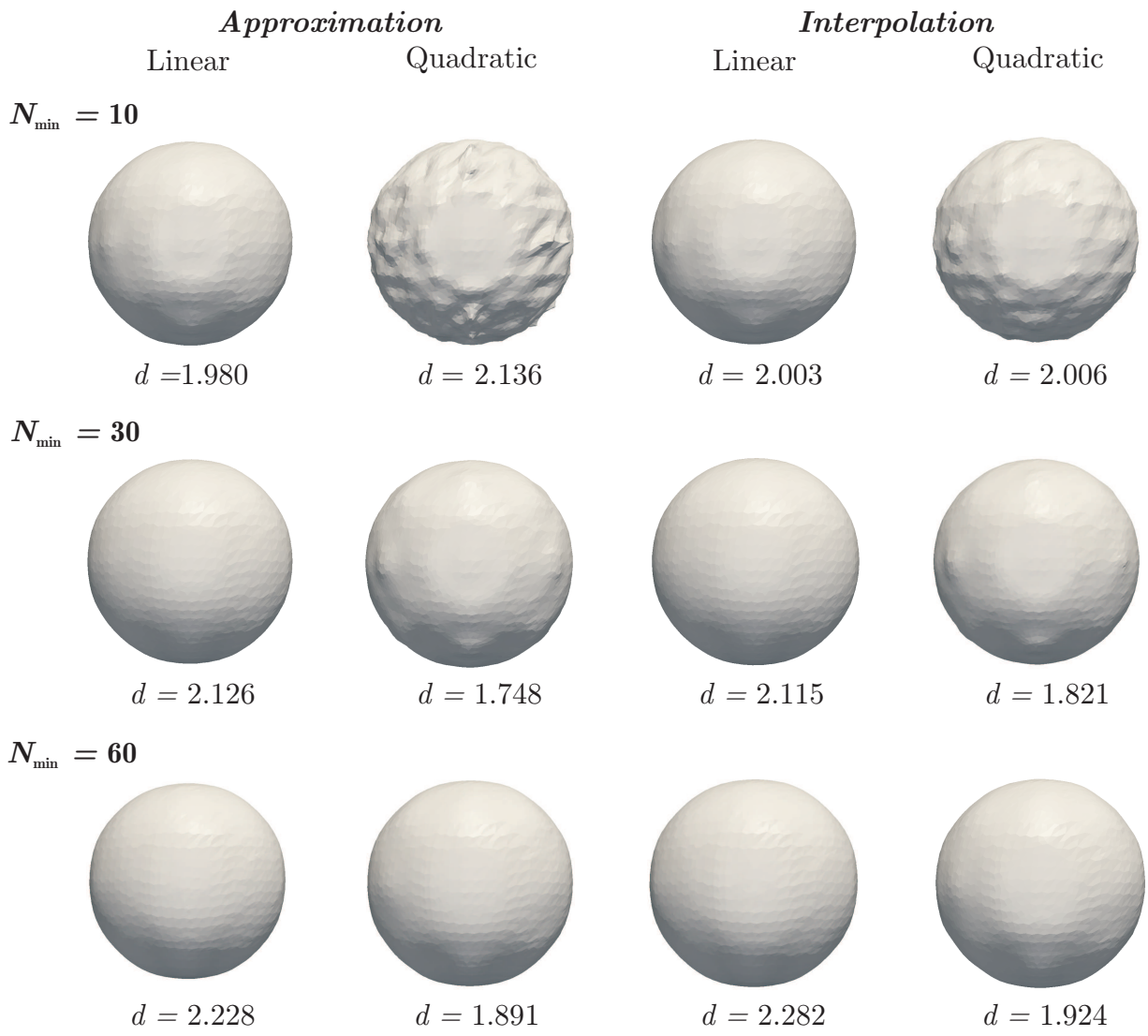


FIGURE 3.9: ***Surface reconstruction from a binary image****. Sphere. Illustration of the reconstructed models for several input parameters. d denotes the Hausdorff distance.*

TABLE 3.3: *Characteristics of the aluminium foam and the mandible datasets.*

	dimensions [voxels]	**spacing** [mm]	**points**
aluminium foam	(146,146,157)	(0.060,0.060,0.062)	171201
mandible	(306, 283, 166)	(0.36,0.36,0.5)	998532

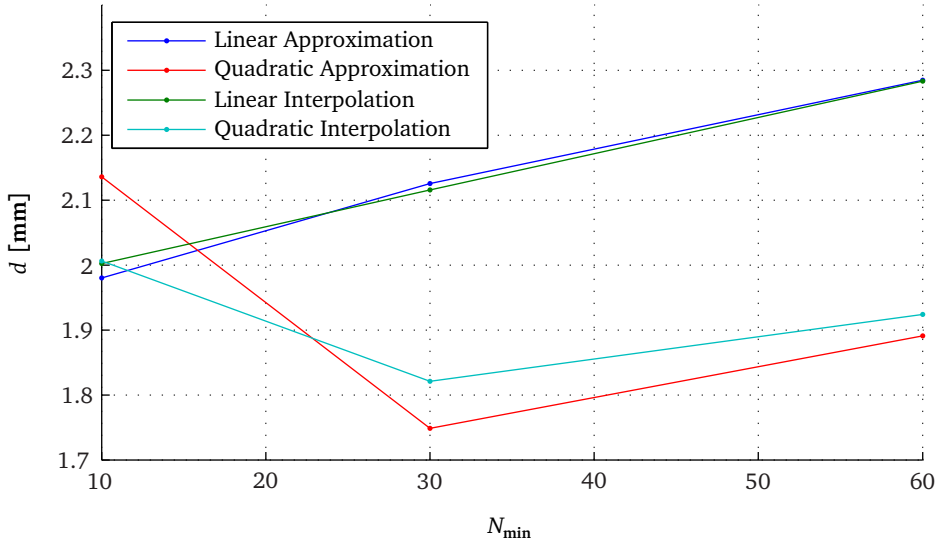


FIGURE 3.10: ***Surface reconstruction from a binary image****. Sphere. Obtained Hausdorff distances for several values of* N_{min}.

complexity of the input image and gives an idea of the sampling resolution. The characteristics of the input dataset are reported in Table 3.3.

Figure 3.11 shows that higher values of $N_{\min}$ may help to extract all the trabeculae of a truss-like structure, which were lost by discretisation during μCT-scanning. In these cases, quadratic functions will therefore be recommended. Hausdorff distances are indicated below the figures. Again, the best match between reconstructed model and input points is obtained for a quadratic function, with a finely tuned $N_{\min}$ parameter.

3.7.3 Surface reconstruction from a multi-label image

Figure 3.12 illustrates how our strategy to define multi-material volumes with a set of implicit functions, explained in Section 3.5, is applied for the reconstruction of the lower mandible. The characteristics of the input dataset are also summarised in Table 3.3.

The patient's mandible comprises 13 teeth, each of which were segmented using a different label, resulting in a multi-label image (Figure 3.12, Upper Left). 14 sets of points and associated sets of normals were extracted for each of the resulting 14 regions. These are illustrated in Figure 3.6, Upper Right, where the normals have been coloured according to their orientation. Taking the vocabulary of Section 3.5, the mandible is considered as an *open* surface, intersected by the teeth boundaries (Figure 3.12, Lower Left). This means that, when reconstructing the total multi-region system, the surface of the mandible will form the outer boundary of the structure but stops at its intersection with the teeth. Each

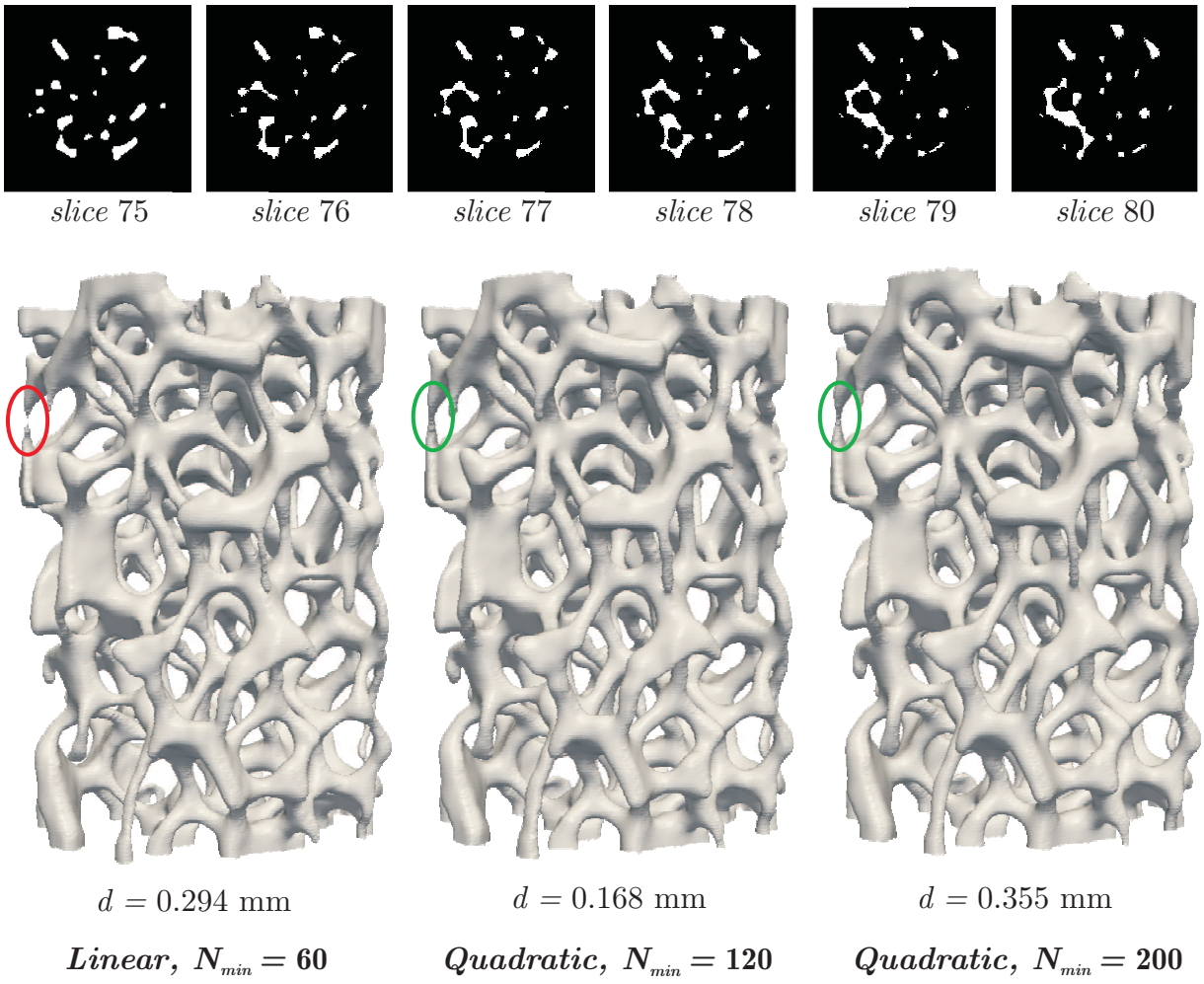
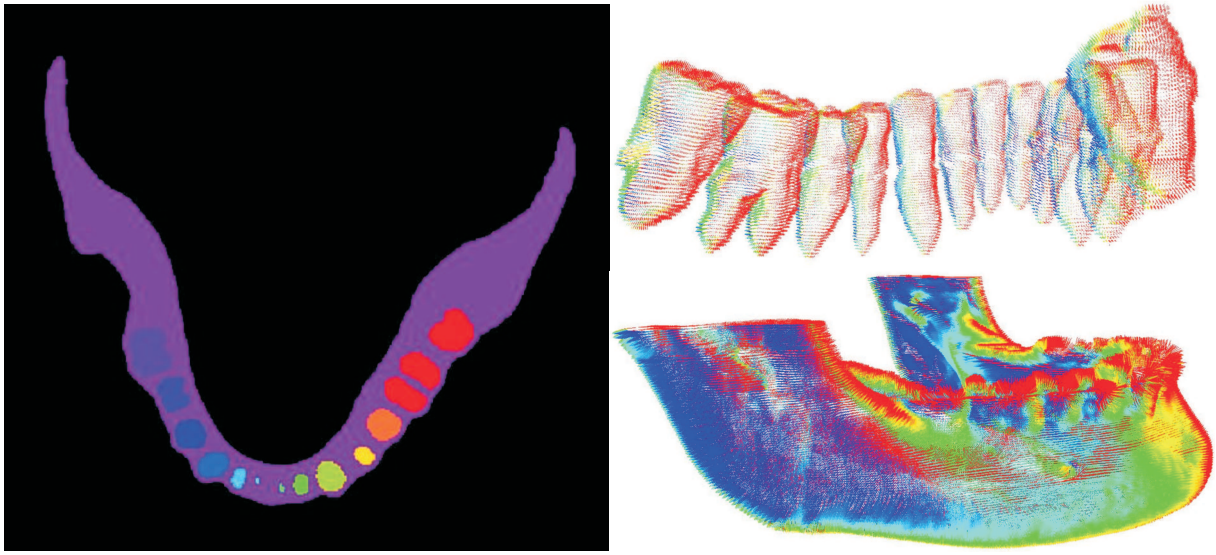


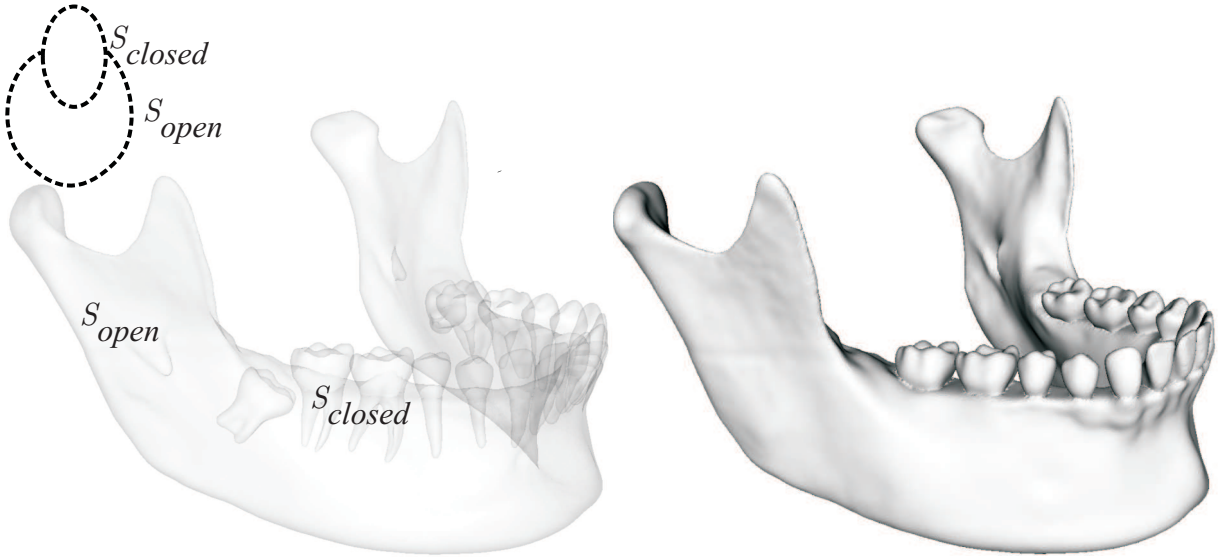
FIGURE **3.11:** ***Surface reconstruction from a binary image.*** *Aluminium foam. The red circle indicates a loss of connection of the trabeculae whereas the green circles indicate that these connections are preserved.*

tooth is bounded by one closed surface. The definition of this heterogeneous object will enable us to generate a consistent triangular mesh of the inner and outer boundaries in the next section, therefore, the generated volume mesh will be suitable for finite element analysis. The information of the material regions labelled in the segmented dataset is transferred throughout the process, so that in the end, the mandible and the teeth can be distinguished in the finite element analysis, and different material properties assigned, even though they form one unique object.



multi-label 3D image

several sets of points: mandible and teeth



Mandible: Open surface, intersected by the Teeth: Closed surfaces.

multi-material structure, with several inner regions

FIGURE 3.12: ***Surface reconstruction from a multi-label image.*** *Upper Left: 2D slice from a multi-label segmented 3D CT-scan of the lower jaw. Upper Right: Sets of points, and associated normals, extracted from the multi-label image. One set of points corresponds to one coloured region in the multi-label image. Lower Left: Multi-material object description using the concept of open and closed surfaces introduced in Section 3.3. Lower Right: Multi-material implicitly defined mandible.*

3.8 Conclusions

In this section, a procedure to extract the geometries from scanned objects has been presented. The main advantage of this surface reconstruction algorithm is that it enables smooth interpolation of the scanned datasets, thus avoiding the jagged edges of the majority of mesh generation methods. Our algorithm, is based on the multi-level Partition of Unity approach for surface reconstruction from a cloud of points [136, 138].

Our major contributions to the approach is its extension to reconstruct geometries from segmented datasets, in the view of patient-specific meshing. First a strategy to define a set of points and associated normals from segmented uni-label and multi-label images has been presented (Sections 3.4.1 and 3.5 respectively). Second, interpolation weight functions rather than approximating ones have been proposed (Section 3.4.2). Third, the use of linear instead of quadratic local functions has been investigated (Section 3.4.3). And, last but not least, an efficient strategy to represent multi-material structures with a set of distance functions has been defined (Section 3.5). The latter enables generalization of the procedure to the creation of biological structures having several inner boundary surfaces, defining several material regions within the structure.

The algorithm has been implemented in the finite element software Metafor and is successfully being used to generate patient-specific finite element meshes as attested by several peer reviewed journal and conference papers [49, 56–58, 60, 116, 135] and illustrated in the next chapters of this dissertation.

Results indicate the wide range of application of our algorithm. It has been used to generate three-dimensional implicit models from a set of points (Section 3.7.1), from binary segmented images (Section 3.7.2) and from multi-label images (Section 3.7.3). The use of interpolating functions rather than approximating functions, within a multi-level Partition of Unity surface reconstruction approach, has shown no tremendous improvement of the generated models in regard to their geometric accuracy, as some results do show lower Hausdorff distances (Table 3.2) and other do not (Figure 3.10). However, the use of linear instead of quadratic functions adds rapidity and robustness. It enables fast surface reconstruction from, possibly noisy, sets of points. The decrease in reconstruction time is mainly due to the fact that no system must be solved to compute the linear function coefficiants as was needed for a quadratic function (Section 3.4.3). The robustness has been observed for all input datasets. Results have shown that a better match between input data and output model may be obtained with quadratic functions, if the parameter $N_{\min}$, defining the minimum set of points per subdivision cell, is finely tuned. However, low values of this parameter create surfaces with irregularities and low geometric accuracy. Linear functions have the great advantage of generating valid results for all values of this parameter,

valid meaning that the geometric approximation is still within the defined limits of one voxel width and with no spurious parts so that direct application of a surface triangulation algorithm will generate topologically correct meshes, suitable for further volume mesh generation and finite element analysis.

Chapter 4

Multi-domain tetrahedral mesh generation and adaptation

The approach described in the previous chapter provides one distance function $f(\boldsymbol{x})$ for a single-material tissue and a set of distance functions $f_i(\boldsymbol{x})$ for multi-domain structures. These functions approximate the point sets extracted from the tissue boundaries in the segmented image. In this chapter, a strategy to generate a surface mesh of the tissue boundaries is proposed. The main particularity of the approach is that it is capable of generating *valid* meshes even in the case of multiple interconnected tissues. The term *valid* meaning, valid in the sense of the finite element method, that is to say, with no gaps nor overlays at the material interfaces, and, with node-to-node and edge-to-edge connections only. The generated surface mesh is a triangular mesh and is used as basis for further tetrahedral volume mesh generation.

4.1 Literature review

4.1.1 Mesh generation strategies

Mesh generation approaches may be classified into three categories: spatial decomposition methods, advancing front methods and Delaunay refinement methods.

4.1.1.1 Spatial decomposition methods

Spatial decomposition methods generate meshes based on a subdivision of space based on grids, quadtrees in 2D or octrees in 3D. These methods originated in the eighties [10, 188] and many variants have been proposed since then. The simplest approach that may be classified into this category is voxel-based meshing: a grid comprising the object is defined and each grid cell, or image voxel, is turned into a hexahedron if located inside the object. Obviously no good surface representation will be generated with this method. A better boundary approximation is obtained by further subdividing each cube into tetrahedra, as illustrated by the Red Green tetrahedral meshing technique of Molino et al. [120], or by successively deforming the boundary elements [190].

The most popular approaches to extract a surface mesh of the object's boundaries are based on the marching cubes, proposed by Lorensen and Cline [110], [6, 8, 80, 97, 187]. In the marching cubes, a bounding box enclosing the data is defined and sampled into a regular 3D Cartesian grid, thus forming a series of cubic cells. Each possible intersection case of this cube with the object is tabularised and leads to creating of one or two triangles. A popular extension of the marching cubes is the marching tetrahedra [134]. In the latter, each grid cells are further subdivided into 5 or 6 tetrahedra, and the triangulation problem is therefore reduced to the triangulation of a tetrahedron.

4.1.1.2 Advancing front methods

Advancing front methods or moving front approaches. These methods extend a mesh of the object's boundaries, called initial front, to the third dimension by incrementally filling the object's volume with tetrahedra that are made up of a triangle from the front connected to an existing vertex of the mesh or to an inserted vertex in the inner area bounded by the front. The quality of the resulting volume mesh highly depends on the quality of the initial boundary triangular mesh. Advancing front mesh generation methods were first developed by Lo [109], and have been extensively studied since then [39, 186].

4.1.1.3 Delaunay refinement mesh generation

. In Delaunay refinement mesh generation methods Delaunay triangulation is used to generate an initial triangular surface mesh. This produces a *coarse* mesh that is then refined by iteratively adding mesh nodes. A popular implementation of this approach is available through the software Gmsh, proposed by Geuzaine and Remacle [73].

4.1.2 Multi-material mesh generation

The traditional meshing algorithms are not suited for the creation of valid or consistent multi-tissue meshes. In general, when a heterogeneous object needs to be reconstructed, the user applies the meshing algorithms iteratively, for each material domain. Therefore, if the researcher needs to model the lower jaw and attribute distinct properties to the teeth and the mandible (see the dataset presented in the previous Chapter, Figure 3.12), he will will first generate a mesh of the mandible without the teeth, and then re-apply the meshing algorithm several times to generate meshes for the teeth. A significant effort will then be required to merge the nodes and elements on the interface between the teeth and the mandible. Inevitable, there will be voids and mismatched nodes at the interface between these two materials. The result will be visually appealing but totally in-adaptable for finite element simulations.

The first generalisation of the marching cubes algorithm to segmented images containing several segmented regions, also called multi-label images, has been proposed by Hege et al. [80]. In the latter, up to three materials may meet at each grid cell. Wu and Sullivan Jr [187] extended this number to up to eight different labels per cell. A multiple-material version of the marching tetrahedra algorithm has been proposed by Müller [124]. Other approaches to multi-tissue mesh generation are based on dual contouring [17, 18, 148], volume subdivision [108, 191] and Delaunay refinement algorithms [23, 118, 144].

Our mesh generation approach is based on the marching tetrahedra for its simplicity and because it does not suffer from the ambiguity problems of the marching cubes. Solutions to these ambiguities, which are inherent to data sampling, have been proposed in [38, 121, 128, 170]. The use of the marching tetrahedra algorithm is also an efficient way to circumvent the problem.

The classical marching tetrahedra algorithm is presented in the next section. In Section 4.3, we generalise the marching tetrahedra algorithm to the triangulation of implicitly-defined multi-material structures.

4.2 Classical marching tetrahedra algorithm

The basic ideas of the marching cubes and the marching tetrahedra are similar. A bounding box enclosing the data is defined and sampled into a regular 3D Cartesian grid. This grid defines a series of cubic cells obtained by taking eight grid vertices at a time. In the marching cubes the global triangulation problem is reduced to the triangulation of these cubic cells [110]. In the marching tetrahedra the grid cells are further subdivided into 5

or 6 tetrahedra. The triangulation problem is therefore reduced to the triangulation of a tetrahedron. The subdivision scheme used in this article splits the initial cubic cell into six identical tetrahedra [134].

A tetrahedron has four vertices, let us name them from A to D. On each vertex the value of the distance function is computed. A vertex with a positive value is located inside the surface and a negative value corresponds to the outside of the surface. Hence, the surface crosses the tetrahedron when the vertices values change sign on the tetrahedron. In that case, the cell is called an *active cell* and a triangulation of the intersecting surface in the tetrahedron is computed. In order to automatically determine this triangulation, a binary index is constructed by processing the tetrahedral vertices in the predetermined order ABCD. Each bit of the index is 0 when the corresponding vertex value is negative and 1 otherwise. This index permits the differentiation of the $2^4 = 16$ cases of intersection. All these cases are tabulated so that, from a given index, the triangles to be created are rapidly obtained. The 16 possible cases are reduced to 3 by symmetry, leading to no triangulation (Figure 4.1 (a)), the creation of one triangle (Figure 4.1 (b)) or the creation of two triangles (Figure 4.1 (c)). In this single-material marching tetrahedra algorithm mesh nodes are created on tetrahedron edges only. Linear interpolation of the tetrahedron vertices values is used to position the newly created nodes. As each tetrahedron face is shared by another cell, the obtained triangulated surface is continuous across adjacent cells.

4.3 Multi-material marching tetrahedra algorithm

The traditional marching tetrahedra approach is a binary decision routine: each tetrahedral vertex is either located inside or outside the structure. In the multi-material case, we use several distance functions to describe the material boundaries composing the structure (Section 3.5). Consequently, a tetrahedron may be crossed by the zero level of several distance functions. Therefore, new triangulations patterns must be defined to generalise the marching tetrahedra algorithm. In order to define these new patterns, two rules need to be kept in mind. First, to ensure continuity each triangular face shared by adjacent tetrahedra must have the same splitting pattern. Second, each material domain must be separated from the other material domains by a surface mesh.

Based on the distance functions, a material label is assigned to each tetrahedral vertex. These labels correspond to the sign of the distance function in the classical marching tetrahedra approach. Depending on the number of different material labels on the tetrahedron, four cases are defined:

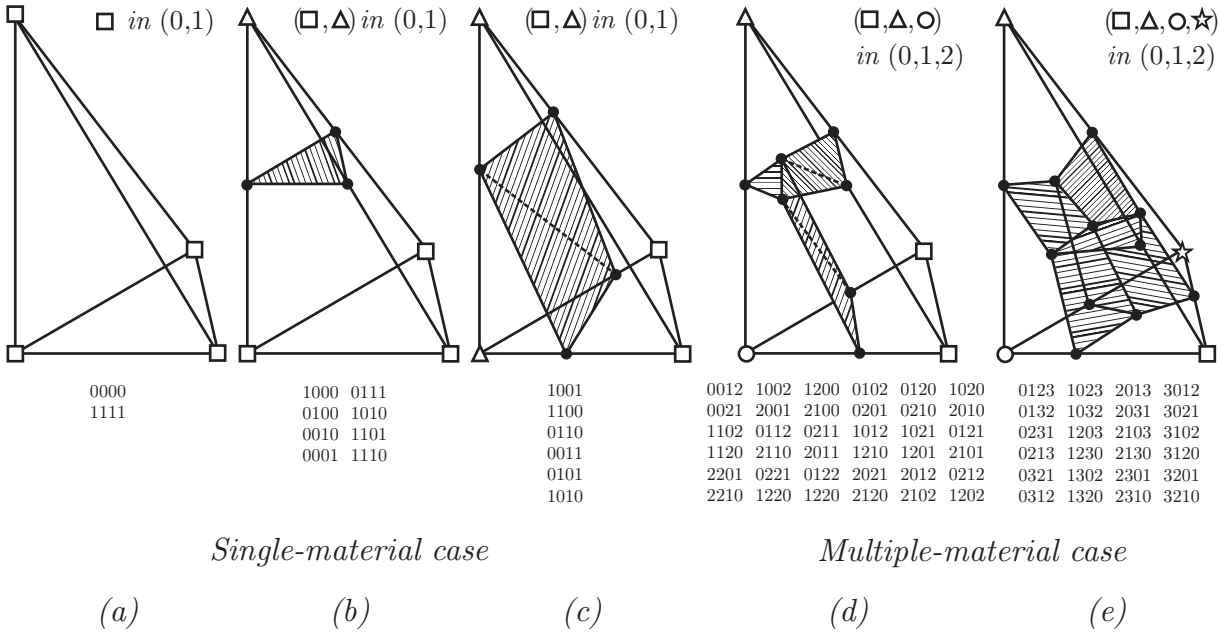


FIGURE 4.1: ***Marching tetrahedra algorithm.*** *(a,b,c) In the single material case, three general triangulation patterns are seen, according to the sign of the distance function evaluated at the tetrahedron vertices: (a) no triangulation, (b) the creation of one triangle and (c) the creation of two triangles. (d,e) In the multi-material case, two triangulation patterns are added to manage the cases in which the tetrahedron falls into three (d) and four (e) distinct material domains. The distinct regions in which the tetrahedron vertices may fall are indicated above each triangulation pattern. The indexes indicated below the triangulation cases are used to define the triangles that must be created for a particular intersection of the surface in the tetrahedron.*

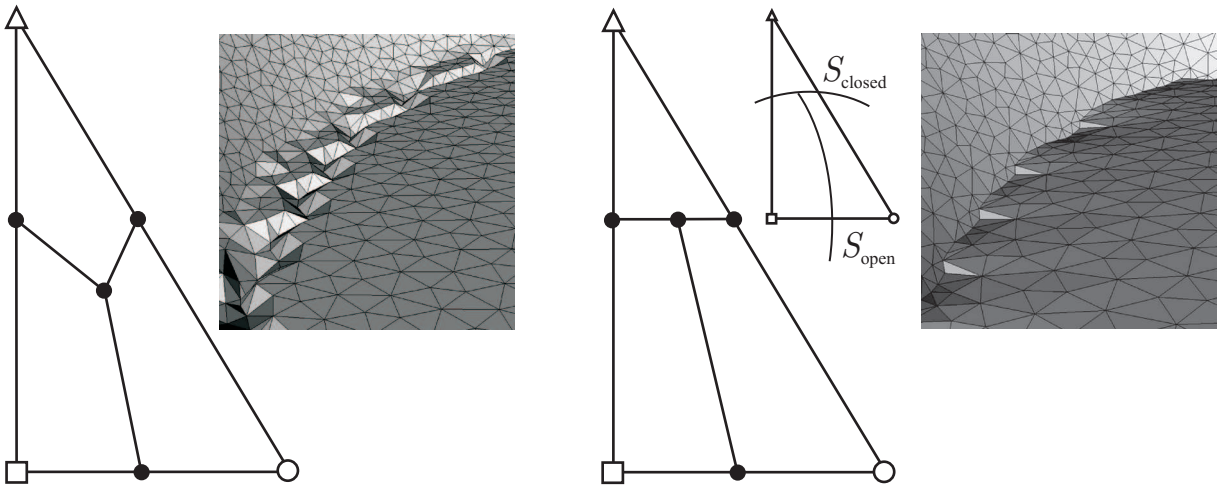
1. All material labels are identical: no surface crosses the tetrahedron and no triangulation is created (Figure 4.1 (a)).

2. Two different material labels: the tetrahedron is crossed by the zero level of a unique distance function and the triangulation defined in the classical marching tetrahedra algorithm is used (Figure 4.1 (b,c)).

3. Three different material labels: the tetrahedron is located at the interface between three materials (Figure 4.1 (d)).

4. Four different material labels: the tetrahedron is located at the interface between four materials (Figure 4.1 (e)).

The two first cases correspond to the classical marching tetrahedra algorithm. The triangulations used in the two last cases are described below.



(a) center of tetrahedron *(b) center of closed surface*

FIGURE 4.2: ***Multi-material marching tetrahedra algorithm.*** *Positioning of a mesh node on an intersected tetrahedral face. (a) Placing the node at the centre of the tetrahedron leads to visually unappealing material junctions. (b) In our algorithm the node is positioned on the closed surface crossing the tetrahedron which leads to a smoother interface.*

Figure 4.1 (d) illustrates the triangulation pattern used in case 3 when the tetrahedron vertices are located in three distinct materials. The definition of the distance functions presented in Section 3.5 leads us to deduce that this case corresponds to the crossing of two functions: a closed surface and an open surface. In order to delimit the three material regions, five of the six tetrahedron edges are inevitably located in two different materials. New nodes are created on these five edges. Their position is determined by linear interpolation of the appropriate distance function evaluated at the edge vertices. Moreover, two tetrahedral faces have vertices located in two different materials domains, whereas the two other faces have three distinct labels on their vertices. In the latter case, a face node must be created in order to respect the requirement stating that separating surfaces must be created between each material domain. As the tetrahedral face is crossed by both distance functions, the question on the location of this new face node must be addressed. The simplest solution would be to place this node at the centre of the three nodes created on the edges of the tetrahedral face. However, as illustrated in Figure 4.2 (a), this strategy produces a small depression in the surface near the interface. A better solution is obtained by noticing that, out of the three edge nodes, two nodes are located on the same closed surface (Figure 4.3). Placing the face node at middle-distance of these two edge nodes gives a smoother interface representation of this closed surface (Figure 4.2 (b)).

The same methodology was used to determine the last triangulation case encountered when the four tetrahedron nodes are located in four distinct materials. Even though this

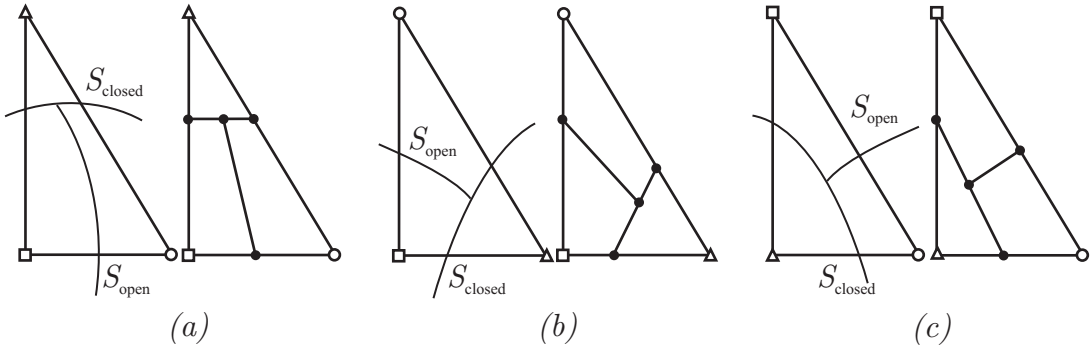


FIGURE 4.3: ***Multi-material marching tetrahedra algorithm****. The new node is created at middle-distance of the two nodes located on the edges of the tetrahedron that are crossed by the closed surface.*

case rarely happens, its definition allows us to obtain a truly general algorithm capable of processing any labelled volume. The triangulation used in this latter case is illustrated in Figure 4.1 (e). As seen in the figure, six edge nodes, four face nodes, and a node located at the centre of the tetrahedron are created to form the five triangles needed to represent the intersection of the surfaces in this tetrahedron.

4.4 Decimation of multi-material surface meshes

Thanks to the implicit surface reconstruction the sampling grid of the marching tetrahedra may be adjusted according to the minimum feature size that should be preserved. The resolution of this sampling grid must be high enough in order to capture all image details. However, increasing the grid resolution rapidly leads to inconvenient mesh sizes. The goal of mesh decimation is to reduce the total number of mesh nodes while retaining the main features.

Our mesh decimation algorithm consists in a vertex decimation scheme. Multiple passes are made over the mesh nodes. During an iteration each node is tested for deletion. If the decimation criterion is met, the node is deleted and the resulting hole in the mesh is re-triangulated. Different criteria and triangulation schemes are used according to the node type. Indeed, in a multi-material surface mesh, nodes are of three types: *surface nodes*, *edge nodes* or *corner nodes* (Figure 4.4). *Surface nodes* are surrounded by a complete cycle of triangles and each edge connected to this node has only two adjacent triangles. *Edge nodes* are located on a junction-line; they have two neighbouring edges that are used by three triangles and two neighbouring edge nodes. *Corner nodes* are located on a junction between four triangular meshes. Practically, referring to Figure 4.4, surface nodes correspond to the nodes created on tetrahedron edges during the triangulation process, edge

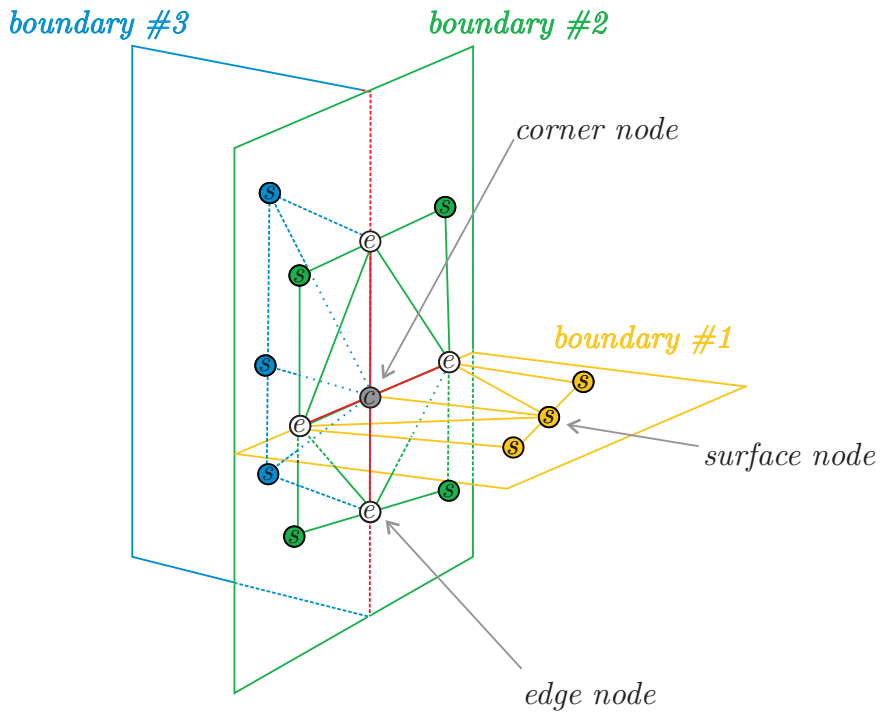


FIGURE 4.4: ***Decimation of multi-material surface meshes.*** *The mesh nodes are classified as surface node, edge node or corner node.*

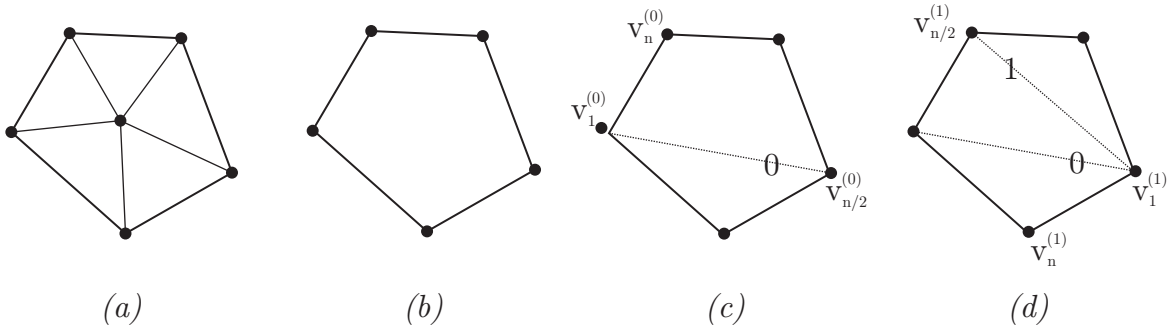


FIGURE 4.5: ***Decimation of multi-material surface meshes.*** *Retriangulation scheme used in our algorithm. (a) A mesh node that satisfies the decimation criterion is deleted. (b) This creates a hole that is re-triangulated using an iterative procedure. (c) The loop is divided in two by connecting two opposite nodes. (d) Each new loop is subdivided until only three nodes remain, thus forming a mesh triangle.*

nodes correspond to nodes created on faces of a tetrahedron, and corner nodes correspond to nodes created in the centre of the tetrahedron.

A specific decimation criterion is used for each node type. A surface node is deleted if its shortest neighbouring edge size is below a threshold and if the local gradient, evaluated as the maximum angle formed by the normals of the neighbouring triangles, is lower than a

threshold. An edge node is deleted if one of its two neighbouring junction edges is shorter than threshold. Corner nodes are not tested for deletion.

Deleting a node and the adjacent triangles creates a hole that needs to be re-triangulated (Figure 4.5). For surface nodes this hole may be seen as loop. This loop is triangulated using a recursive loop-splitting procedure: it is divided into two halves and both the resulting loops are divided again until only three vertices remain [152] (Figure 4.5). For edge nodes an edge that joins the two remaining edge nodes is first created. This new junction edge divides the hole into three loops; each of which are then triangulated like above.

Vertex decimation is included in the triangulation process: a newly created node is tested for removal as soon as its neighbourhood is formed. By combining the marching tetrahedra and the mesh simplification methods we are able to drastically reduce the numbers of nodes and triangles stored in memory.

4.5 Mesh adaptation

The objective of this last step is to improve the quality of the generated surface mesh while preserving its geometric accuracy.

A simple, yet effective, way for improving the quality of the triangles composing the surface mesh is Laplacian smoothing. The simplest Laplacian filters move each vertex located at position $\boldsymbol{x}_0^i$ to a new position $\boldsymbol{x}_0^{i+1}$ that is the average of its adjacent vertices:

$$\boldsymbol{x}_0^{i+1} = \frac{1}{n}\sum_{j=1}^{n} \boldsymbol{x}_j^i \tag{4.1}$$

where $\boldsymbol{x}_j^i$ are the coordinates of the nodes connected to the current vertex $\boldsymbol{x}_0^i$.

Our mesh quality enhancement approach is based on a Laplacian smoothing, but two main improvements are proposed. First, mesh deformation is avoided by constraining the mesh nodes on their original surface. Indeed, Laplacian smoothing has the undesirable effect of rounding over the corners and reduces the volume of convex regions. In the proposed algorithm the distance functions to the material boundaries are known. Hence, the mesh nodes may easily be projected back on their surface at each step of the smoothing procedure. Second, particular care is taken to keep the junction-lines between three materials as smooth as possible. In section 4.3 we defined the junction-line between three materials as the intersection of a closed surface and an open surface. The nodes located at the junction-line, called edge nodes, were positioned on the closed surface so as to avoid surface irregularities. For the same motivations edge nodes are kept on the closed surface

during the smoothing process. Thus, for edge nodes, Equation 4.1 is applied on the subset of the boundary nodes located on the closed surface.

4.6 Applications and results

The proposed procedure for smooth multi-material mesh generation was tested on CT scans of a femur, a lumbar spine, a mandible, a thorax as well as a μCT-scan of an aluminium foam. The geometric approximation error and the quality of the obtained meshes are analysed in this section.

4.6.1 Datasets

The CT-scans of a femur, a lumbar spine, and a mandible were obtained form the OsiriX medical imaging repository[1]. The CT-scan of the thorax was provided by the Department of Weapon Systems and Ballistics, Royal Military Academy, Brussels, Belgium. The μCT-scan of the aluminium foam was provided by ETH-Zürich [129]. The dimensions of the regions of interest and image spacing are indicated in Table 4.1. All five datasets are anisotropic: their in-plane resolution (x,y direction) is different from their slicing resolution (z direction).

The grey-level medical images were segmented semi-automatically using 3D Slicer [142]. The aluminium foam dataset provided was already segmented. For each labelled image the points located on the boundaries of the material regions were extracted and their respective normals were computed (Section 3.4.1). The sizes of the extracted point clouds are indicated in Table 4.1.

The image of the femur was segmented into four regions corresponding to cortical bone, dense trabecular bone at the bone extremities, and low density spongious bone in the shaft of the femur (Figure 4.6 (a)). An implicit definition of this multi-label image was obtained using a combination of four multi-level Partition of Unity functions, two closed functions corresponding to the exterior and interior boundaries of the cortical bone, and two open surfaces delimiting the high and low density regions of trabecular bone (see Section 3.5 for more details on how we proposed to defined multi-material structures with a set of implicit functions).

The dataset called lumbar spine actually consists of three vertebrae of the lumbar spine. Each vertebra was segmented into cortical and spongious bone. The resulting 6 material regions are described implicitly using 6 closed MPU functions, illustrated in Figure 4.7 (b).

[1] $http://pubimage.hcuge.ch:8080/$

TABLE 4.1: ***Characteristics of the input datasets***. *dimensions, spacing, number of material regions and number of points in the extracted point cloud.*

	dimensions	**spacing** [mm]	**regions**	**point cloud**
femur	(179, 196, 546)	(0.74, 0.74, 1.0)	4	847129
lumbar spine	(176,181,261)	(0.59, 0.59, 0.5)	6	677042
mandible	(306, 283, 166)	(0.36,0.36,0.5)	18	998532
thorax	(511, 511, 436)	(0.71,0.71,1.50)	10	
aluminium	(146,146,157)	(0.060,0.060,0.062)	1	171201

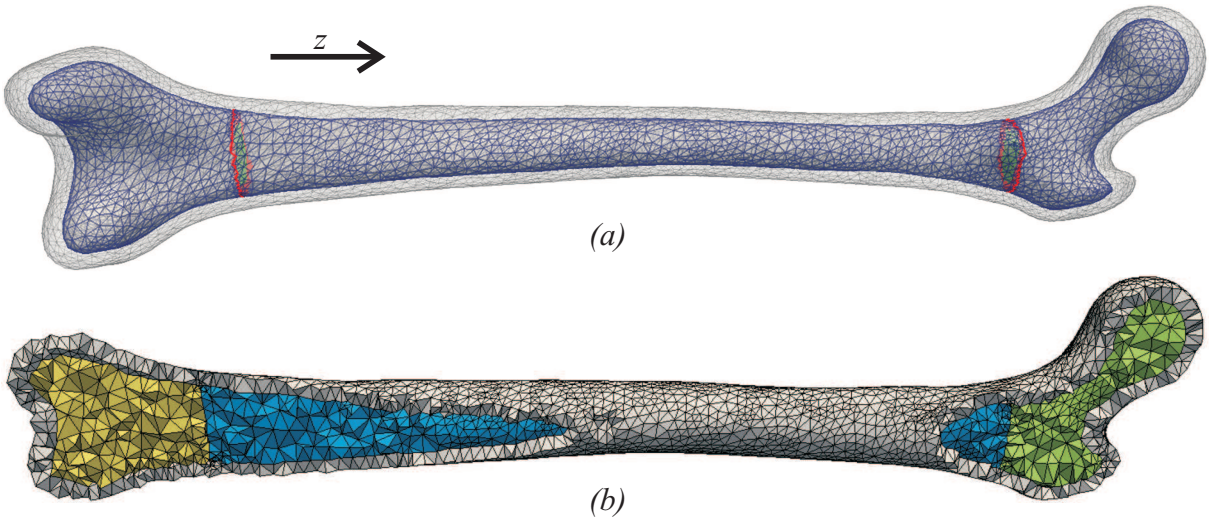


FIGURE 4.6: ***Multi-tissue mesh obtained from CT-scans of a human femur.*** *The femur was segmented into four regions: cortical bone, high-density spongious bone at the bone extremities and low-density spongious bone in the body of the femur. (a) Multi-material surface mesh where the outer boundary is semi-opaque to show the inner surface meshes. (b) Cut through the volume mesh the femur obtained from the above surface mesh.*

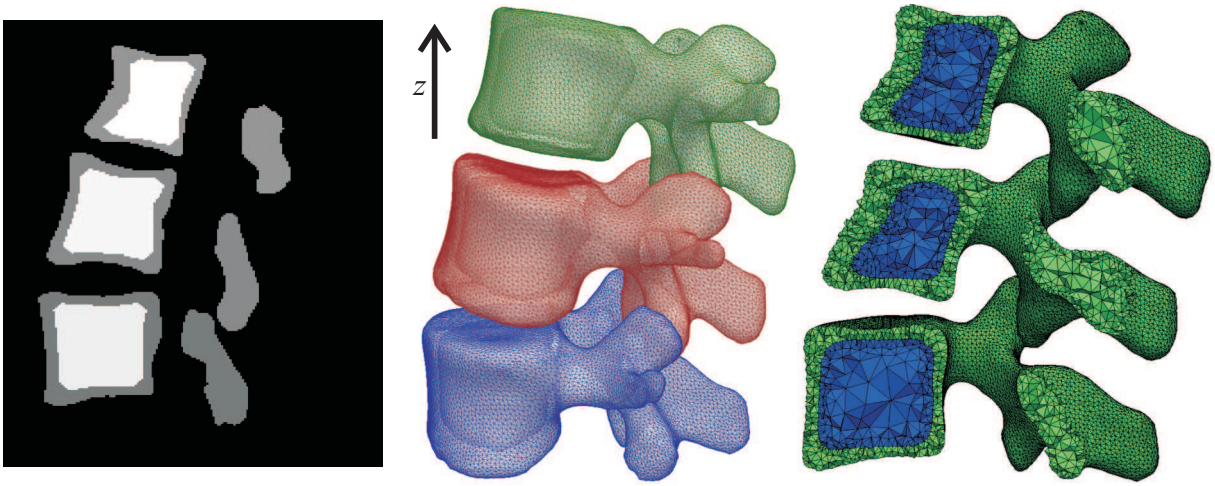


FIGURE 4.7: ***Our meshing procedure performed on a CT-scan of lumbar spine.*** *(a) Each vertebra is segmented into cortical and spongious bone. (b) Multi-material mesh obtained from the segmented data with our algorithm. (c) Cut through the volume mesh the lumbar spine generated from the surface mesh with TetGen [158].*

The lower jaw was segmented into 14 distinct regions using one label for the mandible and 13 labels for the 13 teeth (Figure 3.12). The teeth were extracted as closed surfaces and the mandible as an open surface, as already explained in Section 3.7. This definition yields smooth surfaces of the teeth at the junction with the mandible, i.e. the depression seen in the meshes which are created with other multi-domain meshing algorithms is avoided (Figure 4.8).

The CT-scan of the thorax was segmented using 10 labels: left and right lungs, other internal organs, left and right shoulder blade as well as four connected regions for the thoracic wall: spine and connected ribs, left and right cartilage and sternum. Figure 4.9(b) shows this labelling in an axial slice of the initial dataset. The first six regions (lungs, other organs and shoulder blades) have single closed boundaries, so that our meshing algorithm will generate a single closed triangular surfaces for these parts. The thoracic wall is composed of 4 distinct regions, the sternum, cartilaginous plates on each side of the sternum and the spine with attached ribs. Therefore heart, aorta oesophagus, trachea and stomach were considered as a whole, as well as the vertebrae, the inter-vertebral discs and the ribs. We believe that this simplification should have no impact on the finite element results for the targeted application, which is the study of non-lethal weapons [135]. The result provided by the surface mesher implemented in this thesis work can be seen as the union of several closed triangular surfaces and a multi-material surface mesh. It is displayed in Figure 4.9(c), where the skin is also shown.

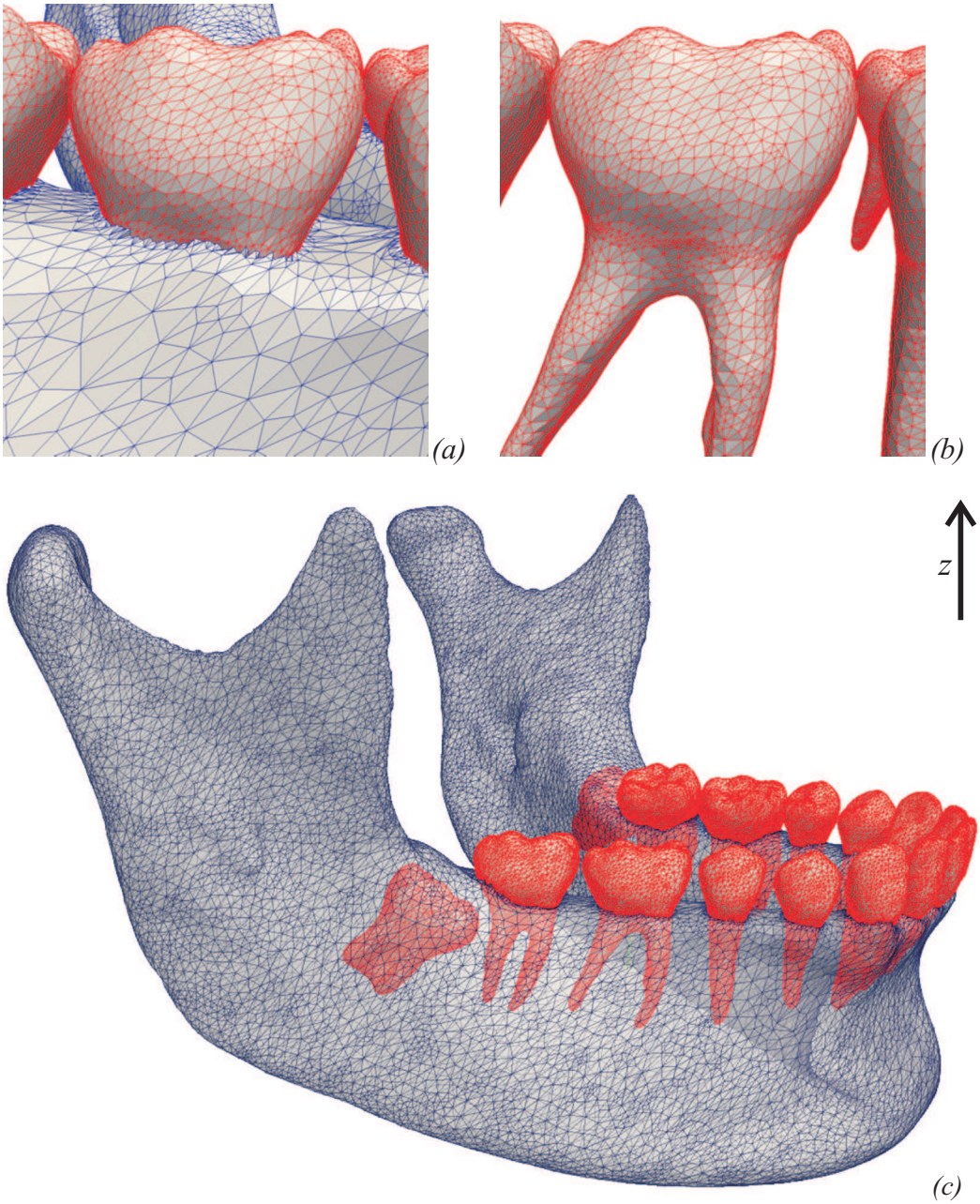
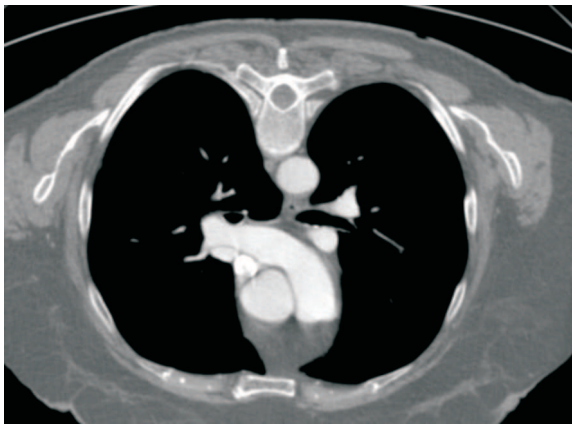
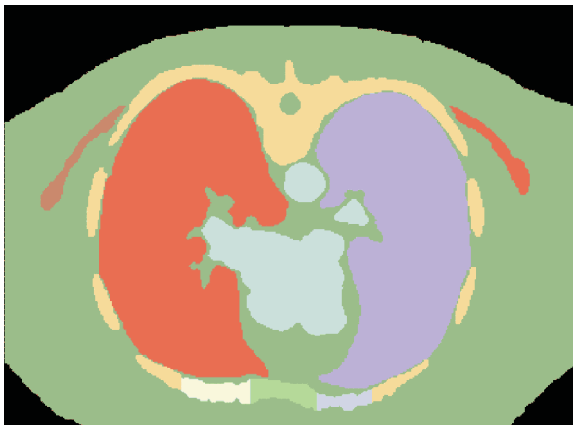


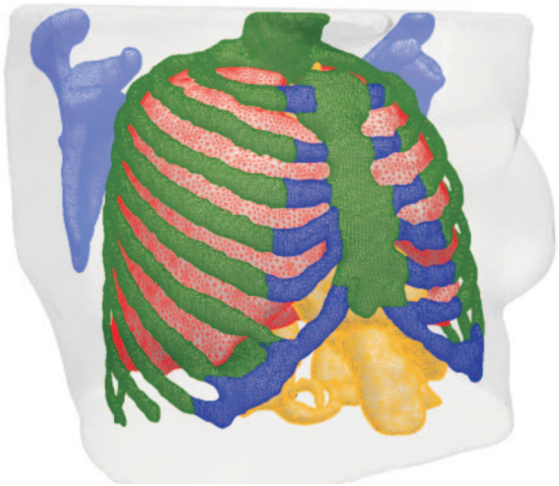
FIGURE 4.8: ***Multi-region surface mesh of the mandible obtained from segmented medical scans with our algorithm.*** *(a) Surface mesh of the mandible, the teeth are visible behind the translucent mesh of the mandible. (b) An enlargement of the mandible-tooth junction shows that the surface meshes join each other consistently. (c) Surfaces of the teeth are smooth, even at material junctions.*



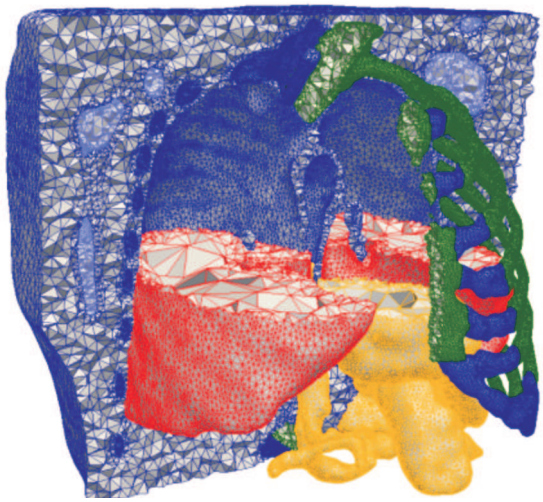
(a) CT-scan



(b) Segmentation

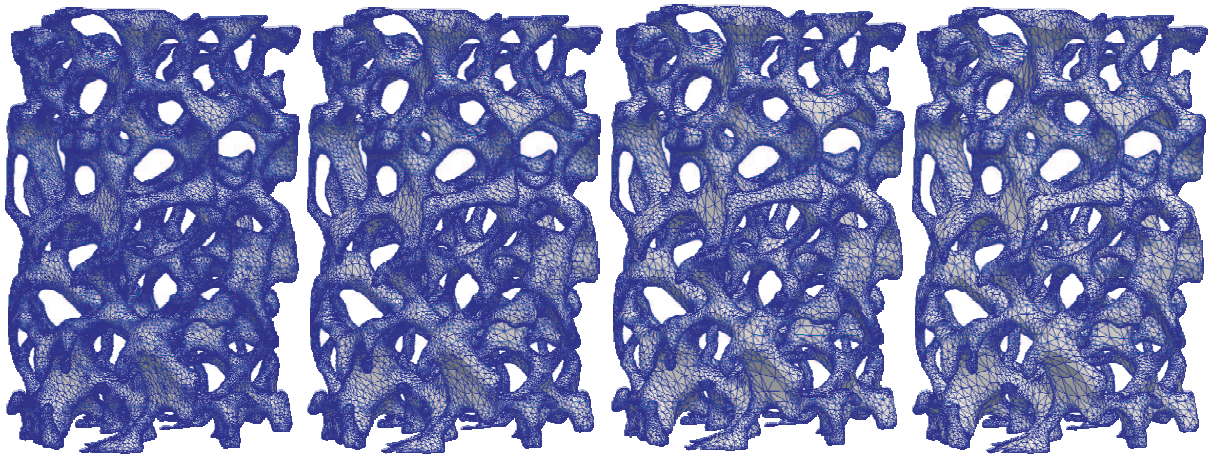


(c) Multi-material surface mesh



(d) Multi-material volume mesh

FIGURE 4.9: ***Multi-region surface mesh of the thorax obtained from segmented medical scans with our algorithm.*** *(a) Axial slice extracted from the provided three-dimensional CT-scan. (b) Segmentation of (a) performed with 3D Slicer [142]. (c) Multi-material surface mesh obtained from (b) via our meshing algorithm. (d) Volume mesh obtained from the surface mesh (c) via TetGen [158]. Project in collaboration with the Royal Military Academy, Brussels [135].*



156 647 *ddls* 117 559 *ddls* 89 490 *ddls* 70 580 *ddls*

FIGURE 4.10: ***Adaptive meshes of an aluminium foam.*** *Successive application of mesh decimation and adaptation results in adaptive meshes, for which element sizes are adapted to the local curvature.*

TABLE 4.2: ***Multi-domain tetrahedral mesh generation and adaptation.*** *Input parameters.*

	$N_{\min}$	$\varepsilon_{\max}$	Δx_s	q
femur	50	$7\,\Delta x$	$4\,\Delta x$	87%
lumbar spine	75	$2\,\Delta x$	$3.3\,\Delta x$	0%
mandible	50	$3\,\Delta x$	$2.5\,\Delta x$	83%
thorax (organes)	200	$5\,\Delta x$	$2\,\Delta x$	60%
thorax (cage)	100	$2\,\Delta x$	$2\,\Delta x$	0%
aluminium	120	$1\,\Delta x$	$1\,\Delta x$	63%

The dataset of the aluminium foam was provided as a binary image by ETH-Zürich so that no segmentation was needed and it could be utilised as such. This dataset was presented in Chapter 3.7 and is illustrated in Figure 3.11 therein.

4.6.2 Parameters setting

The whole meshing process requires the determination of four parameters: $N_{\min}$ and $\varepsilon_{\max}$ control the smoothness of the reconstructed geometry, Δx_s defines the spacing of the sampling grid of the triangulation algorithm, and q specifies the required level of decimation.

The values of $N_{\min}$ and $\varepsilon_{\max}$, indicated in Table 4.2, were obtained as follows. First, the error tolerance $\varepsilon_{\max}$ was adjusted according to the geometric approximation required for the application. This parameter defines the acceptable distance error to the extracted point cloud. It guides the recursive octree-based subdivision process: a cell is divided until the local approximation error is below $\varepsilon_{\max}$. Therefore, it is usually defined as a multiple of the image spacing. A minimum value for $\varepsilon_{\max}$ certainly is half a voxel spacing, so as to avoid stair-stepped artefacts in the reconstruction geometries, but one voxel width gives better results in practice. When a lower resolution is used for the triangulation grid, then $\varepsilon_{\max}$ should be relaxed in accordance.

The minimum number of points required in a subdivision cell, $N_{\min}$, is also indicated in Table 4.2. As explained and illustrated in Section 3.7, this parameter is more tricky to adjust. $N_{\min}$ must be low enough so that the subdivision process may reach the level that gives an approximation error lower than $\varepsilon_{\max}$. However, there is a minimum number of points required in order to accurately evaluate the local approximation functions, particularly when quadratic local functions are used to approximate or interpolate the set of points in a subdivision cell. We recommend a value of $N_{\min}$ above 50 for quadratic functions. Any values above 2 can be used for linear functions.

For the thorax, we used different values for the parameters of the surface reconstruction algorithm, $N_{\min}$ and $\varepsilon_{\max}$, in order to obtain a smoother but less precise result for the inner organs and a geometrically more accurate (with respect to the segmentation) result for the thoracic wall. Our algorithm enables this differentiation easily, however, the same sampling grid has to be used in the triangulation algorithm. Variation of the mesh density from tissue to tissue may however be obtained by our mesh adaptation techniques. In the case of the thorax, the inner organs were decimated, as opposed to the spine and ribs.

Then one has to choose between linear or quadratic, approximating or interpolating functions. Default parameters are linear interpolating functions as these have been proved to give a fast and accurate surface reconstruction for all datasets, noisy or not and for all values of $N_{\min}$. Quadratic approximating functions may be used with caution when the input segmentation is clean and when the object's curvatures are low with respect to the image resolution. In that case, a higher geometric accuracy may be achieved, for appropriate values of $N_{\min}$, as has been illustrated in Section 3.7.

During the triangulation process a sampling grid enclosing the object of interest is defined. The spacing of this grid, called Δx_s, is user-defined. The value is adjusted according to the minimum feature size that must be preserved. When all details of the image are significant, Δx_s is taken equal to the original image spacing Δx. When, however, a high

TABLE 4.3: ***Multi-domain tetrahedral mesh generation and adaptation.*** *Mesh statistics.*

	surface mesh		**volume mesh**	
	nodes	cells	nodes	cells
femur	7956	16114	14052	70600
lumbar spine	52896	105772	90218	419309
mandible	52183	106666	119788	597439
thorax	484203	337451	834111	3227903
aluminium	57865	116338	65941	200482

level of detail is not required, greater values reduce triangle count and mesh generation time.

The last parameter q gives the required level of decimation. As indicated in Table 4.2, we reduced the number of mesh nodes by more than 80% for both the femur and the mandible and by 60% for the aluminium foam.

The obtained surface meshes are illustrated in Figure 4.6 (a), Figure 4.7 (b), Figure 4.8 (a), Figure 4.9 (c) Figure 4.10, for the femur, the lumbar spine, the mandible, the thorax and the aluminium foam respectively. The sizes of these meshes are indicated in Table 4.3.

From these multi-material surface meshes multi-region volume meshes were generated using TetGen [158]. Figure 4.6 (b), Figure 4.9 (d) and Figure 4.7 (c) represent a cut through the volume mesh of the femur, the thorax and the vertebrae. As seen in the figures, the interfaces between different material regions are compatible in the sense of the finite element method. In other words, no gaps nor overlays exist on the material boundaries and the connections between the elements are node-on-node, edge-on-edge, and triangle-on-triangle. The numbers of nodes and tetrahedra composing the volume meshes are also indicated in Table 4.3.

4.6.3 Geometric accuracy

The geometric accuracy of the surface meshes is evaluated by taking each boundary point extracted from the segmented image in turn and measuring its distance to the mesh. This error represents the distance between the original segmented data and the final surface mesh. It includes the geometric error introduced by implicit surface reconstruction and the error resulting from subsequent triangulation and mesh adaptation. It is clear that an

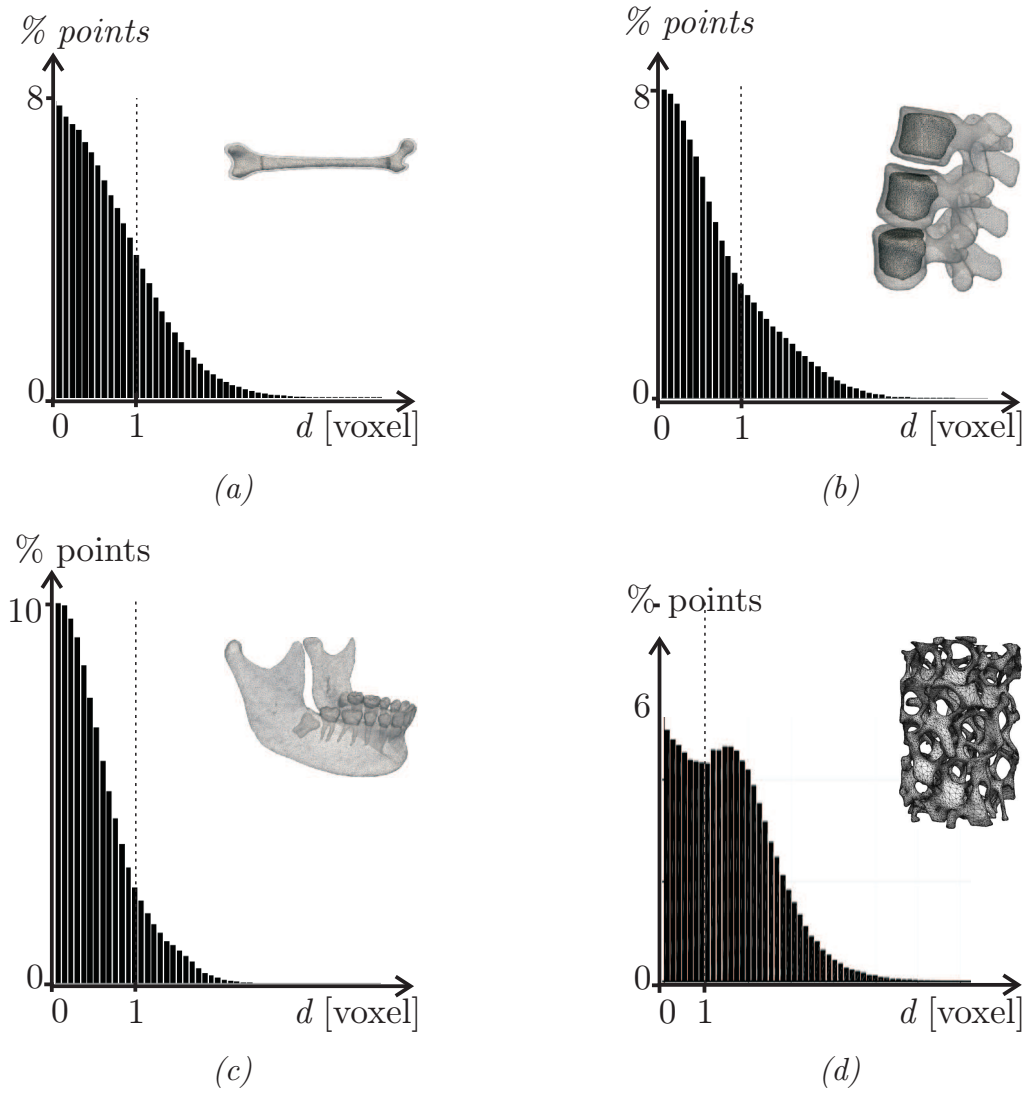


FIGURE 4.11: ***Geometric accuracy.*** *Histogram of the geometric approximation errors, calculated as the distance between the extracted points and the mesh. (a) Femur. (b) Lumbar spine. (c) Mandible.*

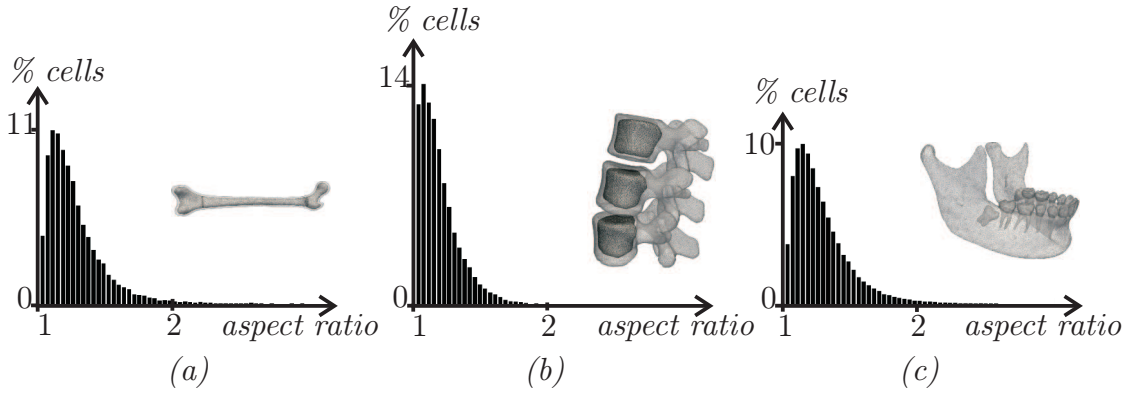


FIGURE 4.12: ***Mesh quality.*** *Histogram of the aspect ratios of the mesh triangles. (a) Femur. (b) Lumbar spine. (c) Mandible.*

error smaller than half a voxel width means that the mesh is in perfect agreement with the segmented data due to the discretisation of the image. By computing this distance for the meshes illustrated in Figures 4.6 (a), Figure 4.7 (b), and Figure 4.8 (a), 36% (femur), 39% (lumbar spine), 48% (mandible) and 28% (aluminium foam) of the extracted points are located less than half a voxel away from the mesh. Moreover, 64% (femur), 66% (lumbar spine), 78% (mandible) and 59% (aluminium foam) of the extracted points are located less than one voxel away from the mesh. A histogram of the obtained geometric errors is drawn for each dataset in Figure 4.11. Even though the meshes do not represent the segmented data exactly, we think that the obtained geometric accuracy is acceptable considering that the segmentation has been performed semi-automatically without applying any subsequent image filtering methods.

4.6.4 Mesh quality

Obtaining surface meshes of high quality is of primal importance as it determines the quality of the volume mesh and the convergence rate of the time integration of the resulting finite element model.

We evaluate the triangular mesh quality by computing the triangle aspect ratios which is defined as the ratio of the longest edge over the shortest one. A histogram of these values is drawn for the femur, the lumbar spine and the mandible in Figure 4.12. In Figure 4.13, histograms of the aspect ratios the several sub-meshes (lungs, shoulder blades, organs and thoracic wall) comprising the multi-material mesh of the thorax are drawn. The obtained meshes are of excellent quality, the maximum aspect ratio being below 3 for all three datasets. This quality is sufficient for further volume mesh generation and accurate finite element computation.

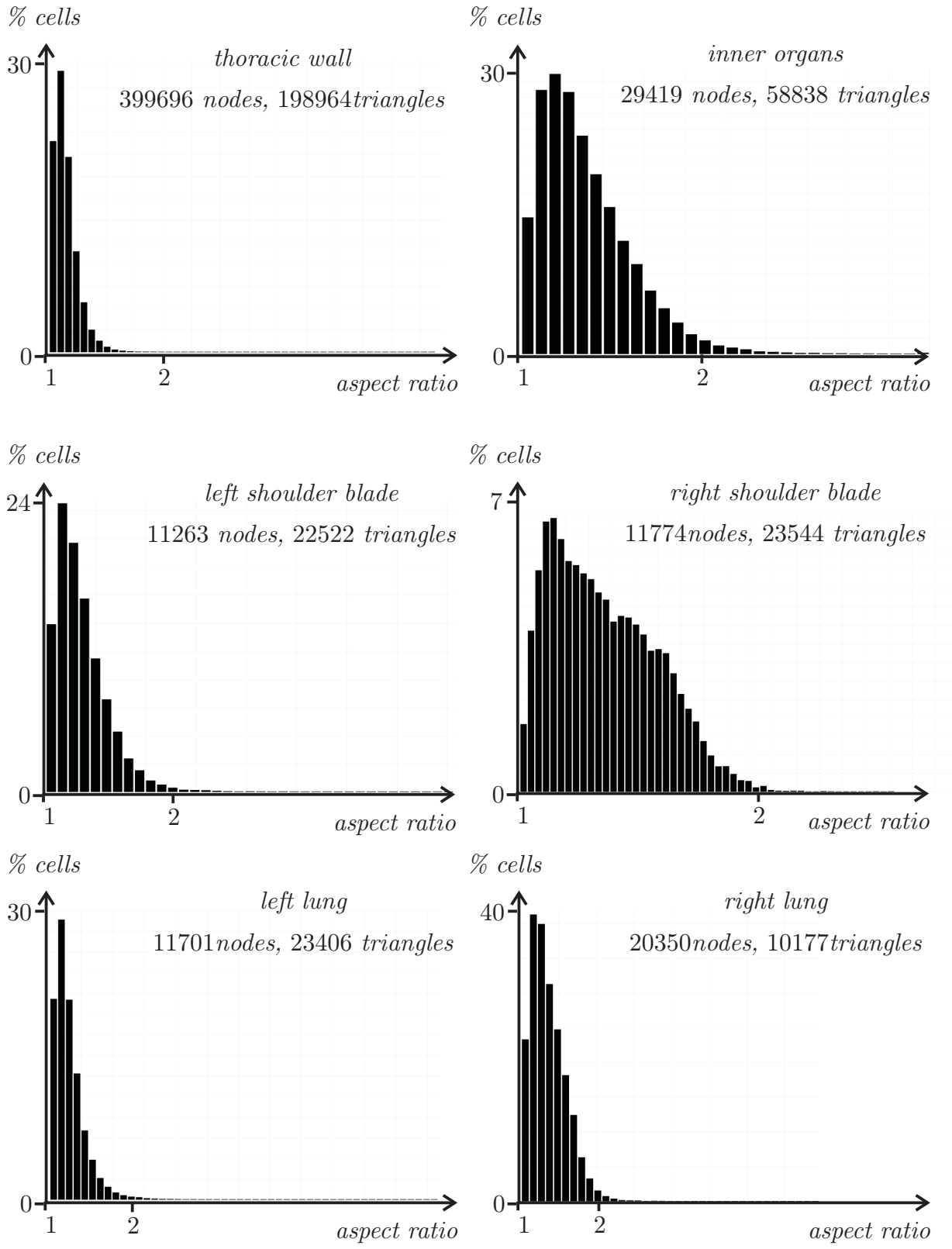


FIGURE 4.13: ***Mesh quality evaluation for the thorax.*** *Histograms of the triangles aspect ratios.*

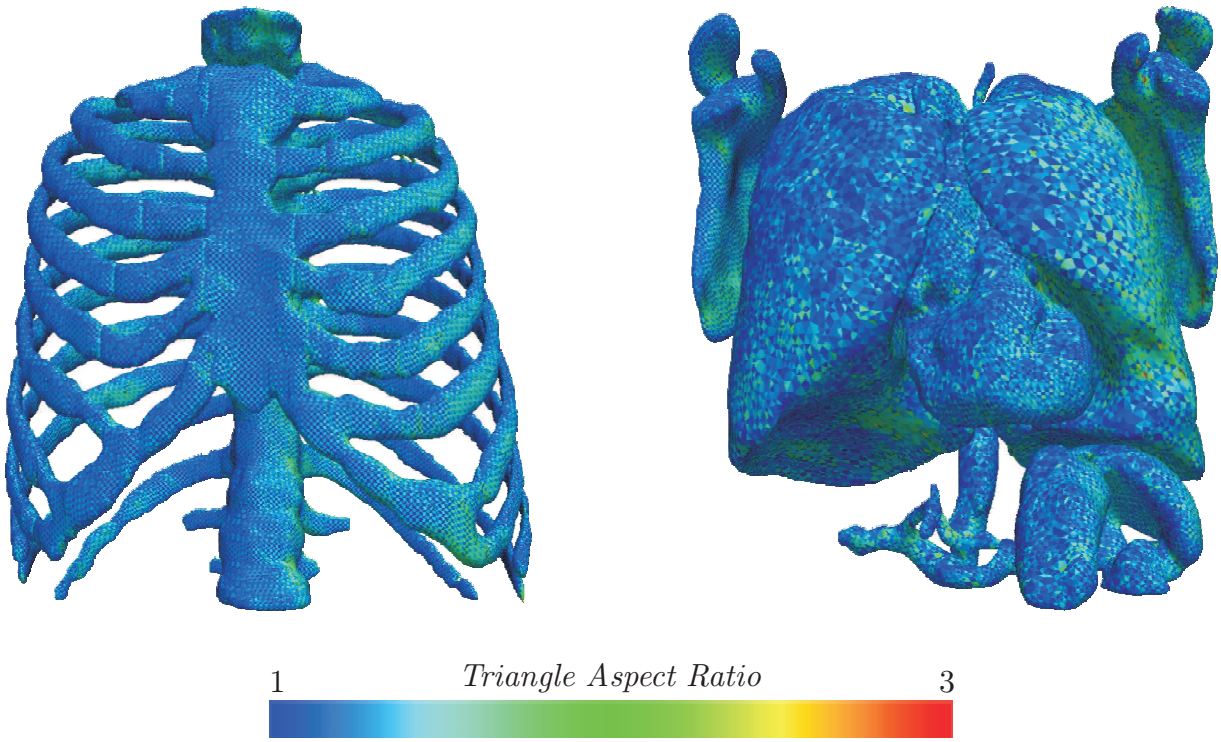


FIGURE 4.14: ***Mesh quality evaluation for the thorax.*** *The surface mesh triangles are coloured with respect to their aspect ratio.*

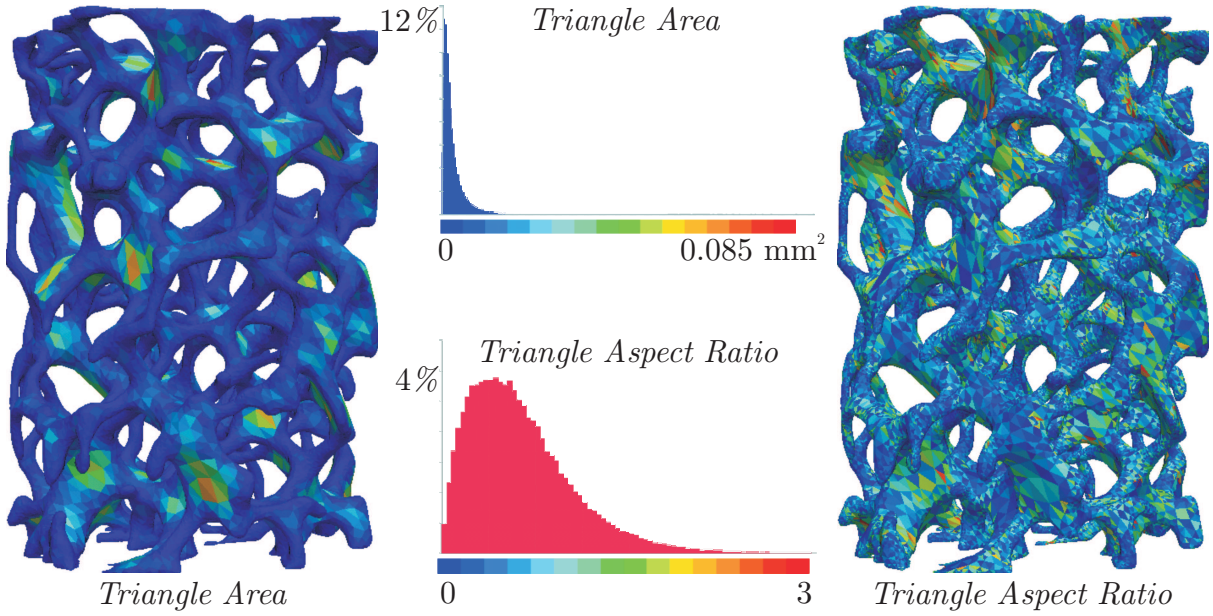


FIGURE 4.15: ***Mesh quality evaluation for the an aluminium foam.***

In Figure 4.14 the surface mesh triangles are coloured with respect to their aspect ratio. The mesh of the thoracic wall was separated from the other surfaces meshes to allow a better visualisation of the quality fields.

Figure 4.15 gives the mesh quality for the aluminium foam. In addition to triangle area and triangle aspect ratio histograms, these properties are also illustrated on the actual mesh.

4.7 Conclusions

This chapter, on mesh generation, along with the previous chapter, on surface reconstruction, present a mesh generation strategy capable of producing triangle surface meshes from multi-tissue segmented medical datasets. The proposed method solves the two main issues of patient-specific mesh generation. First, the typical staircase artefacts resulting from image discretisation are avoided by computing a smooth description of the tissue boundaries prior to triangulation (Chapter 3). Second, multi-tissue mesh generation is enabled by using a multi-material version of the marching tetrahedra method (Chapter 4). The main contributions of this chapter are (1) an efficient implementation of a multi-material marching tetrahedra algorithm, based on a novel description of multi-material structures, (2) a strategy to accurately position interface nodes during mesh generation, which greatly improves the quality of the meshes along material junctions, (3) a multi-material decimation scheme that may be used during and/or after mesh generation and (4) a volume-preserving mesh adaptation filter.

The efficiency of our meshing procedure was illustrated on five datasets: a femur, a set of three lumbar vertebrae, a mandible with its teeth, a thorax and an aluminium foam. Multi-material meshes of respectively 4, 6, 18 and 10 material regions and a single material mesh were created from the segmented volumes. In each case topologically consistent meshes were obtained with no gaps or overlays at the material junctions. The results also show a very small geometric approximation error and satisfactory triangle aspect ratios.

Chapter 5

Multi-domain hexahedral mesh generation and adaptation

5.1 Motivation

In the previous chapter (Chapter 4), an efficient strategy to create tetrahedral meshes from multi-material biological structures was proposed. A key strength of the proposed algorithm is that it reconstructs the geometries of anatomical tissues very accurately. Indeed the use of the surface reconstruction algorithm presented in Chapter 3 enables recovery of the natural smoothness of biological structures, lost by discretisation during the scanning process, whilst keeping the surface boundaries within the limits imposed by the voxel values in the medical image. This surface reconstruction algorithm is used prior to mesh generation as well as in a post-processing step. In the latter, the quality of the generated mesh is improved by repositioning the surface mesh nodes but constraining them to remain on the implicitly defined surface of the surface reconstruction algorithm.

The idea investigated in this chapter is to use our surface reconstruction algorithm to smooth the jagged edges produced by a voxel conversion algorithm.

This resulting meshing strategy outputs hexahedral meshes and therefore avoids problems arising from using the standard linear tetrahedral element in finite element simulations of incompressible materials. Moreover, it outputs meshes with smooth surface boundaries, so that the stress concentration and contact problems arising with voxel-based meshes are also solved. This novel meshing algorithm is also extremely time-efficient as voxel conversion is straightforward and because our surface reconstruction algorithm uses an octree-based subdivision scheme so that its computation time depends on the complexity of the structure rather than the image size. Finally, the approach is well adapted for the

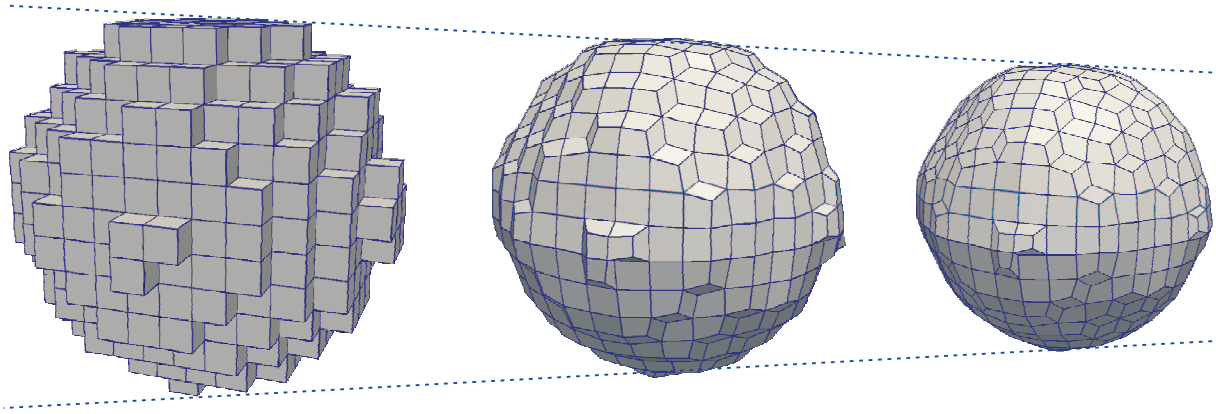


FIGURE 5.1: ***Laplacian smoothing and mesh shrinkage.*** *Illustration of the mesh shrinkage occurring during Laplacian smoothing a voxel-based mesh.*

generation of structures composed of several material domains and, also, allows to assign heterogeneous material properties based on the image greyscale values[1] [47].

The major drawback in smoothing a voxel-based hexahedral mesh is that it generates distorted elements along the objects' boundaries. Finite element simulations will be performed at the end of this dissertation to study the effect of this distortion on simulations results. A strategy to control and limit element deterioration during the meshing procedure is also proposed. As opposed to our tetrahedral mesh generation procedure (Chapter 4), voxel based meshing also has the disadvantage that the mesh resolution is not user-controlled: it is fixed by the image resolution.

Finally a key motivation for the implementation of this new meshing algorithm is to be able to present a comparison of different meshing strategies and their effect on finite element simulations. Researchers tend to stay in their specific field, mesh generation or finite element simulation, so that these comparisons are relatively rare.

5.2 Literature Review

Voxel-based meshing enables fast and automatic generation of hexahedral meshes from scanned data. It has extensively been used for micro-FE simulations of trabecular bone [28, 69, 82, 172]. Because of its simplicity voxel-conversion has also been perfomed for macroscopic studies of the femur [42], the distal radius [143] and the vertebral body [47, 94].

[1] This is an alternative option to multi-domain mesh generation, that will be investigated in future studies.

Nevertheless, the basic conversion of one voxel into a brick element produces jagged edges. These jagged edges have been proven detrimental to the accuracy of finite element results [173]. Non-smooth surface boundaries also jeopardises contact modelling and simulation [33, 78]. Subsequent smoothing of these boundaries improves the accuracy of the finite element results [28].

Several smoothing approaches have been proposed [12, 28, 33, 78]. The two major problems faced by these researchers are (1) element distortion (2) mesh shrinkage. Element distortion inevitably occurs when smoothing the boundary of a voxel-based brick mesh. An efficient remedy is to split the distorted elements into prisms [12], thus creating hybrid prism-hexahedral meshes. Unfortunately, not all finite element software include the possibility of finite element simulations from hybrid meshes. The second challenge, mesh shrinkage, comes from the iterative smoothing of volumes using classical smoothing algorithms. Because the underlying image is not taken into account during node repositioning, the resulting mesh is no longer a good representation of the scanned object after smoothing. Controlling mesh shrinkage is particularly important for the modelling of trabecular bone where mesh shrinkage may lead to the collapsing of trabecular connections and where simulations results highly depend on the volume of the trabeculae.

5.3 Proposed approach

The proposed approach is summarised in Figure 5.2. The input of the hexahedral mesh generator is a three-dimensional, usually segmented, image, or equivalently, a set of two-dimensional parallel scans (Figure 5.2 (a)). Either binary images or grey-scale images are accepted by our algorithm. In all cases, voxels with zero value will be considered as background.

Voxels with non-zero values are turned into brick elements in the first step of our algorithm (Figure 5.2 (b)). Each generated element is associated with its voxel value, provided by the input image.

In a second step, the set of points and associated normals, needed for surface reconstruction (Section 3.3) is computed (Figure 5.2 (c)). The procedure is similar to the one presented in Section 3.4.1, but this time, the points are extracted from the voxel mesh and not from the segmented image. The resulting input points and normals sets are identical to the ones obtained directly from the segmented dataset. However extracting these sets directly from the mesh allows us to keep in the computer memory only one representation of the geometry at a time. To extract these sets from the voxel mesh, the method of Section 3.4.1 is extended as follows. The mesh nodes belonging to the surface of the

voxel mesh are first identified as those having less than eight cell-neighbours. For each of these generated boundary points, a normal is computed by averaging the face-normals of the neighbouring elements. These normals are then iteratively smoothed using Equation (3.27), in order to improve the quality of the reconstructed surface (see Section 3.4.1 for more details).

From these sets of input points and normals, an implicit function approximating the distance to this set of points, and henceforth the distance to the object's boundaries, is constructed by the multi-level partition of unity surface reconstruction method detailed in Section 3.3 (Figure 5.2 (d)).

The last step consists in iteratively deforming the brick mesh towards the zero-level of the defined $\mathcal{C}^1$ distance function. However, we use an alternate version of the mesh adaptation algorithm defined for our patient-specific tetrahedral mesh generation method (Section 4.5). Indeed, in the case of a hexahedral mesh, a better result is achieved by slowing down the node projection algorithm. This allows us to add a criterion on a maximum-allowed element distortion, so that the mesh nodes are projected towards $f(\boldsymbol{x}) = 0$ only to the point where the maximum allowed distortion is achieved. The resulting mesh adaptation algorithm can be summarised as follows:

Until a criterion is met, perform those two steps successively:

1. Loop over the nodes belonging to the boundary mesh and projects these nodes towards the target surface. For each mesh node belonging to the surface of the voxel-based mesh, i.e. for each node having less than eight cell-neighbours:

   (a) Compute the node neighbourhood defined as its closest node-neighbours, i.e. the nodes connected to the current node via an edge (three to five nodes).
   (b) Reposition the node at the centre of its neighbourhood, $\boldsymbol{x}_0$.
   (c) Compute the node's target position $\boldsymbol{x}_t$ on the implicit surface surface $f(\boldsymbol{x}) = 0$, using a Newton-Raphson procedure.
   (d) Reposition the current node at $0.5\boldsymbol{x}_0 + 0.5\boldsymbol{x}_t$, that is to say, at mid-distance towards the target surface.

2. Equilibrate the mesh in volume. For each mesh node located inside the voxel-based mesh:

   (a) Compute the node neighbourhood defined as its closest node neighbours, i.e. the six nodes connected to the current node via an edge.
   (b) Reposition the current node at the centre of its neighbourhood.

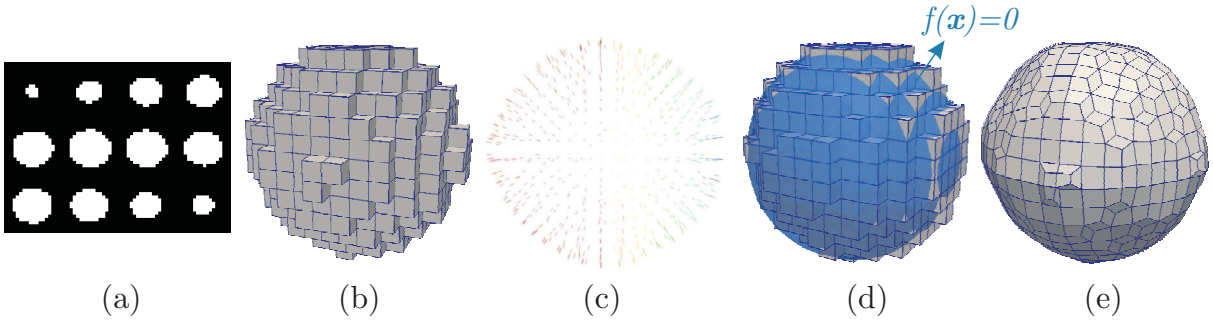


FIGURE 5.2: ***Proposed patient-specific hexahedral mesh generation procedure.*** *(a) Set of segmented image slices, obtained from the scanning of a sphere. (b) Brick mesh obtained by voxel conversion. (c) Set of points and associated normals extracted from the boundary surface mesh of the generated voxel mesh. (d) Representation of the surface boundary by a smooth implicit function thanks to our surface reconstruction algorithm. (e) Iterative smoothing and projection of the mesh boundary nodes towards the zero-level of the implicit distance function* $f(\boldsymbol{x}) = 0$.

The criterion used to terminate the projection algorithm result in a trade-off between surface smoothness and element deformation.

Maximum surface smoothness is achieved when the boundary nodes are located on $f(\boldsymbol{x}) = 0$. Therefore, in our algorithm, a measure of the surface smoothness is computed as the maximum distance of the surface nodes to the target smooth surface, which is given by the value of the implicit function at this mesh node $f(\boldsymbol{x}_n)$.

The distortion of a hexahedral element is measured by [162]

$$d_h = \frac{8 \min_{\xi_k} \left(J_{\xi_k} \right)}{V_h} \tag{5.1}$$

where $\min_{\xi_k}(J_{\xi_k})$ is the minimum determinant of Jacobian matrix evaluated at each Gauss point ξ_k and V_h is the volume of the hexahedron. This quality measure gives a result between 0 (very distorted element) and 1 (cubic element).

In the case of hexahedral mesh generation from a segmented image containing a set of different material regions, labelled with different voxel values, the user may choose to smooth the inner boundaries along with the outer boundary. The procedure presented in Section 3.5 is then used to create a representation of the multi-material object with a set of implicit distance functions. During the initial voxel-conversion algorithm, boundary nodes are labelled according to the boundary, inner or outer, they belong to. These nodes are then iteratively projected towards their corresponding target implicit surface using the mesh adaptation algorithm presented in this chapter.

5.4 Applications and results

5.4.1 Hexahedral mesh generation of truss-like structures

Advances in microfocus 3D computer tomography and magnetic resonance imaging have made it possible to perform numerical simulations on microstructures such as metal foams and trabecular bones. The most commonly used technique to perform these simulations is finite element simulation, called μ-finite element simulation within this framework. The main challenge faced by μFE analysis is the building of a finite element mesh. Indeed, due to the limited image resolution and the complexity of the involved structures, classical meshing techniques fail to accurately and robustly reconstruct the architecture.

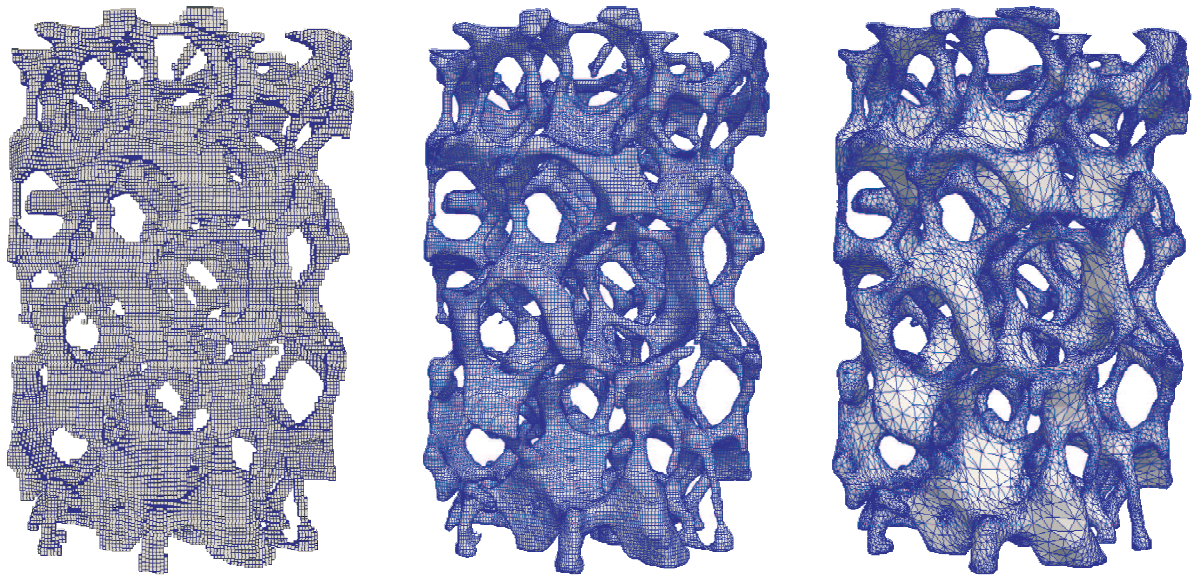
The meshing procedures proposed in literature may be classified as *model-based* or *model-free*. Model-based approaches rely on model assumptions (e.g. periodicity, open-cell or closed-cell) combined with statistical data extracted from the images and use specific methods to reconstruct the microstructure. Model-free approaches take the segmented image as input and reconstruct the geometry as smoothly and accurately as possible. Unlike model-based approaches, they are applicable to any kind of structures and offer a geometric accuracy that cannot be achieved with model assumptions.

The idea of this section is to investigate different types of model-free approaches for the modelling of microstructures:

1. voxel-conversion
2. our proposed *mpu-smoothed* voxel-based meshing strategy, presented in this chapter
3. our proposed tetrahedral mesh generation strategy (Chapter 4)

The first dataset that is considered is the *Aluminum foam* of Figure 5.3, already presented in Sections 3.7 and 4.6. The aluminium foam resembles the complex architecture of trabecular bone, which is typically the type of dataset for which voxel-conversion is used in literature. Figure 5.3 illustrates the meshes obtained by means of voxel-conversion (Left), the proposed hexahedral mesh generation strategy (Middle) and the proposed tetrahedral mesh generation approach (Right). Five projection iterations were performed during our mpu-based hexahedral mesh generation, which allowed the mesh nodes belonging to the boundary surface to be totally projected on the implicitly defined surface. The objective in this case was more to illustrate that smooth surfaces may be achieved with our algorithm rather than creating a good quality finite element mesh. Element distortion controlled mesh adaptation is illustrated for the next dataset.

The second dataset that is considered is a μ-CT scan of the cancellous tissue of a deer (Cervus Elaphus) antler, prepared at the Department of Clinical Sciences, Faculty of Vet-



295 870 *nodes*
208311 *hexahedra*

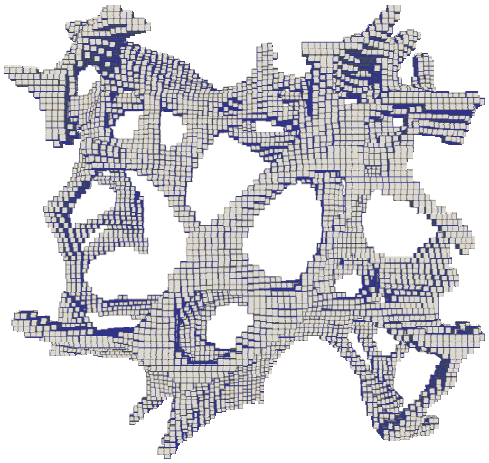
295 870 *nodes*
208311 *hexahedra*

104 969 *nodes*
333 543 *tetrahedra*

FIGURE 5.3: ***Voxel-based, hexahedral and tetrahedral meshes of an aluminium foam.*** *Meshes of the aluminium foam described in Figure 3.11. Left: voxel-conversion. Middle: proposed hexaheral hexahedral mesh generation strategy. Right: proposed tetrahedral mesh generation approach.*

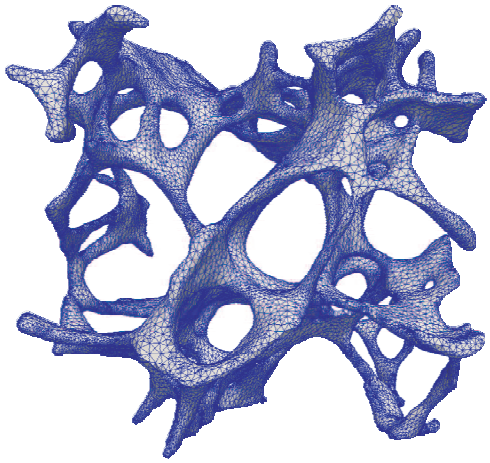
erinary Medicine, University of Liège [102]. Figure 5.4 illustrates the meshes obtained by (1) voxel-conversion, (2) our hexahedral mesh generation procedure with one projection iteration, (3) our hexahedral mesh generation procedure with two projection iterations, (4) our tetrahedral mesh generation procedure (Chapter 4). Visually the smoothness of the generated structure is improved by successive projection of the boundary surface nodes on the multi-level partition of unity implicit surface. Nevertheless, the boundary surfaces are not as smooth as for the tetrahedral mesh. The reason for this obviously is that during our voxel-based mesh smoothing, boundary nodes are moved towards the implicitly defined surface; but, their movement is restricted in order to avoid too large distortions of the hexahedral elements.

Figure 5.5 illustrate the same four meshes, but the elements are coloured according to their quality. For hexahedral elements the quality measure used is the element distortion (5.1), for tetrahedral elements, the triangle ratio is used.



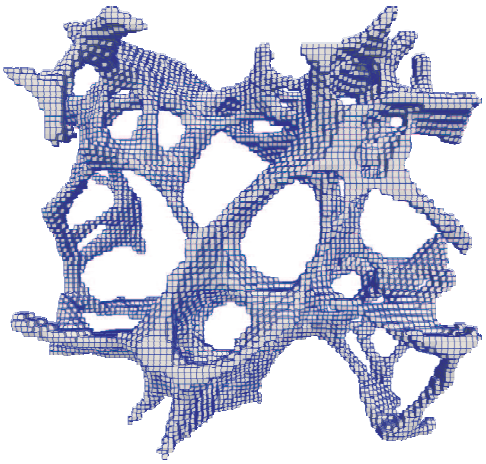
voxel-based

65 144 *nodes*
38 581 *hexahedra*
0.672 mm^3



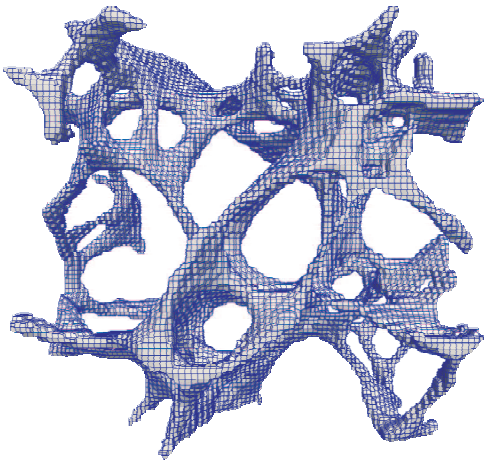
proposed tetrahedral mesh

66 013 *nodes*
232 859 *hexahedra*
0.677 mm^3



proposed hexahedral mesh - 1

65 144 *nodes*
38 581 *hexahedra*
0.643 mm^3



proposed hexahedral mesh - 2

65 144 *nodes*
38 581 *hexahedra*
0.619 mm^3

FIGURE 5.4: ***Voxel-based, hexahedral and tetrahedral meshes generated of a deer-antler.*** *Mesh generation from a μ-CT scan of the cancellous tissue of a deer antler [102].*

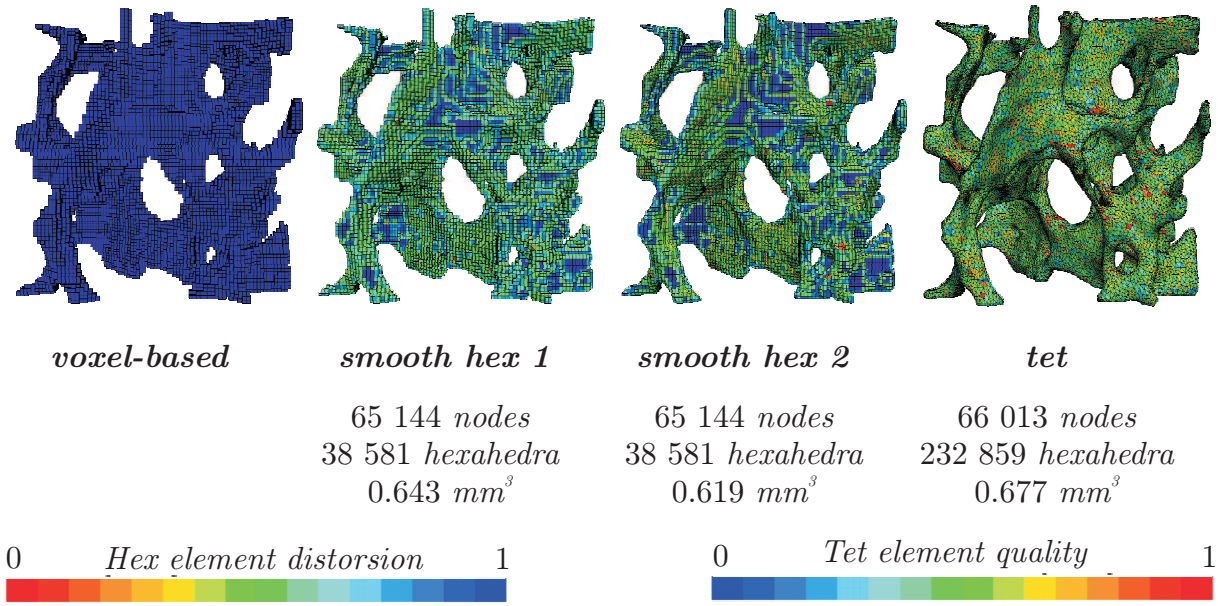
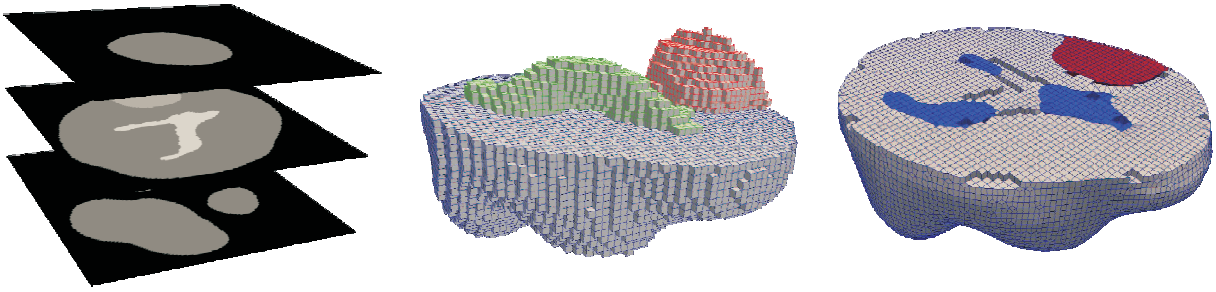


FIGURE 5.5: ***Voxel-based, hexahedral and tetrahedral meshes generated of a deer-antler.*** *Evaluation of the mesh qualities.*

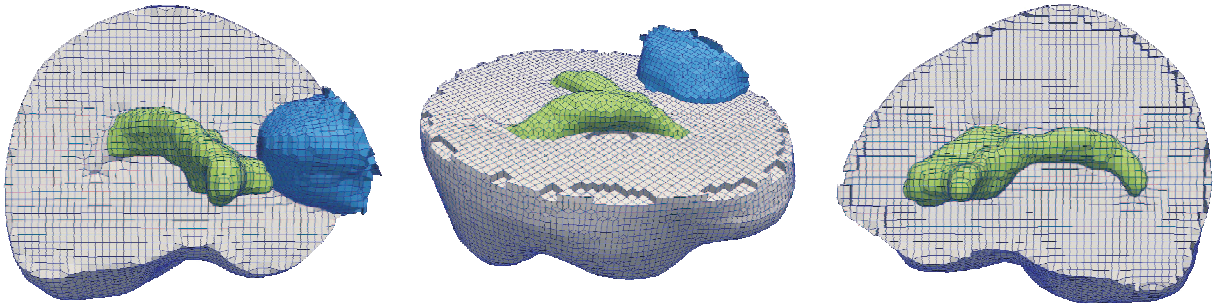
5.4.2 Hexahedral mesh generation of multi-material structures

Figure 5.6 illustrates our hexahedral mesh generation approach in the case of non-binary segmented datasets. A magnetic resonance, three-dimensional, image of the brain, of dimensions $256 \times 256 \times 60$ and spacing $0.9375 \times 0.9375 \times 2.5$ mm was segmented into three material domains: healthy brain, ventricles and tumour (Figure 5.6, Left). In a first pre-processing step, a nearest-neighbour reslice filter was applied on the dataset in order (1) achieve an isotropic spacing and (2) down-sample the data by a factor three. An isotropic spacing is required for the generation of cubic, as opposed to elongated, brick elements. The down-sampling was used to limit the number of hexahedral elements generated during the voxel conversion algorithm. The produced image had an isotropic spacing of $2.8 \times 2.8 \times 2.8$ mm and $86 \times 86 \times 53$ voxels. The first step of our algorithm consists in turning each of these voxels into hexahedra. The resulting voxel mesh is shown in Figure 5.6, Middle. As illustrated with colours, the voxel values of the initial image are kept in memory during voxel conversion. Outer and inner surface boundaries are then iteratively smoothed using the distortion controlled mesh adaptation method presented above. This algorithm also includes a procedure to propagate this smoothing towards the interior of the material volume regions. The latter is more noticeable in Figure 5.7, where the hexahedral mesh of the healthy brain part has been cut along the three planes of the coordinate system.



Segmented dataset *voxel mesh* *mpu-smoothed mesh*

FIGURE 5.6: ***Hexahedral mesh generation of multi-material structures***. *Illustration of our hexahedral mesh generation procedure on a multi-label dataset of the brain.*



x-*cut* z-*cut* y-*cut*

FIGURE 5.7: ***Multi-material hexahedral mesh of the brain obtained by the proposed hexahedral mesh generation procedure.*** *The part of the mesh corresponding to healthy brain and labelled as such in the initial segmented scans has been cut along the three planes of the coordinate system.*

5.5 Conclusions

The hexahedral mesh generation procedure presented in this chapter is a good alternative to tetrahedral mesh generation, when a tetrahedral mesh is not desired; for example under incompressibility conditions when the finite element software used does not include a non-locking tetrahedral finite element.

The algorithm produces fairly smooth mesh boundaries which is important for the accuracy of the results computed by finite element analysis.

The main drawback of the proposed approach is that distorted elements are generated along the surface boundaries. This is an inevitable result of the smoothing of voxel meshes. However this mesh distortion is user-controlled: the user may define a maximum-allowed hexahedral element distortion. Moreover, as opposed to recent approaches proposed in literature [12], our surface smoothing and mesh adaptation algorithm includes a strategy

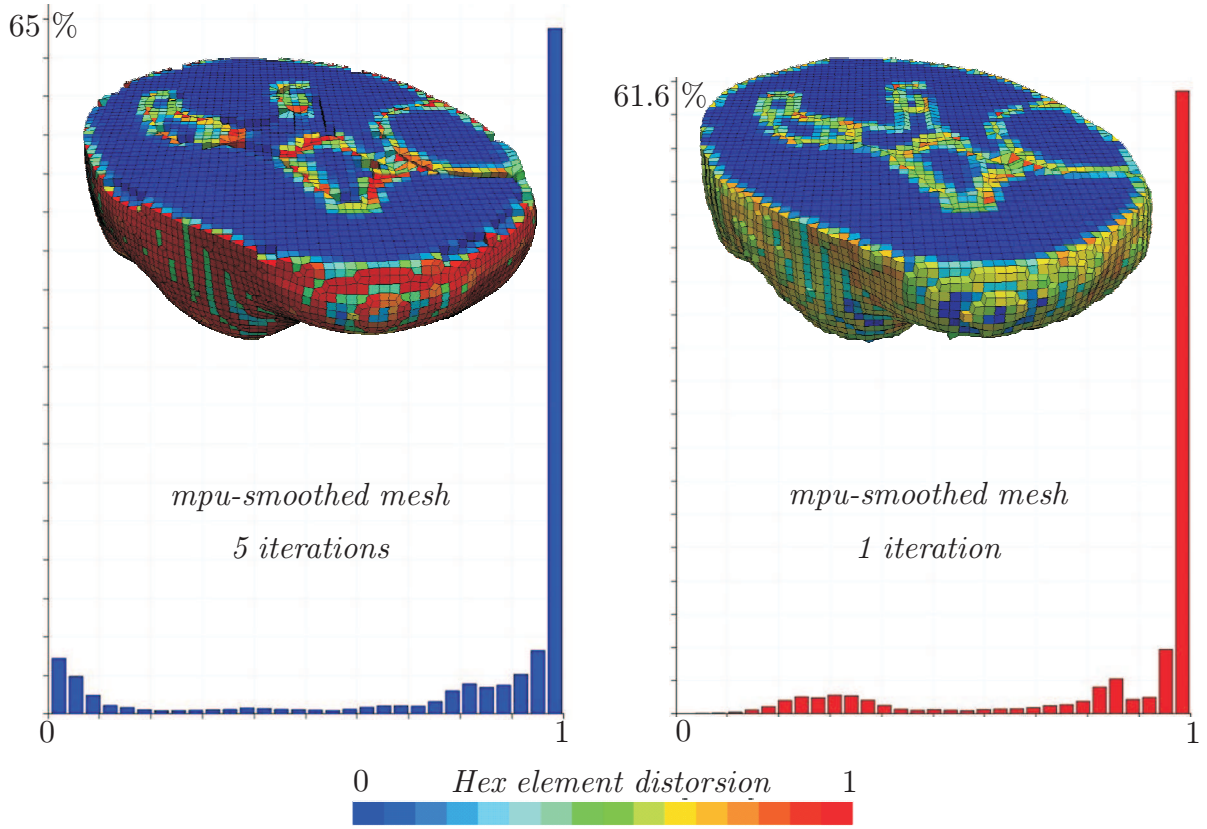


FIGURE 5.8: ***Multi-material hexahedral mesh of the brain obtained by the proposed hexahedral mesh generation procedure.*** *Quality analysis. Left: Element distortion field and histogram obtained for a completely smoothed hexahedral mesh. Right: Element distortion field and histogram obtained when the mesh adaptation algorithm is stopped after one iteration in order to avoid large element distortion.*

to propagate the surface smoothing within the volume; which limits element distortion. To solve this problem distorted hexahedral elements along the boundaries could be subdivided into prism or tetrahedral elements in the future.

Finite element simulations will be performed at the end of this dissertation to evaluate the efficiency of this approach as compared to our tetrahedral mesh generation approach.

Part II

Unlocking the linear tetrahedron

Notations

General Rules

a , A , α	:	*italic character*, scalar
$\boldsymbol{a}$, $\boldsymbol{A}$, $\boldsymbol{\alpha}$	:	*bold italic character*, vector
		(except the Cauchy stress tensor $\boldsymbol{\sigma}$ and its deviator $\boldsymbol{s}$)
$\mathbf{A}$	:	*bold upright in upper case*, matrix or second order tensor
$\boldsymbol{x}$, x	:	*lower case*, variable in the current (spatial) configuration
$\boldsymbol{X}$, X	:	*upper case*, variable in the reference (material) configuration
0 , as in $\mathbf{B}^0$	:	*superscript or subscript*, reference (material) configuration
e , as in V_e	:	*lower case, superscript or subscript*, element contribution
I , as in V_I	:	*upper case, superscript or subscript*, nodal value
δ_{ij}	:	Kronecker symbol $\delta_{ij} = 1$ if $i = j$ and 0 otherwise
$\nabla_0 \cdot$	:	gradient with respect to the reference configuration $\nabla_0 \cdot = \frac{\partial \cdot}{\partial \boldsymbol{X}}$
$\nabla \cdot$	:	gradient with respect to the current configuration $\nabla \cdot = \frac{\partial \cdot}{\partial \boldsymbol{x}}$
$\cdot$, as in $\boldsymbol{a} \cdot \boldsymbol{b}$	:	contraction of inner indices;
	:	$\boldsymbol{a} \cdot \boldsymbol{b} = a_i b_i$, $\mathbf{A} \cdot \boldsymbol{b} = A_{ij} b_j$, $\mathbf{A} \cdot \boldsymbol{B} = A_{ij} B_{jk} = A_{ij} B_{kj} = \mathbf{A}\mathbf{B}^T$
: , as in $\mathbf{A} : \mathbf{B}$	:	double contraction of inner indices; $\mathbf{A} : \mathbf{B} = A_{ij} B_{ij}$, $\mathbf{C} : \mathbf{D} = C_{ijkl} D_{kl}$
$\dot{}$, as in $\dot{\boldsymbol{u}}$	:	*superscript*, first order time derivative (total derivative)
$\ddot{}$, as in $\ddot{\boldsymbol{u}}$	:	*superscript*, second order time derivative (total derivative)

General Remarks

- Coordinate system indices are denoted $i, j, \ldots$ in the current configuration and $A, B, \ldots$ in the reference configuration.
- The Einstein summation convention is used. Therefore, when an index occurs more than once in the same expression, the expression is implicitly summed over all possible values for that index.

- In the finite element method, the global system of equations, governing the behaviour of the domain Ω, consists in the assembly of local equations, governing the behaviour in the sub-domains Ω^e. Apart from the assembly operator, governing equations are similar. Therefore, the subscript or superscript e will often be dropped in this dissertation in order to simplify the notations.

Variables

$\boldsymbol{b}$	:	body force
$\mathbf{B}$	:	left Cauchy-Green deformation tensor
$\mathbf{C}$	:	right Cauchy-Green deformation tensor
$\boldsymbol{d}$	:	nodal displacements stored in Voigt form
$\mathbf{D}$	:	rate-of-deformation tensor
$\mathbf{E}$	:	Green-Lagrange strain tensor
$\dot{\mathbf{E}}$	:	material strain rate tensor
$\mathbf{f}^{\text{int}}$, $\boldsymbol{f}_I^{\text{int}}$, f_{iI}^{int}	:	internal nodal forces
$\mathbf{f}^{\text{ext}}$, $\boldsymbol{f}_I^{\text{ext}}$, f_{iI}^{ext}	:	external nodal forces
$\mathbf{F}$, F_{ij}	:	deformation gradient; $F_{ij} = \partial x_i / \partial X_j$
$\mathbf{G}^0$, G_{Ii}^0	:	discrete material gradient operator; $G_{Ii}^0 = \partial N_I / \partial X_i = \nabla_0 \mathbf{N}_I$
$\mathbf{G}$, G_{Ii}	:	discrete spatial gradient operator; $G_{Ii} = \partial N_I / \partial x_i = \nabla \mathbf{N}_I$
J	:	determinant of Jacobian matrix between spatial and material coordinates
J_ξ	:	determinant of Jacobian matrix between spatial and element coordinates
J_ξ^0	:	determinant of Jacobian matrix between material and element coordinates
$\mathbf{K}$	:	linear stiffness matrix
$\mathbf{K}^{\text{int}}$, $\mathbf{K}^{\text{ext}}$	:	tangent stiffness matrix for internal and external forces
$\mathbf{K}^{\text{mat}}$, $\mathbf{K}^{\text{geo}}$	:	material and geometric tangent stiffness matrices
$\boldsymbol{n}^0$, $\boldsymbol{n}$	:	normal vector to the initial and current boundary of the domain
$\mathbf{P}$	:	first Piola-Kirchhoff or nominal stress tensor
p	:	pressure
$\boldsymbol{s}$	:	deviator of Cauchy stress tensor
$\mathbf{S}$	:	second Piola-Kirchhoff stress tensor
$\boldsymbol{t}$	:	traction force
$\boldsymbol{u}$, u_i	:	displacement field (non discretised)
$\mathbf{u}$, u_{iI}	:	matrix of nodal displacements
$\mathcal{U}$	:	space of kinematically admissible displacements
$\boldsymbol{v}$, v_i	:	velocity field
V , v	:	volume in the reference and current configuration
$\mathcal{W}^{\text{int}}, \mathcal{W}^{\text{ext}}, \mathcal{W}^{\text{kin}}$	:	internal, external and kinetic work
$\boldsymbol{x}$, x	:	spatial (Eulerian) coordinates
$\boldsymbol{X}$, X	:	material (Lagrangian) coordinates
Γ , Γ_0	:	boundary of the body in the current and initial configuration
$\Gamma_{\boldsymbol{u}}$	:	displacement boundary: part of boundary where displacement is prescribed
$\Gamma_{\boldsymbol{t}}$	:	traction boundary: part of boundary where traction is prescribed

Π	:	energy potential
ρ , ρ_0	:	mass density
$\boldsymbol{\sigma}$	:	Cauchy stress tensor
Ω , Ω_0	:	domain of current and initial configuration
${}^h\Omega$, ${}^h\Omega_0$	:	reference to the meshed domain
$\square$	:	element's parent domain

Chapter 6

Background

6.1 The Finite Element Method

The finite element (FE) method is a powerful tool to evaluate stresses and strains in solids with geometric or material non-linearities. In many cases and especially in Biomechanics, the equilibrium equations and constitutive laws are highly non-linear so that an analytical solution cannot be found. In that situation, the finite element method is a good alternative to evaluate the strains and stresses in response to the solid's loading history.

The reader is supposed to be familiar with the basics of the FE method, and is referred to reference manuals otherwise [86, 194]. The first step in finite element modelling is the subdivision of the initial domain into non-overlapping elements, connected to each other at their nodes and on their edges. This procedure, called meshing, has been the purpose of the first part of this dissertation. The displacement field is then evaluated at the mesh nodes by expressing the equilibrium equations at each node. The overall field is evaluated within an element by interpolating its nodal values using shape functions. Strains and stresses are evaluated at quadrature points using strain-displacements relations and constitutive laws respectively. The formulation of an adequate tetrahedral finite element is the purpose of the present chapter.

This chapter is organised as follows. In Section 6.2, we introduce some elements of continuum mechanics. This will enable the reader to get familiar with the notations used in this thesis. Section 6.3 introduces the principle of virtual work and the principle of virtual power. In Section 6.4, an expression for the internal tangent stiffness matrix is obtained by successive linearisation and discretisation of the principle of virtual work equation. Finally, implicit and explicit time integration are introduced in Section 6.5. All together, these Sec-

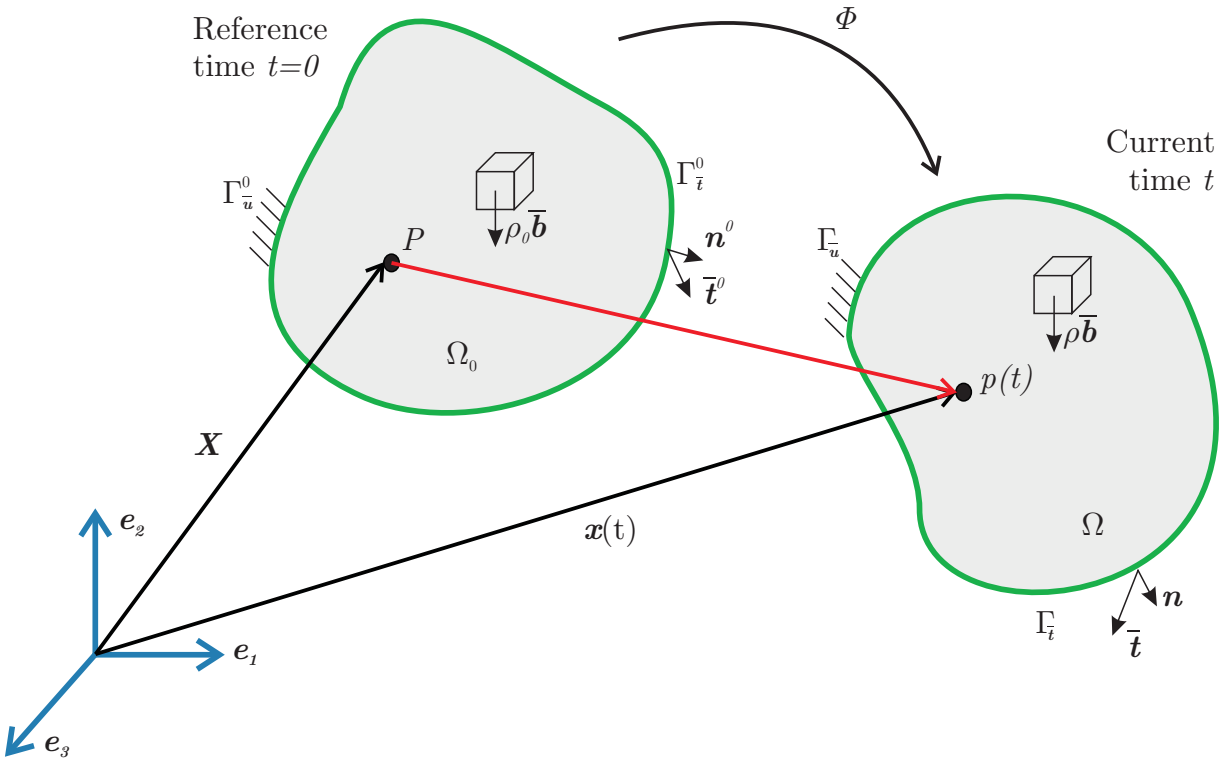


FIGURE 6.1: ***Continuum mechanics.*** *Description of the motion of a deformable body and introduction of the notations used in this chapter.*

tions will give the reader the basis to understand Chapter 7 on finite element formulations that solve the locking problems of the standard linear tetrahedron.

6.2 Continuum mechanics

6.2.1 Notations

Figure 6.1 introduces the notations used in this work. The body Ω_0 is imagined as being an assemblage of material particles P that are labelled by the coordinates $\boldsymbol{X}$ at time $t = 0$. This configuration is called the initial or reference configuration. We use upper case letters and the sup or superscript 0 to refer to the initial configuration. At time t, the particles p, part of the deformed body Ω, are located by the coordinates $\boldsymbol{x}$. We use lower case letters for variables in the current configuration.

The boundary of the volume is denoted Γ^0 in the reference configuration and Γ in the current configuration. This boundary is split into the part of the boundary where displacement boundary conditions are applied $\Gamma^0_{\bar{u}}$ and $\Gamma_{\bar{u}}$ and the part where traction boundary

conditions are prescribed $\Gamma^0_{\bar{t}}$ and $\Gamma_{\bar{t}}$, such that $\Gamma^0 = \Gamma^0_{\bar{u}} \cup \Gamma^0_{\bar{t}}$ and $\Gamma^0_{\bar{u}} \cap \Gamma^0_{\bar{t}} = 0$, and equivalently, in the current configuration: $\Gamma = \Gamma_{\bar{u}} \cup \Gamma_{\bar{t}}$ and $\Gamma_{\bar{u}} \cap \Gamma_{\bar{t}} = 0$. Moreover, the body may be subjected to body forces, noted $\rho_0 \bar{\boldsymbol{b}}$ in the reference configuration and $\rho \bar{\boldsymbol{b}}$ in the current configuration.

6.2.2 Motion

As introduced above, the position of a material point P is noted $\boldsymbol{X}$ in the reference configuration and $\boldsymbol{x}$ in the current configuration. There is a one to one mapping between the current and the reference configuration given by $\boldsymbol{x} = \phi(\boldsymbol{X}, t)$ and $\boldsymbol{X} = \phi(\boldsymbol{x}, t)^{-1}$.

We also define the displacement in the reference configuration:

$$\boldsymbol{u}(\boldsymbol{X}, t) = \boldsymbol{x}(\boldsymbol{X}, t) - \boldsymbol{X} \tag{6.1}$$

and in the current configuration:

$$\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{x}(t) - \boldsymbol{X}(\boldsymbol{x}) \tag{6.2}$$

The velocity of a particle is given by

$$\boldsymbol{v} = \dot{\boldsymbol{x}} = \frac{d\boldsymbol{x}}{dt} = \frac{\partial \boldsymbol{x}}{\partial t} \tag{6.3}$$

and its acceleration by

$$a_i(\boldsymbol{x}, t) = \dot{v}_i = \frac{d}{dt} v_i = \frac{\partial v_i(x_k, t)}{\partial t} + \frac{\partial v_i}{\partial x_k} \frac{\partial x_k}{\partial t} \tag{6.4}$$

6.2.3 Deformation gradient

6.2.3.1 Jacobian matrix or deformation gradient

Let us consider two neighbouring material points in the initial configuration, $\boldsymbol{X}$ and $\boldsymbol{X} = \boldsymbol{X} + d\boldsymbol{X}$, and follow their movement. After time t, their respective spatial positions are $\boldsymbol{x}$ and $\boldsymbol{x} = \boldsymbol{x} + d\boldsymbol{x}$. The *deformation gradient*, also called the *Jacobian matrix*, gives us information on how the infinitesimal vector $d\boldsymbol{X}$ deforms into $d\boldsymbol{x}$

$$d\boldsymbol{x} = \mathbf{F}\, d\boldsymbol{X}\,, \qquad dx_i = F_{iA}\, dX_A \tag{6.5}$$

Therefore the deformation gradient between the current and the reference configuration is defined by

$$\mathbf{F} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}\,, \qquad F_{iA} = \frac{\partial x_i}{\partial X_A} \tag{6.6}$$

6.2.3.2 Jacobian or volume change ratio

The determinant of $\mathbf{F}$, called the *Jacobian* and noted J, measures the change in volume between the current and reference configuration, around the considered material point. To underline this key property, the Jacobian is also called the *volume change ratio*.

$$d\upsilon = \det(\mathbf{F})\ dV = J\ dV \tag{6.7}$$

or,

$$\det(\mathbf{F}) = J = \frac{d\upsilon}{dV} \tag{6.8}$$

where V and υ are the volume in the reference and the current configuration respectively.

6.2.3.3 Volumetric-isochoric split of the deformation gradient

When dealing with incompressible or nearly incompressible materials it is often necessary to separate the volumetric from the isochoric (volume preserving, distortional) components of the deformation. The volumetric-isochoric split of the deformation gradient has been introduced by Flory [64]:

$$\mathbf{F} = \mathbf{F}^{\text{iso}}\ \mathbf{F}^{\text{vol}} \tag{6.9}$$

where the isochoric component of deformation gradient $\mathbf{F}^{\text{iso}}$ is defined by

$$\mathbf{F}^{\text{iso}} = (\det(\mathbf{F}))^{-\frac{1}{3}}\ \mathbf{F} = J^{-\frac{1}{3}}\ \mathbf{F} \tag{6.10}$$

and the volumetric component of the deformation gradient $\mathbf{F}^{\text{vol}}$ is defined by

$$\mathbf{F}^{\text{vol}} = (\det(\mathbf{F}))^{\frac{1}{3}}\ \mathbf{I} = J^{\frac{1}{3}}\ \mathbf{I} \tag{6.11}$$

so that, by construction,

$$\begin{aligned} \det\left(\mathbf{F}^{\text{iso}}\right) &= 1 \\ \det\left(\mathbf{F}^{\text{vol}}\right) &= J = \det(\mathbf{F}) \end{aligned} \tag{6.12}$$

One may verify that determinant of the isochoric deformation gradient $\mathbf{F}^{\text{iso}}$ equals one, meaning that, taking account of (6.8), the associated deformation is indeed volume preserving. The dilational part of the deformation is actually defined by the volumetric deformation gradient $\mathbf{F}^{\text{vol}}$, the determinant of which is the volume change ratio of the overall deformation.

6.2.4 Strain tensors

Several strain tensors are used in literature to measure the deformation of a body. The most common are introduced here.

The *right Cauchy-Green deformation tensor* **C** is given in terms of the deformation gradient **F** as

$$\mathbf{C} = \mathbf{F}^T \mathbf{F}\,, \qquad C_{AB} = F^T_{iA} F_{iB} \tag{6.13}$$

The *left Cauchy-Green deformation tensor* or *Finger tensor* **B** is given by

$$\mathbf{B} = \mathbf{F}\mathbf{F}^T\,, \qquad B_{ij} = F_{iA} F^T_{jA} \tag{6.14}$$

The *Lagrangian* or *Green strain tensor* or *Green-Lagrange strain tensor* is defined as

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{C} - \mathbf{I}\right)\,, \qquad E_{AB} = \frac{1}{2}\left(C_{AB} - \delta_{AB}\right) \tag{6.15}$$

where δ_{AB} is the Kronecker symbol.

The time derivative of the Green-Lagrange stain tensor (6.15) is called the *material strain rate tensor*:

$$\dot{\mathbf{E}} = \frac{1}{2}\,\dot{\mathbf{C}} = \frac{1}{2}\left(\dot{\mathbf{F}}^T\,\mathbf{F} + \mathbf{F}^T\,\dot{\mathbf{F}}\right)\,, \qquad \dot{E}_{AB} = \frac{1}{2}\,\dot{C}_{AB} = \frac{1}{2}\left(\dot{F}^T_{iA} F_{iB} + F^T_{iA}\dot{F}_{iB}\right) \tag{6.16}$$

The spatial counterpart of the material strain rate tensor is the *rate of deformation* strain tensor:

$$\mathbf{D} = \mathbf{F}^{-T}\,\dot{\mathbf{E}}\,\mathbf{F}^{-1}\,, \qquad D_{ij} = F^{-T}_{iA}\,\dot{E}_{AB}\,F^{-1}_{jB} \tag{6.17}$$

6.2.5 Stress tensors

The *Cauchy stress tensor* at point p of the body Ω is denoted $\boldsymbol{\sigma}$. The Cauchy stress tensor is symmetric, $\sigma_{ij} = \sigma_{ji}$, as this is the condition for the rotational equilibrium of the body.

Some alternative stress representations are used in this dissertation.

The *first Piola-Kirchhoff stress tensor* **P**, also called the nominal stress, the Piola stress tensor, the Piola-Kirchhoff 1 (PK1) stress tensor, the Boussinesq stress tensor or the Lagrange stress tensor, is defined as

$$\mathbf{P} = J\,\boldsymbol{\sigma}\,\mathbf{F}^{-T}\,, \qquad P_{iA} = J\,\sigma_{ij}\,F^{-T}_{jA} \tag{6.18}$$

The first Piola-Kirchhoff stress tensor **P** is generally non-symmetric as **F** is non-symmetric and $\boldsymbol{\sigma}$ is symmetric. It is a two-point tensor as it is related to the material and the current

configuration. The Piola stress tensor is work conjugate to the rate of the deformation gradient $\dot{\mathbf{F}}$, so that the internal work may be written as

$$\delta \mathcal{W}^{\text{int}} = \int_{\Omega_0} \mathbf{P} : \delta \dot{\mathbf{F}} \, d\Omega_0 = \int_{\Omega_0} P_{iA} \delta \dot{F}_{iA} \, d\Omega_0 \tag{6.19}$$

The *second Piola-Kirchhoff stress tensor* $\mathbf{S}$ is defined as

$$\begin{aligned} \mathbf{S} &= \mathbf{F}^{-1} \mathbf{P} \,, \qquad & S_{AB} &= F^{-1}_{Ai} P_{iB} \\ \mathbf{S} &= J \mathbf{F}^{-1} \boldsymbol{\sigma} \, \mathbf{F}^{-T} \,, \qquad & S_{AB} &= J \, F^{-1}_{iB} \, \sigma_{ij} \, F^{-T}_{jA} \end{aligned} \tag{6.20}$$

As opposed to the first Piola-Kirchhoff stress tensor (PK1), the second Piola-Kirchhoff stress (PK2) tensor is symmetric and completely related to the material configuration. It is work conjugate to the material strain rate tensor $\dot{\mathbf{E}}$ (6.16),

$$\delta \mathcal{W}^{\text{int}} = \int_{\Omega_0} \mathbf{S} : \delta \dot{\mathbf{E}} \, d\Omega_0 = \int_{\Omega_0} S_{AB} \delta \dot{E}_{AB} \, d\Omega_0 \tag{6.21}$$

The nominal stress or first Piola-Kirchhoff tensor $\mathbf{P}$ is an unsymmetric two-point tensor and as such is not completely related to the material configuration. Therefore, the second Piola-Kirchhoff tensor $\mathbf{S}$ is often preferred. The PK2 stress tensor is related to the nominal stress tensor $\mathbf{P}$ and the Cauchy stress tensor $\boldsymbol{\sigma}$ as follows

6.2.6 Volumetric-isochoric split of the stress

Some unlocking formulations of the next chapter separate the volumetric and the isochoric components of the stress tensor.

The volumetric-isochoric split of the Cauchy stress tensor is written as

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^{\text{iso}} + p\mathbf{I} \,, \qquad p = \frac{1}{3}\text{tr}\boldsymbol{\sigma} \tag{6.22}$$

where p may be viewed as the hydrostatic pressure.

From (6.18), the volumetric-deviatoric of the first Piola-Kirchhoff stress tensor reads

$$\mathbf{P} = \mathbf{P}^{\text{iso}} + p \, J \, \mathbf{F}^{-T} \,, \qquad \mathbf{P}^{\text{iso}} = J \, \boldsymbol{\sigma}^{\text{iso}} \, \mathbf{F}^{-T} \tag{6.23}$$

And, taking account of (6.20), we obtain for second Piola-Kirchhoff stress tensor

$$\mathbf{S} = \mathbf{S}^{\text{iso}} + p \, J \, \mathbf{C}^{-1} \,, \qquad \mathbf{S}^{\text{iso}} = J \, \mathbf{F}^{-1} \, \boldsymbol{\sigma}^{\text{iso}} \, \mathbf{F}^{-T} \tag{6.24}$$

6.3 Total and updated Lagrangian formulations

The meshes presented in this work are Lagrangian meshes, that is they move together with the material. Hence, boundaries remain coincident with element edges, as opposed to Eulerian or ALE meshes [24]. Also, quadrature points move with the material so that the constitutive laws are always evaluated at the same material points, which greatly simplifies the approach.

Two approaches are commonly used for the development of Lagrangian finite elements:

- *Total Lagrangian Formulation*: derivatives and integrals are taken with respect to the Lagrangian (material) coordinates $\boldsymbol{X}$.
- *Updated Lagrangian Formulation*: derivatives and integrals are taken with respect to the Eulerian (spatial) coordinates $\boldsymbol{x}$.

In the following sections, we recall the key equations of both formulations. However, even though different stress and strain tensors are typically used in these two formulations, the expressions may be transformed from one formulation to the other. This is obvious since the underlying mechanics of the two formulations are identical.

6.3.1 Total Lagrangian Formulation

In the total Lagrangian formulation, integrals are taken over the initial configuration, which plays the role of the reference configuration, and derivatives are taken with respect to material coordinates. Moreover, stresses are expressed in terms of the first Piola-Kirchhoff tensor $\mathbf{P}$ (6.18) and the deformation gradient $\mathbf{F}$ (6.9) is used as a strain measure.

6.3.1.1 Principle of Virtual Work

We define $\mathcal{U}$, the space of kinematically admissible displacements, that is, that satisfy the displacement constraints of the continuous problem. $\mathcal{U}_0$ is the space of kinematically admissible displacements with the functions vanishing where they are prescribed. Also, Ω_0 and Ω represent the initial (reference) and current domain occupied by the body and Γ^0 and Γ represent the boundary of the body in the reference and current configuration.

The principle of virtual work is stated as follows:

If $\mathbf{u} \in \mathcal{U}$, then if

$$\delta \mathcal{W} = \delta \mathcal{W}^{\text{int}} - \delta \mathcal{W}^{\text{ext}} + \delta \mathcal{W}^{\text{kin}} = 0 \quad \forall \delta \mathbf{u} \in \mathcal{U}_0 \tag{6.25}$$

then linear and angular momentum balance (Equations (6.26) and (6.27) hereunder), traction boundary conditions (6.28) and internal continuity equations (6.29) are satisfied[1]:

$$\nabla_0 \cdot \mathbf{P} + \rho_0 \bar{\mathbf{b}} = \rho_0 \dot{\mathbf{v}}, \qquad \frac{\partial P_{iA}}{\partial X_A} + \rho_0 \bar{b}_i = \rho_0 \dot{v}_i \quad \text{in} \quad \Omega_0 \tag{6.26}$$

$$\mathbf{P}\mathbf{F}^T = \mathbf{F}\mathbf{P}^T, \qquad P_{iA} F_{Aj} = F_{iA} P_{Aj} \quad \text{in} \quad \Omega_0 \tag{6.27}$$

$$\mathbf{P} \cdot \boldsymbol{n}^0 = \bar{\boldsymbol{t}}^0, \qquad P_{iA} n_A^0 = \bar{t}_i^0 \quad \text{on} \quad \Gamma^0_{\bar{t}_i} \tag{6.28}$$

$$[\![\mathbf{P} \cdot \boldsymbol{n}^0]\!] = 0, \qquad [\![P_{iA} n_A^0]\!] = 0 \quad \text{on} \quad \Gamma^0_{\text{int}} \tag{6.29}$$

In Equation (6.25), internal, external and kinetic virtual work are defined by

$$\delta \mathcal{W}^{\text{int}} = \int_{\Omega_0} \mathbf{P} : \delta \mathbf{F} \, d\Omega_0 = \int_{\Omega_0} P_{iA} \delta F_{iA} \, d\Omega_0 \tag{6.30}$$

$$\delta \mathcal{W}^{\text{ext}} = \int_{\Omega_0} \rho_0 \, \delta \boldsymbol{u} \cdot \bar{\boldsymbol{b}} \, d\Omega_0 + \int_{\Gamma^0_{\bar{t}}} \delta \boldsymbol{u} \cdot \bar{\boldsymbol{t}}^0 \, d\Gamma_0 = \int_{\Omega_0} \rho_0 \, \delta u_i \bar{b}_i \, d\Omega_0 + \int_{\Gamma^0_{\bar{t}}} \delta u_i \bar{t}_i^0 \, d\Gamma_0 \tag{6.31}$$

$$\delta \mathcal{W}^{\text{kin}} = \int_{\Omega_0} \rho_0 \, \delta \mathbf{u} \cdot \dot{\mathbf{v}} \, d\Omega_0 = \int_{\Omega_0} \rho_0 \, \delta u_i \dot{v}_i \, d\Omega_0 \tag{6.32}$$

6.3.1.2 Discrete equations

In this section, the finite element equations for the Total Lagrangian formulation are presented. These are obtained from the principle of virtual work by subdividing the initial domain Ω_0 into elements Ω_0^e. In this work, the nodes of the resulting mesh are denoted X_I with $I = 1$ to n_N. The finite element method approximates the motion by

$$\boldsymbol{x}(\boldsymbol{X}, t) = \boldsymbol{x}(t) \, \boldsymbol{N}_I(\boldsymbol{X}), \qquad x_i(\boldsymbol{X}, t) = x_{Ii}(t) \, N_I(\boldsymbol{X}) \tag{6.33}$$

the velocity by

$$v_i(\boldsymbol{X}, t) = \dot{x}_i(\boldsymbol{X}, t) = \dot{x}_{Ii}(t) \, N_I(\boldsymbol{X}) \tag{6.34}$$

and the acceleration by

$$a_i(\boldsymbol{X}, t) = \dot{v}_i(\boldsymbol{X}, t) = \ddot{x}_{Ii}(t) \, N_I(\boldsymbol{X}) \tag{6.35}$$

[1] $[\![f]\!]$ designates the jump in $f(X)$, $[\![f(X)]\!] = f(X + \epsilon) - f(X - \epsilon) \quad \text{for} \quad \epsilon \to 0$

We choose to interpolate the virtual displacement field $\delta\boldsymbol{u}$, appearing in (6.31) and (6.32), and the virtual velocity $\delta\boldsymbol{v}$ (used in the principle of virtual power hereafter) in the same way

$$\delta\boldsymbol{u}(\boldsymbol{X},t)=\delta\boldsymbol{u}_I(t)N_I(\boldsymbol{X}),\qquad \delta u_i(\boldsymbol{X},t)=\delta u_{Ii}(t)N_I(\boldsymbol{X}) \tag{6.36}$$

and

$$\delta\boldsymbol{v}(\boldsymbol{X},t)=\delta\boldsymbol{v}_I(t)N_I(\boldsymbol{X}),\qquad \delta v_i(\boldsymbol{X},t)=\delta v_{Ii}(t)N_I(\boldsymbol{X}) \tag{6.37}$$

Remarks

- In the above, the interpolation functions $N_I(\boldsymbol{X})$ depend on the material coordinates only whereas the nodal coordinates $x_{Ii}(t)$ and the virtual displacements $\delta\boldsymbol{u}_I(t)$ are functions of time only.
- The finite elements considered in this dissertation are isoparametric: position, displacements, velocities and accelerations are all interpolated in the same way. Therefore $N_I(\boldsymbol{X})$ will be alternately called *interpolation function* or *shape function*.
- The nodal unknowns are considered functions of time even in static, equilibrium problems. Indeed this parameter is needed in non-linear problems to be able to follow the evolution of the load. In many cases, t is simply a monotonically increasing parameter.

Replacing (6.33), (6.35) and (6.36) in the virtual work equation (6.25), taking account of (6.30),(6.31) and, (6.32), noting that $F_{iA}=\frac{\partial x_i}{\partial X_A}$ and finally remembering that the virtual virtual work equation must be true for all kinematically admissible virtual displacements $\delta\boldsymbol{u}$, we obtain the discretised equations of motion:

$$M_{ijIJ}\ddot{x}_{Jj}+f^{\text{int}}_{Ii}=f^{\text{ext}}_{Ii} \tag{6.38}$$

with, the internal nodal forces:

$$f^{\text{int}}_{Ii}=\int_{\Omega_0}P_{iA}\frac{\partial N_I}{\partial X_A}d\Omega_0=\int_{\Omega_0}P_{iA}G^0_{IA}d\Omega_0=\int_{\square}P_{iA}G^0_{IA}J^0_\xi d\square \tag{6.39}$$

the external nodal forces:

$$f^{\text{ext}}_{Ii}=\int_{\Omega_0}N_I\rho_0\bar{b}_i d\Omega_0+\int_{\Gamma^0_{t_i}}N_I\bar{t}^0_i d\Gamma_0=\int_{\square}N_I\rho_0\bar{b}_i J^0_\xi d\square \tag{6.40}$$

and the mass matrix:

$$M_{ijIJ} = \delta_{ij} \int_{\Omega_0} \rho_0 N_I N_J d\Omega_0 = \delta_{ij} \int_{\square} \rho_0 N_I N_J J^0_\xi d\square \tag{6.41}$$

In (6.39), we have defined the $\mathbf{G}^0$-matrix that contains the derivatives of the shape functions with respect to the material coordinates $G^0_{IA} = \partial N_I / \partial X_A$. The corresponding matrix in the current configuration, often called the **B**-matrix in literature, is designated by **G** and given by $G_{Ii} = \partial N_I / \partial x_i$.

Also, in above equations, integrals over the initial domain Ω_0 are transformed into integrals over the element's parent domain $\square$ by scaling the integrand with the determinant of the Jacobian of the transformation between the initial and the parent domain J^0_ξ.

6.3.2 Updated Lagrangian Formulation

The *Cauchy stress Lagrangian formulation* is most efficient for many applications. This formulation is equivalent to the Total Lagrangian formulation, but expressed in terms of the spatial coordinates, that is to say, with respect to the current configuration. In recent literature, the Cauchy stress Lagrangian formulation is called *Updated Lagrangian formulation*, even though originally the *Updated Lagrangian formulation* referred to a formulation in which the last known equilibrium configuration was taken as reference. In the Updated Lagrangian formulation, stresses are generally expressed in terms of the Cauchy stresses $\boldsymbol{\sigma}$ and the rate-of-deformation **D** is used as a measure of strain rate.

6.3.2.1 Principle of Virtual Power

In the framework of the Updated Lagrangian formulation, the weak form of the momentum equation, the traction boundary condition and the interior stress continuity condition is called the principle of virtual power.

The principle of virtual power is stated as:

If σ_{ij} is a smooth function of the displacements and the velocities and $v_i \in \mathcal{V}$, then if

$$\delta\mathcal{P} = \delta\mathcal{P}^{\text{int}} - \delta\mathcal{P}^{\text{ext}} + \delta\mathcal{P}^{\text{kin}} = 0 \quad \forall \delta v_i \in \mathcal{V}_0 \tag{6.42}$$

then momentum equation, traction boundary equations and jump condition are satisfied:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \bar{\boldsymbol{b}} = \rho \dot{\boldsymbol{v}} \ , \qquad \frac{\partial \sigma_{ji}}{\partial x_j} + \rho \bar{b}_i = \rho \dot{v}_i \quad \text{in} \quad \Omega \tag{6.43}$$

$$\boldsymbol{n}\cdot\boldsymbol{\sigma}=\bar{t}_i\,,\qquad n_j\sigma_{ji}=\bar{t}_i\quad\text{on}\quad\Gamma_{t_i}\tag{6.44}$$

$$[\![\boldsymbol{n}\cdot\boldsymbol{\sigma}]\!]=0\,,\qquad [\![n_j\sigma_{ji}]\!]=0\quad\text{on}\quad\Gamma_{\text{int}}\tag{6.45}$$

In Equation (6.42), the internal, external and kinetic virtual power are defined by

$$\delta\mathcal{P}^{\text{int}}=\int_\Omega\boldsymbol{\sigma}:\delta\mathbf{D}\,d\Omega=\int_\Omega\sigma_{ij}\delta D_{ij}\,d\Omega=\int_\Omega\sigma_{ij}\frac{\partial\left(\delta v_i\right)}{\partial x_j}\,d\Omega\tag{6.46}$$

$$\delta\mathcal{P}^{\text{ext}}=\int_\Omega\delta\mathbf{v}\cdot\rho\bar{\mathbf{b}}\,d\Omega+\int_{\Gamma_{\bar{t}}}\left(\delta\mathbf{v}\cdot\mathbf{e}_j\right)\left(\bar{\mathbf{t}}\cdot\mathbf{e}_j\right)\,d\Gamma=\int_\Omega\delta v_i\rho\bar{b}_i\,d\Omega+\int_{\Gamma_{t_j}}\delta v_j\bar{t}_j\,d\Gamma\tag{6.47}$$

$$\delta\mathcal{P}^{\text{kin}}=\int_\Omega\delta\mathbf{v}\cdot\rho\dot{\mathbf{v}}\,d\Omega=\int_\Omega\delta v_i\rho\dot{v}_i\,d\Omega\tag{6.48}$$

6.3.2.2 Discrete equations

Finite element discretisation of the principle of virtual power (6.42), with the help of (6.34) and (6.37), yields the discrete equations of motion:

$$M_{ijIJ}\dot{v}^{\text{int}}_{Jj}+f^{\text{int}}_{Ii}=f^{\text{ext}}_{Ii}\quad\text{for}\quad(I,i)\notin\Gamma_{v_i}\tag{6.49}$$

with the internal nodal forces:

$$f^{\text{int}}_{Ii}=\int_\Omega\frac{\partial N_I}{\partial x_j}\sigma_{ji}d\Omega=\int_\Omega G_{Ij}\sigma_{ji}d\Omega\tag{6.50}$$

the external nodal forces:

$$f^{\text{ext}}_{Ii}=\int_\Omega N_I\rho\bar{b}_i d\Omega+\int_{\Gamma_{t_i}}N_I\bar{t}_i d\Gamma\tag{6.51}$$

and the mass matrix:

$$M_{ijIJ}=\delta_{ij}\int_{\Omega_0}\rho_0 N_I N_J d\Omega_0=\delta_{ij}\int_\square\rho_0 N_I N_J J^0_\xi d\square\tag{6.52}$$

6.4 Consistent Linearisation and Tangent Stiffness Matrix

The principle of virtual work has been previously expressed in the Total Lagrangian formulation as (6.25):

$$\delta\mathcal{W}=\delta\mathcal{W}^{\text{int}}-\delta\mathcal{W}^{\text{ext}}+\delta\mathcal{W}^{\text{kin}}=0\tag{6.53}$$

In non-linear finite element analysis the above equation is non-linear, both the internal and the external works being non-linear. A linearised model of the virtual work equation (6.53) around a state k defined by the nodal positions $\boldsymbol{x}^k$, here-under called current or trial solution, is obtained by computing the Taylor expansion of the virtual work and dropping the higher order terms. The virtual work equation is linearised in the direction of an increment $\boldsymbol{\eta}$ as,

$$\delta\mathcal{W}\left(\boldsymbol{x}^{k+1},\delta\boldsymbol{u}\right)\approx\delta\mathcal{W}^{\text{LIN}}\left(\boldsymbol{x}^{k+1},\delta\boldsymbol{u}\right)=\delta\mathcal{W}\left(\boldsymbol{x}^{k},\delta\boldsymbol{u}\right)+\text{D}\delta\mathcal{W}\left(\boldsymbol{x}^{k},\delta\boldsymbol{u}\right)\left[\boldsymbol{\eta}\right]=0 \quad (6.54)$$

The second term of (6.54), $\text{D}\delta\mathcal{W}\left(\boldsymbol{x}^k,\delta\boldsymbol{u}\right)$, represents the directional derivative of the virtual work. The directional derivative of $\delta\mathcal{W}$ at $\boldsymbol{x}^k$ in the direction $\boldsymbol{\eta}$ is computed by introducing a parameter ϵ and computing the first-order Taylor series expansion of $\delta\mathcal{W}\left(\boldsymbol{x}^k+\epsilon\boldsymbol{\eta},\delta\boldsymbol{u}\right)$ around $\epsilon=0$ (please refer to Appendix B.1 for more details on directional derivatives):

$$\text{D}\delta\mathcal{W}\left(\boldsymbol{x}^k,\delta\boldsymbol{u}\right)\left[\boldsymbol{\eta}\right]=\left.\frac{d}{d\epsilon}\right|_{\epsilon=0}\delta\mathcal{W}\left(\boldsymbol{x}^k+\epsilon\boldsymbol{\eta},\delta\boldsymbol{u}\right) \quad (6.55)$$

Let us try to understand this equation. The principle of virtual work (6.25) and (6.53) is constructed by associating an arbitrary kinematically admissible virtual displacement $\delta\boldsymbol{u}$ to each particle of the domain. At a trial solution position $\boldsymbol{x}^k$, the virtual work $\delta\mathcal{W}$ will have some value, probably not equal to zero as required for equilibrium. The directional derivative of the virtual work $\text{D}\delta\mathcal{W}\left[\boldsymbol{\eta}\right]$ is simply the change in $\delta\mathcal{W}$ due to $\boldsymbol{x}^k$ changing to $\boldsymbol{x}^{k+1}=\boldsymbol{x}^k+\boldsymbol{\eta}$. Note that the virtual displacement $\delta\boldsymbol{u}$ is not allowed to change during this incremental change. The directional derivative will help us to adjust the current configuration, defined by $\boldsymbol{x}^k$, in order to bring the internal forces into equilibrium with the external forces, via (6.54) and using a Newton–Raphson procedure. In other words, the directional derivative of the virtual work equation will be the source of the tangent matrix $\mathbf{K}_T$.

In order to simplify the notations, the expression for $\mathbf{K}_T$ will be derived in the quasi-static case. In this case, the kinematic energy term vanishes from the virtual work equation

$$\delta\mathcal{W}^{\text{kin}}=0 \quad (6.56)$$

Let us also assume for simplicity that the loading is independent of the deformation, i.e. that the forces are conservative, so that the linearisation of $\mathcal{W}^{\text{ext}}$ vanishes. This is generally the case for the loading due to body forces but not for the surface forces as they depend on the normal to the current boundary surface. But, because we are interested in the expression of the internal tangent stiffness matrix $\mathbf{K}_T^{\text{int}}$ only, this hypothesis of conservative loading will not influence our results, that is to say, the expression found for $\mathbf{K}_T^{\text{int}}$ will be

valid even for non-conservative loads.

$$\left.\frac{d\delta\mathcal{W}^{\text{ext}}}{d\epsilon}\right|_{\epsilon=0} = 0 \tag{6.57}$$

With the two hypothesis (6.56) and (6.57), linearisation of the virtual work (6.54) gives the following equilibrium condition

$$\delta\mathcal{W}^{\text{LIN}}\left(\boldsymbol{x}^{k+1}, \delta\boldsymbol{u}\right) = \delta\mathcal{W}\left(\boldsymbol{x}^{k}, \delta\boldsymbol{u}\right) + \mathrm{D}\delta\mathcal{W}^{\text{int}}\left(\boldsymbol{x}^{k}, \delta\boldsymbol{u}\right)\left[\boldsymbol{\eta}\right] = 0 \tag{6.58}$$

with $\delta\mathcal{W}$ still being given by the principle of virtual work (6.25) and (6.53):

$$\delta\mathcal{W} = \delta\mathcal{W}^{\text{int}} + \delta\mathcal{W}^{\text{ext}} + \delta\mathcal{W}^{\text{kin}} \tag{6.59}$$

In the next sections, we derive the expression for the internal tangent stiffness matrix $\mathbf{K}_T^{\text{int}}$, by first performing a Newton-Raphson linearisation of the internal work equation (Section 6.4.1), discretising the obtained linearised equations (Section 6.4.2) and then identifying the tangent stiffness $\mathbf{K}^{\text{int}}$ (Section 6.4.3). The latter is used in the Newton-Raphson iteration algorithm to update the nodal displacements in order to enforce equilibrium of the structure.

6.4.1 Linearisation of the Virtual Work

6.4.1.1 Virtual work expressed in terms of the first Piola-Kirchhoff stress

In the neighbourhood of the current point $\boldsymbol{x}^k$, the internal virtual work expressed in terms of the nominal stress is (6.30):

$$\delta\mathcal{W}^{\text{int}}\left(\boldsymbol{x}^{k} + \epsilon\boldsymbol{\eta}, \delta\boldsymbol{u}\right) = \int_{\Omega_0} \mathbf{P}(\mathbf{F}(\epsilon)) : \delta\mathbf{F}\, d\Omega_0 = \int_{\Omega_0} P_{iA}(\mathbf{F}(\epsilon))\, \delta F_{iA}\, d\Omega_0 \tag{6.60}$$

The virtual deformation gradient $\delta\mathbf{F}$ is, by definition, the directional derivative of the deformation gradient in the direction of a virtual displacement $\delta\boldsymbol{u}$ (which is assumed to be constant during a Newton-Raphson iteration $\boldsymbol{x}^k \rightarrow \boldsymbol{x}^{k+1}$, see Section (6.53)). Also, the directional derivative of the deformation gradient $D\mathbf{F}\left[\delta\boldsymbol{u}\right]$ is computed in Appendix (B.4). Hence, taking account of (B.11), we obtain

$$\delta\mathbf{F} = D\mathbf{F}\left[\delta\boldsymbol{u}\right] = \nabla_0\delta\boldsymbol{u} \tag{6.61}$$

Consequently, (6.60) becomes

$$\delta\mathcal{W}^{\text{int}}\left(\boldsymbol{x}^{k} + \epsilon\boldsymbol{\eta}, \delta\boldsymbol{u}\right) = \int_{\Omega_0} \mathbf{P}(\mathbf{F}(\epsilon)) : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 = \int_{\Omega_0} P_{iA}(\mathbf{F}(\epsilon))\, \frac{\partial\, \delta u_i}{\partial X_A}\, d\Omega_0 \tag{6.62}$$

Taking the directional derivative, given by (B.3), of the virtual internal work (6.62) gives:

$$D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] = \left.\frac{d\delta\mathcal{W}^{\text{int}}}{d\epsilon}\right|_{\epsilon=0} = \int_{\Omega_0}\left[\left.\frac{d\mathbf{P}}{d\mathbf{F}}\right|_{\mathbf{F}(\boldsymbol{x}^k)} : \left.\frac{d\mathbf{F}(\epsilon)}{d\epsilon}\right|_{\epsilon=0}\right] : \nabla_0\delta\boldsymbol{u}\, d\Omega_0$$
$$= \int_{\Omega_0}\left.\frac{dP_{iA}}{dF_{jB}}\right|_{\mathbf{F}(\boldsymbol{x}^k)}\left.\frac{dF_{jB}(\epsilon)}{d\epsilon}\right|_{\epsilon=0}\frac{\partial\delta u_i}{\partial X_A}\, d\Omega_0 \tag{6.63}$$

We define the tangent modulus

$$\mathbf{A} = \left.\frac{d\mathbf{P}}{d\mathbf{F}}\right|_{\mathbf{F}^\star}, \qquad A_{iAjB} = \frac{\partial P_{iA}}{\partial F_{jB}} \tag{6.64}$$

Also, as detailed in Appendix B.4, the linearisation of the deformation gradient $D\mathbf{F}[\boldsymbol{\eta}]$ gives

$$D\mathbf{F}[\boldsymbol{\eta}] = \left.\frac{d\mathbf{F}(\epsilon)}{d\epsilon}\right|_{\epsilon=0} = \nabla_0\boldsymbol{\eta}\,, \qquad DF_{iA}[\eta] = \left.\frac{dF_{iA}(\epsilon)}{d\epsilon}\right|_{\epsilon=0} = \frac{\partial\eta_i}{\partial X_A} \tag{6.65}$$

Inserting (6.64) and (6.65) into (6.63),

$$D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] = \int_{\Omega_0}\left(\mathbf{A} : \nabla_0\boldsymbol{\eta}\right) : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 = \int_{\Omega_0} A_{iAjB}\frac{\partial\eta_j}{\partial X_B}\frac{\partial\delta u_i}{\partial X_A}\, d\Omega_0 \tag{6.66}$$

Finally the linearised virtual work equation in the reference configuration (6.58) becomes

$$\int_{\Omega_0}\left(\mathbf{A} : \nabla_0\boldsymbol{\eta}\right) : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 = -\int_{\Omega_0}\mathbf{P} : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 + \rho_0\int_{\Omega_0}\bar{\boldsymbol{b}}\cdot\delta\boldsymbol{u}\, d\Omega_0 + \int_{\Gamma^0_{\bar{\boldsymbol{t}}}}\bar{\boldsymbol{t}}^0\cdot\delta\boldsymbol{u}\, d\Gamma_0$$
$$\int_{\Omega_0} A_{iAjB}\frac{\partial\eta_j}{\partial X_B}\frac{\partial\delta u_i}{\partial X_A}\, d\Omega_0 = -\int_{\Omega_0} P_{iA}\frac{\partial\delta u_i}{\partial X_A}\, d\Omega_0 + \rho_0\int_{\Omega_0}\bar{b}_i\,\delta u_i\, d\Omega_0 + \int_{\Gamma^0_{\bar{t}_i}}\bar{t}^0_i\,\delta u_i\, d\Gamma_0 \tag{6.67}$$

6.4.1.2 Virtual work expressed in terms of the PK2 stresses

Recall from (6.20) that the internal virtual work can be expressed in a Lagrangian form as,

$$\delta\mathcal{W}^{\text{int}}\left(\boldsymbol{x}^k + \epsilon\boldsymbol{\eta}, \delta\boldsymbol{u}\right) = \int_{\Omega_0}\mathbf{S} : \delta\mathbf{E}\, d\Omega_0 = \int_{\Omega_0} S_{AB}\,\delta E_{AB}\, d\Omega_0 \tag{6.68}$$

Taking the directional derivative of this expression, at $\boldsymbol{x}^k$ and in the direction of $\boldsymbol{\eta}$, and using the product rule for directional derivatives gives

$$\begin{aligned} D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] = \left.\frac{d\delta\mathcal{W}^{\text{int}}}{d\epsilon}\right|_{\epsilon=0} &= \int_{\Omega_0} \frac{d}{d\epsilon}\left(\mathbf{S} : \delta\mathbf{E}\right)|_{\epsilon=0}\, d\Omega_0 \\ &= \int_{\Omega_0} \left.\frac{d\mathbf{S}}{d\epsilon}\right|_{\epsilon=0} : \delta\mathbf{E} + \left.\frac{d\delta\mathbf{E}}{d\epsilon}\right|_{\epsilon=0} : \mathbf{S}\, d\Omega_0 \\ &= \int_{\Omega_0} \left(\frac{d\mathbf{S}}{d\mathbf{E}} : \left.\frac{d\mathbf{E}}{d\epsilon}\right|_{\epsilon=0}\right) : \delta\mathbf{E} + \left.\frac{d\delta\mathbf{E}}{d\epsilon}\right|_{\epsilon=0} : \mathbf{S}\, d\Omega_0 \\ &= \int_{\Omega_0} \left(\mathbf{C} : D\mathbf{E}\left[\boldsymbol{\eta}\right]\right) : \delta\mathbf{E} + D\delta\mathbf{E}\left[\boldsymbol{\eta}\right] : \mathbf{S}\, d\Omega_0 \end{aligned} \tag{6.69}$$

where we have defined the material elasticity tensor $\mathbf{C}$

$$\mathbf{C} = \frac{d\mathbf{S}}{d\mathbf{E}}, \qquad C_{ABCD} = \frac{dS_{AB}}{dE^{GL}_{CD}} \tag{6.70}$$

It is important to distinguish the Green Lagrange strain $\mathbf{E}$ and the virtual Green Lagrange strain $\delta\mathbf{E}$ in (6.69):

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^T\mathbf{F} - \mathbf{I}\right) \tag{6.71}$$

$$\delta\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^T\delta\mathbf{F} + \delta\mathbf{F}^T\mathbf{F}\right) \quad \text{with} \quad \delta\mathbf{F} = \frac{\partial\,\delta\boldsymbol{u}}{\partial\boldsymbol{X}} = \nabla_0\delta\boldsymbol{u} \tag{6.72}$$

Taking account of (6.72) and (B.11), the directional derivative of the virtual Green Lagrange strain gives

$$\begin{aligned} D\delta\mathbf{E}\left[\boldsymbol{\eta}\right] &= \frac{1}{2}\left[\left(D\mathbf{F}\left[\boldsymbol{\eta}\right]\right)^T\nabla_0\delta\boldsymbol{u} + \left(\nabla_0\delta\boldsymbol{u}\right)^T D\mathbf{F}\left[\boldsymbol{\eta}\right]\right] \\ &= \frac{1}{2}\left[\left(\nabla_0\boldsymbol{\eta}\right)^T\nabla_0\delta\boldsymbol{u} + \left(\nabla_0\delta\boldsymbol{u}\right)^T\nabla_0\boldsymbol{\eta}\right] \end{aligned} \tag{6.73}$$

Substituting (6.73) into (6.69) and noting the symmetry of $\mathbf{S}$ gives the linearised principle of virtual work in the reference configuration as,

$$\begin{aligned} D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] &= \int_{\Omega_0} \left(\mathbf{C} : D\mathbf{E}\left[\boldsymbol{\eta}\right]\right) : \delta\mathbf{E} + \left[\left(\nabla_0\delta\boldsymbol{u}\right)^T\nabla_0\boldsymbol{\eta}\right] : \mathbf{S}\, d\Omega_0 \\ &= \int_{\Omega_0} C_{ABCD}\, DE^{GL}_{CD}\left[\boldsymbol{\eta}\right]\, \delta E^{GL}_{AB} + \frac{\partial\,\delta u_i}{\partial X_A}\frac{\partial\,\eta_i}{\partial X_B}\, S_{AB}\, d\Omega_0 \end{aligned} \tag{6.74}$$

6.4.1.3 Virtual work expressed in terms of the Cauchy stresses

To derive the spatial version of the linearised virtual work equation, we will start from the previous result of the linear virtual work in terms of the nominal stress (6.66) and recall that material and spatial virtual work functionals are equivalent. Hence the spatial form is obtained using the standard relations

$$\nabla \boldsymbol{q} = \nabla_0 \boldsymbol{q}\ \mathbf{F}^{-1} \quad \text{and} \quad \int_\Omega q(\boldsymbol{x}) d\Omega = \int_{\Omega_0} J(\boldsymbol{X}) q(\boldsymbol{X}) d\Omega_0 \tag{6.75}$$

With these relations, the linearised internal virtual (6.66) work becomes,

$$\begin{aligned} D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] &= \int_{\Omega_0} \left(\mathbf{A} : \nabla_0 \boldsymbol{\eta}\right) : \nabla_0 \delta\boldsymbol{u}\ d\Omega_0 \\ &= \int_{\Omega_0} A_{iAjB} \frac{\partial \eta_j}{\partial X_B} \frac{\partial \delta u_i}{\partial X_A}\ d\Omega_0 \\ &= \int_\Omega \frac{1}{J} A_{iAjB} \left(\frac{\partial \eta_j}{\partial x_l} F_{lB}\right) \left(\frac{\partial \delta u_i}{\partial x_k} F_{kA}\right)\ d\Omega \\ &= \int_\Omega \frac{\partial \eta_j}{\partial x_l} \left(\frac{1}{J} F_{lB} A_{iAjB} F_{kA}\right) \frac{\partial \delta u_i}{\partial x_k}\ d\Omega \\ &= \int_\Omega \frac{\partial \eta_j}{\partial x_l}\ a_{ijkl}\ \frac{\partial \delta u_i}{\partial x_k}\ d\Omega \\ &= \int_\Omega \left(\mathbf{a} : \nabla \boldsymbol{\eta}\right) : \nabla \delta\boldsymbol{u}\ d\Omega \end{aligned} \tag{6.76}$$

where we have defined the spatial tangent modulus

$$a_{ijkl} = \frac{1}{J} F_{kB} A_{iAjB} F_{lA} \tag{6.77}$$

In the end, the linearised principle of virtual work in the spatial configuration reads

$$\int_\Omega \frac{\partial \eta_j}{\partial x_l}\ a_{ijkl}\ \frac{\partial \delta u_i}{\partial x_k}\ d\Omega = -\int_\Omega \frac{\partial \delta u_i}{\partial x_j}\ \sigma_{ij}\ d\Omega + \int_\Omega \rho\ \bar{b}_i \delta u_i\ d\Omega + \int_{\Gamma_{t_i}} \bar{t}_i\ \delta u_i\ d\Gamma \tag{6.78}$$

6.4.2 Discretisation of the linearised equations

Linearised expressions for the internal work were obtained in the previous sections. Here, we discretise the obtained expressions using finite element discretisation. This will help us to find an expression for the internal stiffness matrix in the next chapter, Chapter 6.4.3.

6.4.2.1 Virtual work expressed in terms of the first Piola-Kirchhoff stress

Recall the linearised principle of virtual work in the reference configuration (6.67):

$$\int_{\Omega_0} A_{iAjB} \frac{\partial \eta_j}{\partial X_B} \frac{\partial \delta u_i}{\partial X_A} \, d\Omega_0 = -\int_{\Omega_0} P_{iA} \frac{\partial \delta u_i}{\partial X_A} \, d\Omega_0 + \rho_0 \int_{\Omega_0} \bar{b}_i \, \delta u_i \, d\Omega_0 + \int_{\Gamma^0_{t_i}} \bar{t}^0_i \, \delta u_i \, d\Gamma_0 \quad (6.79)$$

Finite element semi-discretisation[2] of this equation is performed by discretising the domain and its boundary into elements $\Omega_0 \to^h \Omega_0, \Gamma^0_t \to^h \Gamma^0_t$ and interpolating the displacement fields using shape functions $\eta_J(\boldsymbol{X}, t) = \eta_{Jj}(t) N_J(\boldsymbol{X})$ and $\delta u_i(\boldsymbol{X}, t) = \delta u_{Ii}(t) N_I(\boldsymbol{X})$ (6.36):

$$\begin{aligned} &\int_{{}^h\Omega_0} A_{iAjB} \frac{\partial N_J}{\partial X_B} \eta_{Jj} \frac{\partial N_I}{\partial X_A} \delta u_{Ii} \, d\Omega_0 \\ &= -\int_{{}^h\Omega_0} P_{iA} \frac{\partial N_I}{\partial X_A} \delta u_{Ii} \, d\Omega_0 + \rho_0 \int_{{}^h\Omega_0} \bar{b}_i \delta u_{Ii} N_I \, d\Omega_0 + \int_{{}^h\Gamma^0_{\bar{t}}} \bar{t}^0_i \delta u_{Ii} N_I \, d\Gamma_0 \end{aligned} \quad (6.80)$$

re-arranging the terms,

$$\begin{aligned} &\left\{ \int_{{}^h\Omega_0} \frac{\partial N_I}{\partial X_A} A_{iAjB} \frac{\partial N_J}{\partial X_B} \, d\Omega_0 \right\} \eta_{Jj} \delta u_{Ii} \\ &= -\left\{ \int_{{}^h\Omega_0} P_{iA} \frac{\partial N_I}{\partial X_A} \, d\Omega_0 \right\} \delta u_{Ii} + \left\{ \rho_0 \int_{{}^h\Omega_0} \bar{b}_i N_I \, d\Omega_0 \right\} \delta u_{Ii} + \left\{ \int_{{}^h\Gamma^0_{\bar{t}}} \bar{t}^0_i N_I \, d\Gamma_0 \right\} \delta u_{Ii} \end{aligned} \quad (6.81)$$

Because this equation must be true for all δu_{Ii} it simplifies into

$$\left\{ \int_{{}^h\Omega_0} \frac{\partial N_I}{\partial X_A} A_{iAjB} \frac{\partial N_J}{\partial X_B} \, d\Omega_0 \right\} \eta_{Jj} = -\int_{{}^h\Omega_0} P_{iA} \frac{\partial N_I}{\partial X_A} \, d\Omega_0 + \rho_0 \int_{{}^h\Omega_0} \bar{b}_i N_I \, d\Omega_0 + \int_{{}^h\Gamma^0_{\bar{t}}} \bar{t}^0_i N_I \, d\Gamma_0 \quad (6.82)$$

In matrix notation, this gives

$$\left\{ \int_{{}^h\Omega_0} \mathbf{G}_0^T \mathbf{A} \mathbf{G}_0 \, d\Omega_0 \right\} \boldsymbol{\eta} = -\int_{{}^h\Omega_0} \mathbf{G}_0^T \mathbf{P} \, d\Omega_0 - \rho_0 \int_{{}^h\Omega_0} \mathbf{N}^T \bar{\boldsymbol{b}} \, d\Omega_0 - \int_{{}^h\Gamma^0_{\bar{t}}} \mathbf{N}^T \bar{\boldsymbol{t}}^0 \, d\Gamma_0 \quad (6.83)$$

In the end, the linearised virtual work equation (6.58) and (6.83) is satisfied if and only if (6.83) is satisfied.

[2]The term *semi-discretisation* is used because the equation is discretised in space but not in time

6.4.2.2 Virtual work expressed in terms of the Cauchy stresses

In the current configuration, discretisation of the linearised principle of virtual work (6.78) using the same methodology gives

$$\left\{ \int_{^h\Omega} \frac{\partial N_I}{\partial x_j} \, a_{ijkl} \, \frac{\partial N_J}{\partial x_l} \, d\Omega \right\} \eta_{Jk} = \int_{\Omega} G_{Ij} \, \sigma_{ji} \, d\Omega + \int_{\Omega} N_I \, \rho \, \bar{b}_i \, d\Omega + \int_{\Gamma_{t_i}} N_I \bar{t}_i \, d\Gamma \tag{6.84}$$

and, in matrix notation,

$$\left\{ \int_{^h\Omega} \mathbf{G}^T \, \mathbf{a} \, \mathbf{G} \, d\Omega \right\} \boldsymbol{\eta} = \int_{\Omega} \mathbf{G}^T \, \boldsymbol{\sigma} \, d\Omega + \rho \int_{\Omega} \mathbf{N}^T \, \bar{\mathbf{b}} \, d\Omega + \int_{\Gamma_{\mathbf{t}}} \mathbf{N}^T \, \bar{\mathbf{t}} \, d\Gamma \tag{6.85}$$

6.4.3 Tangent Stiffness matrix and Newton-Raphson solution procedure

Equations (6.83) and (6.85) may be expressed in the form

$$\mathbf{K}_T^{\text{int}} \boldsymbol{\eta} = -\boldsymbol{r} \tag{6.86}$$

where we have defined *global tangent stiffness matrix* or *system Jacobian matrix* in the reference configuration

$$\mathbf{K}_T^{\text{int}} = \int_{^h\Omega_0} \mathbf{G}_0^T \, \mathbf{A} \, \mathbf{G}_0 \, d\Omega_0 \tag{6.87}$$

and in the current configuration

$$\mathbf{K}_T^{\text{int}} = \int_{^h\Omega} \mathbf{G}^T \, \mathbf{a} \, \mathbf{G} \, d\Omega_0 \tag{6.88}$$

as well as the *residual* or *out of balance force* vector in the reference configuration

$$\boldsymbol{r} = \underbrace{\int_{^h\Omega_0} \mathbf{G}_0^T \, \mathbf{P} \, d\Omega_0}_{f^{\text{int}}} \underbrace{- \rho_0 \int_{^h\Omega_0} \mathbf{N}^T \, \bar{\boldsymbol{b}} \, d\Omega_0 - \int_{^h\Gamma_{\bar{t}}^0} \mathbf{N}^T \, \bar{\boldsymbol{t}}^0 \, d\Gamma_0}_{f^{\text{ext}}} \tag{6.89}$$

and in the current configuration

$$\boldsymbol{r} = \underbrace{\int_{^h\Omega} \mathbf{G}^T \, \boldsymbol{\sigma} \, d\Omega}_{f^{\text{int}}} \underbrace{- \rho \int_{^h\Omega} \mathbf{N}^T \, \bar{\boldsymbol{b}} \, d\Omega - \int_{^h\Gamma_{\bar{t}}} \mathbf{N}^T \, \bar{\boldsymbol{t}}^0 \, d\Gamma}_{f^{\text{ext}}} \tag{6.90}$$

The tangent stiffness matrix $\mathbf{K}_T$ represents the change in the internal forces due to a change in the nodal positions from $\boldsymbol{x}^k$ to $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \boldsymbol{\eta}$. In other words, the tangent stiffness matrix

$\mathbf{K}_T$ gives the modification $\boldsymbol{\eta} = \Delta\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{k+1} - \boldsymbol{x}^k$ that needs to be applied to the nodal positions in order to achieve equilibrium. This Newton-Raphson iterative procedure can be expressed as

$$\mathbf{K}_T \Delta\boldsymbol{x}^{k+1} = -\boldsymbol{r}^k \quad \text{with} \quad \Delta\boldsymbol{x}^{k+1} = \boldsymbol{x}^{k+1} - \boldsymbol{x}^k \tag{6.91}$$

Most often, in order to facilitate convergence of the solution, the external load is applied to the system in a series of increments.

$$\boldsymbol{f} = \sum_{n=1}^{N} \Delta\boldsymbol{f}_n \quad \text{with} \quad \Delta\boldsymbol{f}_n = \boldsymbol{f}_n - \boldsymbol{f}_{n-1} \quad \text{and} \quad \boldsymbol{f}_n = \boldsymbol{f}_n^{\text{int}} - \boldsymbol{f}_n^{\text{ext}} \tag{6.92}$$

The Newton-Raphson solution procedure (6.91) is then applied at each time step n. The more the number of increments, the easier it is to find a converted solution $\boldsymbol{r}^{(k)} = 0$ for a particular time step n.

In the context of the finite element method, this global matrix (6.87) is obtained by assembly of the element stiffness matrices:

$$\mathbf{K}_T = \mathop{\mathcal{A}}_{e=1}^{n_{\text{elem}}} \mathbf{K}_T^e \tag{6.93}$$

where the element stiffness matrices are defined by

$$\mathbf{K}_T^e = \int_{^h\Omega^e} (\mathbf{G}^e)^T \boldsymbol{a} \mathbf{G}^e \, d\Omega \qquad \text{or} \qquad \mathbf{K}_T^e = \int_{^h\Omega_0^e} \left(\mathbf{G}_0^e\right)^T \mathbf{A}^e \mathbf{G}_0^e \, d\Omega_0 \tag{6.94}$$

6.5 Time integration

In the previous sections, an expression for the internal tangent stiffness was found in both reference and current configurations. A quasi-static problem was considered to simplify the notations, with no restrictions to the validity of the final expressions obtained for $\mathbf{K}^{\text{int}}$ (6.87) and (6.88). In this section, explicit and implicit time integration procedures are reviewed. Obviously, kinematic terms will now be taken into account and added in the equations.

6.5.1 Explicit time integration

The semi-discretised equations of motion (discretised in space but not yet in time) are

$$\mathbf{M}\boldsymbol{a}_n = \boldsymbol{f}_n = \boldsymbol{f}^{\text{ext}}(\boldsymbol{x}_n, t_n) - \boldsymbol{f}^{\text{int}}(\boldsymbol{x}_n, t_n) \tag{6.95}$$

subject to displacement boundary conditions

$$g_I(\mathbf{x}_n) = 0, \quad I = 1, \ldots, n_c \tag{6.96}$$

The central difference method is typically used to discretise (6.95) in time

$$\boldsymbol{v}_{n+1/2} = \boldsymbol{v}_{n-1/2} + \Delta t_n \mathbf{M}^{-1} \boldsymbol{a}_n \tag{6.97}$$

$$= \boldsymbol{v}_{n-1/2} + \Delta t_n \mathbf{M}^{-1} \left(\boldsymbol{f}^{\text{ext}}(\boldsymbol{x}_n, t_n) - \boldsymbol{f}^{\text{int}}(\boldsymbol{x}_n, t_n) \right) \tag{6.98}$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \Delta t_{n+1/2} \boldsymbol{v}_{n+1/2} \tag{6.99}$$

This update of the nodal velocities and displacements can be performed without solving any equations provided that the mass matrix $\mathbf{M}$ is diagonal. For this reason the lumped mass matrix is almost always used in explicit time integration. In that case, the equations of motions (6.95) may be written at each node I

$$m_I \, \boldsymbol{a}_I = \boldsymbol{f}_I = \boldsymbol{f}_I^{\text{ext}} - \boldsymbol{f}_I^{\text{int}} \tag{6.100}$$

where the mass m_I represents the assembled lumped mass at node I which is typically formed by adding the contributions of the elements $e = 1, ..., n_{e,I}$ surrounding node I

$$m_I = \sum_{e=1}^{n_I} m_I^e \tag{6.101}$$

Each elemental contribution of the nodal mass is obtained by integrating the shape function corresponding to node I, N_I over the mass of the element.

$$m_I^e = \int_{V^e} \rho_0 N_I \, dV \tag{6.102}$$

6.5.2 Implicit time integration

The semi-discrete momentum equations, at a state defined by the nodal positions $\boldsymbol{x}_{n+1}$, are expressed as (6.95):

$$\mathbf{M} \boldsymbol{a}_{n+1} + \boldsymbol{f}^{\text{int}} \left(\boldsymbol{x}_{n+1} \right) - \boldsymbol{f}^{\text{ext}} \left(\boldsymbol{x}_{n+1} \right) = \mathbf{0} \tag{6.103}$$

Discretising this equation using Chung-Hulbert generalised-α time integration method [41], we obtain a set of non-linear algebraic equations in the nodal positions

$$\left(1 - \alpha_M\right) \mathbf{M} \ddot{\boldsymbol{x}}_{n+1} + \alpha_M \mathbf{M} \ddot{\boldsymbol{x}}_n + \left(1 - \alpha_F\right) \left(\boldsymbol{f}_{n+1}^{\text{int}} - \boldsymbol{f}_{n+1}^{\text{ext}} \right) + \alpha_F \left(\boldsymbol{f}_n^{\text{int}} - \boldsymbol{f}_n^{\text{ext}} \right) = \mathbf{0} \tag{6.104}$$

For particular choices of the parameters α_M and α_F, other well-knowm time integration procedures are recovered:

- $\alpha_M = \alpha_F = 0$ leads to Newmark time integration procedure [131]
- $\alpha_M = 0$ gives Hilber-Hughes-Taylor time integration procedure [81]

- $\alpha_F = 0$ gives Wood-Bossak-Zienkiewicz time integration procedure [185]

This set of non-linear equations is usually solved using the Newton-Raphson method. The out-of-balance forces at time integration step (k) and configuration $n+1$ are defined by

$$\boldsymbol{r}(\boldsymbol{x}_{n+1}^{(k)}) = \left(1-\alpha_M\right)\mathbf{M}\ddot{\boldsymbol{x}}_{n+1}^{(k)} + \alpha_M\mathbf{M}\ddot{\boldsymbol{x}}_n + \left(1-\alpha_F\right)\left(\boldsymbol{f}_{n+1}^{\text{int},(k)} - \boldsymbol{f}_{n+1}^{\text{ext},(k)}\right) + \alpha_F\left(\boldsymbol{f}_n^{\text{int}} - \boldsymbol{f}_n^{\text{ext}}\right) = \mathbf{0} \tag{6.105}$$

From the nodal position vector $\boldsymbol{x}^{(k)}$, defined at iteration k, the Newton correction to the nodal positions $\delta\boldsymbol{x}$ is obtained by (6.91)

$$\mathbf{K}_T\Delta\boldsymbol{x}^{(k)} = -\boldsymbol{r}^{(k-1)} \quad \text{with} \quad \Delta\boldsymbol{x}^{(k)} = \left(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\right) \tag{6.106}$$

with the tangent stiffness matrix or system Jacobian matrix defined by

$$\mathbf{K}_T = \frac{\partial\left\{\left(1-\alpha_M\right)\mathbf{M}\ddot{\boldsymbol{x}}_{n+1} + \left(1-\alpha_F\right)\left(\boldsymbol{f}_{n+1}^{\text{int}} - \boldsymbol{f}_{n+1}^{\text{ext}}\right)\right\}}{\partial\mathbf{x}_{n+1}} \tag{6.107}$$

6.6 The standard linear tetrahedron

In this section we introduce the relations for the standard linear tetrahedron that will be needed to present non-locking formulations in the next chapter. A volumetric/isochoric split of the internal work is introduced (Section 6.6.1). This split is then performed for the first Piola-Kirchhoff stress tensor (Section 6.6.2). Finally, expressions for the volumetric and isochoric contributions to the nodal internal forces are obtained (Section 6.6.3).

6.6.1 Strain energy function

The internal work is expressed in the reference configuration by (6.30),

$$\mathcal{W}^{\text{int}} = \int_{\Omega_0} \mathbf{P} : \mathbf{F}\, d\Omega_0 \tag{6.108}$$

Even though the final equations are valid for all materials, we assume for simplicity that the material is hyperelastic and characterized by the existence of a strain energy function w^{int},

$$\mathcal{W}^{\text{int}} = \int_{\Omega_0} w^{\text{int}}(\mathbf{F}) d\Omega_0 \tag{6.109}$$

As explained in Section 6.2.3.3, the deformation gradient may be split into a volumetric $\mathbf{F}^{\text{vol}}$ and an isochoric contribution $\mathbf{F}^{\text{iso}}$,

$$\begin{aligned}\mathbf{F}^{\text{iso}} &= (\det(\mathbf{F}))^{-\frac{1}{3}}\,\mathbf{F} = J^{-\frac{1}{3}}\mathbf{F}\\ \mathbf{F}^{\text{vol}} &= (\det(\mathbf{F}))^{\frac{1}{3}}\,\mathbf{I} = J^{\frac{1}{3}}\mathbf{I}\end{aligned} \tag{6.110}$$

For an incompressible material, $J = 1$ so that the volumetric component of the deformation gradient is unity, $\mathbf{F}^{\text{vol}} = \mathbf{I}$. Let us call the distortional strain energy function, noted w^{iso}, the isochoric contribution to the strain energy function, i.e. the energy generated from the volume preserving part of a deformation. Hence, by definition,

$$w^{\text{int,iso}}(\mathbf{F}) = w^{\text{int}}(\mathbf{F}^{\text{iso}}) \tag{6.111}$$

For example, in the simple case of a neo-Hookean material, the isochoric strain energy $w^{\text{int,iso}}$ is expressed in terms of $\mathbf{F}^{\text{iso}}$ as

$$\begin{aligned}w^{\text{int,iso}}(\mathbf{F}) &= \mu\,\left((\det\mathbf{F})^{-\frac{2}{3}}\,\mathbf{F}:\mathbf{F} - 3\right)\\ &= \mu\,\left(\mathbf{F}^{\text{iso}}:\mathbf{F}^{\text{iso}} - 3\right)\end{aligned} \tag{6.112}$$

where μ is the shear modulus.

A nice way to enforce the incompressibility condition is to add a volumetric energy component $w^{\text{int, vol}}$ to the distortional component $w^{\text{int,iso}}$, so that the total strain energy density function is given by the sum

$$w^{\text{int}}(\mathbf{F}) = w^{\text{int,iso}}(\mathbf{F}) + w^{\text{int, vol}}(J) \tag{6.113}$$

and the total strain energy or internal work is given by (6.109)

$$\begin{aligned}W^{\text{int}}(\mathbf{F}) &= \int_{\Omega_0} w^{\text{int,iso}}(\mathbf{F})\,d\Omega_0 + \int_{\Omega_0} w^{\text{int, vol}}(J)\,d\Omega_0\\ &= W^{\text{int,iso}}(\mathbf{F}) + W^{\text{int, vol}}(J)\end{aligned} \tag{6.114}$$

Typically, the volumetric contribution $w^{\text{int, vol}}(J)$ is defined as

$$w^{\text{int, vol}}(J) = \frac{1}{2}\kappa(J-1)^2 \tag{6.115}$$

where κ can be viewed as a penalty number so that incompressibility is enforces for large values of κ, typically $\kappa/\mu > 10^3$. However, for compressible materials that happen to have a hyperelastic strain energy function in the form (6.109), κ represents a true material property, namely the bulk modulus.

6.6.2 First Piola Kirchhoff stress tensor

The first Piola Kirchhoff stress tensor, also called the nominal stress, is obtained from (6.109) and taking account of (6.113) and (6.115),

$$\begin{aligned}\mathbf{P} &= \frac{\partial w^{\text{int}}}{\partial \mathbf{F}} \\ &= \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}} + \frac{\partial w^{\text{int,vol}}}{\partial \mathbf{F}} \\ &= \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}} + \frac{dw^{\text{int,vol}}}{dJ}\frac{\partial J}{\partial \mathbf{F}} \\ &= \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}} + \frac{dw^{\text{int,vol}}}{dJ}\frac{\partial J}{\partial \mathbf{F}} \\ &= \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}} + \kappa(J-1)\frac{\partial J}{\partial \mathbf{F}} \\ &= \mathbf{P}^{\text{iso}} + \mathbf{P}^{\text{vol}}\end{aligned} \tag{6.116}$$

A similarity may be found by recalling the volumetric-deviatoric split of the first Piola-Kirchhoff stress tensor in Section 6.2.6, Equation (6.23):

$$\mathbf{P} = \mathbf{P}^{\text{iso}} + p\,J\,\mathbf{F}^{-T}\,, \qquad \mathbf{P}^{\text{iso}} = J\,\boldsymbol{\sigma}^{\text{iso}}\,\mathbf{F}^{-T} \tag{6.117}$$

Let us now demonstrate that

$$\frac{\partial J}{\partial \mathbf{F}} = J\mathbf{F}^{-T} \tag{6.118}$$

so that, from (6.116), the hydrostatic pressure is given by

$$p = \frac{dw^{\text{int,vol}}}{dJ} = \kappa(J-1) \tag{6.119}$$

As detailed in Appendix B, the directional derivative of the determinant of a matrix is given by

$$D\det(\mathbf{A})[\mathbf{U}] = \det(\mathbf{A})\left(\mathbf{A}^{-T} : \mathbf{U}\right) \tag{6.120}$$

Hence for $\mathbf{A} = \mathbf{U} = \mathbf{F}$

$$DJ[\mathbf{F}] = J\left(\mathbf{F}^{-T} : \mathbf{F}\right) \tag{6.121}$$

Also from Appendix B, the directional derivative and the partial derivative are related by

$$DG[\Delta\mathbf{U}] = \sum_{I,J=1}^{3} \frac{\partial G}{\partial U_{IJ}}\Delta U_{IJ} = \frac{\partial G}{\partial \mathbf{U}} : \Delta\mathbf{U} \tag{6.122}$$

Hence, for $G = \det\mathbf{F} = J$ and $\Delta\mathbf{U} = \mathbf{F}$

$$DJ[\mathbf{F}] = \frac{\partial J}{\partial \mathbf{F}} : \mathbf{F} \tag{6.123}$$

Comparing (6.121) and (6.123) gives

$$\frac{\partial J}{\partial \mathbf{F}} = J\mathbf{F}^{-T} \tag{6.124}$$

so that the volumetric component of the first Piola-Kircchoff stress tensor is (6.116),

$$\mathbf{P}^{\text{vol}} = p\,J\,\mathbf{F}^{-T}\,, \quad \text{with} \quad p = \kappa(J-1) \tag{6.125}$$

An expression for isochoric component of the first Piola-Kircchoff stress tensor $\mathbf{P}^{\text{iso}}$ may be found in the particular case of a neo-Hookean model. For neo-Hookean materials, the isochoric component of the total strain energy function is given by

$$\begin{aligned} w^{\text{int,iso}} &= \frac{1}{2}\mu\left((\det\mathbf{F})^{-\frac{2}{3}}(\mathbf{F}:\mathbf{F}) - 3\right) \\ &= \frac{1}{2}\mu\left(\left(\mathbf{F}^{\text{iso}}:\mathbf{F}^{\text{iso}}\right) - 3\right) \end{aligned} \tag{6.126}$$

Derivation of the above gives the isochoric component of the first Piola-Kirchhoff tensor, for the standard linear tetrahedron and in the case of a neo-Hookean material:

$$\mathbf{P}^{\text{iso}} = \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}} = \mu(\det\mathbf{F})^{-\frac{2}{3}}\left(\mathbf{F} - \frac{1}{3}(\mathbf{F}:\mathbf{F})\mathbf{F}^{-T}\right) \tag{6.127}$$

6.6.3 Nodal internal forces

In this section, an expression for the nodal internal forces $\boldsymbol{f}_I^{\text{int}}$ of the standard linear tetrahedron is obtained, through the linearisation of the of the virtual internal work $\mathcal{W}^{\text{int}}$. Because we introduced a split of this internal strain energy (6.115), the computed nodal forces will also be split into volumetric and isochoric componenents.

Let us first consider that the domain is meshed with linear tetrahedrons. The integral over the domain (6.109) may thus be evaluated by adding the n_e elemental contributions,

$$\mathcal{W}^{\text{int}} = \sum_e^{n_e} \mathcal{W}^{\text{int},e} \tag{6.128}$$

Taking account of (6.113), we have, for one element,

$$\begin{aligned} \mathcal{W}^{\text{int},e} &= \int_{{}^h\Omega_{0,e}} w^{\text{int,iso}}(\mathbf{F}^{\text{iso},e})d\Omega_{0,e} + \int_{{}^h\Omega_{0,e}} w^{\text{int,vol}}(J_e)d\Omega_{0,e} \\ &= \underbrace{w^{\text{int,iso}}(\mathbf{F}^{\text{iso},e})V_e}_{\mathcal{W}^{\text{int,iso},e}} + \underbrace{w^{\text{int,vol}}(J_e)V_e}_{\mathcal{W}^{\text{int,vol},e}} \end{aligned} \tag{6.129}$$

with $J_e = \upsilon_e / V_e$. For the whole meshed domain, we have, from (6.128),

$$\begin{aligned} \mathcal{W}^{\text{int}} &= \sum_{e=1}^{n_e} w^{\text{int,iso}}(\mathbf{F}^{\text{iso},e}) V_e + \sum_{e=1}^{n_e} w^{\text{int,vol}}(J_e) V_e \\ &= \mathcal{W}^{\text{int,iso}}(\mathbf{F}^{\text{iso}}) + \mathcal{W}^{\text{int,vol}}(J) \end{aligned} \tag{6.130}$$

By definition, the virtual work is computed by taking the directional derivative of the work in the direction of a virtual displacement $\delta \boldsymbol{u}$:

$$\begin{aligned} \delta \mathcal{W}^{\text{int}} &= \delta \mathcal{W}^{\text{int,iso}} + \delta \mathcal{W}^{\text{int,vol}} \\ &= D\mathcal{W}^{\text{int,iso}}[\delta \boldsymbol{u}] + D\mathcal{W}^{\text{int,vol}}[\delta \boldsymbol{u}] \end{aligned} \tag{6.131}$$

6.6.3.1 Volumetric nodal internal forces

The volumetric nodal internal forces are obtained by considering the volumetric part of (6.131)

$$\delta \mathcal{W}^{\text{int,vol}} = D\mathcal{W}^{\text{int,vol}}(J)[\delta \boldsymbol{u}] \tag{6.132}$$

At the element level, this gives,

$$\delta \mathcal{W}^{\text{int,vol},e} = D\mathcal{W}^{\text{int,vol},e}(J_e)[\delta \mathbf{u}] \tag{6.133}$$

Developing this equation and using (6.129), we obtain

$$\delta \mathcal{W}^{\text{int,vol},e} = \frac{dw^{\text{int,vol}}}{dJ_e} V_e \, DJ_e[\delta \mathbf{u}] \tag{6.134}$$

As has been done in (6.116) and from (6.115), we define the element pressure by

$$p_e = \frac{dw^{\text{int,vol},e}}{dJ_e} \tag{6.135}$$

Moreover, the directional derivative of J_e has been computed in Appendix B.5, (B.14),

$$DJ_e[\delta \boldsymbol{u}] = J_e \, \text{div}(\delta \boldsymbol{u}) = J_e \frac{\partial \delta u_i}{\partial x_i} \tag{6.136}$$

Using finite element interpolation of the displacement field over the element (6.36), we have

$$DJ_e[\delta \mathbf{u}] = J_e \, \frac{\partial N_I^e}{\partial x_i} \delta u_{iI} = J_e \, \nabla \boldsymbol{N}_I^e \cdot \delta \boldsymbol{u}_I \tag{6.137}$$

with an implicit sum over the nodal indices $I = 1, \dots 4$ for the linear tetrahedron.

Replacing (6.135) and (6.137) into (6.133),

$$\begin{aligned} \delta \mathcal{W}^{\text{int,vol},e} &= p_e V_e J_e \, \nabla \boldsymbol{N}_I^e \cdot \delta \boldsymbol{u}_I \\ &= p_e \upsilon_e \, \nabla \boldsymbol{N}_I^e \cdot \delta \boldsymbol{u}_I \end{aligned} \tag{6.138}$$

Summing the element contributions (6.128), we obtain the global volumetric internal virtual work :

$$\delta \mathcal{W}^{\text{int,vol}} = \sum_{e=1}^{n_e} p_e \upsilon_e \nabla \boldsymbol{N}_I^e \cdot \delta \boldsymbol{u}_I \tag{6.139}$$

Finally, we identify the volumetric component of the nodal internal force at node I for the standard linear tetrahedron as

$$\boldsymbol{f}_{\text{vol},I}^{\text{int}} = \sum_{e=1}^{n_e} p_e \upsilon_e \nabla \boldsymbol{N}_I^e \tag{6.140}$$

6.6.3.2 Isochoric nodal internal forces

Let us now consider the isochoric part of (6.131), and obtain an expression for the isochoric nodal internal forces. The directional derivative of the isochoric strain energy gives

$$\begin{aligned} D\mathcal{W}^{\text{int,iso}}[\delta \boldsymbol{u}] &= \sum_{e=1}^{n_e} V_e D w^{\text{int,iso}}[\delta \boldsymbol{u}] \\ &= \sum_{e=1}^{n_e} V_e \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}_e} : D\mathbf{F}_e[\delta \boldsymbol{u}] \end{aligned} \tag{6.141}$$

First, we have from (6.116), for the isochoric component of the first Piola-Kirchhoff stress tensor:

$$\mathbf{P}^{\text{iso}} = \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}} \tag{6.142}$$

Second, the directional derivative of the deformation gradient gives, from Appendix B.4,

$$D\mathbf{F}[\delta \boldsymbol{u}] = \nabla_0 \delta \boldsymbol{u} \tag{6.143}$$

Introducing (6.142) and (6.143) into (6.141) gives

$$D\mathcal{W}^{\text{int,iso}}[\delta \boldsymbol{u}] = \sum_{e=1}^{n_e} V_e \mathbf{P}^{\text{iso}} : \nabla_0 \delta \boldsymbol{u} \tag{6.144}$$

Discretising the virtual displacement field in the linear tetrahedron using finite element semi-discretisation (6.36) we obtain,

$$
\begin{aligned}
&= \sum_{e=1}^{n_e} V_e P_{iA}^{\text{iso}} \frac{\partial N_I^e}{\partial X_A} \delta u_{Ii} \\
&= \sum_{e=1}^{n_e} V_e \mathbf{P}^{\text{iso}} \nabla_0 \boldsymbol{N}_I^e \cdot \delta \boldsymbol{u}_I \qquad (6.145) \\
&= \boldsymbol{f}_{\text{iso},I}^{\text{int}} \cdot \delta \boldsymbol{u}_I \qquad (6.146)
\end{aligned}
$$

Finally, we identify the isochoric component of the nodal internal force at node I for the standard linear tetrahedron as

$$
\boldsymbol{f}_{\text{iso},I}^{\text{int}} = \sum_{e=1}^{n_e} V_e \mathbf{P}^{\text{iso}} \nabla_0 \boldsymbol{N}_I^e \qquad (6.147)
$$

So that in the end the nodal internal force for the standard linear tetrahedron may be computed as the sum of its isochoric (6.146) and volumetric (6.140) contributions

$$
\begin{aligned}
\boldsymbol{f}_I^{\text{int}} &= \boldsymbol{f}_{\text{iso},I}^{\text{int}} + \boldsymbol{f}_{\text{vol},I}^{\text{int}} \\
&= \sum_{e=1}^{n_e} V_e \, \mathbf{P}^{\text{iso}} \, \nabla_0 \boldsymbol{N}_I^e + \sum_{e=1}^{n_e} p_e \upsilon_e \nabla \boldsymbol{N}_I^e \qquad (6.148)
\end{aligned}
$$

6.7 Conclusions

This chapter introduced the basic equations of finite element analysis that will be needed to present locking-free formulations for the tetrahedra in the next chapter. The deformation gradient or Jacobian matrix $\boldsymbol{F}$ was presented and we have seen that its determinant gives the volume change between the current and the reference configuration. The principle of virtual work and the principle of virtual power were introduced and the discretised equations of motion were obtained by finite element discretisation. Linearisation of the virtual work equation was performed, and an expression for the tangent stiffness matrix was obtained by subsequent finite element discretisation. A volumetric-isochoric split was performed on the deformation gradient and on the Cauchy stress tensor. This split was also performed on the internal work and the PK1 stress tensor and expressions for the volumetric and the isochoric contributions of the internal forces were obtained.

Chapter 7

Towards a locking-free formulation for the linear tetrahedron

It is well-known that the performance of low-order finite elements becomes extremely poor as the incompressible limit is approached. Problems where incompressibility is encountered include the analysis of rubbery solids, which are typically modelled as nearly incompressible[1] hyperelastic materials, as well as the analysis of J2 elasto-plastic metals, for which an isochoric plastic flow is generally assumed (von Mises plasticity). In these situations an overstiff behaviour, called *volumetric locking*, is observed as a consequence of the inability of low-order interpolation polynomials to adequately represent general volume-preserving deformation fields.

Volumetric locking can be eliminated by employing higher-order finite elements. However, due to their simplicity and robustness, low-order elements are often preferred in large-scale computations. In the case of hexahedral meshes, an effective and popular unlocking solution is to use the hexahedron with reduced or selective reduced integration [15, 63], sometimes with hourglassing stabilization. Unfortunately, as seen in the previous chapters, it is not always possible to mesh complex geometries, automatically and without jeopardizing the geometric representation, with hexahedrons so that tetrahedral meshes are more practical in Computational Biomechanics. Hence, there is a need for low-order tetrahedral elements that behave properly without volumetric locking.

To tackle the problem two main classes of approaches have been proposed: approaches based on a split of the governing equations and approaches that do not rely on a split of the equations.

[1] In this work we often call *nearly incompressible* materials as *incompressible* materials in order to simplify the prose.

The first category of non-locking low order triangular or tetrahedral finite element formulations are based on a split of the governing equations. This decomposition technique has first been introduced in the framework of fluid mechanics by Chorin [40]. On this basis, Schneider et al. [151] introduced a split of the pressure and the velocity field with equal order interpolation of both fields to solve problems in fluid mechanics (other types of interpolation are possible). Later, Zienkiewicz et al. [193] introduced a split of the displacement and the pressure field for problems in solid mechanics where explicit time integration is used. In fact, the Stokes problem in fluid mechanics (expressed in terms of velocity and pressure fields) is equivalent to the incompressible linear elasticity problem (expressed in terms of displacement and pressure fields) so that many formulations developed in the framework of fluid mechanics have inspired the solid mechanics community. The same year, in 1998, the now very popular Average Nodal Pressure (ANP) approach has been introduced by Bonet and Burton [25]. Here, pressure and displacement are still considered as independent fields, but the pressure is averaged on the triangle or tetrahedral nodes. Because the number of nodes in a tetrahedral mesh is lower than the number of elements, the number of constraints is reduced and volumetric locking is obviated. Other *nodal based formulations* derived from the ANP have then been proposed. Dohrmann et al. [55], then followed by Bonet et al. [27], proposed to average the full strain tensor at each node in order to alleviate the possible additional shear locking. Unfortunately, the proposed formulation becomes sensitive to the hourglassing effect. A tentative approach to stabilize this hourglassing effect has been proposed in an implicit framework by Puso and Solberg [147]. In the field of Biomechanics, the Average Nodal Pressure tetrahedron has been used by Joldes et al. [91] to model brain deformations during neurosurgery. Joldes et al. [91] also extended the ANP formulation to tetrahedral meshes containing multiple material domains.

The second category of approaches do not rely on a split of the equations. An early formulation to solve the Stokes equation is the MINI element proposed by Arnold et al. [7]. In this formulation, the pressure and the velocity fields are approximated by C^0 continuous linear interpolations and the velocity is augmented by a cubic bubble function to satisfy the Babuska-Brezzi condition. In solid mechanics, volume bubble functions have been employed within a mixed formulation to create non-locking tetrahedral finite elements for small and finite elastic strains [165]. The MINI element has successfully been used to simulate metal forming processes [45]. Another solution to solve the Stokes problem with equal order interpolation of pressure and velocity, proposed by Hughes et al. [85], is to employ a Petrov-Galerkin method augmented by Galerkin least squares stabilization terms. A third solution is the method of incompatible modes, which consist of a decomposition of the displacement field into a compatible and an incompatible part [182]. A similar approach to the method of incompatible modes is the method of enhanced assumed strains proposed

by Simo and Rifai [160]. In this fourth approach, the strain is introduced as an additional field and constructed in order to pass the patch test, instead of being derived from the symmetric gradient of an incompatible displacement field. As showed in Mahnken et al. [113], the mixed method of incompatible modes and the mixed method of enhanced strains may both be derived from a five field weak formulation involving compatible displacements, incompatible displacements, pressure, enhanced strains and stresses.

In parallel to the *nodal based formulations* and the *mixed formulations* presented above, the *F-bar-Patch* methodology has been proposed by de Souza Neto et al. [52]. The idea is to apply the volumetric constraints on non-overlapping patches of finite elements. Even though this method does not exactly fulfil the Babuska-Brezzi conditions, it has shown to be quite effective in removing volumetric locking. However, the definition of non-overlapping patches of element is quite tedious in 3D, so that the authors end up in subdividing each tetrahedron into six smaller ones in order to be able to apply their method.

In the following sections, we first present the formulations that are relevant to our work. Section 7.1 reviews nodal based formulations, which are suitable for applications with explicit time integration. Section 7.2 presents the F-bar methodology for hexahedrons and the F-bar-patched method for tetrahedrons. In Sections 7.3 and 7.4 we present two successive ideas to remove the spurious stiffness of the standard tetrahedron. The first idea will give what we call the face or node-neighbourhood patch volume change ratio linear tetrahedron, and is based on an attempt to solve the problem of tedious patch definition in the F-bar-patch methodology [52]. The second proposed element formulation will be called the Average Elemental Jacobian (AEJ) tetrahedral element and was inspired from both nodally integrated tetrahedral formulations and the F-bar methodology.

7.1 Nodal-based formulations

In this section, nodal-based formulations for incompressible or nearly incompressible applications are reviewed chronologically. These formulations all have in common that new nodal volumes are defined so that the incompressibility constraints are imposed on these nodal volumes instead of on each element, thus reducing the number of constraints imposed. As will be seen in this section, nodal-based formulations were developed in the context of explicit dynamic finite element simulations, and, the new nodal quantities are obtained by using the same averaging process as for the nodal masses in the lumped mass matrix.

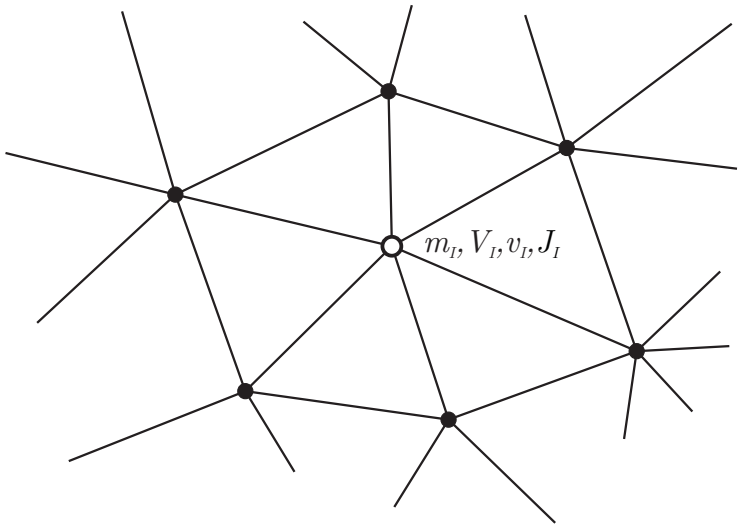


FIGURE 7.1: ***The Average Nodal Pressure Linear Tetrahedron****, ANP, proposed by Bonet and Burton [25].*

7.1.1 The Average Nodal Pressure Linear Tetrahedron

The standard formulation for the linear tetrahedron leads to volumetric locking because the volume of the n_e elements is required to be constant. Hence, the motion of a mesh is controlled by $3n_n$ degrees of freedom and n_e constraints. In a tetrahedral mesh, the ratio between the number of elements n_e and the number of nodes n_n is typically of more than 5 to 1; so we typically have $5n_n$ constraints for $3n_n$ degrees of freedom. As a consequence, the motion of the mesh is too constrained and locking occurs.

The solution proposed by Bonet and Burton [25] to overcome the locking of the linear tetrahedron is to enforce the volumetric constraint over the volume attached to a node instead than over the volume of an element. Going back to the remark made in the previous paragraph, we may deduce that this solution typically reduces the number of imposed constraints by 5.

In practice, the average nodal pressure tetrahedron is proposed in an explicit time integration framework and the formulation consists in a re-definition of the volumetric component of the nodal internal forces $\boldsymbol{f}^{\text{int}}_{\text{vol},I}$, the development of which has been detailed in the case of the standard linear tetrahedron in Section 6.6.3.1.

7.1.1.1 Nodal volumes and average nodal volumetric strain

Bonet and Burton [25] define nodal volumes V_I using the same procedure than for the nodal masses in the lumped mass matrix (Equations (6.101) and (6.102))

$$V_I = \sum_{e=1}^{n_{e,I}} V_{I,e} \quad \text{with} \quad V_{I,e} = \int_{\Omega_{0,e}} N_I \, d\Omega_0 = \frac{1}{4} V_e \tag{7.1}$$

and, in the current configuration,

$$v_I = \sum_{e=1}^{n_{e,I}} v_{I,e} \quad \text{with} \quad v_{I,e} = \int_{\Omega_e} N_I \, d\Omega = \frac{1}{4} v_e \tag{7.2}$$

A current-to-initial nodal volume ratio, or average nodal volumetric strain, is then defined as

$$J_I = \frac{v_I}{V_I} = \frac{\sum_{e=1}^{n_{e,I}} v_e}{\sum_{e=1}^{n_{e,I}} V_e} \tag{7.3}$$

with $n_{e,I}$ the number of elements connected to node I.

The above relationship may also be rewritten in the form:

$$J_I = \frac{1}{V_I} \int_{\Omega_0} J N_I \, d\Omega_0 \tag{7.4}$$

which may be compared to the lumped mass at node I:

$$m_I = \int_{\Omega_0} \rho_0 N_I \, d\Omega_0 \tag{7.5}$$

7.1.1.2 Volumetric strain energy

Let us assume that the strain energy density function can be decomposed into a volumetric and an isochoric component as already presented for the linear tetrahedron in (6.113):

$$W^{\text{int}}(\mathbf{F}) = \int_{\Omega_0} w^{\text{int,iso}}(\mathbf{F}) \, d\Omega_0 + \int_{\Omega_0} w^{\text{int, vol}}(J) \, d\Omega_0 \tag{7.6}$$

The volumetric strain density function $w^{\text{int, vol}}(J)$ is approximated by assuming that the volume ratio J remains constant over the volume attached to each node. Therefore the volumetric internal work is computed by summing up the individual *nodal* contributions (as compared to *element* contributions for the standard linear tetrahedron (6.129)),

$$\begin{aligned} \mathcal{W}^{\text{int,vol}} &= \int_{\Omega_0} w^{\text{int,vol}}(J) d\Omega_0 \\ &\approx \sum_{I=1}^{n} w^{\text{int,vol}}(J_I) V_I \end{aligned} \tag{7.7}$$

where n denotes the number of nodes in the mesh.

7.1.1.3 Volumetric internal forces

Taking the directional derivative of the volumetric internal work in the direction of a virtual displacement $\delta \boldsymbol{u}$ gives the volumetric virtual work. Taking account of (7.7),

$$\begin{aligned} \delta \mathcal{W}^{\text{int,vol}} &= D\mathcal{W}^{\text{int,vol}}\left[\delta \boldsymbol{u}\right] \\ &= \sum_{I=1}^{n} V_I \, Dw^{\text{int,vol}}\left[\delta \boldsymbol{u}\right] \\ &= \sum_{I=1}^{n} V_I \left.\frac{dw^{\text{int,vol}}}{dJ}\right|_{J=J_I} DJ_I\left[\delta \boldsymbol{u}\right] \end{aligned} \tag{7.8}$$

In a similar way to (6.116), we define the average nodal pressure

$$p_I = \left.\frac{dw^{\text{int,vol}}}{dJ}\right|_{J=J_I} = \kappa\left(J_I - 1\right) = \kappa\left(\frac{\upsilon_I - V_I}{V_I}\right) \tag{7.9}$$

The directional derivative of the average nodal volumetric strain appearing in (7.8) is computed with the help of equations (7.2), (7.3) and (6.136):

$$\begin{aligned} DJ_I\left[\delta \boldsymbol{u}\right] &= \frac{1}{V_I} D\upsilon_I\left[\delta \boldsymbol{u}\right] \\ &= \frac{1}{V_I} \sum_{e=1}^{n_{e,I}} \frac{1}{4} D\upsilon_e\left[\delta \boldsymbol{u}\right] \\ &= \frac{1}{V_I} \sum_{e=1}^{n_{e,I}} \frac{1}{4} V_e \, DJ_e\left[\delta \boldsymbol{u}\right] \\ &= \frac{1}{V_I} \sum_{e=1}^{n_{e,I}} \frac{1}{4} V_e \, J_e \, \frac{\partial \delta u_i^e}{\partial x_i} \end{aligned} \tag{7.10}$$

Using finite element interpolation of the displacement field over the element (6.36), the expression becomes

$$\begin{aligned} DJ_I\left[\delta \boldsymbol{u}\right] &= \frac{1}{V_I} \sum_{e=1}^{n_{e,I}} \frac{1}{4} V_e \, J_e \, \frac{\partial N_J^e}{\partial x_i} \delta u_{iJ} \\ &= \frac{1}{V_I} \sum_{e=1}^{n_{e,I}} \frac{1}{4} \upsilon_e \, \nabla \boldsymbol{N}_J \cdot \delta \boldsymbol{u}_J \end{aligned} \tag{7.11}$$

Replacing (7.9) and (7.11) into the equation of the virtual volumetric work (7.8) gives

$$\begin{aligned}\delta \mathcal{W}^{\text{int,vol}} &= \sum_{I=1}^{n} \sum_{e=1}^{n_{e,I}} \frac{1}{4} p_I \upsilon_e \, \nabla \boldsymbol{N}_J \cdot \delta \boldsymbol{u}_J \\ &= \sum_{e=1}^{n_{e,I}} \left(\sum_{I=1}^{4} \frac{1}{4} p_I \right) \upsilon_e \, \nabla \boldsymbol{N}_J \cdot \delta \boldsymbol{u}_J \\ &= \sum_{e=1}^{n_{e,I}} \bar{p}_e \upsilon_e \, \nabla \boldsymbol{N}_J \cdot \delta \boldsymbol{u}_J \end{aligned} \tag{7.12}$$

where we have defined the average element pressure:

$$\bar{p}_e = \frac{1}{4} \sum_{I=1}^{4} p_I \tag{7.13}$$

The latter would correspond to the pressure computed at the centroid of the tetrahedral element by linear interpolation of the nodal values.

The volumetric component of the internal nodal force at node I is identified from (7.12) as

$$\mathbf{f}_{\text{vol},I}^{\text{int}} = \sum_{e=1}^{n_{e,I}} \bar{p}_e \upsilon_e \nabla \boldsymbol{N}_I \tag{7.14}$$

Compared to the standard element (6.140), the expression is similar except for the element pressure that is now computed as an average of the nodal pressures: p_e in (6.140) becomes $\bar{p}_e$ in (7.14).

7.1.2 Extension to domains with multiple materials

Joldes et al. [91] have extended the Average Nodal Pressure (ANP) element, proposed by Bonet and Burton [25] and presented above, for a better handling of material interfaces. Indeed, in the case of multiple interfaces, the element pressure can no longer be computed by (7.9) as it is not clear which bulk modulus should be used for the nodal pressure computation.

The solution proposed by Joldes et al. [91] consists in defining a different nodal volume for each material type α converging at node I:

$$\upsilon_I^{(\alpha)} = \sum_{e=1} n_I^{(\alpha)} \frac{1}{4} \upsilon_e \tag{7.15}$$

where $n_I^{(\alpha)}$ represents the number of elements of material type α sharing node I. Different nodal Jacobians $J_I^{(\alpha)}$ are then computed from these material volumes, $J_I^{(\alpha)} = \upsilon_I^{(\alpha)} / V_I^{(\alpha)}$.

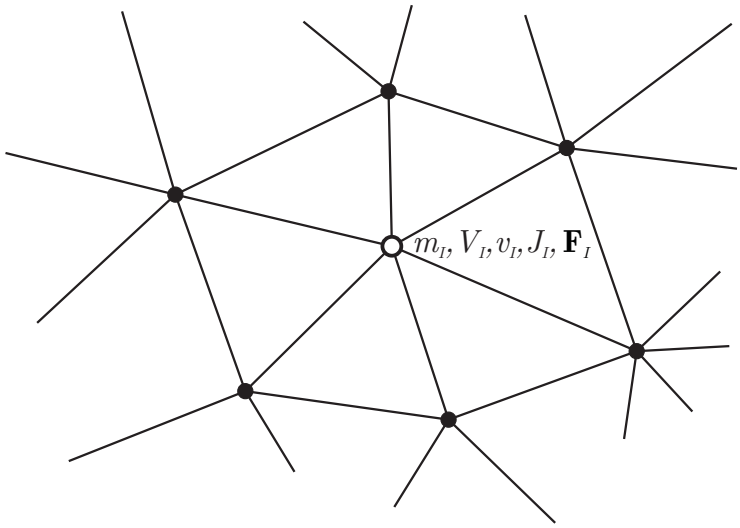


FIGURE 7.2: ***The Average Nodal Strain Linear Tetrahedron.*** *proposed by Dohrmann et al. [55]*

Eventually, these material Jacobians are used along with the material bulk modulus to define a nodal pressure for each material α as

$$p_I^{(\alpha)} = \kappa^{(\alpha)} \left(J_I^{(\alpha)} - 1 \right) \tag{7.16}$$

This solution is very straightforward. It may however cause implementation problems as several pressures must be stored at each node.

7.1.3 The Average Nodal Strain Linear Tetrahedron

The average nodal pressure element proposed by Bonet and Burton [25] presented in Section 7.1.1 successfully removes the volumetric locking but spurious shear locking may still exist. To overcome this problem, Dohrmann et al. [55] proposed to apply the nodal averaging process (7.4) on the whole strain tensor rather on the on the volumetric part uniquely. The resulting formulation, which was presented for small strain elasticity, has subsequently been extended to the large strain elasto-plastic regime by Bonet et al. [27], still in the framework of explicit time integration.

7.1.3.1 Nodal deformation gradient

The definition of nodal volumes V_I and nodal Jacobians J_I are identical to the average nodal pressure approach [25] presented in Section 7.1.1. The major difference in the average nodal strain formulation is that a nodal deformation gradient $\mathbf{F}_I$ is now defined, using the

same averaging process than for V_I and J_I. This nodal deformation gradient $\mathbf{F}_I$ replaces the standard element deformation gradient $\mathbf{F}_e = \frac{\partial x_e}{\partial X_e}$ as the main kinematics variable to define the deformation of the solid.

The nodal deformation gradient $\mathbf{F}_I$ is defined as

$$\mathbf{F}_I = \frac{1}{V_I}\int_{\Omega_0} \mathbf{F} N_I d\Omega_0 = \frac{1}{V_I}\frac{1}{4}\sum_{e=1}^{n_{e,I}} V_e \mathbf{F}_e = \frac{\frac{1}{4}\sum_{e=1}^{n_{e,I}} V_e \mathbf{F}_e}{\frac{1}{4}\sum_{e=1}^{n_{e,I}} V_e} = \frac{\sum_{e=1}^{n_{e,I}} V_e \mathbf{F}_e}{\sum_{e=1}^{n_{e,I}} V_e} \tag{7.17}$$

where the second relation results from the fact that the element shape functions are linear so that the point-wise deformation gradient is constant within an element. These equations should be compared with the relations for the nodal Jacobian obtained for the average nodal pressure element (7.3) and (7.4).

For materials in which the volumetric and isochoric responses are uncoupled, for example in the case of Von-Mises plasticity, a modified averaged nodal deformation gradient is defined as

$$\bar{\mathbf{F}}_I = \left(\frac{J_I}{\det \mathbf{F}_I}\right)^{\frac{1}{3}} \mathbf{F}_I \tag{7.18}$$

so that the determinant of $\bar{\mathbf{F}}_I$ is given by the average nodal Jacobian J_I, defined by (7.3) or equivalently by (7.4) . This modification has its importance as the current mesh volumes are correctly evaluated in (7.4) whereas the determinant of the nodal deformation gradient (7.17) is only an asymptotic approximation.

Decomposing this the modified averaged nodal deformation gradient $\bar{\mathbf{F}}_I$ into its volumetric and volume preserving components using the standard relations for the volumetric/isochoric split of the deformation gradient presented in Section 6.2.3.3 gives, by construction,

$$\begin{aligned} \bar{\mathbf{F}}_I^{\text{iso}} &= \mathbf{F}_I^{\text{iso}} \\ \bar{\mathbf{F}}_I^{\text{vol}} &= J_I^{\frac{1}{3}}\mathbf{I} \end{aligned} \tag{7.19}$$

7.1.3.2 Total strain energy

Because the kinematics is now defined via the nodal deformation gradient, the internal work is written as a sum of nodal strain energies. And, supposing that the strain energy is constant over at the nodes of the linear tetrahedrons,

$$\mathcal{W}^{\text{int}}(\mathbf{F}) = \int_{\Omega_0} w^{\text{int}}(\mathbf{F})\, d\Omega_0 = \sum_{I=1}^{n} V_I w^{\text{int}}(\mathbf{F}) \tag{7.20}$$

Let us assume as previously that the strain energy density function can be decomposed into a volumetric and an isochoric components as presented in Equation (6.113), (7.20) becomes

$$\mathcal{W}^{\text{int}}(\mathbf{F}) = \sum_{I=1}^{n} V_I \, w^{\text{int,iso}}(\mathbf{F}_I) + \sum_{I=1}^{n} w^{\text{int, vol}}(J_I) \tag{7.21}$$

7.1.3.3 Internal forces

The nodal internal force vector is obtained as usual by differentiating the total strain energy in the direction of a virtual displacement $\delta \boldsymbol{u}$

$$\delta \mathcal{W}^{\text{int}} = D\mathcal{W}^{\text{int}}[\delta \boldsymbol{u}] = D\mathcal{W}^{\text{int,vol}}[\delta \boldsymbol{u}] + D\mathcal{W}^{\text{int,iso}}[\delta \boldsymbol{u}] \tag{7.22}$$

The volumetric term of this equation is, by construction, the same as for the average nodal pressure tetrahedron 7.1.1. Therefore, only the isochoric component of the nodal internal forces is obtained here.

The directional derivative of the isochoric strain energy gives

$$\begin{aligned} D\mathcal{W}^{\text{int,iso}}[\delta \boldsymbol{u}] &= \sum_{I=1}^{n} V_I \, Dw^{\text{int,iso}}(\mathbf{F}_I)[\delta \boldsymbol{u}] \\ &= \sum_{I=1}^{n} V_I \, \frac{\partial w^{\text{int,iso}}}{\partial \mathbf{F}_I} : D\mathbf{F}_I[\delta \boldsymbol{u}] \\ &= \sum_{I=1}^{n} V_I \, \mathbf{P}_I^{\text{iso}} : D\mathbf{F}_I[\delta \boldsymbol{u}] \\ &= \sum_{I=1}^{n} V_I \, P_{I,iA}^{\text{iso}} DF_{I,iA}[\delta \boldsymbol{u}] \end{aligned} \tag{7.23}$$

where the volume preserving component of the first Piola-Kirchhoff stress tensor $\mathbf{P}^{\text{iso}}$ has been introduced in Sections 6.2.6 and 6.6.2 but is here computed using the nodal deformation gradient $\mathbf{P}_I^{\text{iso}} = \mathbf{P}^{\text{iso}}(\mathbf{F}_I)$.

Let us now compute the directional derivative of the nodal deformation gradient given by (7.17)

$$\begin{aligned} D\mathbf{F}_I[\delta \boldsymbol{u}] &= \frac{1}{V_I} \frac{1}{4} \sum_{e=1}^{n_{e,I}} V_e \, D\mathbf{F}_e[\delta \boldsymbol{u}] \\ &= \frac{1}{V_I} \frac{1}{4} \sum_{e=1}^{n_{e,I}} V_e \, \left(\nabla_0 \delta \boldsymbol{u} \right) \\ DF_{I,iA}[\delta \boldsymbol{u}] &= \frac{1}{V_I} \frac{1}{4} \sum_{e=1}^{n_{e,I}} V_e \, \frac{\partial \delta u_i}{\partial X_A} \end{aligned} \tag{7.24}$$

where the second relation comes from Equation (B.11).

Using the finite element shape function to interpolate the displacement field within the elements from its nodal displacements gives

$$D\mathbf{F}_{I,iA}[\delta\boldsymbol{u}] = \frac{1}{V_I}\frac{1}{4}\sum_{e=1}^{n_{e,I}} V_e \frac{\partial N_J}{\partial X_A}\delta u_{iJ} \tag{7.25}$$

where the implicit summing convention is assumed on index J.

Substituting this result in (7.23) and re-arranging the terms, we have

$$\begin{aligned} D\mathcal{W}^{\text{int,iso}}[\delta\boldsymbol{u}] &= \sum_{I=1}^{n} P^{\text{iso}}_{I,iA}\,\frac{1}{4}\sum_{e=1}^{n_{e,I}} V_e\,\frac{\partial N_J}{\partial X_A}\delta u_{iJ} \\ &= \sum_{e=1}^{n_e} V_e\left(\sum_{I=1}^{4}\frac{1}{4}P^{\text{iso}}_{I,iA}\right)\frac{\partial N_J}{\partial X_A}\delta u_{iJ} \\ &= \sum_{e=1}^{n_e} V_e\,\bar{P}^{\text{iso}}_{e,iA}\,\frac{\partial N_J}{\partial X_A}\delta u_{iJ} \\ &= \sum_{e=1}^{n_e} V_e\,\bar{\mathbf{P}}^{\text{iso}}_{e}\,\nabla_0\boldsymbol{N}_J\cdot\delta u_J \\ &= \boldsymbol{f}^{\text{int,iso}}_{I}\cdot\delta u_I \end{aligned} \tag{7.26}$$

where the element isochoric component of the nominal stress $\bar{\mathbf{P}}^{\text{iso}}_e$ represents the average of the nodal stress tensors $\mathbf{P}^{\text{iso}}_I$ obtained at the four nodes of the linear tetrahedral element

$$\bar{\mathbf{P}}^{\text{iso}}_{e} = \frac{1}{4}\sum_{I=1}^{4}\mathbf{P}^{\text{iso}}_{I} \tag{7.27}$$

In the end, we identify the volume preserving part of the nodal internal forces

$$\boldsymbol{f}^{\text{int,iso}}_{I} = \sum_{e=1}^{n_{e,I}} V_e\,\bar{\mathbf{P}}^{\text{iso}}_{e}\,\nabla_0\mathbf{N}_I \tag{7.28}$$

The main difference with previous formulations is that the element stresses are now computed as an average of nodal stresses, which are obtained by evaluating the constitutive equations with the nodal deformation gradient $\mathbf{F}_I$, which, in turn, has been defined as an weighted average of the deformation gradients of the neighbouring elements.

Recalling the previous result for the volumetric part of the forces (7.14), we have, for the total nodal internal forces

$$\boldsymbol{f}^{\text{int}}_{I} = \boldsymbol{f}^{\text{int,vol}}_{I} + \boldsymbol{f}^{\text{int,iso}}_{I} = \sum_{e=1}^{n_{e,I}} \bar{p}_e\upsilon_e\nabla\boldsymbol{N}_I + \sum_{e=1}^{n_{e,I}} V_e\,\bar{\mathbf{P}}^{\text{iso}}_{e}\,\nabla_0\mathbf{N}_I \tag{7.29}$$

which should be compared with the expression obtained for the standard linear tetrahedron (6.148).

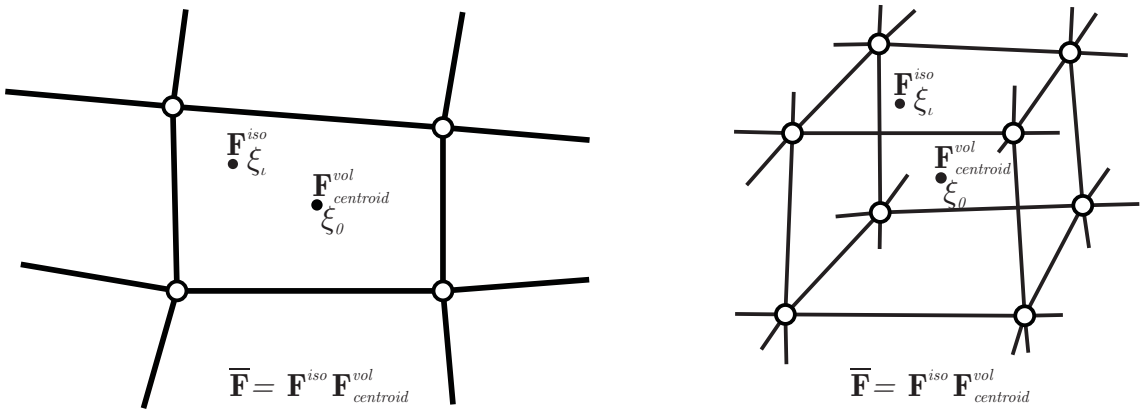


FIGURE 7.3: ***F-bar quadrilateral and hexahedral elements.*** *The modified deformation gradient* $\bar{\mathbf{F}}$ *of the F-bar method is defined as the composition of the isochoric component of* $\mathbf{F}$ *with the volumetric component of* $\mathbf{F}_{centroid}$

7.1.3.4 Conclusions

In the average nodal pressure element, the volumetric locking was removed by averaging the element Jacobians (volumetric deformation) at the mesh nodes. The aim of the present formulation is to also remove the locking in bending by using a similar averaging approach for the isochoric part of the deformation gradient (volume preserving deformation).

Unfortunately, spurious low energy modes appear in the average nodal strain approach and stabilization of the tetrahedron is needed [55]. An efficient procedure to stabilize the average nodal strain element has been proposed by Puso and Solberg [147].

7.2 F-bar and F-bar-patched methods

The *F-bar* methodology for hexahedral elements and its extension to tetrahedral elements have been proposed by de Souza Neto et al. [51, 52]. The idea is to define a modified deformation gradient, called F-bar and denoted $\bar{\mathbf{F}}$. This modified deformation gradient is then used to compute the stresses.

7.2.1 Quadrilateral and Hexahedral F-bar Elements

This section is dedicated to the presentation of the F-bar method to overcome locking effects in low order quadrilateral and hexahedral elements.

7.2.1.1 Definition of a Modified Deformation Gradient $\bar{\mathbf{F}}$

First, the volumetric/isochoric split (6.9) is applied to the deformation gradient $\mathbf{F}$ at the Gauss point of interest as well as to the deformation gradient $\mathbf{F}_{\text{centroid}}$ computed at the centroid of the element (Figure 7.3):

$$\begin{aligned} \mathbf{F} &= \mathbf{F}^{\text{iso}}\,\mathbf{F}^{\text{vol}} \\ \mathbf{F}_{\text{centroid}} &= \mathbf{F}^{\text{iso}}_{\text{centroid}}\,\mathbf{F}^{\text{vol}}_{\text{centroid}} \end{aligned} \tag{7.30}$$

with

$$\begin{aligned} \mathbf{F}^{\text{iso}} &= (\det(\mathbf{F}))^{-\frac{1}{3}}\,\mathbf{F} = J^{-\frac{1}{3}}\,\mathbf{F} && \text{and} \quad \mathbf{F}^{\text{iso}}_{\text{centroid}} = \left(\det(\mathbf{F}_{\text{centroid}})\right)^{-\frac{1}{3}}\,\mathbf{F}_{\text{centroid}} \\ \mathbf{F}^{\text{vol}} &= (\det(\mathbf{F}))^{\frac{1}{3}}\,\mathbf{I} = J^{\frac{1}{3}}\,\mathbf{I} && \text{and} \quad \mathbf{F}^{\text{vol}}_{\text{centroid}} = \left(\det(\mathbf{F}_{\text{centroid}})\right)^{\frac{1}{3}}\,\mathbf{I} \end{aligned} \tag{7.31}$$

The modified deformation gradient $\bar{\mathbf{F}}$ of the F-bar method is defined as the composition of the isochoric component of $\mathbf{F}$ with the volumetric component of $\mathbf{F}_{\text{centroid}}$,

$$\bar{\mathbf{F}} = \mathbf{F}^{\text{iso}}\,\mathbf{F}^{\text{vol}}_{\text{centroid}} = \left(\frac{\det(\mathbf{F}_{\text{centroid}})}{\det(\mathbf{F})}\right)^{\frac{1}{3}}\mathbf{F} \tag{7.32}$$

This formulation leads to the following two important properties for the isochoric and volumetric deformation gradient. First, the isochoric part of the modified deformation gradient at a Gauss point $\bar{\mathbf{F}}^{\text{iso}}$ equals the isochoric part of the original deformation gradient at the Gauss point $\mathbf{F}^{\text{iso}}$. Second, the volumetric part of the modified deformation gradient at a Gauss point $\bar{\mathbf{F}}^{\text{vol}}$ equals the volumetric part the deformation gradient evaluated at the centre of the element $\mathbf{F}^{\text{vol}}_{\text{centroid}}$. Indeed,

$$\begin{aligned} \bar{\mathbf{F}}^{\text{iso}} &= \left(\det(\bar{\mathbf{F}})\right)^{-\frac{1}{3}}\,\bar{\mathbf{F}} = \det(\mathbf{F}_{\text{centroid}})^{-\frac{1}{3}}\,\left(\frac{\det(\mathbf{F}_{\text{centroid}})}{\det(\mathbf{F})}\right)^{\frac{1}{3}}\mathbf{F} = (\det(\mathbf{F}))^{-\frac{1}{3}}\,\mathbf{F} = \mathbf{F}^{\text{iso}} \\ \bar{\mathbf{F}}^{\text{vol}} &= \left(\det(\mathbf{F}_{\text{centroid}})\right)^{\frac{1}{3}}\,\mathbf{I} = \mathbf{F}^{\text{vol}}_{\text{centroid}} \end{aligned} \tag{7.33}$$

This formulation implies that, for materials for which the isochoric and volumetric constitutive responses are uncoupled, the pressure is constant over the quadrangular or hexahedral element.

During the finite element simulation, the standard deformation gradient $\mathbf{F}$ is replaced by the modified deformation gradient $\bar{\mathbf{F}}$ for the computation of the stresses at the Gauss points. It is important to note that the classical Cauchy stress is used here to compute the stresses so that the formulation is adequate for all material laws.

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}\left(\boldsymbol{\alpha}^{n}, \mathbf{B}\right) \tag{7.34}$$

where $\boldsymbol{\alpha}^n$ denotes the set of internal variables of the model at time t_n and $\mathbf{B} = \mathbf{F}\,\mathbf{F}^T$ is the Cauchy-Green tensor. In order to simplify the notations and because we are only interested in the dependence of $\boldsymbol{\sigma}$ on $\mathbf{F}$ we will use the notation

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}\left(\mathbf{F}\right) \tag{7.35}$$

instead of (7.34) in this dissertation.

The element internal force is computed in the same way than for the standard element (6.50):

$$\begin{aligned} \boldsymbol{f}_I^{\text{int},e} &= \int_{\Omega_e} \nabla \mathbf{N}_I^T\, \boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right) d\Omega_e = \int_{\Omega_e} \mathbf{G}_I^T\, \boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right) d\Omega_e \\ f_{Ii}^{\text{int},e} &= \int_{\Omega_e} \frac{\partial N_I}{\partial x_j}\, \sigma_{ji}\left(\bar{\mathbf{F}}\right) d\Omega_e = \int_{\Omega_e} G_{Ij}\, \sigma_{ji}\left(\bar{\mathbf{F}}\right) d\Omega_e \end{aligned} \tag{7.36}$$

In the above, the *standard* **G**-matrix is used to compute the nodal internal forces. This is thanks to the fact that, in contrast with other methods [27, 123, 161], the assumed deformation gradient has been introduced in the stress constitutive functional rather than in the corresponding strain energy functional. Hence, this approach is easier to implement in existing displacement-based element routines.

Equivalently, in the Total Lagrangian formulation, the first Piola Kirchhoff stress is computed using the F-bar deformation gradient in the constitutive equations

$$\mathbf{P} = \mathbf{P}\left(\bar{\mathbf{F}}\right) \tag{7.37}$$

and the element internal forces are given by

$$\begin{aligned} \boldsymbol{f}_I^{\text{int},e} &= \int_{\Omega_e} \mathbf{P}\left(\bar{\mathbf{F}}\right) \nabla_0 \mathbf{N}_I d\Omega_e = \int_{\Omega_e} \mathbf{P}\left(\bar{\mathbf{F}}\right) \mathbf{G}_{IA}^0 d\Omega_e \\ f_{Ii}^{\text{int},e} &= \int_{\Omega_e} P_{iA}\left(\bar{\mathbf{F}}\right) \frac{\partial N_I}{\partial X_A} d\Omega_e = \int_{\Omega_e} P_{iA}\left(\bar{\mathbf{F}}\right) G_{IA}^0 d\Omega_e \end{aligned} \tag{7.38}$$

7.2.1.2 Consistent Linearization and Tangent Stiffness Matrix

In the following, we derive the expression for the consistent tangent stiffness matrix $\mathbf{K}_T^{\text{int}}$ of the F-bar hexahedral element. The procedure is identical to the one presented in Section 6.4, but, because the stresses now depend on the modified deformation gradient, linearisation of the constitutive equation $\frac{d\mathbf{P}}{d\mathbf{F}}$ in (6.63) must be performed with respect to the modified deformation gradient $\bar{\mathbf{F}}$, given by (7.32):

$$\bar{\mathbf{F}} = \left(\frac{\det\left(\mathbf{F}_{\text{centroid}}\right)}{\det\left(\mathbf{F}\right)}\right)^{\frac{1}{3}} \mathbf{F} = \left(\frac{J_{\text{centroid}}}{J}\right)^{\frac{1}{3}} \mathbf{F} = \left(\frac{\bar{J}}{J}\right)^{\frac{1}{3}} \mathbf{F} \tag{7.39}$$

The directional derivative of the virtual internal work (6.63) becomes

$$D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] = \int_{\Omega_0} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 \tag{7.40}$$

The first Piola-Kirchhoff stress $\mathbf{P}$ is expressed in terms of the Cauchy stress $\boldsymbol{\sigma}$ as follows:

$$\mathbf{P}(\mathbf{F}) = \det(\mathbf{F})\,\boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right)\mathbf{F}^{-T} = \left(\frac{\det\left(\mathbf{F}_{\text{centroid}}\right)}{\det(\mathbf{F})}\right)^{-\frac{2}{3}}\bar{\mathbf{P}}\left(\bar{\mathbf{F}}\right) = \left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}}\bar{\mathbf{P}}\left(\bar{\mathbf{F}}\right) \tag{7.41}$$

where we have defined

$$\bar{\mathbf{P}}\left(\bar{\mathbf{F}}\right) = \det\left(\bar{\mathbf{F}}\right)\boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right)\bar{\mathbf{F}}^{-T} = \bar{J}\boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right)\bar{\mathbf{F}}^{-T} \tag{7.42}$$

Computation of the directional of $\mathbf{P}$, appearing in (7.40) requires the computation of $d\bar{\mathbf{P}}/d\bar{\mathbf{F}}$, $D\mathbf{F}\left[\boldsymbol{\eta}\right]$, $DJ\left[\boldsymbol{\eta}\right]$ and $D\bar{J}\left[\boldsymbol{\eta}\right]$. These developments are detailed in Appendix C.1.

Let us define the two-point tangent modulus computed from the F-bar deformation gradient

$$\mathbf{A}\left(\bar{\mathbf{F}}\right) = \frac{d\bar{\mathbf{P}}}{d\bar{\mathbf{F}}} \qquad A_{iAjB}\left(\bar{\mathbf{F}}\right) = \frac{d\bar{P}_{iA}}{d\bar{F}_{jB}} \tag{7.43}$$

The linearisation of the deformation gradient $D\mathbf{F}\left[\boldsymbol{\eta}\right]$ is given in Appendix B.4 and linearisation of the $DJ\left[\boldsymbol{\eta}\right]$ is given in Appendix B.5. Using a similar approach, we compute the linearisation of the F-bar Jacobian[2]:

$$D\bar{J}\left[\boldsymbol{\eta}\right] = \bar{J}\left(\mathbf{F}^{-T} : \nabla_{0,\text{centroid}}\boldsymbol{\eta}\right) \tag{7.44}$$

Substituting (7.43) and (7.44) into (C.6) and (C.6) into (7.40), taking account of the relations for the directional derivative of $\mathbf{F}$ (B.11) and J (B.14) as well as the properties of the tensor product (A.4), discretising the resulting expression using finite element approximation and re-arranging the terms, we obtain the directional derivative of the virtual internal work of a finite element (Please refer to Appendix C.1 for more details and intermediate equations):

$$\begin{aligned} D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = \delta\boldsymbol{u}\cdot\int_{\Omega_e}\mathbf{G}^T\,\mathbf{a}(\bar{\mathbf{F}})\,\mathbf{G}\,d\Omega_e\cdot\boldsymbol{\eta} \\ + \delta\boldsymbol{u}\cdot\int_{\Omega_e}\mathbf{G}^T\,\mathbf{q}(\bar{\mathbf{F}})\left(\mathbf{G}_{\text{centroid}} - \mathbf{G}\right)\,d\Omega_e\cdot\boldsymbol{\eta} \end{aligned} \tag{7.45}$$

[2] $\nabla_{0,\text{centroid}}$ denotes the gradient with respect to the reference configuration computed at the centroid the element as opposed to ∇_0 which is computed at a Gauss point

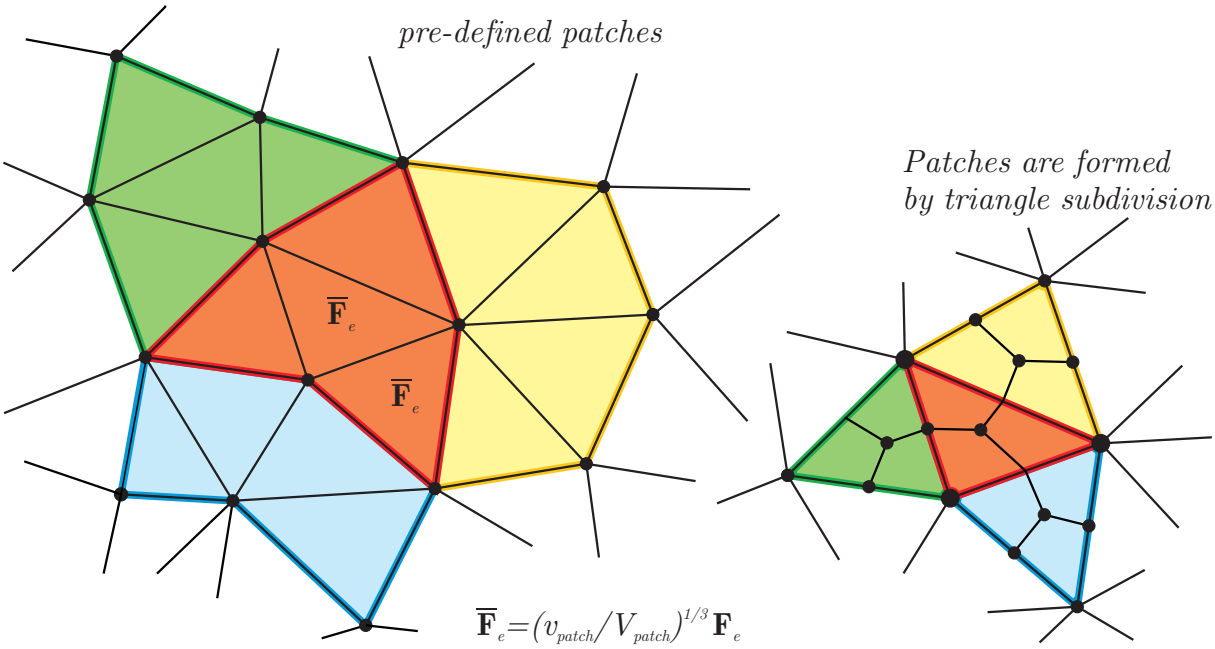


FIGURE 7.4: $\bar{\mathbf{F}}$-*patch formulation.*

with

$$a_{ijkl}\left(\bar{\mathbf{F}}\right) = \frac{1}{\bar{J}}\,\bar{F}_{kB}\,\bar{A}_{iAjB}\left(\bar{\mathbf{F}}\right)\,\bar{F}_{lA} \tag{7.46}$$

and having defined

$$\mathbf{q}(\bar{\mathbf{F}}) = \frac{1}{3}\mathbf{a}(\bar{\mathbf{F}}) : (\mathbf{I}\otimes\mathbf{I}) - \frac{2}{3}\left[\bar{\boldsymbol{\sigma}}(\bar{\mathbf{F}})\otimes\mathbf{I}\right] \tag{7.47}$$

We identify the tangent stiffness matrix for element e:

$$\mathbf{K}^{\text{int},e} = \int_{\varphi\Omega_e} \boldsymbol{G}^T\,\mathbf{a}(\bar{\mathbf{F}})\,\boldsymbol{G}\,d\Omega_e + \int_{\varphi\Omega_e} \boldsymbol{G}^T\mathbf{q}(\bar{\mathbf{F}})\left(\boldsymbol{G}_{\text{centroid}} - \boldsymbol{G}\right)d\Omega_e \tag{7.48}$$

In this expression, $\boldsymbol{G}_{\text{centroid}}$ is the discrete spatial gradient operator in the current configuration evaluated at the centroid of the element.

The first term of (7.48) is identical to the tangent stiffness of the standard element (6.88). The second term requires little additional computation effort, as the computation of discrete gradient at the element centroid $\boldsymbol{G}_{\text{centroid}}$ and the matrix $\mathbf{q}$ are quite simple and straightforward.

7.2.2 Triangular and Tetrahedral F-bar-patched Elements

The formulation presented in the previous section cannot easily be extended to low-order simplex elements because these elements produce a uniform strain, so that the deformation gradient is constant over the element and $\mathbf{F}_{\text{centroid}} = \mathbf{F}$. de Souza Neto et al. [52] overcome

this issue by applying the F-bar methodology on *patches* of triangular or tetrahedral elements. The resulting element is called the F-bar-patched linear tetrahedral element.

7.2.2.1 Definition of a Modified Deformation Gradient $\bar{\mathbf{F}}$

In the F-bar-patched methodology, the incompressibility constraints are enforced over a patch of simplex[3] elements, rather than over separate individual elements.

The mesh is first subdivided into a set of non-overlapping patches of elements $\mathscr{P}$. A unique deformation gradient is then defined for each patch $\mathscr{P}$:

$$\bar{\mathbf{F}}_e = \left(\frac{v_{\text{patch}}}{V_{\text{patch}}} \right)^{\frac{1}{3}} \mathbf{F}_e \,, \qquad \forall e \in \mathscr{P} \tag{7.49}$$

The initial and current volumes of the patch are computed by adding the volumes of the individual elements composing the patch:

$$\begin{aligned} v_{\text{patch}} &= \sum_{q \in \mathscr{P}} v_q = \sum_{q \in \mathscr{P}} V_q \det(\mathbf{F}_q) \\ V_{\text{patch}} &= \sum_{q \in \mathscr{P}} V_q \end{aligned} \tag{7.50}$$

where v_q and V_q denote the volume of the element in the current and reference configuration respectively.

The determinant of the modified deformation gradient, also called the F-bar or modified Jacobian, is the ratio of the current to the initial volume of the patch:

$$\bar{J}_e = \det \bar{\mathbf{F}}_e = \frac{v_{\text{patch}}}{V_{\text{patch}}} \tag{7.51}$$

From this equation, we understand that using the F-bar-patched deformation gradient (7.49) under incompressibility constraints results in enforcing a constant volume over predefined patches of elements. The individual elements from a patch may suffer volume change during deformation.

Similarly to the F-bar methodology for hexahedral elements (Section 7.2.1), the only difference with the conventional finite element method is that the Cauchy stresses are now computed with this modified deformation gradient:

$$\boldsymbol{\sigma} = \boldsymbol{\sigma} \left(\boldsymbol{\alpha}^n, \bar{\mathbf{F}}\, \bar{\mathbf{F}}^T \right) \tag{7.52}$$

where $\boldsymbol{\alpha}^n$ denotes the set of internal variables of the model at time t_n and $\bar{\mathbf{B}} = \bar{\mathbf{F}}\, \bar{\mathbf{F}}^T$ is the Cauchy-Green tensor computed with the F-bar deformation gradient. The notation $\boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right)$ is again used here-after to simplify the notations.

[3] triangles in the two-dimensional space and tetrahedra in the three-dimensional space.

7.2.2.2 Consistent Linearization and Tangent Stiffness Matrix

In contrast to the F-bar methodology for quadrilaterals and hexahedrons, the nodal internal forces now depend on the degrees of freedom of all elements composing the patch. Indeed, a change of volume of one element of the patch results in a modification of the deformation gradient $\bar{\mathbf{F}}$ of all elements of the patch. Therefore, stresses computed with the bar-patched deformation gradient are altered even for the elements for which the volume did not change. This results in several non-diagonal terms appearing in the global tangent stiffness matrix.

The internal tangent stiffness matrices for an element e are obtained by linearisation of the virtual work equation (6.63)

$$D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] = \int_{\Omega_0} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] : \nabla_0 \delta\boldsymbol{u}\, d\Omega_0 \tag{7.53}$$

where the nominal stress $\mathbf{P}$ depends on the F-bar-patched deformation gradient $\bar{\mathbf{F}}$ given by (7.49).

The procedure is similar to the one performed for the F-bar-hexahedral element. More specifically, $\mathbf{P}$ and $\bar{\mathbf{P}}$ are still related by (7.41) and (7.42) so that the directional derivative of $\mathbf{P}$ appearing in (7.53) should be computed via (C.6) and thus requires the computation of $d\bar{\mathbf{P}}/d\bar{\mathbf{F}}_e$, $D\mathbf{F}_e\left[\boldsymbol{\eta}\right]$, $DJ_e\left[\boldsymbol{\eta}\right]$ and $D\bar{J}_e\left[\boldsymbol{\eta}\right]$. The first three expressions are still given by (7.43), (B.11) and (B.14) respectively. The latter however, the directional derivative of the determinant of the modified deformation gradient $D\bar{J}_e\left[\boldsymbol{\eta}\right]$, is now given by

$$D\bar{J}_e\left[\boldsymbol{\eta}\right] = \frac{1}{V_{\text{patch}}} \sum_{q\in\mathscr{P}} v_q \left(\mathbf{F}_q^{-T} : \nabla_0 \boldsymbol{\eta}_q\right) \tag{7.54}$$

Introducing this into the equation for the directional derivative of $\mathbf{P}$ (Appendix C.1) and then into the equation for the directional derivative of the virtual internal work (7.53) gives, after a lengthy but straightforward calculation:

$$\begin{aligned} D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = {} & \delta\boldsymbol{u} \cdot \int_{{}^h\Omega_e} \mathbf{G}_e^T\, \mathbf{a}(\bar{\mathbf{F}})\, \mathbf{G}_e\, d\Omega_e \cdot \boldsymbol{\eta} \\ & + \delta\boldsymbol{u} \cdot \left(\frac{v_e}{v_{\text{patch}}} - 1\right) \int_{{}^h\Omega_e} \mathbf{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \mathbf{G}_e\, d\Omega_e \cdot \boldsymbol{\eta} \\ & + \delta\boldsymbol{u} \cdot \frac{1}{v_{\text{patch}}} \sum_{q\in\mathscr{P}, q\neq e} v_q \int_{{}^h\Omega_e} \mathbf{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \mathbf{G}_q\, d\Omega_e \cdot \boldsymbol{\eta} \end{aligned} \tag{7.55}$$

with $\mathbf{a}\left(\bar{\mathbf{F}}\right)$ and $\mathbf{q}\left(\bar{\mathbf{F}}\right)$ given by (7.46) and (7.47) respectively.

Consequently, consistent linearisation of the virtual work equation gives rise to the following elemental tangent stiffness matrices[4]:

$$\mathbf{K}_{ee}^{\text{int}} = \int_{^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{a}(\bar{\mathbf{F}}) \, \boldsymbol{G}_e \, d\Omega + \left(\frac{\upsilon_e}{\upsilon_{\text{patch}}} - 1 \right) \int_{^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \boldsymbol{G}_e \, d\Omega$$
$$\mathbf{K}_{eq}^{\text{int}} = \frac{\upsilon_q}{\upsilon_{\text{patch}}} \int_{^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \boldsymbol{G}_q \, d\Omega \tag{7.56}$$

Matrix $\mathbf{K}_{ee}$ has a similar form than for the F-bar element. Its rows and columns are associated with the degrees of freedom of element e only. The matrices $\mathbf{K}_{eq}$ give the influence of the nodal displacements of element $q \in \mathscr{P}; q \neq e$ on the internal force components of element e. Its rows are associated with element e and its columns are associated with element q. Both matrices are generally unsymmetric, regardless the material model adopted.

7.2.3 Discussion

The size of the patches to be defined dictates the efficiency of the approach. The more elements in the patch, the greater the constraint relaxation. However, allowing too many elements in a patch leads to an excessive relaxation of the incompressibility constraint and spurious zero-energy modes. On the other hand, too few elements in a patch leads to insufficient constraint relaxation and locking. de Souza Neto et al. [52] recommend patches of three triangular elements in a two-dimensional analysis and patches of eight tetrahedra in a three-dimensional analysis.

The major drawback of the method is that it requires the subdivision of the initial mesh into a set of non-overlapping element patches. In a two-dimensional or axisymmetric problem, the splitting of triangular mesh can be done without too much problems. However, the definition of the patches in 3D is a tedious task. The authors end up by first creating an initial tetrahedral mesh and then splitting each tetrahedral element into 8 tetrahedrons so that each tetrahedron may be labelled according to its parent tetrahedron. Of course, this spoils the whole methodology as the total number of degrees of freedom is greatly increased and can become prohibitive for real-life biomedical applications.

[4] Subscript T has been removed from the tangent stiffness matrix to clarify the notations: $\mathbf{K}_{ee}^{\text{int}}$ should be understood as $\mathbf{K}_{T,ee}^{\text{int}}$

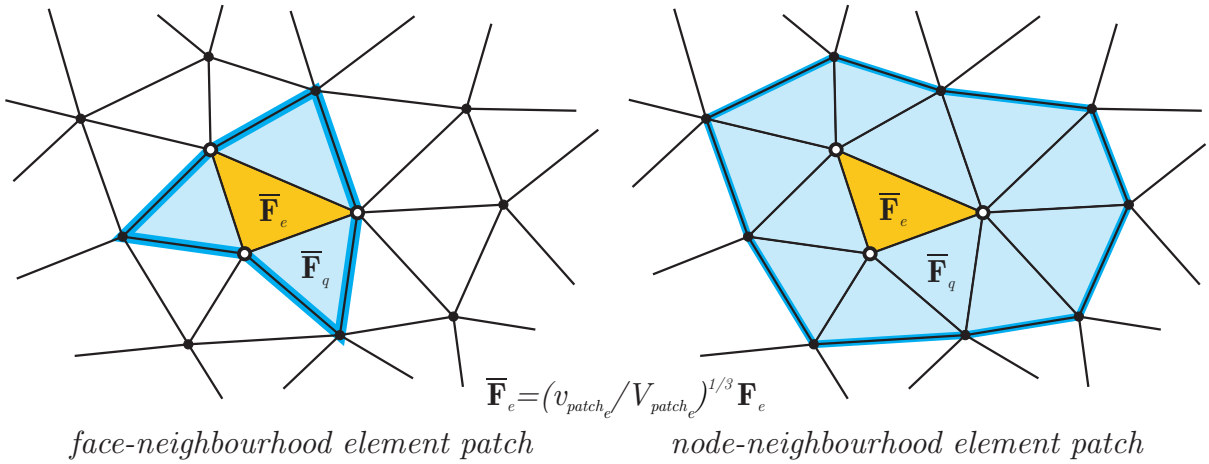


FIGURE 7.5: ***Proposed face- and node-neighbourhood patch volume change ratio tetrahedral elements formulations.***

7.3 Contribution 1: a face- or node-neighbourhood patch volume change ratio linear tetrahedron

In this section we present a face or node-neighbourhood patch volume change ratio linear tetrahedron, which constitute a first attempt of this thesis work to solve the problem of tedious patch definition in the F-bar-patch methodology (see above discussion).

7.3.1 Modified Deformation Gradient

The major drawback of the F-bar-patch tetrahedra formulation is that it requires the predefinition of non-overlapping patches (Section 7.2). The idea proposed in this section and investigated through numerical tests in the following chapter is to enforce, for each element, the incompressibility constraint over the element and its neighbours. In other words, patches elements are defined around each element in order to calculate a modified deformation gradient over this patch. However, the patches are overlapping each other so that no predefinition of the patches is required. Two types of patches will be investigated, formed either the element itself and its face-neighbours, or by the element itself and its node-neighbours. These two types of patches are presented in Figure 7.5.

The proposed modified deformation gradient is given by

$$\bar{\mathbf{F}}_e = \left(\frac{\bar{J}_e}{J_e}\right)^{\frac{1}{3}} \mathbf{F}_e \tag{7.57}$$

with

$$\bar{J}_e = \frac{v_{\text{patch}_e}}{V_{\text{patch}_e}} = \frac{\sum_{q\in\mathscr{P}_e} v_q}{\sum_{q\in\mathscr{P}_e} V_q} = \frac{\sum_{q\in\mathscr{P}_e} V_q \det\mathbf{F}_q}{\sum_{q\in\mathscr{P}_e} V_q} \tag{7.58}$$

so that a new patch is computed for every element in the mesh. We therefore use the notation v_{patch_e} and V_{patch_e}, as compared to v_{patch} and V_{patch} used for the F-bar-patch element in 7.49.

7.3.2 Consistent element tangent stiffness matrix

The internal tangent stiffness matrices for an element e are obtained by linearisation of the virtual work equation (6.63)

$$D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] = \int_{\Omega_0} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 \tag{7.59}$$

where the nominal stress now $\mathbf{P}$ depends on the F-bar-patched deformation gradient $\bar{\mathbf{F}}$ given by (7.49).

The procedure is similar to the one performed for the previous F-bar and F-bar-patch formulations 7.2.1 and 7.2.2. More specifically, $\mathbf{P}$ and $\bar{\mathbf{P}}$ are still related by (7.41) and (7.42) so that the directional derivative of $\mathbf{P}$ appearing in (7.59) should be computed via (C.6) and thus requires the computation of $d\bar{\mathbf{P}}/d\bar{\mathbf{F}}_e$, $D\mathbf{F}_e\left[\boldsymbol{\eta}\right]$, $DJ_e\left[\boldsymbol{\eta}\right]$ and $D\bar{J}_e\left[\boldsymbol{\eta}\right]$. The first three expressions are still given by (7.43), (B.11) and (B.14) respectively. The latter however, the directional derivative of the determinant of the modified deformation gradient $D\bar{J}_e\left[\boldsymbol{\eta}\right]$, is now given by

$$D\bar{J}_e\left[\boldsymbol{\eta}\right] = \frac{1}{V_{\text{patch}_e}} \sum_{q\in\mathscr{P}} V_q J_q \left(\mathbf{F}_q^{-T} : \nabla_0\boldsymbol{\eta}_q\right) \tag{7.60}$$

where the only difference with (7.54) is that the patch is now defined over each element (V_{patch_e} instead of V_{patch}).

Introducing this into the equation for the directional derivative of $\mathbf{P}$ (C.6) and then into the equation for the directional derivative of the virtual internal work (7.59) gives, after calculation:

$$\begin{aligned} D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = {} & \delta\boldsymbol{u}\cdot\int_{{}^h\Omega_e} \mathbf{G}_e^T\, \mathbf{a}(\bar{\mathbf{F}})\, \mathbf{G}_e\, d\Omega_e\cdot\boldsymbol{\eta} \\ & + \delta\boldsymbol{u}\cdot\left(\frac{v_e}{v_{\text{patch}_e}} - 1\right)\int_{{}^h\Omega_e} \mathbf{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \mathbf{G}_e\, d\Omega_e\cdot\boldsymbol{\eta} \\ & + \delta\boldsymbol{u}\cdot\frac{1}{v_{\text{patch}_e}} \sum_{q\in\mathscr{P}, q\neq e} v_q \int_{{}^h\Omega_e} \mathbf{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \mathbf{G}_q\, d\Omega_e\cdot\boldsymbol{\eta} \end{aligned} \tag{7.61}$$

with $\mathbf{a}\left(\bar{\mathbf{F}}\right)$ and $\mathbf{q}\left(\bar{\mathbf{F}}\right)$ given by (7.46) and (7.47) respectively.

Consequently, consistent linearisation of the virtual work equation gives the following elemental tangent stiffness matrices:

$$\begin{aligned}\mathbf{K}_{ee}^{\text{int}} &= \int_{^h\Omega_e} \boldsymbol{G}_e^T\, \mathbf{a}(\bar{\mathbf{F}})\, \boldsymbol{G}_e\, d\Omega + \left(\frac{\upsilon_e}{\upsilon_{\text{patch}_e}} - 1\right) \int_{^h\Omega_e} \boldsymbol{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \boldsymbol{G}_e\, d\Omega \\ \mathbf{K}_{eq}^{\text{int}} &= \frac{\upsilon_q}{\upsilon_{\text{patch}_e}} \int_{^h\Omega_e} \boldsymbol{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \boldsymbol{G}_q\, d\Omega \end{aligned} \tag{7.62}$$

Again, comparing these stiffness terms with those obtained for the F-bar-patch tetrahedron (7.56), we notice that the only difference lies in the definition of the patches.

7.3.3 Two-dimensional case

For a plain strain problem, the deformation gradient is modified as follows:

$$\bar{\mathbf{F}}_e = \left[\begin{array}{c|c} \bar{\mathbf{F}}_{e,\text{plane}} & \begin{matrix}0 & 0\end{matrix} \\ \hline \begin{matrix}0 & 0\end{matrix} & 1 \end{array}\right] \tag{7.63}$$

with

$$\bar{\mathbf{F}}_{e,\text{plane}} = \left(\frac{\bar{J}_e}{J_e}\right)^{\frac{1}{2}} \mathbf{F}_{e,\text{plane}} \tag{7.64}$$

and $\bar{J}_e$ stiff being given by (7.58):

$$\bar{J}_e = \frac{\upsilon_{\text{patch}_e}}{V_{\text{patch}_e}} = \frac{\sum_{q\in\mathscr{P}_e} \upsilon_q}{\sum_{q\in\mathscr{P}_e} V_q} = \frac{\sum_{q\in\mathscr{P}_e} V_q \det\mathbf{F}_q}{\sum_{q\in\mathscr{P}_e} V_q} \tag{7.65}$$

The stiffness terms for the plane-strain case are still given by (7.62) but, the $\mathbf{q}$ of (7.62) is now computed by:

$$\mathbf{q}(\bar{\mathbf{F}}) = \frac{1}{2}\mathbf{a}(\bar{\mathbf{F}}) : (\mathbf{I}\otimes\mathbf{I}) - \frac{1}{2}\left[\bar{\boldsymbol{\sigma}}(\bar{\mathbf{F}})\otimes\mathbf{I}\right] \tag{7.66}$$

7.3.4 Border elements and multi-material meshes

A special treatment is adopted for the elements lying on the border of the domain. In our algorithm, these elements are detected by computing the number of face neighbours n_e of the tetrahedron. An element lies on the border of the domain if this number is less than 4, $n_e < 4$, whereas interior elements have four direct neighbours, $n_e = 4$.

Similarly, for meshes that are constituted of several material regions, the nodes that are located on the interface of material regions are considered as *border elements*.

On border elements, the modified Jacobian is defined as

$$\bar{J}_e = \alpha \, \frac{\upsilon_{\text{patch}_e}}{V_{\text{patch}_e}} + (1-\alpha)\, J_e \tag{7.67}$$

with $\alpha = n_e/4$. For interior elements, $\alpha = 1$, this equation is equivalent to (7.58). For an isolated element, $\alpha = 0$, and the traditional formulation is recovered.

Applying the same linearisation procedure as above, we obtain following element contributions to the consistent tangent stiffness matrix:

$$\begin{aligned}
\mathbf{K}_{ee} &= \int_{{}^h\Omega_e} \boldsymbol{G}^T \mathbf{a}(\bar{\mathbf{F}}_e) \boldsymbol{G} \, d\Omega + \left(\left(\frac{\alpha}{\upsilon_{\text{patch}_e}} + \frac{1-\alpha}{\bar{J}_e V_e} \right) \upsilon_e - 1 \right) \int_{{}^h\Omega_e} \boldsymbol{G}^T \mathbf{q}(\bar{\mathbf{F}}_e) \boldsymbol{G} \, d\Omega \\
\mathbf{K}_{eq} &= \alpha \frac{\upsilon_q}{\upsilon_{\text{patch}_e}} \int_{{}^h\Omega_e} \boldsymbol{G}^T \mathbf{q}(\bar{\mathbf{F}}_e) \boldsymbol{G}_q \, d\Omega
\end{aligned} \tag{7.68}$$

which is equivalent to (7.62) for interior elements characterised by $\alpha = 1$.

7.3.5 Discussion

The definition of the element patches in the proposed formulation is much simpler than in the F-bar-patched formulation (Section 7.2). Indeed, in our formulation, patches are simply formed by an elements and its face-neighbours. Numerical applications in Chapter 8 will investigate whether or not this formulation is effective in removing the locking of the standard linear tetrahedron, observed under incompressibility constraints.

7.4 Contribution 2: an Average Elemental Jacobian (AEJ) tetrahedral element

Andrade Pires et al. [5] have proposed an implicit version of the average nodal pressure (ANP) triangular element initially proposed by Bonet and Burton [25] and presented in Section 7.1.1. To obtain the expression of the consistent tangent stiffness matrix needed, the authors re-cast the original concept the average nodal *pressure* element in terms of an average *volume change ratio* within the framework of the F-bar method (Section 7.2). The idea is to average nodally defined Jacobians over the element to obtain a modified elemental Jacobian. In this way, Andrade Pires et al. [5] obtained a linear triangle for

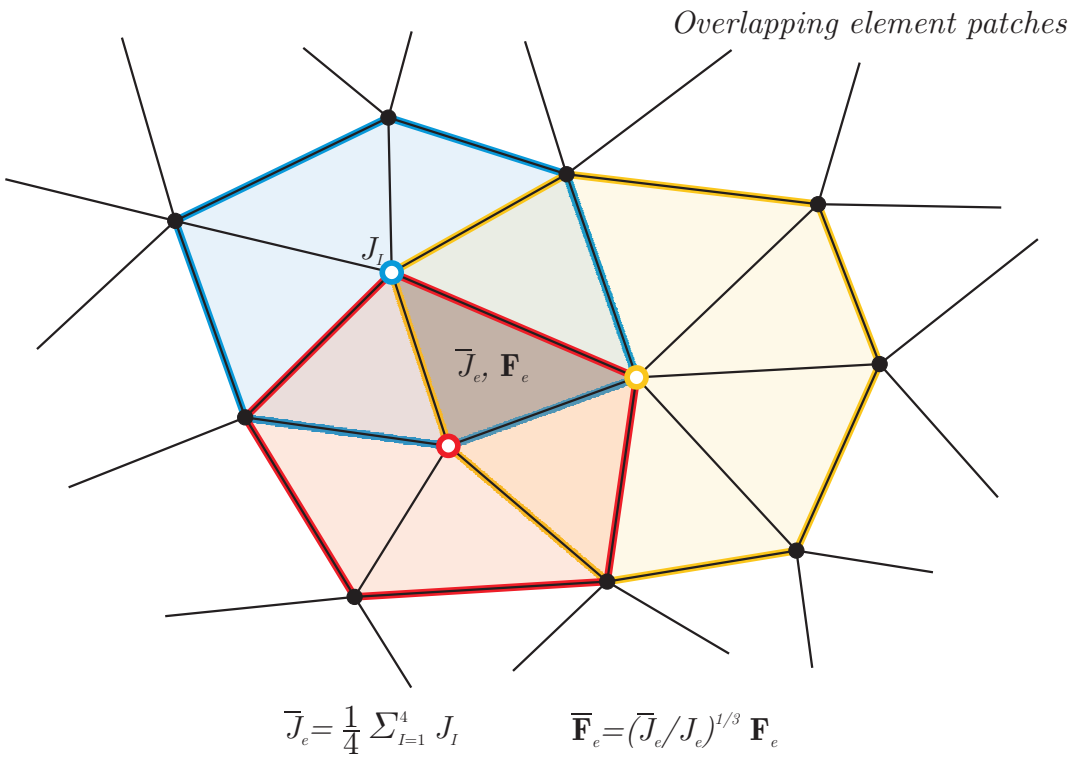


FIGURE 7.6: ***Proposed Average Elemental Jacobian (AEJ) tetrahedral element formulation.***

implicit plane strain and axisymmetric analysis of nearly incompressible solids under finite strains. In this section, we extend the concept to the third dimension and obtain a *three-dimensional* implicit version of average nodal pressure tetrahedral element. We call it the Average Elemental Jacobian (AEJ) tetrahedral element.

7.4.1 Definition of a modified deformation gradient $\bar{\mathbf{F}}$

As proposed by Bonet and Burton [25], we define nodal volumes at each mesh node by summing the contributions of the tetrahedral elements sharing node I (Equations (7.1) and (7.2)):

$$\begin{aligned} V_I &= \sum_{q \in \mathscr{P}_I} \frac{1}{4} V_q \\ v_I &= \sum_{q \in \mathscr{P}_I} \frac{1}{4} v_q = \sum_{q \in \mathscr{P}_I} \frac{1}{4} V_q \det \mathbf{F}_q \end{aligned} \tag{7.69}$$

where $\mathscr{P}_I$ is the patch of elements attached to node I.

The nodal volume ratio is then defined as (7.3):

$$J_I = \frac{v_I}{V_I} = \frac{\sum_{q\in\mathscr{P}_I} v_q}{\sum_{q\in\mathscr{P}_I} V_q} = \frac{\sum_{q\in\mathscr{P}_I} V_q \, \det\mathbf{F}_q}{\sum_{q\in\mathscr{P}_I} V_q} \tag{7.70}$$

Let us now define an average element volume ratio $\bar{J}_e$ for the tetrahedron by averaging the four nodal volume ratios J_I of the tetrahedron:

$$\bar{J}_e = \frac{1}{4}\sum_{I=1}^{4} J_I \tag{7.71}$$

where the sum $\sum_I^4$ represents the sum over the four nodes of the tetrahedral element e.

The idea is to enforce the incompressibility constraint by imposing this average volume ratio $\bar{J}_e$ to remain constant. To achieve this, a modified deformation gradient $\bar{\mathbf{F}}_e$ is defined by scaling the true deformation gradient $\mathbf{F}_e$ so that its determinant becomes equal to $\bar{J}_e$, as has been done in the previous F-bar approaches (Sections 7.2.1,7.2.2 and 7.3):

$$\bar{\mathbf{F}}_e = \left(\frac{\bar{J}_e}{J_e}\right)^{\frac{1}{3}} \mathbf{F}_e \tag{7.72}$$

During the finite element simulation, the standard deformation gradient $\mathbf{F}$ is replaced by the modified deformation gradient $\bar{\mathbf{F}}$ for the computation of the stresses at the Gauss points[5].

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}\left(\boldsymbol{\alpha}^n, \bar{\mathbf{F}}\,\bar{\mathbf{F}}^T\right) \tag{7.73}$$

where $\boldsymbol{\alpha}^n$ denotes the set of internal variables of the model at time t_n and $\bar{\mathbf{B}} = \bar{\mathbf{F}}\,\bar{\mathbf{F}}^T$ is the Cauchy-Green tensor computed with the F-bar deformation gradient.

7.4.1.1 Consistent linearisation and tangent stiffness matrices

In the following, we derive the element contributions to the global consistent tangent stiffness matrix $\mathbf{K}_T$ by performing the Newton-Raphson linearisation of the virtual work equation. This linearisation procedure has been detailed in Section 6.4 for the conventional finite element formulation and in Sections 7.2.1.2 and 7.2.2.2 for the F-bar hexahedral element and the F-bar-patched tetrahedral element respectively.

Linearisation of the strain energy gives, for one element,

$$D\delta\mathcal{W}^{\text{int},e}\left[\boldsymbol{\eta}\right] = \int_{\Omega_0} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 \tag{7.74}$$

[5]The notation $\boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right)$ here-after to simplify the notations

where the nominal stresses $\mathbf{P}$ depend on the F-bar-patched deformation gradient $\bar{\mathbf{F}}$ given by (7.72).

Hence, linearisation of the element virtual internal work requires computation of the directional derivative of $\mathbf{P}$, (7.41)

$$D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] = D\left\{\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}}\bar{\mathbf{P}}\left(\bar{\mathbf{F}}\right)\right\}\left[\boldsymbol{\eta}\right] \tag{7.75}$$

with $\bar{\mathbf{P}}$ given by (7.42)

$$\bar{\mathbf{P}}\left(\bar{\mathbf{F}}\right) = \det\left(\bar{\mathbf{F}}\right)\boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right)\bar{\mathbf{F}}^{-T} = \bar{J}\boldsymbol{\sigma}\left(\bar{\mathbf{F}}\right)\bar{\mathbf{F}}^{-T} \tag{7.76}$$

We obtain (C.6)

$$\begin{aligned} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] = &-\frac{2}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{5}{3}}\frac{1}{J^2}\left[J\,D\bar{J}\left[\boldsymbol{\eta}\right] - \bar{J}\,DJ\left[\boldsymbol{\eta}\right]\right]\bar{\mathbf{P}} \\ &+\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}}\frac{d\bar{\mathbf{P}}}{d\bar{\mathbf{F}}} : \left[\frac{1}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}}\frac{1}{J^2}\left[J\,D\bar{J}\left[\boldsymbol{\eta}\right] - \bar{J}\,DJ\left[\boldsymbol{\eta}\right]\right]\mathbf{F} + \left(\frac{\bar{J}}{J}\right)^{\frac{1}{3}}D\mathbf{F}\left[\boldsymbol{\eta}\right]\right] \end{aligned} \tag{7.77}$$

The latter requires the computation of $d\bar{\mathbf{P}}/d\bar{\mathbf{F}}_e$, $D\mathbf{F}_e\left[\boldsymbol{\eta}\right]$, $DJ_e\left[\boldsymbol{\eta}\right]$ and $D\bar{J}_e\left[\boldsymbol{\eta}\right]$. The first three expressions are identical to the previous F-bar approaches and are therefore given by (7.43), (B.11) and (B.14) respectively. The directional derivative of the determinant of the modified deformation gradient $D\bar{J}_e\left[\boldsymbol{\eta}\right]$ must be computed for the current formulation (7.71). With the help of (B.14), we obtain

$$D\bar{J}_e\left[\boldsymbol{\eta}\right] = \frac{1}{4}\sum_{I}^{4}\left\{\frac{1}{\sum_{q\in\mathscr{P}_I}V_q}\sum_{q\in\mathscr{P}_I}V_q J_q\left(\mathbf{F}_q^{-T} : \nabla_0\boldsymbol{\eta}_q\right)\right\} \tag{7.78}$$

Following steps are similar to the previous explained F-bar formulations. The directional derivative of the element Jacobian (7.78) and the other three derivatives (7.43), (B.11) and (B.14) are inserted into the linearised nominal stress (7.77) and the result $D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right]$ is then substituted into the linearised virtual strain energy (7.74). The obtained expression is

eventually transformed to the spatial configuration. After calculations this gives,

$$
\begin{aligned}
D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = & \int_{\Omega_e} \left[\boldsymbol{a}(\bar{\mathbf{F}}) : \nabla\boldsymbol{\eta}_e\right] : \nabla\delta\boldsymbol{u}_e \, d\Omega_e \\
& - \int_{\Omega_e} \left[\mathbf{q}(\bar{\mathbf{F}}) : \nabla\boldsymbol{\eta}_e\right] : \nabla\delta\boldsymbol{u}_e \, d\Omega_e \\
& + \int_{\Omega_e} \left[\mathbf{q}(\bar{\mathbf{F}}) : \left(\frac{1}{4\bar{J}_e}\sum_{I=1}^{4}\frac{1}{4V_I}\sum_{q\in\mathscr{P}_I} v_q \, \nabla\boldsymbol{\eta}_q\right)\right] : \nabla\delta\boldsymbol{u}_e \, d\Omega_e
\end{aligned}
\tag{7.79}
$$

Discretising this expression using the finite element method and isolating $q = e$ in the third term we obtain,

$$
\begin{aligned}
D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = & \, \delta\boldsymbol{u}_e \cdot \int_{{}^h\Omega_e} \mathbf{G}_e^T \, \mathbf{a}(\bar{\mathbf{F}}) \, \mathbf{G}_e \, d\Omega_e \cdot \boldsymbol{\eta}_e \\
& + \delta\boldsymbol{u}_e \cdot \left(\frac{1}{4\bar{J}_e}\sum_{I=1}^{4}\frac{v_e}{4V_I} - 1\right) \int_{{}^h\Omega_e} \mathbf{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \mathbf{G}_e \, d\Omega_e \cdot \boldsymbol{\eta}_e \\
& + \delta\boldsymbol{u}_e \cdot \left(\frac{1}{4\bar{J}_e}\sum_{I=1,\, q\in\mathscr{P}_I}^{4}\frac{v_q}{4V_I}\right) \int_{{}^h\Omega_e} \mathbf{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \mathbf{G}_q \, d\Omega_e \cdot \boldsymbol{\eta}_q
\end{aligned}
\tag{7.80}
$$

We identify the following tangent stiffness matrices

$$
\begin{aligned}
\mathbf{K}_{ee}^{\text{int}} &= \int_{{}^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{a}(\bar{\mathbf{F}}) \, \boldsymbol{G}_e \, d\Omega + \left(\frac{1}{4\bar{J}_e}\sum_{I=1}^{4}\frac{v_e}{4V_I} - 1\right) \int_{{}^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \boldsymbol{G}_e \, d\Omega \\
\mathbf{K}_{eq}^{\text{int}} &= \left(\frac{1}{4\bar{J}_e}\sum_{I=1,\, q\in\mathscr{P}_I}^{4}\frac{v_q}{4V_I}\right) \int_{{}^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \boldsymbol{G}_q \, d\Omega
\end{aligned}
\tag{7.81}
$$

with

$$
\mathbf{q}(\bar{\mathbf{F}}) = \frac{1}{3}\mathbf{a}(\bar{\mathbf{F}}) : (\mathbf{I}\otimes\mathbf{I}) - \frac{2}{3}\left[\bar{\boldsymbol{\sigma}}(\bar{\mathbf{F}})\otimes\mathbf{I}\right] \tag{7.82}
$$

7.4.2 Two-dimensional case

In two-dimensions, the formula gives:

$$
\begin{aligned}
\mathbf{K}_{ee}^{\text{int}} &= \int_{{}^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{a}(\bar{\mathbf{F}}) \, \boldsymbol{G}_e \, d\Omega + \left(\frac{1}{3\bar{J}_e}\sum_{I=1}^{3}\frac{v_e}{3V_I} - 1\right) \int_{{}^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \boldsymbol{G}_e \, d\Omega \\
\mathbf{K}_{eq}^{\text{int}} &= \left(\frac{1}{3\bar{J}_e}\sum_{I=1,\, q\in\mathscr{P}_I}^{3}\frac{v_q}{3V_I}\right) \int_{{}^h\Omega_e} \boldsymbol{G}_e^T \, \mathbf{q}(\bar{\mathbf{F}}) \, \boldsymbol{G}_q \, d\Omega
\end{aligned}
\tag{7.83}
$$

which is different than the stiffness terms of the original 2D formulation proposed by Andrade Pires et al. [5] on which this section was based.

7.4.3 Border elements and multi-material meshes

It is not clear how the nodal volume ratio J_I (7.70) and henceforth, the average element volume ratio $\bar{J}_e$ (7.71) for nodes lying on the border of the meshed domain, or at the interface between several material regions. It is however extremely important to take these *border* elements into account in our formulation as significant differences in the results of the finite element simulations may be observed depending on the modified deformation gradient defined for these border elements (results from 1 to 2 have been observed on classic benchmarks). No details on the best treatment to be adopted for border elements have been proposed in the original, 2D only, implementation of the average nodal volume change ratio triangle, and our stiffness terms being different anyway, this section results from extensive use and testing of the present formulation.

In the end, the formulation that has been proven to give the best results for our Average Elemental Jacobian tetrahedral element is:

First compute the nodal volume change ratio for the nodes for which the whole neighbourhood is formed. When all four tetrahedron nodes are located on a border, the standard formulation is used. Otherwise, compute the average volume change ratio as

$$\bar{J}_e = \sum_{I=1, I\notin\delta\Omega} J_I \tag{7.84}$$

where the sum is over the tetrahedron nodes that are not located on the border of the meshed domain.

To compute the terms of the stiffness matrix, the sum in (7.82) is also taken over the $n_I \leq 4$ element's interior nodes only. Resulting in the following expressions for $\mathbf{K}^{\text{int}}_{ee}$ and $\mathbf{K}^{\text{int}}_{eq}$:

$$\begin{aligned}
\mathbf{K}^{\text{int}}_{ee} &= \int_{^h\Omega_e} \boldsymbol{G}_e^T\, \mathbf{a}(\bar{\mathbf{F}})\, \boldsymbol{G}_e\, d\Omega + \left(\frac{1}{n_I\, \bar{J}_e} \sum_{I=1, I\notin\delta\Omega}^{n_I} \frac{v_e}{n_I\, V_I} - 1 \right) \int_{^h\Omega_e} \boldsymbol{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \boldsymbol{G}_e\, d\Omega \\
\mathbf{K}^{\text{int}}_{eq} &= \left(\frac{1}{n_I\, \bar{J}_e} \sum_{I=1, I\notin\delta\Omega,\, q\in\mathscr{P}_I}^{n_I} \frac{v_q}{n_I\, V_I} \right) \int_{^h\Omega_e} \boldsymbol{G}_e^T\, \mathbf{q}(\bar{\mathbf{F}})\, \boldsymbol{G}_q\, d\Omega
\end{aligned} \tag{7.85}$$

7.5 Conclusions

In this chapter, unlocking solutions for the linear tetrahedron were presented.

In Section 7.1 popular nodal-based formulations were reviewed: the average nodal pressure (ANP) linear tetrahedron proposed by Bonet and Burton [25] and the average nodal strain linear tetrahedron proposed by Dohrmann et al. [55]. In nodal-based formulations,

the incompressibility constraints are enforced on newly defined nodal volumes instead of on each element. These formulations are proposed in the context of explicit finite element simulations were a lumped mass matrix is used.

In Section 7.2 the F-bar methodology for quadrilateral and hexahedral elements, and its extensions to triangular and tetrahedral elements, the F-bar-patched method, are presented. The idea is to define a modified deformation gradient $\bar{\mathbf{F}}$ over the element, which is used to compute the stresses in the traditional way. These methods are suitable for implicit finite element analysis and an expression for the stiffness terms of the tangent stiffness matrix is proposed. Even though the idea is interesting, the F-bar-patch tetrahedron proposed by de Souza Neto et al. [52] is useless in practice because it requires the definition of non-overlapping patches of tetrahedral elements, for which no automatic algorithm is yet available.

In Section 7.3 and Section 7.4 two successive ideas to remove the locking of the standard linear tetrahedron, valid for explicit and implicit finite element analysis, are presented. The first proposal is a F-bar-patch tetrahedron in which, for each element, the incompressibility constraints are enforced over the element itself and its neighbours. Both the element's node and the face-neighbourhood are investigated. In the second proposal, a nodal Jacobian is defined at the element's node as the ratio between current and initial nodal volumes; the definition of nodal volumes being identical to nodal-based formulations. A modified element Jacobian is then defined by averaging the nodal Jacobians. This modified Jacobian is used to define the modified deformation gradient of the F-bar methodologies. The terms of the internal tangent stiffness matrix are obtained by linearisation of the internal virtual work equation, followed by finite element discretisation. The two-dimensional and the multi-material case were also investigated.

Both formulations were implemented in the finite element code Metafor. In the next chapter, finite element simulations will be performed using these new elements, in order to determine their efficiency in removing the incompressibility and shear locking which occurs with the standard linear tetrahedron.

Chapter 8

Numerical applications

This chapter investigates the unlocking performance of the two proposed element formulations, the face- and node- neighbourhood patch volume change ratio linear tetrahedral elements, called *f-patchJ tet* and *n-patchJ tet*, as well as, the Average Elemental Jacobian (AEJ) tetrahedron, *AEJ* . Even though these elements have been constructed to remove volumetric locking only, their capability to remove shear locking will also be investigated.

A major advantage of the proposed formulations is that they can be used for any material laws without additional implementation efforts. This is illustrated in this chapter through the use of several constitutive equations: compressible and incompressible linear elasticity, neo-Hookean (large deformation) and elasto-plastic with Von Mises plasticity. Also, the elements can be used in explicit and implicit problems, as will also be shown hereafter.

The numerical applications considered in this chapter are classical benchmark tests from the literature so that the performance of our new finite element formulations will be tested against the most popular unlocking solutions in literature. Table 8.1 presents all the element formulations that will be investigated in this chapter.

8.1 Cook's membrane

The Cook's membrane is frequently used to assess the convergence properties of finite elements near the incompressibility limit, under combined shear and bending strains [35, 51, 52, 112, 159]. Both the two-dimensional plane-strain and the three-dimensional Cook's membrane are investigated in this work. The geometry of the membrane is given in Figure 8.1. The left vertical edge is clamped and a distributed shearing load is applied to

TABLE 8.1: ***Finite elements used in this chapter.*** *The proposed elements are indicated in bold.*

2D triangular elements		Reference
T1	standard, linear	
T2	standard, quadratic	
f-patchJ tri	face-neighbours-patch volume change ratio, linear	Section 7.3.3
n-patchJ tri	node-neighbours-patch volume change ratio, linear	Section 7.3.3
AEJ tet	Average Elemental Jacobian , linear	Section 7.4.2
F-bar-patch tri	modified deformation gradient or F-bar, linear	[52]
AndradePires2004	average nodal volume formulation, linear	[5]
3D tetrahedral elements		
T1	standard, linear	
T2	standard, quadratic	
f-patchJ tet	face-neighbours-patch volume change ratio, linear	Section 7.3
n-patchJ tet	node-neighbours-patch volume change ratio, linear	Section 7.4
AEJ tet	Average Elemental Jacobian , linear	Section 7.4
Puso	stabilised nodally integrated, linear	[147]
Dohrmann	nodal-based uniform strain, linear	[55]
Klaas	linear u, linear p, stabilised mixed	[96]
T1P1ES3ST	linear u, linear p, area bubble, enhanced strains with stab.	[112, 113]
T1P1ES12ST	linear u, linear p, volume bubble, enhanced strains with stab.	[112, 113]
2D quadrilateral elements		
STD quad	standard, linear	
SRI quad	selective reduced integration, linear	Metafor [111]
F-bar quad	modified deformation gradient or F-bar, linear	[51]
CP4R	reduced integration and hourglass control, linear	[114]
3D hexahedral elements		
STD hex	standard, linear	
SRI hex	selective reduced integration, linear	Metafor [111]
EAS hex	enhanced assumed strains, linear	Metafor [111]

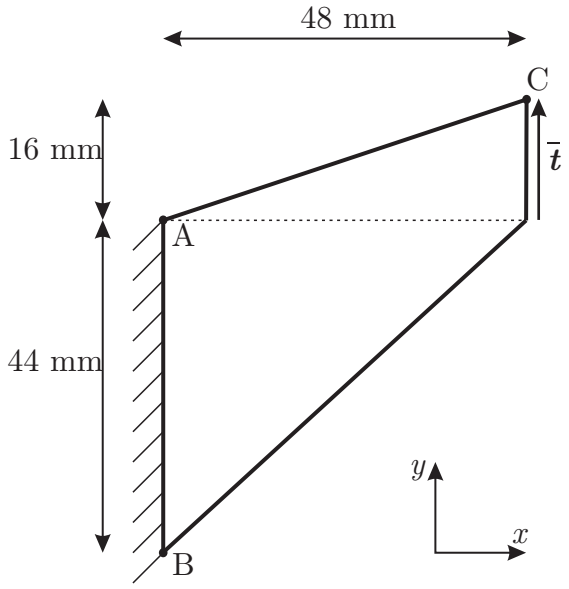


FIGURE 8.1: ***2D Cook's membrane.*** *Geometry and Loading.*

the opposite edge. The latter is applied in the plane of the element facets and follows the orientation of the facets throughout the deformation. Several material properties, element formulations and mesh sizes will be investigated hereinafter.

8.1.1 Two-dimensional case

The plane-strain Cook's membrane example has been used as benchmark to assess the convergence properties of enhanced element formulations [5, 51, 52, 99, 159], including by Simo and Armero [159] for their enhanced assumed strain element, de Souza Neto et al. [51] for their modified deformation gradient $\bar{\mathbf{F}}$ quadrilateral element and by de Souza Neto et al. [52] for their $\bar{\mathbf{F}}$-based triangle.

As has been done in the literature, a regularized neo-Hookean material with shear modulus $\mu = 80.1938$ MPa and bulk modulus $k = 40.0942$ x 10^4 MPa is adopted. Corresponding values for the Young's modulus and Poisson's ratio are $E = 240.5654$ MPa and $\nu = 0.4999$. Note that near incompressibility is achieved for this value of the Poisson's ratio. A distributed shearing load of $\bar{\boldsymbol{t}} = 6.25$ N/mm is applied on the right vertical edge of the specimen, which makes a total resultant shearing force of $\boldsymbol{f} = 100$ MPa/mm. This load is applied incrementally within an implicit time integration scheme.

Also, several mesh sizes are considered. These meshes were obtained by first constructing quadrilateral meshes of $2\times2, 3\times3, 5\times5, 8\times8, 16\times16$ and 32×32 elements and then subdividing each quadrilateral into two.

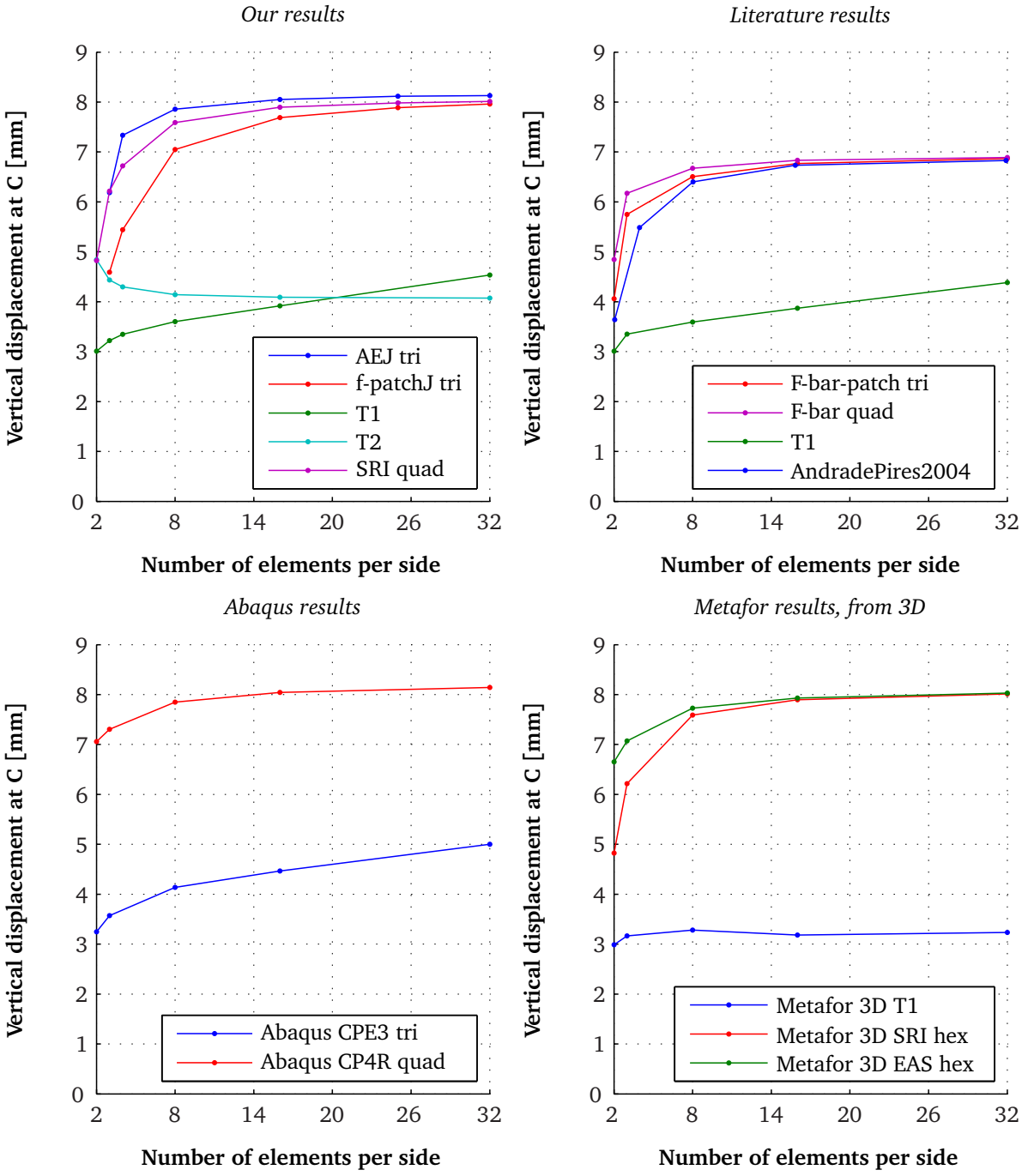


FIGURE 8.2: ***Plane-strain Cook's membrane.*** *Convergence of the solution with mesh refinement. (a) Comparison of the results obtained with the proposed elements, AEJ and patchJ tri with other 2D elements of Metafor. (b) popular elements from literature [5, 52]. (c) popular Abaqus elements. (d) 3D elements of Metafor, using a 2D-equivalent model of the Cook's membrane.*

TABLE 8.2: ***Plane strain Cook's membrane.*** *Vertical displacement at point C obtained for various finite element formulations and for the finer mesh:* 32 *elements per side, giving* 32×32 *quadrilateral elements and* $32 \times 32 \times 2$ *triangular elements and* 33×33 *nodes.*

Element type	**Source**	u_y **[mm]**
classical 2D triangular elements		
T1	Metafor 2D	4.5329
T1	de Souza Neto et al. [52]	4.38168
T1 (CPE3)	Abaqus	5.00295
T2	Metafor 2D	4.0725
unlocking 2D triangular elements		
F-bar-patch tri	de Souza Neto et al. [52]	6.8670
AndradePires2004	Andrade Pires et al. [5]	6.8292
f-patchJ tri	Metafor 2D	**7.9624**
AEJ	Metafor 2D	**8.1276**
2D quadrilateral elements		
F-bar-quad	de Souza Neto et al. [52]	6.8915
SRI quad	Metafor	8.0144
CP4R	Abaqus	8.14209
3D elements		
T1	Metafor 3D, 2D-equivalent test	3.23223
SRI hex	Metafor 3D, 2D-equivalent test	8.01448
EAS hex	Metafor 3D, 2D-equivalent test, equivalent linear elastic law	8.03002

Figure 8.2 shows the final vertical displacement obtained at the upper right corner of the panel (point C in Figure 8.1) for several discretisations.

Figure 8.2 (a) shows the convergence of the two proposed elements, *f-patchJ tri* and *AEJ tri* implemented in the finite element software Metafor [111]. Two other finite elements of Metafor are represented for comparison: the standard linear triangle, *T1*, and the selective reduced integrated quadrilateral, called *SRI quad* in this work.

Figure 8.2 (b) is a copy of the results presented by de Souza Neto et al. [52] for the standard linear triangle, the *F-bar-patch* triangle and the *F-bar* quadrilateral.

Comparing Figure 8.2 (a) and (b), we verify that similar displacement values are obtained for the standard linear tetrahedron (T1), meaning that results produced by our in-house code Metafor are similar to those obtained by de Souza Neto et al. [52] when using the same finite element. This guarantees that our model is identical to the one used by de Souza Neto et al. [52].

Interestingly, the obtained tip displacements are higher for the proposed formulations *AEJ tri* and *f-patch tri* than for the *F-bar-patch tri* and *F-bar quad*. This means that our formulations remove the locking behaviour of *T1* better than these *F-bar* formulations, at least in the two-dimensional case. The obtained curves are close to those obtained for the selective reduced integrated quadrilateral, *quad SRI*, which has been proven to be effective in removing the volumetric locking [145] and has been extensively used in Metafor [111].

Furthermore, our Average Elemental Jacobian triangular element give results that are significantly better than the algorithm it was inspired from, i.e. the *AEJ* in Figure 8.2, Upper Right, converges towards a limit of 8.13 mm whereas the curve of Andrade Pires et al. [5], represented in Figure 8.2 (b) tends towards the lower value of 6.83 mm.

The lower graphs of Figure 8.2 permits a second check of the good behaviour of the proposed finite elements. The graph on the left represents the convergence of two common 2D elements of Abaqus[1], the 3-node linear triangle CPE3 and the 4-node linear quadrilateral with reduced integration and hourglass control (CP4R). The convergence of standard linear triangle has already been presented for two other implementations of the element: Metafor (upper left graph) and literature (upper right graph). In all three cases the standard linear triangle exhibits volumetric and shear locking. The default plane-strain quadrilateral of Abaqus, CP4R, presents however a very good behaviour under incompressibility constraints with a flexibility of the membrane that is similar to the one obtained with the *SRI quad* of Metafor and our new *AEJ* triangular formulation (see upper left graph).

[1]To build the equivalent model in Abaqus, the constants $C10 = \mu/2 = 40.0969$ MPa and $D1 = 2/k = 4.98825$ MPa^{-1}of the Neo-Hookean law were defined.

The last graph, Figure 8.2 (d) depicts results obtained using a 3D model of the Cook's membrane, but restricting the membrane to deform in its plane, so that the 3D problem is equivalent to the 2D case presented above. Figure 8.5 depicts the boundary conditions used. This dodge allows us to compare the proposed 2D unlocking formulations against the well-established 3D non-locking hexahedral elements of Metafor: with selective reduced integration and with the hexahedron with enhanced assumed strain. The SRI hexahedron has been proven to be effective in removing volumetric locking and the enhanced assumed strain hexahedron, *EAS hex*, is designed to remove both volumetric and shearing locking [32]. The latter was not designed to work with hyperelastic material laws. Consequently, the equivalent linear elastic material law was used in that case. The graph of Figure 8.2 (d) indicates that both hexahedral elements converge towards the same limit of approximately 8 mm for fine meshes, with a faster convergence rate for the EAS hexahedron. The third curve on the graph refers to the convergence of the vertical displacement at point C for the Cook's membrane meshed with the standard linear tetrahedron. As expected, we observe a very stiff behaviour in that case.

Table 8.2 gives the final displacement values obtained for the various elements presented above. The numerical values correspond to the vertical displacement at point C obtained for the finer mesh of 32 elements per side (33×33 nodes). These values indicate that the CP4R hexahedral element of Abaqus is the most flexible, closely followed by the proposed Average Elemental Jacobian (AEJ) triangle.

Figure 8.3 depicts the stress field obtained for the proposed formulations as well as for the standard linear triangle and the selective reduced integrated quadrilateral of Metafor. Minimum and maximum values of the Von Mises stress are indicated on the corresponding location on the membrane. Because these extrema are larger for the Average Elemental Jacobian (AEJ) tetrahedron than for the face-neighbourhood-patch nodal volume ratio, *f-patchJ tri*, we may deduce that the second formulation has a smoothing effect on the stress field.

Figure 8.4 compares the pressure field obtained for the proposed *AEJ* element and the pressure field obtained by [5] for their F-bar-based average nodal volume change ratio element. Even though the proposed *AEJ* formulation is based on Andrade Pires et al. [5], different pressure fields are observed. Whereas [5] observed a checkerboard pattern, our *AEJ* formulation provides a realistic pressure distribution.

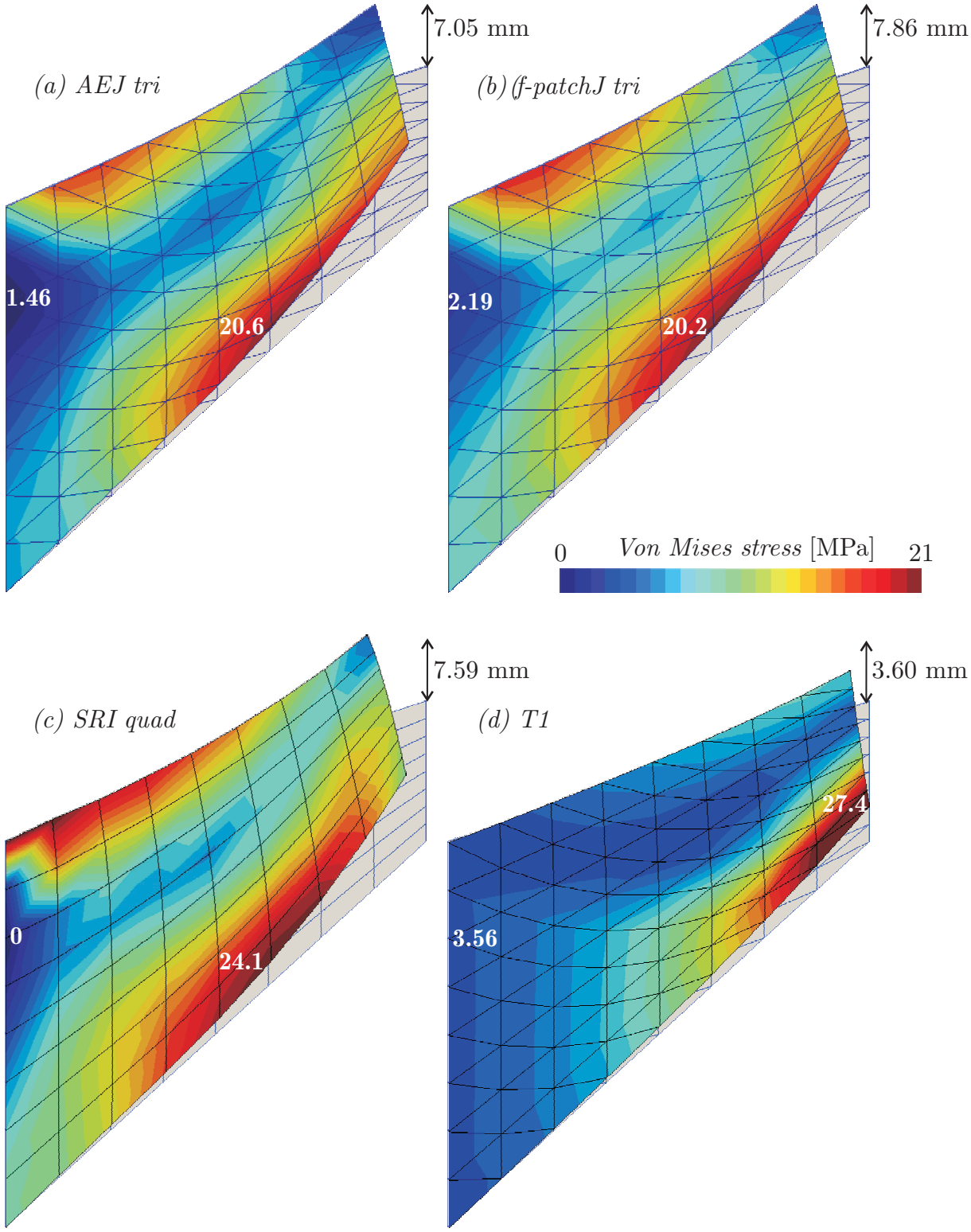


FIGURE 8.3: ***Plane strain Cook's membrane.*** *Von Mises stress fields for (a) AEJ , (b) f-patchJ tri, (c) SRI quad, (d) T1. The displacement and stress values indicated on the picture were obtained with the depicted mesh resolution (8 elements per side).*

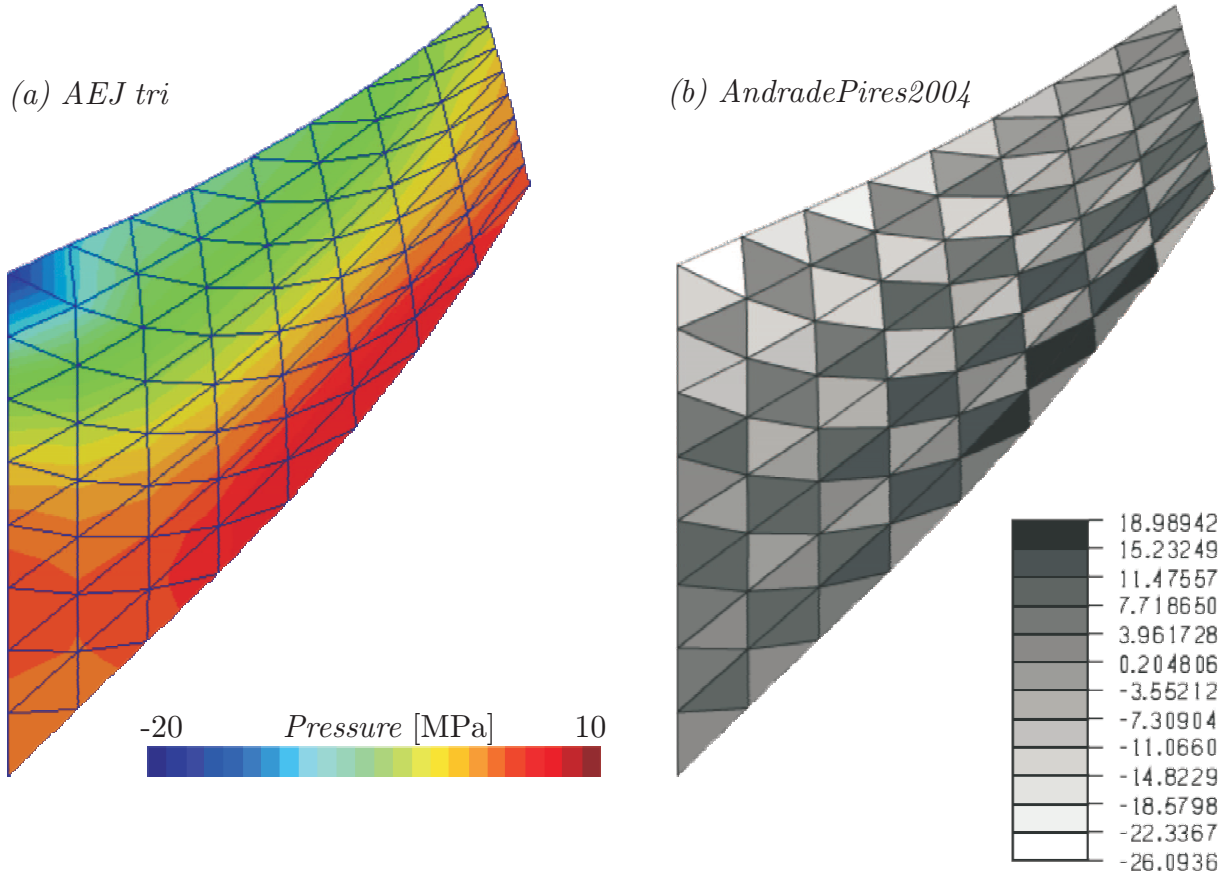


FIGURE 8.4: ***Plane strain Cook's membrane.*** *Pressure fields. (a) Average Elemental Jacobian (AEJ) linear triangle. (b) F-bar-based average nodal volume change ratio triangle proposed by Andrade Pires et al. [5]. This picture has been extracted from the original article [5].*

8.1.2 Three-dimensional case

The Cook's membrane is also a classical benchmark to assess the performance of hexahedral and tetrahedral finite elements [31, 35, 71, 96, 101, 112, 132, 147, 181]. Dimensions of the specimen are identical to the two-dimensional case, represented in Figure 8.1, with an additional thickness of 5 mm (Figure 8.5).

Four different material behaviours will be evaluated:

An incompressible Neo-Hookean hyperelastic material with shear modulus $\mu = 0.8$ MPa and bulk modulus $k = 8000$ MPa, associated with an applied shearing force $\bar{\boldsymbol{t}} = 0.0625$ MPa. The Neo-Hookean Cook's membrane has been studied, with similar parameter values, by Klaas et al. [96] and, with other values for the shear and bulk modulus, by Widany et al. [181] and Gee et al. [71]. The value of the ratio $k/\mu = 10000$ demonstrates the incompressibility of the material.

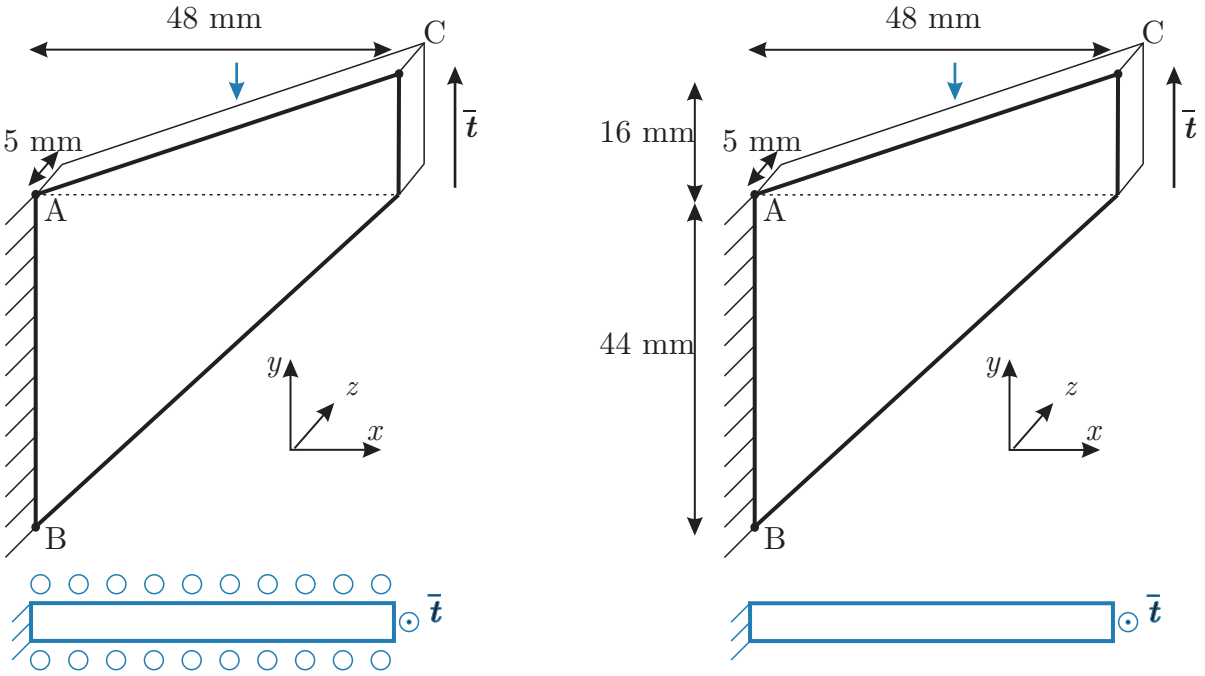


FIGURE 8.5: ***3D Cook's membrane.*** *Geometry and Loading. Left: 2D-equivalent model used in our 2D study. Right: 3D model used to assess the performance of the proposed tetrahedral formulations.*

A nearly incompressible linear elastic material with Poisson's ratio $\nu = 0.4999$ and Young's modulus $E = 1000$ MPa, associated with an applied shearing force of $\bar{\boldsymbol{t}} = 10$ MPa. Again, the obtained results may be compared with Caylak et al. [35], Mahnken and Caylak [112] and Laschet et al. [101].

A compressible linear elastic material with Poisson's ratio $\nu = 0.33$ and Young's modulus $E = 1000$ MPa, associated with an applied shearing force $\bar{\boldsymbol{t}} = 10$ MPa. Results obtained may be compared with the group Caylak et al. [35], Laschet et al. [101], Mahnken and Caylak [112], as they have used similar material properties.

An elasto-plastic material with Poisson's ratio $\nu = 0.333$, Young's modulus $E = 70$ MPa, Yield stress $\sigma_y = 0.243$ MPa and tangent modulus $E_T = 1$ MPa, associated with an applied shearing force $\bar{\boldsymbol{t}} = 0.1125$ MPa. The elasto-plastic Cook's membrane has been studied by Caylak et al. [35], Laschet et al. [101], Mahnken and Caylak [112] as well as Puso and Solberg [147].

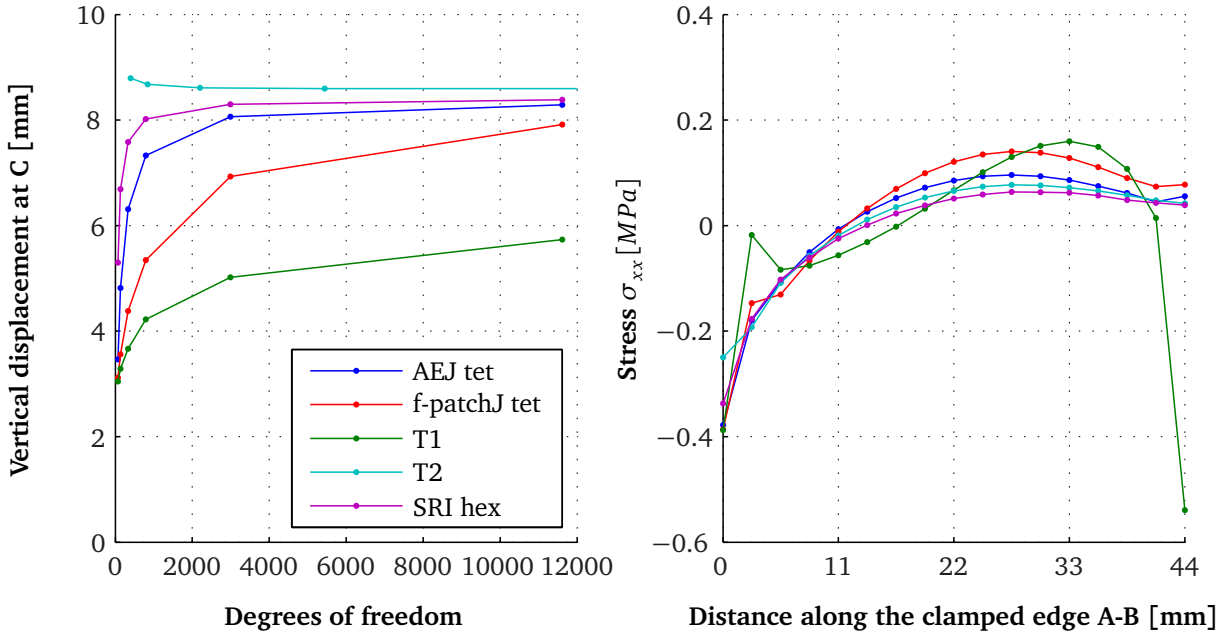


FIGURE 8.6: ***Neo-Hookean Cook's membrane in 3D.*** *Left: Vertical tip displacement at C, convergence versus mesh refinement. Right: Stress σ_{xx} along the clamped edge $A-B$.*

8.1.2.1 Incompressible Neo-Hookean hyperelastic material law

The left graph of Figure 8.6 gives the top corner vertical displacement, $\boldsymbol{u}_y$ at point C in Figure 8.5 for different element formulations and mesh sizes. For the finest mesh, the displacement at C was calculated to be $\boldsymbol{u}_y(\textit{f-patchJ tet}) = 7.914$ for the face-neighbours-patch volume change ratio linear tetrahedron and $\boldsymbol{u}_y(\textit{AEJ}) = 8.287$ for the Average Elemental Jacobian linear tetrahedron. These results are both close to the quadratic tetrahedral element: $\boldsymbol{u}_y(T2) = 8.595$ and the *SRI* hexahedral element: $\boldsymbol{u}_y(T2) = 8.384$, compared the standard linear tetrahedral who appears to be very stiff: $\boldsymbol{u}_y(T1) = 5.733$. Moreover, both our elements seem to be more efficient in removing the combined shear and volume locking than the stabilised mixed linear displacement- linear pressure tetrahedral element: $\boldsymbol{u}_y(\textit{Klaas}) = 7.17$ proposed by Klaas et al. [96]. These displacement values have been reported in Figures 8.7 and 8.8. Finally, the left graph of Figure 8.6 indicates that our *AEJ tet* element converges faster than our *f-patchJ tet* element.

Figure 8.6, Right, shows the stress distributions σ_{xx} along the clamped edge (edge A-B in Figure 8.5) for the five element formulations drawn in Figure 8.6. The σ_{xx} stress field distribution is also depicted on the Cook's membrane in Figure 8.7. However, the graph of Figure 8.6 presents the stresses computed at the tetrahedron Gauss Point but extrapolated to the element's nodes, and recorded for the nodes located at central-thickness of the clamped side of the 3D Cook's membrane; whereas Figure 8.7 shows the stress

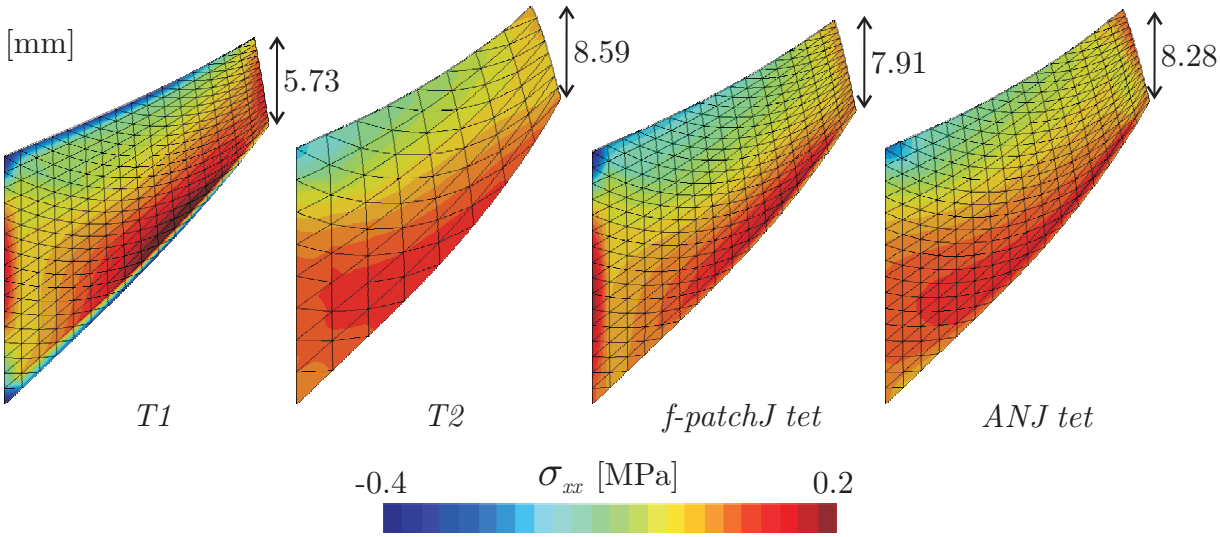


FIGURE 8.7: ***Neo-Hookean Cook's membrane in 3D.*** *σ_{xx} stress distribution for the standard linear tetrahedron T1, the quadratic tetrahedron T2, our face-neighbours-patch volume change ratio tetrahedron f-patchJ tet and our Average Elemental Jacobian tetrahedron, AEJ, .*

distribution on the front side of the membrane. If we consider the curve corresponding to *SRI hex* as the reference solution, the results may be ordered as how well they approach this reference curve: first the second order tetrahedral element *T2*, second our *AEJ* linear tetrahedral element and third our *f-patchJ* tetrahedral element. The curves for these three formulations present the same trend than that of the *SRI hex* but they are shifted towards the higher stress values; meaning that all three formulations exhibit a spurious stiffness. *T1* gives a less smoother stress distribution so that this formulation may be classified far behind the other four elements.

Figure 8.8 shows the Von Mises stress distribution for the two standard formulations *T1* and *T2* as well as for our two proposed elements *f-patchJ tet* and *AEJ* on Cook's membranes of mesh resolution 17 degrees of freedom per side (8 elements per side for the quadratic tetrahedron and 16 for the linear formulations). Clearly, our *AEJ tet* shows a very close result to the quadratic tetrahedron. The Von Mises stress field is smoother for our *f-patchJ tet* element, and even more for the standard linear tetrahedron *T1*.

8.1.2.2 Nearly incompressible linear elastic material law

Figure 8.9 presents the convergence of the solution with mesh refinement for different element formulations in the incompressible case. The graph on the left shows the final vertical tip displacement at point C for the *T1*, *T2*, *Quad SRI* and *Quad STD* elements, as

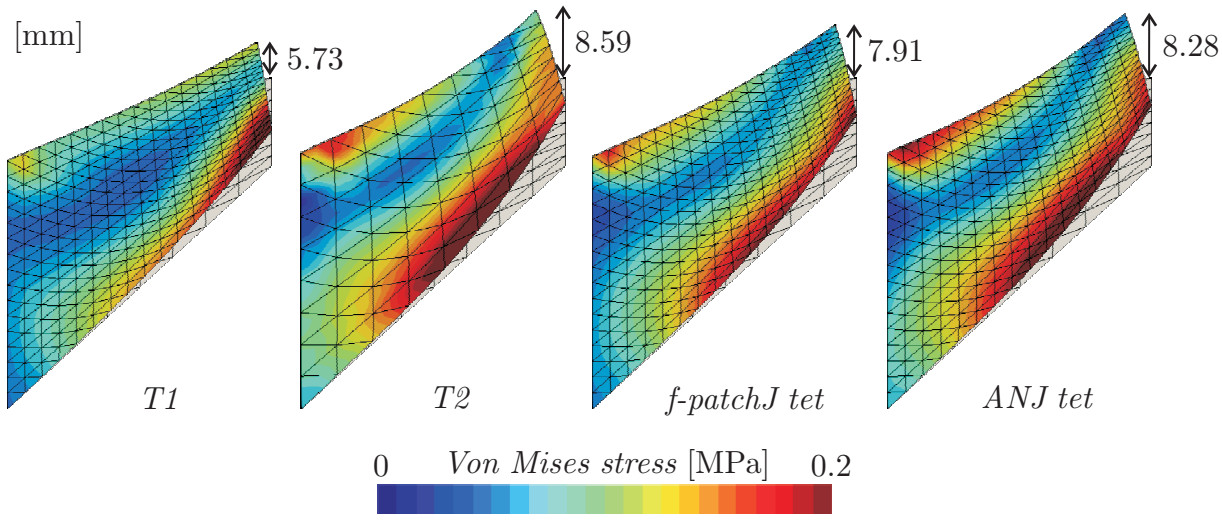


FIGURE 8.8: ***Neo-Hookean Cook's membrane in 3D.*** *Von Mises stress distribution for the standard linear tetrahedron T1, the quadratic tetrahedron T2, our face-neighbours-patch volume change ratio tetrahedron f-patchJ tet and our Average Elemental Jacobian tetrahedron, AEJ, .*

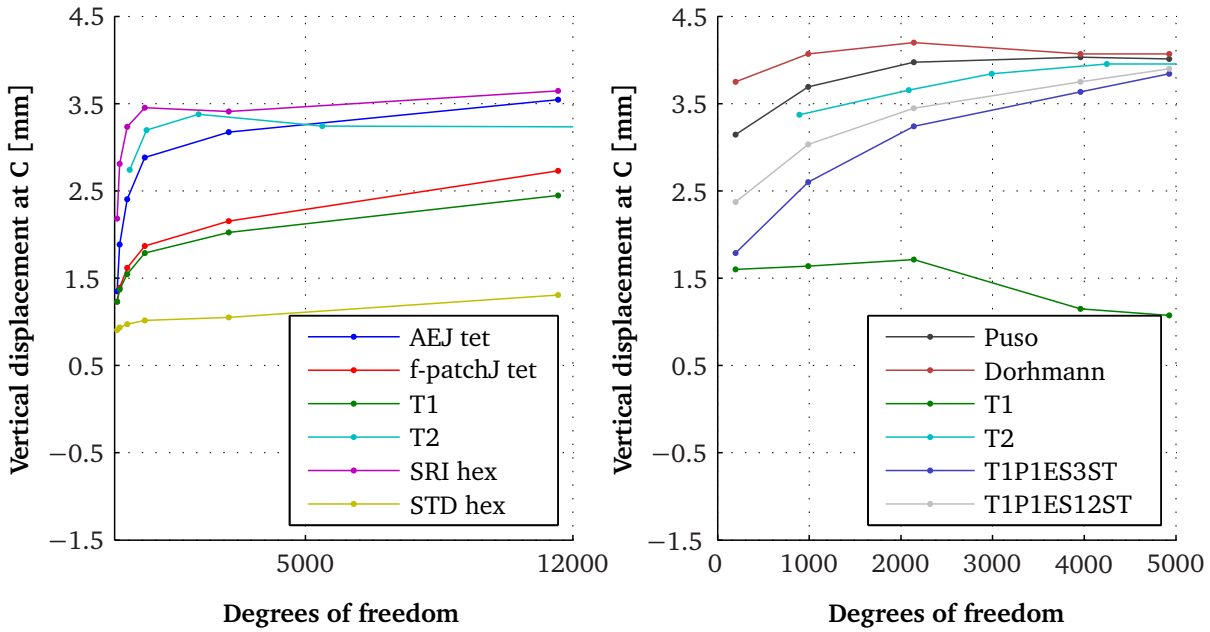


FIGURE 8.9: ***Nearly incompressible linear elastic Cook's membrane in 3D.*** *Vertical tip displacement $\boldsymbol{u}_y$ at C, convergence of with mesh refinement. Left: Results computed with Metafor. Right: Results extracted from the presentation of Caylak et al. [35], which were published in Mahnken and Caylak [112].*

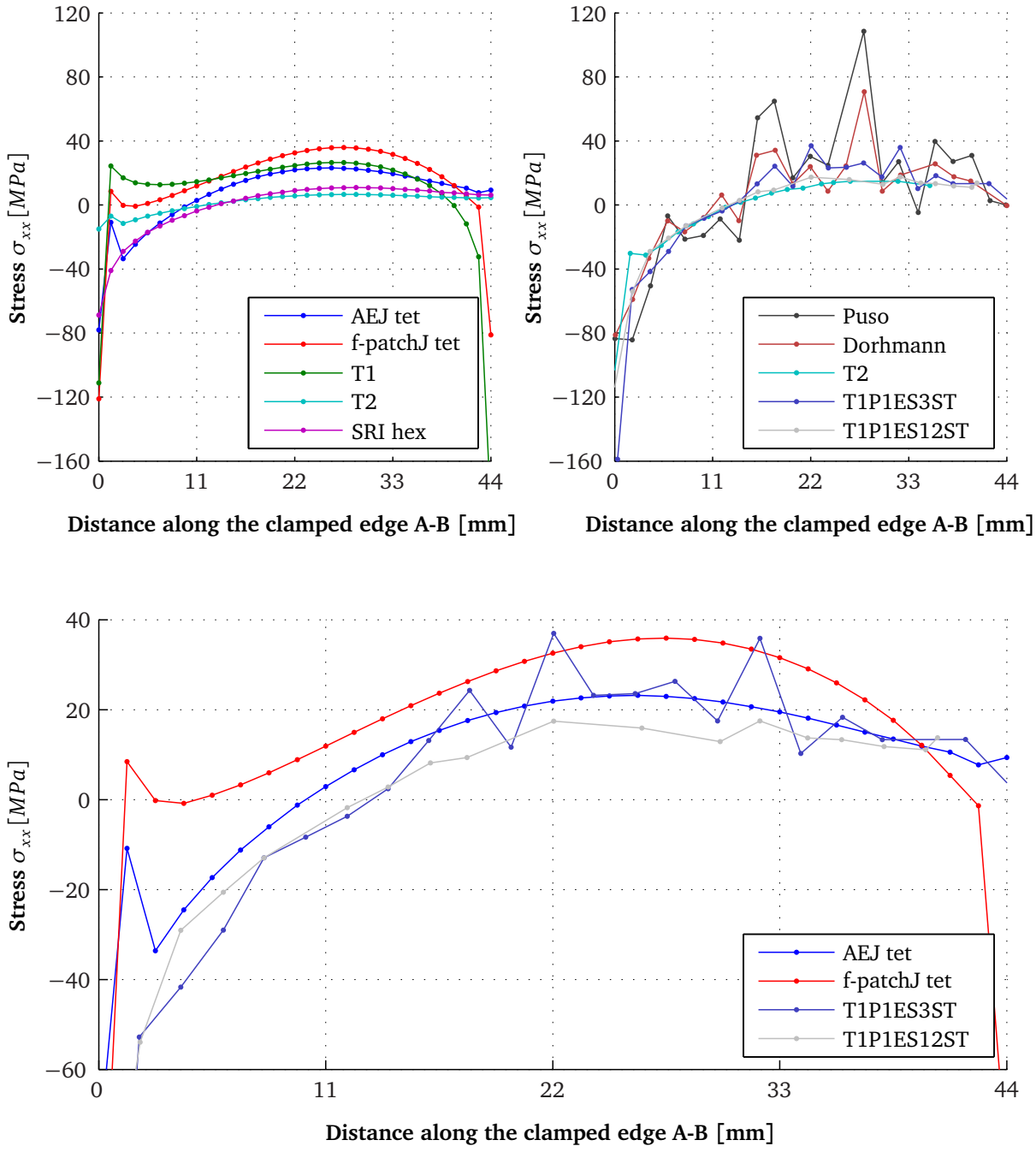


FIGURE 8.10: ***Nearly incompressible linear elastic Cook's membrane in 3D.*** *Stress σ_{xx} along the clamped edge $A-B$ (see Figure 8.2). Upper Left: Results computed with Metafor. Upper Right: Results extracted from the presentation of Caylak et al. [35], which were published in Mahnken and Caylak [112]. Lower: zoom of the upper graphs, for the proposed formulations and the best elements of Mahnken and Caylak [112].*

well as for the two proposed elements *f-patchJ tet* and *AEJ tet* , all of which were computed with Metafor. The graph on the right presents literature results. *T1P1ES3ST* and *T1P1ES12ST* are two mixed displacement-pressure tetrahedral elements, respectively with area and volume bubble functions, presented in Mahnken and Caylak [112]. The stabilised nodally integrated of Puso and Solberg [147] and the nodal-based uniform strain element of Dohrmann et al. [55] are two very popular unlocking solutions from literature. All curves were extracted from the presentation of Caylak et al. [35]. Please notice the difference in the scale of the horizontal axes.

Looking at Figure 8.9 Left, we observe a good performance of our *AEJ tet* element. For the finest mesh, the vertical displacement is $\boldsymbol{u}_y(AEJtet) = 3.55$ mm, which is close to the value obtained for the *SRI hex* of Metafor $\boldsymbol{u}_y(SRIhex) = 3.64$ mm, taken as reference solution in this study. The corresponding vertical tip displacement for the standard linear tetrahedron is $\boldsymbol{u}_y(SRIhex) = 2.4$ mm. The second order tetrahedron, *T2*, behaves well for coarse meshes but diverges from the ideal solution when the number of degrees of freedom increases; thus making our *AEJ tet* a better option for fine meshes. Our *f-patchJ tet* formulation is not powerful enough to remove the totality of the locking of the standard tetrahedron: the convergence is slow and the computed tip displacements are only slightly higher than those obtained with *T1*. The standard hexahedron is very stiff, much stiffer than the standard tetrahedron.

Comparing Metafor results, Figure 8.9, Left, with literature Figure 8.9, Right, we observe an overall under-estimation of the vertical displacement at point C, for all Metafor elements. This difference is observed even for the standard elements *T1* and *T2*, even though the same model with the same incompressible material properties than Mahnken and Caylak [112] have been used. The origin of this difference could not yet be found. Future investigations will include a comparison with Abaqus [114] elements as well as with other literature results.

Figure 8.10 shows the stress σ_{xx} along the clamped edge $A-B$ for the same elements (see Figure 8.2). Again, the graph on the left shows the result computed with Metafor, and the graph on the right refers to the results of Mahnken and Caylak [112]. The stress curves were obtained in similar manner than for the neo-Hookean material, Section 8.1.2.1, i.e. by extrapolating the stress values, computed at the Gauss point(s) of the element, to the nodes. Apart from the irregularities observed along the clamped edge, which are due to this extrapolation, we do not observe the high irregularities in stress distribution along the clamped edge as do Puso and Dorhmann [112]. Results for the standard hexahedron are not indicated as they fall out of the graph due to the high locking of this element.

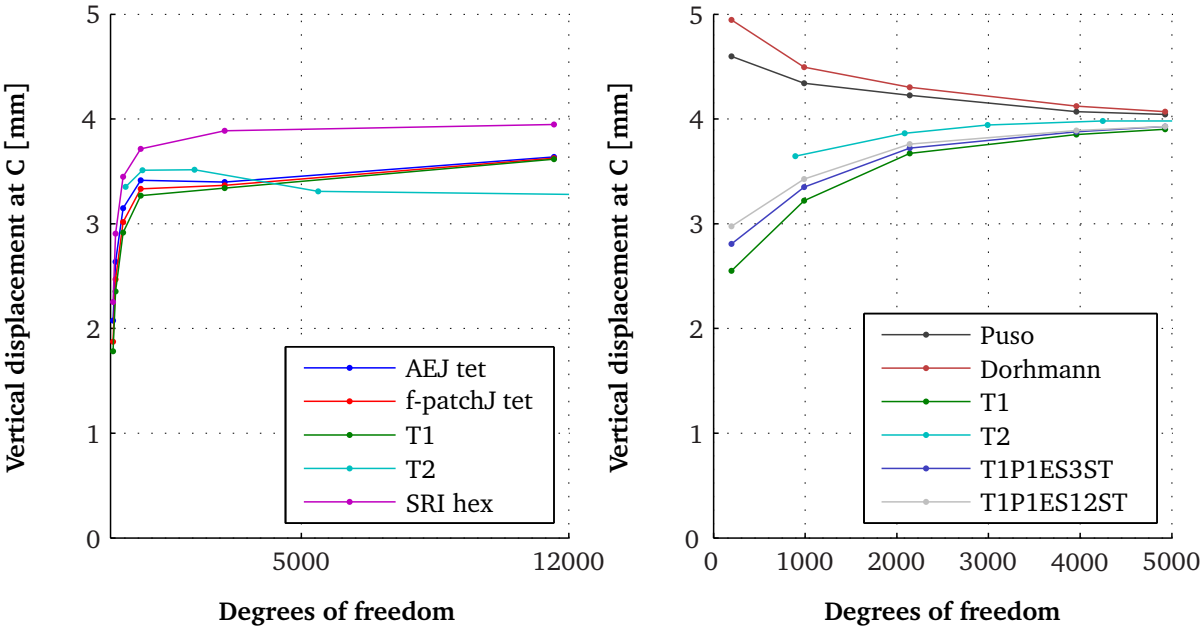


FIGURE 8.11: ***Compressible linear elastic Cook's membrane in 3D.*** *Left: Results computed with Metafor. Right: Results extracted from the presentation of Caylak et al. [35], which were published in Mahnken and Caylak [112].*

8.1.2.3 Compressible linear elastic material law

Figure 8.11 presents the convergence results for the compressible Cook's membrane in three-dimensions. Again, the results that we obtained with Metafor (Figure 8.11, Left) are plotted against the results of Mahnken and Caylak [112] (Figure 8.11, Right). As for the incompressible case, the standard and the quadratic tetrahedron of Metafor do not quite give the same results than in Mahnken and Caylak [112]. Results obtained with the selective reduced integration hexahedral element of Metafor does however converge to the same limit of $\boldsymbol{u}_y = 4$ than the formulations presented in Mahnken and Caylak [112]. Apart from this observation, results for the two proposed formulations *AEJ tet* and *f-patchJ tet* are identical to those of the standard linear tetrahedron *T1* in the compressible case, which is reassuring. As already observed for the incompressible Cook's membrane the quadratic tetrahedron diverges from the reference solution when finer meshes are used.

8.1.2.4 Elasto-plastic material law

Figures 8.12 and 8.13 present the convergence of the vertical tip displacement at point C and the stress distribution σ_{xx} along the clamped edge respectively for the elasto-plastic three-dimensional Cook's membrane. The final tip displacement for this element, for a

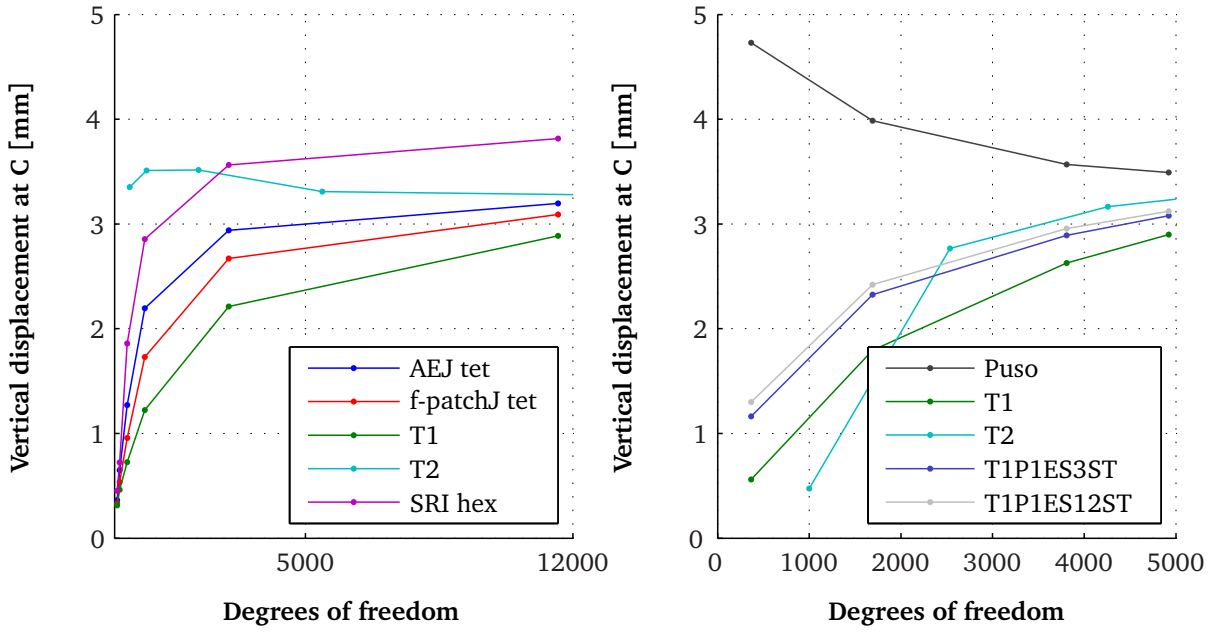


FIGURE 8.12: ***Elasto-plastic Cook's membrane in 3D.*** *Vertical tip displacement* $\mathbf{u}_y$ *at C, convergence of with mesh refinement. Left: Results computed with Metafor. Right: Results extracted from the presentation of Caylak et al. [35].*

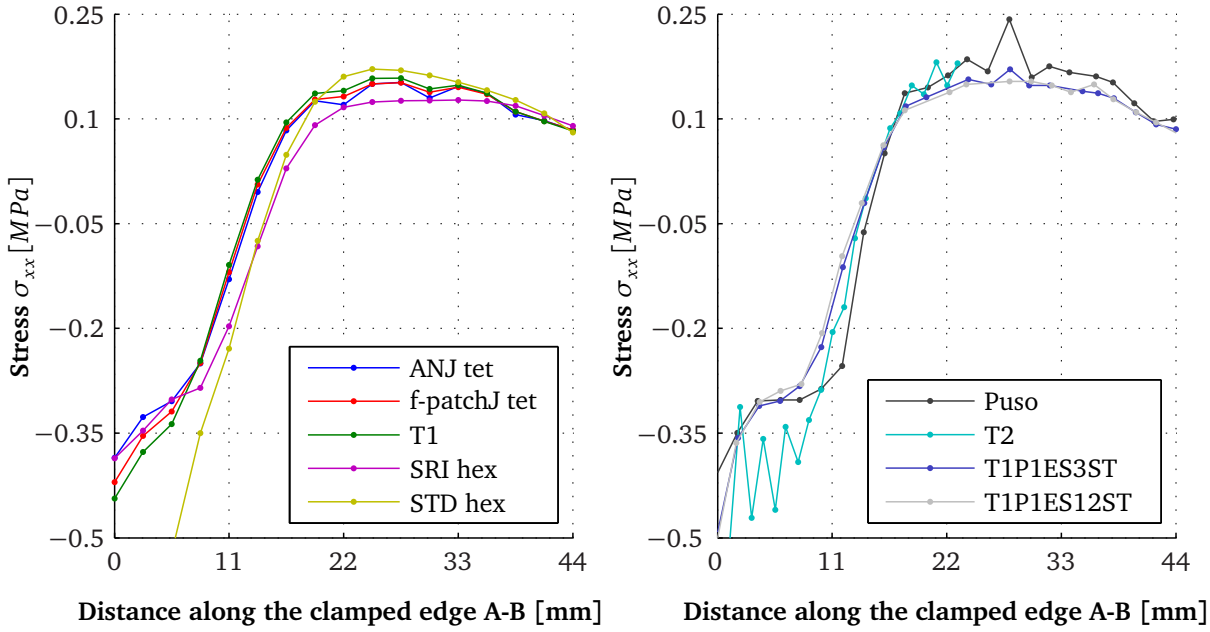


FIGURE 8.13: ***Elasto-plastic Cook's membrane in 3D.*** *Stress* σ_{xx} *along the clamped edge* $A-B$ *(see Figure 8.2). Left: Results computed with Metafor. Right: Results extracted from the presentation of Caylak et al. [35].*

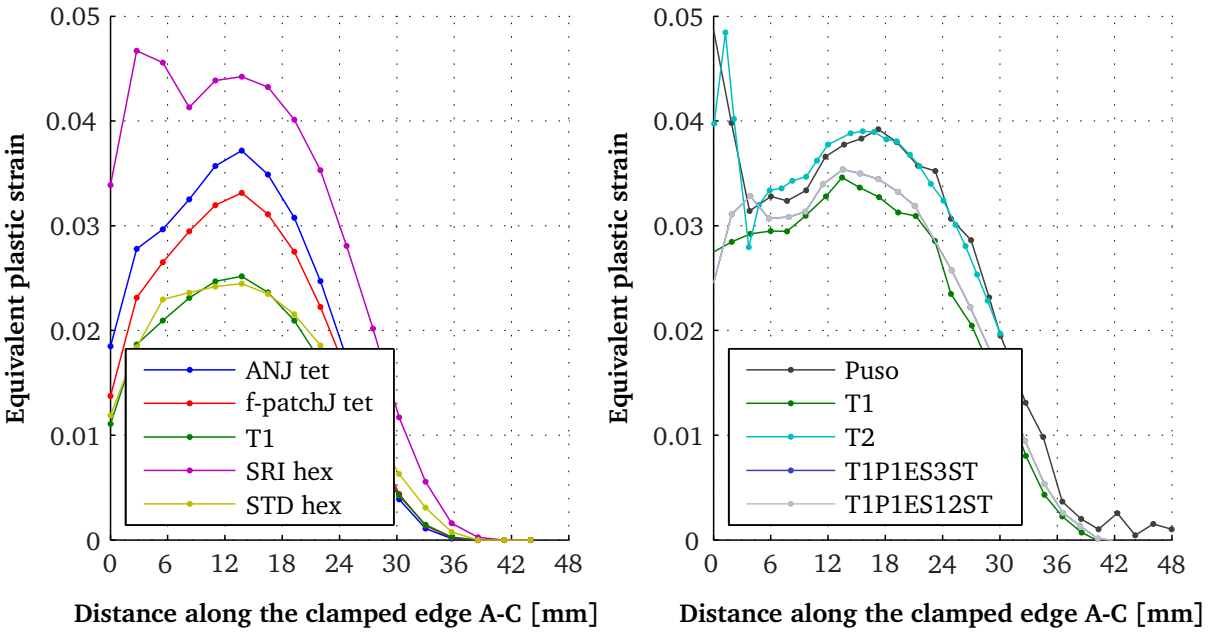


FIGURE 8.14: ***Elasto-plastic Cook's membrane in 3D.*** *Equivalent plastic strain along the upper edge $A-C$ (see Figure 8.2). Left: Results computed with Metafor. Right: Results extracted from the presentation of Caylak et al. [35].*

mesh size of 5000 ddls, is $\boldsymbol{u}_y(AEJ) = 2.673$. This value is lower, meaning that our element is stiffer, than the one obtained for both stabilised enhanced assumed strain elements of Mahnken and Caylak [112] with area and bubble functions, $\boldsymbol{u}_y(T1P1ES12ST) = 3.122$ and $\boldsymbol{u}_y(T1P1ES3ST) = 3.08$ [2].

Figure 8.14 presents the distribution of the equivalent plastic strain along the upper edge of the Cook's membrane. Results indicate that when no unlocking formulation is used, i.e. for the standard linear tetrahedron and hexahedron, the plastic flow is highly under-estimated. Higher values of the equivalent plastic strain are predicted by both our unlocking proposals *AEJ tet* and *f-patchJ tet*. Results obtained with our *AEJ tet* lie in between the linear displacement/linear pressure enhanced strain formulation with stabilisation and volume-or-area bubble function proposed by Mahnken and Caylak [112] and the stabilised nodally integrated tetrahedron of Puso and Solberg [147].

[2] These results for the elasto-plastic Cook's membrane were not presented in the article [112] itself but in the associated conference presentation [35].

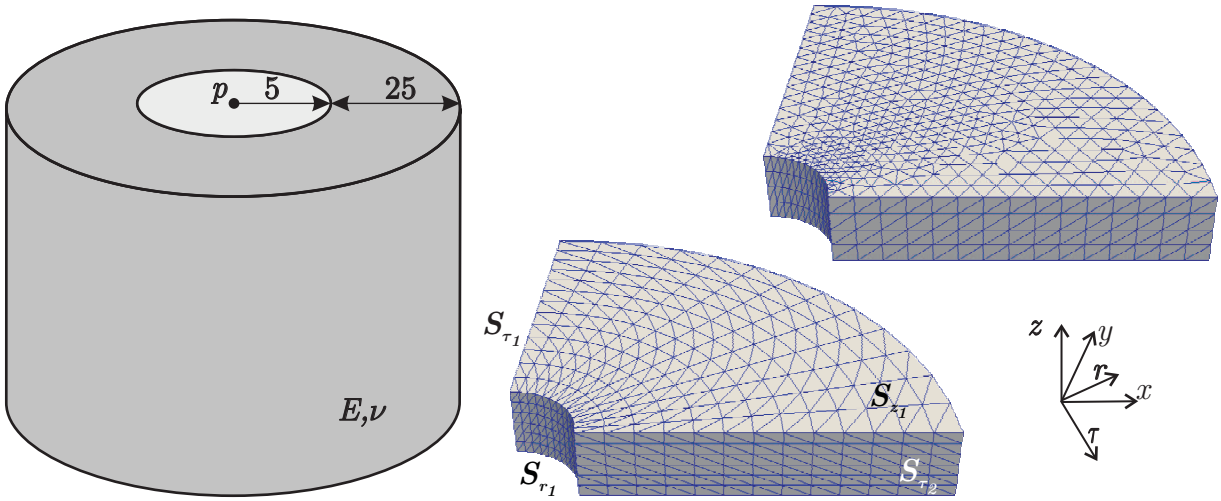


FIGURE 8.15: ***Cylinder under internal pressure.*** *Geometry, loading and initial structured and unstructured meshes.*

8.2 Thick-walled cylinder under internal pressure

The second example is a the finite element simulation of a thick-walled[3] cylinder under internal pressure. The numerical investigation is done in the three-dimensional space, but plane strain conditions are assumed. The initial geometry and mesh of the problem are represented in Figure 8.15. Using symmetry conditions only a quarter of a cylinder is considered, the surfaces S_{τ_1} and S_{τ_2} are constrained along their normal direction. The front and back surfaces S_{z_1} and S_{z_2} are also constrained along their normal direction in order to enforce the plane strain condition. A pressure p is applied on the internal surface S_{r_1}.

The exact elastic solution for the thick-walled cylinder under internal pressure, in the case of small strains, can be obtained by expressing the equilibrium equations in polar coordinates, the strain-displacements equations and Hooke's law and then assuming that the cylinder is long enough to ensure that plane sections remain plane [130]. The radial displacement depends on the elastic properties of the cylinder, its inner and outer radii and the applied internal pressure:

$$u(r) = \frac{(1+\nu)pr_1^2}{E(r_2^2 - r_1^2)} \left(\frac{r_2^2}{r} + (1-2\nu)r \right) \tag{8.1}$$

[3] A thick-walled cylinder or tube is one where the thickness of the wall $r_2 - r_1$ is greater than one-tenth of the radius r_2: $r_2 - r_1 > 0.1r_2$.

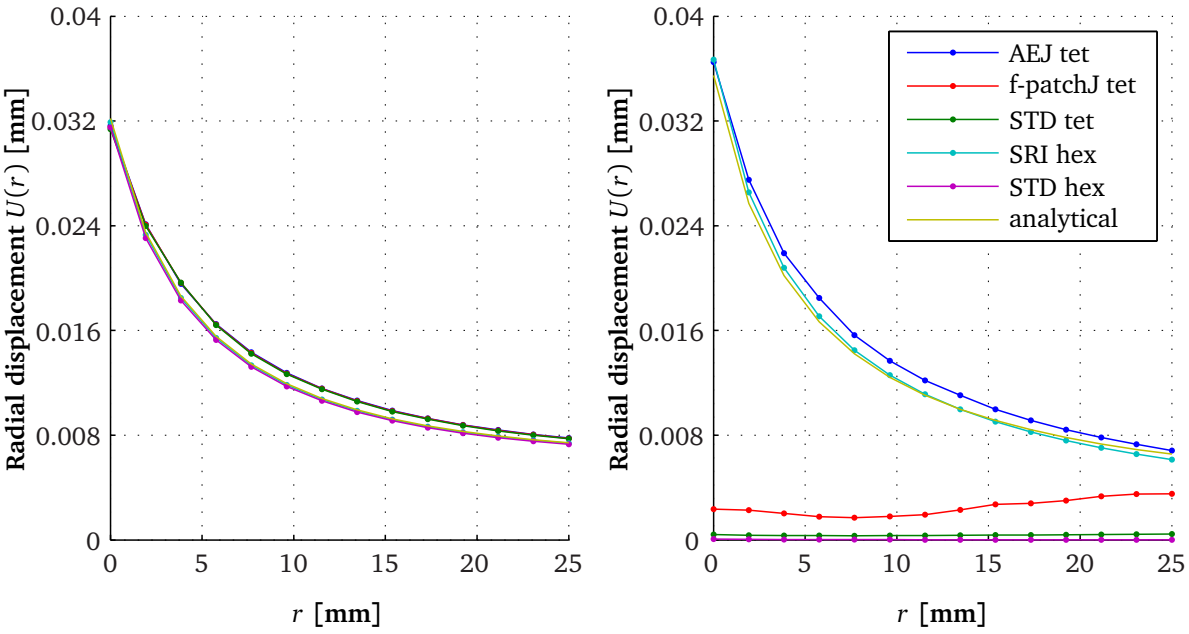


FIGURE 8.16: ***Cylinder under internal pressure.*** *Unstructured mesh. Radial displacement. Left:* $\nu = 0.33$. *Right:* $\nu \approx 0.5$.

The radial and tangential or circumferential stresses within the tube depend on the inner pressure and the inner and outer radii:

$$\sigma_r(r) = \frac{pr_1^2 - p\frac{r_1^2 r_2^2}{r^2}}{r_2^2 - r_1^2} \tag{8.2}$$

$$\sigma_\tau(r) = \frac{pr_1^2 + p\frac{r_1^2 r_2^2}{r^2}}{r_2^2 - r_1^2} \tag{8.3}$$

Among these stresses, the tangential stress is the highest.

For comparison purposes, we take the same material and geometrical properties than Mahnken and Caylak [112]. The inner and outer radii of the cylinder are set to $r_1 = 5$ mm and $r_2 = 30$ mm and an internal pressure of $p = 1000$ MPa is imposed (Figure 8.15). Two materials are investigated: a compressible linear elastic material with Young's modulus $E = 210$ GPa and Poisson's ratio $\nu = 0.33$ and a nearly incompressible linear elastic material with the same Young's modulus but associated with a Poisson's ratio of $\nu = 0.49995$. A quasi-static integration scheme is used and the tangent stiffness matrix is computed analytically.

Also, two meshes are considered: a structured mesh, obtained by constructing a hexahedral mesh of $13 \times 13 \times 6$ elements and then subdividing each hexahedron in 6, leading to 1372 nodes, and an unstructured mesh of 2180 nodes. This gives problem sizes of 4116 and 6540 degrees of freedom, respectively. The six elements on the thickness of the cylinder are

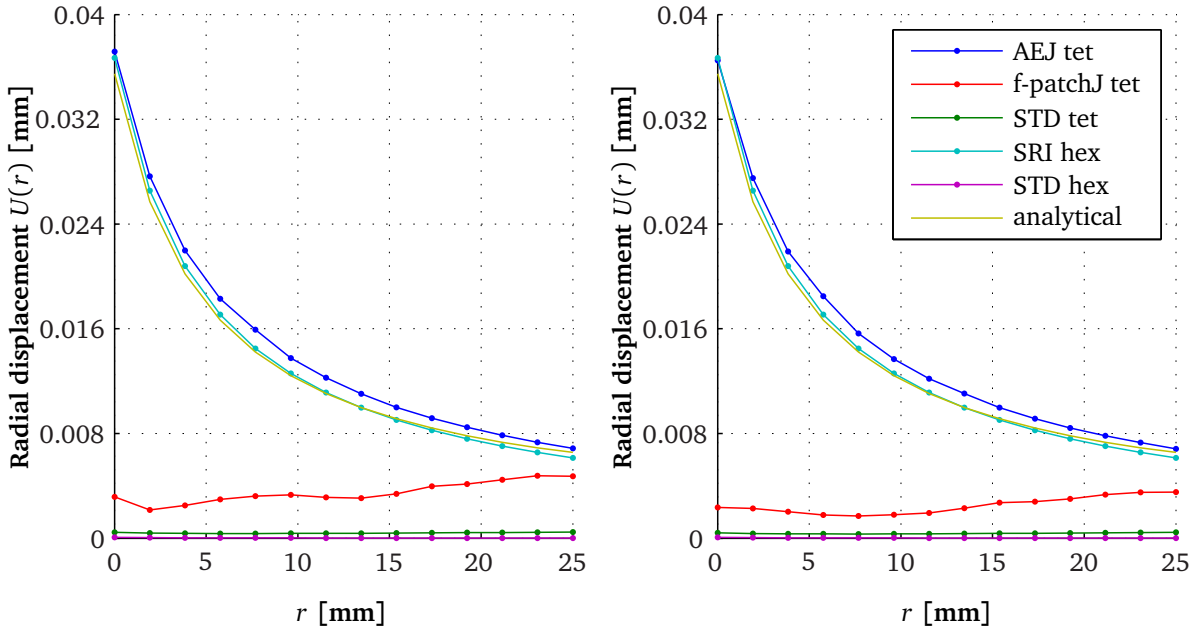


FIGURE 8.17: ***Cylinder under internal pressure.*** *Unstructured mesh, $\nu \approx 0.5$. Left: Radial displacement u_y along S_{τ_1}. Right: Radial displacement u_x along S_{τ_2}.*

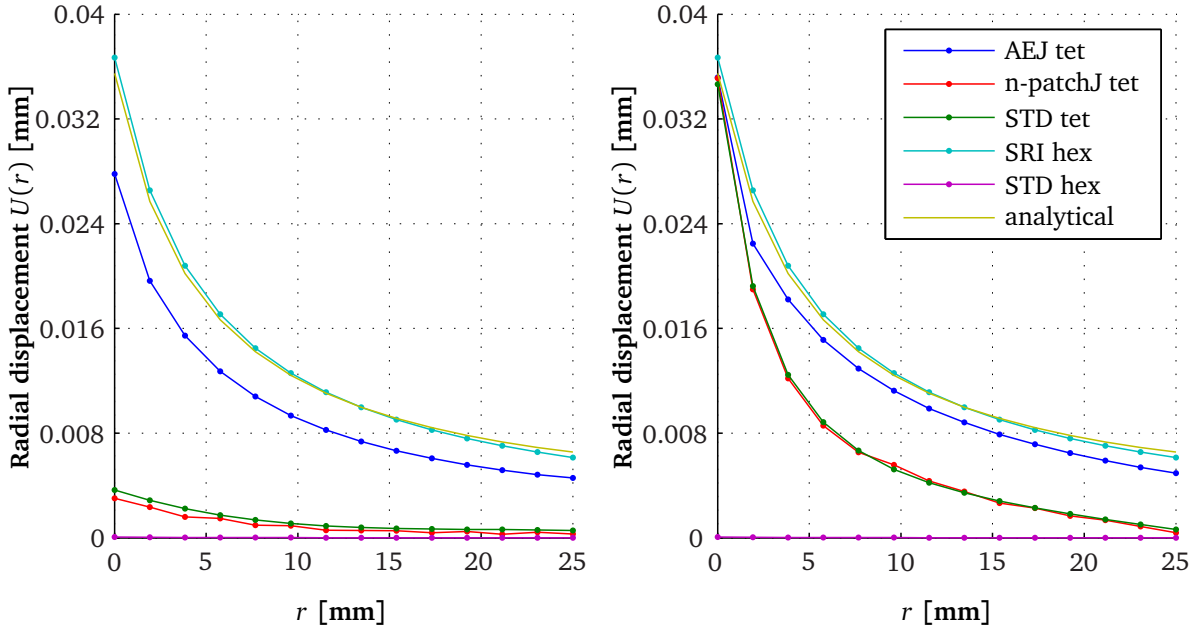


FIGURE 8.18: ***Cylinder under internal pressure.*** *Structured mesh, $\nu \approx 0.5$. Left: Radial displacement u_y along S_{τ_1}. Right: Radial displacement u_x along S_{τ_2}.*

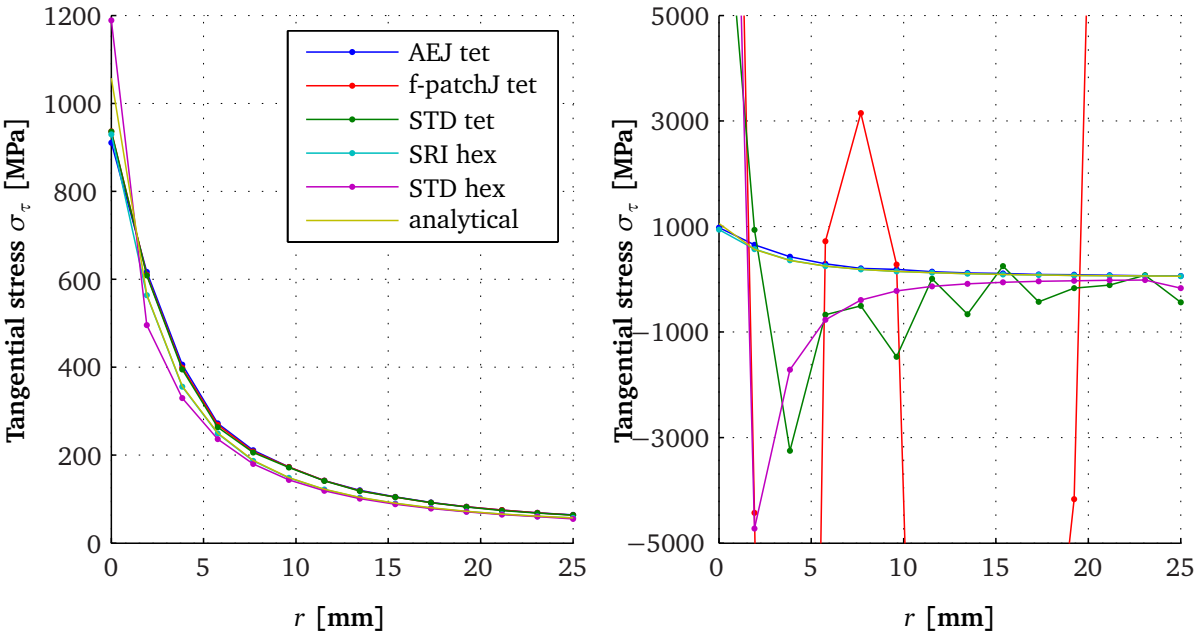


FIGURE 8.19: ***Cylinder under internal pressure.*** *Unstructured mesh. Tangential stress. Left:* $\nu = 0.33$. *Right:* $\nu \approx 0.5$.

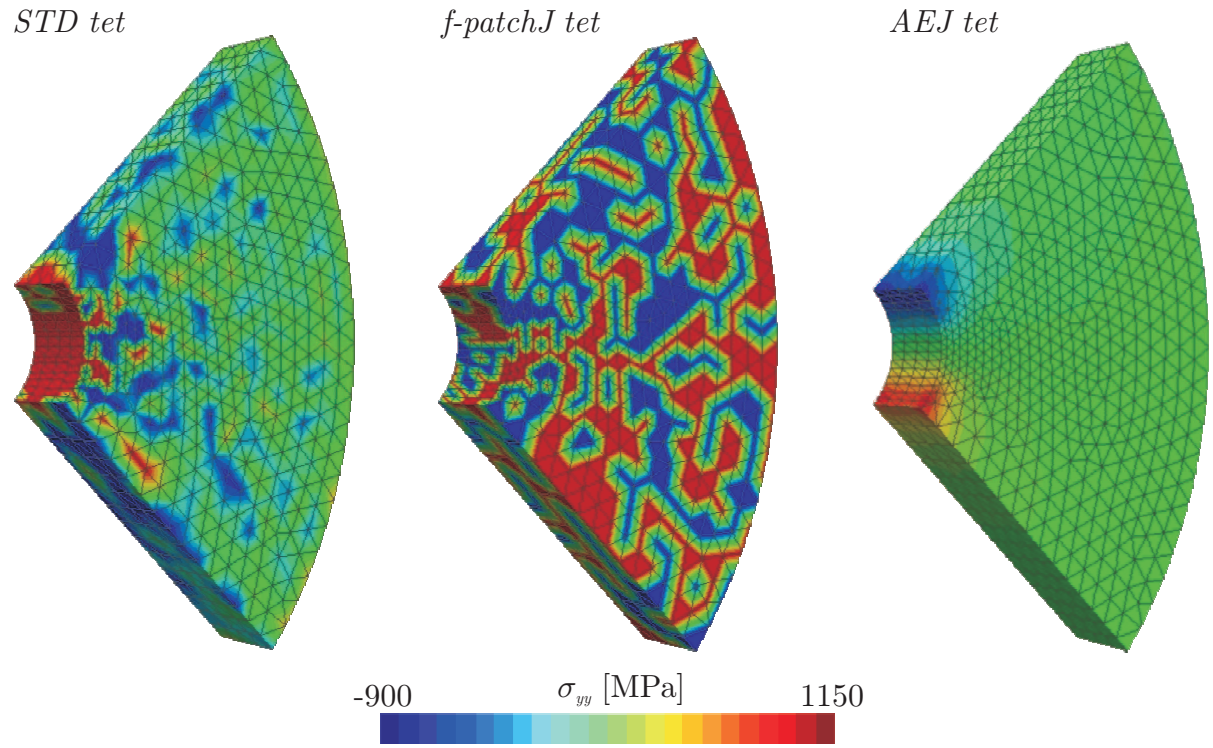


FIGURE 8.20: ***Cylinder under internal pressure.*** *Unstructured mesh.* $\nu \approx 0.5$. *Stress field* σ_{yy}. *Left: standard linear tetrahedron. Centre: patch volume ratio linear tetrahedron. Right: Average Elemental Jacobian (AEJ) tetrahedron.*

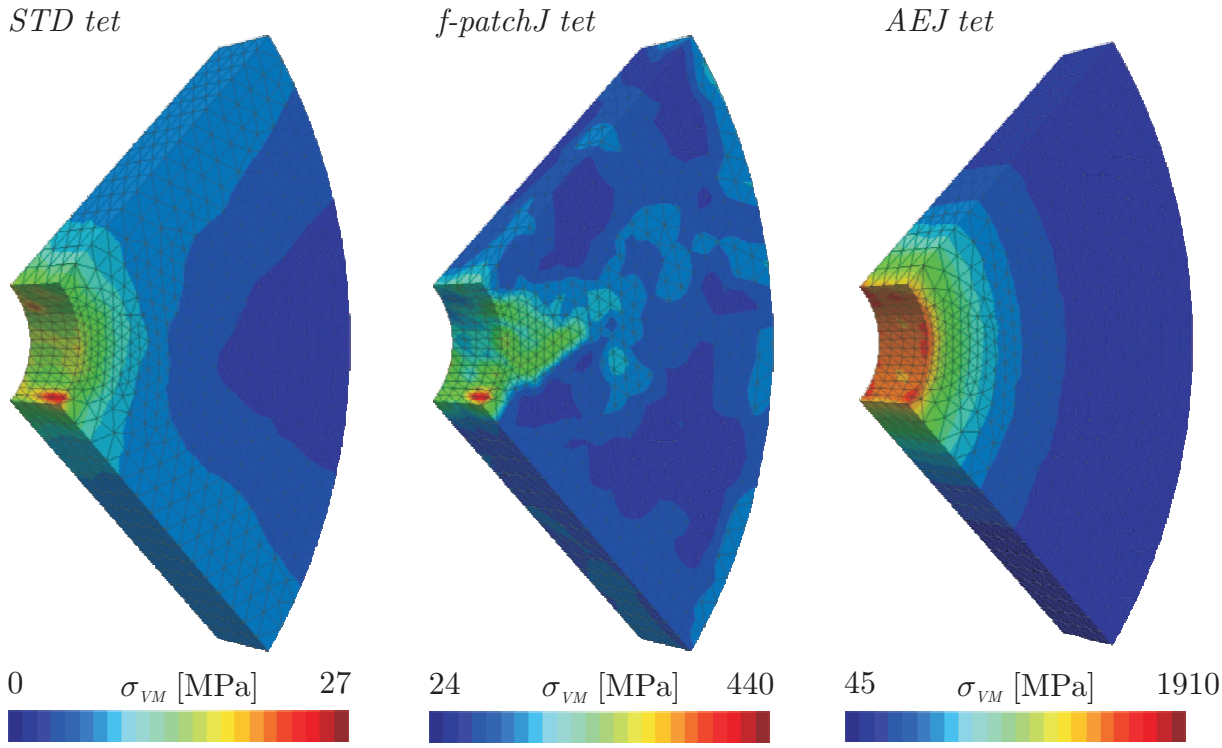


FIGURE 8.21: ***Cylinder under internal pressure.*** *Unstructured mesh.* $\nu \approx 0.5$. *Von Mises Stress field* σ_{VM}.

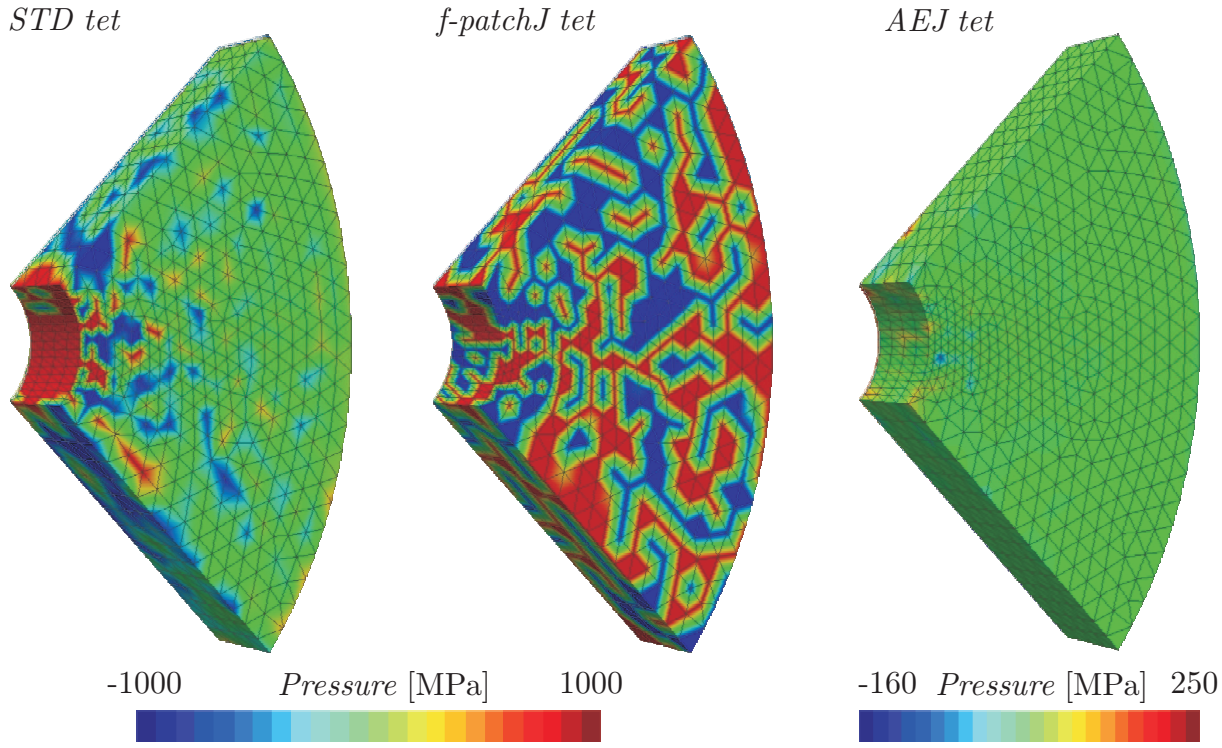


FIGURE 8.22: ***Cylinder under internal pressure.*** *Unstructured mesh.* $\nu \approx 0.5$. *Pressure field.*

needed to assess the performance of the proposed unlocking formulations because the latter require information from the element's neighbours to compute its modified deformation gradient.

In Figure 8.16 we compare the radial displacement, computed as the displacement along the x-coordinate on facet S_{τ_2} (Figure 8.15), for the two proposed unlocking formulations *AEJ tet* and *patchJ tet*, the standard linear tetrahedron, *T1*, and the standard and SRI hexahedral elements, *STD hex* and *SRI hex*. The analytical solution (8.1) is also represented. For the compressible case (Figure 8.15, Left), the deviations are almost negligible, even though a small deviation to the analytical curve can be noticed for the tetrahedral elements. For $\nu \approx 0.5$ (Figure 8.15, Right), the standard linear tetrahedron and the standard linear hexahedron show very large deviations from the analytical solution, due to locking. The face-neighbourhood-based patch volume ratio tetrahedron *f-patchJ tet* also shows to be very stiff. The curve obtained for the Average Elemental Jacobian tetrahedron, *AEJ*, is, however, very satisfactory: the deviation observed with the analytical curve is similar to the compressible case, so that we hypothesis that this deviation is inherent to the use of tetrahedra instead of hexahedrons.

Figure 8.17 and 8.18 show the radial displacement $u(r)$ for the the unstructured and structured mesh respectively. The graphs on the left represent the radial displacement on the facets S_{τ_1} and the graphs on the right represent the radial displacement on the facets S_{τ_2} (see Figure 8.15 for the location of these facets). For the structured mesh, the radial displacement of facets S_{τ_1} and S_{τ_2} are not similar. Also, comparing both figures (structured and unstructured mesh), we observe that the structured mesh introduces an additional stiffness to the problem, both *AEJ* curves, on facets S_{τ_1} and S_{τ_2}, being under the analytical solution.

In Figure 8.19 the tangential stress σ_τ, taken as σ_{yy} on S_{τ_2}, is depicted for the different element formulations. The analytical solution (8.3) is also shown. The unstructured mesh was used to compute these graphs, results obtained with the structured mesh are similar. Again, for the compressible material (Figure 8.19, Left), all curves are close the analytical solution. Results obtained for $\nu = 0.444495$ are again more heterogeneous. To help understanding these curves, we represented the stress fields obtained for the standard linear tetrahedron, the f-patch volume ratio tetrahedron and the Average Elemental Jacobian tetrahedron in Figure 8.20. Clearly an oscillatory behaviour is observed for the standard linear tetrahedron and, even more, for the f-patch volume ratio tetrahedron. The Average Elemental Jacobian element provides a smooth field and the corresponding curve in Figure 8.19, Right, lies very close to the analytical solution.

TABLE 8.3: ***Elasto-plastic Taylor bar impact.*** *Results for several finite element formulations, including the proposed f-patchJ tet and AEJ tet elements.*

Element type	**dof**	**CPU time**	**steps** [mm]	**Final Radius** R_f [mm]	**Final Height** h_f
T1	1836	1min 14s	1408	4.19	21.2
f-patchJ tet	1836	2min 40s	1604	4.90	20.9
n-patchJ tet	1836	4min 57s	1621	4.92	21
AEJ tet	1836	22min 33s	3061	6.52	21.2
STD hex	1836	58s	1023	4.64	20.3
SRI hex	1836	1min 59s	4374	7.05	21.5
AEJ tet	19802	17h 25min 55s	10734	6.862	20.8
SRI hex	19802	53min 53s	11857	7.129	21.4

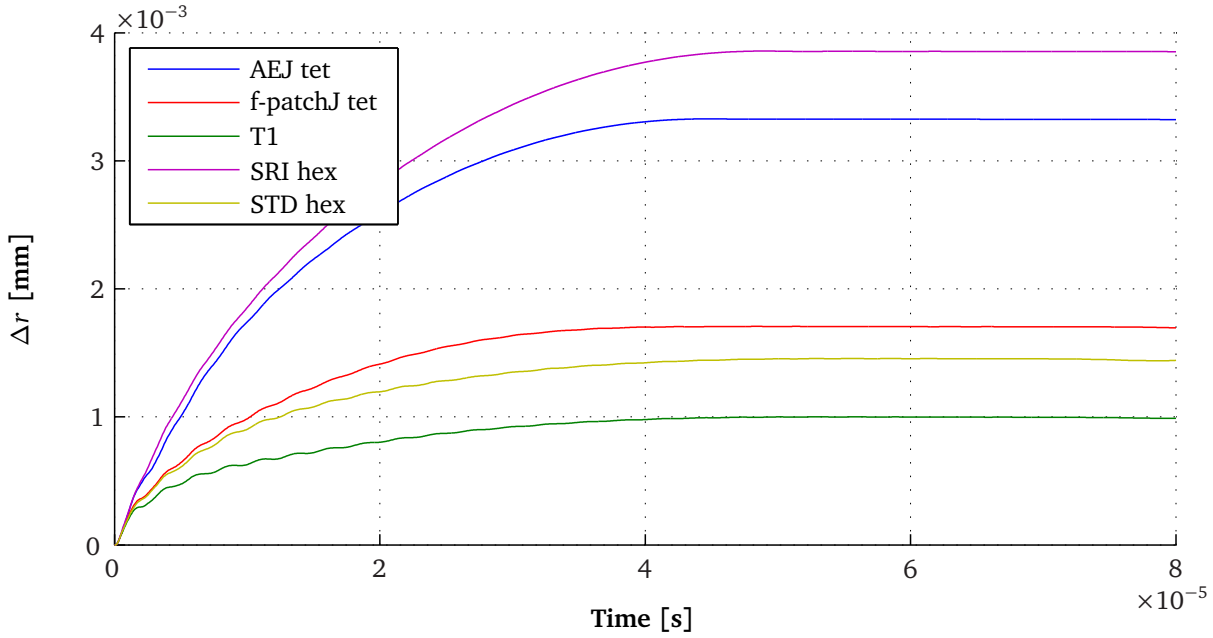


FIGURE 8.23: ***Elasto-plastic Taylor bar impact.*** *Radius increase versus time.*

8.3 Taylor bar impact

The Taylor bar impact, the impact of a cylindrical rod at high speed, is a classic benchmark to study plasticity. It has been studied in the particular case of unlocking formulations for the linear tetrahedron by Puso and Solberg [147] to assess the performance of their

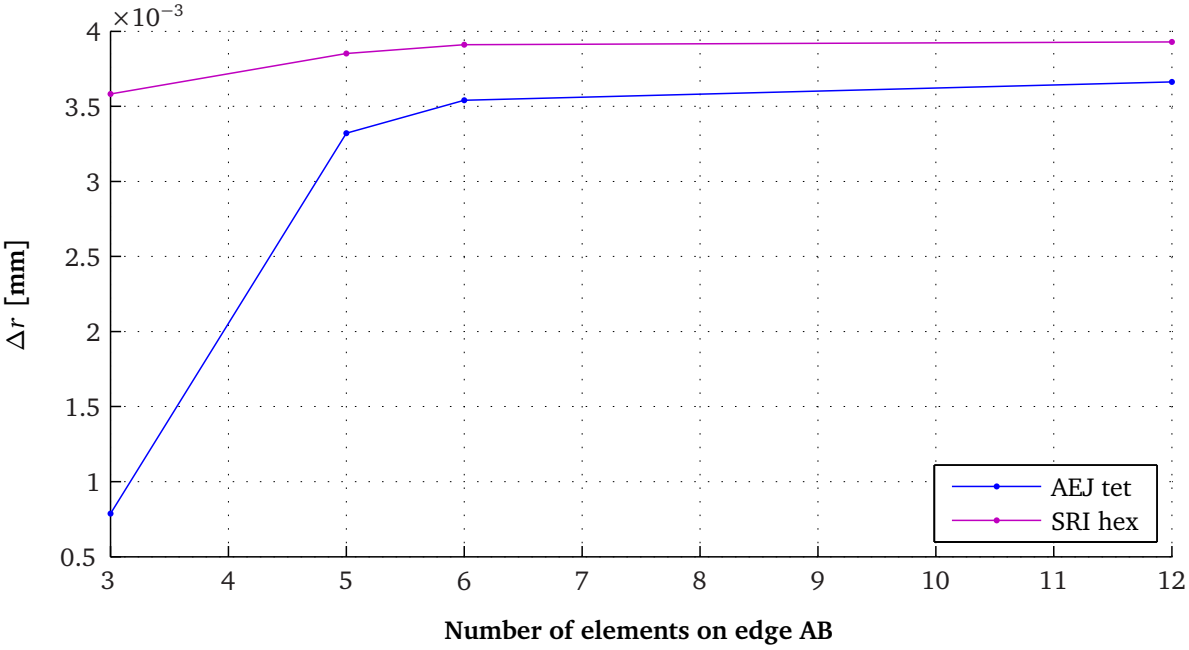


FIGURE 8.24: ***Elasto-plastic Taylor bar impact.*** *Convergence of the solution with mesh refinement.*

stabilised nodally integrated linear tetrahedron. The latter is a nodal-based formulation and thus specially suited for explicit time integration. The unlocking formulations proposed in this dissertation were developed in an implicit framework. And, the previous sections showed that the proposed Average Elemental Jacobian (AEJ) linear tetrahedron in particular was well-suited for implicit finite element analysis. This section will assess the performance of the proposed elements for high speed dynamics finite element analysis using an explicit time integration scheme.

The Taylor bar problem studies the impact with a rigid surface of a cylindrical rod moving with high speed. The bar is modelled as an elasto-plastic material with a Young's modulus of $E = 117$ GPa, a Poisson's ratio of $\nu = 0.35$, an initial yield stress of 0.4 GPa and a hardening modulus of $H = 0.1$ GPa. The initial length of the bar is 32.4 mm and the initial radius is $R_i = 3.2$ mm. A solution is obtained for an initial velocity of 227 m/s. The interval of 80 μs has been analysed.

An three-dimensional representation of the bar is studied. However, thanks to symmetry constraints, only a quarter of the cylinder is modelled. The bar is discretised by generating an initial hexahedral mesh of $5 \times 5 \times 30$ elements and then subdividing each hexahedron into 6 tetrahedra for the tetrahedral meshes. This gives models of size 1836 degrees of freedom. With the aim of studying the convergence of the solution with mesh refinement,

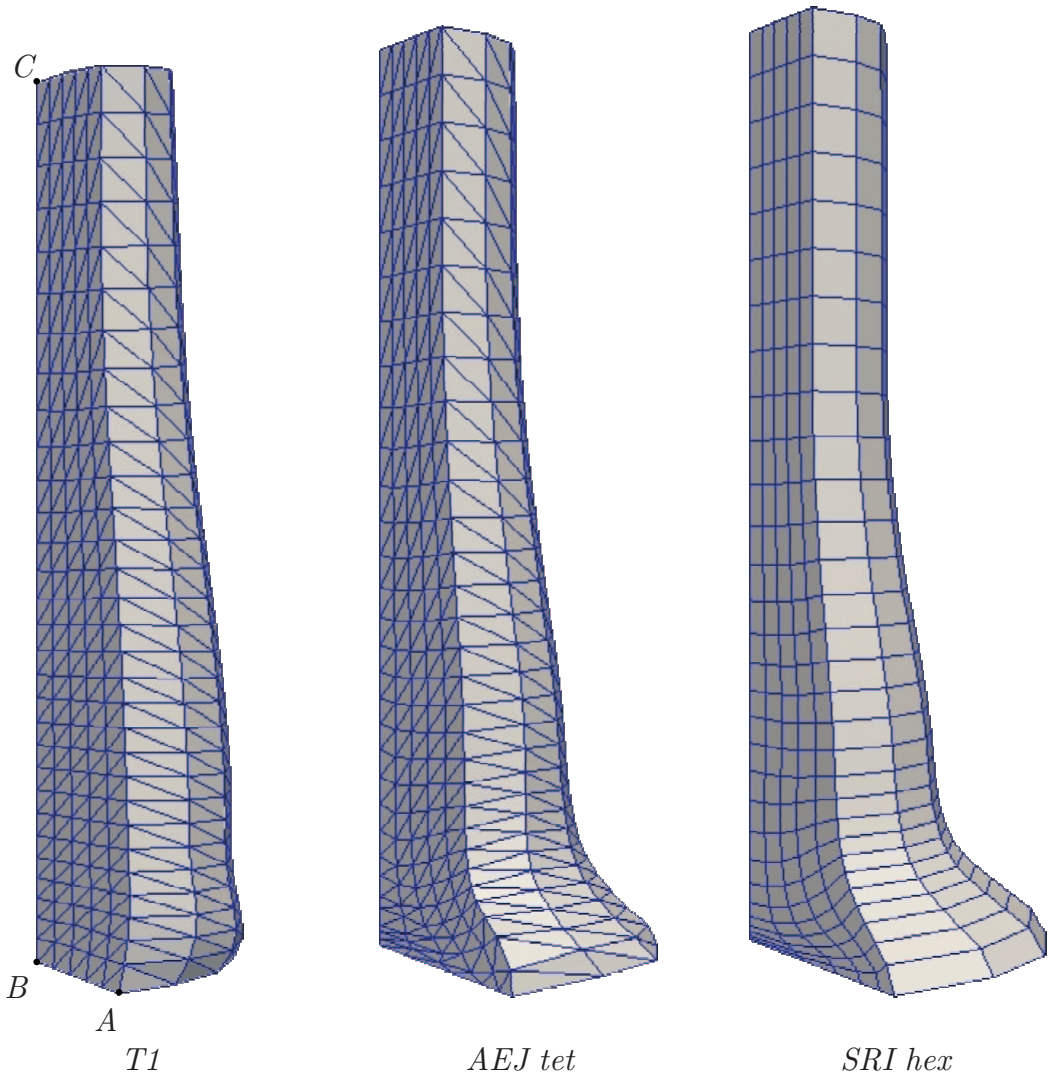


FIGURE 8.25: ***Elasto-plastic Taylor bar impact.*** *Illustration of the studied tetrahedral and hexahedral models. The pictures are the final deformed shapes of the Taylor bar modelled with three different finite elements. The corresponding front views are indicated in Figure 8.26.*

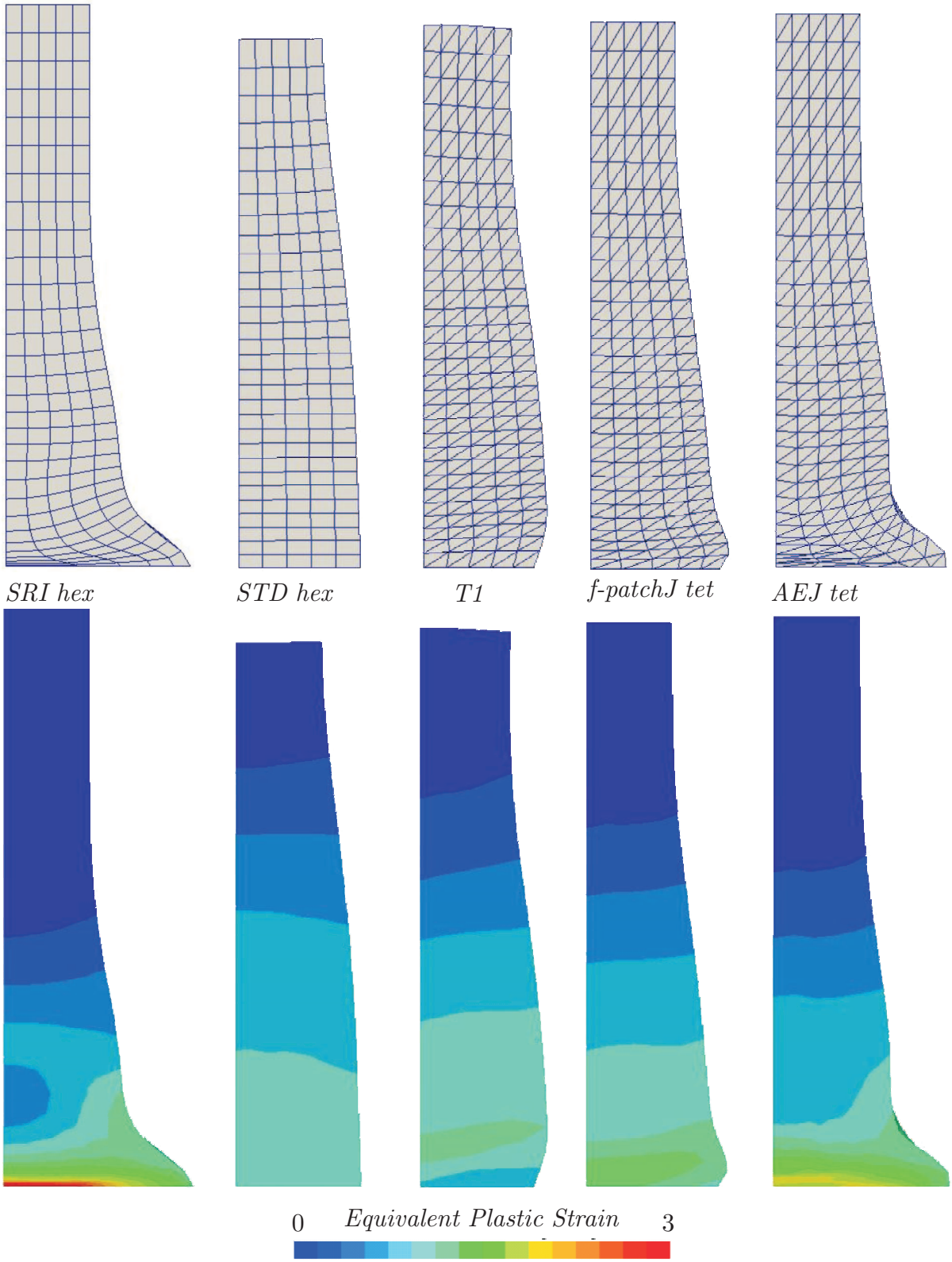


FIGURE 8.26: ***Elasto-plastic Taylor bar impact.*** *Upper: Final deformed shapes obtained for several finite element formulations, including the proposed f-patchJ tet and AEJ tet elements. Lower: Equivalent plastic strain distribution for the same finite element formulations.*

discretisations of $3 \times 3 \times 15$, $6 \times 6 \times 45$ and $12 \times 12 \times 50$ hexahedral elements were also generated.

Table 8.3 presents the obtained results, in terms of computer time, number of steps, final radius (distance A-B) and final height (distance B-C) for the standard linear tetrahedron, the proposed elements *f-patchJ tet*, *f-patchJ tet* and *AEJ tet* , the standard linear hexahedron and the hexahedron with selective reduced integration (The points A, B and C are indicated in Figure 8.25). Comparing the results obtained for our face and node-neighbourhood patch elements, both formulations output similar results, with the node-neighbourhood patch element being slightly less stiff but at the cost of an increase in computer time and memory. Also, simulation times for our *AEJ tet* , approximately 22 minutes, may appear quite high as compared to simulation times for the other elements, which is below 5 minutes. This is due to our implementation of this element in which we do not keep any additional nodal quantity in memory, neither the volumes V_I and v_I, nor the Jacobian J_I, so that these values are recomputed for each element and at each step. Keeping these three nodal quantities in memory will lead to a drastic decrease in computation time in the future.

The radius increase of the cylindrical bar during the impact is presented in Figure 8.23. This graphs clearly illustrates the superiority of our *AEJ tet* as compared to our *f-patchJ tet*. But, our *AEJ tet* element is still stiffer than the *SRI hex*, which means that a certain amount of locking is still present. Possible explanation for this is that the *AEJ tet* element has not yet converged for the illustrated mesh resolution, whereas the *SRI hex* has. In an attempt to answer this question, the convergence of the solution with mesh refinement is proposed in Figure 8.24.

In Figure 8.24 the x-axis indicates the mesh resolution, evaluated as the number of elements on edge A-B (see Figure 8.25); and the y-axis gives the radius increase obtained using either our *AEJ tet* or the *SRI quad*. This graph indicates that much finer meshes are needed to obtain a converged solution with *AEJ tet* element than is the case with the *SRI hex*. Simulations results for the finest mesh are reported in Table 8.3.

The final deformed shapes obtained using different formulations are shown in Figure 8.26 to help the reader to assess the differences in final lengths and radii visually (the pictures are the front views of the 3D quarter of the cylindrical bars). As presented in Table 8.3, the final length of the bar is identical for the standard tetrahedron *T1* and our Average Elemental Jacobian tetrahedron, *AEJ*, but the final radii are significantly different. The explanation for this is illustrated in Figure 8.26, where we observe that the upper extremity of the cylindrical rod is skewed in the *T1* case.

Figure 8.26, Lower, presents the equivalent plastic strain distribution for the five finite element formulations considered. Clearly all formulations predict a lower plastic flow than

the SRI hexahedron. In decreasing order we have

$$\begin{aligned}
\bar{\epsilon}^p(\textit{SRI hex}) &= 3.123,\\
\bar{\epsilon}^p(\textit{AEJ tet}) &= 2.122,\\
\bar{\epsilon}^p(\textit{n-patchJ tet}) &= 1.269,\\
\bar{\epsilon}^p(\textit{f-patchJ tet}) &= 1.242,\\
\bar{\epsilon}^p(\textit{STD tet}) &= 0.924, \text{and}\\
\bar{\epsilon}^p(\textit{STD hex}) &= 0.777
\end{aligned}$$

8.4 Concluding Remarks

The goal of this chapter was to assess the efficiency of the two un-locking ideas presented in the previous chapter and implemented in Metafor [111]. Both two-dimensional and three-dimensional applications were considered. Several materials were investigated, including compressible and incompressible linear elastic, neo-Hookean and elasto-plastic. Also, quasi-static implicit and dynamic explicit tests were performed. Finally, all three applications were popular benchmarks taken from literature so that our element could be compared with the most popular formulations of the literature.

Clearly, the proposed face- or node- neighbourhood patch volume change ratio linear simplex element is not satisfactory: only a part of the locking behaviour of the standard tetrahedron is removed, the stress field obtained in the case of Cook's membrane is smoothed so that the extrema are under-estimated (Figure 8.3), the stress distribution obtained in the case of the cylinder under internal pressure is highly sporadic (Figure 8.20), and the equivalent plastic strain in the Taylor bar is also under-estimated (Figure 8.26).

Our Average Elemental Jacobian tetrahedron, proposed for the two-dimensional case by Andrade Pires et al. [5] but corrected and extended to third dimension in this dissertation (Section 7.4), appears to be very efficient in removing both shear and volumetric locking.

Results for the plane-strain Cook's membrane are better than in the original article of Andrade Pires et al. [5] (Figure 8.2), which may be due to our corrections applied to the stiffness terms of the tangent stiffness matrix (see Section 7.4.2). Obtained vertical tip displacements for the 2D Cook's membrane are also higher, meaning that the membrane is less stiff, and closer to the results obtained with the selective reduced integrated quadrilateral than the displacements obtained by de Souza Neto et al. [52] for the F-bar-patch triangle and F-bar quadrilateral. Also, in contrast to the original element of Andrade Pires et al. [5], a smooth distribution of the hydrostatic pressure over the membrane is observed.

In three-dimensions, the proposed Average Elemental Jacobian (AEJ) tetrahedron is also very satisfactory. The neo-Hookean Cook's membrane test shows a good convergence of the element with mesh refinement and a correct stress distribution over the membrane. For the nearly incompressible linear elastic material law, the obtained results are also very close to those obtained with the selective reduced integrated hexahedral element, which has been proved to remove volumetric locking effectively. In the compressible case, i.e. when there is no locking, the proposed *AEJ* element outputs identical results to the standard linear tetrahedron, which is reassuring. Finally, the elasto-plastic Cook's membrane showed slightly under-estimated displacement values as compared to the curves presented in Mahnken and Caylak [112]. However, both the stress and the equivalent plastic strain distributions are correctly evaluated.

The performance of the proposed low-order tetrahedral element has also been assessed against the analytical solution of the thick-walled cylinder under internal pressure. Deviations observed for our Average Elemental Jacobian are small, especially when an unstructured mesh is used. Moreover the stress and pressure distribution over the cylinder do not exhibit the classical checkerboard pattern observed when locking occurs.

Finally, the benchmark of the Taylor bar shows that the proposed element substantially reduces the locking of the linear tetrahedron in high-speed explicit analyses. However final radii and height of the bar are not identical to the solution obtained for the hexahedral element with selective reduced integration. The convergence of the proposed element with mesh refinement is also slower than the one observed with the selective reduced integrated hexahedral element.

Part III

Biomechanical Applications

Chapter 9

Introduction

In Part 1, we developed procedures to generate patient-specific finite element meshes from segmented images. These procedures can be employed whatever the number of tissues and the geometries in the segmented dataset. In Part 2, we proposed a locking-free tetrahedral element that can be used for finite element simulations with incompressibility or Von Mises plasticity. Here, we illustrate the suitability of the above developments to solve actual biomechanical problems.

The first application is the finite element analysis of the compression of a deer antler cancellous bone. Several types of meshing methods, hexahedral and tetrahedral, are studied and their influence on the results of the finite element simulation is assessed.

The second application is the finite element modelling of intra-operative brain shift deformation, based on pre-operative and intra-operative scan-data. We use both our meshing algorithm and our non-locking tetrahedral element to improve a previously proposed biomechanical model of the brain [175].

The third application is the finite element study of dog humeral fractures. The developments of this thesis were used to create a multi-material model of a dog humerus. The influence of the skeletal development (young versus adult dog), the elbow configuration (flexion-extension and exo-endoration angles) and the load direction on stress distribution within the humerus is analysed; and possible fracture types are deduced.

All three applications are the result of collaborative projects that could benefit from the meshing algorithms and/or the non-locking tetrahedral element developed in this thesis. However, the purpose of this chapter is more to illustrate the possibilities and application range of our image - to - FE model approach, than to bring solutions to actual problems of biomechanics.

Chapter 10

Influence of meshing strategy on finite element analysis of cellular structures

The goal of this chapter is to compare the different meshing strategies presented in Part 1 and analyse their influence on the results of a micro-finite element compression test of a cellular structure.

10.1 Building of the finite element model

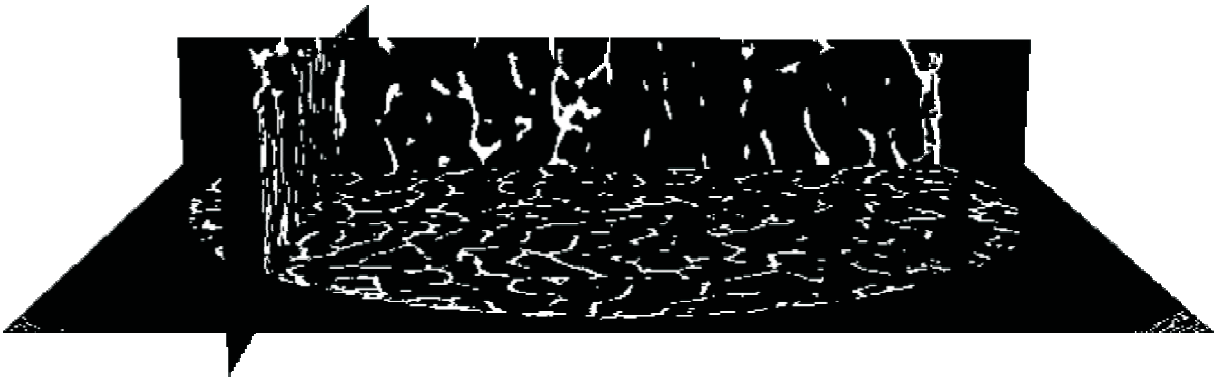
10.1.1 Image acquisition and preparation

Figure 10.1 (a), shows the initial dataset. It is a $\mu-$CT scan of the central core of a deer antler[1]. This 3D image was acquired with a X-Ray micro-tomography imaging system at the Department of Clinical Sciences, Faculty of Veterinary Medicine, University of Liège. This dataset was introduced in [102] and has already been used in previous studies [49, 50, 117].

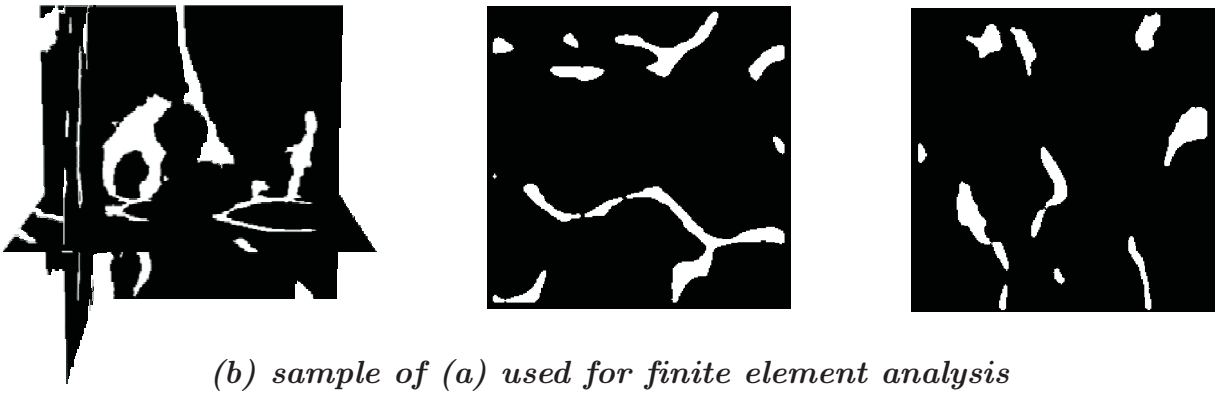
The size of the acquired 3D image is $1835 \times 1843 \times 350$ voxels (i.e. approximately 1.2 billion voxels). For this study, in order to capture the geometry of the trabeculae accurately, the out-of-plane resolution[2] was set to be equal to the in-plane resolution, i.e. 8.64 μm. Therefore, the size of bone sample was $15.9 \times 15.9 \times 3.02$ mm^3. The sample is characterised by an apparent volume density of 18.7%.

[1]The antlers of cervids are constituted of bone tissue: a central core of cancellous bone, surrounded by a thick layer of compact bone

[2]The out-of-plane resolution, z-resolution in this work, is the resolution that is perpendicular to the scanning direction. It is usually less than the in-plane resolution, xy-resolution in this work.



(a) initial μ-CT scan of a deer antler cancellous bone



(b) sample of (a) used for finite element analysis

FIGURE 10.1: ***Finite element study of deer antler cancellous bone.*** *Initial and re-sampled dataset.*

Only a subset of this image was used for micro-finite element modelling. Indeed, the whole dataset would lead to prohibitive mesh sizes and simulation times. Moreover, we are interested in comparing the influence of several mesh strategies on simulation results, and not, in the scope of this thesis, in the validation of the micro-finite element model in comparison with experimental tests. Hence, for the present study, modelling the whole specimen was not necessary. Therefore a cubic sample of dimensions $245 \times 245 \times 245$ voxels (i.e. approximately 14.7 million voxels) was selected within the initial dataset, outputting a cubic specimen of $2.11 \times 2.11 \times 2.11$ mm^3. The resulting 3D image is represented in Figure 10.1 (b).

10.1.2 Mesh generation

Four different meshes were created from the binary 3D image of Figure 10.1 (b). These are represented in Figure 10.2. Mesh (a) was generated by the very simple voxel-conversion technique (see Chapter 5). The 38581 hexahedrons in the mesh correspond to the num-

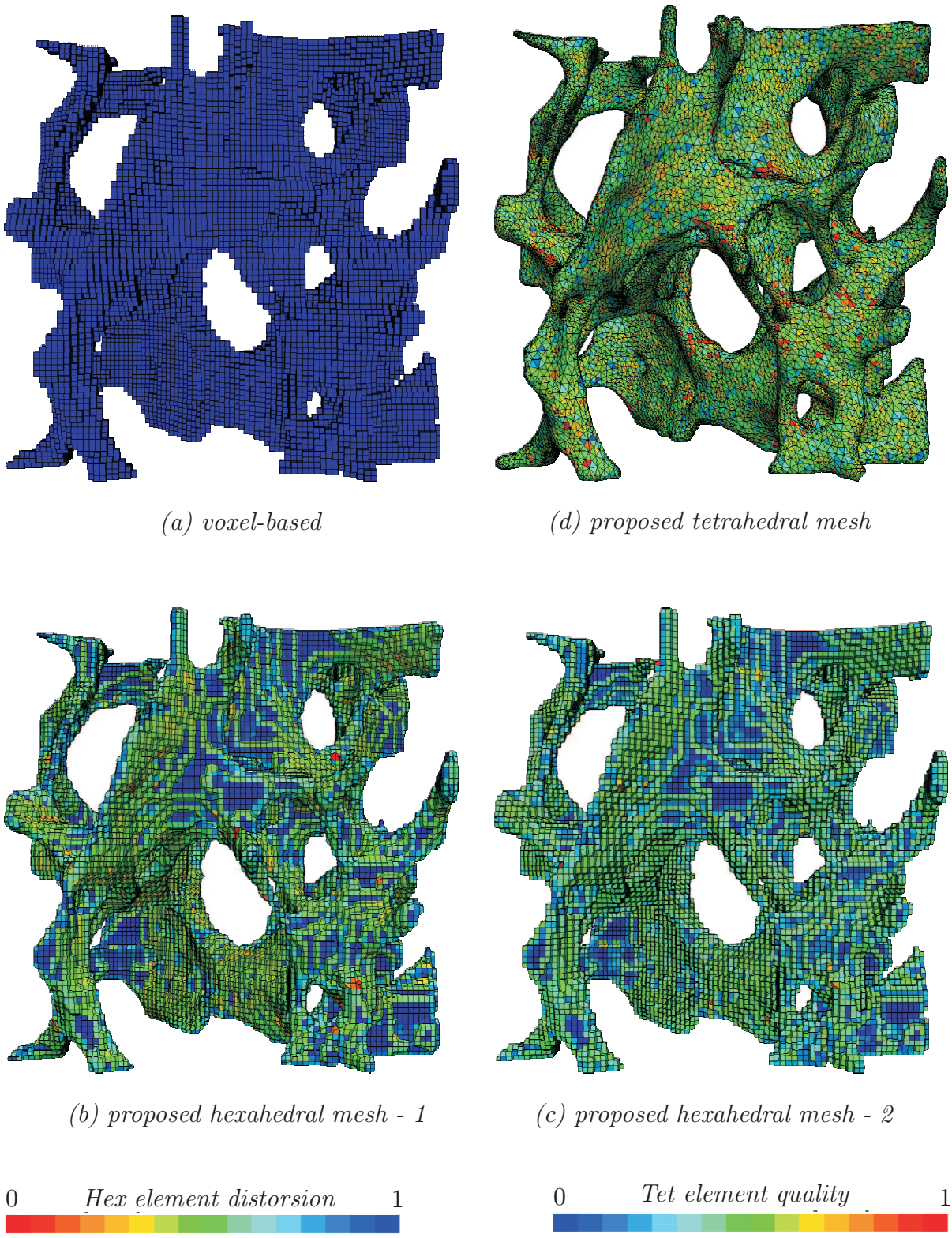
ber of foreground voxels in the input data, and therefore to the cancellous bone microstructure. Mesh (b) was obtained by smoothing mesh (a) with the algorithm proposed in Section 5.3 and specially developed to avoid mesh distortion and shrinkage. In contrast, classical smoothing algorithms would require many efforts and manual steps to avoid the collapsing of the trabeculae in the mesh. The proposed hexahedral meshing algorithm however generates a valid mesh automatically, without user-interaction. Mesh (c) is similar to mesh (b) but with a higher smoothing level. Mesh (d) is a tetrahedral volume mesh generated via the tetrahedral image-based meshing algorithm presented in this work (Chapter 4). It comprises 66013 nodes, and 232859 tetrahedra, which is slightly more than the 65144 nodes of the hexahedral meshes (a) (b) and (c). A more rigorous analysis of the obtained meshes is presented in Section 5.4.1. Let us just recall that the volumes of the different models are approximately identical.

10.1.3 Boundary conditions

During the finite element simulation, the bottom of the sample is rigidly fixed in all directions while the top of the specimen is forced to move downwards. The nodes located at the top of the specimen are forced to move downwards, until 10% of compression of the specimen's height is obtained; their displacement in the other directions is not constrained.

10.1.4 Material Properties

Material properties were obtained through the experimental testing of the sample represented in Figure 10.1 which was performed within the Laboratory of Chemical Engineering, Department of Applied Chemistry of the University of Liege [50]. From this compression test an apparent Young's modulus for the whole specimen was obtained: $E_{\text{app}} = 61.21$ MPa. From this apparent modulus of elasticity, the actual Young's modulus of the trabeculae was calculated using mathematical relations developed for the analysis of open cell pore topologies. The latter take into account the cell's relative density, which is 8.87 % for this sample. Using this approach, a Young's modulus of 7.8 GPa was found for the trabeculae. This value is in accordance with values reported in litterature. Indeed, Akhtar et al. [2] obtained a Young's modulus of 8.1 GPa. Furthermore, we used a Poisson's ration of $\nu = 0.3$, which is classical for bone trabeculae. Finally, a simple linear elastic material law was used in this study, mainly because no literature could be found on the non elastic behaviour of deer cancellous bone.



(a) voxel-based

(d) proposed tetrahedral mesh

(b) proposed hexahedral mesh - 1

(c) proposed hexahedral mesh - 2

FIGURE 10.2: ***Finite element study of deer antler cancellous bone.*** *Finite element meshes.*

10.2 Finite element simulations

Finite element simulations were performed using Metafor [111]. The use of a finite deformations code (geometrically and materially non-linear) was essential because the compressed samples showed evidence of localised large strains and large rotations.

Figure 10.3 shows the Von Mises stress fields, rendered on the final deformed meshes at 10% of compression. Observing the stress fields, we do not notice significant differences between the three hexahedral meshes. The stress distribution is however slightly different for the tetraheral mesh, with maximal Von Mises stresses located on different trabeculae. Figure 10.4 present the deformed models for the voxel-based and the tetrahedral mesh. The figure enables to visually compare the geometric difference obtained. Some local bucklings do occur in opposite directions in the two models. A comparison with experimental data should be made in order to determine which model better represents the real compressive behaviour of the sample.

The graph in Figure 10.5 presents the force-versus-displacement curves for the four micro-FE models. First, we obtain the interesting observation that smoothing of voxel-based meshes, by means of the specifically designed algorithm of Chapter 5, has the effect of lowering the apparent Young's modulus. Indeed, the slope of the force-displacement curves is lower for the smoothed hexahedral meshes than for the voxel based mesh. Computing the apparent Young's modulus at 2% of global compression for the four models we obtain: $E_{\text{app}} = 134.7$ MPa for the voxel-based mesh, $E_{\text{app}} = 126$ MPa for the hexahedral mesh with one level of smoothing, $E_{\text{app}} = 108.8$ MPa for the for the hexahedral mesh with two level of smoothing, and $E_{\text{app}} = 123.6$ MPa for the tetrahedral mesh. Therefore, the value obtained for the tetrahedral mesh lies in between the values of the hexahedral meshes, which is the case for small strains. For larger strains, from 3% of compression as reported on the graph, the tetrahedral model appears to be stiffer than the hexahedral models.

10.3 Conclusions

This example demonstrates that mesh generation is a crucial step in finite element modelling because it has a real influence on simulation results. From the same three-dimensional image, we generated four different meshes and analysed their influence on a compression test. For small deformations, at 2% of global compression, all models gave similar results. However, when deformations were increased, the tetrahedral model became stiffer than the hexahedral ones. Also, smoothing a voxel-based mesh decreased its global stiffness and the computed apparent Young's modulus of the model. Finite element models of other struc-

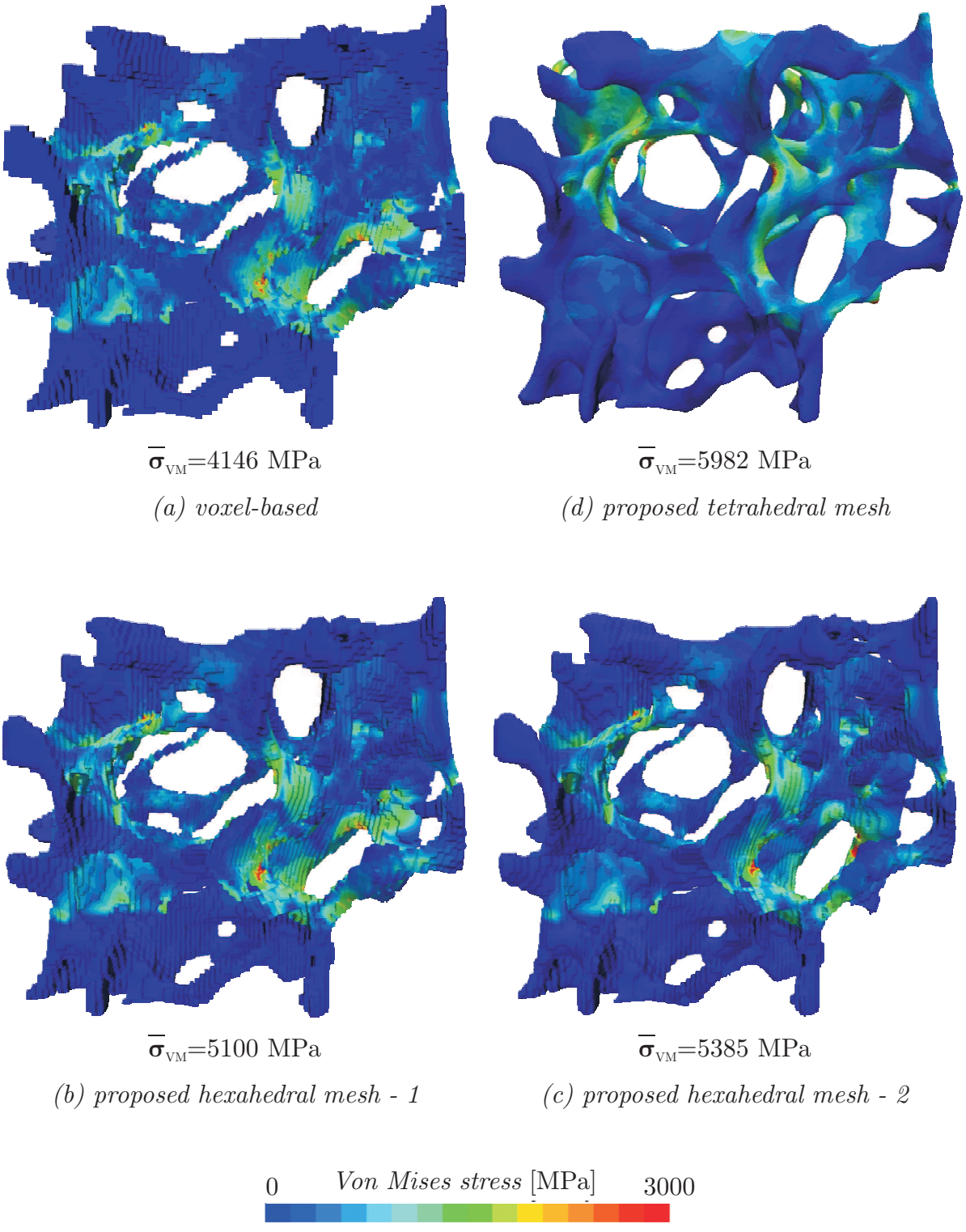
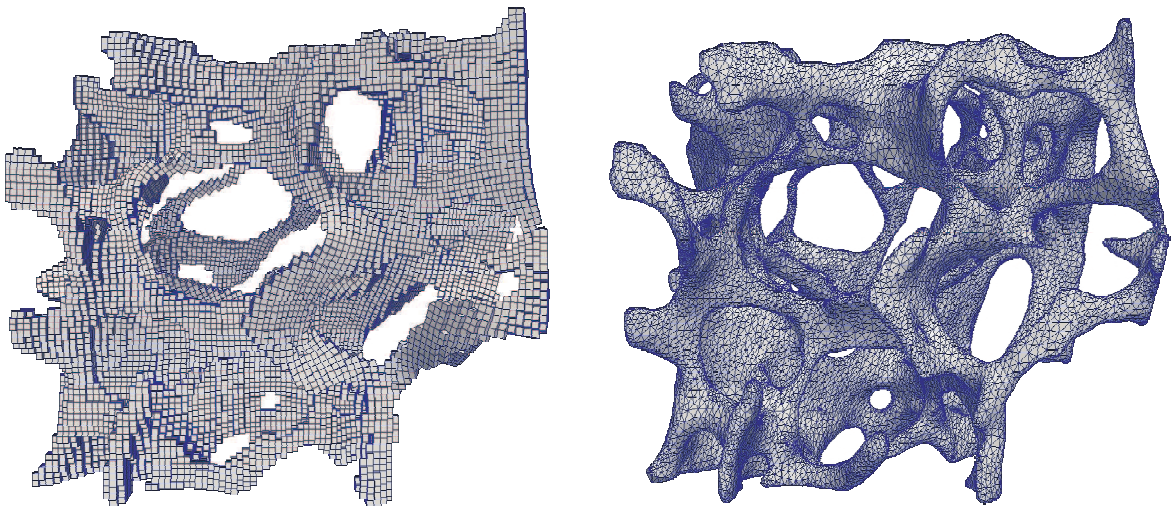


FIGURE 10.3: ***Finite element study of deer antler cancellous bone.*** *Von Mises stress fields for the four different finite element models considered.*



voxel-based *proposed tetrahedral mesh*

FIGURE 10.4: ***Finite element study of deer antler cancellous bone.*** *Deformed models at* 10% *of compression.*

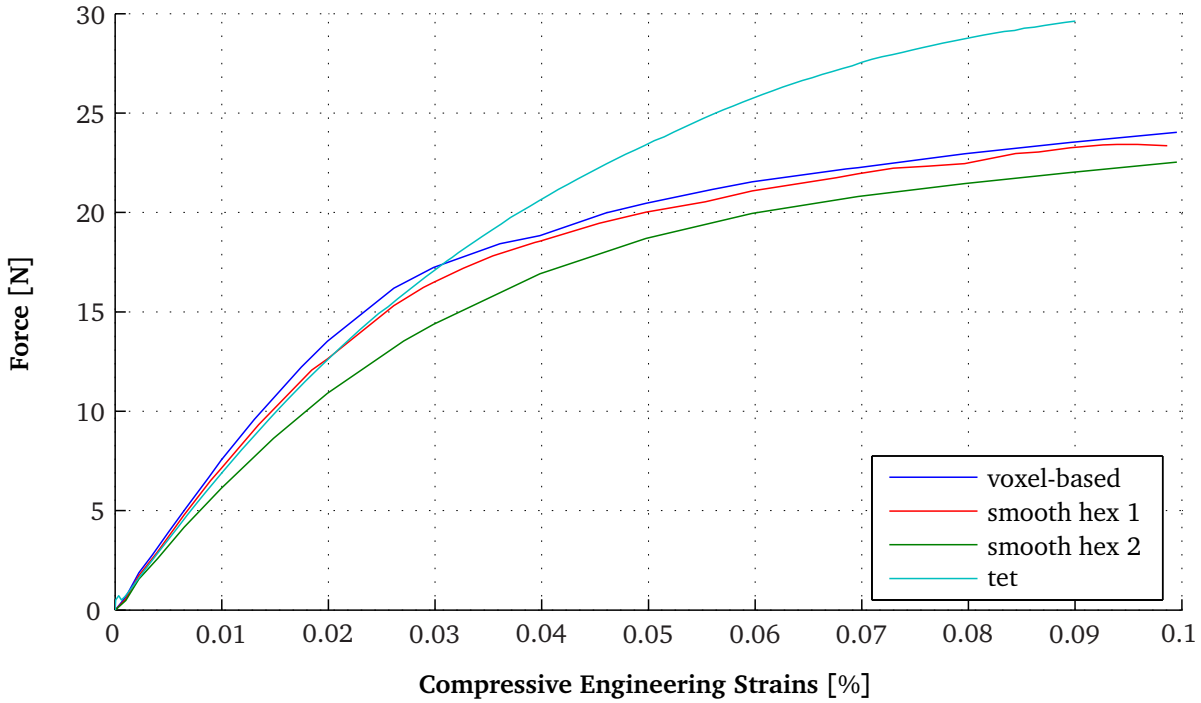


FIGURE 10.5: ***Finite element study of deer antler cancellous bone.*** *Computed force displacement curves for the four meshes considered.*

tures should be created, and simulations performed, in order to determine whether this conclusion is always valid. From the finite element simulations, we deduced an apparent stiffness of the sample comprised between 123.6 and 134.7 MPa, depending on the model used. This is higher than the experimental value of 61.21 MPa obtained for the larger sample. Therefore, the studied sample is probably no sufficiently representative of the larger sample. In the future, finite element simulations will be performed on a larger and more representative sub-sample. We will then be able to validate our results and determine the best meshing strategy for the micro-FE modelling of cellular tissues.

Chapter 11

Finite element modelling of brain shift deformation during image-guided neurosurgery

This chapter illustrates how image-guided neurosurgery may benefit from the present work. Indeed, a biomechanical model of the brain is able to provide the displacement field needed to improve the accuracy of image-guided neurosurgery systems. The idea originates from a previous work at the University of Liege [175]. Here, we improve the procedure by providing a new meshing approach and non-locking tetrahedral finite elements.

11.1 Context

Image-guided neurosurgery systems relate the 3D pre-operative images of the patient to the 3D patient's coordinates (Figure 11.1). This system enables the surgeon to position its instruments in the patient's brain by looking at the screen, where the current position of his surgical instrument is superposed on the patient's preoperative scans. However, the accuracy of this system is jeopardised by the fact that the brain deforms during the surgery, so that the preoperative scans displayed on the screen do not reflect the patient's current reality. The fall in accuracy begins when the surgeon opens the skull, which results in a leakage of cerebrospinal fluid and an equalisation of the pressures in and out the skull. This deformation, called *brain shift*, can be relatively important with displacements of the cortex of several millimetres. Because the brain deforms during the operation, first due to the skull opening and then due to surgical acts such as cuts, retractions and resections, the preoperative images become less and less representative.

FIGURE 11.1: ***Image-guided neurosurgery systems.*** *This technique relates the 3D pre-operative images of the patient to the 3D patient's coordinates.(CHU of Liège, Belgium).*

The idea developed at the University of Liege several years ago was to use a finite element model of the patient's brain in order to compute the intra-operative deformations [176]. The preoperative 3D scans can then be deformed accordingly. Obviously recording all forces and displacements imposed by the surgeon during the surgical intervention was unthinkable in a first step so that the registration technique was based on new images, taken intra-operatively. The latter reflect the actual brain, at a particular time of the operation. However the quality of these intra-operative images if far less than the high-quality of the pre-operative images. Furthermore, not all medical imaging modalities can be taken intra-operatively. Therefore, being able to calculate the deformation of the brain and successively deform the preoperative images to match the patient's actual reality is substantial.

Finite element computation of the intra-operative brain deformation is performed as follows. From the intra-operative images and with the help of the initial preoperative image, the current boundary conditions of the finite element model are deduced. Practically, the correspondence between several anatomical landmarks in the images is established and the displacement of these landmarks between both images is computed. These displacements are then applied to the finite element model of the patient's brain. The role of the finite element simulation is to calculate, from a sparse set of imposed displacements, the displacement throughout the brain. With the computed full displacement field, the preoperative images can be deformed.

The model of Vigneron [175] lacked an automatic meshing procedure to build the finite element model from the patient's segmented preoperative scans. Also, locking of the standard tetrahedron was observed during the finite element simulations, so that an elastic compressible material law had to be used. In this work, we use our original tetrahedral meshing procedure, presented in Part 1 of this thesis, to generate a tetrahedral mesh from the patient's 3D images. Moreover, volumetric locking is avoided by using the newly developed tetrahedral element from Part 2 of this work.

11.2 Building of patient-specific biomechanical model

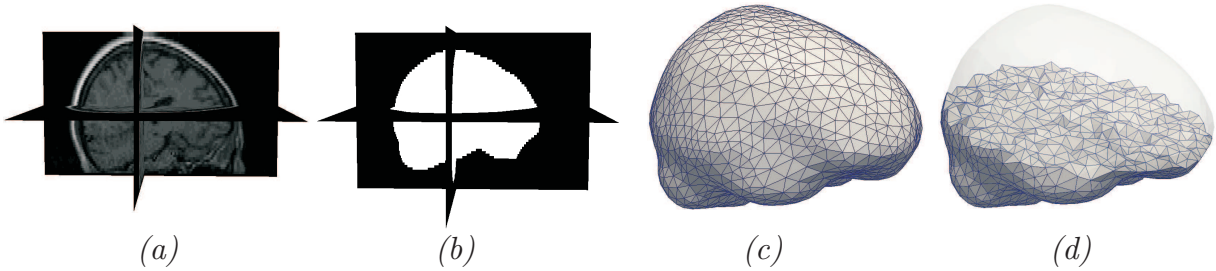
A finite element model of the brain was obtained by segmenting the pre-operative images, generating the finite element mesh, assigning material properties, defining boundary conditions and choosing an appropriate tetrahedral element formulation. Each of these steps are detailed in the following paragraphs.

Segmentation of the 3D medical image of the brain was performed semi-automatically using 3D Slicer [142]. The 3D image used is in fact an intra-operative Magnetic Resonance Image (iMRI) acquired with the 0.5 T intra-operative GE Signa scanner of the Brigham and Women's Hospital, Boston, USA (Figure 11.2 (a)). The iMRI image size is $256 \times 256 \times 60$ voxels and the voxel size is $0.9375 \times 0.9375 \times 2.5$ mm. The image was segmented into one region only, so that the result was a binary image. The result is shown in Figure 11.2 (b).

Mesh generation was performed using the algorithm illustrated in Figure 2.7, Section 2.5.1 and presented in details in Chapters 3 and 4. From the segmented image, our mesher outputs a tetrahedral mesh of the volume without further user interaction; even though the meshing algorithm consists of three specific steps: (1) geometry extraction in the form of an analytic function, (2) triangulation of the brain cortex, (3) creation of a tetrahedral mesh of the volume from the triangular surface mesh of the closed surface obtained in step 2. The resulting mesh, shown in Figure 11.2 (d), comprises 4759 nodes and 24585 tetrahedrons.

An appropriate finite element formulation is used for the tetrahedral finite element. The formulation developed in Chapter 7.4 solves the problems of excessive stiffness of the model under quasi-incompressibility constraints. For comparison purposes, both the standard tetrahedron and our new formulation will be used in the finite element simulations here-after.

Linear elastic material properties are assigned to the model. In the initial work, Vigneron [175] used a compressible material with $\nu = 0.45$ to model brain tissue, even



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

FIGURE 11.2: ***Tetrahedral mesh generation of a brain MRI.*** *(a) Initial 3D iMRI image. (b) Segmented 3D image. (c) Volume mesh obtained by application of the meshing method developed in this thesis work in the segmented 3D image.*

though the incompressibility of brain media is widely adopted and recently demonstrated (see e.g. [105] and references therein). Thanks to the non-locking tetrahedral element developed and implemented in this work, a quasi-incompressible material law with $\nu = 0.49995$ may now be used. It is believed that this modification will improve the accuracy of the obtained deformation field and henceforth, the reliability of non-rigid registration method.

The boundary conditions are applied identically to Vigneron [175], i.e. through an imposed displacement of the cortex. The latter was obtained by recording the displacement of the brain surface into two successive intra-operative images: one taken just before skull opening and the other taken after the opening of the skull, i.e. after brain-shift. Details on how to obtain this displacement field may be found in [62, 175]. It is important to realise that these boundary conditions are not very realistic. The brain-shift deformation, occurring at the opening of the skull, is far more complex than the imposed displacement of a surface. It is a combination of cerebrospinal fluid leakage and drainage, and, in the case of a tumour, brain pressure release. However, imposing cortex displacements is still, at the current state of research, the most popular solution for the finite element modelling of brain-shift [184].

11.3 Finite element simulations and results

Four finite element simulations were performed:

- with a compressible linear elastic material law $\nu = 0.45$ and $E = 3$ MPa; and the standard tetrahedral element formulation
- with a compressible linear elastic material law $\nu = 0.45$ and $E = 3$ MPa; and the proposed non-locking tetrahedral element formulation

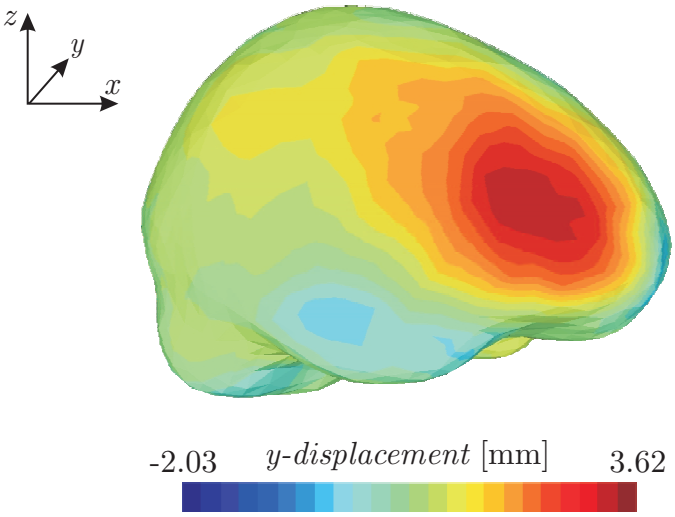


FIGURE 11.3: ***FEM-based modelling of brain shift deformation in Image-Guided Neurosurgery.*** *Applied displacement field.*

- with a quasi-incompressible linear elastic material law $\nu = 0.49995$ and $E = 3$ MPa; and the standard tetrahedral element formulation
- with a quasi-incompressible linear elastic material law $\nu = 0.499905$ and $E = 3$ MPa; and the proposed non-locking tetrahedral element formulation

In each case, the displacement field resulting from a previous work [175] was applied to the mesh nodes belonging to the brain exterior surface. A displacement vector is applied to all the boundary nodes, but varies in direction and magnitude from node to node. It was deduced by taking MRI images taken before and after opening of the skull and computing the displacement of the cortex between both images. The applied displacement field is shown in Figure 11.3. The displacement field is mainly applied along the y-direction, which is the direction shown on the figure. We may deduce that in the considered surgical intervention there was a brain-shift of approximately 3.62 mm.

Figure 11.4 shows the pressure fields obtained for the four finite element simulations performed. Let us remind that in this thesis, the pressure is defined as negative one third of the trace of the stress tensor

$$p = \frac{\mathrm{tr}\left(\sigma_{11} + \sigma_{22} + \sigma_{33}\right)}{3} \tag{11.1}$$

Therefore, the pressure is negative in compression and positive in extension.

The top figures of Figure 11.4 were obtained for a compressible brain material. In that case, the standard linear tetrahedron behaves well and there is no interest in using our non-locking tetrahedron as both formulations give solution. For a quasi-incompressible

material law however, we obtain a sporadic pressure field when using the classic, standard, tetrahedral element formulation, as shown in Figure 11.4, Lower Left. Using the proposed non-locking tetrahedron allows us to recover a realistic pressure distribution (Figure 11.4, Lower Right).

The pressure values computed in the quasi-incompressible case, $\nu = 0.49995$, lower figures, are much higher than those obtained in the compressible case, $\nu = 0.45$, upper figures. The reason for this is that the whole boundary, the brain cortex, is constrained via imposed displacements. In future simulations, compressible ventricles could be added to the model in order to model a release of cerebrospinal fluid. This would also lead in a drop in the observed extremal pressure values.

Figure 11.5 shows, for a specific slice, the displacement field obtained throughout the brain's volume. When the objective of the simulation is the non-rigid registration of brain intra-operative images, these volume displacements are the sole requested output of the simulation. The obtained maximal displacement values are indicated on the corresponding locations on the charts. As expected taking the incompressibility of the brain into account has an influence on the obtained displacement results. Results indicate that the displacement values were under-estimated by 1.3% in the model of [175]. Furthermore, using an approprate tethrahedral element for the quasi-incompressible model is of equal importance. Indeed, the two charts on the right indicate that with the standard linear tetrahedron results are over-estimated by 1.6%.

11.4 Conclusions

In this chapter, we created a patient-specific finite element model of the brain and used the model to simulate the brain shift deformation occurring after skull opening during a surgical intervention. The overall objective is to improve the accuracy of current image-guided neurosurgery systems. Practically, intra-operative brain deformation was modelled using a similar approach as Vigneron [175], but, two significant improvements were made. First, the required finite element mesh was generated by the tetrahedral mesh generator implemented in this work, thus removing the many manual steps of the previous approach. The resulting gain in automaticity will be even more relevant in a future work when multiple anatomical structures will be taken into account in the model. Second, the brain tissue was modelled using a quasi-incompressible material law, more realistic than the compressible one, previously used.

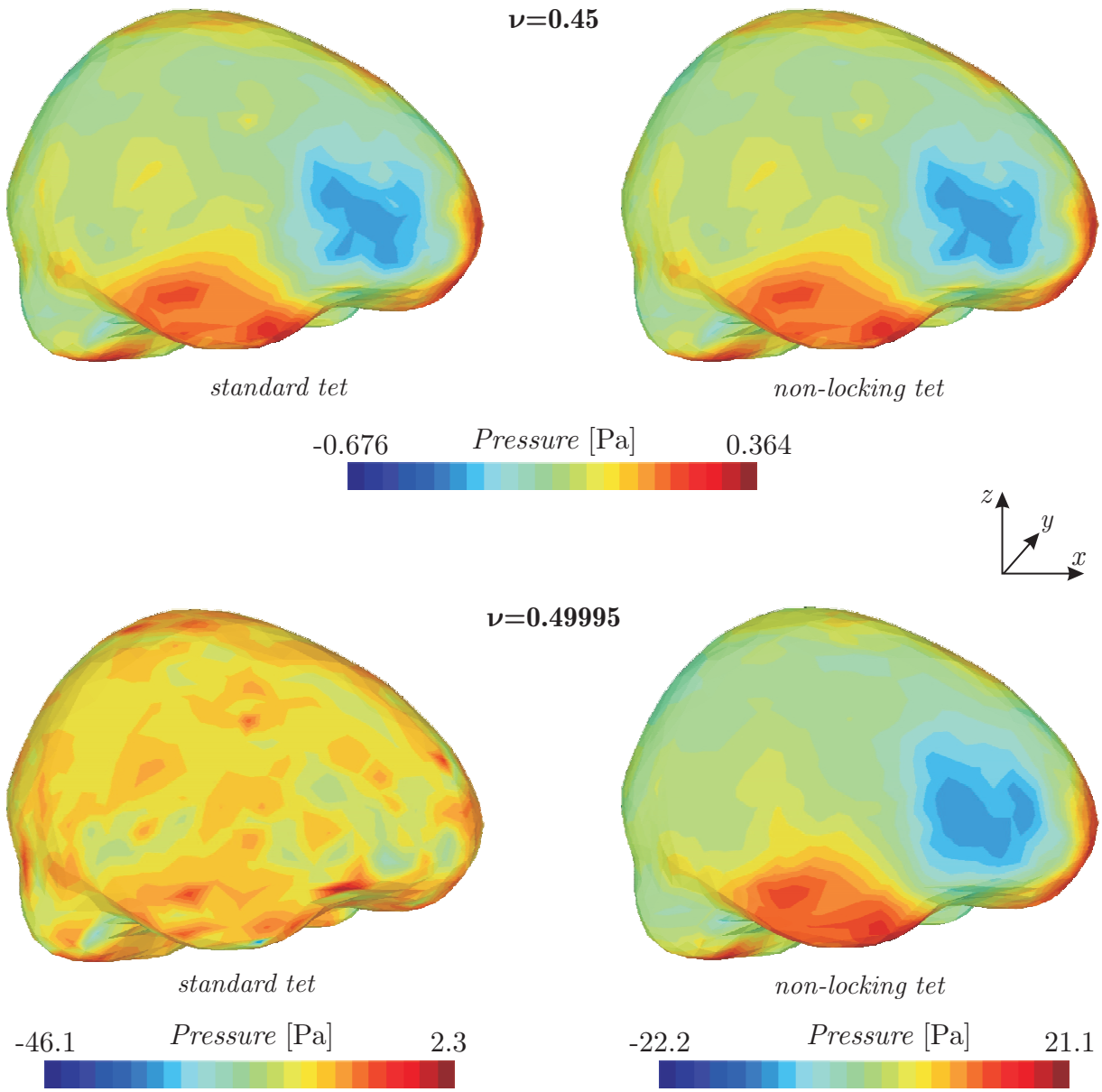


FIGURE 11.4: ***FEM-based modelling of brain shift deformation in Image-Guided Neurosurgery.*** *Obtained pressure fields for the four models considered: with a compressible and incompressible linear elastic material law, and with the standard or our non-locking tetrahedral element formulation.*

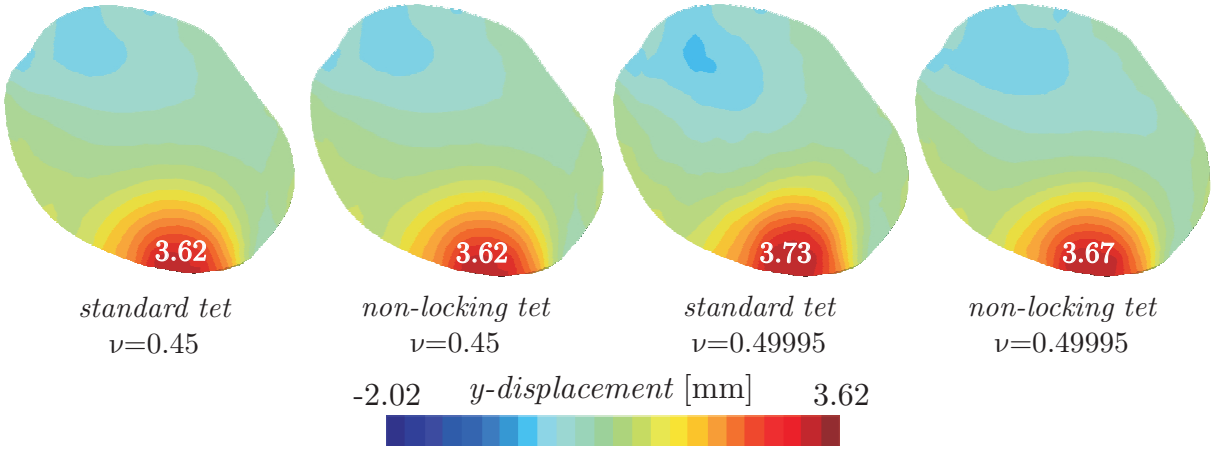


FIGURE 11.5: ***FEM-based modelling of brain shift deformation in Image-Guided Neurosurgery.*** *Volume displacement fields drawn in a brain xy-slice selected at $z = 65$ mm for the four models considered: with a compressible and incompressible linear elastic material law, and with the standard or our non-locking tetrahedral element formulation.*

Chapter 12

Modelling of canine humeral condylar fractures

Condylar fractures are among the most frequent humeral fractures seen in dogs after a fall [19]. Different types of fractures (lateral, medial, bicondylar) may occur, depending on the age of the dog and the position of the elbow during the impact. The goal of this work is to understand the effects of bone positioning and skeletal development on canine humeral fractures by means of the finite element method, using the developments of the previous chapters.

Lateral and Medial condylar fracture *Bi-condylar "Y" and "T" fractures*

FIGURE 12.1: ***Classification of humeral condylar fractures.*** *Source: Moores [122].*

12.1 Context

Humeral condylar fractures are common in dogs and are often associated with minor trauma such as fall. Lateral condylar fractures are most common, while medial and bicondylar ("Y" or "T") fractures occur less frequently [19]. These three types of humeral fracture occurring in dogs are illustrated in Figure 12.1. Moreover, lateral fractures are most prevalent in young dogs, before the ossification of the humeral condyle [115]. On the other hand, bicondylar fractures are typically seen in skeletally mature dogs. It is believed that lateral condylar fractures are due to an excessive force carried by the radius, which articulates with the lateral part of the humerus.

The objective of this study is to verify the pathogenesis of condylar fractures and to determine the influence of bone positioning as well as skeletal development on the fracture type; by means of the finite element method.

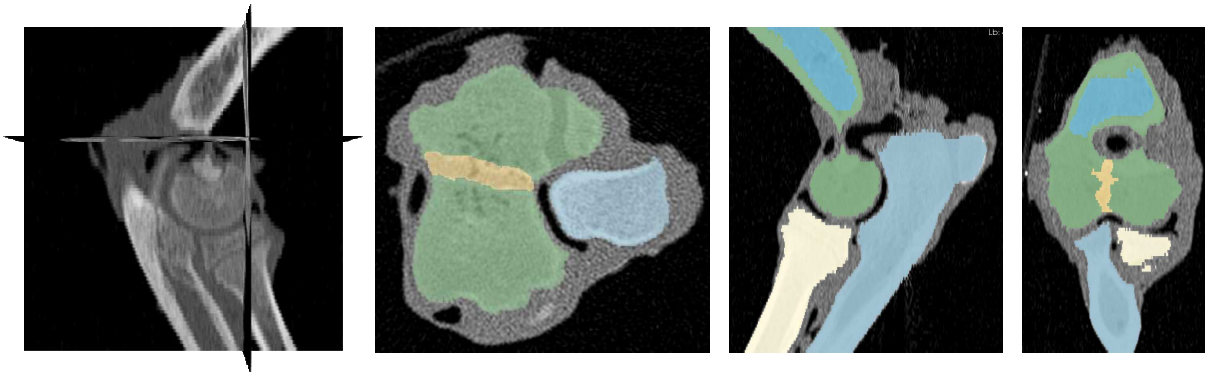
12.2 Dataset preparation

12.2.1 Image acquisition

A computed tomographic scan of the right forelimb of a four months old beagle was taken at the Veterinary School of the University of Liège. The elbow was scanned in a physiological position, with a flexion-extension angle of 150°. Three bones were represented in the image, the humerus, the ulna and the radius. However, only the extremities of these three bones were scanned (approximately one third). The input data is illustrated in Figure 12.2, Left. It has dimensions of $512 \times 512 \times 88$ voxels with anisotropic voxels of $0.115234 \times 0.115234 \times 0.699951$ mm^3. Acquiring high resolution scans was important for this application in order to capture the elbow joint. With a higher out-of-plane spacing, the three bones would appear fused in the scans, which would jeopardize the accuracy of the segmentation and the accuracy of the simulation.

12.2.2 Segmentation

Segmentation of the dataset was performed with 3D Slicer [142] using the following procedure. This three-dimensional image was segmented in order to delineate the three bones composing the elbow: humerus, radius and ulna. The humerus was further subdivided into cortical bone, trabecular bone, medullary cavity and cartilaginous plate. Because no



Initial dataset *Segmentation*

FIGURE 12.2: ***Modelling of a dog elbow fracture.*** *Left: Initial dataset. Right: Segmentation of the dataset, the humerus cortical bone is represented in green, the humerus cancellous bone in dark blue, the growth plate in yellow, the radius in white and the ulna in light blue.*

scans were available for adult dogs, the adult dog model was created from the young dog model by replacing the cartilaginous growth plate in the multi-valued segmented image by epiphyseal trabecular bone.

12.3 Finite element study of the influence of elbow configuration on fracture type

In a first study, the effects of flexion/extension, radioulnar exo/endo-rotation (rotation of radius and ulna around the longitudinal axis of the humerus) as well as abduction/adduction (angle between the direction of loading and the longitudinal axis of the humerus) on the fracture type have been studied in order to determine the conditions under which lateral, medial and bicondylar humeral fractures occur. These angles are represented in Figure 12.3. Because of the number of cases involved (three different angles and three possible values for each angle) we only consider the skeletally mature dog for this study. Also, no failure criterion is yet taken into account. The comparison of adult and young dog, as well as the inclusion of a failure criterion, will be performed on a smaller set of possible elbow configurations, in Section 12.4.

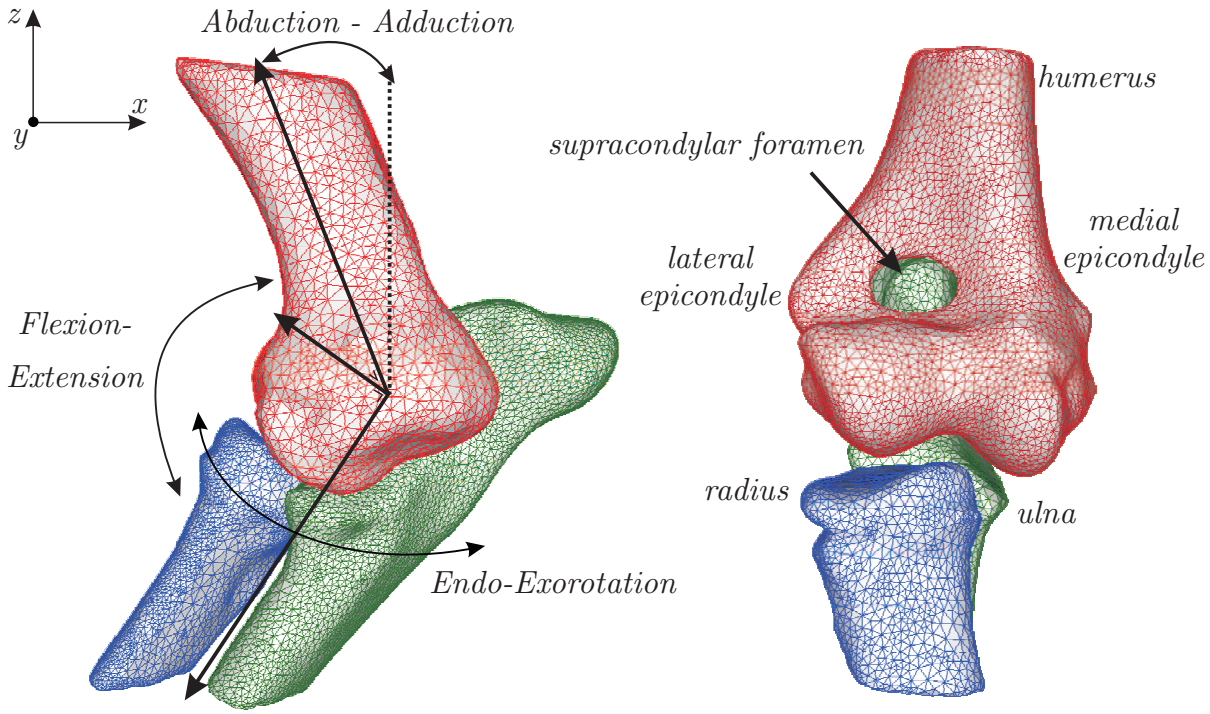


FIGURE 12.3: ***finite element study of the influence of elbow configuration on fracture type.*** *Left: Definition of flexion/extension, radioulnar exo/endo-rotation and abduction/adduction angles. Right: Anatomy of the dog elbow.*

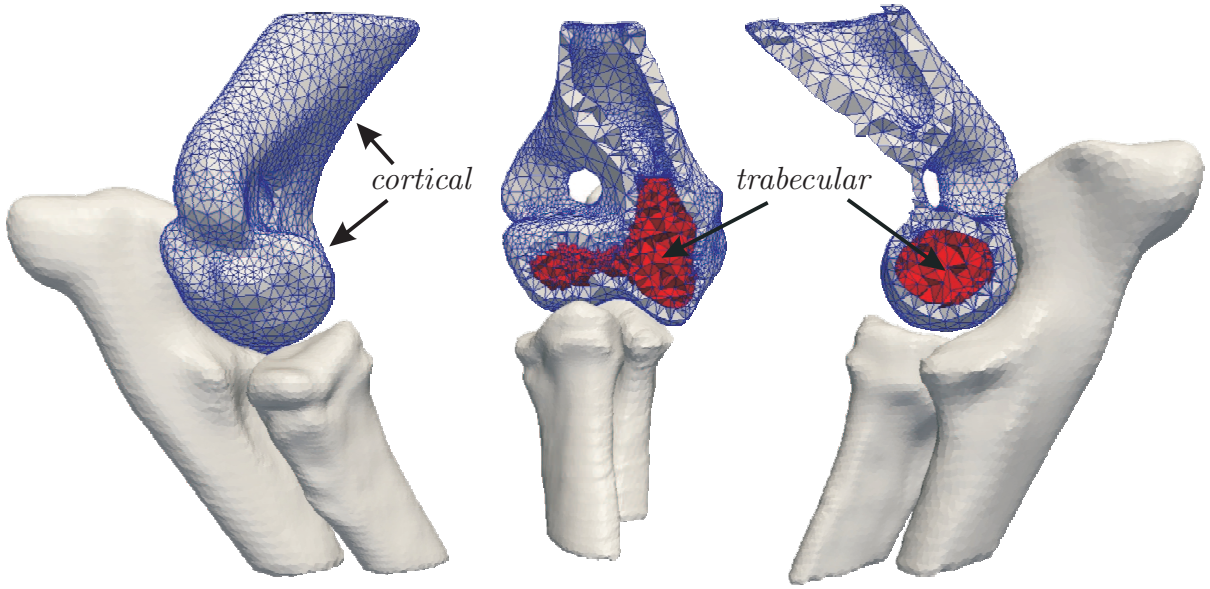


FIGURE 12.4: ***Finite element study of the influence of elbow configuration on fracture type.*** *Finite element mesh obtained from the CT-data. The humerus is subdivided into two material regions, cortical bone and trabecular bone. The medullary cavity was left empty in this model. In middle and right pictures, we performed a cut through the volume mesh of humerus in order to show its inner structures.*

TABLE 12.1: ***finite element study of the influence of elbow configuration on fracture type.*** *Mesh sizes.*

	nodes	**cells**
Humerus	53239 nodes	293307 tetrahedra
- *Cortical*	26397 nodes	192615 tetrahedra
- *Trabecular*	26841 nodes	100692 tetrahedra
Radius	1336 nodes	2668 triangles
Ulna	1657 nodes	3310 triangles

12.3.1 Mesh generation

The multi-material tetrahedral mesher presented in Chapter 4 of this work was applied on the segmented image of the dog elbow in extension. For the humerus, a multi-material mesh was created. For this first study, we decided to represent the cortical and epiphyseal trabecular bone only and leave the medullary cavity empty because the sparse medullary trabecular bone does not really participate in the load transfer. For the radius and the ulna, only a surface mesh of the boundaries of the bones was created. These surface meshes are needed in the simulations to define the radiohumeral and the humeroulnar contacts. The obtained model is represented in Figure 12.4, and the corresponding mesh sizes are reported in Table 12.1.

12.3.2 Finite element modelling

To model different elbow configurations, the meshes of the ulna and the radius were rotated relative to each other, and the axis of the applied load was changed. The axes of rotation were defined through the localisation of anatomical landmarks in the model and with the help of veterinary surgeons. In further studies, these different configurations will be obtained through comparison with 3D scans of the elbow in the desired configuration, in order to ensure a physiological positioning of the elbow.

Finite element simulations at 60, 130 and 150 degrees of flexion-extension, -10, 0 and 10 degrees of exo-endorotation angle and -20,0 and 20 degrees of adduction-abduction angle were performed.

TABLE 12.2: ***finite element study of the influence of elbow configuration on fracture type.*** *Material parameters used in the first study.*

	Young's modulus (MPa)	**Poisson's ratio**	**Yield stress (MPa)**	**Hardening Parameter (MPa)**
Cortical bone	2660	0.3	100	266
Epipyseal trabecular bone	2110	0.3	19.1	105

After correct bone positioning, a vertical displacement was applied on the radius and ulna at a speed of 140 mm/min. This speed is too slow to model an impact, but it is the maximum speed allowed by the experimental testing facility that will be used at the University of Liège to validate these results.

The radius and the ulna are considered as rigid body, to simplify the model and because we are only interested in the stresses and strains within the humerus. Their surface meshes, extracted from the CT images, enable us to define their contact with the humerus.

A frictionless contact is defined between the humerus and the ulna as well as between the humerus and the radius. The contact is modelled using the penalty method, which allows to soften the contact between the bones and therefore implicitly take the articular cartilage into account.

A dynamic implicit Chung-Hulbert time integration scheme is used.

Material parameters were taken from literature [92] and are reported in Table 12.2. Both cortical and trabecular bone are modelled by elasto-plastic material laws.

12.3.3 Results and discussion

Simulations were performed until the failure Von Mises stress of cortical bone in compression was reached in an element. For all configurations, we reported the bone entering first into contact with the humerus (radius or ulna or both), and we tried to deduce the most probable fracture pattern from the observation of the Von Mises stress fields. Both informations are reported in Table 12.3. As already noticed, no failure criterion is implemented for this preliminary study.

TABLE 12.3: ***Finite element study of the influence of elbow configuration.*** *For various configurations of the dog elbow, deduction of the most probable fracture pattern from the observation of the Von Mises stress field within the humerus, computed by finite element simulations.*

Flexion Extension	**Endorotation(+) Exorotation(-)**	**Abduction(+) Adduction(-)**	**Bone Contact**	**Fracture Type**
60°	0°	0°	Ulna	Lateral
60°	0°	-20°	Ulna	Lateral
60°	0°	20°	Ulna	Medial
60°	10°	0°	Ulna	Y
60°	10°	-20°	Ulna	Y or Lateral
60°	10°	20°	Ulna	Medial
60°	-10°	0°	Ulna	Lateral
60°	-10°	-20°	Ulna	Lateral
60°	-10°	20°	Ulna	Lateral
130°	0°	0°	Ulna	Medial
130°	0°	-20°	Radius and Ulna	Lateral
130°	0°	20°	Ulna	Medial or Y
130°	10°	0°	Radius	Lateral
130°	10°	-20°	Radius	Lateral
130°	10°	20°	Radius and Ulna	Y or Medial
130°	-10°	0°	Radius	Lateral
130°	-10°	-20°	Radius	Lateral
130°	-10°	20°	Radius	Lateral
150°	0°	0°	Radius and Ulna	Medial
150°	0°	-20°	Ulna	Lateral
150°	0°	20°	Ulna	Medial
150°	10°	0°	Ulna	Lateral
150°	10°	-20°	Ulna	Lateral
150°	10°	20°	Ulna	Y
150°	-10°	0°	Radius	Lateral
150°	-10°	-20°	Radius	Lateral
150°	-10°	20°	Radius	Lateral

The results reported in Table 12.3 confirm the clinical observation that lateral condylar fractures occur most frequently. Medial and Y fractures do occur for some configurations, and are always caused by the interaction of the ulna with the humerus. These results may be explained as follows. The radius articulates with the lateral part of the humerus and thus causes lateral fractures in all cases; the ulna however articulates with the central part of the distal humerus and may cause either lateral, medial or more complex fractures. Moreover, the lateral condyle of the humerus is more fragile, as its cross-section, measured as the distance between the supracondylar foramen and the surface of the lateral epicondyle (see Figure 12.4), is thinner; which is the reason why lateral condylar fracures are predominant, even in flexion where the ulna is the only bone impacting the humerus.

Figure 12.5 illustrate the obtained Von Mises stress fields in six of the considered cases. Even though some elbow configurations lead to stress fields that clearly indicate a higher solicitation of the lateral, or the medial, condyle; it is not always easy to determine the fracture type from the observation of the Von Mises stress field. Also, we do not observe high stresses between the articular surface and the supracondylar foramen, where the fracture is suppose to initiate. This is due to the limitations of our model. First, the model that was considered here is an adult dog model for which the bone is fully formed. However, condylar fractures are most often observed in young dogs for which the bone is not fully formed in this location. Second, the Von Mises stress is not the right variable to assess failure: the difference of bone strength in tension and in compression should be taken into account. These two limitations are solved in the next section.

12.4 Finite element simulation of a condylar fractures

12.4.1 Introduction

Do humeral condylar fractures occur in flexion or in extension of the elbow ?

In this second study, we have performed four finite element simulations, in two configurations: in extension (150°) and in flexion (60°) and for skeletally mature and skeletally immature bones. Through the implementation of a failure criterion, we simulate the fracture of the bone. The failure load, the pattern of the fracture, its initiation point and propagation were recorded.

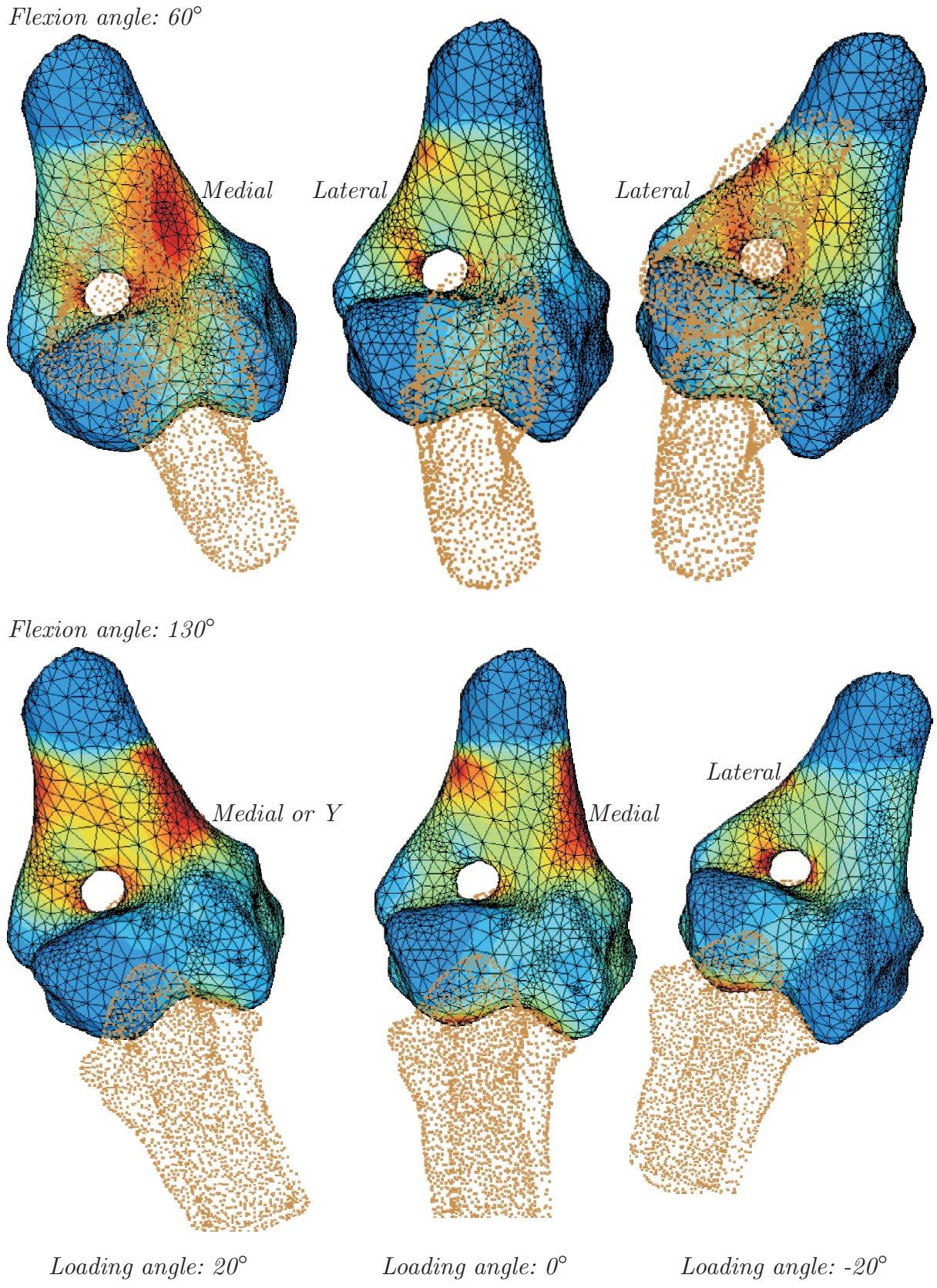


FIGURE 12.5: ***Finite element study of the influence of elbow configuration.*** *Results of the finite element simulations with an exo-endorotation angle of 0 degrees.*

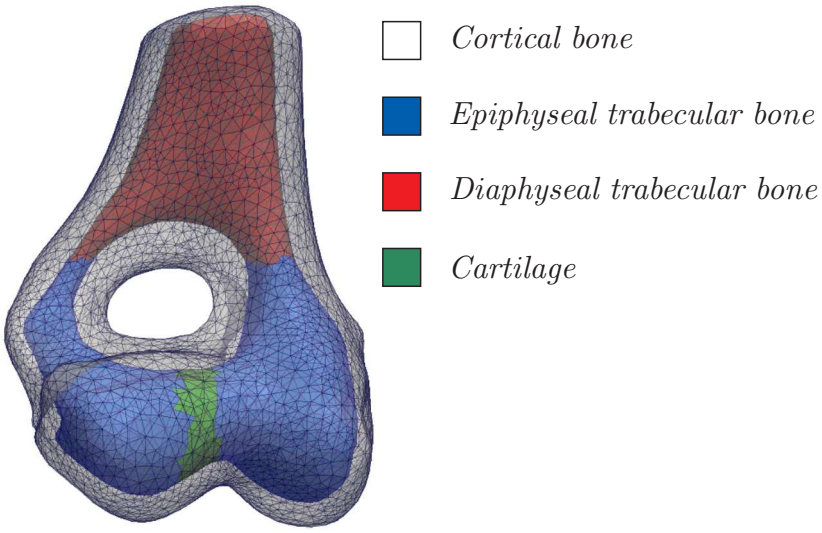


FIGURE 12.6: ***Modelling of a dog elbow fracture.*** *Finite element mesh obtained from the CT-data. The humerus is subdivided into four material regions, corresponding to cortical bone (semi-transparent), trabecular bone (blue), medullary cavity (red) and cartilage (green).*

12.4.2 Mesh generation

Finite element meshes were generated from the segmented multi-label 3D images presented in Section 12.2 using the multi-material tetrahedral mesher proposed in this work. Two different meshes of the humerus, young and adult, were generated:

The skeletally immature humerus model (young dog) comprises four distinct regions, with different material properties: cortical bone, epiphyseal trabecular bone, diaphysial trabecular bone and cartilage.

The skeletally mature humerus model is constituted by three distinct regions, with different material properties: cortical bone, epiphyseal trabecular bone and diaphysial trabecular bone.

Figure 12.6 illustrates the mesh obtained for the skeletally immature humerus. In the skeletally mature one, the cartilaginous growth plate is replaced by epiphyseal trabecular bone.

12.4.3 Finite element modelling

Separating the mesh in different regions enables us to apply distinct material properties to cortical bone, trabecular bone and cartilage. The cortical bone is modelled as a elastoplastic

transversely isotropic material, having a lower Young's modulus in the transverse direction than along the longitudinal axis [54, 92]. Epiphyseal and diaphysial trabecular bone are both modelled as an elastoplastic isotropic materials, but lower Young's modulus, yield stress and hardening parameter are used for the diaphyseal bone [92]. For both cortical and trabecular bone, a Von Mises Plasticity criterion is used with an isotropic linear hardening. The cartilage is modelled as a linear elastic material. Table 12.4 gives the material parameters used in our finite element simulations.

The modified Mohr-Coulomb failure criterion is used to assess the failure pattern of the humerus. This failure criterion enables us to apply a lower failure stress in tension than in compression. Indeed, experimental results have shown that the Von Mises failure criterion, because it assumes equal strength in tension and in compression, was not able to clinically reproduce the observed failure patterns [61].

The Mohr-Coulomb criterion, commonly used for materials with different behaviour in tension and compression, is written as

$$\frac{\sigma_1}{\sigma_t} - \frac{\sigma_3}{\sigma_c} = 1 \tag{12.1}$$

with $\sigma_1, \sigma_2, \sigma_3$ ($\sigma_3 \leq \sigma_2 \leq \sigma_1$) the principal stresses, σ_c, the failure stress in compression and σ_t, the failure stress in tension. This failure criterion is traditionally used for non-cohesive materials such as soils. Keyak and Rossi [93] first used this criterion for bone tissue and obtained a good agreement with experimental results. For materials for which the ultimate strength in tension is less than half its strength in compression, $\sigma_t \leq 0.5\sigma_c$, Keyak and Rossi [93] obtained, for the specific case of the modelling of bone fracture caused by fall, better results using the modified Mohr-Coulomb failure criterion [157]:

$$\begin{cases} \frac{\sigma_t}{\sigma_1} = 1 & \text{when} \quad \frac{\sigma_1}{\sigma_3} \leq -1 \\ \frac{\sigma_c - \sigma_t}{\sigma_c \sigma_t} \cdot \sigma_1 - \frac{\sigma_3}{\sigma_c} = 1 & \text{otherwise} \end{cases} \tag{12.2}$$

A extended review of bone failure criteria can be found in Doblaré et al. [54].

Boundary conditions, contact, loading and time integration are identical to the first study. These are detailed in Section 12.3.2 and illustrated in Figure 12.7, Left.

12.4.4 Results and discussion

Lateral humeral fractures were observed for both the skeletally mature and skeletally immature dog elbow in extension, see Figure 12.7. These fractures occurred at an applied load of 2.49 kN for the young dog, and 2.61 kN for the adult dog. The radius impacted

TABLE 12.4: ***Finite element modelling of a dog elbow.*** *Material parameters used in the finite element simulations .*

	Young's modulus (MPa)	**Poisson's ratio**	**Yield stress (MPa)**	**Hardening Parameter (MPa)**	**Failure stress (MPa)**
Cortical bone	long.: 2660 trans.: 1596 shear: 570	long.: 0.3 trans.: 0.3	100	266	compression: 186 tension: 93
Epipyseal trabecular bone	2110	0.3	19.1	105	compression: 21 tension: 10.5
Diaphyseal trabecular bone	1055	0.3	9.6	52	compression: 10.5 tension: 5.2
Cartilage	1	0.45	-	-	0.015

the humerus first and was then followed by the ulna. The cracks initiated on the articular surface, between the lateral and medial condyles and then extended towards the supracondylar foramen. A second crack then appeared on the lateral part of the supratrochlear foramen and propagated through the lateral epicondyle (the anatomy of the dog humerus is recalled in Figure 12.3).

The typical fracture patterns described in Section 12.1 were not obtained for the models in flexion. Indeed, in these cases, two cracks initiated on the lateral and medial parts of the supratrochlear foramen and then propagated through lateral and medial epicondyles simultaneously. No elements were fractured on the articular surface. Obtained failure loads are also higher compared to the simulations in extension: 3.6 kN for the adult humerus and 3.5 kN for the immature humerus. A possible explanation for this is that canine elbow fractures occur in extension of the elbow, and not in flexion.

12.5 Conclusions and future work

In this chapter we investigated an actual question of veterinary surgeons through finite element modelling. The methods developed for this thesis work associated with an efficient collaboration with the veterinary school of the university of Liège have made it possible to solve the whole process of problem well-posedness, image acquisition, segmentation,

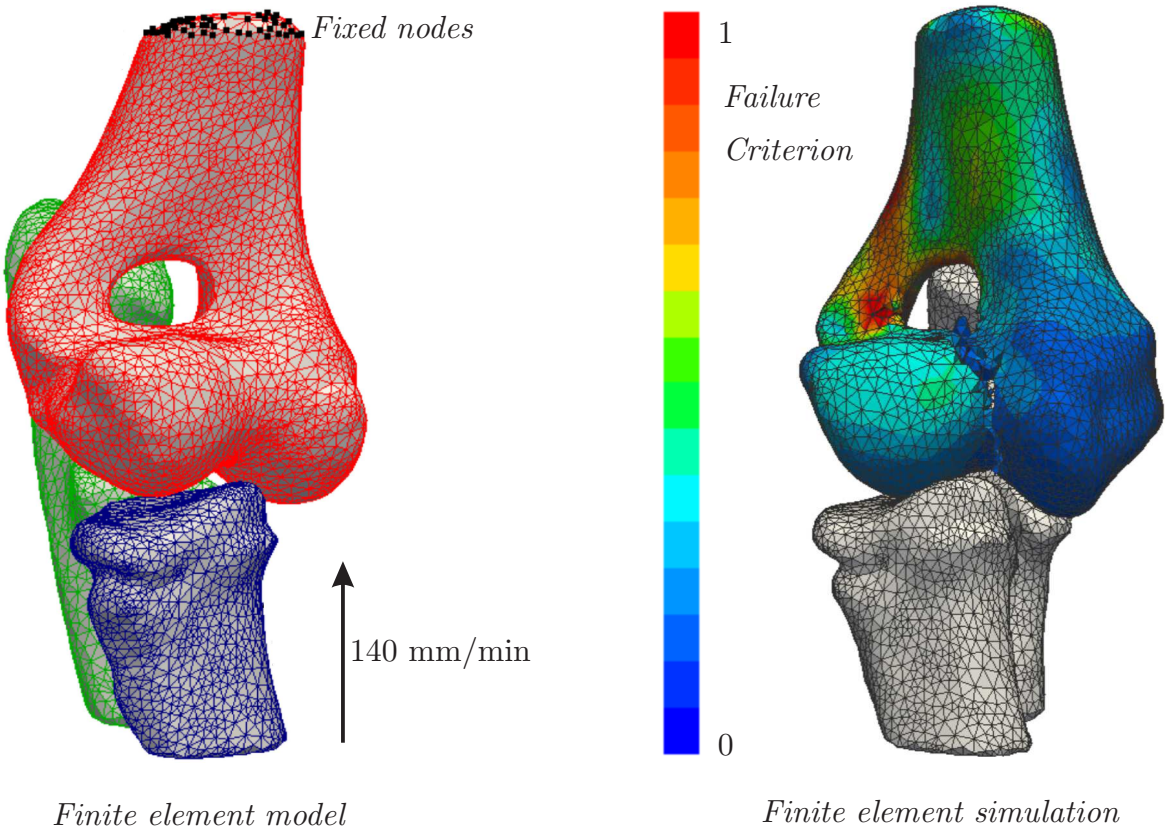


FIGURE 12.7: ***Finite element modelling of a dog elbow.*** *Left: Applied boundary conditions. Right: Lateral condylar fracture obtained as a result of the finite element simulation.*

mesh generation and model creation, finite element simulations and analysis of the results. However, this study is only a preliminary study, and reliable results will only be obtained after the consideration of the following:

- A correct modelling of the bone orthotropy. The bone is a composite material constituted by collagen fibres in a mineral matrix. Therefore, the Young's modulus in the direction of the fibres is higher than in the transverse direction. In the above second study, this anisotropy was taken into account. However, all fibres were assumed to lie in the same direction: along the longitudinal axis of the humerus. Instead, this direction should be extracted from imaging data and should be allowed to vary locally through the humerus.
- The literature on elasticity and strength properties of canine bone is quite poor. Mechanical tests should be performed in order to determine adequate material parameters for our model. Also, distinct elasticity and strength properties should be used for the young and adult dog models.

- In this preliminary work we used the modified Mohr Coulomb failure criterion to model bone failure. With the advances in bone failure research, other failure criteria could be investigated.

- What really determines the stress-strain distribution within the humerus is its contact with the radius and the ulna at the elbow joint. Therefore, it is very important to model this joint accurately. Our meshing algorithm is well adapted for this problem, because it allows this accurate representation of the geometries. However, in the above study the three bones were rotated in the computer along *manually* defined axis; even though this bone positioning was verified by veterinary surgeons, it does introduce geometric inaccuracies in the representation of the joint capsule. Instead, distinct CT scans should be used for the different configurations of the elbow.

Conclusions

Chapter 13

Conclusions

13.1 General conclusions

From image acquisition to finite element simulations, the entire biomechanical modelling pipeline was investigated in this work. Researchers usually concentrate on one particular problem of this pipeline: image processing, geometric representation, mesh generation or numerical modelling; and consider their problem as an independent setting. Instead, we studied the pipeline as a whole and designed it to meet the final goals of understanding the human body and improving patients' health. Therefore special attention was given to robustness and automatisation. With this aim in mind, all developments were integrated in the finite element code Metafor: a segmented 3D image may now be loaded in the software and, after definition of the required meshing and simulation parameters, finite element results may be obtained without further user interaction. Also, when designing this pipeline, we tried not to restrict it to specific biomechanical applications. Therefore our algorithms can be employed to model most tissues of the living body, provided they are solid. Applications in this thesis include the modelling of heterogeneous (multi-material) structures, complex 3D geometries like cellular structures and incompressible tissues.

Notwithstanding the overall generality and wide application range of the proposed image - to - FE model pipeline, this dissertation presents a detailed analysis and novel contributions to two major issues of biomechanical modelling: patient-specific mesh generation and the removal of the locking behaviour of the linear tetrahedron under near incompressibility constraints. These two topics form the substance of Part 1 and Part 2 of this work. In Part 3 the developments of Part 1 and Part 2 are used to perform finite element analyses of actual problems of biomechanics, in a view of illustrating the possible applications of this work.

13.2 Patient-specific finite element mesh generation

In the first part of this work, finite element meshes are obtained from patient-specific data.

Chapter 2 gives an extensive review of patient-specific meshing strategies. Numerous image-to-mesh algorithms, with different specificities and application range, are suggested in the literature and it is often very difficult for a young researcher to understand the nuances between them. The chapter presents several meshing options, with their advantages and disadvantages, and gives some practical recommendations on the meshing strategy to adopt in regard to the targeted application. Indeed, in some cases, when image segmentation must be avoided or automatised or when several types of finite elements must be generated, it is better to design the meshing strategy specifically for the tissue of interest. Also, when the same tissue will be modelled several times, e.g. for successive patients, specific algorithms are available to deform a mesh in a way that it fits an other patient's data. The meshing strategy proposed in this thesis is more suited for isolated studies. it generates tetrahedral finite element meshes from three-dimensional segmented images. There is no restriction on the number or type of tissues represented in the input data. Particular emphasis was placed on the elimination of segmentation noise from the model's boundaries and on the ability to create heterogeneous models. In response to these two requirements, the proposed meshing strategy comprises two steps: first, the extraction of smooth boundaries from the discrete segmented input dataset; second, the generation of consistent multi-material finite element meshes. These two steps were presented in Chapter 3 and Chapter 4.

In Chapter 3 a smooth representation of the object is constructed from the discrete, jagged, segmented image, and this, prior to mesh generation. The main advantages for this are (1) aliasing or staircase artefacts are alleviated, (2) the result is more robust to segmentation noise, (3) the user may define the mesh resolution freely, independently of the image resolution, (4) geometric accuracy is ensured to remain unchanged during possible subsequent mesh adaptation steps. The surface reconstruction algorithm was taken from literature, but its use for the generation of multi-domain, or multi-material, tetrahedral meshes is an original contribution of this work. Initially developed for surface reconstruction from a cloud of points, the multi-level Partition of Unity (MPU) implicit surface reconstruction approach was extended to reconstruct geometries from segmented datasets, in the view of patient-specific meshing:

- A strategy to define a set of points and associated normals from segmented uni-label and multi-label images has been presented.

- Interpolation weight functions rather than the original approximating ones have been proposed. However, results computed on several datasets showed no significant improvement of the generated models in regard to their geometric accuracy, as some results do show lower Hausdorff distances and others do not.

- The use of linear instead of quadratic local functions has been investigated. Results indicate that using linear functions adds rapidity and robustness. It enables fast surface reconstruction from, possibly noisy, sets of points. The decrease in reconstruction time is mainly due to the fact that no system must be solved to compute the coefficients of the linear function as was needed for the quadratic function. The robustness has been observed for all the input datasets considered. Results have shown that a better match between input and output model may be obtained with quadratic functions, if the parameter N_{min}, defining the minimum set of points per subdivision cell of the MPU method, is finely tuned. However, low values of this parameter creates surfaces with irregularities and low geometric accuracy. Instead, linear functions have the great advantage to generate valid results for all values of this parameter, valid meaning that the geometric approximation is still within the defined limits of one voxel width and with no spurious parts so that direct application of a surface triangulation algorithm will generate topologically correct meshes, suitable for further volume mesh generation and finite element analysis.

- An efficient strategy to represent multi-material structures with a set of distance functions has been defined. This allowed us to generalise the MPU surface reconstruction procedure to the implicit representation of biological structures having several inner boundary surfaces, defining several material regions within the structure.

The surface reconstruction algorithm defines a distance function $f(\boldsymbol{x})$ for a single-material tissue and a set of distance functions $f_i(\boldsymbol{x})$ for multi-domain structures. These functions give an approximation of the distance to the tissue boundaries in the initial segmented 3D dataset.

In Chapter 4, a strategy to generate a surface mesh of the tissue boundaries is proposed. The main particularity of the approach is that it is capable of generating valid meshes even in the case of multiple interconnected tissues. The term valid meaning, valid in the sense of the finite element method, that is to say, with no gaps nor overlays at the material interfaces, and, with node-to-node and edge-to-edge connections only. The generated surface meshes are triangular meshes and may be used as input to classical tetrahedral volume mesh generators. The novelties of our meshing strategy are:

- an efficient implementation of a multi-material marching tetrahedra algorithm, based on a novel description of multi-material structures;
- a strategy to accurately position interface nodes during mesh generation, which greatly improves the quality of the meshes along material junctions;
- a multi-material decimation scheme that may be used during and/or after mesh generation;
- a multi-material mesh adaptation filter that uses the proposed surface reconstruction algorithm to keep the fidelity with respect to the initial segmented data.

The resulting image-to-mesh approach is efficient in generating patient-specific finite element meshes as attested by the examples presented at the end of Chapter 4 and in the third part of this dissertation as well as the several peer-reviewed and conference papers [49, 56–58, 60, 116, 135] published.

In Chapter 5, patient-specific hexahedral meshes are created by combining the proposed surface reconstruction algorithm (Chapter 3) with a classical voxel-conversion algorithm. This resulting, novel, meshing strategy has the following advantages:

- It outputs hexahedral meshes and therefore avoids problems arising from using the standard linear tetrahedral element in finite element simulations of incompressible and nearly incompressible materials.
- It outputs meshes with smooth surface boundaries, so that the stress concentration and contact problems arising with voxel-based meshes are solved.
- It is extremely time-efficient as voxel conversion is straightforward and because our surface reconstruction algorithm uses an octree-based subdivision scheme so that its computation time depends on the complexity of the structure rather than the image size.
- It is well adapted for the generation of structures composed of several material domains, as illustrated at the end of the chapter.
- It allows to assign heterogeneous material properties based on the image greyscale values easily.

The major drawback in smoothing a voxel-based hexahedral mesh is that it generates distorted elements along the objects' boundaries. Therefore, a strategy to control and limit element deterioration during the meshing procedure was proposed. In comparison to our

tetrahedral mesh generation procedure, this hexahedral mesher also has the disadvantage that the mesh resolution is not user-controlled: it is fixed by the image resolution.

13.3 Locking-free formulations for the linear tetrahedron

In the second part of this work, unlocking formulations for the low-order tetrahedral element were investigated. Indeed, because of the complexity of the geometries involved, tetrahedral meshes are often more practical in computational biomechanics. However, the behaviour of the standard linear tetrahedron becomes extremely poor as the incompressible limit is approached and an overstiff behaviour, called locking, is observed. Problems where incompressibility is encountered include the analysis of rubbery solids, which are typically modelled as incompressible hyperelastic materials, as well as the analysis of J2 elasto-plastic metals, for which an isochoric plastic flow is generally assumed (von Mises plasticity). Volumetric locking can be eliminated by employing higher-order finite elements. But, due to their simplicity and robustness, low-order elements are often preferred in large-scale non-linear computations.

In Chapter 7 popular non-locking formulations from literature were presented and two novel formulations were proposed. First, popular nodal-based formulations were reviewed: the average nodal pressure (ANP) linear tetrahedron proposed by Bonet and Burton [25] and the average nodal strain linear tetrahedron proposed by Dohrmann et al. [55]. In nodal-based formulations, the incompressibility constraints are enforced on newly defined nodal volumes instead of at each Gauss point. These formulations are well suited for explicit dynamics analysis where a lumped mass matrix is used. Second, the F-bar methodology for quadrilateral and hexahedral elements, and its extensions to triangular and tetrahedral elements, the F-bar-patched method, were presented. The idea of F-bar methods is to define a modified deformation gradient, denoted by $\bar{\mathbf{F}}$, over the element, which is used to compute the stresses in the traditional way. These methods are suitable for implicit finite element analysis and an expression for the stiffness terms of the tangent stiffness matrix is proposed. Even though the idea is interesting, the F-bar-patch tetrahedron proposed by de Souza Neto et al. [52] is not very useful in practice because it requires the definition of non-overlapping patches of tetrahedral elements, for which no automatic algorithm is yet available. In Sections 7.3 and 7.4 two successive ideas to remove the locking of the standard linear tetrahedron were presented.

The first proposal is a F-bar-patch tetrahedron in which, for each element, the incompressibility constraints are enforced over the element itself and its neighbours. Both the element's node and the face-neighbourhood were investigated through a set of 2D and 3D

benchmarking numerical tests in Chapter 8. Finite element results obtained with this formulation were not very satisfactory: only a part of the locking behaviour of the standard tetrahedron was removed, the stress field obtained in the case of Cook's membrane appeared to be smoothed, and the extremal values of the stresses were under-estimated. In the case of the cylinder under internal pressure, the observed stress field was highly sporadic. And, in the third benchmark, the Taylor bar impact, the equivalent plastic strain was, also, under-estimated.

In the second proposal, a nodal Jacobian is defined at the element's node as the ratio between current and initial nodal volumes; the definition of nodal volumes being identical to nodal-based formulations. A modified element Jacobian is then defined by averaging the nodal Jacobians. This average elemental Jacobian (AEJ) is used to define the modified deformation gradient of the F-bar methodologies. This formulation was proposed for the two-dimensional case only by Andrade Pires et al. [5]. However, we obtained a different expression for the stiffness terms of the internal tangent stiffness matrix, by successive linearisation and finite element discretisation of the internal virtual work equation. The formulation was then extended to 3D explicit and implicit cases; the resulting locking-free linear tetrahedron is an original contribution of this thesis. To summarise, the proposed Average Elemental Jacobian (AEJ) formulation:

- is suitable for both explicit and implicit analysis, as illustrated by the numerical examples of Chapter 8;
- preserves the displacement-based structure of the finite element equations, as opposed to mixed finite element methods for the unlocking of the linear tetrahedron; it is therefore easier to implement in existing FE software;
- can be used with any, strain-driven, constitutive model; the only difference with the traditional finite element resolution is that the stresses are computed with a modified deformation gradient;
- for implicit analyses, allows the use of the full Newton-Raphson algorithm to solve the global equilibrium equations and quadratic convergence rates are ensured; indeed, the exact expression of the stiffness terms in the tangent stiffness matrix are presented;
- can be used for heterogeneous solids constituted of several materials. A special treatment is applied to the elements lying on the border of the mesh or at material interfaces.

The performance of this element has been assessed using three classical benchmarking tests from literature: the 2D and 3D Cook's membrane, the thick-walled cylinder under

internal pressure and the Taylor bar impact. Implicit quasi-static and explicit dynamics simulations were performed, using various constitutive models. Results indicate that the proposed element considerably removes both the volumetric and the shear locking of the standard tetrahedron with linear shape functions. Also, a correct distribution of pressure, Von Mises stress and equivalent plastic strain was observed in all cases.

Some dissimilarities with literature results, observed for all Metafor tetrahedral elements in the case of the compressible linear elastic Cook's membrane, will be investigated in the future. Also, in the case of the elasto-plastic Taylor bar impact, a convergence study mesh refinement indicated that the proposed *AEJ* element converges slower than locking-free hexahedral elements. More studies will be performed in the future in order to confirm this observation. Finally, the minimum number of elements required on the thickness of the Cook's membrane, or equivalently, on the thickness of the thick-walled cylinder, to obtain a locking-free solution will also be determined in the future.

In future studies, the implementation of the proposed Average Elemental Jacobian linear tetrahedron in Metafor will be improved. Indeed, in the current implementation the nodal volume change ratios are re-computed, which involves a loop over the neighbouring elements, for each new element. Instead these nodal quantities should be stored in memory. This should substantially reduce the computation times.

13.4 Applications of this work and Perspectives

Part 3 of this dissertation allowed us to integrate the developments of Part 1 and Part 2 and illustrate possible applications of this work. The proposed meshing techniques and tetrahedral element formulation were used to perform patient-specific finite element analyses for three different studies: the compression of cellular structures, the simulation of intra-operative brain deformation and the analysis of canine humeral fractures.

The first illustration allowed the comparison of the different image-based meshing strategies proposed in this thesis, through the micro-FE analysis of a deer antler cancellous bone. A voxel-based mesh was generated using the simple voxel-conversion procedure. Two hexahedral meshes, one with acceptable element quality but irregular boundaries and the other with distorted elements but smooth boundaries, were generated using the hexahedral mesher presented in Chapter 5. A tetrahedral mesh with approximately the same number of nodes and volume was generated using the tetrahedral mesher proposed in Chapter 4. The results of the finite element simulations, presented in Chapter 10, show a fairly similar behaviour of the different models. But, we observed a decrease in the overall stiffness of voxel-based meshes after smoothing. Also, depending on the level of compression, the

tetrahedral model is, globally, as stiff or stiffer than the hexahedral models. In future studies a comparison with experimental data will be done in order to determine which model better reproduces the mechanical behaviour of the sample.

In a second study, both our multi-material tetrahedral mesher and our locking-free tetrahedral element were employed to compute the intra-operative brain deformations resulting from skull opening. the proposed biomechanical model is more accurate in predicting the displacement field of the brain volume than previous models. Within the framework of image-guided neurosurgery, this model could help to deform the patient's preoperative images during surgery in order to follow the brain deformation. In future studies, the modelling of surgical acts such as retraction and resections will be studied; as well as the inclusion of other structures like the ventricles; and the definition of adequate boundary conditions.

In the last study, multi-material finite element models of the dog elbow were created to investigate under which conditions condylar fractures of the humerus occur. Despite the complexity of the model (several material regions with distinct material properties, orthotropic and elasto-plastic material laws, bone contact and bone failure) preliminary results could be obtained.

The above biomechanical applications indicate that the approach developed in this thesis for patient-specific finite element modelling is promising.

Appendix

Appendix A

Tensor Analysis

A.1 Tensor Product

The tensor product, also called dyadic product, of two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$, denoted

$$\boldsymbol{u} \otimes \boldsymbol{v} \tag{A.1}$$

is the tensor that maps each vector $\boldsymbol{w}$ into the vector $\boldsymbol{u}\,(\boldsymbol{v} \cdot \boldsymbol{w})$,

$$(\boldsymbol{u} \otimes \boldsymbol{v}) \cdot \boldsymbol{w} = \boldsymbol{u}\,(\boldsymbol{v} \cdot \boldsymbol{w}) \tag{A.2}$$

The tensor product of two matrices $\mathbf{A}$ and $\mathbf{B}$, denoted

$$\mathbf{A} \otimes \mathbf{B} \tag{A.3}$$

is the tensor that maps each second order tensor $\mathbf{W}$ into the second order tensor $\mathbf{A}(\mathbf{B} : \mathbf{W})$,

$$(\mathbf{A} \otimes \mathbf{B}) : \mathbf{W} = \mathbf{A}(\mathbf{B} : \mathbf{W}) \tag{A.4}$$

Appendix B

Directional derivatives

In Finite Element Analysis, the nonlinear equilibrium equations are generally solved via a Newton-Raphson iteration procedure. The latter requires a linearisation of the equilibrium equations which requires an understanding of the directional derivative. Directional derivatives are thoroughly explained in Bonet and Wood [26]: *A directional derivative is a generalization of a derivative in that it provides the change in an item due to a small change in something upon which the item depends*.

B.1 Directional derivative of a function

Consider a function G at a state defined by the positions $\boldsymbol{x}$, $G(\boldsymbol{x})$, and its value due to an increment in position $\boldsymbol{x}+\boldsymbol{\eta}$, $G(\boldsymbol{x}+\boldsymbol{\eta})$. The directional derivative of G at $\boldsymbol{x}$ in the direction $\boldsymbol{\eta}$, noted $DG(\boldsymbol{x})[\boldsymbol{\eta}]$, represents the gradient of G in the direction $\boldsymbol{\eta}$: it gives a linear approximation of the increment of G due to the increment in position $\boldsymbol{\eta}$,

$$DG(\boldsymbol{x})\left[\boldsymbol{\eta}\right] = G(\boldsymbol{x}+\boldsymbol{\eta}) - G(\boldsymbol{x}) \tag{B.1}$$

In order to evaluate this derivative, we introduce a parameter ϵ, used to scale the displacements $\boldsymbol{\eta}$. This enables us to evaluate function G at $\boldsymbol{x}+\boldsymbol{\eta}$ using a first-order Taylor series expansion around $\epsilon = 0$,

$$G\left(\boldsymbol{x}+\epsilon\boldsymbol{\eta}\right) \approx G\left(\boldsymbol{x}\right) + \epsilon \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} G\left(\boldsymbol{x}+\epsilon\boldsymbol{\eta}\right) \tag{B.2}$$

Taking $\epsilon = 1$ in (B.2) and comparing it to (B.1) gives a useful equation to evaluate the directional derivative:

$$DG(\boldsymbol{x})[\boldsymbol{\eta}] = \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} G\left(\boldsymbol{x}+\epsilon\boldsymbol{\eta}\right) \tag{B.3}$$

Using the concept of the directional derivative, a function $G\left(\boldsymbol{x}+\boldsymbol{\eta}\right)$ can now be linearised as

$$G\left(\boldsymbol{x}+\boldsymbol{\eta}\right) \approx G^{\mathrm{LIN}}\left(\boldsymbol{x}+\boldsymbol{\eta}\right) = G\left(\boldsymbol{x}\right) + DG(\boldsymbol{x})[\boldsymbol{\eta}] \tag{B.4}$$

B.2 Directional derivative and partial derivatives

The directional derivative of a function G with respect to an increment tensor $\Delta\mathbf{U}$ is given by

$$DG\left[\Delta\mathbf{B}\right] = \sum_{I,J=1}^{3} \frac{\partial G}{\partial B_{IJ}} \Delta B_{IJ} = \frac{\partial G}{\partial \mathbf{B}} : \Delta\mathbf{B} \tag{B.5}$$

B.3 Directional derivative of the determinant of a matrix

The concept of the directional derivative is more general than the above. For example, the linearisation of the determinant of a matrix $\mathbf{A}$ in the direction of the change in a matrix $\mathbf{B}$ can be computed using the concept of the directional derivative. From (B.4),

$$\det\left(\mathbf{A}+\mathbf{B}\right) \approx \det\left(\mathbf{A}\right) + D\det\left(\mathbf{A}\right)\left[\mathbf{B}\right] \tag{B.6}$$

Computing the directional derivative by application of Equation (B.3) gives

$$\begin{aligned} D\det\left(\mathbf{A}\right)\left[\mathbf{B}\right] &= \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \det\left(\mathbf{A}+\epsilon\mathbf{B}\right) \\ &= \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \det\left(\mathbf{A}\left(\mathbf{I}+\epsilon\mathbf{A}^{-1}\mathbf{B}\right)\right) \\ &= \det\left(\mathbf{A}\right) \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \det\left(\mathbf{I}+\epsilon\mathbf{A}^{-1}\mathbf{B}\right) \end{aligned} \tag{B.7}$$

Now, let us remember that the characteristic equation of a matrix $\mathbf{B}$ with eigenvalues $\lambda_1^{\mathbf{B}}, \lambda_2^{\mathbf{B}}, \lambda_3^{\mathbf{B}}$ is given by

$$\det\left(\mathbf{B}-\mu\mathbf{I}\right) = \left(\lambda_1^{\mathbf{B}}-\mu\right)\left(\lambda_2^{\mathbf{B}}-\mu\right)\left(\lambda_3^{\mathbf{B}}-\mu\right) \tag{B.8}$$

Replacing $\mathbf{B}$ by $\epsilon\mathbf{A}^{-1}\mathbf{B}$ and taking $\mu=-1$ gives

$$\det\left(\epsilon\mathbf{A}^{-1}\mathbf{B}+\mathbf{I}\right) = \left(\epsilon\lambda_1^{\mathbf{A}^{-1}\mathbf{B}}+1\right)\left(\epsilon\lambda_2^{\mathbf{A}^{-1}\mathbf{B}}+1\right)\left(\epsilon\lambda_3^{\mathbf{A}^{-1}\mathbf{B}}+1\right) \tag{B.9}$$

so that (B.7) becomes,

$$\begin{aligned} D\det(\mathbf{A})[\mathbf{B}] &= \det(\mathbf{A}) \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \left(\epsilon\lambda_1^{\mathbf{A}^{-1}\mathbf{B}}+1\right)\left(\epsilon\lambda_2^{\mathbf{A}^{-1}\mathbf{B}}+1\right)\left(\epsilon\lambda_3^{\mathbf{A}^{-1}\mathbf{B}}+1\right) \\ &= \det(\mathbf{A})\left(\lambda_1^{\mathbf{A}^{-1}\mathbf{B}}+\lambda_2^{\mathbf{A}^{-1}\mathbf{B}}+\lambda_3^{\mathbf{A}^{-1}\mathbf{B}}\right) \\ &= \det(\mathbf{A})\,\mathrm{tr}\left(\mathbf{A}^{-1}\mathbf{B}\right) \\ &= \det(\mathbf{A})\left(\mathbf{A}^{-T}:\mathbf{B}\right) \end{aligned} \tag{B.10}$$

B.4 Linearisation of the deformation gradient

The deformation gradient $\mathbf{F}$ can be linearised around the current point $\boldsymbol{x}=\boldsymbol{\phi}(\boldsymbol{X},t)$ in the direction of a small displacement in the current configuration $\boldsymbol{u}(\boldsymbol{x})$ by taking its directional derivative:

$$\begin{aligned} D\mathbf{F}[\boldsymbol{u}] &= \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \mathbf{F}\left(\boldsymbol{\phi}(\boldsymbol{X},t)+\epsilon\boldsymbol{u}\right) \\ &= \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \frac{\partial\left(\boldsymbol{\phi}(\boldsymbol{X},t)+\epsilon\boldsymbol{u}\right)}{\partial\boldsymbol{X}} \\ &= \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \left(\frac{\partial\boldsymbol{\phi}(\boldsymbol{X},t)}{\partial\boldsymbol{X}}+\epsilon\frac{\partial\boldsymbol{u}}{\partial\boldsymbol{X}}\right) \\ &= \frac{\partial\boldsymbol{u}}{\partial\boldsymbol{X}} \\ &= \nabla_0\boldsymbol{u} \\ &= (\nabla\boldsymbol{u})\cdot\frac{\partial\boldsymbol{x}}{\partial\boldsymbol{X}} \\ &= (\nabla\boldsymbol{u})\cdot\mathbf{F} \end{aligned} \tag{B.11}$$

where ∇_0 represents the gradient with respect to the initial configuration and ∇, the gradient with respect to the current configuration .

B.5 Linearisation of the Jacobian

The directional derivative of the Jacobian J with respect to an increment $\boldsymbol{u}(\boldsymbol{x})$ in the spatial configuration is

$$DJ[\boldsymbol{u}] = DJ(\mathbf{F})[\boldsymbol{u}] \tag{B.12}$$

The chain rule may be applied as the directional derivative satisfy the usual properties of the derivative:

$$DJ\left[\boldsymbol{u}\right] = DJ(\mathbf{F})D\mathbf{F}\left[\boldsymbol{u}\right] \tag{B.13}$$

Taking account of previous results on the linearisation of the determinant of a matrix (B.10) and the linearisation of the deformation gradient (B.11), we have

$$\begin{aligned} DJ\left[\boldsymbol{u}\right] &= J\left(\mathbf{F}^{-T} : \nabla_0 \boldsymbol{u}\right) \\ &= J\ \mathrm{tr}\left(\mathbf{F}^{-1}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}}\right) \\ &= J\ \mathrm{tr}\left(\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\right) \\ &= J\ \mathrm{tr}(\nabla \boldsymbol{u}) \\ &= J\ \mathrm{div}(\boldsymbol{u}) \end{aligned} \tag{B.14}$$

Appendix C

F-bar methodology

C.1 Virtual internal work linearisation for the F-bar Hexahedron

The directional derivative of the virtual internal work for the F-bar Hexahedron is (7.40)

$$D\delta\mathcal{W}^{\text{int}}\left[\boldsymbol{\eta}\right] = \int_{\Omega_0} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] : \nabla_0\delta\boldsymbol{u}\, d\Omega_0 \tag{C.1}$$

and, for one element

$$D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = \int_{\Omega_{0,e}} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] : \nabla_0\delta\boldsymbol{u}\, d\Omega_{0,e} \tag{C.2}$$

Taking account of (7.41), the directional derivative of $\mathbf{P}$ appearing in (C.1) is given by

$$\begin{aligned} D\mathbf{P}\left(\bar{\mathbf{F}}\right)\left[\boldsymbol{\eta}\right] &= D\left\{\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}}\bar{\mathbf{P}}(\bar{\mathbf{F}})\right\}\left[\boldsymbol{\eta}\right] \\ &= D\left\{\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}}\right\}\left[\boldsymbol{\eta}\right]\ \bar{\mathbf{P}}(\bar{\mathbf{F}}) + \left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}} D\left\{\bar{\mathbf{P}}(\bar{\mathbf{F}})\right\}\left[\boldsymbol{\eta}\right] \end{aligned} \tag{C.3}$$

with

$$D\left\{\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}}\right\}\left[\boldsymbol{\eta}\right] = -\frac{2}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{5}{3}}\frac{1}{J^2}\left[J\, D\bar{J}\left[\boldsymbol{\eta}\right] - \bar{J}\, DJ\left[\boldsymbol{\eta}\right]\right] \tag{C.4}$$

and

$$D\left\{\bar{\mathbf{P}}(\bar{\mathbf{F}})\right\}[\boldsymbol{\eta}] = \frac{d\bar{\mathbf{P}}}{d\bar{\mathbf{F}}}\, D\bar{\mathbf{F}}[\boldsymbol{\eta}] \tag{C.5}$$

Developing the directional derivative of $\bar{\mathbf{F}} = (\bar{J}/J)^{1/3}\,\mathbf{F}$ and then replacing (C.4) and (C.5) in (C.3) gives

$$\begin{aligned} D\mathbf{P}\left(\bar{\mathbf{F}}\right)[\boldsymbol{\eta}] = &-\frac{2}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{5}{3}} \frac{1}{J^2}\left[J\, D\bar{J}[\boldsymbol{\eta}] - \bar{J}\, DJ[\boldsymbol{\eta}]\right]\bar{\mathbf{P}} \\ &+\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}} \frac{d\bar{\mathbf{P}}}{d\bar{\mathbf{F}}} : \left[\frac{1}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}} \frac{1}{J^2}\left[J\, D\bar{J}[\boldsymbol{\eta}] - \bar{J}\, DJ[\boldsymbol{\eta}]\right]\mathbf{F} + \left(\frac{\bar{J}}{J}\right)^{\frac{1}{3}} D\mathbf{F}[\boldsymbol{\eta}]\right] \end{aligned} \tag{C.6}$$

Substituting the two-point tangent modulus (7.43), the directional derivative of $\mathbf{F}$ (B.11), the directional derivative of J (B.14) and the directional derivative of $\bar{J}$ (7.44) into (C.3) gives

$$\begin{aligned} D\mathbf{P}\left(\bar{\mathbf{F}}\right)[\boldsymbol{\eta}] = &\left(\frac{\bar{J}}{J}\right)^{-\frac{1}{3}} \boldsymbol{A}(\bar{\mathbf{F}}) : \nabla_0\boldsymbol{\eta} \\ &-\frac{2}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}} \left[\mathbf{F}^{-T} : \left(\nabla_{0,\text{centroid}}\boldsymbol{\eta} - \nabla_0\boldsymbol{\eta}\right)\right]\bar{\mathbf{P}}(\bar{\mathbf{F}}) \\ &+\frac{1}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{1}{3}} \boldsymbol{A}(\bar{\mathbf{F}}) : \left[\left[\mathbf{F}^{-T} : \left(\nabla_{0,\text{centroid}}\boldsymbol{\eta} - \nabla_0\boldsymbol{\eta}\right)\right]\mathbf{F}\right] \end{aligned} \tag{C.7}$$

Property (A.4) for the tensor product of matrices gives

$$\begin{aligned} \left(\mathbf{F}^{-T} : \nabla_0\delta\boldsymbol{u}\right)\boldsymbol{M} &= \left(\mathbf{I} : \nabla_0\delta\boldsymbol{u}\,\mathbf{F}^{-1}\right)\boldsymbol{M} \\ &= (\mathbf{I} : \nabla\delta\boldsymbol{u})\,\boldsymbol{M} \\ &= (\boldsymbol{M}\otimes\mathbf{I}) : \nabla\delta\boldsymbol{u} \end{aligned} \tag{C.8}$$

so that (C.7) becomes

$$\begin{aligned} D\mathbf{P}\left(\bar{\mathbf{F}}\right)[\boldsymbol{\eta}] = &\left(\frac{\bar{J}}{J}\right)^{-\frac{1}{3}} \boldsymbol{A}(\bar{\mathbf{F}}) : \nabla_0\boldsymbol{\eta} \\ &-\frac{2}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}} \left(\bar{\mathbf{P}}(\bar{\mathbf{F}})\otimes\mathbf{I}\right) : \left(\nabla_{\text{centroid}}\boldsymbol{\eta} - \nabla\boldsymbol{\eta}\right) \\ &+\frac{1}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{1}{3}} \boldsymbol{A}(\bar{\mathbf{F}}) : \left((\mathbf{F}\otimes\mathbf{I}) : \left(\nabla_{\text{centroid}}\boldsymbol{\eta} - \nabla\boldsymbol{\eta}\right)\right) \end{aligned} \tag{C.9}$$

Substituting this result into the equation of the directional derivative of the internal virtual work (C.2) gives

$$\begin{aligned} D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = &\left(\frac{\bar{J}}{J}\right)^{-\frac{1}{3}} \int_{\Omega_e} \left(\boldsymbol{A}(\bar{\mathbf{F}}) : \nabla_0 \boldsymbol{\eta}\right) : \nabla_0 \delta\boldsymbol{u}\, d\Omega_e \\ &- \frac{2}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{2}{3}} \int_{\Omega_e} \left[\left(\bar{\mathbf{P}}(\bar{\mathbf{F}}) \otimes \mathbf{I}\right) : \left(\nabla_{\text{centroid}}\boldsymbol{\eta} - \nabla\boldsymbol{\eta}\right)\right] : \nabla_0 \delta\boldsymbol{u}\, d\Omega_e \\ &+ \frac{1}{3}\left(\frac{\bar{J}}{J}\right)^{-\frac{1}{3}} \int_{\Omega_e} \left[\boldsymbol{A}(\bar{\mathbf{F}}) : \left((\mathbf{F} \otimes \mathbf{I}) : \left(\nabla_{\text{centroid}}\boldsymbol{\eta} - \nabla\boldsymbol{\eta}\right)\right)\right] : \nabla_0 \delta\boldsymbol{u}\, d\Omega_e \end{aligned} \tag{C.10}$$

Transforming this to the spatial domain using relations (6.75) gives

$$\begin{aligned} D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = &\int_{\Omega_e} \left[\boldsymbol{a}(\bar{\mathbf{F}}) : \nabla\boldsymbol{\eta}\right] : \nabla\delta\boldsymbol{u}\, d\Omega_e \\ &+ \int_{\Omega_e} \left[\mathbf{q}(\bar{\mathbf{F}}) : \left(\nabla_{\text{centroid}}\boldsymbol{\eta} - \nabla\boldsymbol{\eta}\right)\right] : \nabla\delta\boldsymbol{u}\, d\Omega_e \end{aligned} \tag{C.11}$$

with

$$a_{ijkl}\left(\bar{\mathbf{F}}\right) = \frac{1}{\bar{J}}\, \bar{F}_{kB}\, \bar{A}_{iAjB}\left(\bar{\mathbf{F}}\right)\, \bar{F}_{lA} \tag{C.12}$$

and having defined

$$\mathbf{q}(\bar{\mathbf{F}}) = \frac{1}{3}\boldsymbol{a}(\bar{\mathbf{F}}) : (\mathbf{I} \otimes \mathbf{I}) - \frac{2}{3}\left[\bar{\boldsymbol{\sigma}}(\bar{\mathbf{F}}) \otimes \mathbf{I}\right] \tag{C.13}$$

Discretising (C.11) using the finite element approximating functions (6.36) gives

$$\begin{aligned} D\delta\mathcal{W}^{\text{int,e}}\left[\boldsymbol{\eta}\right] = &\,\delta\boldsymbol{u} \cdot \int_{\Omega_e} (\nabla\mathbf{N})^T\, \boldsymbol{a}(\bar{\mathbf{F}})\, \nabla\mathbf{N}\, d\Omega_e \cdot \boldsymbol{\eta} \\ &+ \delta\boldsymbol{u} \cdot \int_{\Omega_e} (\nabla\mathbf{N})^T\, \mathbf{q}(\bar{\mathbf{F}})\left(\nabla_{\text{centroid}}\mathbf{N} - \nabla\mathbf{N}\right)\, d\Omega_e \cdot \boldsymbol{\eta} \end{aligned} \tag{C.14}$$

Bibliography

[1] **Adam, C. J., Pearcy, M. J., and Askin, G. N.** Patient-specific finite element analysis of single rod adolescent idiopathic scoliosis surgery. In "First Asian-Pacific Conference on Biomechanics," pages 201–202, Osaka, Japan, 2004.

[2] **Akhtar, R., Daymond, M., Almer, J., and Mummery, P.** Elastic strains in antler trabecular bone determined by synchrotron X-ray diffraction. *Acta Biomaterialia*, 4(6):1677 – 1687, 2008.

[3] **Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., and Silva, C.** Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9 (1):3–15, 2003.

[4] **Amenta, N., Choi, S., and Kolluri, R. K.** The power crust. In "Proceedings of the sixth ACM symposium on Solid modeling and applications," pages 249–266, New York, USA, 2001.

[5] **Andrade Pires, F., de Souza Neto, E., and de la Cuesta Padilla, J.** An assessment of the average nodal volume formulation for the analysis of nearly incompressible solids under finite strains. *Communications in Numerical Methods in Engineering*, 20(7):569–583, 2004.

[6] **Andújar, C., Brunet, P., Chica, A., Navazo, I., Rossignac, J., and Vinacua, A.** Optimizing the topological and combinatorial complexity of isosurfaces. *Computer-Aided Design*, 37(8): 847–857, 2005.

[7] **Arnold, D., Brezzi, F., and Fortin, M.** A stable finite element for the stokes equations. *Calcolo*, 21:337–344, 1984.

[8] **Audette, M., Delingette, H., Fuchs, A., Burgert, O., and Chinzei, K.** A topologically faithful, tissue-guided, spatially varying meshing strategy for computing patient-specific head models for endoscopic pituitary surgery simulation. *Computer Aided Surgery*, 12(1):43–52, 2007.

[9] **Babuška, I., Caloz, G., and Osborn, J.** Special finite element methods for a class of second order elliptic problems with rough coefficients. *SIAM Journal on Numerical Analysis*, pages 945–981, 1994.

[10] **Baker, B., Grosse, E., and Rafferty, C.** Nonobtuse triangulation of polygons. *Discrete and Computational Geometry*, 3(1):147–168, 1988.

[11] **Bandak, F.** Shaken baby syndrome: a biomechanics analysis of injury mechanisms. *Forensic science international*, 151(1):71–79, 2005.

[12] **Bardyn, T., Reyes, M., Larrea, X., and Büchler, P.** Influence of Smoothing on Voxel-Based Mesh Accuracy in Micro-Finite Element. *Computational Biomechanics for Medicine*, pages 85–93, 2010.

[13] **Béchet, E., Cuilliere, J.-C., and Trochu, F.** Generation of a finite element MESH from stereolithography (STL) files. *Computer-Aided Design*, 34(1):1 – 17, 2002.

[14] **Bellomo, N., De Angelis, E., and Preziosi, L.** Review Article: Multiscale Modeling and Mathematical Problems Related to Tumor Evolution and Medical Therapys. *Journal of Theoretical Medicine*, 5(2):111–136, 2003.

[15] **Belytschko, T. and Bindeman, L.** Assumed strain stabilization of the 4-node quadrilateral with 1-point quadrature for nonlinear problems. *Computer Methods in Applied Mechanics and Engineering*, 88(3):311–340, 1991.

[16] **Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., and Taubin, G.** The Ball-Pivoting Algorithm for Surface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5:349–359, 1999.

[17] **Bertram, M., Reis, G., Lengen, R. H. V., Köhn, S., and Hagen, H.** Non-manifold mesh extraction from time-varying segmented volumes used for modeling a human heart. In "Proceedings of the IEEE/Eurographics Symposium on Visualization," pages 1–10, 2005.

[18] **Bischoff, S. and Kobbelt, L.** Extracting Consistent and Manifold Interfaces from Multi-valued Volume Data Sets. In "Bildverarbeitung für die Medizin 2006," pages 281–285. Springer, 2006.

[19] **Böhme, B., d'Otreppe, V., Ponthot, J., and Balligand, M.** Considerations on the Pathophysiology of Canine Condylar Fractures by Finite Element Analysis. In "Proceedings of the 20th Annual Scientific Meeting of the European College of Veterinary Surgeons," Belgium, Ghent, July 2011.

[20] **Boissonnat, J. and Yvinec, M.** *Algorithmic geometry*. Cambridge University Press, 1998.

[21] **Boissonnat, J.-D.** Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics*, 3(4):266–286, October 1984.

[22] **Boissonnat, J.-D. and Cazals, F.** Smooth surface reconstruction via natural neighbour interpolation of distance functions. In "Proceedings of the sixteenth annual ACM symposium on Computational geometry," pages 223–232, New York, USA, 2000.

[23] **Boltcheva, D., Yvinec, M., and Boissonnat, J.** Feature preserving Delaunay mesh generation from 3D multi-material images. 28(5):1455–1464, 2009.

[24] **Boman, R.** *Développement d'un formalisme Arbitraire Lagrangien Eulérien tridimensionnel en dynamique implicite. Application aux opérations de mise à forme.* PhD thesis, University of Liège, 2010.

[25] **Bonet, J. and Burton, A.** A simple average nodal pressure tetrahedral element for incompressible and nearly incompressible dynamic explicit applications. *Communications in Numerical Methods in Engineering*, 14(5):437–449, 1998.

[26] **Bonet, J. and Wood, R.** *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press, 1997.

[27] **Bonet, J., Marriott, H., and Hassan, O.** An averaged nodal deformation gradient linear tetrahedral element for large strain explicit dynamic applications. *Communications in Numerical Methods in Engineering*, 17(8):551–561, 2001.

[28] **Boyd, S. and Müller, R.** Smooth surface meshing for automated finite element model generation from 3D image data. *Journal of Biomechanics*, 39(7):1287–1295, 2006.

[29] **Braude, I., Marker, J., Museth, K., Nissanov, J., and Breen, D.** Contour-based surface reconstruction using MPU implicit models. *Graphical Models*, 69(2):139 – 157, 2007.

[30] **Bro-Nielsen, M.** Finite element modeling in surgery simulation. In "Proceedings of the IEEE," volume 86, pages 490–503, 1998.

[31] **Broccardo, M., Micheloni, M., and Krysl, P.** Assumed-deformation gradient finite elements with nodal integration for nearly incompressible large deformation analysis. *International Journal for Numerical Methods in Engineering*, 78(9):1113–1134, 2009.

[32] **Bui, Q., Papeleux, L., and Ponthot, J.** Numerical simulation of springback using enhanced assumed strain elements. *Journal of Materials Processing Technology*, 153-154:314 – 318, 2004.

[33] **Camacho, D., Hopper, R., Lin, G., and Myers, B.** An improved method for finite element mesh generation of geometrically complex structures with application to the skullbase. *Journal of Biomechanics*, 30(10):1067–1070, 1997.

[34] **Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., and Evans, T. R.** Reconstruction and representation of 3D objects with radial basis functions. In "Proceedings of the 28th annual ACM conference on Computer graphics and interactive techniques," pages 67–76, New York, USA, 2001.

[35] **Caylak, I., Laschet, G., and Mahnken, R.** Comparison of mixed and nodal based formulations for linear tetrahedral elements in elasticity and plasticity. Technical report, RWTH Aachen University, 2008.

[36] **Cebral, J., Castro, M., Appanaboyina, S., Putman, C., Millan, D., and Frangi, A.** Efficient pipeline for image-based patient-specific analysis of cerebral aneurysm hemodynamics: technique and sensitivity. *IEEE Transactions on Medical Imaging*, 24(4):457–467, 2005.

[37] **Cebral, J., Castro, M., Burgess, J., Pergolizzi, R., Sheridan, M., and Putman, C.** Characterization of cerebral aneurysms for assessing risk of rupture by using patient-specific computational hemodynamics models. *American Journal of Neuroradiology*, 26(10):2550–2559, 2005.

[38] **Chernyaev, E.** Marching cubes 33: Construction of topologically correct isosurfaces. *Institute for High Energy Physics, Moscow, Russia, Report CN/95-17*, 1995.

[39] **Choi, W., Kwak, D., Son, I., and Im, Y.** Tetrahedral mesh generation based on advancing front technique and optimization scheme. *International Journal for Numerical Methods in Engineering*, 58(12):1857–1872, 2003.

[40] **Chorin, A.** A numerical method for solving incompressible viscous flow problems. *Journal of Computational Physics*, 2(1):12–26, 1967.

[41] **Chung, J. and Hulbert, G.** A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-alpha method. *Journal of Applied Mechanics*,

60:371, 1993.

[42] **Cody, D., Gross, G., J Hou, F., Spencer, H., Goldstein, S., and P Fyhrie, D.** Femoral strength is better predicted by finite element models than QCT and DXA. *Journal of Biomechanics*, 32(10):1013–1020, 1999.

[43] **Collet, A., Desaive, T., Pierard, L., Boman, R., and Dauby, P.** Influence of thermo-electric coupling on pacemaker activity generated by mechano-electric feedback in a one-dimensional ring-shaped model of cardiac fiber. In "Proceedings of ISB 2011, International Society of Biomechanics," 2011.

[44] **Cotin, S., Delingette, H., and Ayache, N.** Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):62–73, 1999.

[45] **Coupez, T., Marie, S., and Ducloux, R.** Parallel 3D simulation of forming processes including parallel remeshing and reloading. In "ECCOMAS conference on numerical methods in engineering," pages 738–743, France, Paris, September 1996.

[46] **Cowin, S.** *Bone mechanics handbook*, volume 20. CRC press, Boca Rato, 2001.

[47] **Crawford, R., Cann, C., and Keaveny, T.** Finite element models predict in vitro vertebral body compressive strength better than quantitative computed tomography. *Bone*, 33(4): 744–750, 2003.

[48] **Curless, B. and Levoy, M.** A volumetric method for building complex models from range images. In "Proceedings of the 23rd annual ACM conference on Computer graphics and interactive techniques," pages 303–312, 1996.

[49] **de Bien, C., Mengoni, M., d'Otreppe, V., Freichels, H., Jérôme, C., Ponthot, J., Léonard, A., and Toye, D.** Development of a biomechanical model of deer antler cancellous bone based on X-ray microtomographic images. In "Micro-CT User Meeting 2012-Abstract Book," 2012.

[50] **de Bien, C.** Analyse de la relation entre les propriétés mécaniques et la microstructure de matériaux cellulaires : contribution au développement d'une méthodologie basée sur le suivi microtomographique de tests de compression. Master's thesis, University of Liège, 2010.

[51] **de Souza Neto, E., Peric, D., Dutko, M., and Owen, D.** Design of simple low order finite elements for large strain analysis of nearly incompressible solids. *International Journal of Solids and Structures*, 33(20 - 22):3277 – 3296, 1996.

[52] **de Souza Neto, E., Pires, F., and Owen, D.** F-bar-based linear triangles and tetrahedra for finite strain analysis of nearly incompressible solids. Part I: formulation and benchmarking. *International Journal for Numerical methods in Engineering*, 62(3):353–383, 2005.

[53] **Dey, T. and Goswami, S.** Provable surface reconstruction from noisy samples. In "Proceedings of the twentieth annual ACM symposium on Computational Geometry," pages 330–339, 2004.

[54] **Doblaré, M., García, J., and Gómez, M.** Modelling bone tissue fracture and healing: a review. *Engineering Fracture Mechanics*, 71(13-14):1809–1840, 2004.

[55] **Dohrmann, C., Heinstein, M. W., Jung, J., Key, S. W., and Witkowski, W. R.** Node-based uniform strain elements for three-node triangular and four-node tetrahedral meshes.

International Journal for Numerical Methods in Engineering, 47(9):1549–1568, 2000.

[56] **d'Otreppe, V., Boman, R., and Ponthot, J.** Smooth multiple-region mesh generation for biomedical applications. In "Proceedings of ECCM 2010, European Conference on Computational Mechanics," France, Paris, May 2010.

[57] **d'Otreppe, V., Böhme, B., Balligand, M., and Ponthot, J.** Finite element simulation of canine humeral condylar fractures. In "Proceedings of ISB 2011, International Society of Biomechanics," Brussels, Belgium, July 2011.

[58] **d'Otreppe, V., Mengoni, M., Boman, R., and Ponthot, J.** Image-based Finite Element Mesh Generation for Microstructures. In "ACOMEN 2011, International Conference on Advanced COmputational Methods in ENgineering," Belgium, Liège, November 2011.

[59] **d'Otreppe, V.** Application of mesh morphing techniques to the spine. Technical report, Queensland University of Technology - University of Liège, 2012.

[60] **d'Otreppe, V., Boman, R., and Ponthot, J.-P.** Generating smooth surface meshes from multi-region medical images. *International Journal for Numerical Methods in Biomedical Engineering*, 28(6-7):642–660, 2012.

[61] **Fenech, C. and Keaveny, T.** A cellular solid criterion for predicting the axial-shear failure properties of bovine trabecular bone. *Journal of Biomechanical Engineering*, 121:414, 1999.

[62] **Ferrant, M., Nabavi, A., Macq, B., Black, P., Jolesz, F., Kikinis, R., and Warfield, S.** Serial registration of intraoperative MR images of the brain. *Medical Image Analysis*, 6(4):337–359, 2002.

[63] **Flanagan, D. and Belytschko, T.** A uniform strain hexahedron and quadrilateral with orthogonal hourglass control. *International Journal for Numerical Methods in Engineering*, 17 (5):679–706, 1981.

[64] **Flory, P. J.** Thermodynamic relations for high elastic materials. *Transactions of the Faraday Society*, 57:829–838, 1961.

[65] **Formaggia, L., Nobile, F., Quarteroni, A., and Veneziani, A.** Multiscale modelling of the circulatory system: a preliminary analysis. *Computing and Visualization in Science*, 2(2): 75–83, 1999.

[66] **Franke, R. and Nielson, G.** Smooth interpolation of large sets of scattered data. *International Journal for Numerical Methods in Engineering*, 15(11):1691–1704, 1980.

[67] **Frey, P.** About surface remeshing. Technical report, INRIA, Gamma Project, Domaine de Voluceau, Rocquencourt, 2000.

[68] **Frey, P.** Generation and adaptation of computational surface meshes from discrete anatomical data. *International Journal for Numerical Methods in Engineering*, 60(6):1049–1074, 2004.

[69] **Fyhrie, D., Hamid, M., Kuo, R., and Lang, S.** Direct three-dimensional finite element analysis of human vertebral cancellous bone. In "Transactions of the Annual Meeting of the Orthopaedic Research Society," volume 17, page 551, 1992.

[70] **Garcıa, J., Doblaré, M., and Cegonino, J.** Bone remodelling simulation: a tool for implant design. *Computational Materials Science*, 25(1):100–114, 2002.

[71] **Gee, M., Dohrmann, C., Key, S., and Wall, W.** A uniform nodal strain tetrahedron with isochoric stabilization. *International Journal for Numerical Methods in Engineering*, 78(4): 429–443, 2009.

[72] **Geris, L., Gerisch, A., Sloten, J., Weiner, R., and Oosterwyck, H.** Angiogenesis in bone fracture healing: a bioregulatory model. *Journal of theoretical biology*, 251(1):137–158, 2008.

[73] **Geuzaine, C. and Remacle, J.-F.** Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[74] **Gibson, S.** Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. In "MICCAI 1998, Medical Image Computing and Computer-Assisted Intervention," volume 1496 of *Lecture Notes in Computer Science*, pages 888–898, 1998.

[75] **Gibson, S. F. F.** Using distance maps for accurate surface representation in sampled volumes. In "Proceedings of the ACM-IEEE symposium on Volume visualization," pages 23–30, New York, USA, 1998.

[76] **Gignac, D., Aubin, C., Dansereau, J., and Labelle, H.** Optimization method for 3D bracing correction of scoliosis using a finite element model. *European Spine Journal*, 9(3):185–190, 2000.

[77] **Grimson, W., Ettinger, G., White, S., Lozano-Perez, T., Wells Iii, W., and Kikinis, R.** An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization. *IEEE Transactions on Medical Imaging*, 15(2):129–140, 1996.

[78] **Grosland, N. and Brown, T.** A voxel-based formulation for contact finite element analysis. *Computer Methods in Biomechanics and Biomedical Engineering*, 5(1):21–32, 2002.

[79] **Hausdorff, F.** Dimension und äußeres Maß. *Mathematische Annalen*, 79(1):157–179, 1918.

[80] **Hege, H., Seebass, M., Stalling, D., and Zöckler, M.** *A generalized marching cubes algorithm based on non-binary classifications*. preprint SC 97-05, Berlin, Germany, 1997.

[81] **Hilber, H., Hughes, T., and Taylor, R.** Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering and Structural Dynamics*, 5(3): 283–292, 1977.

[82] **Hollister, S., Brennan, J., and Kikuchi, N.** A homogenization sampling procedure for calculating trabecular bone effective stiffness and tissue level stress. *Journal of Biomechanics*, 27(4):433–444, 1994.

[83] **Holzapfel, G. A.** Computational Biomechanics of Soft Biological Tissue. In "Encyclopedia of Computational Mechanics," John Wiley & Sons, Ltd, 2004.

[84] **Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W.** Surface reconstruction from unorganized points. *ACM SIGGRAPH Computer Graphics*, 26:71–78, July 1992.

[85] **Hughes, T. J., Franca, L. P., and Balestra, M.** A new finite element formulation for computational fluid dynamics: V. Circumventing the babuska-brezzi condition: a stable Petrov-Galerkin formulation of the stokes problem accommodating equal-order interpolations. *Computer Methods in Applied Mechanics and Engineering*, 59(1):85 – 99, 1986.

[86] **Hughes, T.** *The finite element method: linear static and dynamic finite element analysis*. Prentice-hall, 1987.

[87] **Hunter, P., Li, W., McCulloch, A., and Noble, D.** Multiscale modeling: Physiome project standards, tools, and databases. *Computer*, 39(11):48–54, 2006.

[88] **Huttenlocher, D., Klanderman, G., and Rucklidge, W.** Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[89] **Innocenti, B., Truyens, E., Labey, L., Wong, P., Victor, J., and Bellemans, J.** Journal of Orthopaedic Surgery and Research. *Journal of orthopaedic surgery and research*, 4:26, 2009.

[90] **Innocenti, B., van Jonbergen, H., Labey, L., and Verdonschot, N.** Influence of Design on Potential Periprosthetic Stress Shielding: A Finite Element Analysis. *Journal of Bone & Joint Surgery, British Volume*, 94(SUPP XL):74–74, 2012.

[91] **Joldes, G., Wittek, A., and Miller, K.** Non-locking tetrahedral finite element for surgical simulation. *Communications in Numerical Methods in Engineering*, 25(7):827–836, 2009.

[92] **Kaneps, A., Stover, S., and Lane, N.** Changes in canine cortical and cancellous bone mechanical properties following immobilization and remobilization with exercise. *Bone*, 21(5): 419–423, 1997.

[93] **Keyak, J. and Rossi, S.** Prediction of femoral fracture load using finite element models: an examination of stress-and strain-based failure theories. *Journal of Biomechanics*, 33(2):209, 2000.

[94] **Keyak, J., Meagher, J., Skinner, H., and Mote Jr., C.** Automated three-dimensional finite element modelling of bone: A new method. *Journal of Biomedical Engineering*, 12(5):389–397, 1990.

[95] **Kidwell, C., Chalela, J., Saver, J., Starkman, S., Hill, M., Demchuk, A., Butman, J., Patronas, N., Alger, J., Latour, L., et al.** Comparison of MRI and CT for detection of acute intracerebral hemorrhage. *JAMA: The Journal of the American Medical Association*, 292(15): 1823–1830, 2004.

[96] **Klaas, O., Maniatty, A., and Shephard, M. S.** A stabilized mixed finite element method for finite elasticity: Formulation for linear displacement and pressure interpolation. *Computer Methods in Applied Mechanics and Engineering*, 180(1-2):65 – 79, 1999.

[97] **Kobbelt, L., Botsch, M., Schwanecke, U., and Seidel, H.** Feature sensitive surface extraction from volume data. In "Proceedings of the 28th annual ACM conference on Computer graphics and interactive techniques," pages 57–66, 2001.

[98] **Kolluri, R.** Provably good moving least squares. *ACM Trans. Algorithms*, 4(2):18:1–18:25, May 2008.

[99] **Krysl, P. and Zhu, B.** Locking-free continuum displacement finite elements with nodal integration. *International Journal for Numerical Methods in Engineering*, 76(7):1020–1043, 2008.

[100] **Lancaster, P. and Salkauskas, K.** Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.

[101] **Laschet, G., Caylak, I., Benke, S., and Mahnken, R.** Locally integrated node-based formulations for four-node tetrahedral meshes. *International Journal for Numerical Methods in Engineering*, pages 1 – 6, 2009.

[102] **Léonard, A., Guiot, L., Pirard, J., Crine, M., Balligand, M., and Blacher, S.** Non-destructive characterization of deer (Cervus Elaphus) antlers by X-ray microtomography coupled with image analysis. *Journal of Microscopy*, 225(3):258–263, 2007.

[103] **Lerios, A., Garfinkle, C., and Levoy, M.** Feature-based volume metamorphosis. In "Proceedings of the 22nd annual ACM conference on Computer graphics and interactive techniques," pages 449–456, 1995.

[104] **Levin, D.** The approximation power of moving least-squares. *Mathematics of Computation*, 67(224):1517–1532, 1998.

[105] **Libertiaux, V.** *Contribution to the study of the mechanical properties of brain tissue: fractional calculus-based model and experimental characterization*. PhD thesis, University of Liège, 2010.

[106] **Little, J. and Adam, C.** The effect of soft tissue properties on spinal flexibility in scoliosis: biomechanical simulation of fulcrum bending. *Spine*, 34(2):E76, 2009.

[107] **Little, J., Pearcy, M., and Pettet, G.** Parametric equations to represent the profile of the human intervertebral disc in the transverse plane. *Medical and Biological Engineering and Computing*, 45(10):939–945, 2007.

[108] **Liu, Y., Foteinos, P., Chernikov, A., and Chrisochoides, N.** Multi-tissue mesh generation for brain images. In "Proceedings of the 19th International Meshing Roundtable," pages 367–384. Springer, 2010.

[109] **Lo, S.** A new mesh generation scheme for arbitrary planar domains. *International Journal for Numerical Methods in Engineering*, 21(8):1403–1426, 1985.

[110] **Lorensen, W. E. and Cline, H. E.** Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21:163–169, August 1987.

[111] **LTAS-MN^{2}L**. ULg. `http://metafor.ltas.ulg.ac.be/`.

[112] **Mahnken, R. and Caylak, I.** Stabilization of bi-linear mixed finite elements for tetrahedra with enhanced interpolation using volume and area bubble functions. *International Journal for Numerical Methods in Engineering*, 75(4):377–413, 2008.

[113] **Mahnken, R., Caylak, I., and Laschet, G.** Two mixed finite element formulations with area bubble functions for tetrahedral elements. *Computer Methods in Applied Mechanics and Engineering*, 197(9–12):1147 – 1165, 2008.

[114] **Manual, A. U.** Version 6.5, Hibbitt, Karlsson and Sorensen. *Inc., Pawtucket, RI*, 2004.

[115] **Marcellin-Little, D.** Humeral fractures in dogs. *FOCUS*, 8(3), 1998.

[116] **Mengoni, M., d'Otreppe, V., and Ponthot, J.** A bone remodelling model for long term orthodontic tooth movement. *Journal of Biomechanics*, 45:iv, S1–S677, 2012.

[117] **Mengoni, M.** *On the development of an integrated bone remodeling law for orthodontic tooth-movements models using the Finite Element Method*. PhD thesis, University of Liège, 2012.

[118] **Meyer, M., Whitaker, R., Kirby, R. M., Ledergerber, C., and Pfister, H.** Particle-based Sampling and Meshing of Surfaces in Multimaterial Volumes. *IEEE Transactions on Visualization*

and Computer Graphics, 14:1539–1546, November 2008.

[119] **Miller, K.** *Biomechanics of brain for computer integrated surgery*. Warsaw University of Technology Publishing House Warsaw, Poland, 2002.

[120] **Molino, N., Bridson, R., Teran, J., and Fedkiw, R.** A crystalline, red green strategy for meshing highly deformable objects with tetrahedra. In "12th International Meshing Roundtable," pages 103–114. Citeseer, 2003.

[121] **Montani, C., Scateni, R., and Scopigno, R.** A modified look-up table for implicit disambiguation of marching cubes. *The Visual Computer*, 10(6):353–355, 1994.

[122] **Moores, A.** Humeral condylar fractures and incomplete ossification of the humeral condyle in dogs. *In Practice*, 28(7):391, 2006.

[123] **Moran, B., Ortiz, M., and Shih, C.** Formulation of implicit finite element methods for multiplicative finite deformation plasticity. *International Journal for Numerical Methods in Engineering*, 29(3):483–514, 1990.

[124] **Müller, H.** Boundary extraction for rasterized motion planning. In T. K. Horst Bunke and H. Noltemeier, editors, "Modelling and Planning for Sensor Based Intelligent Robot Systems," pages 41–50, 1995.

[125] **Nahum, A. and Melvin, J.** *Accidental injury: biomechanics and prevention*. Springer Verlag, 2002.

[126] **Narayan, K., Rao, K., and Sarcar, M.** *Computer aided design and manufacturing*. PHI Learning Pvt. Ltd., 2008.

[127] **Nash, M. and Hunter, P.** Computational mechanics of the heart. *Journal of elasticity*, 61(1): 113–141, 2000.

[128] **Natarajan, B.** On generating topologically consistent isosurfaces from uniform samples. *The Visual Computer*, 11(1):52–62, 1994.

[129] **Nazarian, A. and Müller, R.** Time-lapsed microstructural imaging of bone failure behavior. *Journal of Biomechanics*, 37(1):55–65, 2004.

[130] **Negi, L.** *Strength Of Materials*. 2007.

[131] **Newmark, N.** A method of computation for structural dynamics. *Journal of the Engineering Mechanics Division*, 85(7):67–94, 1959.

[132] **Nguyen-Thoi, T., Vu-Do, H., Rabczuk, T., and Nguyen-Xuan, H.** A node-based smoothed finite element method (NS-FEM) for upper bound solution to visco-elastoplastic analyses of solids using triangular and tetrahedral meshes. *Computer Methods in Applied Mechanics and Engineering*, 199(45-48):3005–3027, 2010.

[133] **Nielson, G. M.** Dual Marching Cubes. In "Proceedings of the IEEE conference on Visualization," pages 489–496, Washington, DC, USA, 2004.

[134] **Ning, P. and Bloomenthal, J.** An Evaluation of Implicit Surface Tilers. *IEEE Computer Graphics and Applications*, 13:33–41, 1993.

[135] **Nsiampa, N., d'Otreppe, V., Robbe, C., Papy, A., and Ponthot, J.** Development of a thorax CT-scan based finite element model for thoracic injury assessment. 2011.

[136] **Ohtake, Y., Belyaev, A., and Seidel, H.** Multi-scale and Adaptive CS-RBFs for Shape Reconstruction from Clouds of Points. *Advances in Multiresolution for Geometric Modelling*, pages 143–154, 2005.

[137] **Ohtake, Y., Belyaev, A., and Bogaevski, I.** Mesh regularization and adaptive smoothing. *Computer-Aided Design*, 33(11):789 – 800, 2001.

[138] **Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., and Seidel, H.-P.** Multi-level partition of unity implicits. *ACM Transactions on Graphics*, 22:463–470, July 2003.

[139] **o'Rourke, J.** *Computational geometry in C*. Cambridge University Press, 1998.

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

[141] **O'Toole, R., Jaramaz, B., DiGioia, A., Visnic, C., and Reid, R.** Biomechanics for preoperative planning and surgical simulations in orthopaedics. *Computers in Biology and Medicine*, 25(2):183–191, 1995.

[142] **Pieper, S., Halle, M., and Kikinis, R.** 3D Slicer. In "Proceedings of the IEEE International Symposium on Biomedical Imaging: From Nano to Macro," volume 1, pages 632–635, 2004.

[143] **Pistoia, W., Van Rietbergen, B., Lochmüller, E., Lill, C., Eckstein, F., and Rüegsegger, P.** Estimation of distal radius failure load with micro-finite element analysis models based on three-dimensional peripheral quantitative computed tomography images. *Bone*, 30(6): 842–848, 2002.

[144] **Pons, J., Ségonne, F., Boissonnat, J., Rineau, L., Yvinec, M., and Keriven, R.** High-Quality Consistent Meshing of Multi-label Datasets. In "Information Processing in Medical Imaging," volume 4584 of *Lecture Notes in Computer Science*, pages 198–210. Springer, 2007.

[145] **Ponthot, J.-P.** *Traitement unifié de la Mécanique des Milieux Continus solides en grandes transformations par la méthode des éléments finis*. PhD thesis, Liège, Belgium, 1995.

[146] **Preparata, F. and Shamos, M.** Introduction. *Computational Geometry*, pages 1–35, 1985.

[147] **Puso, M. and Solberg, J.** A stabilized nodally integrated tetrahedral. *International Journal for Numerical Methods in Engineering*, 67(6):841–867, 2006.

[148] **Reitinger, B., Bornik, A., and Beichel, R.** Consistent mesh generation for non-binary medical datasets. *Bildverarbeitung für die Medizin 2005*, pages 183–187, 2005.

[149] **Richter, E.** Basic biomechanics of dental implants in prosthetic dentistry. *The Journal of prosthetic dentistry*, 61(5):602–609, 1989.

[150] **Savchenko, V. V., Pasko, A. A., Okunev, O. G., and Kunii, T. L.** Function Representation of Solids Reconstructed from Scattered Surface Points and Contours. *Computer Graphics Forum*, 14(4):181–188, 1995.

[151] **Schneider, G., Raithby, G., and Yovanovich, M.** Finite-element solution procedures for solving the incompressible, Navier-Stokes equations using equal order variable interpolation. *Numerical Heat Transfer, Part B: Fundamentals*, 1(4):433–451, 1978.

[152] **Schroeder, W. J., Zarge, J. A., and Lorensen, W. E.** Decimation of triangle meshes. *ACM SIGGRAPH Computer Graphics*, 26:65–70, July 1992.

[153] **Sermesant, M., Moireau, P., Camara, O., Sainte-Marie, J., Andriantsimiavona, R., Cimrman, R., Hill, D., Chapelle, D., and Razavi, R.** Cardiac function estimation from MRI using a heart model and data assimilation: Advances and difficulties. *Medical Image Analysis*, 10 (4):642–656, 2006.

[154] **Shen, C., O'Brien, J., and Shewchuk, J.** Interpolating and approximating implicit surfaces from polygon soup. *ACM Transactions on Graphics*, 23(3):896–904, 2004.

[155] **Shepard, D.** A two-dimensional interpolation function for irregularly-spaced data. In "Proceedings of the 23rd ACM national conference," pages 517–524, 1968.

[156] **Shephard, M. and Georges, M.** Automatic three-dimensional mesh generation by the finite octree technique. *International Journal for Numerical Methods in Engineering*, 32(4):709–749, 1991.

[157] **Shigley, J., Mischke, C., Budynas, R., Liu, X., and Gao, Z.** *Mechanical engineering design*, volume 89. McGraw-Hill New York, 1989.

[158] **Si, H.** Tetgen: a quality tetrahedral mesh generator and three-dimensional Delaunay triangulator. Technical Report I.4, Weierstrass-Institute, Berlin, 2006.

[159] **Simo, J. C. and Armero, F.** Geometrically non-linear enhanced strain mixed methods and the method of incompatible modes. *International Journal for Numerical Methods in Engineering*, 33(7):1413–1449, 1992.

[160] **Simo, J. and Rifai, M.** A class of mixed assumed strain methods and the method of incompatible modes. *International Journal for Numerical Methods in Engineering*, 29(8):1595–1638, 1990.

[161] **Simo, J., Taylor, R., and Pister, K.** Variational and projection methods for the volume constraint in finite deformation elasto-plasticity. *Computer Methods in Applied Mechanics and Engineering*, 51(1-3):177–208, 1985.

[162] **Stimpson, C., Ernst, C., Knupp, P., Pébay, P., and Thompson, D.** The Verdict Library Reference Manual. 2007.

[163] **Taubin, G.** Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:1115–1138, 1991.

[164] **Taylor, C. and Figueroa, C.** Patient-specific modeling of cardiovascular mechanics. *Annual review of biomedical engineering*, 11:109–134, 2009.

[165] **Taylor, R.** A mixed-enhanced formulation tetrahedral finite elements. *International Journal for Numerical Methods in Engineering*, 47(1-3):205–227, 2000.

[166] **Taylor, Z., Cheng, M., and Ourselin, S.** High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. *IEEE Transactions on Medical Imaging*, 27(5):650–663, 2008.

[167] **Tobor, I., Reuter, P., and Schlick, C.** Efficient reconstruction of large scattered geometric datasets using the partition of unity and radial basis functions. *Journal of WSCG 2004*, 12 (3):467–474, 2004.

[168] **Torii, R., Oshima, M., Kobayashi, T., Takagi, K., and Tezduyar, T.** Fluid–structure interaction modeling of a patient-specific cerebral aneurysm: influence of structural modeling.

Computational Mechanics, 43(1):151–159, 2008.

[169] **Turk, G. and O'brien, J. F.** Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21:855–873, October 2002.

[170] **Van Gelder, A. and Wilhelms, J.** Topological considerations in isosurface generation. *ACM Transactions on Graphics*, 13(4):337–375, 1994.

[171] **van Jonbergen, H., Innocenti, B., Gervasi, G., Labey, L., and Verdonschot, N.** Differences in the stress distribution in the distal femur between patellofemoral joint replacement and total knee replacement: a finite element study. *Journal of Orthopaedic Surgery and Research*, 7(1):28, 2012.

[172] **Van Rietbergen, B., Weinans, H., Huiskes, R., and Odgaard, A.** A new method to determine trabecular bone elastic properties and loading using micromechanical finite-element models. *Journal of Biomechanics*, 28(1):69–81, 1995.

[173] **Viceconti, M., Bellingeri, L., Cristofolini, L., and Toni, A.** A comparative study on different methods of automatic mesh generation of human femurs. *Medical Engineering and Physics*, 20(1):1–10, 1998.

[174] **Viceconti, M., Taddei, F., Van Sint Jan, S., Leardini, A., Cristofolini, L., Stea, S., Baruffaldi, F., and Baleani, M.** Multiscale modelling of the skeleton for the prediction of the risk of fracture. *Clinical Biomechanics*, 23(7):845–852, 2008.

[175] **Vigneron, L.** *FEM/XFEM-Based Modeling of Brain Shift, Resection, and Retraction for Image-Guided Surgery*. PhD thesis, University of Liège, 2008.

[176] **Vigneron, L., Boman, R., Ponthot, J., Robe, P., Warfield, S., and Verly, J.** Enhanced FEM-based modeling of brain shift deformation in Image-Guided Neurosurgery. *Journal of Computational and Applied Mathematics*, 234(7):2046–2053, 2010.

[177] **Wang, M. Y. and Wang, X.** A level-set based variational method for design and optimization of heterogeneous objects. *Computer-Aided Design*, 37(3):321 – 337, 2005.

[178] **Wang, S. W. and Kaufman, A. E.** Volume sampled voxelization of geometric primitives. In "Proceedings of the IEEE conference on Visualization," pages 78–84, Washington, DC, USA, 1993.

[179] **Weinberg, L.** The biomechanics of force distribution in implant-supported prostheses. *International Journal of Oral and Maxillofacial Implants*, 8:19–19, 1993.

[180] **Wendland, H.** Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, 4:389–396, 1995.

[181] **Widany, K.-U., Caylak, I., and Mahnken, R.** Stabilized Mixed Tetrahedrals with Volume and Area Bubble Functions at Large Deformations. *Proceedings in Applied Mathematics and Mechanics*, 10(1):227–228, 2010.

[182] **Wilson, E., Taylor, R., Doherty, W., and Ghaboussi, J.** Incompatible displacement models (isoparametric finite elements in solid and thick shell structural analysis). *Numerical and computer methods in structural mechanics*, pages 43–57, 1973.

[183] **Wittek, A., Miller, K., Kikinis, R., and Warfield, S.** Patient-specific model of brain deformation: Application to medical image registration. *Journal of Biomechanics*, 40(4):919–929, 2007.

[184] **Wittek, A., Kikinis, R., Warfield, S., and Miller, K.** Brain Shift Computation Using a Fully Nonlinear Biomechanical Model. In "MICCAI 2005, Medical Image Computing and Computer-Assisted Intervention," volume 3750 of *Lecture Notes in Computer Science*, pages 583–590, 2005.

[185] **Wood, W., Bossak, M., and Zienkiewicz, O.** An alpha modification of Newmark's method. *International Journal for Numerical Methods in Engineering*, 15(10):1562–1566, 1980.

[186] **Wu, B. and Wang, S.** Automatic triangulation over three-dimensional parametric surfaces based on advancing front method. *Finite Elements in Analysis and Design*, 41(9-10):892–910, 2005.

[187] **Wu, Z. and Sullivan Jr, J.** Multiple material marching cubes algorithm. *International Journal for Numerical Methods in Engineering*, 58(2):189–207, 2003.

[188] **Yerry, M. and Shephard, M.** Automatic three-dimensional mesh generation by the modified-octree technique. *International Journal for Numerical Methods in Engineering*, 20(11):1965–1990, 1984.

[189] **Yoshizawa, S., Belyaev, A. G., and Seidel, H.-P.** A Simple Approach to Interactive Free-Form Shape Deformations. In "Proceedings of the IEEE Pacific Conference on Computer Graphics and Applications," page 471, Los Alamitos, CA, USA, 2002.

[190] **Zhang, Y. and Bajaj, C.** Adaptive and quality quadrilateral/hexahedral meshing from volumetric data. *Computer Methods in Applied Mechanics and Engineering*, 195(9-12):942–960, 2006.

[191] **Zhang, Y., Hughes, T. J., and Bajaj, C. L.** An automatic 3D mesh generation method for domains with multiple materials. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):405–415, 2010.

[192] **Zhao, H.-K., Osher, S., and Fedkiw, R.** Fast Surface Reconstruction Using the Level Set Method. In "Proceedings of the IEEE Workshop on Variational and Level Set Methods in Computer Vision," pages 194–201, Los Alamitos, CA, USA, 2001.

[193] **Zienkiewicz, O. C., Rojek, J., Taylor, R. L., and Pastor, M.** Triangles and tetrahedra in explicit dynamic codes for solids. *International Journal for Numerical Methods in Engineering*, 43(3):565–583, 1998.

[194] **Zienkiewicz, O., Taylor, R., and Taylor, R.** *The finite element method for solid and structural mechanics*, volume 2. Butterworth-Heinemann, 2005.