ON MESHLESS AND NODAL-BASED NUMERICAL METHODS FOR FORMING PROCESSES

an adaptive smoothed finite element method

W. QUAK

ON MESHLESS AND NODAL-BASED NUMERICAL METHODS FOR FORMING PROCESSES

AN ADAPTIVE SMOOTHED FINITE ELEMENT METHOD

W. Quak

Samenstelling van de promotiecommissie:

voorzitter en secretaris:	
Prof. dr. F. Eising	Universiteit Twente
promotor:	
Prof. dr. ir. J. Huétink	Universiteit Twente
assistent promotor:	
Dr. ir. A.H. van den Boogaard	Universiteit Twente
leden:	
Prof. dr. E. Cueto	Universidad de Zaragoza
Prof. dr. ir. H.W.M. Hoeijmakers	Universiteit Twente
Prof. dr. rer. -nat S. Luding	Universiteit Twente
Dr. ir. R.H.J. Peerlings	Technische Universiteit Eindhoven
Prof. dr. ir. L.J. Sluys	Technische Universiteit Delft

ISBN 978-90-365-3238-9

DOI 10.3990/1.9789036532389

1st printing September 2011

Keywords: meshless methods, nodal-based methods, adaptive smoothed finite elements, numerical simulations, forming processes

This thesis is prepared with LaTeX by the author and printed by Ipskamp Drukkers, Enschede, from an electronic document.

The cover of this thesis is designed by Hanneke Quak and the author.

ON MESHLESS AND NODAL-BASED NUMERICAL METHODS FOR FORMING PROCESSES

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. H. Brinksma,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 7 oktober 2011 om 14.45 uur

door

Wouter Quak

geboren op 22 juni 1982
te Hengelo(OV), Nederland

Dit proefschrift is goedgekeurd door de promotor:

Prof. dr. ir. J. Huétink

en de assistent promotor:

Dr. ir. A.H. van den Boogaard

Summary

Currently, the main tool for the simulation of forming processes is the finite element method. Unfortunately, for processes involving very large deformations, for instance extrusion or forging, finite elements based on a Lagrangian description of the kinematics are problematic. Results can become inaccurate and even lose their physical meaning as a result of the distortion of the finite elements.

There are techniques available to avoid or reduce element distortion problems. The essence of these techniques is to decouple the motion of the material from the motion of the element mesh. Examples are ALE/Eulerian formulations or re-meshing strategies. Nonetheless, a Lagrangian description of motion is often preferred. By expressing the governing equations in material coordinates there is no need for numerical convection schemes or mapping algorithms. Hence the drawbacks related to these numerical schemes are of no concern. Besides, keeping track of the free boundary is straightforward.

In the 1980s, a new group of numerical methods emerged. This group is entitled meshless methods, and aims at avoiding problems related to the use of a mesh. Whereas finite elements base their shape functions on elements, meshless methods base their shape functions purely on nodal positions. Consequently, this nodal-based approach does not restrict the relative motion of nodes by shape criteria related to elements. The goal of the research as presented in this thesis is to develop a meshless method for solving forming processes involving large deformations in a more efficient manner than currently possible with finite elements.

The first step in this research was to select a single method for further development out of the large number of methods that have been proposed in the course of time. For this comparative study three meshless shape functions and two numerical integration schemes to evaluate the weak form were selected. It can be concluded that diffuse meshless shape functions, like moving least squares and local maximum entropy approximations, are more accurate than simple linear interpolation upon a Delaunay triangle. However, the computational effort for these two diffuse functions is approximately a factor seven to fifteen higher than for the linear triangle interpolation. Concerning the numerical integration of the weak form, the use of a Gaussian integration scheme results either in volumetric locking or in instabilities. A nodal integration scheme, on the contrary, performs very well. Volumetric locking is absent and good accuracy is obtained on highly irregular nodal grids.

Taking these two conclusions on shape functions and numerical integration into consideration, the linear triangle interpolation in combination with a nodal integration scheme was chosen for further development. For this combination, an extension to large deformations was presented. The new method, named Adaptive Smoothed Finite Elements (ASFEM), is based on a cloud of nodes following a Lagrangian description of motion and a triangulation algorithm that sets up the connectivity between these nodes for each increment. As a result of the nodal integration scheme, the constitutive behaviour is evaluated at the nodal positions. The re-triangulation of the cloud of nodes does therefore not require the mapping of state variables associated with the material model.

A computer code incorporating an implementation of the ASFEM method was developed and tested on the simulation of two forming processes. Firstly, the forging of a steel circular rod was analysed. Since the deformations in this process are large, but not as extreme as for instance for an extrusion problem, a finite element simulation based on a Lagrangian formulation was performed of the same problem. Close agreement is observed between the results of both methods. For the ASFEM simulations, triangles with an optimal shape were generated for each increment. As a result, it was shown that the triangulation of the ASFEM method is of better quality than the finite element mesh. Secondly, the extrusion of a simplified aluminium profile was simulated. After the simulation of the start-up phenomena, the ASFEM simulation reached a 'steady state'. The results at this state compare well to a Eulerian finite element simulation. To conclude, for both processes large deformations were simulated successfully without failure of the algorithm as a result of problems related to the mesh.

Samenvatting

Tegenwoordig is de eindige-elementenmethode het voornaamste gereedschap om een vormgevingsproces te simuleren. Echter, voor processen waarin zeer grote vervormingen optreden, zoals het extrusie- of smeedproces, is het gebruik van eindige-elementen in combinatie met een Lagrangiaanse beschrijving van de kinematica problematisch. Door de vervorming van de eindige-elementen kunnen de resultaten onnauwkeurig worden en zelfs hun fysische betekenis verliezen.

Er zijn methoden beschikbaar om het probleem van element vervorming te voorkomen of te verminderen. De essentie van deze methoden is om de beweging van het materiaal te ontkoppelen van de beweging van de elementen mesh. Voorbeelden hiervan zijn Euleriaanse/ALE beschrijvingen en re-meshing strategieën. Desalniettemin wordt aan een Lagrangiaanse beschrijving van de kinematica in veel gevallen de voorkeur gegeven. Door de vergelijkingen uit te drukken in Lagrangiaanse coördinaten zijn numerieke convectie- of mappingschema's niet nodig en voorkomt men de bijbehorende nadelen van deze algoritmen. Bovendien is de beschrijving van de vrije rand in Lagrangiaanse coordinaten ongecompliceerd.

Omstreeks 1980 zijn ontwikkelingen begonnen betreffende een nieuwe groep van numerieke methoden. Deze groep, bekend onder de naam 'meshless methoden', heeft als doel het voorkomen van problemen die verband houden met het gebruik van een mesh. Waar de shapefuncties in het geval van eindige-elementen zijn gebaseerd op een mesh, zijn voor meshless methoden de shapefuncties uitsluitend gebaseerd op de posities van de knooppunten. Deze knooppunt-gebaseerde beschrijving van het continuüm legt geen restricties op aan de relatieve verplaatsing van knopen door criteria, die zijn gerelateerd aan de kwaliteit van de mesh. Het doel van het in dit proefschrift behandelde onderzoek is om een meshless methode te ontwikkelen, waarmee grote vervormingen in een vormgevingsproces efficiënter kunnen worden gesimuleerd dan thans mogelijk is met eindige-elementen.

De eerste stap van dit onderzoek betreft de selectie van een enkele methode uit de talrijke meshless methoden die in de loop van de tijd zijn ontwikkeld. Voor deze vergelijkende studie zijn drie meshless shapefuncties en twee numerieke integratieschema's bestudeerd. De eerste conclusie volgend uit dit onderzoek is dat diffuse meshless shapefuncties, zoals moving least squares- en local maximum entropy functies, een hogere nauwkeurigheid geven dan simpele lineaire interpolatie gebaseerd op driehoeken. De twee eerstgenoemde functies vergen echter wel een factor 7 tot 15

meer rekentijd dan de laatstgenoemde voor hetzelfde aantal knooppunten. De tweede conclusie betreft de numerieke integratie van de zwakke formulering. Het gebruik van een Gaussiaans integratieschema resulteert in locking verschijnselen of instabiele resultaten. Een knooppunt-integratieschema daarentegen geeft goede resultaten, volumetrische locking is afwezig en de nauwkeurigheid op irreguliere rasters is goed.

Op basis van deze twee conclusies betreffende shapefunctie en integratieschema is de lineaire driehoeksinterpolatie in combinatie met het knooppunt-integratieschema gekozen voor verdere ontwikkeling. Een extensie voor grote vervormingen is ontwikkeld voor deze combinatie. De resulterende methode, genaamd Adaptive Smoothed Finite Elements (ASFEM), gaat uit van een wolk van knopen die een Lagrangiaanse beschrijving van de beweging volgt, alsmede een triangulatie algoritme, waarmee de connectiviteit tussen de knopen voor iedere tijdstap opnieuw wordt bepaald. Doordat het constitutief gedrag wordt geëvalueerd op de posities van de knooppunten, kan de knopenwolk opnieuw worden getrianguleerd, zonder de toestandsvariabelen van het materiaalmodel te projecteren op de nieuwe triangulatie.

Een computerprogramma met een implementatie van de methode is ontwikkeld en getest op de simulatie van twee vormgevingsprocessen. Ten eerste is het smeden van een rond stuk stalen staf gesimuleerd. Aangezien de vervormingen die optreden bij dit proces niet dermate groot zijn zoals bijvoorbeeld bij een extrusieproces, is een Lagrangiaanse FEM berekening van hetzelfde proces uitgevoerd. Beide simulaties komen in grote mate met elkaar overeen. Voor elke tijdstap van de ASFEM simulatie zijn driehoeken met een optimale vorm gegenereerd en als resultaat hiervan is de kwaliteit van de ASFEM triangulatie beter dan de mesh van de FEM simulatie. Ten tweede is het extruderen van een vereenvoudigd aluminium profiel gesimuleerd. Enige tijd na het opstarten van de ASFEM simulatie wordt een ‘stationaire’ toestand bereikt. De resultaten van deze toestand komen goed overeen met een Euleriaanse FEM berekening. Afsluitend kan er worden geconcludeerd dat in beide vormgevingsprocessen grote vervormingen succesvol zijn gesimuleerd zonder het falen van het algoritme door problemen gerelateerd aan het gebruik van een mesh.

Nomenclature

Roman

$\mathbf{a}$	vector of coefficients
$\mathbf{d}$	displacement degrees of freedom
e	residual or error
$\mathbf{e}$	base vector
$\mathbf{f}$	volume force
$\mathbf{g}$	vector related to Lagrangian multpliers
h	average nodal spacing
$\mathbf{n}$	normal vector on the boundary of the domain
$\mathbf{m}$	vector of volumetric operator
$\mathbf{p}$	polynomial base vector
$\mathbf{r}$	base vector
$\mathbf{s}$	location vector
t	time
$\mathbf{t}$	traction force on a surface
$\mathbf{u}$	displacement vector
$\mathbf{v}$	velocity vector
$\mathbf{x}$	location vector
$\mathbf{B}$	strain-displacement matrix
$\mathbf{C}$	tangent of constitutive law
$\mathbf{D}$	rate of deformation
E	Young's modulus
$\mathbf{F}$	force vector
$\mathbf{F}$	deformation gradient
$\mathbf{G}$	matrix related to Lagrangian multipliers
$\mathbf{K}$	stiffness matrix
$\mathbf{L}$	velocity gradient
$\mathbf{N}$	displacement interpolation matrix
$\mathbf{P}$	regression matrix
$\mathbf{R}$	rotation tensor
S	the boundary of a domain
$\mathbf{S}$	norm matrix

$\mathbf{T}$	stress matrix
$\mathbf{U}, \mathbf{V}$	displacement degrees of freedom
V	volume
V_i	volume accompanying node i
V_{ijk}	triangle spanned by node i, j and k
W	work or energy
$\mathbf{W}$	spin tensor
$\mathbf{X}$	location vector

Greek

α	α-shape parameter
β, γ, μ	parameters to control the domain of influence
δ	prefix for a virtual quantity
δ_{ij}	Kronecker delta
$\boldsymbol{\varepsilon}$	linear strain tensor
ϑ	inf-sup value
κ	stabilisation parameter
λ	Eigenvalue
$\boldsymbol{\lambda}$	vector of Lagrangian multipliers or Eigenvalues
ν	Poisson's ratio
ϕ_i	trial shape function of node i
ψ_i	test shape function of node i
$\boldsymbol{\xi}$, $\mathcal{X}$	location vectors
$\boldsymbol{\sigma}$	Cauchy stress tensor
τ	time
ω	kernel function
Γ	boundary
Δ	prefix for an incremental quantity
Π	energy or potential
Ω	the volume of a domain
$\Re^n$	n-dimensional space

Various

$\{\dots\}$	definition of a set
$\{\dots\}, [\dots]$	tensors in Voigt form
$\tilde{a}$	a is a prescribed quantity
$\bar{a}$	a is an assumed quantity
$\hat{a}$	a is an transformed quantity
$\dot{a}$	material time derivative of a
$\vert$	restricted by
$\Vert \dots \Vert$	norm

$\in$	element in set
$\forall$	for all elements out of set
$\emptyset$	an empty set
inf	infimum
sup	supremum
$\cdot$	single tensor contraction
:	double tensor contraction
$\times$	tensor cross product
$\nabla, \overrightarrow{\nabla}, \overleftarrow{\nabla}$	spatial gradient operators
$d(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{b})$	Euclidean distance between point $\mathbf{x}_\mathrm{a}$ and $\mathbf{x}_\mathrm{b}$
C^n	continuity of the n-th order
$\ldots^\mathrm{T}$	transpose
$\ldots_\mathrm{int}$	internal
$\ldots_\mathrm{ext}$	external
$\ldots_\mathrm{eq}$	equivalent
$\ldots_\mathrm{dev}$	deviatoric part
$\ldots_\mathrm{vol}$	volumetric part
$\ldots^\mathrm{J}$	related to the Jaumann rate
$\ldots_\mathrm{stab}$	related to stabilisation
N_nod	number of nodes in the domain
$\mathrm{tr}(\mathbf{a})$	the trace of tensor $\mathbf{a}$
$\mathrm{diag}[\mathbf{a}]$	sum of the diagonal terms of $\mathbf{a}$
$\mathbf{a} \Longleftrightarrow \{\mathbf{a}\}$	$\mathbf{a}$ in tensor form is equivalent to $\mathbf{a}$ in Voigt form

Contents

1

Introduction

Numerical simulation tools are becoming of crucial importance in the field of engineering. The increase in computing power and the ever more advanced and sophisticated mathematical models have stimulated their widespread use in all disciplines of engineering.

In many cases, the term 'numerical simulation tools' can simply be replaced by the term 'finite elements'. The Finite Element Method (FEM) is a tool that is currently being widely used for the simulation of engineering problems. First developments on finite elements started in the 1940s though only in the 1980s, after the widespread accessibility to computers, the full potential of the method could be exploited.

Since their introduction, various types of elements have been developed. Examples are shell elements, thermal elements, solid elements, beams, bars, mixed elements and many more. Because of their sound mathematical fundament, their results can be trusted if used in a proper manner. The finite element method has proven to be a valuable simulation tool for a large variety of forming processes and other problems in solid mechanics. Various commercial software programs are available, aimed at solving problems in a specific field, for instance in solid mechanics, fluid mechanics or thermal mechanics.

1.1 Motivation

In the field of forming processes, finite element simulations are frequently performed in order to gain valuable knowledge on the forming process at hand. The simulations can help to design and optimise the forming process, with the aim of producing products without defects and within required specifications.

Unfortunately, for forming processes involving very large deformations, the use of finite elements in a Lagrangian formulation becomes less straightforward and successful. Simulations such as extrusion or forging can be especially problematic as a result of the large relative motions of Lagrangian material points. Imagine for

instance the extrusion process in which the starting geometry of the process, a massive round billet, is converted to a final geometry, a slender long profile. This change in shape of the material can take place only if extreme deformations are applied to the material. The main problem with finite elements in this case is that their performance degrades if they become distorted. A quadrilateral element is most accurate if it is square. Making it for instance trapezoidal, has a negative effect on the accuracy of the results. Further distortion will eventually give nonphysical outcomes as a result of singularities or other numerical artifacts originating from the element's formulation.

There are techniques available to solve this drawback of finite elements, of which the main point is to decouple the deformation of the material from the deformation of the finite element mesh. So instead of 'etching' the mesh into the deforming material, which is known as the Lagrangian formulation of motion, the motion of the mesh and of the material are decoupled. A Eulerian or ALE (arbitrary Lagrangian-Eulerian) description aims at preserving the quality of the elements by allowing material to flow over element boundaries, similarly as in fluid mechanics. Another option is to simply re-mesh the domain, either locally or globally, at certain time steps. For several reasons, however, a Lagrangian description of motion is still preferred. Expressing the governing equations in material coordinates allows for the easy incorporation of arbitrary material models. Drawbacks of convection or mapping algorithms as required for ALE or re-meshing strategies are absent. Besides, keeping track of the free boundary is straightforward.

Since the 1980s a new group of methods has emerged under the name 'meshless methods'. The main idea of these methods is to avoid the use of a mesh completely, such that issues related to mesh distortion are simply of no concern. The part to be analysed is represented by a set of nodes and the interaction between these nodes is not defined a priori by a set of interjacent elements. These interactions or, better said, shape functions are based on the nodal positions only. There is no need for a user-defined mesh to determine the connectivity between the nodes. With this nodal-based definition of the shape functions, two neighbouring nodes can be connected initially, while they can be far apart and not connected at a later stage of the simulation. Since finite elements base their shape functions upon a mesh which is constructed a priori, nodes remain connected irrespective of their current position. This element-based approach leads to the distortion problem as discussed previously. Hence, the nodal-based formulation, which does not constrain the relative motion of nodes by means of a shape quality criterion, is expected to be better suited for solving problems involving very large deformations.

Several forming processes involving the simulation of very large deformations can benefit from meshless or nodal-based methods. Examples of these processes are forging, extrusion, injection moulding, backward extrusion, cutting, rolling and friction stir-welding. Hence there is a large potential for a numerical method which simulates these type of processes efficiently.

1.2 Objective

The amount of deformation that can be simulated in a forming process in a Lagrangian formulation is restricted by limitations related to the numerical method used. The goal of the research as presented in this thesis is to develop a meshless or nodal-based method aimed at solving large deformations more efficiently than currently possible with finite elements. Not only is a good accuracy and a low computational cost of importance in order to make a meshless method efficient, but also the amount of deformation that can be simulated, the ease with which a model can be created (pre-processing) and the flexibility of the method in adaptive strategies (for instance in h-refinement) requires attention. Besides these practical demands, two other requirements need a closer look in order to develop a method that can be applied successfully in the simulation of forming processes.

Firstly, the forming of a product usually involves the nearly incompressible plastic flow of the material. In order to obtain physical results for this case, the developed method should be free of volumetric locking. Methods that suffer from volumetric locking are of no use for the simulation of plastic flow, and hence they are not suited for the simulation of most forming processes.

Secondly, the process time of the processes of interest are of such duration that terms related to the dynamics of the process can be neglected. An implicit time integration scheme is therefore the most sensible choice for the discretisation of time. As a consequence of this choice, it should be possible to extract a good linearisation of the force vector for this method in order to obtain convergence in a Newton–Raphson procedure.

1.3 Outline

Firstly, Chapter 2 of this thesis will present a general introduction to meshless methods. An overview of meshless methods will be given and the 'building blocks' of the methods will be explained by their basic formulations. After this introduction it will become clear that there are many meshless methods, and choosing the method that is most suited for the simulation of forming processes is not trivial. Therefore Chapter 3 will present a study which is aimed at selecting only one meshless method from the large number of meshless methods available. The presented research focuses on aspects of the method that are of relevance to the simulation of forming processes. The method that is selected for further development requires a stabilisation, for which a new strategy is proposed in Chapter 4. Whereas all previous developments were formulated for small deformations, Chapter 5 presents an extension to large deformations by using an updated Lagrangian formulation. The resulting method is entitled ASFEM and its formulations required for implementation into a computer code will be presented. Thereafter a set of small tests is performed in order to prove the validity of the newly proposed method. Finally, the applicability of the method in forming processes is investigated by simulating a forging and extrusion process in Chapter 6.

1.4 Notation

This thesis uses the notation and the accompanying symbols from various fields of research. For the majority of the formulas as presented in this thesis, the notation used will correspond to the commonly used notation from the originating discipline. However, for clarity and to avoid confusion, some remarks on the notation used are given below.

In more mathematically oriented finite element literature, an approximated field is usually denoted with the subscript 'h'. In this thesis, this subscript will not be used. An approximated displacement field is for instance simply given by $\mathbf{u}$ and not by $\mathbf{u}_{\mathrm{h}}$. If it concerns an exact field, the subscript 'exact' will be added. The exact displacement field is for example denoted by $\mathbf{u}_{\mathrm{exact}}$.

Tensors are always printed bold. A first order tensor is a bold letter of normal size. A second or higher order tensor is denoted with a bold capital letter. A tensor in between curly brackets { } or straight brackets [ ] is in Voigt form. See Appendix A for details on the difference between tensor and Voigt notation. Note that sets will be defined by curly brackets as well. From the context it will be clear whether a set or a tensor in Voigt form is meant.

In order to avoid the use of many symbols which are all related to the same object, one symbol will be used. The symbol Ω, for instance, can refer to the volume of a domain (a scalar quantity), the integration boundaries of a volume integral, or the set of points contained in a volume. From the context it will be clear what is meant.

2

An Introduction to Meshless Methods

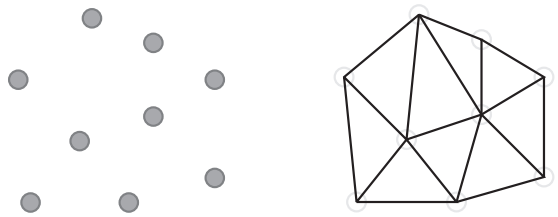
2.1 Introduction

In this chapter, the main properties and formulations of meshless methods will be presented and explained so that the reader is familiar with meshless concepts, which will appear later on in this thesis. Moreover, both a chronological as well as a categorial overview of meshless methods will be given in order to provide a view on the current state of the meshless techniques. Firstly, however, some basic questions regarding meshless methods will be answered. Several definitions are given in order to pinpoint the case in which a numerical method can be called a meshless method. Secondly, the most interesting properties of meshless methods are given which have motivated the research in this field for the last two decades.

This chapter is structured as follows. The basic aspects of meshless methods will be discussed in this section. The categorial and chronological overview of developments in the meshless field is given in Section 2.2. The main constituents of meshless methods and their accompanying formulations will be given in Sections 2.3, 2.4 and 2.5. This chapter will end with a closure and discussion in Section 2.6.

2.1.1 What is a Meshless Method?

Various definitions are given in literature which try to answer this question. Most of these definitions refer to the use of a mesh and the position of nodes in space. A node is a point in space where the independent degrees of freedom are evaluated. A mesh is a set of geometrical shapes, for instance triangles, quadrilaterals or tetrahedra, of which the vertices coincide with the nodal positions. See Figure 2.1 for a graphical representation of these two mathematical concepts.



(a) a cloud of nodes (b) a mesh

Figure 2.1: An illustration of the two main mathematical objects used for numerical strategies in computational continuum mechanics.

A first definition that is most trivial to define when reading the name 'meshless method' is:

> A meshless method is a method that does not depend on a mesh at all.

Although this definition seems to fit the term 'meshless' accurately, the definition is very stringent and many methods that are generally regarded as meshless methods do not meet this definition. Methods that do meet this definition are known by the name of 'truly' meshless methods. When compared to the complete set of meshless methods, this group is relatively small. Only a few methods operate without a mesh at all.

A broader definition is given by Idelsohn *et al.* [71]. Their definition is based on the two mathematical concepts of shape functions and nodes, where the first refers to a function used to approximate a field quantity. Their definition is as follows:

> A meshless method is a method in which the shape functions only depend on nodal positions.

The first thing that is apparent from the definition of Idelsohn *et al.* is that it refers only to the shape functions, not to any other component of the method. This can imply that a mesh can be used, as long as it is not used to construct shape functions. Clearly, this can be somewhat confusing since this means that a meshless method that conforms to this definition can still use a mesh. Many meshless methods nowadays operate in this manner. The shape functions are not defined upon a mesh, but other routines of the method, for instance the numerical integration, are based on a mesh. Typically, this mesh should be supplied by the user.

Several meshless methods that are of interest when simulating large deformations do not use a user-defined mesh since this mesh can get distorted easily. This subgroup of meshless methods conform to the following definition:

> A meshless method is a method in which the interactions of nodes only depend on their positions.

The idea here is that there can be a mesh, as long as it is not a fixed mesh generated by the user. A cloud of nodes is sufficient for the meshless method, and the way in which these nodes mutually interact is up to the method, leaving the question open as to whether there is a mesh to compute these interactions or not. As a result, the name meshless methods does not fit this definition well. Since the formulation of these methods is strongly dependent on the concept of nodes, the name 'nodal-based methods' would be more suitable. Finite elements, for example, clearly do not meet this definition. For a finite element simulation, the interaction between nodes is element-based. Not only a cloud of nodes is required, but also a list of connectivity which defines the elements.

To conclude, there is no unique definition which defines the group of meshless methods. However, it can be stated that in general the number of meshless methods is much larger than one may expect from the name meshless methods.

2.1.2 Why use a Meshless Method?

The first method that was regarded as a meshless method was developed in the 1980s. Since then, this group of methods has drawn considerable attention and many new methods have been proposed. The number of meshless methods is vast and developments are still ongoing. Below, four of the most interesting properties are given which have motivated the research in this field for the last three decades.

Firstly, the results obtained with meshless methods are *less mesh dependent* than results obtained for instance with finite elements. For this reason, meshless methods have attracted much attention, especially for the simulation of localised phenomena. Examples are crack analysis and the development of shearbands. The path along which the crack or shearband develops should be as independent of the numerical approximation as possible. The amount of literature on this topic is extensive. One of the first developments in the field can be found in Belytschko *et al.* [19].

Secondly, *large deformations* can be simulated more easily since the shape functions are not defined upon a mesh or at least not upon a fixed mesh. Numerical artifacts such as element distortion or element entanglement, as occur with finite elements, are of lesser concern. Meshless methods become especially beneficial for simulations in which material points move non-uniformly with large displacements relative to neighbouring points. Examples can be found for instance in hydrodynamical problems, such as the sloshing of water or a wave rolling onto the beach. Other problems in this category are explosive problems such as bullets hitting armour, or birds impacting an airplane. For forming processes, research has been mainly restricted to bulk forming processes. Examples are extrusion, forging and upsetting. An overview of literature and accompanying references on the topic of meshless methods and large deformations will be given in Section 2.2.

Higher-order approximants can be constructed with relative ease for some meshless methods. Increasing the order of the approximation is straightforward. For instance,

the analysis of shell structures can be simulated with simpler formulations. The amount of publications on this topic is limited. One of the first developments in the field is by Krysl and Belytschko [73].

2.2 An Overview of Meshless Methods

This section gives an overview of the major developments in the field of meshless methods. Firstly a chronological overview of the major developments will be presented. Thereafter an overview on developments of meshless methods in the field of forming processes will be presented. Finally a categorial overview of meshless methods will be given. This overview will be used in the sections thereafter in order to explain the basic concepts and formulations of meshless methods.

2.2.1 A General Overview

The method of Smooth Particle Hydrodynamics (SPH) is regarded as the first meshless method. Proposed in 1977 by Lucy [91], the method was initially intended for use in astrophysical problems. However, Monaghan [92] showed in 1988 that the method can also be used for the simulation of fluids. Especially the simulation of highly transient fluid dynamical problems suits the method well. The application of the method in solid mechanics, on the other hand, is limited by two drawbacks, which are low accuracy and the lack of stability. The first drawback of the method is not easily remedied, and therefore the most simple cure is to simply use many smooth particles. On the second drawback, the instability, much research has been performed. Studies and modifications have been proposed for instance by Dyka and Ingel [47], Swegle *et al.* [117], Morris [94], Hicks *et al.* [62] and Monaghan [93]. Besides these two drawbacks, a major advantage of the method is its low computational cost.

The concept of the method of smooth particle hydrodynamics is appealing as a result of its simplicity and ability to handle large deformations easily. Over time, several methods have been proposed based on this concept, but with modifications in order to get better stability or accuracy. An example is the Reproducing Kernel Particle Method (RKPM) by Liu *et al.* [89]. Much research has been undertaken to improve this method and to apply it on the simulation of large deformations in solids, for instance by Liu *et al.* [86–88] and Aluru [6]. The method of Corrected Smoothed Particle Hydrodynamics (CSPH) was proposed by Bonet *et al.* [21, 22] and tries to solve similar deficiencies of the method of smooth particle hydrodynamics in order to get good performance in solid mechanics.

The methods that were discussed above can be regarded as 'truly' meshless methods. The methods do not use a mesh, neither for their shape functions, nor for numerical integration purposes. However, many meshless methods do not fall into this category, and use a mesh, mostly for the purpose of numerical integration. The first method in this category is the Diffuse Element Method (DEM) by Nayroles *et al.* [95]. Note that the method should not be confused with the Discrete Element Method, which has the same abbreviation. The diffuse element method uses shape functions which

are comparable to the functions as used for the method of SPH, but requires a user-defined mesh for numerical integration. The concept of the diffuse element method was modified by Belytschko *et al.* [18] and was named the element-free Galerkin method (EFG). Compared to SPH, these methods offer good accuracy and stability, but are complex and computationally expensive. Whereas the method of SPH finds its applications in fluid mechanics, the methods of DEM and EFG are more oriented to solid mechanics. The amount of publications on the element-free Galerkin method is vast. Especially the numerical integration of the equations has attracted much attention, for instance by Beissel and Belytschko [15], Dolbow and Belytschko [43], Kwon *et al.* [77], Chen *et al.* [33] and Puso *et al.* [101]. Nevertheless, accurate and cost-effective numerical integration remains an outstanding topic. Since numerical integration closely relates to volumetric locking, much research has been done on this topic, for example by Dolbow and Belytschko [44], Askes *et al.* [10], Vidal *et al.* [122], Huerta *et al.* [65] and Recio *et al.* [107, 108]. Two methods that share similarities to the methods of DEM and EFG are the Point Interpolation Method (PIM) by Liu and Gu [83] and the Meshless Local Petrov Galerkin Method (MLPG) by Atluri and Zhu [12].

Whereas the methods of DEM and EFG usually use a fixed user-defined mesh, the Natural Element Method (NEM), as proposed by Traversoni [120], uses a computer generated mesh. Based upon this mesh, natural elements are defined which share similarities with finite elements. Various developments have been made, for instance by Braun and Sambridge [25], Sukumar *et al.* [113, 114] and Cueto *et al.* [37].

The Particle Finite Element Method (PFEM), as proposed by Idelsohn *et al.* [71], can be positioned in the same category as the the natural element method. Very similarly, a computer generated mesh is used to construct shape functions. Some developments have followed thereafter, for example by Idelsohn *et al.* [69, 70]. The field of applications is similar to that of the method of smooth particle hydrodynamics.

One of the latest developments in the field of meshless methods is the maximum entropy approximation (max-ent). The first method to use a maximum entropy principle in computational continuum mechanics is the method of max-ent polygonal interpolants by Sukumar [112]. Thereafter, Arroyo and Ortiz [9] proposed the local maximum entropy (local max-ent) approximation, which offers some interesting properties for the simulation of solids. Further developments in the field have been done by Sukumar and Wright [115] and Ortiz *et al.* [98, 99], among others. A high order max-ent method for accurate simulations in solid mechanics was developed by González *et al.* [55]. The same numerical integration issues exist as for the method of EFG and DEM.

An overview of the meshless developments is presented in Table 2.1 in chronological order. The table displays the name of the method, the abbreviation of the name to which it is referred in literature, the year in which the method was proposed and the authors who proposed the method. Several meshless methods other than the methods mentioned previously have been added to the table for completeness, but will not be discussed elsewhere in this chapter.

For the interested reader, several reviews on meshless methods can be found in

literature. Reviews on SPH, DEM, RKPM and EFG which are the typical 'diffuse' meshless methods are given in Li and Liu [80], Huerta *et al.* [64], Duarte [45], Babuŝka *et al.* [13] and Belytschko *et al.* [16]. An overview of developments for NEM is given by Cueto *et al.* [37]. Books that discuss meshless methods are Liu [82], Li and Liu [81] and Zienkewicz and Taylor [131] among others.

Table 2.1: A chronological overview of major developments in the field of meshless methods.

name	abbreviation	when	references
Smooth Particle Hydrodynamics	SPH	1977	Lucy [91]
Diffuse Element Method	DEM	1992	Nayroles *et al.* [95]
Element-free Galerkin method	EFG	1994	Belytschko *et al.* [18]
Natural Element Method	NEM	1994	Traversoni [120]
Material Point Method	MPM	1994	Sulsky *et al.* [116]
Reproducing Kernel Particle Method	RKPM	1995	Liu *et al.* [89]
hp-clouds	...	1996	Duarte and Oden [46]
Meshless Local Petrov–Galerkin	MLPG	1998	Atluri and Zhu [12]
Corrected Smooth Particle Hydrodynamics	CSPH	2000	Bonet and Kulasegaram [21]
Method of Finite Spheres	...	2000	De and Bathe [38]
Finite Cloud Method	...	2001	Aluru and Li [7]
Point Interpolation Method	PIM	2001	Liu and Gu [83]
Particle Finite Element Method	PFEM	2003	Idelsohn *et al.* [71]
Least Squares Meshfree method	LSM	2003	Kwon *et al.* [75]
Maximum Entropy	max-ent	2004	Sukumar [112], Arroyo and Ortiz [9]

2.2.2 Forming Processes and Meshless Methods

One of the main motivations for using a meshless method, is that meshless methods are in general well suited to simulate large deformations. Hence, these methods have been applied to simulate forming processes and as a result the field of meshless methods in forming processes is not blank. Table 2.2 presents a short overview of various developments in this field. It must be noted however, that although developments have taken place, the application of meshless methods in practical problems as encountered in industry is limited and not comparable with the current state of the art on finite elements for instance. A method that is currently increasingly more popular is the method of SPH which appears to perform successful in the simulation of highly transient problems in fluid dynamics.

Table 2.2: An overview of literature on meshless methods in forming processes.

process	method	reference
upsetting, forging	CSPH	Bonet and Kulasegaram [21]
forging, upsetting	RKPM	Chen *et al.* [29, 31]
rolling	RKPM	Shangwu *et al.* [110]
forging, backward extrusion	RKPM	Wang *et al.* [124]
rolling, forging, backward extrusion	RKPM	Xiong *et al.* [126–128]
extrusion	RKPM	Yoon and Chen [130]
forging, backward extrusion	EFG	Li and Belytschko [79]
forging	EFG	Guedes *et al.* [58]
backward extrusion	EFG	Guo and Nakanishi [61]
upsetting	SPH/EFG	Kwon *et al.* [76, 78]
cutting, mould filling	NEM	Cueto *et al.* [37]
extrusion	NEM	Alfaro *et al.* [1, 3–5], Filice *et al.* [51]
friction stir welding	NEM	Alfaro *et al.* [2]
resin transfer moulding	NEM	García *et al.* [52]
forging, backward extrusion	PIM	Hu *et al.* [63]
extrusion	SPH/EFG	Quak *et al.* [106]

2.2.3 A Categorial Overview

As stated previously, there are a vast number of meshless methods and it can be difficult to identify the points on which the methods differ. Sometimes methods do not even differ under certain assumptions. In order to explain and present the main aspects and formulations of meshless methods, a categorisation will be made by isolating the components of which a meshless method consist. For this categorisation, a method will be subdivided in three components. *Any* of the meshless method as given previously can be decomposed in these three essential features. A short

introduction to each of these three components will be given below. Figure 2.2 gives an illustration of the categorisation.

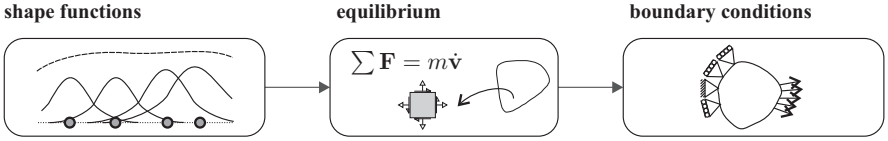


Figure 2.2: The main components of a meshless method.

The shape function is typically the most characteristic component of a meshless method. Somehow, the infinite space containing all possible solutions is decreased to a space of finite size which can be solved by a computer. Shape functions are a tool for making an approximated displacement field by using a select number of degrees of freedom, thereby creating a space of finite dimension. There are numerous ways to construct shape functions and the most interesting techniques will be given in Section 2.3.

Secondly, Newton's second law should hold in the domain. For continuum mechanics, this law results in a partial differential equation. A strategy has to be chosen in order to enforce the equation on the domain. Some commonly used techniques will be described in Section 2.4.

Finally, boundary conditions should be applied. This might appear trivial for readers familiar with finite element analysis, but for many meshless methods this is not the case. Because of the definition of the shape function, special strategies have to be adopted to enforce prescribed displacements on the boundary of the body. The three most commonly used techniques will be discussed in Section 2.5.

An overview of the methods that will be presented in the following sections is given in Figure 2.3.

2.3 Discretisation of Space

In this section, various techniques for constructing meshless shape functions will be discussed. This section is organised as follows. Firstly, in Section 2.3.1, general definitions of commonly used symbols are given. Secondly, in Section 2.3.2, the purpose of a shape function is briefly explained. Section 2.3.3 will discuss required or beneficial conditions for the construction of shape functions. In all the succeeding sections, the most commonly used techniques for constructing meshless shape functions will be presented (Section 2.3.4 to 2.3.9). For illustrative purposes, two commonly used finite element shape functions will be given in Section 2.3.10.

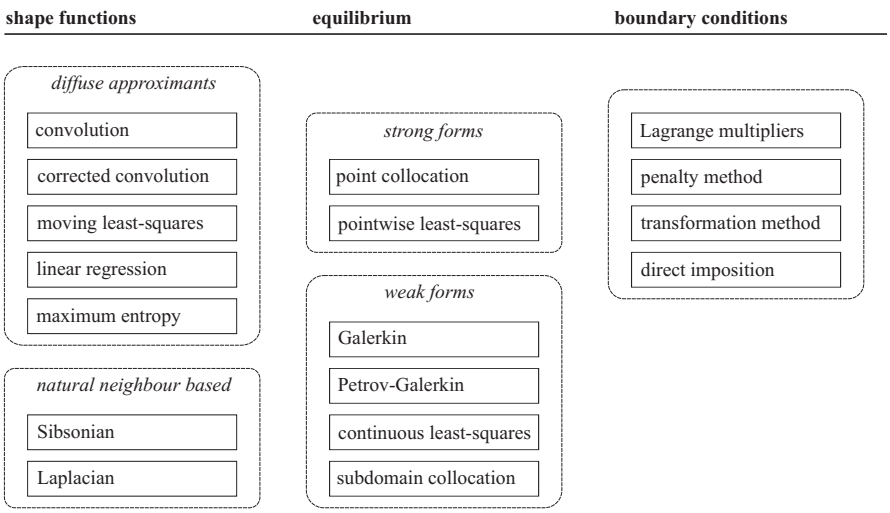


Figure 2.3: A categoric overview of meshless methods.

2.3.1 Preliminary Definitions

Figure 2.4 shows a body in space. Vectors $\mathbf{x}$ and $\boldsymbol{\xi}$ contain the locations of arbitrary points in the body. Vector $\mathbf{x}_i$ points to a node. The set Γ contains all points on the boundary. The subsets Γ_u and Γ_t contain all points out of Γ that have a prescribed displacement or a prescribed traction respectively. Vectors $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{t}}$ contain the prescribed quantities corresponding to the subsets Γ_u and Γ_t respectively. All points in the body are contained in set Ω. The outward normal on Γ is given by vector $\mathbf{n}$. Note that symbols Γ and Ω will also be used to indicate the boundaries of an integral, in the case of surface or volume integration respectively.

2.3.2 The Purpose of a Shape Function

Assume a space of points $\mathbf{x}$, an approximated displacement field $\mathbf{u}(\mathbf{x})$ and a field of displacement degrees of freedom $\mathbf{d}(\mathbf{x})$. In the case of no shape function, a set of degrees of freedom $\mathbf{d}(\mathbf{x})$ has to be computed for every point $\mathbf{x}$ of the displacement field $\mathbf{u}(\mathbf{x})$. For the complete body Ω, which contains an infinite number of points, this requires solving an infinite number of degrees of freedom, which is clearly impossible. Instead, the total field $\mathbf{u}(\mathbf{x})$ is approximated by a finite set of degrees of freedom which are multiplied by a set of yet to be defined shape functions. As a result, the degrees of freedom are not required for every point, but only for a small set of specific points, also known as nodes $\mathbf{x}_i$. The resulting system is of finite size and can be solved

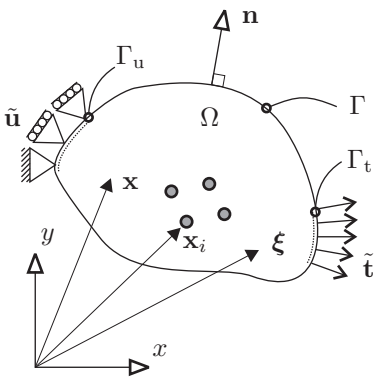


Figure 2.4: Some preliminary definitions on the used symbols.

by a computer. The approximated displacement field is defined as:

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x})\mathbf{d}_i \tag{2.1}$$

where the total number of nodes in the domain is N_{nod} and the shape function of node i is given by $\phi_i(\mathbf{x})$. The displacement degrees of freedom at point $\mathbf{x}_i$, are contained in vector $\mathbf{d}_i$. This vector is defined in 2D as follows:

$$\mathbf{d}_i = \left\{ \begin{array}{cc} d_x^i & d_y^i \end{array} \right\}^{\mathrm{T}} \tag{2.2}$$

where subscripts x and y denote the directions. Note that in some of the following derivations, the degrees of freedom are expressed as a function of the coordinates. This is denoted by vector $\mathbf{d}(\mathbf{x})$, which contains the degrees of freedom of an arbitrary point $\mathbf{x}$. Similarly as in Equation (2.2), vector $\mathbf{d}(\mathbf{x})$ is defined as:

$$\mathbf{d}(\mathbf{x}) = \left\{ \begin{array}{cc} d_x(\mathbf{x}) & d_y(\mathbf{x}) \end{array} \right\}^{\mathrm{T}} \tag{2.3}$$

2.3.3 Properties of Shape Functions

Unfortunately, not all shape functions $\boldsymbol{\phi}(\mathbf{x})$ that one can think of are of good use for solving a partial differential equation. There are properties, either compulsory or beneficial, which are of concern when defining a shape function. The first two conditions that will be discussed below are obligatory. The third condition is not obligatory, though it will be shown that meeting this condition simplifies the resulting method considerably.

Continuity

The term continuity is used to express the smoothness of a function. If the derivatives to the n-th order of a function ϕ are continuous, the function possesses C^n continuity. In formula form this is stated as:

$$\text{if} \quad \lim_{\mathbf{x}\to\boldsymbol{\xi}} \frac{\partial^n \phi(\mathbf{x})}{\partial \mathbf{x}^n} = \frac{\partial^n \phi(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}^n} \qquad \forall\, \boldsymbol{\xi} \in \Omega \quad \text{then} \quad \phi(\mathbf{x}) \text{ is } C^n \tag{2.4}$$

The derivatives to the n-th order of function ϕ should be single valued for all points $\boldsymbol{\xi}$ in the domain Ω. For instance: the first derivative of a C^1 function is continuous for all points in the domain. The second derivative of this function is not continuous. An illustration of the concept of continuity is given in Figure 2.5. Demands on the order of continuity of a shape function arise from the method that is used to discretise the partial differential equation. See Section 2.4 for these methods.

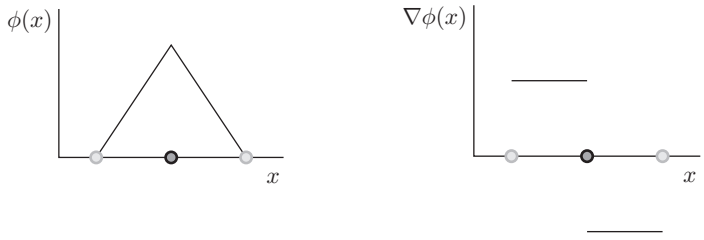


Figure 2.5: An illustration of the order of continuity (function $\phi(x)$ is C^0).

Reproducibility

The reproducibility of an approximated displacement field is the ability of this field to reproduce polynomials. For problems in computational solid mechanics, the conditions of zeroth and first order reproducibility are usually of concern. These conditions are defined as follows:

$$\sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}) = 1 \qquad \forall\, \mathbf{x} \in \Omega \tag{2.5}$$

$$\sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x})\mathbf{x}_i = \mathbf{x} \qquad \forall\, \mathbf{x} \in \Omega \tag{2.6}$$

Equation (2.5) is the zeroth order reproducibility condition. If this condition is satisfied, rigid body modes of the domain can be simulated without introducing strain in the body. Note that this condition is also known as the partition of unity. The condition of first order reproducibility is given in Equation (2.6). The shape functions should reproduce a linear polynomial field exactly if the condition is to hold. The

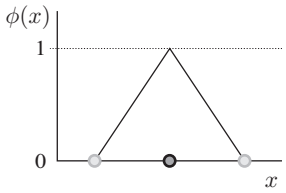
patch test is a commonly used test to examine this condition. In general, problems can be expected if Equations (2.5) and (2.6) are not satisfied. Note that Equations (2.5) and (2.6) are also referred to as the consistency conditions or the completeness conditions.

Kronecker delta Property

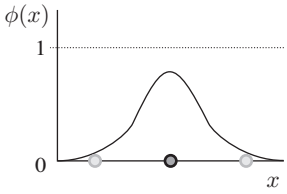
The Kronecker delta property is especially of interest when displacement boundary conditions need to be prescribed. The condition is defined as:

$$\phi_i(\mathbf{x}_j) = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i \neq j \end{cases} \tag{2.7}$$

The value of a shape function belonging to node i is 1 at the position of this node and is zero on all other nodes. As a result, the Kronecker delta property forces the shape functions to have an interpolative, local character. Shape functions that do not meet this requirement will be referred to as diffuse, non-local shape functions. These functions overlap neighbouring nodes, thus invalidating the Kronecker delta property. If the Kronecker delta property is satisfied, as is the case for interpolants, prescribed boundary displacements can be easily applied. For diffuse approximations this is not the case and a special method has to be adopted to enforce these prescribed displacements. Section 2.5 will explain this topic in more detail.



(a) a function satisfying Equation (2.7)



(b) a function not satisfying Equation (2.7)

Figure 2.6: An illustration of the Kronecker delta property.

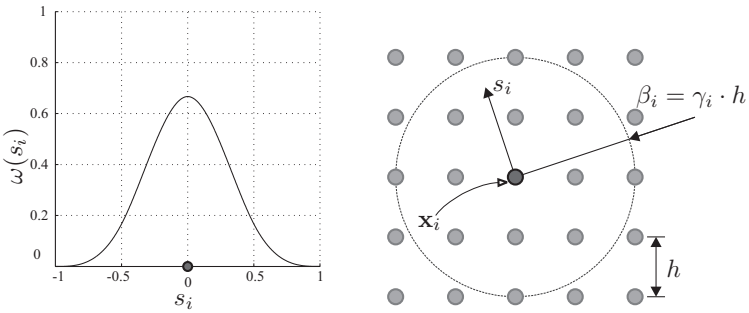
2.3.4 Convolution

The convolution technique was introduced in the first meshless method; namely the method of smooth particle hydrodynamics by Lucy [91]. The convolution integral constructs a smooth approximating displacement field $\mathbf{u}(\mathbf{x})$ by multiplying a window function with the displacement degrees of freedom and integrating the result over the domain. This is stated in formula form as follows:

$$\mathbf{u}(\mathbf{x}) = \int_{\Omega} \omega\left(\mathbf{x} - \boldsymbol{\xi}, \gamma\right) \mathbf{d}(\boldsymbol{\xi}) \, \mathrm{d}\Omega \tag{2.8}$$

where $\boldsymbol{\xi}$ is a coordinate over which the integration is carried out, Ω is the domain, γ is a parameter that controls the domain of influence or footprint of the kernel function ω and $\mathbf{d}(\boldsymbol{\xi})$ contains the degrees of freedom at point $\boldsymbol{\xi}$.

Equation (2.8) states that the displacement at a certain *fixed* point $\mathbf{x}$, is assumed to be a function of the displacement degrees of freedom of the surrounding points $\boldsymbol{\xi}$. As a result of the kernel function, the approximated displacement field $\mathbf{u}$ is a smooth version of the field of degrees of freedom $\mathbf{d}$. As an illustration: by choosing a Dirac delta function as kernel function ω, there is no smoothing effect of the convolution integral, and therefore $\mathbf{d}(\mathbf{x})$ is equal to $\mathbf{u}(\mathbf{x})$. Note that ω is also referred to in literature as window, weigh or weighting function. Appendix B gives some commonly used kernel functions as found in literature.



(a) a cubic spline kernel function (b) a regular grid of nodes

Figure 2.7: An illustration of kernel function ω generated on a grid of nodes.

In Equation (2.8), vector $\mathbf{d}(\boldsymbol{\xi})$ contains the degrees of freedom which are required at each point $\boldsymbol{\xi}$. In order to have a set of degrees of freedom which is not of infinite size, only at the nodal positions $\mathbf{x}_i$, $\mathbf{d}$ will be defined. Equation (2.8) is approximated as follows:

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \omega\left(\mathbf{x} - \mathbf{x}_i, \gamma_i\right) \mathbf{d}_i \Omega_i \tag{2.9}$$

where γ_i determines the domain of influence for kernel function $\omega_i(\mathbf{x})$, N_{nod} is the number of nodes in the domain and Ω_i is a volume associated with node i. The shape function $\phi_i(x)$, as defined in Equation (2.1) can now be expressed as:

$$\phi_i(\mathbf{x}) = \omega\left(\mathbf{x} - \boldsymbol{\xi}_i, \gamma_i\right) \Omega_i \tag{2.10}$$

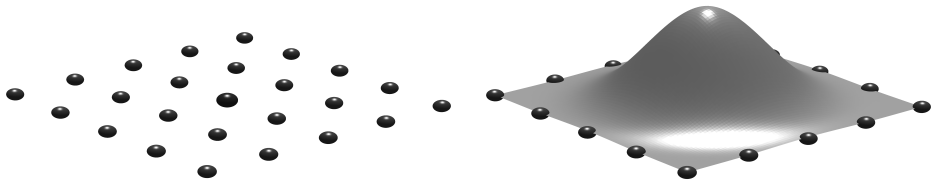
The kernel function ω is generated on a dimensionless coordinate s_i as follows:

$$\omega\left(\mathbf{x} - \boldsymbol{\xi}_i, \gamma_i\right) = w(s_i) \tag{2.11}$$

$$\text{where} \quad s_i = \frac{\|\mathbf{x} - \boldsymbol{\xi}_i\|}{\beta_i} = \frac{\|\mathbf{x} - \boldsymbol{\xi}_i\|}{\gamma_i \cdot h} \tag{2.12}$$

where parameter β_i is an absolute measure of the footprint or domain of influence of the kernel function of node i. It is a product of the user-set parameter γ_i and the nodal spacing h such that an absolute value for the size of the footprint is obtained. Figure 2.7 gives an example of a cubic spline kernel function being generated on a regular grid of nodes. Figure 2.8 shows the shape function made with that same spline function in 2D. The shape function of the node in the centre of the grid is plotted. Note that the order of continuity of the shape function is equal to the order of continuity of the used kernel function ω.

A benefit of shape functions constructed with the convolution integral is their low computational cost. However, there are drawbacks to these functions, which is why they are less preferred in continuum mechanics. Firstly, the zeroth and first order reproducibility condition are usually not met, especially at the boundary of the domain. The patch test is therefore not satisfied and strain can be predicted in rigid body modes. Secondly, the Kronecker delta property is not met, which complicates the prescription of displacement degrees of freedom at the boundary. Section 2.5 explains the consequence of not meeting the Kronecker-Delta property and gives methods that are used to enforce boundary conditions for this case.



(a) a regular grid of nodes (b) a convoluted spline

Figure 2.8: A convolution shape function.

2.3.5 Corrected Convolution

Improvements of the convolution integral have been proposed in order to overcome the lack of reproducibility. The main point of these improvements is to introduce a function C which is defined such that constant and linear polynomials are reproduced. In formula form:

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} C(\mathbf{x}, \mathbf{x}_i)\omega\left(\mathbf{x} - \mathbf{x}_i, \gamma_i\right) \Omega_i \mathbf{d}_i \tag{2.13}$$

where $C(\mathbf{x}, \mathbf{x}_i)$ is a function that will be defined such that the reproducibility conditions are met. The resulting shape function is:

$$\phi_i(\mathbf{x}) = C(\mathbf{x}, \mathbf{x}_i)\omega(\mathbf{x} - \mathbf{x}_i, \gamma_i)\Omega_i \tag{2.14}$$

where coefficient $C(\mathbf{x}, \mathbf{x}_i)$ is computed by considering:

$$\begin{aligned} &\text{for fixed } \mathbf{x} \text{ find } C(\mathbf{x}, \mathbf{x}_i) \\ &\text{such that } \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}) = 1 \\ &\text{and } \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}) \mathbf{x}_i = \mathbf{x} \end{aligned} \tag{2.15}$$

Shape functions of this type are found for instance in the method of corrected smooth particle hydrodynamics by Bonet and Kulasegaram [21] or the reproducing kernel particle method by Liu *et al.* [89]. Computational effort is required to evaluate $C(\mathbf{x}, \mathbf{x}_i)$. Although the modified function reproduces constant and linear polynomials, the Kronecker delta property is not met.

2.3.6 Moving Least Squares

Moving least squares approximations were firstly introduced in the field of computational solid mechanics by means of the diffuse element method by Nayroles *et al.* [95]. Since then, the strategy has been adopted in the element-free Galerkin method by Belytschko *et al.* [18] and the meshless local Petrov–Galerkin method by Atluri and Zhu [12]. On some occasions, the MLS shape functions are also found in the reproducing kernel particle method, the natural element method or the method of smooth particle hydrodynamics.

The starting point of the moving least squares approximation is the assumption that the displacement field can be described locally by a polynomial multiplied by a set of coefficients:

$$u(\mathbf{x}, \boldsymbol{\xi}) = \mathbf{p}(\boldsymbol{\xi})^{\mathrm{T}} \mathbf{a}(\mathbf{x}) \tag{2.16}$$

where $\mathbf{p}(\boldsymbol{\xi})$ is a vector containing the components of a polynomial basis at the point $\boldsymbol{\xi}$ and $\mathbf{a}(\mathbf{x})$ is the corresponding set of coefficients at point $\mathbf{x}$. Equation (2.16) states that the displacement at a *fixed* point $\mathbf{x}$ is determined by a polynomial basis in $\boldsymbol{\xi}$ times a set of parameters which are defined for that point $\mathbf{x}$. Polynomials of required order can be included in vector $\mathbf{p}(\boldsymbol{\xi})$. An example of a simple linear polynomial field and its coefficients in 2D are:

$$\mathbf{p}(\boldsymbol{\xi}) = \left\{ \begin{array}{ccc} 1 & \xi & \eta \end{array} \right\}^{\mathrm{T}} \tag{2.17}$$

$$\mathbf{a}(\mathbf{x}) = \left\{ \begin{array}{ccc} a_o(\mathbf{x}) & a_1(\mathbf{x}) & a_2(\mathbf{x}) \end{array} \right\}^{\mathrm{T}} \tag{2.18}$$

The parameters $\mathbf{a}(\mathbf{x})$ belonging to the polynomial basis $\mathbf{p}(\boldsymbol{\xi})$ are found by minimising a potential expressing the residual between the approximated displacement field and the nodal displacements $\mathbf{d}_i$, similarly to a 'normal' least squares fit:

$$\Pi_{\text{mls}}(\mathbf{x}) = \int_{\Omega} \omega(\mathbf{x} - \boldsymbol{\xi}, \gamma) \left(d_x(\boldsymbol{\xi}) - \mathbf{p}(\boldsymbol{\xi})^{\mathrm{T}} \mathbf{a}(\mathbf{x}) \right)^2 \mathrm{d}\Omega \tag{2.19}$$

where ω is a similar kernel function as used in Section 2.3.4. Appendix B gives several example kernel functions.

There are two significant differences that make Equation (2.19) not a standard least squares fit. Firstly, kernel function ω varies throughout the domain. Secondly, coefficients $\mathbf{a}(\mathbf{x})$ are also allowed to vary throughout the domain, hence the name 'moving' in moving least squares. In order to compute the polynomial coefficients, one has to specify the coordinates at which these coefficients should be obtained. Now, instead of having an infinite set of degrees of freedom $\mathbf{d}$ for all points $\boldsymbol{\xi}$, the set is limited to only the nodal degrees of freedom $\mathbf{d}_i$:

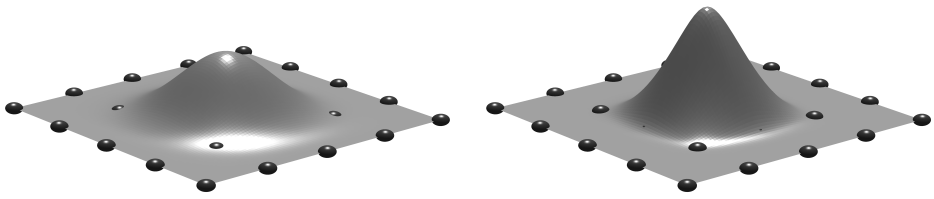
$$\Pi_{\text{mls}}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \omega(\mathbf{x} - \mathbf{x}_i, \gamma_i) \left(d_x^i - \mathbf{p}(\mathbf{x}_i)^{\text{T}} \mathbf{a}(\mathbf{x})\right)^2 \Omega_i \tag{2.20}$$

where Ω_i is usually omitted. In order to obtain coefficients $\mathbf{a}(\mathbf{x})$, the potential $\Pi_{\text{mls}}(\mathbf{x})$ is minimised with respect to parameters $\mathbf{a}(\mathbf{x})$:

$$\text{for fixed } \mathbf{x} \text{ find } \min_{\mathbf{a}(\mathbf{x})} \Pi_{\text{mls}} \tag{2.21}$$

Substitution of the coefficients $\mathbf{a}(\mathbf{x})$ for which the potential Π_{mls} is at its minimum into Equation (2.16) gives an expression for $\phi(\mathbf{x})$. The minimisation of Equation (2.21) must be performed for location $\mathbf{x}$ at which the shape function values or gradients are required.

The main benefit of shape functions constructed with the MLS technique is that continuity and reproducibility can be of arbitrary order. The order of continuity depends on the kernel function ω. If ω has for instance a continuity of C^2, ϕ has a continuity of C^2. The reproducibility of the functions depends on the order of the polynomial basis $\mathbf{p}(\boldsymbol{\xi})$. Figure 2.9 shows a MLS shape function for two settings of γ. Using a high γ results in a diffuse approximation and setting γ low drives the MLS function towards interpolation. A drawback of the shape functions is that the Kronecker delta property is not satisfied. A second drawback is that the minimisation of Equation (2.21) is computationally demanding.



(a) $\gamma = 2.0$ (b) $\gamma = 1.5$

Figure 2.9: A MLS shape function for various settings of γ.

2.3.7 Linear Regression

Linear regression is a technique, that, similarly as for the convolution integral or the least squares method, can be used to fit a field through a set of nodal values. The point interpolation method as developed by Liu and Gu [83] uses this linear regression to construct the shape functions. The starting formulation is very similar to that of the method of moving least squares.

Firstly, a potential is constructed which describes the error between the field of degrees of freedom $\mathbf{d}(\boldsymbol{\xi})$ and a polynomial space:

$$\Pi_{\text{reg}}(\mathbf{x}, \boldsymbol{\xi}) = \omega(\mathbf{x} - \boldsymbol{\xi}, \gamma) \left(d_x(\boldsymbol{\xi}) - \mathbf{p}(\boldsymbol{\xi})^{\text{T}} \mathbf{a}(\mathbf{x}) \right) \tag{2.22}$$

where $\mathbf{p}$ and $\mathbf{a}$ are given in Equations (2.17) and (2.18) respectively. Kernel function ω is defined as follows:

$$\omega(\mathbf{x} - \boldsymbol{\xi}, \gamma) = \begin{cases} 1 & \text{for} \quad \|\mathbf{x} - \boldsymbol{\xi}\| \leq \gamma \\ 0 & \text{for} \quad \|\mathbf{x} - \boldsymbol{\xi}\| > \gamma \end{cases} \tag{2.23}$$

where γ is a parameter controlling the number of points $\boldsymbol{\xi}$ in the neighbourhood of $\mathbf{x}$ that contribute to the potential Π_{reg} for a fixed point $\mathbf{x}$. The estimated displacement at a point $\mathbf{x}$ is determined by considering Equation (2.22) only at the nodal locations:

$$\Pi_{\text{reg}}(\mathbf{x}, \mathbf{x}_i) = \omega(\mathbf{x} - \mathbf{x}_i, \gamma_i) \left(d_x^i - \mathbf{p}(\mathbf{x}_i)^{\text{T}} \mathbf{a}(\mathbf{x}) \right) \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.24}$$

A set of parameters $\mathbf{a}(\mathbf{x})$ is sought for which Π_{reg} is zero:

$$\begin{aligned} &\text{for fixed } \mathbf{x} \text{ find } \mathbf{a}(\mathbf{x}) \\ &\qquad \text{subject to } \Pi_{\text{reg}}(\mathbf{x}, \mathbf{x}_i) = 0 \end{aligned} \tag{2.25}$$

Shape functions made with a linear regression satisfy the Kronecker delta property and the reproducibility conditions. A major drawback, however, is that a unique set of shape functions is not always obtained. An enhanced version of the linear regression was proposed in the method of Radial Point Interpolation Method (RPIM) by Wang and Liu [125]. These improved shape functions do not suffer from the non-uniqueness issue. Figure 2.10 shows this shape function.

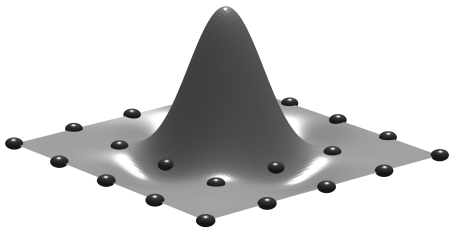


Figure 2.10: A shape function made with a linear regression.

2.3.8 Local Maximum Entropy

The local maximum entropy approximation as proposed by Arroyo and Ortiz [9] is derived differently than the shape functions described previously. Instead of for instance least squares fitting, information theoretic principles are used to construct the shape functions. The main point of the method is the assumption that there is a variable ϕ, which expresses the probability of a nodal value holding at any other arbitrary point in space. This probability distribution, which has yet to be defined, is used to approximate the displacement field. A discrete potential containing both an entropy term related to the probability and a potential expressing the locality of the probability distribution is constructed. This potential is formulated as:

$$\Pi_{\text{lme}} = \beta U(\mathbf{x}, \boldsymbol{\phi}) - H(\boldsymbol{\phi}) \tag{2.26}$$

with the Shannon entropy H defined as:

$$H(\boldsymbol{\phi}) = \sum_{i=1}^{N_{\text{nod}}} \phi_i \ln(\phi_i) \tag{2.27}$$

and the locality function:

$$U(\mathbf{x}, \boldsymbol{\phi}) = \sum_{i=1}^{N_{\text{nod}}} \phi_i \cdot \|\mathbf{x} - \mathbf{x}_i\|^2 \tag{2.28}$$

In the potential Π_{lme}, the parameter β is used to control the compactness of the approximation. This parameter is related to the average spacing of nodes h and the domain of influence parameter μ as follows:

$$\beta = \frac{\mu}{h^2} \tag{2.29}$$

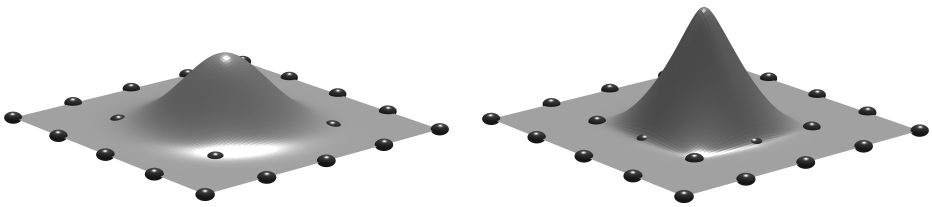
By setting μ high or low, either compact or diffuse shape functions respectively can be obtained. The probability distribution $\boldsymbol{\phi}$ or, similarly, the shape functions are found by minimising Equation (2.26):

$$\begin{aligned} &\text{for fixed } \mathbf{x} \text{ find } \min_{\phi_i(\mathbf{x})} \Pi_{\text{lme}} \\ &\text{subject to } \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}) = 1 \\ &\qquad\qquad \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}) \mathbf{x}_i = \mathbf{x} \end{aligned} \tag{2.30}$$

This minimisation is constrained by Equations (2.5) and (2.6) such that the zeroth and first order reproducibility conditions are satisfied. For a detailed description of the minimisation the reader is referred to Arroyo and Ortiz [9]. Figure 2.11 shows a local maximum entropy shape function for two settings of μ. If μ is moved towards infinity, the local maximum entropy shape function are identical to linear interpolation upon

a Delaunay triangulation. Note that for degenerate cases, for instance if four nodes are positioned in a perfect square, the triangle interpolation and the local maximum entropy functions are not equal. For a low setting of μ, the local maximum entropy is very similar to the moving least squares function.

An interesting aspect of local maximum entropy shape functions is their behaviour at the boundary. The convex hull of a cloud of nodes consists of all nodes which lie on a convex polygon enclosing the body Ω. Shape functions of nodes on this convex hull possess the Kronecker delta property. Shape functions of internal nodes have a value of zero at the convex hull of the domain. Nevertheless, this local behaviour at the boundary does not prevent the shape functions from being diffuse internally. Moreover, the shape functions are C^∞ continuous on points in the domain.



(a) $\mu = 1.5$ (b) $\mu = 3.0$

Figure 2.11: A LME shape function for various settings of μ.

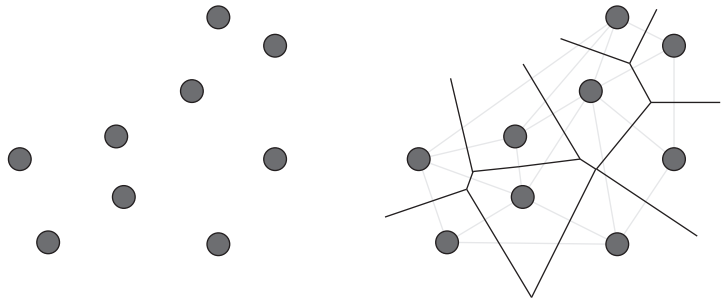
2.3.9 Natural-Neighbour Interpolants

The first meshless method using the natural-neighbour interpolation is the natural element method as proposed by Traversoni [120]. A more recent method that uses the natural-neighbour interpolation is the Particle Finite Element method (PFEM) method by Idelsohn *et al.* [71]. Constructing shape functions by using the natural-neighbour method is a different approach when compared to the methods presented before. Instead of diffuse shape functions, the natural-neighbour approximations are local and are defined upon a 'mesh'. The main idea is to make a Voronoi tessellation of a cloud of nodes and to use that tessellation for constructing the shape functions. This Voronoi tessellation can be computed uniquely for any arbitrary cloud of nodes. Hence there is no need for user involvement to build this mesh and there are no requirements regarding the location of the nodes.

The definition of a Voronoi cell is as follows. A Voronoi cell of node j is a set of points consisting of all points that lie closer to this node than to any other node. In mathematical formulation for 2D, this can be stated as:

$$V_j = \left\{ \mathbf{x} \in \mathbb{R}^2 \mid d(\mathbf{x}, \mathbf{x}_j) < d(\mathbf{x}, \mathbf{x}_i) \; \forall \; i \neq j \right\} \tag{2.31}$$

where $d(\mathbf{x}, \mathbf{x}_i)$ is the Euclidean distance between point $\mathbf{x}$ and node $\mathbf{x}_i$. Figure 2.12 displays a Voronoi tessellation of a cloud of nodes. Figure 2.12(a) shows the cloud of nodes. Figure 2.12(b) shows the tessellation.



(a) cloud of nodes (b) Voronoi tessellation

Figure 2.12: A Voronoi tessellation of a cloud of nodes.

Based on the Voronoi tessellation, various shape functions can be constructed. Two commonly used functions are the Sibson and the Laplace interpolation. Figure 2.13 gives an example of a Sibson shape function based on the Voronoi diagram of a cloud of nodes.

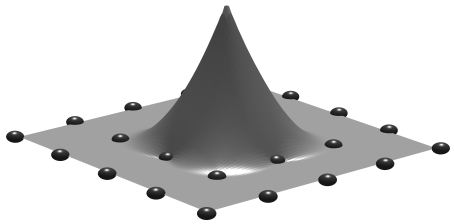


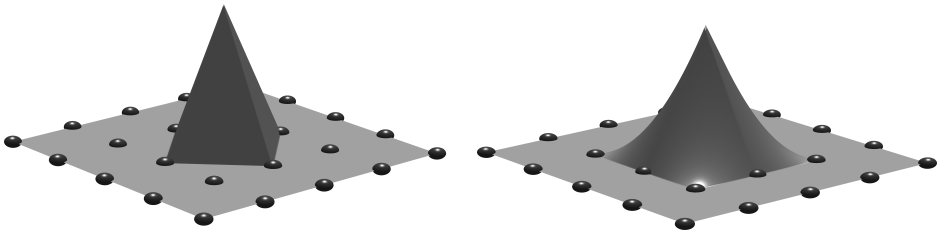
Figure 2.13: A shape function made with a Sibson interpolation.

The most interesting property of the natural-neighbour interpolation is that the Kronecker delta property is satisfied such that boundary conditions can be prescribed easily. Furthermore, constant and linear polynomials are reproduced by the functions.

2.3.10 FEM interpolants

Formulations for finite element interpolations can be found in most books on finite element analysis, for instance in Zienkiewicz and Taylor [131] among others. Whereas the shape functions as discussed above depend on the positions of the nodes, finite elements shape functions depend mainly on the mesh as supplied by the user. The benefit of finite element shape functions is their low computational cost. A drawback is that the functions become inaccurate and can even get singular if the mesh gets distorted. Figures 2.14(a) and 2.14(b) show a linear triangle interpolation and a linear

quadrilateral interpolation respectively. FEM interpolations satisfy the Kronecker delta property and reproduce constant and linear polynomials.



(a) linear triangle interpolation (b) quadrilateral interpolation

Figure 2.14: Two finite element shape functions. To generate these functions, a regular mesh based on the nodal grid was used.

2.4 Discretisation of Equilibrium

There are various ways to discretise a partial differential equation in order to get a set of solvable algebraic equations. The methods as introduced in the following sections will be presented as weighted residual methods. The main point of a weighted residual method is to force the unbalance in the equilibrium to zero in an averaged sense. For more detailed information on this topic, the reader is referred to Zienkiewicz and Taylor [132] or Bathe [14]. Note that the quasi-static equilibrium equations will be used in this section. Acceleration terms will be omitted.

The equilibrium equation which should hold for all points in the body is stated by the following differential equation:

$$\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f} = \mathbf{0} \qquad \forall\, \mathbf{x} \in \Omega \tag{2.32}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\overleftarrow{\nabla}$ is a vector containing the spatial gradient operator and $\mathbf{f}$ is a body force. Note that in this section only the discretisation of the equilibrium equation will be discussed. The boundary conditions to which Equation (2.32) is subjected will be presented in Section 2.5.

Equation (2.32) represents equilibrium of forces in strong form. By multiplying this strong form by a virtual displacement $\delta\mathbf{u}$ and integrating the product over the domain, the weighted equilibrium equation is obtained:

$$\int_{\Omega} \delta\mathbf{u} \cdot \left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f} \right) \mathrm{d}\Omega = 0 \qquad \forall\, \delta\mathbf{u} \tag{2.33}$$

where $\delta\mathbf{u}$ is the virtual displacement field. Note that Equation (2.32) and Equation (2.33) are fully equivalent. If Equation (2.33) is satisfied for any arbitrary function $\delta\mathbf{u}$,

then equilibrium holds for all points $\mathbf{x}$. Conversely, if all points $\mathbf{x}$ are in equilibrium, Equation (2.32) will be satisfied no matter what function $\delta\mathbf{u}$ is used.

The formula as given in Equation (2.33) is known as the weighted residual. Similarly as for the displacements, the space of virtual displacements is described by a set of shape functions and nodal values:

$$\delta\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \psi_i(\mathbf{x})\delta\mathbf{d}_i \tag{2.34}$$

where $\psi_i(\mathbf{x})$ are the test functions and $\delta\mathbf{d}_i$ are the virtual nodal degrees of freedom. Filling the test functions into the weighted residual equation (2.33) gives:

$$\int_\Omega \sum_{i=1}^{N_{\text{nod}}} \psi_i(\mathbf{x})\delta\mathbf{d}_i \cdot \left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f}\right) \mathrm{d}\Omega = 0 \qquad \forall\, \delta\mathbf{d}_i \tag{2.35}$$

The question remains what test functions $\psi_i(\mathbf{x})$ should be chosen. The remainder of this section presents several options for defining these functions. The methods that are commonly found in literature on meshless methods are given. Firstly the most commonly used discretisations of equilibrium are presented in Sections 2.4.1 and 2.4.2. Discretisation schemes which are found less frequently in the field of meshless methods are shortly discussed for completeness thereafter.

2.4.1 Point Collocation

In a point collocation method, the test function ψ_i is defined as the Dirac delta function in order to force the weighted residual to zero at nodal points. In formula form, ψ_i is defined as:

$$\psi_i(\mathbf{x}) = \begin{cases} \infty & \text{if} \quad \mathbf{x} = \mathbf{x}_i \\ 0 & \text{for all other } \mathbf{x} \end{cases} \tag{2.36}$$
$$\text{and:} \quad \int_\Omega \psi_i(\mathbf{x})\,\mathrm{d}\Omega = 1$$

At a node $\mathbf{x}_i$, function $\psi_i(\mathbf{x})$ goes to infinity and is zero otherwise. After some manipulation, the weighted residual of Equation (2.35) results in the following equation:

$$\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f} = \mathbf{0} \qquad \forall\, \mathbf{x}_i \in \Omega \tag{2.37}$$

As a result, the strong form of the equilibrium equation is required to be zero only at nodes $\mathbf{x}_i$. Two main points should be considered when using a discretisation of this type. Firstly, a system of equations resulting from the test functions as defined in Equation (2.36) is likely to allow spurious patterns in the displacement field $\mathbf{u}(\mathbf{x})$. Stabilisation procedures have to be used in order to suppress this instability. Secondly, since a gradient of stress appears in Equation (2.37), a second order gradient is

required of the displacement field. Hence shape functions $\phi(\mathbf{x})$ should have continuity up to second order (C^2). For very similar reasons, discretising the equilibrium equation with a collocation method poses difficulties for finding a description which allows a general use of material models. Usually, material models relate the strain state to the stress state and do not consider the gradient of the stress state. Having for instance an elasto-plastic algorithm in a strong form is challenging and does not seem to be available in literature. The collocation method is mainly found in the methods of SPH and RKPM.

2.4.2 Galerkin

In a Galerkin procedure, weighing the weak equilibrium is done by using the same shape functions as used to parameterise the displacement field. The space of test functions is equal to the space of trial functions:

$$\psi_i(\mathbf{x}) = \phi_i(\mathbf{x}) \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.38}$$

The weighted residual equation is as follows:

$$\int_\Omega \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}) \delta\mathbf{d}_i \cdot \left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f} \right) \mathrm{d}\Omega = 0 \qquad \forall\, \delta\mathbf{d}_i \tag{2.39}$$

This integral is simplified further by using integration by parts, such that the gradient operator on the stress tensor is eliminated. For the implementation in a computer code, the integral resulting from the integration by parts should be evaluated numerically. Unfortunately, the non-polynomial character of most meshless approximations makes this numerical integration far less straightforward when compared to finite element approximations. The patch test, for example, is not satisfied with a limited set of integration points for a non-polynomial function. Since the introduction of meshless methods using a Galerkin weak form, this topic has drawn considerable attention.

Although the numerical integration of Equation (2.39) is complex, there are many benefits of using the Galerkin weak form. Firstly, system matrices resulting from the Galerkin weak form are symmetric, which saves computational time and storing capacity. Secondly, material models are easily incorporated in a Galerkin scheme. There is no gradient of the stress appearing in the equilibrium equation as is the case for strong form methods. Continuity requirements regarding the shape functions are in general of order C^1, but can be loosened for finite elements to piecewise C^0 continuity. Finally, stability is guaranteed in case of sufficiently independent shape functions and full integration. These benefits lead to the fact that the Galerkin weak form is used in the majority of meshless methods. Examples are the methods of DEM, MPM, EFG, RKPM, NEM, PIM and max-ent.

2.4.3 Petrov–Galerkin

The Petrov–Galerkin method is very similar to the Galerkin method. The main difference between is that for the Petrov–Galerkin method the test functions differ

from the trial functions:

$$\psi_i(\mathbf{x}) \neq \phi_i(\mathbf{x}) \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.40}$$

In contrast to the test functions as used for the collocation method, the test functions of the Petrov–Galerkin method possess sufficient continuity in order to perform integration by parts. The Petrov–Galerkin method has in general less advantageous properties than the Galerkin method. The main drawback of the method is that the resulting system matrices are asymmetric. This will require more computational effort and storage. A numerical integration scheme has to be applied on the weak form. The MLPG method uses the Petrov–Galerkin method to discretise the equilibrium. The same numerical integration difficulties are of concern as are the case for the Galerkin method in Section 2.4.2.

2.4.4 Continuous Least Squares

For the method of continuous least squares, the discretisation error as given in Equation (2.32) is squared, integrated over the domain and minimised to the degrees of freedom. In formula form:

$$\frac{\partial}{\partial \mathbf{d}_i} \int_\Omega \left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f}\right) \cdot \left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f}\right) \, \mathrm{d}\Omega = \mathbf{0} \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.41}$$

2.4.5 Subdomain Collocation

The subdomain collocation procedure demands the residual to be zero not for points but for subdomains. Each node $\mathbf{x}_i$ has a corresponding piece of the domain Ω, noted as Ω_i. An option for instance is to use a Voronoi cell V_i for Ω_i. For each subdomain the following shape functions is defined:

$$\psi_i(\mathbf{x}) = \begin{cases} 1 & \forall\, \mathbf{x} \in \Omega_i \\ 0 & \text{for all other } \mathbf{x} \end{cases} \tag{2.42}$$

Equation (2.35) becomes:

$$\delta \mathbf{d}_i \cdot \int_{\Omega_i} \left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f}\right) \, \mathrm{d}\Omega = 0 \qquad \forall\, \delta \mathbf{d}_i \tag{2.43}$$

2.4.6 Point-wise Least Squares

The formulation of the method of point least squares is comparable to its continuous counterpart as presented previously. The main difference is that the error of the partial differential equation is squared only at the nodes instead of squaring it for every point over the total domain. The method is also known as least squares collocation or over-determined collocation. In formula form the point-wise least squares discretisation of the equilibrium gives:

$$\left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f}\right) \cdot \left(\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f}\right) = 0 \qquad \forall\, \mathbf{x}_i \in \Omega \tag{2.44}$$

No integration is needed. Similar issues are of concern as given in Subsection 2.4.1 for the point collocation method.

2.5 Applying Boundary Conditions

This section discusses methods that are used to impose boundary conditions for diffuse approximations. There is a considerable amount of research on this topic, and many methods have been proposed. Research on the application of boundary conditions in meshless methods can be found for instance in Gosz and Liu [57], Günther and Liu [60], Chen and Wang [32], Pannachet and Askes [100], Fernández-Méndez and Huerta [50] and Guedes and César de Sá [59].

The two necessary boundary conditions for a body in solid mechanics are stated as:

$$\mathbf{u}(\mathbf{x}) - \tilde{\mathbf{u}}(\mathbf{x}) = \mathbf{0} \qquad \forall\, \mathbf{x} \in \Gamma_{\mathrm{u}} \tag{2.45}$$

$$\boldsymbol{\sigma} \cdot \mathbf{n} - \tilde{\mathbf{t}} = \mathbf{0} \qquad \forall\, \mathbf{x} \in \Gamma_{\mathrm{t}} \tag{2.46}$$

where $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{t}}$ are the prescribed displacements and tractions respectively. See Figure 2.4 for an explanation of the used symbols. For the remainder of this section, only the first boundary condition as given in Equation (2.45) will be discussed. The second condition, as shown in Equation (2.46), is in most cases straightforward to apply. A prescribed traction is not an implicit function of the shape functions whereas a prescribed displacement depends implicitly on the used shape function. The succeeding sections will explain this matter in more detail.

The ease with which prescribed displacement boundary conditions can be enforced depends on the type of shape function chosen. Shape functions which satisfy the Kronecker delta property, also referred to as local shape functions or interpolants, allow for a simple and straightforward application of the boundary conditions. The reason for this simple handling of displacement boundary conditions will be explained in more detail below. Thereafter, the same derivation is made, but then for a shape function which does not possess the Kronecker delta property. The implications of having such a diffuse shape function on the application of displacement boundary conditions will be explained. The remainder of this section discusses three methods that are commonly used to prescribe boundary conditions for these diffuse meshless shape functions (Section 2.5.1, 2.5.2 and 2.5.3).

Local Approximants

Displacement boundary conditions are easily prescribed in the case of a method using local shape functions. Examples of such shape functions are finite element interpolants or natural-neighbour interpolants. These functions satisfy the Knonecker-delta property as given in Equation (2.7).

Firstly, Equation (2.45) will be discretised by assuming that the displacement field needs to be prescribed only at the location of the nodes. Note that it is possible to

prescribe the displacement of any arbitrary point in the domain, but for reasons of clarity, the derivation will be given only for the nodal positions. In formula form the set of discretised essential boundary conditions is written as:

$$\mathbf{u}(\mathbf{x}_i) - \tilde{\mathbf{u}}(\mathbf{x}_i) = \mathbf{0} \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.47}$$

Field $\mathbf{u}(\mathbf{x})$ is the approximated displacement field and consists of a set of shape functions and nodal displacement degrees of freedom:

$$\sum_{j=1}^{N_{\text{nod}}} \phi_j(\mathbf{x}_i)\mathbf{d}_j - \tilde{\mathbf{u}}(\mathbf{x}_i) = \mathbf{0} \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.48}$$

If the shape functions used satisfy Equation (2.7), the field values of the displacement are equal to the nodal values. Equation (2.48) can be simplified accordingly:

$$\mathbf{d}_i - \tilde{\mathbf{u}}(\mathbf{x}_i) = \mathbf{0} \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.49}$$

Hence, displacements can be directly imposed on the nodal degrees of freedom, since $\mathbf{d}_i = \tilde{\mathbf{u}}(\mathbf{x}_i)$.

Diffuse Approximants

For the diffuse, non-interpolating shape functions, which are frequently found in meshless methods, the derivation as given above is different. Since the Kronecker delta property does not hold for these functions, there is no further simplification possible of Equation (2.48). As a result, multiple degrees of freedom have their influence on the displacement field at the location of a node. A special method has to be used in order to enforce this constraint on the system. Therefore Section 2.5.1, 2.5.2 and 2.5.3, the penalty method, the method of Lagrangian multipliers and the transformation method are discussed respectively. These methods are frequently found whenever boundary displacements need to be prescribed in combination with diffuse shape functions.

Assume that for the following derivations the potential energy Π_{int} in the body is:

$$\Pi_{\text{int}} = \frac{1}{2}\int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{\varepsilon}\, \mathrm{d}\Omega \tag{2.50}$$

where $\boldsymbol{\varepsilon}$ is a linear strain tensor. The symbol Π_{sys} will be used to express the total energy of the system, which will include the internal potential energy, but possibly also other energy terms that relate to the method used to enforce the boundary conditions.

2.5.1 Penalty Method

The main idea of the penalty method is to modify the energy of the system by adding an energy term related to the prescribed displacements in order to enforce Equation (2.48) on the system. The constraint is penalised by squaring this constraint and by

multiplying the result by a user-defined parameter α_{p}. The resulting penalty energy Π_{pen} is given in formula form by:

$$\Pi_{\mathrm{pen}} = \int_{\Gamma_{\mathrm{u}}} \alpha_{\mathrm{p}} \left(\mathbf{u}(\mathbf{x}) - \tilde{\mathbf{u}}(\mathbf{x})\right) \cdot \left(\mathbf{u}(\mathbf{x}) - \tilde{\mathbf{u}}(\mathbf{x})\right) \, \mathrm{d}\Gamma \tag{2.51}$$

where α_{p} is the parameter to enforce the constraints on the system. The total energy of the system is defined as the summation of the penalty energy and the internal energy:

$$\Pi_{\mathrm{sys}} = \Pi_{\mathrm{int}} + \Pi_{\mathrm{pen}} \tag{2.52}$$

By using shape functions ϕ_i, Equation (2.51) is discretised. Thereafter the solution of the problem is found by searching for the minimum energy of the system:

$$\min_{\{\mathbf{d}\}} \Pi_{\mathrm{sys}} \tag{2.53}$$

where $\{\mathbf{d}\}$ is a vector containing all the displacement degrees of freedom of the body under consideration. See Appendix A for its definition.

The main benefit of the penalty method is its low computational cost and its simplicity. The number of degrees of freedom does not increase because of the method. A drawback is that parameter α must be set manually. Setting α too low will enforce the constraints insufficiently. If α is set too high, the system can become ill-conditioned. As a result, Equation (2.48) will only be approximated and not satisfied exactly in practice.

2.5.2 Lagrangian Multipliers

Prescribing displacement boundary conditions in combination with diffuse shape functions by using the method of Lagrangian multipliers was introduced by Belytschko *et al.* [18]. The method assumes the presence of forces on the boundary that have a direction and a magnitude such that Equation (2.45) is satisfied. These forces cannot be defined a priori, hence they make an extra set of degrees of freedom which have to be solved. These forces are also known by the name 'Lagrangian multipliers' and are expressed by the symbol $\boldsymbol{\lambda}$. The additional energy related to these forces becomes:

$$\Pi_{\mathrm{lag}} = \int_{\Gamma_{\mathrm{u}}} \boldsymbol{\lambda}(\mathbf{x}) \cdot \left(\mathbf{u}(\mathbf{x}) - \tilde{\mathbf{u}}(\mathbf{x})\right) \, \mathrm{d}\Gamma \tag{2.54}$$

where Π_{lag} is the additional energy due to the Lagrangian multipliers and $\boldsymbol{\lambda}$ is the vector field of the Lagrangian multipliers. The total energy in the system is a summation of the internal energy and energy resulting from the Lagrangian multipliers:

$$\Pi_{\mathrm{sys}} = \Pi_{\mathrm{int}} + \Pi_{\mathrm{lag}} \tag{2.55}$$

The solution of the problem is found by searching for the minimum energy of the system not only for the displacement field but also for the Lagrangian multipliers. After discretisation of both fields ($\mathbf{u}(\mathbf{x})$ and $\boldsymbol{\lambda}(\mathbf{x})$), the minimisation is as follows:

$$\min_{\{\mathbf{d}\},\{\boldsymbol{\lambda}\}} \Pi_{\text{sys}} \tag{2.56}$$

where $\{\boldsymbol{\lambda}\}$ is a vector with all degrees of freedom related to the field of Lagrangian multipliers $\boldsymbol{\lambda}(\mathbf{x})$.

The main benefit of using the method of Lagrangian multipliers is that Equation (2.45) is exactly satisfied and not approximated as is the case in the penalty method. The downside is that extra computational effort is required because of the added set of degrees of freedom

2.5.3 Transformation Method

The transformation method was proposed by Chen *et al.* [30]. The main idea of this method is to change the degrees of freedom on which it is not possible to directly prescribe displacements, to a set on which it is possible to prescribe displacements.

First Equation (2.1) is restated for all nodes in the domain:

$$\mathbf{u}(\mathbf{x}_j) = \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}_j)\mathbf{d}_i \qquad \text{for } j = 1...N_{\text{nod}} \tag{2.57}$$

The equation expresses that the shape functions multiplied by the nodal values give the displacement field. However, it is also possible to express this equation the other way around:

$$\mathbf{d}_i = \sum_{j=1}^{N_{\text{nod}}} \hat{\phi}_j(\mathbf{x}_i)\mathbf{u}(\mathbf{x}_j) \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.58}$$

$$= \sum_{j=1}^{N_{\text{nod}}} \hat{\phi}_j(\mathbf{x}_i)\hat{\mathbf{d}}_j \qquad \text{for } i = 1...N_{\text{nod}} \tag{2.59}$$

where $\hat{\phi}_j$ is a yet to be determined function and $\hat{\mathbf{d}}_j$ is a displacement degree of freedom which is equal to the displacement field at the nodal positions. Equation (2.59) is interesting since the field values of the displacement appear on the right-hand side of the equation at the place where previously the nodal degrees of freedom were positioned. The displacement field $\mathbf{u}(\mathbf{x}_j)$, or similarly $\hat{\mathbf{d}}_j$, can be prescribed directly. If function $\hat{\phi}$ is known, the field displacements $\hat{\mathbf{d}}$ can be transformed to the nodal displacements $\mathbf{d}$ or vice versa.

Assume a matrix $[\boldsymbol{\Phi}]$ which maps nodal degrees of freedom $\mathbf{d}$ onto the displacement field $\hat{\mathbf{d}}$, and matrix $[\hat{\boldsymbol{\Phi}}]$ which does exactly the opposite. The components of the matrices can be defined as follows:

$$\Phi_{ij} = \boldsymbol{\phi}_i(\mathbf{x}_j) \qquad \text{and} \qquad \hat{\Phi}_{ij} = \hat{\boldsymbol{\phi}}_i(\mathbf{x}_j) \tag{2.60}$$

Moreover, these two matrices are related by the following equation:

$$[\hat{\mathbf{\Phi}}] = [\mathbf{\Phi}]^{-1} \tag{2.61}$$

So by constructing matrix $\mathbf{\Phi}$ and inverting it, shape functions $\hat{\phi}$ can be found. By using Equation (2.59), $\mathbf{d}$ can be transformed to $\hat{\mathbf{d}}$, where the latter set of degrees of freedom can be used to prescribe the displacement field at nodal locations. Hence it is also possible to minimise for these degrees of freedom:

$$\min_{\{\hat{\mathbf{d}}\}} \Pi_{\text{sys}} \tag{2.62}$$

where $\{\hat{\mathbf{d}}\}$ contains all transformed displacement degrees of freedom in the domain. Displacement degrees of freedom can now simply be applied on this vector. As a result Equation (2.48) will be satisfied exactly.

For the derivations as given above, all the degrees of freedom are mapped on the displacement field. However, a considerable computational effort is required for the inverse operation as given in Equation (2.61). Therefore, a small set of nodes lying close to the prescribed boundary is usually sufficient in order to prescribe the displacement at that location exactly. A strategy for this partitioning was proposed by Chen and Wang [32].

2.6 Closure

In this chapter, an overview on meshless methods was presented. Instead of explaining the small details in which meshless methods differ, a categoric overview was given of the main constituents of meshless methods. The main formulations and properties of these constituents were presented.

After summarising and reviewing the meshless developments, it can be stated that there are two topics that require special attention when applying or developing a meshless method. Firstly, literature shows that the numerical integration of equations is complex. Usually, a large number of integration points is required in order to get sufficient accuracy and stability. This will imply a vast amount of points where material models need to be evaluated, hence the computation time is expected to be large. Secondly, for most of the meshless methods as discussed in this literature review, the computational efficiency is low in general. Shape functions are computationally demanding and the numerical integration is required to be of a high order. The next chapter presents a study in which these two points will be addressed.

3

A Comparative Study of Meshless Approximations

This chapter is based on a journal article by the author et al. [105].

3.1 Introduction

The goal of this thesis is to investigate and to develop a meshless method for the simulation of forming processes. Since many meshless methods have been proposed over time, the first question that arises is, what method or methods are best suited for simulating forming processes? Based on literature only it is difficult to get a clear view on whether a method is effective for these kind of simulations, and if so, how it compares to other methods. The research presented in this chapter will try to answer this question by quantitatively comparing a set of meshless methods.

3.1.1 Background

In Chapter 2 many meshless methods have been discussed. Examining all these methods for their performance in a forming process would be impractical. However, it is possible to restrict the total group of methods to a subgroup that is of interest for further development.

A meshless method can be based on a weak form or a strong form of the equilibrium equation (see Section 2.4). Methods using a strong form of the differential equation are for instance the methods of Smooth Particle Hydrodynamics (SPH) and the Reproducing Kernel Particle Method (RKPM). There is a downside to these methods, for which they will be excluded from further research. In a strong form method, spatial gradients of the stress appear in the equilibrium equations, making the use of material models as commonly used in solid mechanics complicated. For instance,

an elasto-plastic material model that can be used in combination with a strong form formulation is not present in literature. For a weak form formulation this is different. Constitutive relations based on the dependency of stress to strain are easily implemented. Therefore, the research as presented in this chapter will focus on the meshless methods employing a weak form only. Moreover, the Galerkin weak form will be chosen because of its beneficial properties as given in Section 2.4.2.

Although material models can be easily incorporated into a weak form, there is a consequence of choosing this type of discretisation of the equilibrium. The equations resulting from the weak form require numerical integration. Since the technique used to perform this numerical integration has a strong influence on the accuracy and the behaviour in incompressibility, this topic requires special attention. Concerning the first aspect, the accuracy, the numerical integration of non-polynomial meshless shape functions has to be of a high order to obtain accurate results. The patch test, for example, is not satisfied for a limited set of Gaussian integration points, even if the shape functions reproduce linear polynomials. Concerning the second point, the simulation of incompressible media, an integration rule has to be selected carefully. The numerical artifact of volumetric locking can be equally present for meshless methods as it is for finite elements. For a detailed description of volumetric locking the reader is referred to Cook *et al.* [35], Zienkiewicz and Taylor [131] and Bathe [14]. For these two reasons given above, the numerical integration of meshless methods has drawn considerable attention over time. The amount of literature on the topic is extensive and research was done for instance for the element-free Galerkin method by Dolbow and Belytschko [43, 44], Beissel *et al.* [15], and by Askes *et al.* [10]. An integration scheme which satisfies the patch test without using an infeasible number of integration points and seems to be free of volumetric locking is the stabilised conforming nodal integration scheme (SCNI) as proposed by Chen *et al.* [33, 34]. The scheme has been investigated in the case of the natural element method by González *et al.* [56] and by Yoo *et al.* [129].

Similarly, applying this nodal integration to finite elements results in a method with interesting properties. The first concern, satisfying the patch test, is not an issue for classical compatible finite elements since this test is satisfied by default. Volumetric locking, on the contrary, seems to be avoided by applying this type of integration. The first development in this field is the nodal pressure tetrahedral as proposed by Bonet and Burton [20]. Afterwards, similar methods have been developed for instance by Dohrmann *et al.* [42], Pires *et al.* [8], Liu *et al.* [84, 85], Krysl and Zhu [74], Puso and Solberg [102] and Hung *et al.* [68]. The stability of nodal integration for both meshless and finite element approximations was investigated by Puso *et al.* [101].

3.1.2 Objective

The objective of this chapter is to quantitatively examine the performance of meshless approximations and their numerical integration. To investigate this performance, four numerical tests are performed; two in elasticity, one in elasto-plasticity and one inf-sup test.

As explained in preceding sections, the amount of meshless shape functions as proposed in literature is extensive. Therefore, for this study, a subset of three shape functions are chosen such that a representative view on meshless approximations is presented. The first shape function is the moving least squares function. This function is one of the most commonly used approximations in the meshless field. Secondly, a recent development, namely the local maximum entropy approximation, is included in the analysis. This approximation possesses similar properties to the moving least squares approximation, although it can simplify the handling of boundary conditions. Finally a linear interpolation based on a Delaunay triangulation is included. In the finite element method, these shape functions are commonly used for the construction of linear triangular finite elements. The first two shape functions, the moving least squares function and the local maximum entropy function are typical diffuse approximations. Their shape function is based on a domain of influence instead of a mesh. The latter shape function, the triangle interpolation, is a compact shape function. The number of nodes that have an influence on the displacement of a specific point is usually small when compared to diffuse approximations.

Concerning the evaluation of the weak form, two numerical integration schemes will be tested. These are the stabilised conforming nodal integration scheme and a Gaussian integration scheme based on a Delaunay triangulation.

Several combinations can be made by combining a shape function and an integration scheme. Using the triangular interpolation with a Gaussian integration scheme results in a linear triangular finite element [131]. Integrating the same shape function nodally will give a scheme as proposed by Dohrmann *et al.* [42]. The scheme as proposed by Chen *et al.* [33] is obtained by integrating a diffuse approximation nodally. All these combinations will be compared in this study.

3.1.3 Outline

This chapter is organised as follows. Firstly, an introduction to the shape functions used and the integration schemes is given in Section 3.2. A short outline of the computer program as used for the analysis is presented. Section 3.3 gives the results of the numerical study into the performance of all combinations of shape functions and integration schemes. Afterwards the computational efficiency of the methods is compared. The conclusions are given in the final section.

3.2 Governing Equations

3.2.1 General Formulations

The starting point for the derivation of nodal equilibrium is the equation of equilibrium in the strong form:

$$\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f} = \mathbf{0} \qquad \forall\, \mathbf{x} \in \Omega \tag{3.1}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\mathbf{f}$ is a vector representing the body forces and Ω is the domain under consideration. The boundary conditions for the equilibrium equations are:

$$\mathbf{u}(\mathbf{x}) - \tilde{\mathbf{u}}(\mathbf{x}) = \mathbf{0} \qquad \forall\, \mathbf{x} \in \Gamma_{\mathrm{u}} \tag{3.2}$$

$$\boldsymbol{\sigma} \cdot \mathbf{n} - \tilde{\mathbf{t}} = \mathbf{0} \qquad \forall\, \mathbf{x} \in \Gamma_{\mathrm{t}} \tag{3.3}$$

where $\tilde{\mathbf{u}}$ is a prescribed displacement on boundary Γ_{u}, $\tilde{\mathbf{t}}$ is a prescribed traction on the boundary Γ_{t} and $\mathbf{n}$ is the outward normal on the boundary. The nodal equilibrium is obtained by applying a weighted residual formulation on the strong equilibrium of Equation (3.1), using Galerkin's method, and discretising the displacement field:

$$\int_{\Omega} [\mathbf{B}]^{\mathrm{T}} \{\boldsymbol{\sigma}\}\, \mathrm{d}\Omega = \int_{\Gamma_{\mathrm{t}}} [\mathbf{N}]^{\mathrm{T}} \left\{\tilde{\mathbf{t}}\right\} \mathrm{d}\Gamma + \int_{\Omega} [\mathbf{N}]^{\mathrm{T}} \{\mathbf{f}\}\, \mathrm{d}\Omega \tag{3.4}$$

$$\{\mathbf{F}_{\mathrm{int}}\} = \{\mathbf{F}_{\mathrm{ext}}\} \tag{3.5}$$

where $\mathbf{F}_{\mathrm{int}}$ is the internal force vector and $\mathbf{F}_{\mathrm{ext}}$ is the external force vector. The study as presented in this chapter is focussed only on the performance of the numerical integration scheme and the shape function used. The governing equations are therefore simplified by assuming the strain measure to behave linearly and to exclude the effect of large geometrical changes of the body under consideration. As a result, there is no non-linearity present in the model besides non-linearity related to the material model chosen. The matrices $\mathbf{N}$ and $\mathbf{B}$, which relate the nodal displacement vector $\mathbf{d}$ to the field displacements and strains, are defined as:

$$\{\mathbf{u}\} = [\mathbf{N}]\{\mathbf{d}\} \tag{3.6}$$

$$\{\boldsymbol{\varepsilon}\} = [\mathbf{B}]\{\mathbf{d}\} \tag{3.7}$$

Matrix $\mathbf{B}$ contains the terms of the small strain tensor:

$$\boldsymbol{\varepsilon} = \frac{1}{2}\left(\overrightarrow{\nabla}\mathbf{u} + \mathbf{u}\overleftarrow{\nabla}\right) \tag{3.8}$$

The matrices $\mathbf{N}$ and $\mathbf{B}$ are constructed by using shape functions $\boldsymbol{\phi}$. Their formulations are given in Section 3.2.2. The integrator $\int_{\Omega} \ldots \mathrm{d}\Omega$ of Equation (3.4) is worked out with two different numerical integration schemes as will be explained in Section 3.2.3.

For the constitutive equations a linear elastic model and an elasto-plastic model are included. If the latter model is used, the nodal equilibrium represented by Equation (3.5) is found by a Newton–Raphson iterative procedure. Within this procedure a prediction of the displacements is made by linearising the internal force vector:

$$[\mathbf{K}] = \frac{\partial}{\partial \{\mathbf{d}\}^{\mathrm{T}}} \{\mathbf{F}_{\mathrm{int}}\} \tag{3.9}$$

$$= \int_{\Omega} [\mathbf{B}]^{\mathrm{T}} [\mathbf{C}] [\mathbf{B}]\, \mathrm{d}\Omega \tag{3.10}$$

where $\mathbf{K}$ is the stiffness matrix and $\mathbf{C}$ is the (algorithmic) material tangent matrix. In this research, a J2 radial return elasto-plastic material model is used. The formulation

as well as the implementation of this model, including the algorithmic tangent and the stress update, can be found in Simo and Hughes [111], among others. The following set of equations is solved:

$$[\mathbf{K}]\left\{\Delta\mathbf{d}^k\right\} = \left\{\mathbf{F}^k_{\text{ext}}\right\} - \left\{\mathbf{F}^k_{\text{int}}\right\} \tag{3.11}$$

where $\Delta\mathbf{d}^k$ is a vector containing the iterative displacement degrees of freedom and index k denotes the current iteration step. The total vector of displacement degrees of freedom is found by summation of $\Delta\mathbf{d}^k$ over all iterations k.

3.2.2 Shape Functions

Approximating the displacement field in the case of meshless methods or the finite element method is done by a set of shape functions $\boldsymbol{\phi}(\mathbf{x})$. The parameterised displacement field can be represented as:

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x})\mathbf{d}_i \tag{3.12}$$

where N_{nod} is the number of nodes in the model, and $\mathbf{u}$ is the approximated displacement field. The vector $\mathbf{d}_i$ contains the displacement degrees of freedom of node i and is defined for 2D as follows:

$$\mathbf{d}_i = \left\{ \begin{array}{cc} d^i_x & d^i_y \end{array} \right\}^{\mathrm{T}} \tag{3.13}$$

Although there is a lot of freedom in defining $\boldsymbol{\phi}$, for the successful application of a shape function in solid mechanics two properties are essential. These requirements are the two reproducibility conditions as given in Section 2.3.3. Shape functions which satisfy these conditions will reproduce a constant strain field exactly. In this study, three different types of shape functions are included that satisfy both the two conditions. Below, a short summary of their main formulations is presented.

Moving Least Squares

Moving Least Squares (MLS) approximations were first introduced in the field of computational solid mechanics by means of the diffuse element method by Nayroles *et al.* [95]. See Section 2.3.6 for the general formulations of moving least squares shape functions. For the current study, the following polynomial will be used:

$$\mathbf{p}(\boldsymbol{\xi}) = \left\{ \begin{array}{ccc} 1 & \xi & \eta \end{array} \right\}^{\mathrm{T}} \tag{3.14}$$

The kernel function used in 2D is constructed by multiplying two 1D functions ω_1:

$$\omega(\mathbf{x} - \boldsymbol{\xi}, \gamma) = \omega_1\left(s_x\right)\omega_1\left(s_y\right) \tag{3.15}$$

where local coordinates s_x and s_y are defined as follows:

$$s_x = \frac{d(x,\xi)}{\beta} \quad \text{and} \quad s_y = \frac{d(y,\eta)}{\beta} \tag{3.16}$$

The distance between point $\boldsymbol{\xi}$ and $\mathbf{x}$ in x direction is given by $d(x,\xi)$ and similarly, the distance along the y direction is given by $d(y,\eta)$. Note that it is assumed that coordinate systems $\mathbf{x}$ and $\boldsymbol{\xi}$ have the same orientation and magnitude. Parameter β gives an absolute measure of the domain of influence of kernel function ω. This parameter is found by considering $\beta = \gamma \cdot h$, where h is an average measure of the spacing of nodes and γ controls the relative size of the domain of influence of the kernel function. For the 1D kernel function ω_1, a cubic spline is chosen:

$$\omega_1(s) = \begin{cases} \frac{2}{3} - 4\,s^2 + 4s^3 & \text{for} \quad s \leqslant \frac{1}{2} \\ \frac{4}{3} - 4\,s + 4\,s^2 - \frac{4}{3}s^3 & \text{for} \quad \frac{1}{2} < s \leqslant 1 \\ 0 & \text{for} \quad s > 1 \end{cases} \tag{3.17}$$

As a result of the choice of the kernel function ω, the MLS approximation has a continuity of degree 2. For the current analysis, γ is chosen to be equal for all nodes. In a regularly spaced grid, h is equal to the minimum distance between two neighbouring nodes. If the value of parameter γ is increased, the shape functions become more diffuse.

Local Maximum Entropy

Local Maximum Entropy (LME) shape functions were recently introduced by Arroyo and Ortiz [9]. Information-theoretic principles are the foundations for the construction of the shape functions. See Section 2.3.8 for the main formulations.

The compactness of the local maximum entropy approximation is controlled by a user-defined parameter μ. By setting μ either compact or diffuse shape functions can be obtained. For the current research, μ is chosen to be equal for all nodes in the domain. Moving least squares shape functions with high μ-value are diffuse. Setting μ low will force the approximation to interpolation. Because of the definition for μ in local maximum entropy shape functions, this trend is exactly opposite. A high μ-value gives a local approximation and a low value gives a diffuse approximation.

Linear Triangle Interpolation

LME approximations with a high μ value result in a linear interpolation on Delaunay triangles (excluding degenerate cases). For this reason an explicit linear triangular shape function is considered as well. A triangle spanned by vertices $\mathbf{x}_i$, $\mathbf{x}_j$ and $\mathbf{x}_k$ is a Delaunay triangle V_{ijk} if $\mathbf{x}_\mathrm{c}$ is not an empty set:

$$\begin{aligned} V_{ijk} &= \{\mathbf{x} \in \mathrm{conv}\,(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)\} \\ \text{if:} \quad & \mathbf{x}_\mathrm{c} \neq \emptyset \\ \text{where:} \quad & \mathbf{x}_\mathrm{c} = \big\{\mathbf{x} \in \Re^2 \mid d(\mathbf{x},\mathbf{x}_i) = d(\mathbf{x},\mathbf{x}_j) = d(\mathbf{x},\mathbf{x}_k), \\ & \qquad d(\mathbf{x},\mathbf{x}_i) < d(\mathbf{x},\mathbf{x}_l)\ \forall\ l \neq i,j,k\big\} \end{aligned} \tag{3.18}$$

where $\mathbf{x}_\mathrm{c}$ is the circumcentre of the Delaunay triangle. This circumcentre is defined as the centre of a circle intersecting the three vertices of a triangle. The convex set

is denoted by conv() and contains all points that are enclosed in a triangle. Linear shape functions based upon triangles are well known in finite element analysis and expressions can be found in most books on finite element technology; for instance in Zienkiewicz and Taylor [131]:

$$\left\{ \begin{array}{c} \phi_i(\mathbf{x}) \\ \phi_j(\mathbf{x}) \\ \phi_k(\mathbf{x}) \end{array} \right\} = \left[\begin{array}{ccc} x_i & x_j & x_k \\ y_i & y_j & y_k \\ 1 & 1 & 1 \end{array} \right]^{-1} \left\{ \begin{array}{c} x \\ y \\ 1 \end{array} \right\} \qquad \forall\, \mathbf{x} \in V_{ijk} \tag{3.19}$$

where x_i and y_i are the coordinates of node i. The most profound argument for choosing this type of shape function is its low computational cost and its simple formulation. Another advantage is the straightforward imposition of boundary conditions since it possesses the Kronecker delta property for all nodes in the domain. The order of continuity is C^0, hence smooth strain fields are not obtained for limited sets of nodes.

3.2.3 Integration Schemes

In this research, two integration schemes are used to evaluate Equations (3.4) and (3.10) numerically. The first scheme is the well known 'Gauss' integration scheme that is commonly used in finite elements. The second scheme is the stabilised conforming nodal integration scheme, also known by the abbreviation of SCNI. Both schemes are explained in brief below.

'Gauss' Integration

'Gauss' integration of the internal force vector is formulated as follows:

$$\begin{aligned} \{\mathbf{F}_{\text{int}}\} &= \int_\Omega [\mathbf{B}]^{\text{T}} \{\boldsymbol{\sigma}\}\, \mathrm{d}\Omega \\ &\approx \sum_{k=1}^{N_{\text{int}}} [\mathbf{B}(\mathbf{x}_k)]^{\text{T}} \{\boldsymbol{\sigma}(\mathbf{x}_k)\}\, \Delta\Omega_k \end{aligned} \tag{3.20}$$

N_{int} is the total number of integration points in the body and $\mathbf{x}_k$ is the location of an integration point. Usually the summation $\sum_{k=1}^{N_{\text{int}}}$ is split into a sum over elements or integration cells and over integration points for such an element or cell. In this research an integration rule within a triangle is used. The starting point is a cloud of nodes which is triangulated by means of a Delaunay triangulation. Within each triangle an integration rule is defined.

If, for finite elements, the integration rule is chosen in accordance with the interpolation functions, the patch test is satisfied. Details of this test can be found in Zienkiewicz and Taylor [131], among others. Note that non-polynomial meshless approximations in general do not pass the patch test, even if these approximations possess first order reproducibility. Inexact integration precludes in this case an exact evaluation of the linear displacement field.

Nodal Integration

Nodal integration of the internal force vector can be expressed as:

$$\begin{aligned}\{\mathbf{F}_{\text{int}}\} &= \int_{\Omega} [\mathbf{B}]^{\text{T}} \{\boldsymbol{\sigma}\} \, \text{d}\Omega \\ &\approx \sum_{i=1}^{N_{\text{nod}}} [\mathbf{B}(\mathbf{x}_i)]^{\text{T}} \{\boldsymbol{\sigma}(\mathbf{x}_i)\} \, \Delta\Omega_i \end{aligned} \tag{3.21}$$

where N_{nod} is the number of nodes, $\mathbf{x}_i$ is the location of a node and $\Delta\Omega_i$ is the volume accompanying that particular node. Matrix $\mathbf{B}$ is the strain-displacement matrix which is consistent with the displacement field. Several problems arise when using an integration scheme according to Equation (3.21). The first problem is that the patch test is not satisfied for non-polynomial approximations. The second problem is the lack of stability. An improved nodal integration scheme was proposed by Chen *et al.* [33]. This stabilised conforming nodal integration scheme (SCNI) modifies the definition of $\mathbf{B}$ in order to avoid these two problems. The essence of the method is that an assumed displacement gradient at a node is constructed by averaging the displacement gradient over a cell accompanying that node:

$$\bar{\nabla}\mathbf{u}(\mathbf{x}_i) = \frac{1}{\Omega_i} \int_{\Omega_i} \nabla\mathbf{u} \, \text{d}\Omega \tag{3.22}$$

The volume integral can be rewritten by means of the Gauss divergence theorem to a surface integral:

$$\bar{\nabla}\mathbf{u}(\mathbf{x}_i) = \frac{1}{\Omega_i} \int_{\Gamma_i} \mathbf{n}\,\mathbf{u} \, \text{d}\Gamma \tag{3.23}$$

where $\mathbf{n}$ is the outward normal on boundary Γ_i of the cell Ω_i. The assumed strain field used for the integration becomes:

$$\bar{\boldsymbol{\varepsilon}}(\mathbf{x}_i) = \frac{1}{2} \left(\bar{\nabla}\mathbf{u}(\mathbf{x}_i) + (\bar{\nabla}\mathbf{u}(\mathbf{x}_i))^{\text{T}} \right) \tag{3.24}$$

Figure 3.1 gives an illustration on the nodal integration scheme. Section 3.2.5 gives a description on the construction of the cells as displayed in Figure 3.1. The modified $\mathbf{B}$-matrix at node $\mathbf{x}_i$ in 2D becomes:

$$\left[\bar{\mathbf{B}}(\mathbf{x}_i)\right] = \left[\begin{array}{cccc} \bar{\mathbf{B}}_1(\mathbf{x}_i) & \bar{\mathbf{B}}_2(\mathbf{x}_i) & \ldots & \bar{\mathbf{B}}_{N_{\text{nod}}}(\mathbf{x}_i) \end{array} \right] \tag{3.25}$$

where the contribution of shape function j is defined as:

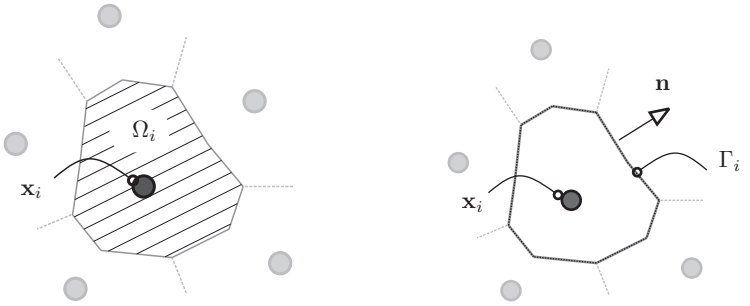
$$\left[\bar{\mathbf{B}}_j(\mathbf{x}_i)\right] = \frac{1}{\Delta\Omega_i} \int_{\Gamma_i} \begin{bmatrix} \phi_j n_1 & 0 \\ 0 & \phi_j n_2 \\ \phi_j n_2 & \phi_j n_1 \end{bmatrix} \text{d}\Gamma \tag{3.26}$$

The resulting internal force vector becomes:

$$\{\mathbf{F}_{\text{int}}\} = \sum_{i=1}^{N_{\text{nod}}} \left[\bar{\mathbf{B}}(\mathbf{x}_i)\right]^{\text{T}} \{\bar{\boldsymbol{\sigma}}(\mathbf{x}_i)\} \, \Delta\Omega_i \tag{3.27}$$

where $\bar{\boldsymbol{\sigma}}(\mathbf{x}_i)$ is the stress tensor computed with the assumed strain $\bar{\boldsymbol{\varepsilon}}(\mathbf{x}_i)$. The stiffness matrix becomes:

$$[\mathbf{K}] = \sum_{i=1}^{N_{\text{nod}}} [\bar{\mathbf{B}}(\mathbf{x}_i)]^{\mathrm{T}} [\mathbf{C}] [\bar{\mathbf{B}}(\mathbf{x}_i)] \Delta\Omega_i \tag{3.28}$$



(a) volume integration according to Equation (3.22)

(b) contour integration according to Equation (3.23)

Figure 3.1: An illustration of the cell integration for SCNI.

An interesting aspect of integrating the weak equations nodally is that a material point is at the location of the node. Especially if more sophisticated, history-dependent material models are used, all data concerning the material model can be stored at the location of the node. This can simplify, for instance, re-meshing or convecting algorithms, such that the spatial distribution of the state variables of the material model is optimally preserved. Note that because of nodal integration the number of nodal connections increases: all shape functions holding a non-zero value in Ω_i contribute to $\bar{\mathbf{B}}(\mathbf{x}_i)$. Matrix $\mathbf{K}$ will therefore become less sparse.

3.2.4 Applying Boundary Conditions

Applying the boundary conditions in case of diffuse meshless shape functions requires a different approach than commonly used for finite elements. For finite elements the Kronecker delta property holds, which implies that the field displacement at the position of the node is equal to the nodal degree of freedom:

$$\mathbf{u}(\mathbf{x}_i) = \mathbf{d}_i \tag{3.29}$$

where $\mathbf{x}_i$ is the nodal location. As explained in Section 2.5, most diffuse meshless approximations do not satisfy this condition. Displacements cannot be enforced by simply prescribing entries in the nodal displacement vector $\mathbf{d}$.

In this chapter, the method of Lagrangian multipliers is applied to enforce prescribed boundary displacements in the manner of Belytschko *et al.* [18]. See Section 2.5.2

for the main formulations of the method of Lagrangian multipliers. The set of nodal degrees of freedom is expanded by a set of Lagrangian multipliers. The system of equations with the added degrees of freedom becomes:

$$\begin{bmatrix} \mathbf{K} & \mathbf{G}^{\mathrm{T}} \\ \mathbf{G} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \Delta \mathbf{d}_i \\ \Delta \boldsymbol{\lambda}_i \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}^i_{\mathrm{ext}} - \mathbf{F}^i_{\mathrm{int}} \\ \mathbf{g} \end{Bmatrix} \tag{3.30}$$

where $\Delta \boldsymbol{\lambda}_i$ is a vector containing the Lagrangian multipliers and matrix $\mathbf{K}$ is the stiffness matrix as defined in Equation (3.10) or Equation (3.28) depending on the integration scheme. Vector $\mathbf{g}$ and matrix $\mathbf{G}$, are defined as:

$$\{\mathbf{g}\} = -\int_{\Gamma} [\mathbf{N}_{\boldsymbol{\lambda}}]^{\mathrm{T}} \{\tilde{\mathbf{u}}\} \, \mathrm{d}\Gamma \tag{3.31}$$

$$[\mathbf{G}] = -\int_{\Gamma} [\mathbf{N}_{\boldsymbol{\lambda}}]^{\mathrm{T}} [\mathbf{N}] \, \mathrm{d}\Gamma \tag{3.32}$$

Equations (3.31) and (3.32) are evaluated with a nodal integration rule as used by Pannachet and Askes [100]:

$$\{\mathbf{g}\} = -\sum_{i=1}^{N_{\mathrm{nod}}} [N_{\boldsymbol{\lambda}}(\mathbf{x}_i)]^{\mathrm{T}} \{\tilde{\mathbf{u}}(\mathbf{x}_i)\} \Gamma_i \tag{3.33}$$

$$[\mathbf{G}] = -\sum_{i=1}^{N_{\mathrm{nod}}} [N_{\boldsymbol{\lambda}}(\mathbf{x}_i)]^{\mathrm{T}} [\mathbf{N}(\mathbf{x}_i)] \, \Gamma_i \tag{3.34}$$

Furthermore, the shape functions related to the Lagrangian multipliers are chosen to have the Kronecker delta property.

$$N^j_{\boldsymbol{\lambda}}(\mathbf{x}_i) = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i \neq j \end{cases} \tag{3.35}$$

As a result only displacement related shape functions need to be evaluated at locations $\mathbf{x}_i$.

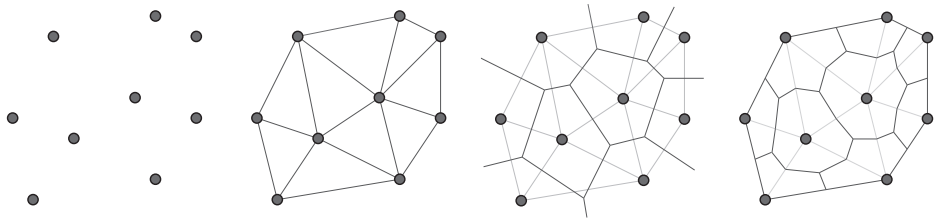
The boundary conditions for the linear triangle interpolation are enforced by using a simple row reduction technique as is standard in finite element analysis. Prescribed displacements are multiplied by corresponding columns in $\mathbf{K}$ and added to the right-hand side of Equation (3.11).

3.2.5 Triangulations and Tessellations

This section will discuss the triangulation and tessellation algorithm as used in the analysis. The SCNI integration scheme requires a tessellation in order to construct the modified strain matrix $\bar{\mathbf{B}}$. The INT shape function and the Gaussian integration scheme are defined upon a triangulation. Moreover, this triangulation is beneficial for the MLS and LME shape functions as well, since this data structure can be used for efficient neighbour searching.

The strategy to obtain these triangles and cells is as follows. First of all, a cloud of nodes is triangulated with a Delaunay triangulation according to Equation (3.18). Afterwards, the concept of α-shapes by Edelsbrunner *et al.* [48] ensures that concave boundaries can be represented without defining each concave boundary section explicitly. A description of the method for continuum mechanics can be found in Cueto *et al.* [36] and Alfaro *et al.* [1].

For the SCNI integration, the most simple choice would be to use the Voronoi tessellation [123], which is the dual of the Delaunay triangulation [40]. If the triangulation is known, the tessellation can be obtained easily and vice versa. There is one drawback however, which makes the use of the Voronoi tessellation troublesome. Cells on the boundary of the domain are not properly defined and can have an infinite volume. Therefore a tessellation technique as proposed by Chen *et al.* [34] is employed. It takes the midpoints of the sides of a Delaunay triangle and the centroid of that triangle. The cell of a node is made by connecting straight lines through these points for all triangles connected to that particular node. Boundaries can be tessellated without problems. If the Delaunay triangulation is known, this tessellation can be constructed with little extra effort. Figure 3.2 gives a visualisation of the geometrical objects as described above.



(a) cloud of nodes (b) Delaunay triangles (c) Voronoi cells (d) modified cells

Figure 3.2: Computational geometrical objects used for the analysis.

3.2.6 Overview of the Implementation

To switch easily between shape function, integration scheme, or method to apply boundary conditions, a code was written of which the architecture will be shortly outlined here.

Figure 3.3 displays a flowchart of the computer program. The components named MLS, LME and INT are abbreviations of moving least squares, local maximum entropy and linear triangular interpolation respectively. The Galerkin weak form is integrated with the two integration schemes. These are the standard 'Gaussian' integration scheme and the stabilised conforming nodal integration scheme abbreviated with STD and SCNI respectively. From a programming point of view, the main difference between these two integration rules is that the STD integration consists of a loop over triangles for the stiffness matrix and internal force vector and

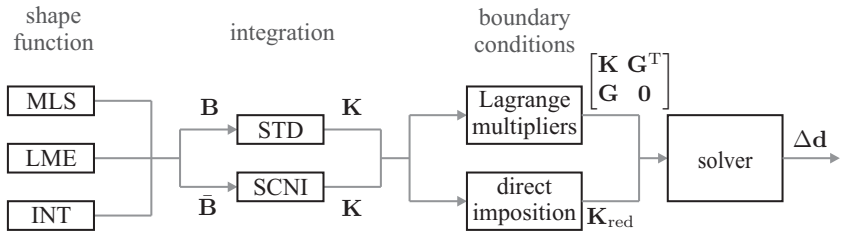


Figure 3.3: A schematic representation of the program as implemented in MATLAB.

the SCNI integration loops over nodes for the assembly procedure. Furthermore, the STD integration requires the gradients of the shape functions whereas for the SCNI integration just the shape function values are required. The averaged derivatives $\bar{\nabla}\mathbf{u}$ follow from the divergence theorem.

The method to apply the boundary conditions is selected depending on which shape function is chosen. Lagrangian multipliers will be used for the LME and MLS functions. For the INT function the row reduction technique is employed, resulting in the reduced stiffness matrix $\mathbf{K}_{\text{red}}$.

3.3 Numerical Performance

3.3.1 Introduction

In this section the numerical performance of the shape functions and integration schemes will be examined. First of all, a test in linear elasticity is performed to examine the accuracy and convergence properties of the shape functions and integration schemes. An infinite plate with a hole is used as test problem. Secondly, an analysis is done to examine the performance of various combinations in elasto-plasticity. The effect of the type of integration, the order of the Gaussian integration and the compactness of the diffuse approximations will be investigated. Finally the computational efficiency of the implementation will be assessed.

For the MLS and LME approximations, parameters γ and μ have to be set. These parameters are set such that the shape functions have a similar domain of influence and that a sufficient number of nodes is included in these domains. For the MLS approximation $\gamma = 2.6$, and for the LME approximation $\mu = 1$, unless stated otherwise. Figure 3.4 gives a visualisation of the shape functions with these parameters. These settings for the domain of influence were found to give a stable response in most cases. Making the shape functions too compact, for instance, can result in failure of the shape function algorithm.

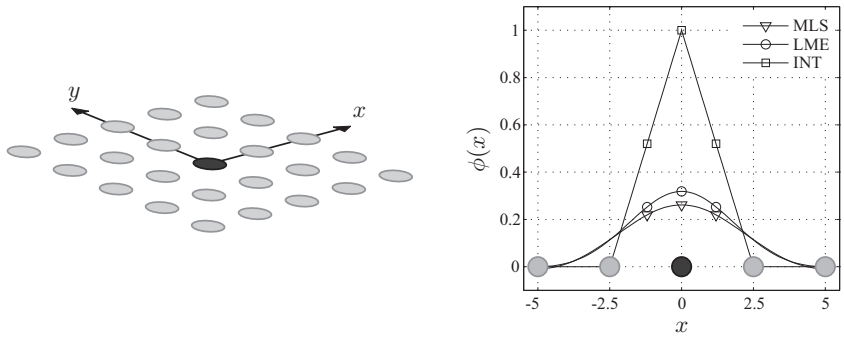


Figure 3.4: The shape functions as used for the analysis.

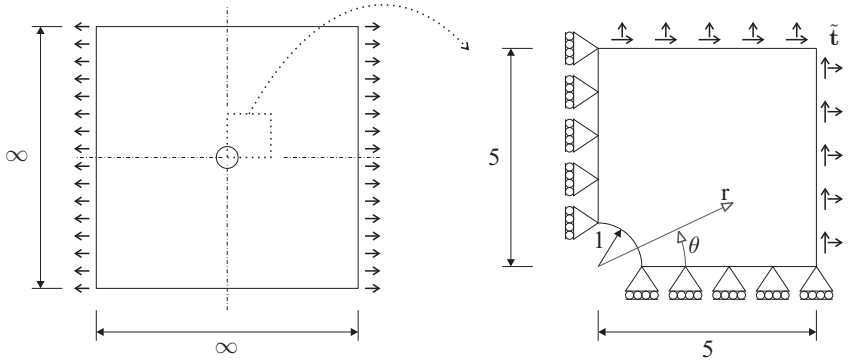


Figure 3.5: The geometry of the infinite plate with a hole problem.

3.3.2 Infinite Plate with a Hole

In this test the accuracy and convergence of all combinations of shape functions and integration schemes are tested on the problem of an infinite plate with a hole. Figure 3.5 shows the geometry of the infinite plate at the left-hand side, and the modelled part of the plate at the right-hand side. The infinite plate is loaded in horizontal direction with a uniform traction. The geometry is simplified by using symmetry conditions and only evaluating a small part of the plate close to the location of the hole. At the symmetry lines the appropriate displacement boundary conditions are imposed, and at the free boundary the known exact stress field is applied. The exact solution of the problem can be found for instance in Timoshenko and Goodier [119]. The exact stress field is:

$$\sigma_{xx}(r,\theta) = 1 - \frac{1}{r^2}\left(\frac{3}{2}\cos 2\theta + \cos 4\theta\right) + \frac{3}{2\,r^4}\cos 4\theta \tag{3.36}$$

$$\sigma_{yy}(r,\theta) = -\frac{1}{r^2}\left(\frac{1}{2}\cos 2\theta - \cos 4\theta\right) - \frac{3}{2\,r^4}\cos 4\theta \tag{3.37}$$

$$\sigma_{xy}(r,\theta) = -\frac{1}{r^2}\left(\frac{1}{2}\sin 2\theta + \sin 4\theta\right) + \frac{3}{2\,r^4}\sin 4\theta \tag{3.38}$$

and the corresponding displacement field is:

$$u_x(r,\theta) = \frac{1}{8\mu}\left(r\,(\kappa+1)\cos\theta + \frac{2}{r}\left((1+\kappa)\cos\theta + \cos 3\theta\right) - \frac{2}{r^3}\cos 3\theta\right) \tag{3.39}$$

$$u_y(r,\theta) = \frac{1}{8\mu}\left(r\,(\kappa-3)\sin\theta + \frac{2}{r}\left((1-\kappa)\sin\theta + \sin 3\theta\right) - \frac{2}{r^3}\sin 3\theta\right) \tag{3.40}$$

Parameters κ and μ are defined for the plane strain case as:

$$\kappa = 3 - 4\nu \tag{3.41}$$

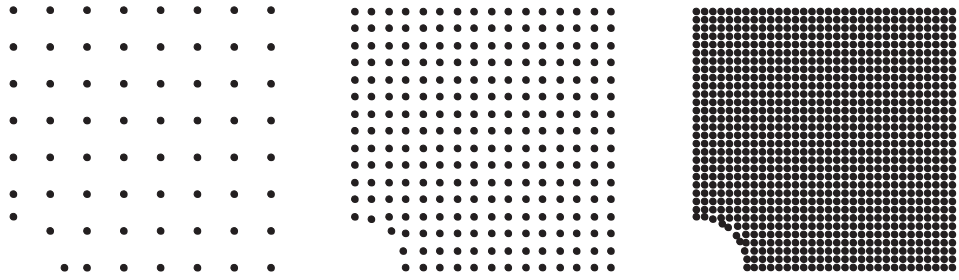
$$\mu = \frac{E}{2\,(1+\nu)} \tag{3.42}$$

The Young's modulus and the Poisson's ratio are $E = 10$ and $\nu = 0.3$ respectively. For the STD integration scheme a three-point integration rule within a triangle is used. The SCNI integration scheme employs a two-point Gauss rule on each of the facets of a cell. To compare the accuracy of a combination of shape function and integration scheme, an error norm on the displacement is used. The error norm is a discrete version of the $\|\cdot\|_{\mathrm{L2}}$-norm and samples the displacement error only at the nodes. This norm is used to avoid the problem of introducing errors in the computation of the integrand of the $\|\cdot\|_{\mathrm{L2}}$ error norm, as was pointed out by González *et al.* [56]. The error is given by:

$$\|e_{\mathrm{u}}\|_2 = \frac{1}{N_{\mathrm{nod}}}\sqrt{\sum_{i=1}^{N_{\mathrm{nod}}} \|\mathbf{u}\,(\mathbf{x}_i) - \mathbf{u}_{\mathrm{exact}}\,(\mathbf{x}_i)\,\|^2} \tag{3.43}$$

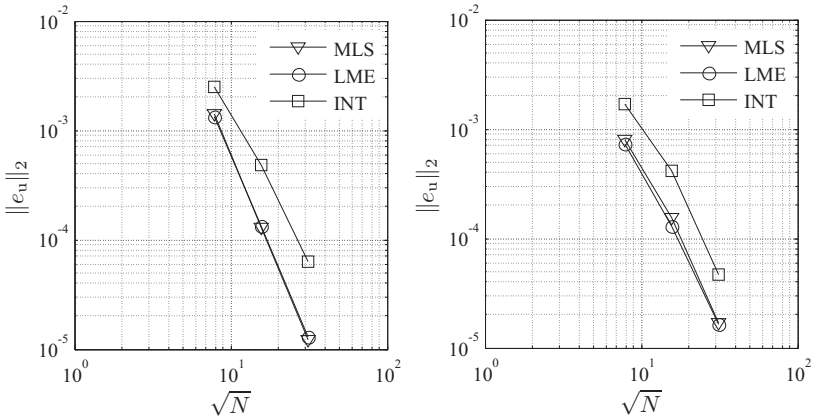
The location of a node is given by $\mathbf{x}_i$, the exact solution is $\mathbf{u}_{\text{exact}}$, and N_{nod} is the total number of nodes.

Figure 3.6 shows three nodal grids from coarse to fine as used for this test. The number of nodes for the grids in the order from coarse to fine are 63, 248 and 993 respectively.



(a) coarse (b) medium (c) fine

Figure 3.6: Three nodal grids for the plate with a hole problem.



(a) STD Gaussian integration (b) SCNI integration

Figure 3.7: The $\|e_u\|_2$ error norm for the plate with hole problem.

Figure 3.7 shows the $\|e_u\|_2$ error for the integration schemes and the three shape functions. First of all, it can be stated that for all combinations, convergence is obtained. Irrespective of the combination of methods chosen, the exact solution is likely to be obtained by increasing the number of nodes. Secondly, when comparing the shape functions it can be seen that, regardless of the integration scheme used, the two diffuse approximations are more accurate than the linear interpolation. The error norms for the MLS and LME shape functions are two to three times smaller than the

error norm for the INT function. The error for the MLS and LME approximation is found to be nearly identical as a result of the selection of parameters γ and μ. Thirdly, the two integration schemes give approximately the same accuracy and rate of convergence for a specific shape function. Compared to the Gaussian integration, integrating the INT function with the SCNI scheme increases the accuracy.

3.3.3 Distortion Analysis

In the following test, the combinations of shape functions and integration schemes are tested on their behaviour on distorted grids. The problem used to analyse this behaviour is the pure bending of a square piece. The details of the problem are given in Figure 3.8. At the right-hand side of the square, a traction in x-direction is prescribed varying linearly between 7.5 and -7.5. The problem is discretised by five different nodal grids as shown in Figure 3.9. Small irregularities and substantial distortions are applied to a regular grid with increasing severity. The energy error norm is monitored over the five grids. This norm is defined as:

$$\|e_{\mathrm{u}}\|_{\mathrm{E}}^2 = \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_{\mathrm{exact}}) : \mathbf{C} : (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_{\mathrm{exact}}) \, \mathrm{d}\Omega \tag{3.44}$$

where $\boldsymbol{\varepsilon}_{\mathrm{exact}}$ is the exact strain field. The exact stress field of the problem can be found by simply considering the prescribed traction on the right-hand side of the square piece of material:

$$\sigma_{xx}(\mathbf{x}) = \frac{3}{2} y \tag{3.45}$$

$$\sigma_{yy}(\mathbf{x}) = 0 \tag{3.46}$$

$$\sigma_{xy}(\mathbf{x}) = 0 \tag{3.47}$$

By using the constitutive behaviour, the exact strain field of the problem can be found:

$$\varepsilon_{xx}(\mathbf{x}) = \frac{3E(1-\nu)}{2(1+\nu)(1-2\nu)} y \tag{3.48}$$

$$\varepsilon_{yy}(\mathbf{x}) = \frac{3E\nu}{2(1+\nu)(1-2\nu)} y \tag{3.49}$$

$$\varepsilon_{xy}(\mathbf{x}) = 0 \tag{3.50}$$

The integral in Equation (3.44) will be evaluated by the two integration schemes given in Section 3.2.3 depending on which integration scheme is used to evaluate the weak form.

Figure 3.10 shows the results for the two integration schemes and the three shape functions. Several conclusions can be drawn from the figures. First of all, for low amounts of distortion (grids 1 and 2), the most accurate results are obtained by employing an MLS or LME shape function with a Gaussian integration scheme. Increasing the amount of distortion for the Gaussian integration scheme increases

the error in the energy norm. For grid 5, the most severely distorted grid, the MLS and LME shape functions could not be constructed for every location $\mathbf{x}$. For the MLS functions, for instance, parameters $\mathbf{a}(\mathbf{x})$ cannot be determined uniquely at these points. Therefore, Figure 3.10 does not include the energy error norm on grid 5 for these approximations. The domain of influence of these shape functions has to be increased, such that sufficient neighbours are present to define these shape functions properly. The Gaussian integrated triangular interpolation is inaccurate both on the regular grid as well as on the irregular grids.

The SCNI integrated solutions seem to be less affected by the distortion in general. Although the MLS function on grid 4 is found to be inaccurate, overall a smaller influence of the distortion is found. Similar to the Gaussian integrated solutions, problems were encountered with the MLS and LME functions on grids 4 and 5. It can be seen that the performance of the INT function improves considerably by employing the SCNI integration.

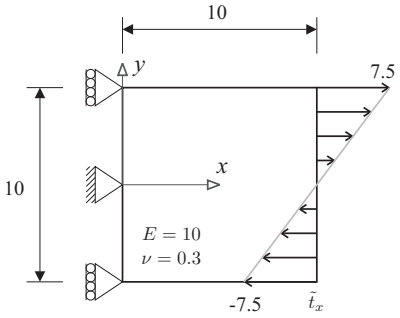
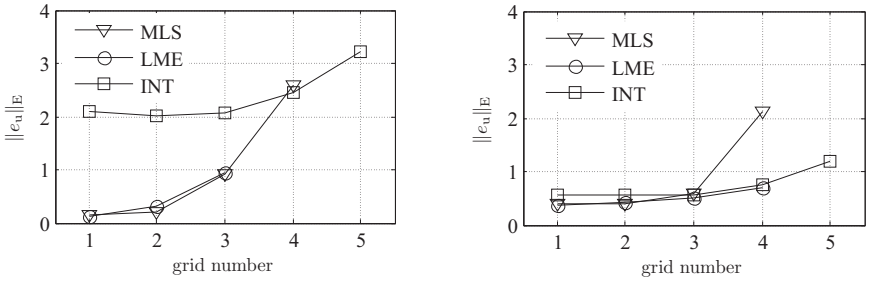


Figure 3.8: The model to examine distortional effects.

Figure 3.9: The grids used to examine the influence of distortion.

3.3.4 Tapered Bar Analysis

In this section the performance of the shape functions and integration schemes is investigated in an elasto-plastic analysis. Two potential problems can be envisaged for a numerical method in plasticity. Firstly there is the problem of volumetric locking. If a numerical scheme suffers from volumetric locking, the response will



(a) STD Gaussian integration (b) SCNI integration

Figure 3.10: The energy error norm with increasing distortion.

become nonphysically stiff in plastic deformation and the pressure associated with this deformation will be overestimated largely. Secondly, the transition from elastic to plastic will lead to a locally high gradient in the strain field. This interface should be distinct and not nonphysically diffuse as one might expect with diffuse meshless approximations.

Figure 3.11(a) shows the model used for the analysis. The tapered bar is loaded at the right-hand side with a prescribed traction. The problem is simplified by using a plane strain assumption and symmetry along the centre line of the bar. A von Mises yield criterion is used in combination with linear hardening:

$$\sigma_{\mathrm{f}} = \sigma_0 + C\varepsilon_{\mathrm{eq}} \tag{3.51}$$

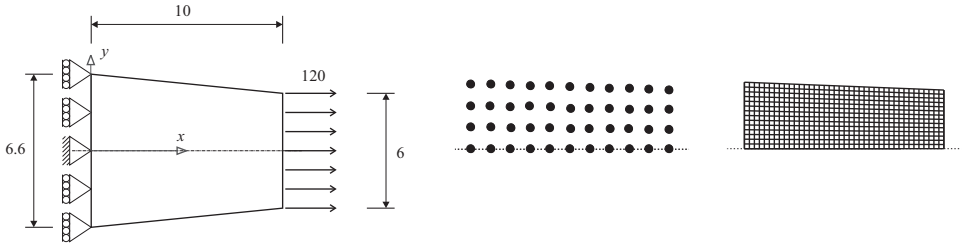
Variables σ_{f} and $\varepsilon_{\mathrm{eq}}$ are the flow stress and equivalent plastic strain respectively. The constants for the hardening law and the elastic part of the deformation are given in Table 3.1. After applying the total load of 120 MPa on the bar, the right-hand side of the bar will have deformed plastically, whereas the left-hand side is still in the elastic domain. Hence it should be possible to observe an elastic-plastic transition region within the bar. Because of the non-linear material response, Equation (3.5) is solved with an iterative-incremental strategy. The stress update is performed with a fully implicit return mapping algorithm.

Table 3.1: Constitutive parameters for the tapered bar problem.

elastic properties	E	210 000	MPa
	ν	0.3	-
plastic properties	C	100	MPa
	σ_0	100	MPa

For the STD integration a three-point and a one-point integration rule is used. For the SCNI integration a two-point Gauss rule is used on each of the facets of the cell.

The nodal grid for the meshless approximations is shown in Figure 3.11(b). The grid has 4 nodes over the height and 11 along the length. The results of the meshless analysis are compared to a reference finite element solution. A dense mesh of linear quadrilateral selective reduced integrated (SRI) finite elements are used to get an accurate prediction of the stress and strain fields. This mesh has 14 elements over the height and 40 along the length and is shown in Figure 3.11(c).



(a) model (b) cloud of nodes (c) quadrilateral mesh

Figure 3.11: The tapered bar problem and the models for the meshless and reference finite element computations. Dimensions are given in millimetres

Influence of Integration

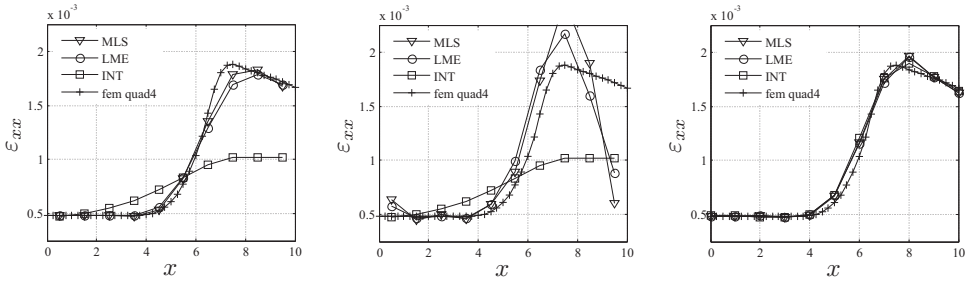
The first test is the simulation of the tapered bar problem with all shape functions and the two integration schemes. For the STD integration scheme a three-point and a one-point integration rule are used. The strain component ε_{xx} is monitored along the symmetry line ($y = 0$). Figure 3.12 shows the results for the three cases where the horizontal axis displays the x coordinate along the line of symmetry.

Figure 3.12(a) shows the results with the three-point STD integration rule. It can be seen that the plastic deformation at the right-hand side of the bar is underestimated by the linear interpolation shape function. As a result of the incompressibility of this plastic deformation, the numerical artifact of volumetric locking deteriorates the results. Since a linear triangular shape function in combination with Gaussian integration rule makes a linear triangular finite element, this poor behaviour in incompressibility is expected. The MLS and LME approximations underestimate the strain $\varepsilon_{\mathrm{xx}}$ slightly but approach the reference finite element solution quite well for the small number of nodes.

If the number of volumetric constraints is reduced by selecting a lower order integration rule, results as shown in Figure 3.12(b) are obtained. For linear interpolation (INT), selecting either a three-point integration rule, or a one-point integration rule does not affect the results. Of course, this is expected for a constant strain field within a triangle. For the LME and MLS approximations the scheme responds spuriously due to the reduced integration. With the chosen parameters to

control the domain of influence of these functions, a one-point integration rule is insufficient.

Figure 3.12(c) shows the results for the nodally integrated approximations. It can be seen that independent of the shape function used, an accurate prediction of the strain is obtained. Even the linear interpolation, known for its poor behaviour in incompressibility in the finite element method, gives good results. At the point $x = 8$ there is a minor overestimation of the strain but in general a good agreement is obtained. Furthermore, despite the use of diffuse shape functions, a distinct transition from the elastic region to the plastic region is predicted.



(a) STD integration (3 ip/triangle)

(b) STD integration (1 ip/triangle)

(c) SCNI integration

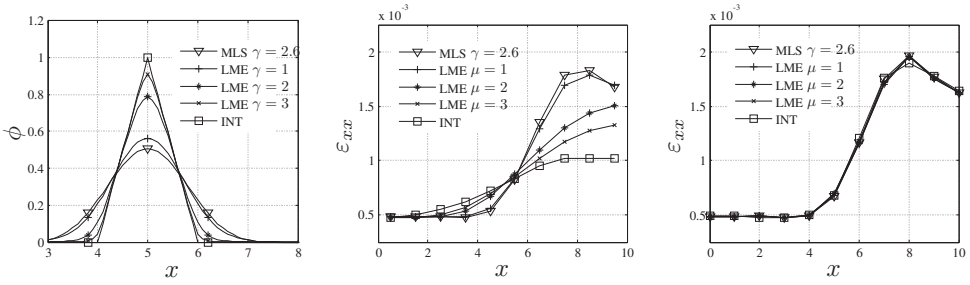
Figure 3.12: Strain in x direction along the symmetry axis.

Influence of Compactness

As shown in Figure 3.12(a), the MLS and LME functions give locking-free results whereas the INT function shows locking when integrated with a standard integration rule. The compactness of the LME shape function is controlled by parameter μ. Increasing parameter μ will make the LME functions equal to the INT functions (excluding degenerate triangulations). Decreasing parameter μ makes the LME functions similar to the MLS functions. Hence it is possible to move from the locking behaviour of the INT function to the non-locking result of the MLS function by setting μ. To examine this effect the tapered bar problem is analysed for three different settings of μ. These settings are $\mu = 1$, $\mu = 2$ and $\mu = 3$.

Figure 3.13(a) shows the shape functions of the node lying on the symmetry axis at the location $(x, y) = (5, 0)$. It can be seen that the local maximum entropy shape function changes shape from the moving least squares function to the interpolation by increasing μ. Figure 3.13 shows the result of the tapered bar analysis for the two integration schemes. Figure 3.13(b) shows that the functions can be moved out of the locking domain by decreasing the compactness. A very similar effect was observed in the case of the element-free Galerkin method by Dolbow and Belytschko [44] and by Askes *et al.* [10]. In Figure 3.13(c) the results of the same test are given but now for the nodal integration scheme. No effect of the compactness on the results is found.

There is no locking, nor is there influence of the compactness of the approximation on the distribution of the strain. The nodal integration scheme gives accurate results for all shape functions.



(a) Shape functions of the node at location $(x, y) = (5, 0)$

(b) STD integration (3 ip/triangle)

(c) SCNI integration

Figure 3.13: The strain in x direction along the symmetry axis for various settings of μ.

Full Field Results

As shown in the previous analysis, the nodal integration scheme gives an accurate prediction of the strain on the symmetry line. In the following analysis the strain is examined outside this symmetry line by presenting contour plots of the equivalent plastic strain. Furthermore, the distribution of the pressure in the domain is investigated as well. For an accurate prediction of the elastic part of the deformation, this pressure should be physical and without numerical artifacts.

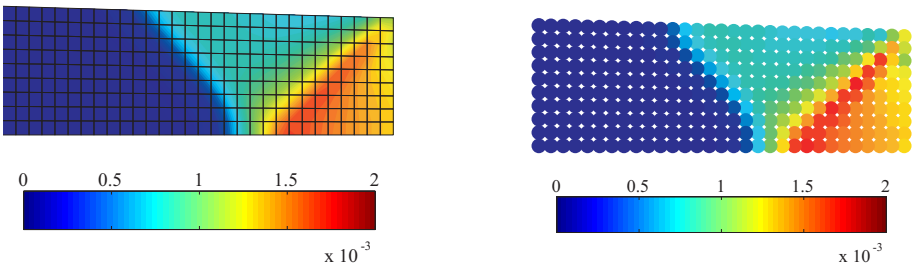
For the meshless computations, the INT function in combination with the SCNI integration scheme is used. A finite element computation is made for reference purposes. Both simulations use the same nodal grid, which has 10 nodes over the height and 31 nodes along the length, as shown in Figure 3.14.

The finite element result and the meshless result are plotted in Figure 3.14(a) and Figure 3.14(b) respectively. The equivalent plastic strain of the nodal integration scheme is plotted directly on the nodes. The integration point values of the finite element solution are presented in a contour plot without using nodal averaging. In general it can be seen that the two distributions correspond to a good extent. An accurate prediction of the equivalent plastic strain is obtained with the nodal integration scheme and the linear triangular interpolation.

Secondly, the distribution of the pressure is investigated. The results for the two simulations are shown in Figure 3.15. Again the patterns are found to be similar for the two simulations. Although the quadrilateral elements with selective reduced integration can suffer from pressure oscillations, these are not observed for this problem. The nodal integration scheme is showing very small oscillations in the

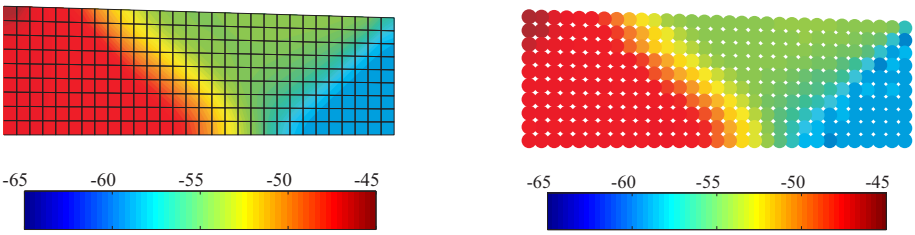
plastic regime at the right-hand side of the bar. In the following section the inf-sup test will be performed in order to examine these oscillations and the locking-free behaviour.

Finally, note that besides the pressure and the equivalent plastic strain, other quantities such as normal stresses and strains were comparing closely between both two simulations. In general it can be concluded that the results of nodal integrated interpolation scheme can be trusted in this example.



(a) FEM quad4 SRI (b) INT+SCNI

Figure 3.14: A contour plot of the equivalent plastic strain.



(a) FEM quad4 SRI (b) INT+SCNI

Figure 3.15: A contour plot of the pressure in MPa.

3.3.5 The Inf-Sup Test

For the successful application of a numerical method in incompressibility, the inf-sup test (or Ladyzhenskaya-Babuška-Brezzi (LBB) test) should be satisfied. The main formulation of this test can be found in Chapelle and Bathe [28] and Bathe [14], among others.

For Gaussian integrated solutions, inf-sup tests can be found in the literature. For instance, a Gaussian integrated triangle interpolation is known to fail the inf-sup test,

as is shown in [28]. For moving least squares approximations Dolbow and Belytschko [44] reported on monitoring the inf-sup value. The approximation appeared to be locking. Huerta *et al.* proposed a modification of the MLS function in order to satisfy the inf-sup test [65]. This method is called the pseudo divergence-free EFG method. Since the LME approximation is performing similar to the MLS function, no differences for the inf-sup test are expected.

For nodal integrated solutions locking-free responses have been reported, though they have not been confirmed by a numerical inf-sup test. In this section, the locking-free response as observed in the tapered bar problem will be examined.

General Formulations

First, two matrices $\mathbf{S}_\mathrm{p}$ and $\mathbf{S}_\mathrm{u}$ are defined, which correspond to the norm on the pressure field and the displacement field respectively:

$$\|\mathbf{u}\|^2 = \int_\Omega (\bar{\nabla}\mathbf{u})^2 \,\mathrm{d}\Omega = \{\mathbf{d}\}^\mathrm{T} [\mathbf{S}_\mathrm{u}] \{\mathbf{d}\} \tag{3.52}$$

$$\|\mathbf{p}\|^2 = \int_\Omega \mathbf{p}^2 \,\mathrm{d}\Omega = \{\mathbf{P}\}^\mathrm{T} [\mathbf{S}_\mathrm{p}] \{\mathbf{P}\} \tag{3.53}$$

where vectors $\{\mathbf{d}\}$ and $\{\mathbf{P}\}$ contain the displacement and pressure degrees of freedom respectively. A matrix $\mathbf{K}_\mathrm{up}$ is defined which expresses the volumetric energy resulting from the pressure and displacement field:

$$[\mathbf{K}_\mathrm{up}] = \int_\Omega [\bar{\mathbf{B}}]^\mathrm{T} \{\mathbf{m}\} [\mathbf{N}_\mathrm{p}] \,\mathrm{d}\Omega \tag{3.54}$$

where vector $\mathbf{N}_\mathrm{p}$ contains the shape functions for the pressure space. For the SCNI a constant pressure is defined within a cell. $\mathbf{N}_\mathrm{p}$ is therefore simply 1. Vector $\mathbf{m}$ is defined as:

$$\{\mathbf{m}\} = \{\ 1 \quad 1 \quad 0\ \}^\mathrm{T} \tag{3.55}$$

The goal of the numerical inf-sup test is to examine matrix $\mathbf{K}_\mathrm{up}$ on locking and spurious pressure oscillations. Therefore, the numerical condition as proposed by Chapelle and Bathe [28] is considered:

$$\inf_{\mathbf{W}} \sup_{\mathbf{V}} \frac{\{\mathbf{W}\}^\mathrm{T} [\mathbf{G}] \{\mathbf{V}\}}{\sqrt{\{\mathbf{W}\}^\mathrm{T} [\mathbf{G}] \{\mathbf{W}\}} \sqrt{\{\mathbf{V}\}^\mathrm{T} [\mathbf{S}_\mathrm{u}] \{\mathbf{V}\}}} = \vartheta \geq \vartheta_\mathrm{b} > 0 \tag{3.56}$$

where ϑ is the inf-sup value that should be bounded by a constant ϑ_b away from zero in order to satisfy the condition. Vectors $\mathbf{W}$ and $\mathbf{V}$ contain the nodal displacements, and matrix $\mathbf{G}$ is defined as:

$$[\mathbf{G}] = [\mathbf{K}_\mathrm{up}] [\mathbf{S}_\mathrm{p}]^{-1} [\mathbf{K}_\mathrm{up}]^\mathrm{T} \tag{3.57}$$

Parameter ϑ can be found by solving the following eigenproblem:

$$[\mathbf{G}]\{\mathbf{V}\} = \lambda [\mathbf{S}_{\mathrm{u}}]\{\mathbf{V}\} \tag{3.58}$$

and taking the square root of the smallest non-zero eigenvalue in the vector of eigenvalues $\boldsymbol{\lambda}$:

$$\vartheta = \sqrt{\lambda_k} \tag{3.59}$$

The index k of this smallest non-zero eigenvalue in vector $\boldsymbol{\lambda}$ can be used to determine the number of spurious pressure oscillations. If n_{u} and n_{p} are the numbers of degrees of freedom in the displacement and pressure vectors respectively, then the number of oscillations k_{pm} is found by evaluating:

$$k_{\mathrm{pm}} = k - (n_{\mathrm{u}} - n_{\mathrm{p}} + 1) \tag{3.60}$$

Applying Displacement Boundary Conditions

To perform the numerical inf-sup test, a set of displacement degrees of freedom has to be prescribed. However, two of the shape functions do not possess the Kronecker delta property. To avoid adding degrees of freedom in the case of the Lagrangian multipliers, the nodal displacements will be mapped on the field displacements with a strategy as proposed by Chen *et al.* [30]. The field displacements can be prescribed directly on the system by row-reduction techniques. See Section 2.5.3 for details on the transformation method.

A mapping matrix $\mathbf{R}$ is constructed by sub-matrices $\boldsymbol{\Phi}_{ij}$ which are defined as follows:

$$\mathbf{R} = \begin{bmatrix} \boldsymbol{\Phi}_{11} & \boldsymbol{\Phi}_{12} & \dots & \boldsymbol{\Phi}_{1N} \\ \boldsymbol{\Phi}_{21} & \boldsymbol{\Phi}_{22} & \dots & \boldsymbol{\Phi}_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\Phi}_{N1} & \boldsymbol{\Phi}_{N2} & \dots & \boldsymbol{\Phi}_{NN} \end{bmatrix} \quad \text{where} \quad \boldsymbol{\Phi}_{ij} = \begin{bmatrix} \phi_j(\mathbf{x}_i) & 0 \\ 0 & \phi_j(\mathbf{x}_i) \end{bmatrix} \tag{3.61}$$

The nodal displacements can be mapped to the field displacements of the nodes by the following equation:

$$\{\hat{\mathbf{d}}\} = [\mathbf{R}]\{\mathbf{d}\} \tag{3.62}$$

where vector $\{\hat{\mathbf{d}}\}$ contains the field displacements at the locations of the nodes and $\mathbf{d}$ contains the nodal displacement degrees of freedom. Matrices $\mathbf{K}_{\mathrm{up}}$ and $\mathbf{S}_{\mathrm{u}}$ are rewritten to the field degrees of freedom as follows:

$$[\hat{\mathbf{K}}_{\mathrm{up}}] = [\mathbf{R}]^{-\mathrm{T}} [\mathbf{K}_{\mathrm{up}}] \tag{3.63}$$

$$[\hat{\mathbf{S}}_{\mathrm{u}}] = [\mathbf{R}]^{-\mathrm{T}} [\mathbf{S}_{\mathrm{u}}] [\mathbf{R}]^{-1} \tag{3.64}$$

Suppressed displacements can be directly prescribed by row-reduction techniques on matrices $\hat{\mathbf{K}}_{\mathrm{up}}$ and $\hat{\mathbf{S}}_{\mathrm{u}}$.

Numerical Results

Figure 3.16 shows the problem as proposed by Chapelle and Bathe [28]. Figure 3.17 shows the nodal grids as used for the test. Both regular grids and irregular grids of various densities are used.

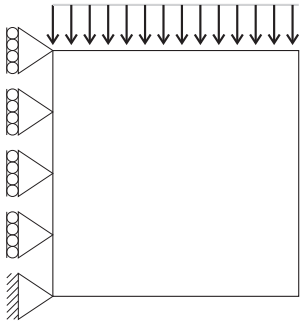


Figure 3.16: Model for the *inf-sup* test.

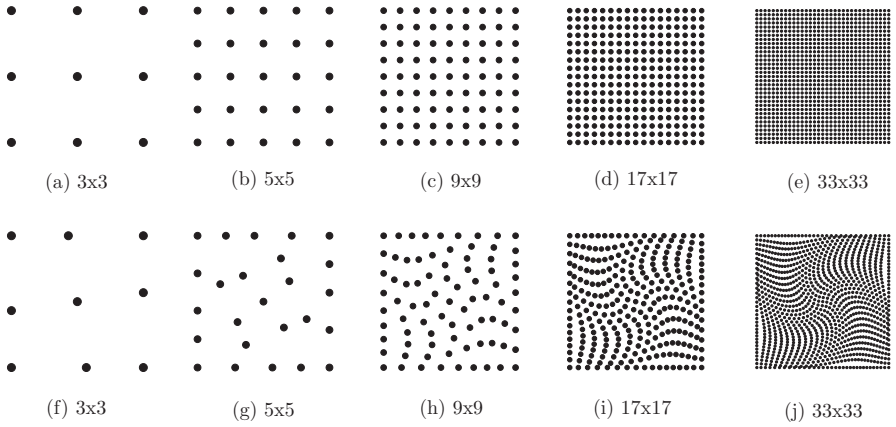
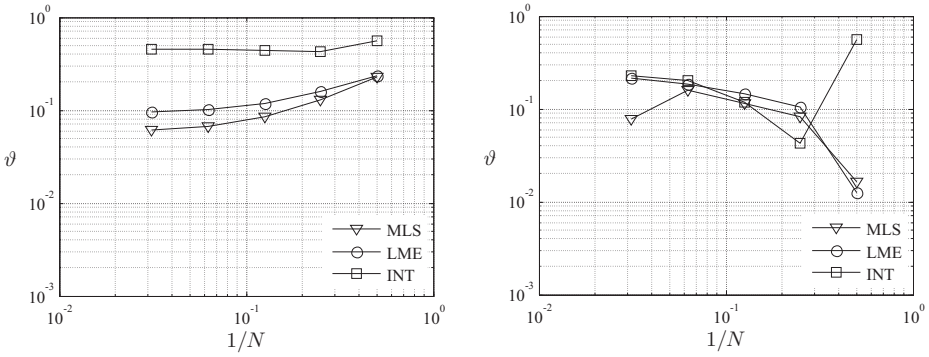


Figure 3.17: Nodes sets for the *inf-sup* test. Figures (a) to (e) show the regular grids. Figures (f) to (j) show the irregular grids.

The results of the *inf-sup* test on the regular grid are given in Figure 3.18(a). Similarly, the results of the irregular grid are given in Figure 3.18(b). It can be seen that for the regular grid as well as the irregular grid the inf-sup test is passed. The ϑ-parameter remains bounded and does not decrease on nodal grids with increasing density. Based on this test it can be concluded that matrix $\mathbf{K}_{\text{up}}$ does not over-constrain the system of equations. Volumetric locking is likely to be absent based on the results of this test.

Although locking appears to be not an issue for the SCNI integration, spurious oscillations are of concern. For all the regular grids, matrix $\mathbf{K}_{\text{up}}$ is rank-deficient. For the diffuse approximations, one zero eigenvalue is detected. The same holds for the triangular interpolation, although here two eigenvalues are zero. Figure 3.19 shows one of the detected checkerboard patterns in the pressure field for the INT function. For the irregular grid, no rank deficiency was detected of matrix $\mathbf{K}_{\text{up}}$. Nevertheless, for instance the same oscillation as shown in Figure 3.19 is present though its corresponding eigenvalue is not zero. To conclude, matrix $\mathbf{K}_{\text{up}}$ appears not to be over-constraining the system, but stable pressure fields cannot be guaranteed.



(a) regular grid, SCNI integration

(b) irregular grid, SCNI integration

Figure 3.18: ϑ - parameter plots.

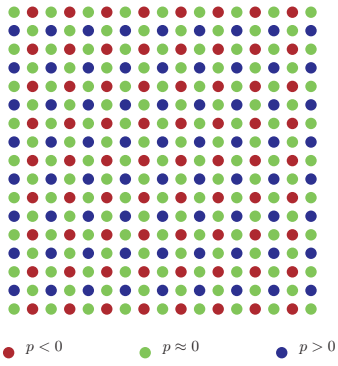


Figure 3.19: A pressure oscillation as detected on the regular grid.

3.3.6 Computational Efficiency

In the following test, the computational efficiency of the components as shown in Figure 3.3 is investigated. To examine this efficiency, two aspects of the code will be monitored. The first aspect is the time to build the stiffness matrix and the second is the required memory allocation for this matrix. The plate with a hole problem with the cloud of nodes as shown in Figure 3.6(b) is used for this test.

The time to build the stiffness matrix includes the computations of the shape functions, setting-up of the integration rule and assembly of all local stiffness matrices in the global stiffness matrix. These local stiffness matrices are constructed per cell or per triangle depending on whether the nodal integration scheme or the Gaussian scheme is selected respectively. The time does not include the neighbour search as it is required for the MLS and LME shape functions, nor is the time for the construction of the Lagrangian multipliers or performing the triangulation or tessellation included. These components were found to be of small influence on the total computation time for this set of nodes. The meshless code is programmed in MATLAB and the simulations are performed on a dual-core 2.3 GigaHertz computer. Although computational times of a computer code are depending on many factors, for instance the skill of the programmer, the trends of the analysis are expected to be representative.

Table 3.2 shows the results for the shape functions and integration schemes in seconds. Several trends can be observed in the table. Firstly, it can be seen that the MLS and LME shape function use considerably more computing time than the simple INT approximation. This could be expected since the INT function requires only a few computations in order to obtain ϕ and the local stiffness matrices of this function are considerably smaller in size. Therefore fewer multiplications are performed in the assembly procedure. Secondly, the SCNI integration scheme is computationally more demanding than the STD integration. For both integration schemes, constructing the tessellation and the triangulation is left out of the analysis. Therefore it can be concluded that the assembly procedure and the construction of the assumed gradient makes the SCNI integration computationally more demanding. The extra computational effort is of little importance for the diffuse approximations. For the INT approximation, however, the type of integration is of relatively more influence on the computation time.

Table 3.2: The time in seconds for constructing **K** for the plate with hole problem .

	MLS	LME	INT
STD	4.589	4.124	0.307
SCNI	7.268	6.099	0.909

Table 3.3 shows the number of non-zero elements in the stiffness matrix for the shape functions and the integration schemes. The total size of the stiffness matrix **K** is 496 by 496. The stiffness matrix does not include the Lagrangian multiplier matrices **G**

and furthermore the stiffness matrix **K** is defined as symmetric and sparse. The table shows that the memory allocation for the INT shape function is less than for the diffuse shape functions. The diffuse character of the LME and MLS approximations has a big influence on the sparseness of the stiffness matrix. Furthermore the number of non-zero elements in **K** increases by employing the stabilised nodal integration. As a result of the compact character of the INT shape function this effect is relatively more pronounced than for the other two shape functions. This will also influence the solution time of the system of equations, especially if the models are large.

Table 3.3: The number of non-zeros in **K** for the plate with hole problem.

	MLS	LME	INT
STD	30028	26476	2802
SCNI	49064	53500	9299

Two remarks have to be made on the results of the timings. Solving the system of equations, the triangulations and the neighbour search routine were found to have a negligible influence on the total computation time. These parts of the algorithm scale more than linear with the number of nodes and therefore their relative contribution to the total computation time will increase compared to the building time of the stiffness matrix. Secondly, in the case of geometrical or material non-linearity, shape functions are required per iteration to build the stiffness matrix and the internal force vector. By storing the shape functions and 're-using' them over the increment, the relative cost of the computationally more demanding shape functions can be reduced. For the MLS and LME functions this can be a time-reducing approach. The difference between the linear interpolation and the MLS and LME approximations as shown in Table 3.2 is expected to be smaller in that case. A drawback of this implementation strategy is that memory has to be allocated to store the gradients of the shape functions.

3.4 Conclusions

In this chapter a numerical analysis was performed on three meshless approximations and two integration schemes. The performance in linear elasticity and in elasto-plasticity was investigated.

It was shown that diffuse shape functions, like the moving least squares function and the local maximum entropy function, offer better accuracy when compared to the linear triangle interpolation in elasticity. The error reduced approximately by a factor of two to three when using the MLS or LME approximation instead of a linear triangle interpolation for the same number of nodes. However, the computational effort for these two diffuse approximations is higher. The stiffness matrix becomes less sparse and the time for building this matrix is higher. The local maximum entropy approximation and the moving least squares approximation were found to perform very similarly if used with the same domain of influence. Furthermore it

was observed that the SCNI integration scheme is less sensitive to distortion than the Gaussian integration scheme.

A test in elasto-plasticity showed that using a Gaussian integration rule gives results that depend strongly on the order of the integration rule and the compactness of the diffuse approximations. Spurious deformation modes as well as volumetric locking can be present, depending on the choices in the integration rule or the compactness of the shape function. The SCNI integration scheme, on the contrary, gives excellent results when compared to a Gaussian integration rule. No trace of volumetric locking is observed. The meshless solutions match the reference solution accurately and the results were found to be nearly independent of the type of shape function used. Moreover, the compactness of the shape function has a negligible influence on the results. To give a more fundamental background of the observed locking-free response of the SCNI integration, the numerical inf-sup test was performed. The SCNI integration appears locking-free, however spurious oscillations were detected.

To conclude, it can be stated that the SCNI integration is the most sensible option for use in combination with incompressible material models, leaving the choice for the shape function open. Simply using a Gaussian integration scheme proves to be not optimal for use in incompressibility. For these cases, the extra computational effort for the SCNI integration is easily justified. For a robust and fast simulation, the method of SCNI in combination with triangle interpolation (INT) is shown to be a very efficient computational scheme.

4

A Simple Enriched Nodal Averaging Strategy

4.1 Introduction

As seen in the previous chapter, the stabilised conforming nodal integration technique has beneficial properties for the simulation of large deformations. The numerical artifact of volumetric locking is absent and the method is relatively insensitive to distortions. Moreover, as a result of the nodal integration of the weak form, data related to the material history can be elegantly stored at the nodes.

The essence of the stabilised conforming nodal integration scheme is a nodal averaging technique used to build an assumed strain field. This strain field is used thereafter in the evaluation of the constitutive behaviour. The method of nodal averaging has been applied to the finite element method as well as meshless methods. Both will be shortly reviewed below. Firstly the developments on the finite element side are discussed and thereafter the developments in the meshless field are summarised.

The first development in the field of nodal averaging is the introduction of an averaged nodal pressure tetrahedral by Bonet and Burton [20]. The starting point of their derivations is a linear four-node tetrahedral finite element. A nodal averaging operation is applied to the volumetric part of the deformation whereas the deviatoric part of the deformation remains unmodified. The reason for this split of the deformation is to avoid the problem of volumetric locking, which is of concern for these elements. Afterwards, the method has been extended for the simulation of large deformations in combination with explicit time integration by Bonet *et al.* [23]. For implicit time integration, formulations are presented by Pires *et al.* [8]. All three articles present locking-free results. An observed drawback, however, is that the pressure prediction can be nonphysically oscillatory.

The developments as given above, employ the nodal averaging operation only on the volumetric part of the deformation. Similarly, schemes have been developed which

average the complete deformation. This type of nodal averaging was proposed by Dohrmann *et al.* [42]. Their scheme applies nodal strain averaging both on triangular and quadrilateral finite elements. Thereafter the nodal averaging strategy has been applied to various types of other finite elements. For instance to linear tetrahedral elements by Puso and Solberg [102] or to a wide selection of linear and quadratic elements by Krysl and Zhu [74]. The stability and accuracy of nodally averaged finite elements has been studied and improved by for instance Liu *et al.* [84], Hung *et al.* [68], Puso and Solberg [102], Broccardo *et al.* [26], Puso *et al.* [101], Gee *et al.* [53], and Nguyen-Xuan [97]. An application of nodal averaged FEM in visco-elastoplasticity is presented by Nguyen-Thoi *et al.* [96]. A study of strain smoothing for finite elements and extended finite elements can be found in Bordas *et al.* [24].

The first meshless method employing a nodal averaging strategy was introduced by Chen *et al.* [33]. Their method is entitled the stabilised conforming nodal integration scheme (SCNI) and was investigated extensively in the preceding chapter. An extension of this method to geometrical non-linearity is proposed by Chen *et al.* [34]. Afterwards developments have taken place in the natural element method by González *et al.* [56] and by Yoo *et al.* [129]. Research on developing a variational form for the strain averaging operator was done by Sze *et al.* [118].

To summarise, developments in nodal averaging have taken place for finite elements as well as for meshless methods. Besides the advantages of employing a nodal averaging strategy, there are also two drawbacks. The first drawback is the low accuracy, especially of the displacement field, as shown in Chapter 3. The second drawback is the lack of stability. Stress fields, for instance, can be spurious under certain circumstances. In this chapter an enriched nodal averaging scheme is proposed which overcomes these issues.

This chapter is organised as follows. Firstly, an introduction to the nodal averaging operation is given in Section 4.2. Secondly, Section 4.3 will propose an enriched nodal averaging. The formulations of the method will be explained and the aspects required for implementation are given afterwards. Section 4.4 presents the results of three tests examining the proposed method. Finally, the conclusion and discussion are given in Section 4.5.

4.2 General Formulations

The starting point for the following derivations is the weak formulation of equilibrium as given in Equation (3.4). To restate this equation in tensor form:

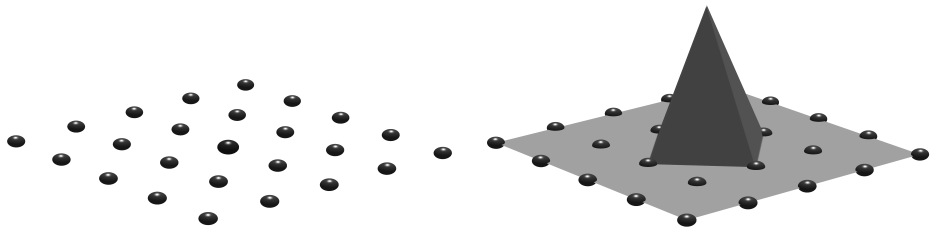
$$\int_{\Omega} \delta\boldsymbol{\varepsilon} : \boldsymbol{\sigma} \, \mathrm{d}\Omega - \int_{\Omega} \delta\mathbf{u} \cdot \mathbf{f} \, \mathrm{d}\Omega - \int_{\Gamma_t} \delta\mathbf{u} \cdot \tilde{\mathbf{t}} \, \mathrm{d}\Gamma = 0 \qquad \forall \, \delta\mathbf{u} \tag{4.1}$$

In order to present the new methodology more clearly, the deformations of the body are assumed to be small. Moreover, for the constitutive model, a linear relation between the small strain tensor and the Cauchy stress is assumed ($\boldsymbol{\sigma} = \mathbf{C} : \boldsymbol{\varepsilon}$), and the change of the geometry resulting from the deformation is neglected.

For the current study, the displacement field $\mathbf{u}$ will be constructed by triangulating a cloud of nodes by means of a Delaunay triangulation and defining a linear shape function within each triangle. A formulation of this shape function can be found in Section 3.2.2. The displacement field and the virtual displacement field will be approximated as follows:

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x})\mathbf{d}_i \qquad \text{and} \qquad \delta\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x})\delta\mathbf{d}_i \tag{4.2}$$

See Figure 4.1 for a visualisation of the linear interpolation based upon the Delaunay triangulation. The shape function of the node at the centre of the grid is plotted. Note that the method presented in this chapter is applicable to any shape function and is not restricted to the linear triangle interpolation. Diffuse approximations, such as the moving least squares function or the local maximum entropy function as given in Section 3.2.2 and 3.2.2, can be used in combination with the methodology as presented in this chapter.



(a) a grid of nodes (b) shape function $\phi(\mathbf{x})$ for the centre node

Figure 4.1: Linear interpolation based on a Delaunay triangulation.

4.2.1 Classical Compatible Strain

After defining shape functions ϕ, it is possible to discretise Equation (4.1). Matrices $\mathbf{N}$ and $\mathbf{B}$ are constructed, which relate the field displacements and strains to the nodal displacement vector $\mathbf{d}$. The strain and the virtual strain in the body are defined by the gradient of the displacement field:

$$\boldsymbol{\varepsilon}(\mathbf{x}) = \frac{1}{2}\left(\overrightarrow{\nabla}\mathbf{u} + \mathbf{u}\overleftarrow{\nabla}\right) \qquad \text{and} \qquad \delta\boldsymbol{\varepsilon}(\mathbf{x}) = \frac{1}{2}\left(\overrightarrow{\nabla}\delta\mathbf{u} + \delta\mathbf{u}\overleftarrow{\nabla}\right) \tag{4.3}$$

The matrices in Voigt form become:

$$\{\mathbf{u}\} = [\mathbf{N}]\{\mathbf{d}\} \qquad \{\delta\mathbf{u}\} = [\mathbf{N}]\{\delta\mathbf{d}\} \tag{4.4}$$

$$\{\boldsymbol{\varepsilon}\} = [\mathbf{B}]\{\mathbf{d}\} \qquad \{\delta\boldsymbol{\varepsilon}\} = [\mathbf{B}]\{\delta\mathbf{d}\} \tag{4.5}$$

where matrices $\mathbf{N}$ and $\mathbf{B}$ contain the function values and gradients of shape function ϕ respectively. Writing Equation (4.1) in Voigt form, and requiring it to hold for all virtual nodal displacements, gives:

$$\int_\Omega [\mathbf{B}]^\mathrm{T} [\mathbf{C}] [\mathbf{B}] \, \mathrm{d}\Omega \, \{\mathbf{d}\} = \int_{\Gamma_\mathrm{t}} [\mathbf{N}]^\mathrm{T} \{\tilde{\mathbf{t}}\} \, \mathrm{d}\Gamma + \int_\Omega [\mathbf{N}]^\mathrm{T} \{\mathbf{f}\} \, \mathrm{d}\Omega \tag{4.6}$$

4.2.2 Nodally Averaged Strain

The method of nodal averaging constructs an assumed strain field which is different from the symmetric gradient of the displacement field. This assumed strain field is constructed as follows:

$$\bar{\boldsymbol{\varepsilon}}(\mathbf{x}) = \frac{1}{\Omega_i} \int_{\Omega_i} \boldsymbol{\varepsilon}(\boldsymbol{\xi}) \, \mathrm{d}\Omega \qquad \forall \, \mathbf{x} \in \Omega_i \qquad \text{for } i = 1...N_\mathrm{nod} \tag{4.7}$$

where $\bar{\boldsymbol{\varepsilon}}$ is the nodally averaged strain field, $\boldsymbol{\xi}$ is a coordinate used for integration and Ω_i is a set containing all points in the cell corresponding to node i. Figure 4.2 shows an illustration of the used cells Ω_i. Moreover, a triangulation is shown on which the shape functions are defined.

The approximation spaces in Voigt form become:

$$\{\bar{\boldsymbol{\varepsilon}}\} = \left[\bar{\mathbf{B}}\right] \{\mathbf{d}\} \qquad \{\delta\bar{\boldsymbol{\varepsilon}}\} = \left[\bar{\mathbf{B}}\right] \{\delta\mathbf{d}\} \tag{4.8}$$

where $\bar{\mathbf{B}}$ is the strain-displacement matrix of the smoothed strain field, and $\delta\bar{\boldsymbol{\varepsilon}}$ is the virtual smoothed strain field. The $\mathbf{B}$-matrix is constructed according to Equation (4.7) as follows:

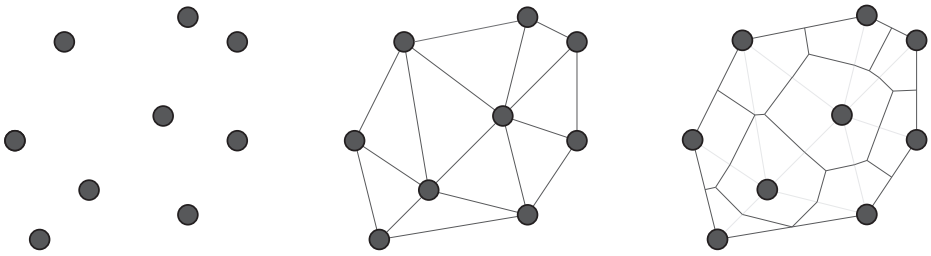
$$\bar{\mathbf{B}}(\mathbf{x}) = \frac{1}{\Omega_i} \int_{\Omega_i} \mathbf{B}(\boldsymbol{\xi}) \, \mathrm{d}\Omega \qquad \forall \, \mathbf{x} \in \Omega_i \qquad \text{for } i = 1...N_\mathrm{nod} \tag{4.9}$$

where matrix $\mathbf{B}$ is the compatible strain-displacement matrix as given in Section 4.2.1. Finally, Equation (4.1) is modified as follows:

$$\int_\Omega \left[\bar{\mathbf{B}}\right]^\mathrm{T} [\mathbf{C}] \left[\bar{\mathbf{B}}\right] \, \mathrm{d}\Omega \, \{\mathbf{d}\} = \int_{\Gamma_\mathrm{t}} [\mathbf{N}]^\mathrm{T} \{\tilde{\mathbf{t}}\} \, \mathrm{d}\Gamma + \int_\Omega [\mathbf{N}]^\mathrm{T} \{\mathbf{f}\} \, \mathrm{d}\Omega \tag{4.10}$$

As mentioned previously, the nodally smoothed strain field has some interesting features. However, there are also two drawbacks which will be addressed in brief below.

The first concern when using the nodal averaged strain field is that, irrespective of the used shape function, its accuracy is of the same order as for a standard Gaussian integrated scheme. A linear triangle finite element is as accurate as its nodally averaged counterpart in the displacement error norm. See Section 3.3.2 for this comparison. Linear triangle finite elements are known for their inaccuracy and are therefore not widely used in engineering practice.



(a) a grid of nodes

(b) a set of triangles used to construct $\mathbf{N}$

(c) a set of cells used to construct $\bar{\mathbf{B}}$

Figure 4.2: The two geometrical concepts used in the method of smoothed finite elements.

The second and main drawback of the nodally averaged strain field is the lack of stability of the resulting weak form. This stability problem was firstly discussed by Puso and Solberg [102]. For regular grids extending to infinity, or infinitely fine regular grids, the weak form is not elliptic. Their main derivations will be discussed in brief below. Firstly, the bilinear form of the problem is defined:

$$a_{\mathrm{h}}(\mathbf{u},\mathbf{u}) = \int_{\Omega} \bar{\boldsymbol{\varepsilon}}(\mathbf{u}) : \mathbf{C} : \bar{\boldsymbol{\varepsilon}}(\mathbf{u}) \, \mathrm{d}\Omega \tag{4.11}$$

where a_{h} gives twice the energy in the body Ω for a displacement field $\mathbf{u}$. Subscript h denotes the dependency of the bilinear form to the nodal spacing h. The magnitude of the displacement field is measured by the first order Sobolev norm which is defined in tensor form as:

$$\|\mathbf{u}\|_1^2 = \int_{\Omega} (\mathbf{u}\cdot\mathbf{u} + \overrightarrow{\nabla}\mathbf{u} : \overrightarrow{\nabla}\mathbf{u}) \, \mathrm{d}\Omega \tag{4.12}$$

Let $\mathbf{V}_{\mathrm{h}}$ be a space containing all displacement fields $\mathbf{u}$ for which this Sobolev norm exists, but excluding all rigid body modes. The ellipticity condition is given as:

$$\inf_{\mathbf{u}\in\mathbf{V}_{\mathrm{h}}} \frac{a_{\mathrm{h}}(\mathbf{u},\mathbf{u})}{\|\mathbf{u}\|_1} > 0 \tag{4.13}$$

This condition states that the energy in the body, divided by the norm, is not allowed to be smaller than or equal to zero for non-zero displacement fields $\mathbf{u}$. As was shown by Puso and Solberg the ellipticity condition is violated for regular grids with increasing density. In formula form, the condition is violated as follows:

$$\lim_{h\to 0} \inf_{\mathbf{u}\in\mathbf{V}_{\mathrm{h}}} \frac{a_{\mathrm{h}}(\mathbf{u},\mathbf{u})}{\|\mathbf{u}\|_1} = 0 \tag{4.14}$$

For an infinitely fine regular mesh, ellipticity does not hold, and for finite meshes ellipticity holds only very weakly. Figure 4.3 gives an illustration for the 1D case in

which the displacement field $\mathbf{u}$ makes the bilinear form $a_{\mathrm{h}}(\mathbf{u}, \mathbf{u})$ zero. Although the displacement field and the compatible strain field are larger than zero, the assumed strain field is zero. As a result, the stiffness matrix of the system is rank deficient. This is nonphysical and a method has to be employed in order to counter this numerical artifact.

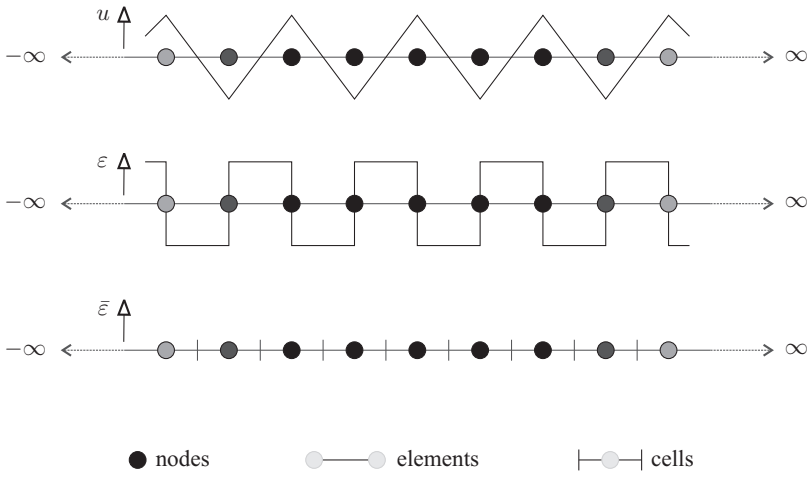
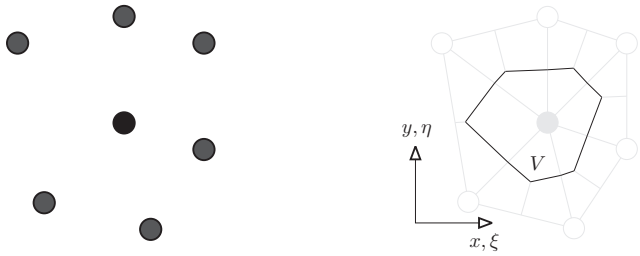


Figure 4.3: An illustration of the instability of the nodal averaging.

4.3 Enriched Nodal Averaging

In this section, the formulations of the enriched nodal averaging strategy will be presented. The essence of this method is a least squares fit between the compatible strain field and the assumed strain field. Instead of directly explaining the enriched nodal averaging scheme, firstly this least squares fit is explained by using it to derive the formulations of the 'standard' nodal averaging as given in the preceding section. Section 4.3.1 will present these derivations. Subsequently, the enriched nodal averaging is presented in Section 4.3.2. Section 4.3.2 will give an illustrative example on the method for enrichments with increasing complexity.

To enhance readability, all further derivations are made for one single cell. After the formulations for one cell are found, the descriptions are re-cast in a form which is applicable to the whole assumed strain field. Figure 4.4 shows a cloud of nodes and the cell under consideration which will be used for further derivations. The volume of a cell is given by V. For the following derivations, coordinate systems $\mathbf{x}$ and $\boldsymbol{\xi}$, which have an equal orientation and magnitude, will be used.



(a) set of nodes (b) nodal averaging cell

Figure 4.4: An illustration of a single nodal averaging cell used for further derivations.

4.3.1 Constant Cell Strain

In this paragraph, the well known formula for nodal averaging will be derived by using a least squares fit. As in the preceding sections, the compatible strain field is given by $\boldsymbol{\varepsilon}$ and the assumed strain field is given by $\bar{\boldsymbol{\varepsilon}}$. This latter field is supposed to be unknown beforehand and an explicit formulation for the field as given in Equation (4.7) is not available. Leaving the choice for this field open, it is still possible to define a least squares fit between $\boldsymbol{\varepsilon}$ and $\bar{\boldsymbol{\varepsilon}}$:

$$\Pi_{\text{ls}} = \int_V \left(\boldsymbol{\varepsilon}(\boldsymbol{\xi}) - \bar{\boldsymbol{\varepsilon}}(\boldsymbol{\xi})\right)^2 \, \mathrm{d}V \tag{4.15}$$

where $\boldsymbol{\xi}$ is the local coordinate used for integration purposes, V is the volume of the cell as shown in Figure 4.4(b) and Π_{ls} is a potential expressing the error between the two strain fields. In Equation (4.15), the choice for the assumed strain field $\bar{\boldsymbol{\varepsilon}}$ is left open. The most simple choice is to assume this field constant within the cell V:

$$\bar{\boldsymbol{\varepsilon}} = 1 \cdot \boldsymbol{\varepsilon}_0 \tag{4.16}$$

where $\boldsymbol{\varepsilon}_0$ is a set of coefficients corresponding to the constant term of the polynomial expansion. Note that $\boldsymbol{\varepsilon}_0$ is a second order tensor but is not a function of the coordinate vector $\boldsymbol{\xi}$. The least squares fit between the compatible strain field and the assumed strain field can now be expressed as:

$$\Pi_{\text{ls}} = \int_V \left(\boldsymbol{\varepsilon}(\boldsymbol{\xi}) - \boldsymbol{\varepsilon}_0\right)^2 \, \mathrm{d}V \tag{4.17}$$

This potential is minimised with respect to variables $\boldsymbol{\varepsilon}_0$:

$$\min_{\boldsymbol{\varepsilon}_o} \Pi_{\text{ls}} \tag{4.18}$$

The most simple way to find this minimum is to require the derivatives of the potential to be zero:

$$\frac{\partial \Pi_{\text{ls}}}{\partial \boldsymbol{\varepsilon}_0} = -2 \int_V \left(\boldsymbol{\varepsilon}(\boldsymbol{\xi}) - \boldsymbol{\varepsilon}_0 \right) \, \mathrm{d}V = 0 \tag{4.19}$$

Rearranging this result gives the well-known form for nodal averaging:

$$\int_V \boldsymbol{\varepsilon}_0 \, \mathrm{d}V = \int_V \boldsymbol{\varepsilon}(\boldsymbol{\xi}) \, \mathrm{d}V \tag{4.20}$$

$$\boldsymbol{\varepsilon}_0 = \frac{1}{V} \int_V \boldsymbol{\varepsilon}(\boldsymbol{\xi}) \, \mathrm{d}V \tag{4.21}$$

This equation holds for a single arbitrary cell in the domain. In order to get an expression for the complete field, Equation (4.21) is rewritten in a general form by considering $V = \Omega_i$. The expression for the complete assumed strain field is given by:

$$\bar{\boldsymbol{\varepsilon}}(\mathbf{x}) = \frac{1}{\Omega_i} \int_{\Omega_i} \boldsymbol{\varepsilon}(\boldsymbol{\xi}) \, \mathrm{d}\Omega \qquad \forall \, \mathbf{x} \in \Omega_i \qquad \text{for } i = 1...N_{\text{nod}} \tag{4.22}$$

To summarise, the definition of the assumed strain field as given in Equation (4.7) can be found by using a least squares fit between the assumed strain field and the compatible strain field.

4.3.2 Linear Cell Strain

In the derivation given above, only a constant term was included in the least squares fit. The onset of the loss of ellipticity is a result of the choice for this simple polynomial basis. Therefore, in the following derivation the nodal strain is expanded by including linear terms in the assumed strain field. The start of the derivations is the following assumed strain field:

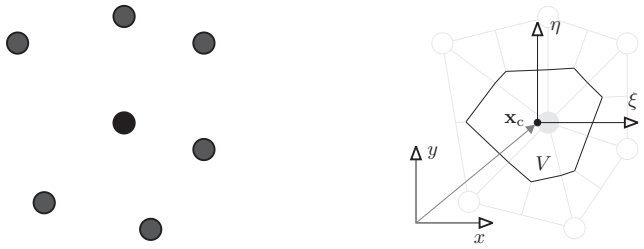
$$\bar{\boldsymbol{\varepsilon}}(\boldsymbol{\xi}) = \boldsymbol{\varepsilon}_0 + \boldsymbol{\varepsilon}_\xi \xi + \boldsymbol{\varepsilon}_\eta \eta \tag{4.23}$$

where $\boldsymbol{\varepsilon}_\xi$ and $\boldsymbol{\varepsilon}_\eta$ are the coefficients of the polynomial expansion in the direction of ξ and η respectively. For further derivations, location vector $\boldsymbol{\xi}$ is defined with its origin in the centre of gravity of the cell V. As a result, the coefficients $\boldsymbol{\varepsilon}_0$, $\boldsymbol{\varepsilon}_\xi$ and $\boldsymbol{\varepsilon}_\eta$ can be fitted to the compatible strain field $\boldsymbol{\varepsilon}$ per coefficient separately. Figure 4.5 gives an illustration of the location vectors used. The centre of gravity of cell V is given by $\mathbf{x}_{\text{c}}$.

Similarly as in Equation (4.17), a least squares potential is constructed which gives a magnitude of error between the two strain fields:

$$\Pi_{\text{ls}} = \int_V \left(\boldsymbol{\varepsilon}(\mathbf{x}) - \bar{\boldsymbol{\varepsilon}}(\boldsymbol{\xi}) \right)^2 \, \mathrm{d}V \tag{4.24}$$

$$= \int_V \left(\boldsymbol{\varepsilon}(\mathbf{x}) - \boldsymbol{\varepsilon}_0 - \boldsymbol{\varepsilon}_\xi \xi - \boldsymbol{\varepsilon}_\eta \eta \right)^2 \, \mathrm{d}V \tag{4.25}$$



(a) set of nodes (b) nodal averaging cell

Figure 4.5: An illustration of a single nodal averaging cell with shifted axis.

The goal is to minimise this potential, now for the three coefficients $\boldsymbol{\varepsilon}_0$, $\boldsymbol{\varepsilon}_\xi$ and $\boldsymbol{\varepsilon}_\eta$:

$$\min_{\boldsymbol{\varepsilon}_o, \boldsymbol{\varepsilon}_\xi, \boldsymbol{\varepsilon}_\eta} \Pi_{\text{ls}} \tag{4.26}$$

The choice for coordinate system $\boldsymbol{\xi}$ allows this minimisation to be performed per coefficient independently. The cross terms in the least squares fit are zero because of the orthogonal polynomials in V:

$$\int_V \boldsymbol{\varepsilon}_0 \cdot \boldsymbol{\varepsilon}_\xi \xi \, \mathrm{d}V = \int_V \boldsymbol{\varepsilon}_0 \cdot \boldsymbol{\varepsilon}_\eta \eta \, \mathrm{d}V = \int_V \boldsymbol{\varepsilon}_\xi \xi \cdot \boldsymbol{\varepsilon}_\eta \eta \, \mathrm{d}V = 0 \tag{4.27}$$

As a result, the following set of equations is obtained:

$$\frac{\partial \Pi_{\text{ls}}}{\partial \boldsymbol{\varepsilon}_0} = -2 \int_V \left(\boldsymbol{\varepsilon}(\boldsymbol{\xi}) - \boldsymbol{\varepsilon}_0 \right) \, \mathrm{d}V = 0 \tag{4.28}$$

$$\frac{\partial \Pi_{\text{ls}}}{\partial \boldsymbol{\varepsilon}_\xi} = -2 \int_V \xi \left(\boldsymbol{\varepsilon}(\boldsymbol{\xi}) - \boldsymbol{\varepsilon}_\xi \xi \right) \, \mathrm{d}V = 0 \tag{4.29}$$

$$\frac{\partial \Pi_{\text{ls}}}{\partial \boldsymbol{\varepsilon}_\eta} = -2 \int_V \eta \left(\boldsymbol{\varepsilon}(\boldsymbol{\xi}) - \boldsymbol{\varepsilon}_\eta \eta \right) \, \mathrm{d}V = 0 \tag{4.30}$$

By rearranging the results, the coefficients can be found:

$$\boldsymbol{\varepsilon}_0 = \frac{1}{\int_V 1 \, \mathrm{d}V} \int_V \boldsymbol{\varepsilon}(\boldsymbol{\xi}) \, \mathrm{d}V \tag{4.31}$$

$$\boldsymbol{\varepsilon}_\xi = \frac{1}{\int_V \xi^2 \, \mathrm{d}V} \int_V \boldsymbol{\varepsilon}(\boldsymbol{\xi}) \xi \, \mathrm{d}V \tag{4.32}$$

$$\boldsymbol{\varepsilon}_\eta = \frac{1}{\int_V \eta^2 \, \mathrm{d}V} \int_V \boldsymbol{\varepsilon}(\boldsymbol{\xi}) \eta \, \mathrm{d}V \tag{4.33}$$

Since all derivations given above are only for an arbitrary single cell V, an expression is constructed for the total field as follows:

$$\bar{\boldsymbol{\varepsilon}}(\mathbf{x}) = \boldsymbol{\varepsilon}_0^i + \boldsymbol{\varepsilon}_\xi^i \xi_i + \boldsymbol{\varepsilon}_\eta^i \eta_i \qquad \forall \, \mathbf{x} \in \Omega_i \qquad \text{for } i = 1 ... N_{\text{nod}} \tag{4.34}$$

where $\boldsymbol{\varepsilon}_0^i$, $\boldsymbol{\varepsilon}_\xi^i$ and $\boldsymbol{\varepsilon}_\eta^i$ are the coefficients of cell i. The local coordinate axes for this cell are given by ξ_i and η_i.

Figure 4.6 shows the enriched nodal strain field for the unstable spurious mode as given before in Figure 4.3. It can be seen that the block pattern of the spurious strain field $\boldsymbol{\varepsilon}$ is captured in the assumed strain field $\boldsymbol{\varepsilon}_\xi$. To conclude, it is possible to construct a linear strain in a cell, and to find expressions with which the linear terms can be computed.

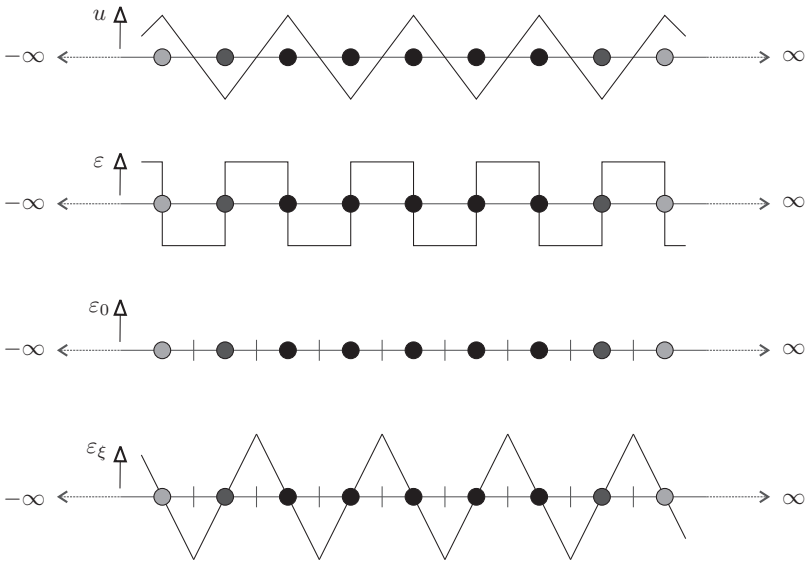


Figure 4.6: An illustration of the enriched cell strain for the spurious mode.

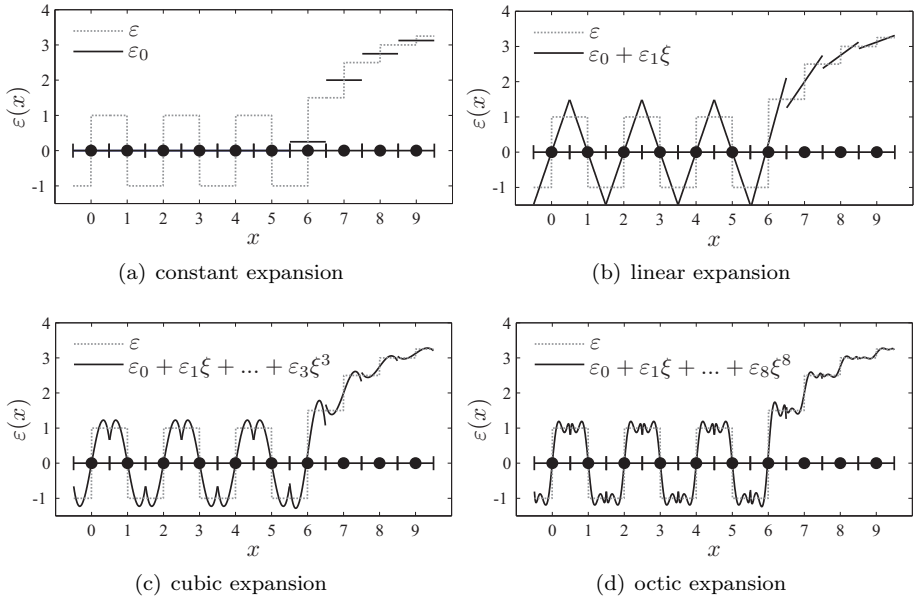
4.3.3 Higher Order Cell Strains

As shown in the section above, it is possible to define a linear strain in a cell which can be used to stabilise the weak form. For further developments presented in this chapter, this linear expansion will prove efficient in suppressing the spurious mode and in improving the accuracy. However, for academic purposes, it is interesting to examine the case in which the order of this expansion is increased. This section will investigate that case by raising the order of the polynomial basis beyond a linear basis.

Assume a predefined strain field $\boldsymbol{\varepsilon}$ as shown in Figure 4.7(a). At the left-hand side of the graph ($0 < x < 5$), this strain field is defined as a block pattern. It is the pattern on which the ellipticity condition is violated for the standard nodal averaging. At the right-hand side of the graph ($5 < x < 10$) a parabolic part is prescribed. Assumed strains with increasing polynomial order will be fitted to this compatible strain field.

The cell strains are shown in Figure 4.7 for various polynomial bases. Figure 4.7(a) shows the standard nodal averaged strain field as defined in Equation (4.22). Figure 4.7(b) shows the result for one linear term (Equation (4.34)). The oscillatory left-hand side of the strain field is captured in the assumed strain field. Figures 4.7(c) and 4.7(d) show a cubic expansion and octic expansion respectively. The compatible strain field is approximated with increasing accuracy. However, because of Runge's phenomenon, the compatible strain field cannot be retrieved exactly in the limit. Overshoots will remain at the discontinuities in the compatible strain field. Therefore the potential Π_{ls} will not be exactly zero, unless a different basis is chosen for the strain in a cell.

For the compatible strain field, it is known that the use of this field results in a stable elliptic weak form. On the contrary, the assumed strain field with only a constant enrichment is unstable, as was demonstrated in Section 4.2.2. As a consequence, it can be concluded that by increasing the polynomial order of the cell strain, it is guaranteed that a stable weak form is retrieved. Since the complexity of the method increases with each added polynomial term in the strain of a cell, only linear terms will be included for the following sections. The stability of the resulting scheme will be investigated in Section 4.4.



(a) constant expansion

(b) linear expansion

(c) cubic expansion

(d) octic expansion

Figure 4.7: Higher order polynomial expansions for cell strains.

4.3.4 System Matrices

In this section, the system matrices resulting from the linear cell strain approximation as given in Section 4.3.2 will be presented. Firstly, the total stiffness matrix is defined as the sum of the sub-stiffness matrices $[\mathbf{K}]_i$:

$$[\mathbf{K}] = \sum_{i=1}^{N_{\text{nod}}} [\mathbf{K}]_i \tag{4.35}$$

where the index i refers to node or cell i. The sub-stiffness matrix for this node is defined as:

$$[\mathbf{K}]_i = \int_{\Omega_i} \left[\bar{\mathbf{B}}\right]_i^{\mathrm{T}} [\mathbf{C}] \left[\bar{\mathbf{B}}\right]_i \, \mathrm{d}\Omega \tag{4.36}$$

where $[\mathbf{B}]_i$ is the strain-displacement matrix for cell i. This assumed strain is defined as:

$$\bar{\boldsymbol{\varepsilon}}(\mathbf{x}) = \boldsymbol{\varepsilon}_0^i + \boldsymbol{\varepsilon}_\xi^i \xi_i + \boldsymbol{\varepsilon}_\eta^i \eta_i \tag{4.37}$$

$$= ([\mathbf{B}_0]_i + [\mathbf{B}_\xi]_i \xi_i + [\mathbf{B}_\eta]_i \eta_i)\{\mathbf{d}\} \tag{4.38}$$

where matrices $[\mathbf{B}_0]_i$, $[\mathbf{B}_\xi]_i$ and $[\mathbf{B}_\eta]_i$ correspond to the strain coefficients of node i. Substituting these three matrices into Equation (4.36) and using the property that the matrices are mutually orthogonal gives the following derivation:

$$[\mathbf{K}]_i = \int_{\Omega_i} ([\mathbf{B}_0]_i^{\mathrm{T}} + [\mathbf{B}_\xi]_i^{\mathrm{T}} \xi_i + [\mathbf{B}_\eta]_i^{\mathrm{T}} \eta_i)[\mathbf{C}]([\mathbf{B}_0]_i + [\mathbf{B}_\xi]_i \xi_i + [\mathbf{B}_\eta]_i \eta_i) \, \mathrm{d}\Omega \tag{4.39}$$

$$= [\mathbf{B}_0]_i^{\mathrm{T}} [\mathbf{C}][\mathbf{B}_0]_i \Omega_i + [\mathbf{B}_\xi]_i^{\mathrm{T}} [\mathbf{C}][\mathbf{B}_\xi]_i \int_{\Omega_i} \xi_i^2 \, \mathrm{d}\Omega + [\mathbf{B}_\eta]_i^{\mathrm{T}} [\mathbf{C}][\mathbf{B}_\eta]_i \int_{\Omega_i} \eta_i^2 \, \mathrm{d}\Omega \tag{4.40}$$

$$= \quad [\mathbf{K}_0]_i \quad + \quad [\mathbf{K}_\xi]_i \quad + \quad [\mathbf{K}_\eta]_i \tag{4.41}$$

Before Equation (4.41) is used for a simulation, two modifications are made. First of all, the method of standard nodal averaging gives a stiffness matrix ($\mathbf{K}_0$) that is locking-free. Adding terms related to a linear strain to this stiffness matrix will add one volumetric constraint to the system per term. In the case of an incompressible material, this system will lock and is therefore useless for the simulation of these materials. To avoid this problem, not the full material response $\mathbf{C}$ is used for matrices $\mathbf{K}_\xi$ and $\mathbf{K}_\eta$, but only the deviatoric part. This deviatoric part of the material tangent is denoted by $\mathbf{C}_{\text{dev}}$ and is defined as:

$$[\mathbf{C}_{\text{dev}}] = [\mathbf{M}][\mathbf{C}][\mathbf{M}] \qquad \text{where:} \qquad [\mathbf{M}] = \begin{bmatrix} 0.5 & -0.5 & 0 \\ -0.5 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.42}$$

The volumetric part of the deformation is not considered for the linear strain terms. As a result, the number of volumetric constraints is the same as in the case of standard

nodal averaging. Secondly, a parameter κ will be introduced to control the number of linear terms in matrix $\mathbf{K}$. The final equation becomes:

$$[\mathbf{K}]_i = [\mathbf{K}_0]_i + \kappa([\mathbf{K}_\xi]_i + [\mathbf{K}_\eta]_i) \tag{4.43}$$

where:

$$[\mathbf{K}_\xi]_i = [\mathbf{B}_\xi]_i^{\mathrm{T}}[\mathbf{C}_{\mathrm{dev}}][\mathbf{B}_\xi]_i \int_{\Omega_i} \xi_i^2 \,\mathrm{d}\Omega \tag{4.44}$$

$$[\mathbf{K}_\eta]_i = [\mathbf{B}_\eta]_i^{\mathrm{T}}[\mathbf{C}_{\mathrm{dev}}][\mathbf{B}_\eta]_i \int_{\Omega_i} \eta_i^2 \,\mathrm{d}\Omega \tag{4.45}$$

Appendix C presents details for the implementation of these matrices.

4.4 Numerical Examples

In this section several numerical examples are presented to show the performance of the enriched nodal averaging. First a test is presented in Section 4.4.1 to investigate the influence of the κ-parameter. Section 4.4.2 will investigate the accuracy of the scheme for an infinite plate with hole. The stability of the enhanced nodal averaging will be examined in Section 4.4.3.

4.4.1 Beam Stretching and Bending

In this test, four aspects of the enriched nodal averaging will be studied. First of all a check is performed to establish whether the patch test is satisfied for regular as well as irregular grids. A beam will be subjected to a uniform tensile load in order to study this first order reproducibility. Secondly the accuracy of the scheme will be tested for the case in which the beam is subjected to a pure bending moment. Thirdly, the influence of the linear terms is studied by varying parameter κ and finally the artifact of volumetric locking will be investigated.

Figure 4.8(a) shows the geometry of the beam and the applied load cases. Load case 1 corresponds to the patch test and case 2 corresponds to the pure bending of the beam. A linear elastic material will be used with a Young's modulus of $E = 75$ and a Poisson's ratio of $\nu = 0.0$. A plane strain assumption applies to the model. Figures 4.8(b) and 4.8(c) give the two nodal grids. To give a magnitude of error, the L2 norm will be used on the error of the displacement field $\mathbf{u}$. This norm is expressed as:

$$\|e_{\mathrm{u}}\|_{\mathrm{L2}} = \sqrt{\int_\Omega \|\mathbf{u}(\mathbf{x}) - \mathbf{u}_{\mathrm{exact}}(\mathbf{x})\|^2 \,\mathrm{d}\Omega} \tag{4.46}$$

The exact displacement field for the beam under tension is:

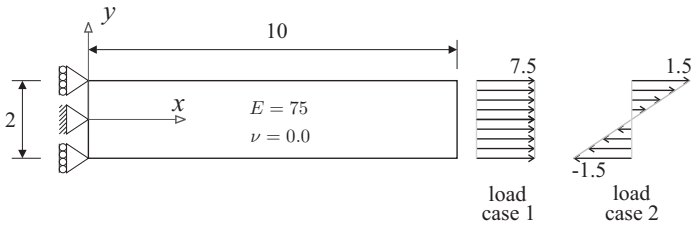
$$u_x = \frac{7.5(1+\nu)(1-\nu)}{E}x \tag{4.47}$$

$$u_y = -\frac{7.5\nu(1+\nu)}{E}y \tag{4.48}$$

and the exact displacement for the pure bending of the beam is:

$$u_x(\mathbf{x}) = \frac{3(1-\nu^2)}{2E}xy \tag{4.49}$$

$$u_y(\mathbf{x}) = -\frac{3(1-\nu^2)}{4E}x^2 - \frac{3\nu(\nu+1)}{4E}y^2 \tag{4.50}$$



(a) beam model

(b) regular node set

(c) irregular node set

Figure 4.8: The geometry and nodal grids for the beam problem.

Table 4.1: The *L2* error in the displacement for the stretching of the beam.

$\|e_u\|_{L2}$	$\kappa = 0$	$\kappa = 0.5$	$\kappa = 1.0$
regular	$3.054 \cdot 10^{-14}$	$8.719 \cdot 10^{-14}$	$1.880 \cdot 10^{-13}$
irregular	$1.792 \cdot 10^{-13}$	$2.125 \cdot 10^{-13}$	$3.268 \cdot 10^{-13}$

Table 4.2: The *L2* error in the displacement for the bending of the beam.

$\|e_u\|_{L2}$	$\kappa = 0$	$\kappa = 0.5$	$\kappa = 1.0$
regular	$9.732 \cdot 10^{-2}$	$1.271 \cdot 10^{-2}$	$5.129 \cdot 10^{-2}$
irregular	$1.547 \cdot 10^{-1}$	$7.376 \cdot 10^{-2}$	$2.588 \cdot 10^{-2}$

Table 4.1 gives the displacement error for load case 1. This patch test should be satisfied for any numerical method in computational solid mechanics. For both the regular and the irregular grid, the error in the displacement field is in the order of the machine precision. Hence, the patch test is satisfied independently of the setting for κ.

Table 4.2 presents the results for load case 2. Parameter κ is set to 0.0, 0.5 and 1.0. It can be seen that by adding linear terms in the expansion, the error is reduced. An

increase in accuracy in the order of one decade in the L2 norm can be obtained. Unfortunately, this minimal error comes for different settings of κ depending on whether the regular or the irregular grid is used.

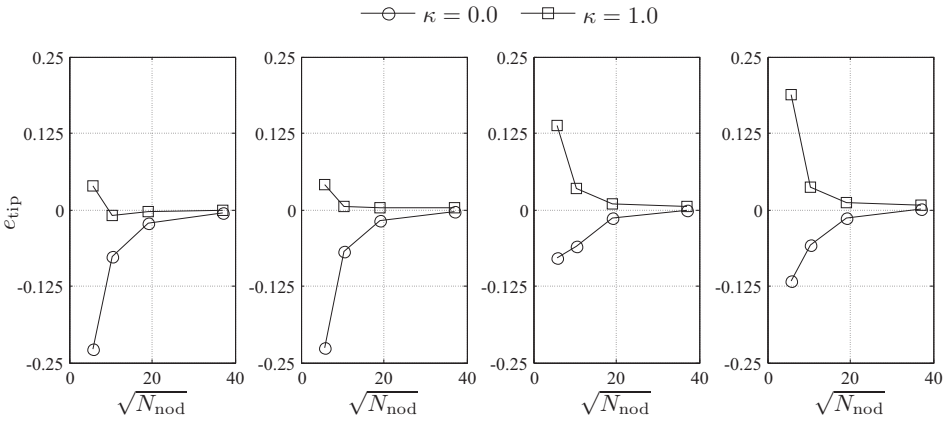


Figure 4.9: The tip displacement of the beam bending problem for various settings of ν and κ.

For the following test, the artifact of volumetric locking of the enriched nodal averaging will be investigated. The beam is discretised with four nodal grids with increasing density. The grids are randomly perturbed similarly as the grid shown in Figure 4.8(c). The number of nodes in x and y direction are 11x3, 21x5, 41x9 and 81x17. The error in the displacement will be monitored at point $(x, y) = (10, 0)$ for load case 2 and is defined as:

$$e_{\text{tip}} = \left\{ \begin{array}{cc} 0 & 1 \end{array} \right\} \left(\mathbf{u}(\mathbf{x}_{\text{tip}}) - \mathbf{u}_{\text{exact}}(\mathbf{x}_{\text{tip}}) \right) \tag{4.51}$$

$$\text{where:} \quad \mathbf{x}_{\text{tip}} = \left\{ \begin{array}{cc} 10 & 0 \end{array} \right\}^{\text{T}} \tag{4.52}$$

The Poisson's ratio is moved from fully compressible ($\nu = 0.0$) to nearly incompressible ($\nu = 0.4999$). Linear terms in the strain field are excluded ($\kappa = 0$) or included ($\kappa = 1.0$) in the computation.

Figure 4.9 shows the results of the analysis. It can be seen that for all settings of ν the tip displacement of the beam is most accurately approximated by using the enriched cell strain. Especially for the results made with the compressible material behaviour, the enriched cell strain approximates the displacement accurately for limited sets of nodes. Secondly, the artifact of volumetric locking seems absent. The nodal averaging gives locking-free results and the same holds for the enhanced nodal averaging. There is no spurious stiffening of the response of the beam for setting ν near the incompressible limit of 0.5. Finally, it is observed that the nodal averaged solution overestimates the tip displacement whereas the enriched nodal averaging underestimates this displacement. For nodal averaging, it is known that the approximation overestimates the displacements [84]. The addition of the stiffness terms related to the linear polynomials as shown in Equation (4.43) makes

the response of the beam more stiff if $\kappa = 1$. The displacement, however, is not underestimated for all nodal grids, but is also overestimated for two grids as can be seen in Figure 4.9(a). The incompatibility of the strain field is likely to cause this non-monotonic convergence to the exact solution.

4.4.2 Infinite Plate With a Hole

The enhanced scheme will be tested on the infinite plate with a hole problem. See Section 3.3.2 for the details of this problem, including the exact displacement field, the loading conditions, the material parameters and the dimensions. Four nodal grids will be used to examine the rate of convergence of the scheme in the L2 error norm. The nodal grids are displayed in Figure 4.10. Parameter κ is set to 0.0, 0.5 and 1.0. The L2 error as given in Equation (4.46) is computed by using the exact displacement field.

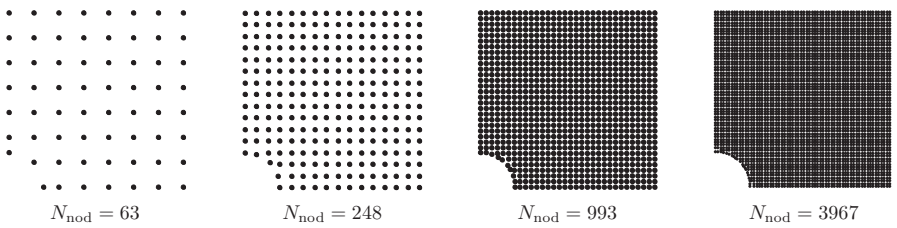


Figure 4.10: Four node sets as used for the plate with a hole problem.

Figure 4.11 shows the results of the test. It can be seen that the rate of convergence is approximately 2 for all three settings. For an approximation which reproduces linear polynomials in the displacement field, the quadratic terms of the exact displacement field are not reproduced by the approximating field and hence they contribute to the error. The rate of convergence of order 2 is therefore expected. A second conclusion that can be drawn from Figure 4.11 is that the unmodified nodal averaging ($\kappa = 0$) is the least accurate. By adding linear strain terms to the approximation, the error is reduced by approximately a factor of 7 for the setting of $\kappa = 0.5$.

4.4.3 Pinched Bar

In the following test, the stability of the scheme is tested. Standard nodal averaging can suffer from spurious modes in the displacement, especially on regular grids as explained in Section 4.2.2.

To examine the stability of the enriched cell strain, the problem shown in Figure 4.12(a) will be analysed. The bar is modelled with a regular grid of nodes as depicted in Figure 4.12(b) such that the spurious mode shown in Figure 4.3 can manifest itself easily. The grid has 5 nodes over the height and 21 nodes along the length. The bar is pinched between two supports on the top surface of the bar and one support on the

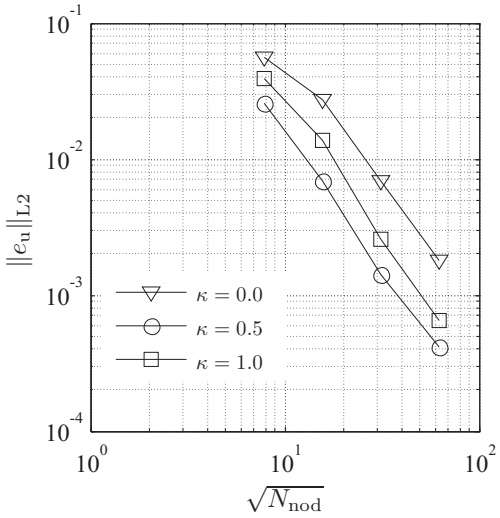
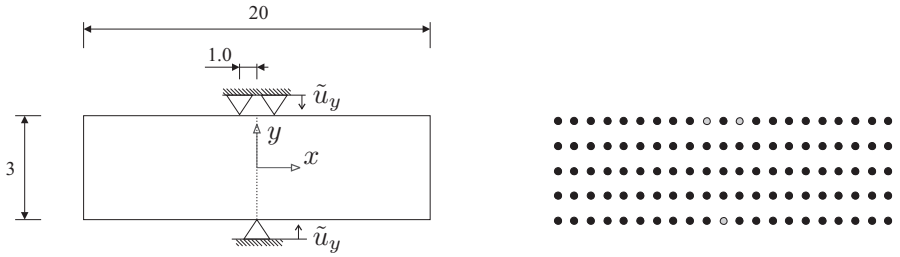


Figure 4.11: Error plots for the plate with a hole for various settings of κ.

bottom surface of the bar. The displacements of the two supports on the top of the bar are prescribed downwards and are fixed horizontally. The support at the bottom is prescribed upwards and fixed horizontally also. The magnitude of the prescribed displacement $\tilde{u}_y$ is 0.2. A plane strain assumption applies to the bar, and the Young's modulus and Poisson's ratio are set to $E = 10$ and $\nu = 0.3$ respectively. To examine the performance of the developed scheme, the vertical displacements of the nodes lying in the centreline of the bar ($y = 0$) will be monitored.



(a) pinched bar model (b) cloud of nodes

Figure 4.12: The model and the nodal grids for the pinched bar problem. The grey nodes are prescribed.

Figure 4.13 shows the vertical displacement of the nodes lying on the horizontal centreline of the bar. Clearly, the unmodified nodal averaging is spurious. The oscillation is triggered at the supports and damps out towards the end of the bar. For

a bar of infinite length, this oscillation will not diminish and will progress identically as shown in Figure 4.3. By adding linear polynomials to the cell strain, the results improve and the spurious mode is suppressed. Setting $\kappa = 0.5$ or $\kappa = 1.0$ eliminates the spurious mode and gives an elliptic response as is expected for this problem.

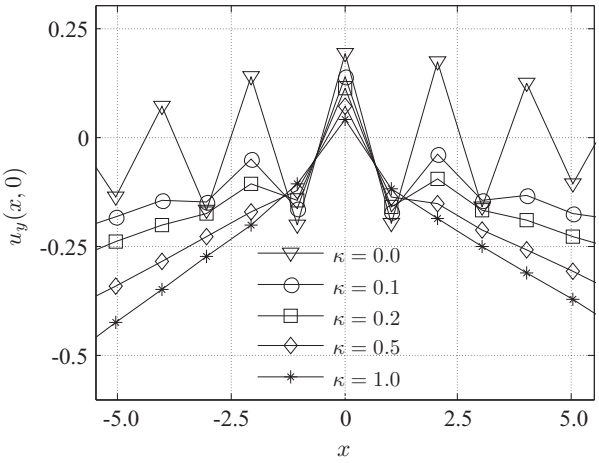
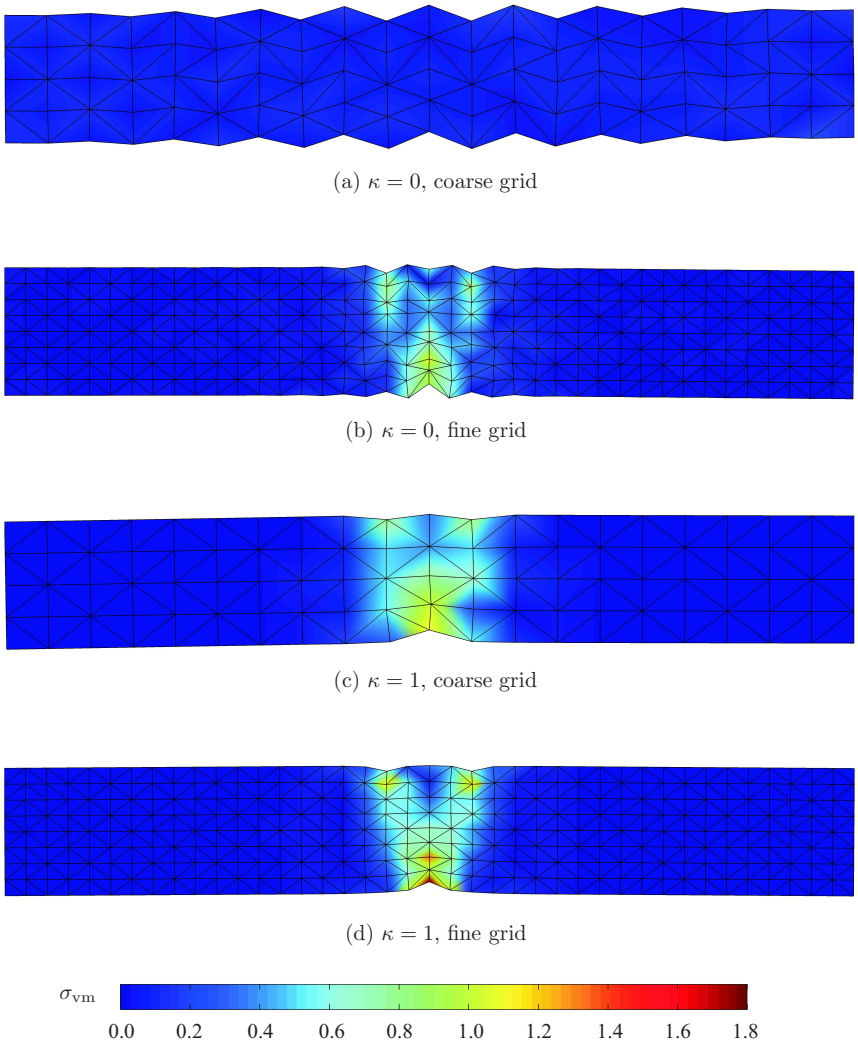


Figure 4.13: The vertical displacement of the nodes at the centreline $y = 0$ of the pinched bar problem.

It can be stated that the displacement field in the centreline of the beam improves considerably by employing the enriched nodal averaging strategy. To inspect the stress field, the stress is monitored over a coarse and fine grid for two settings of κ. Figure 4.14 shows contour plots of the von Mises stress field. The plot is constructed by interpolating the stress field resulting from the constant strain terms $\boldsymbol{\varepsilon}_0$ on the Delaunay triangles.

Figures 4.14(a) and 4.14(b) show the nodal averaged results. Figures 4.14(c) and 4.14(d) show the enriched nodal averaged results. It can be seen that the stress prediction for the nodally averaged results are too low in general and are especially incorrect on the coarse grid. The stress is close to zero throughout the domain and the displacement field is highly oscillatory. For the finer grid, the stress levels become higher and focus around the supports, but the displacements remain oscillatory and small oscillations are present in the stress field. The enriched responses show firstly a stable response in the displacement, independent of the used grid. Secondly, the stress prediction is more smooth and forms the expected stress bands from the top supports to the bottom support.



(a) $\kappa = 0$, coarse grid

(b) $\kappa = 0$, fine grid

(c) $\kappa = 1$, coarse grid

(d) $\kappa = 1$, fine grid

Figure 4.14: Contour plots of the von Mises stress on a coarse and fine grid for two settings of κ.

4.5 Conclusions

In this chapter the stability of the nodal averaging technique was improved by an enriched nodal averaging strategy. The essence of the proposed method is to include linear polynomials in the strain of a cell accompanying a node. It was shown that the proposed scheme improves the stability of the results while remaining free of volumetric locking. Moreover, results indicate that not only the stability of the result improves, but also the accuracy. A numerical parameter κ was introduced to control the amount of stabilisation terms added to the system. It can be concluded that setting this parameter between 0.5 and 1.0 gives satisfactory results as the stability and accuracy improve. The following chapter will propose an extension of the method for use in geometrical non-linearity with implicit time integration.

5

Adaptive Smoothed Finite Elements in Large Deformations

5.1 Introduction

Finite element simulations of large deformation processes, like extrusion or forging, can fail as a result of element distortion. Either many re-meshing steps or a Eulerian formulation is required to avoid this problem. For both solutions, state variables of material points have to be mapped or convected, which reduces the accuracy of the solution.

The nodally averaged triangular finite element was investigated in the previous two chapters and was shown to have favourable properties for the simulation of bulk forming processes. The essence of the method is an assumed strain field which is constructed by a nodal averaging procedure. By this nodal averaging of the strain, the constitutive behaviour is evaluated per node instead of per 'Gaussian' integration point. A consequence is that if nodal positions are kept Lagrangian, the material point in which the constitutive behaviour is evaluated remains at the nodal location. As long as no rearrangement of the nodes takes place, there is no need to convect or map data concerning the material model. The evaluation of the material model is thereby no longer dependent on the linear triangle shape functions. Hence, the cloud of nodes can be re-triangulated without modifying the material data of a point. It is possible to revise the mesh with a Delaunay triangulation for every step in the simulation. By definition, a Delaunay triangulation of a cloud of nodes produces triangles with optimal shape for that set of nodes. If nodes start moving, the triangulation will alter such that distortional effects are minimal.

Nodally averaged finite elements are known by the name of Smoothed FEM (SFEM). Since the proposed method revises the mesh for every step, the method will be named

the Adaptive Smoothed Finite Element Method (ASFEM). This chapter presents the formulations for ASFEM for large deformations. An updated Lagrangian description is chosen in order to easily incorporate the triangulation which alters in time.

This chapter is organised as follows. Firstly the basic governing equations of a body undergoing large deformations are given in Section 5.2. The approximation spaces for the ASFEM method are defined and the equation of virtual work is given. Section 5.3 presents details for the implementation of the method. In Section 5.4 the proposed method is used to simulate several test cases in order to prove its validity in large deformations.

5.2 Governing Equations

The goal of this section is to present the main governing equations of a body being subjected to large deformations and to define the approximation spaces for the ASFEM method. For more detailed information of large deformations in general, the reader is referred to the work of Huétink [66] or Belytschko *et al.* [17].

Figure 5.1 shows an illustration of a body subjected to large deformations. The volume of the body in the undeformed and deformed state are given by Ω_0 and Ω respectively. The spatial location of a point is given by its Eulerian coordinates $\mathbf{x}$. The location of that point in the undeformed configuration is given by its Lagrangian coordinates $\mathbf{X}$. The Eulerian coordinates of a material point are a function of the Lagrangian coordinates $\mathbf{X}$ and the time t, and are therefore expressed as $\mathbf{x}(\mathbf{X}, t)$. As an illustration, at $t = 0$ the Eulerian coordinates are equal to the Lagrangian coordinates ($\mathbf{x}(\mathbf{X}, 0) = \mathbf{X}$). Note that for further derivations also the symbol τ will be used to indicate time.

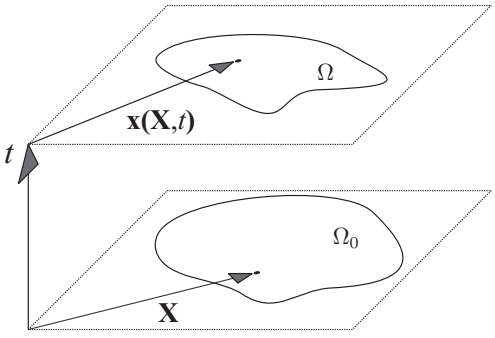


Figure 5.1: A schematic picture of deforming material.

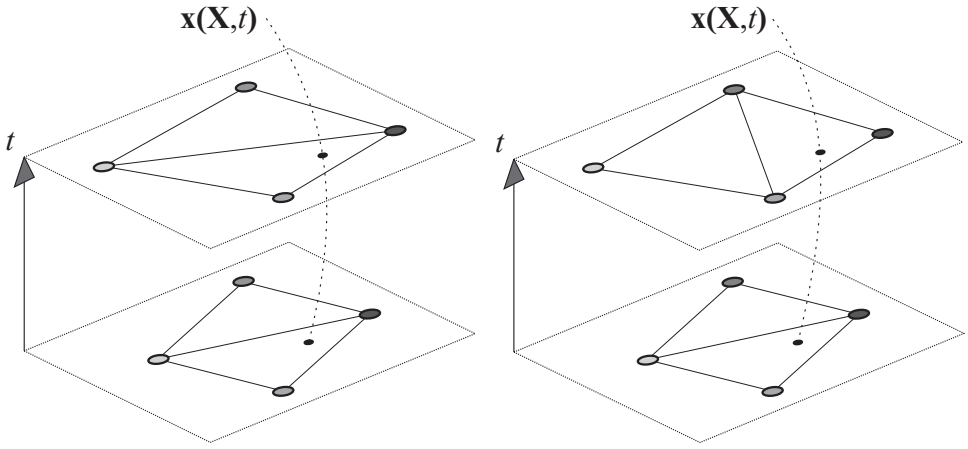
5.2.1 Motion

The motion of an arbitrary point $\mathbf{X}$ in the body Ω can be expressed in a general way by integrating the velocity of that point over time:

$$\mathbf{x}(\mathbf{X}, t) = \mathbf{X} + \int_0^t \mathbf{v}(\mathbf{X}, \tau)\, \mathrm{d}\tau \tag{5.1}$$

where vector $\mathbf{v}$ contains the velocity of a point $\mathbf{X}$ at time τ.

The velocity field as given in Equation (5.1) will be discretised by a set of shape functions. For the proposed method, the shape functions for the scheme are chosen to be simple linear triangular interpolations defined upon the Delaunay triangulation of the cloud of nodes. This triangulation can be constructed either by using the Eulerian coordinates, or the Lagrangian coordinates of the nodes. If the Lagrangian coordinates are used, only one Delaunay triangulation is sufficient since the triangulation will not alter throughout the simulation. If the Eulerian coordinates are used, the triangulation will alter in time. If the nodes move in space, their Eulerian coordinates will change (contrary to their Lagrangian counterpart). Figure 5.2 shows a Delaunay triangulation based on the two coordinate sets. Figure 5.2(a) shows the trajectory of a Lagrangian point traveling in space and the triangulation on which the shape functions are defined. Since the triangulation does not alter, this point remains situated within one triangle, as is the case for classical compatible finite elements. Figure 5.2(b) shows the trajectory of a point and the triangulation based on Eulerian coordinates. In time, the point will 'travel' through several triangles. Hence its shape function values alter in time.



(a) the triangulation based on Lagrangian coordinates

(b) the triangulation based on Eulerian coordinates

Figure 5.2: A Delaunay triangulation based on two sets of coordinates.

Leaving the choice for the definition of the triangulation algorithm open, the approximation of the velocity field can be expressed in a general form as follows:

$$\mathbf{v}(\mathbf{x}) = \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}(\mathbf{X}, t))\dot{\mathbf{d}}_i \tag{5.2}$$

where $\dot{\mathbf{d}}_i$ are the nodal velocities of node i and $\phi_i(\mathbf{x}(\mathbf{X}, t))$ are the shape functions of the same node as a function of the Eulerian coordinates at time t. The movement of a point in the approximated velocity field is expressed as:

$$\mathbf{x}(\mathbf{X}, t) = \mathbf{X} + \int_0^t \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}(\mathbf{X}, \tau))\dot{\mathbf{d}}_i \, \mathrm{d}\tau \tag{5.3}$$

This equation states that the deformed configuration is dependent on the shape functions and the nodal velocities of all previous times t. In the remainder of this section, expressions for the deformed configuration will be derived firstly for the Lagrangian triangulation and thereafter for the Eulerian triangulation.

If the shape functions are defined on a Lagrangian Delaunay triangulation, the following equation holds:

$$\phi_i(\mathbf{x}(\mathbf{X}, t)) = \phi_i(\mathbf{X}) \qquad \text{for } i = 1...N_{\text{nod}} \tag{5.4}$$

This equation states that the values of the shape functions for a Lagrangian point are equal for the deformed configuration $\mathbf{x}$ and the undeformed configuration $\mathbf{X}$. This gives a formulation as is commonly used for finite elements. Equation (5.3) can be simplified as follows:

$$\mathbf{x}(\mathbf{X}, t) = \mathbf{X} + \int_0^t \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{X})\dot{\mathbf{d}}_i \, \mathrm{d}\tau \tag{5.5}$$

$$= \mathbf{X} + \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{X})\mathbf{d}_i \tag{5.6}$$

where $\phi_i(\mathbf{X})$ is a shape function dependent on the Lagrangian coordinates and $\mathbf{d}_i$ are the nodal displacements which are defined as:

$$\mathbf{d}_i = \int_0^t \dot{\mathbf{d}}_i \, \mathrm{d}\tau \tag{5.7}$$

So in order to calculate the location of any point in the body, the shape functions can be computed at $\mathbf{X}$ and simply be multiplied by the nodal displacements. A downside is that if the body deforms severely, the Delaunay triangulation will become distorted and results degrade, as is the case for finite elements.

If the shape functions are defined upon the Eulerian Delaunay triangulation, $\boldsymbol{\phi}$ will be dependent on $\mathbf{x}$ and hence implicitly on time t. Equation (5.4) does not hold:

$$\phi_i(\mathbf{x}(\mathbf{X}, t)) \neq \phi_i(\mathbf{X}) \qquad \text{for } i = 1...N_{\text{nod}}$$

As a result, Equation (5.3) can not be simplified further. The first implication of this equation not being satisfied is that, in order to compute a point's current location $\mathbf{x}$, the contributions of all shape functions ϕ_i that are, or were, non-zero on the trajectory of that point $\mathbf{X}$ have to be considered. The second implication concerns the triangulation algorithm. For a minor change in the nodal positions, the triangulation can alter. See section 5.3.6 for more details on this discrete behaviour of the Delaunay triangulation algorithm. Since the shape functions are defined upon this triangulation, their values will change accordingly. Hence the shape functions are in this case C^{-1} continuous in time, which can cause convergence problems in implicit computations. Both implications can be resolved by using a stepwise evaluation of Equation (5.3). Section 5.3.1 will discuss this approximation and will show that the 'optimal' Eulerian triangulation can be used without the issues envisaged above.

5.2.2 Deformation

For the case of nodal averaging, the rate of deformation is assumed to be different than simply the symmetric gradient of the velocity field. The assumed field is used to describe the deformations of the body and to weigh the strong form of the equilibrium. Whereas in the previous chapters the symbol ($^-$) was used on all assumed variables, this will be dropped for readability, except for the yet to be defined smoothed gradient operator $\bar{\nabla}$.

To start, the smoothed gradient operator for large deformations will be defined. If Ω_i is the volume of a cell corresponding to node i in the deformed configuration, the smoothed gradient is defined as:

$$\bar{\nabla}... = \frac{1}{\Omega_i}\int_{\Omega_i} \nabla... \,\mathrm{d}\Omega \qquad \forall\, \mathbf{x} \in \Omega_i \quad i = 1...N_{\mathrm{nod}} \tag{5.8}$$

The assumed velocity gradient $\mathbf{L}$ can be computed by using the smoothed gradient operator on the velocity field:

$$\mathbf{L} = (\bar{\nabla}\mathbf{v})^{\mathrm{T}} \tag{5.9}$$

The assumed velocity gradient incorporating the approximation for the velocity field is defined as:

$$\mathbf{L} = \left(\bar{\nabla}\sum_{i=1}^{N_{\mathrm{nod}}} \phi_i(\mathbf{x}(\mathbf{X},t))\dot{\mathbf{d}}_i\right)^{\mathrm{T}} \tag{5.10}$$

This equation shows that the velocity gradient depends on the smoothed gradient $\bar{\nabla}$, on the nodal velocities $\dot{\mathbf{d}}$ and on the shape functions ϕ. Section 5.3 will propose a numerical approximation of this equation.

If the velocity gradient is known, the rate of deformation and the spin can be computed as follows:

$$\mathbf{D} = \frac{1}{2}(\mathbf{L} + \mathbf{L}^{\mathrm{T}}) \qquad \text{and} \qquad \mathbf{W} = \frac{1}{2}(\mathbf{L} - \mathbf{L}^{\mathrm{T}}) \tag{5.11}$$

5.2.3 Virtual Work

In the preceding chapters, a set of nodal displacements, with which the nodal equilibrium was satisfied, was easily found as a result of several simplifications made. The strains were assumed small, the change of the geometry caused by deformation was neglected and a linear elastic constitutive model was chosen. For simulations in large deformations, these assumptions cannot be justified since the change of geometry is likely to be the purpose of the simulation and the constitutive model and the strain definitions are non-linear. As a result, the nodal equilibrium is non-linear, and the search for a set of displacements for which nodal equilibrium holds requires a special strategy.

In this research a Newton–Raphson scheme will be used in order to find this equilibrium within a predefined small time step (an increment). This scheme iterates towards the nodal equilibrium by varying the nodal displacements. For one iteration, a linearisation of the force vectors is required in order to predict new nodal displacements. The governing equations required for the Newton–Raphson process will be derived in a general form in this section. Thereafter Section 5.3.3 will extend these derivations by incorporating the discretisation of space.

The principle of virtual work can be found by rewriting Equation (4.1) for large deformations:

$$\delta W = \underbrace{\int_\Omega \delta\mathbf{D} : \boldsymbol{\sigma}\, \mathrm{d}\Omega}_{\delta W_{\text{int}}} - \underbrace{\int_\Omega \delta\mathbf{v}\cdot\mathbf{f}\, \mathrm{d}\Omega - \int_\Gamma \delta\mathbf{v}\cdot\tilde{\mathbf{t}}\, \mathrm{d}\Omega}_{\delta W_{\text{ext}}} = 0 \qquad \forall\, \delta\mathbf{v} \tag{5.12}$$

where δW is the virtual work and $\delta\mathbf{D}$ is the virtual rate of deformation tensor. The internal and external virtual work are given by δW_{int} and δW_{ext} respectively.

The rate of the virtual work, also known as the virtual power, will be used for the linearisation of the force vectors later on. The internal virtual power is found as follows:

$$\begin{aligned}
\delta\dot{W}_{\text{int}} &= \frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \delta\mathbf{D} : \boldsymbol{\sigma}\, \mathrm{d}\Omega && (5.13)\\
&= \int_\Omega \frac{\partial}{\partial t}\left(\delta\mathbf{D} : \boldsymbol{\sigma}\right)\, \mathrm{d}\Omega + \int_\Gamma \left(\delta\mathbf{D} : \boldsymbol{\sigma}\right)\mathbf{v}\cdot\mathbf{n}\, \mathrm{d}\Gamma && (5.14)\\
&= \int_\Omega \frac{\partial}{\partial t}\left(\delta\mathbf{D} : \boldsymbol{\sigma}\right)\, \mathrm{d}\Omega + \int_\Omega \left(\delta\mathbf{D} : \boldsymbol{\sigma}\right)\nabla\cdot\mathbf{v} + \mathbf{v}\cdot\left(\nabla\left(\delta\mathbf{D} : \boldsymbol{\sigma}\right)\right)\, \mathrm{d}\Omega && (5.15)\\
&= \int_\Omega \frac{\mathrm{d}}{\mathrm{d}t}\left(\delta\mathbf{D} : \boldsymbol{\sigma}\right)\, \mathrm{d}\Omega + \int_\Omega \left(\delta\mathbf{D} : \boldsymbol{\sigma}\right)\nabla\cdot\mathbf{v}\, \mathrm{d}\Omega && (5.16)\\
&= \int_\Omega \delta\dot{\mathbf{D}} : \boldsymbol{\sigma}\, \mathrm{d}\Omega + \int_\Omega \delta\mathbf{D} : \dot{\boldsymbol{\sigma}}\, \mathrm{d}\Omega + \int_\Omega \delta\mathbf{D} : \boldsymbol{\sigma}\ \mathrm{tr}(\mathbf{D})\, \mathrm{d}\Omega && (5.17)
\end{aligned}$$

where $\dot{\boldsymbol{\sigma}}$ is the rate of the Cauchy stress and tr() is the trace operator on a tensor. Note that if $\tilde{\mathbf{t}}$ and $\mathbf{f}$ in Equation (5.12) are kept constant in time, the external virtual power $\delta\dot{W}_{\text{ext}}$ is zero. Hence this term does not need to be linearised.

The Cauchy stress tensor should be objective. This implies that the tensor, expressed in a corotational frame, should remain constant for a point undergoing pure rotation. In order to make the stress objective, the Jaumann rate will be adopted. This rate is expressed as:

$$\dot{\boldsymbol{\sigma}} = \mathbf{C}^{\mathrm{J}} : \mathbf{D} + \mathbf{W} \cdot \boldsymbol{\sigma} - \boldsymbol{\sigma} \cdot \mathbf{W} \tag{5.18}$$

where $\mathbf{C}^{\mathrm{J}}$ is the tangent of the constitutive behaviour. By filling Equation (5.18) into Equation (5.17), the resulting virtual power can be simplified further. This simplification is quite involving and the reader is referred to Huétink [66] for a detailed derivation. The resulting equation of virtual power becomes:

$$\delta \dot{W}_{\mathrm{int}} = \int_{\Omega} (\delta \mathbf{L}^{\mathrm{T}} \cdot \mathbf{L} - 2\delta \mathbf{D} \cdot \mathbf{D}) : \boldsymbol{\sigma} \, \mathrm{d}\Omega + \int_{\Omega} \delta \mathbf{D} : \mathbf{C}^{\mathrm{J}} : \mathbf{D} \, \mathrm{d}\Omega + \int_{\Omega} \delta \mathbf{D} : \boldsymbol{\sigma} \ \mathrm{tr}(\mathbf{D}) \, \mathrm{d}\Omega \tag{5.19}$$

Section 5.3.3 gives the system matrices and vectors for the discretised equation of virtual work and its rate form.

The constitutive behaviour is expressed in a rate form. The virtual work, however, is a function of the stress tensor and not of its rate. So in order to compute the stress tensor at time step t, the rate form of the stress needs to be integrated in time:

$$\boldsymbol{\sigma}(t) = \int_0^t \left(\mathbf{C}^{\mathrm{J}} : \mathbf{D} + \mathbf{W} \cdot \boldsymbol{\sigma} - \boldsymbol{\sigma} \cdot \mathbf{W} \right) \, \mathrm{d}\tau \tag{5.20}$$

Section 5.3.2 will discuss an algorithm in order to compute this integral.

5.3 Implementation Aspects

This section will present numerical evaluations of the governing equations as given in Section 5.2. Aspects required for the implementation of the method in a computer code are given.

Firstly, incremental formulations for the velocity field and the rate of deformation will be proposed in Section 5.3.1. Section 5.3.2 will give the used stress-update algorithm. The system matrices, the stabilisation procedure and a definition of the Newton–Raphson method are given in Sections 5.3.3, 5.3.4 and 5.3.5. The triangulation and the tessellation algorithm are given in Section 5.3.6.

5.3.1 Incremental Formulations

For a Newton–Raphson procedure to converge successfully, four conditions need to be satisfied. Firstly, the time step under consideration should be of reasonable size. Convergence is guaranteed only if the initial prediction for the algorithm is sufficiently close to final solution. Secondly, the tangent for the predictor step should be consistent with the force vector for optimal convergence. For convergence in general, which is

not necessarily optimal, the tangent used should be sufficiently close to the consistent tangent. Thirdly, the weak form of the method under consideration should be stable. Non-ellipticity can cause divergence. Finally, and most importantly, the space of the nodal forces is at least smooth with respect to the degrees of freedom. For instance, loads that are C^{-1} continuous with respect to the nodal displacement degrees of freedom within the increment can cause serious convergence problems. Especially for contact mechanics, these type of problems can occur; imagine a node moving in and out of contact over subsequent iterations.

Using a Delaunay triangulation based on Eulerian coordinates results in an internal force vector which can be C^{-1} continuous with respect to the nodal degrees of freedom. Convergence problems can be envisaged. To avoid this problem, the triangulation will be computed only at the start of the increment and will remain unaltered during the increment. For the following derivations, index k will be used as incremental counter. The coordinates of a point in the last converged configuration at time step t_k are referred to as $\mathbf{x}_k$. The coordinates of a point at time step t_{k+1} are referred to as $\mathbf{x}_{k+1}$. Note that the configuration denoted with this subscript is not necessarily a converged configuration. The motion of the points in the body as given in Equation (5.3) will be simplified by using the assumption as given above:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \int_{t_k}^{t_{k+1}} \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}) \dot{\mathbf{d}}_i \, \mathrm{d}\tau \tag{5.21}$$

$$\approx \mathbf{x}_k + \int_{t_k}^{t_{k+1}} \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}_k) \dot{\mathbf{d}}_i \, \mathrm{d}\tau \tag{5.22}$$

$$\approx \mathbf{x}_k + \sum_{i=1}^{N_{\text{nod}}} \phi_i(\mathbf{x}_k) \Delta\mathbf{d}_i \tag{5.23}$$

where $\phi_i(\mathbf{x}_k)$ is the shape function that is only depending on $\mathbf{x}_k$. The incremental nodal displacements $\Delta\mathbf{d}_i$ are defined as:

$$\Delta\mathbf{d}_i = \int_{t_k}^{t_{k+1}} \dot{\mathbf{d}}_i \, \mathrm{d}\tau \tag{5.24}$$

To summarise, the shape functions are computed only at the start of the increment and are used to compute the displacement field for that increment. The integral form of Equation (5.3) is resolved by a piecewise constant approximation and the total displacement can be found by summing all incremental displacements.

For the smoothed gradient, similar issues are of concern as for the velocity field. Therefore, the smoothed gradient operator $\bar{\nabla}$ will be computed only at the start of the increment. The assumed gradient towards the last converged increment is defined as:

$$\bar{\nabla}_k ... = \frac{1}{\Omega_i(t_k)} \int_{\Omega_i(t_k)} \nabla_k ... \, \mathrm{d}\Omega \qquad \forall \, \mathbf{x} \in \Omega_i(t_k) \qquad \text{for } i = 1...N_{\text{nod}} \tag{5.25}$$

where $\Omega_i(t_k)$ is the volume of the cell accompanying node i at time step t_k. Operator ∇_k contains the gradients to the coordinates of the last converged step $\mathbf{x}_k$. In order

to obtain the averaged gradient at any intermediate configuration, the gradient can be easily pushed forward by using the incremental deformation gradient:

$$\Delta\mathbf{F}_k = \left(\bar{\nabla}_k \mathbf{x}_{k+1}\right)^{\mathrm{T}} \tag{5.26}$$

$$= \mathbf{1} + \left(\bar{\nabla}_k \sum_{i=1}^{N_{\mathrm{nod}}} \phi_i(\mathbf{x}_k)\Delta\mathbf{d}_i\right)^{\mathrm{T}} \tag{5.27}$$

The current coordinates $\mathbf{x}_{k+1}$ are given in Equation (5.23) and these coordinates are computed by using the triangulation at the last converged configuration $\mathbf{x}_k$. The smoothed gradient at any intermediate configuration is computed as:

$$\bar{\nabla}_{k+1} = \Delta\mathbf{F}_k^{-\mathrm{T}} \bar{\nabla}_k \qquad \text{for } t_k < t < t_{k+1} \tag{5.28}$$

To summarise, during the increment the triangulation and the smoothed gradient operator will not alter. Since the configuration is not updated for each iteration, but only for each increment, this approach could be best described as an incremental updated Lagrangian approach. If the first three criteria for good convergence of the Newton–Raphson are met (sufficiently small step, consistency, stability), the developed method should converge.

5.3.2 Stress Update Algorithm

The equations governing the constitutive behaviour of the material are defined in a rate form. Equation (5.18) shows the form relating the rate of stress to the rate of deformation. In order to compute the value of the stress tensor at a certain time step t_{k+1}, the rate form of the stress as given in Equation (5.18) has to be integrated in time. The integrand of the rate form over one increment is given by:

$$\int_{t_k}^{t_{k+1}} \dot{\boldsymbol{\sigma}}\, \mathrm{d}\tau = \int_{t_k}^{t_{k+1}} \left(\boldsymbol{\sigma}^{\mathrm{J}} + \mathbf{W}\cdot\boldsymbol{\sigma} - \boldsymbol{\sigma}\cdot\mathbf{W}\right) \mathrm{d}\tau \tag{5.29}$$

The numerical evaluation of this equation is not straightforward, and numerical artifacts can be present in case of inaccurate approximations. This section will briefly discuss the used stress update. For more details on the algorithm, the reader is referred to Simo and Hughes [111].

The strain computed per increment should be numerically approximated such that it is zero in case of rigid rotations. An accurate numerical approximation of this tensor is obtained by using the spatial derivatives of the configuration halfway the increment. This approach is simple and very effective for finding both accurate rotations as well as accurate rotation free strains. In the following derivations, firstly the incremental rotation will be determined. Thereafter the formulas for the incremental strain and the complete constitutive response are given.

Firstly, the time integral of the velocity gradient over the increment, here denoted by $\Delta\mathbf{L}_k$, is determined:

$$\Delta\mathbf{L}_k = 2\left(\Delta\mathbf{F}_k - \mathbf{1}\right)\cdot\left(\Delta\mathbf{F}_k + \mathbf{1}\right)^{-1} \tag{5.30}$$

where $\Delta\mathbf{L}_k$ is the spatial derivative of the incremental displacement towards the midpoint configuration. See Appendix D for the derivation of this equation. The incremental rotation $\Delta\mathbf{R}_k$ is determined by using the Hughes and Winget update [67]:

$$\Delta\mathbf{R}_k = \frac{1}{2}\left(\mathbf{1} - \Delta\mathbf{W}_k\right)^{-1} \cdot \left(\mathbf{1} + \Delta\mathbf{W}_k\right) \tag{5.31}$$
$$\text{where:} \quad \Delta\mathbf{W}_k = \frac{1}{2}\left(\Delta\mathbf{L}_k - \Delta\mathbf{L}_k^{\mathrm{T}}\right)$$

Secondly, the incremental strain, denoted by $\Delta\mathbf{D}_k$, is computed as follows:

$$\Delta\mathbf{D}_k = \frac{1}{2}\sqrt{\Delta\mathbf{R}_k} \cdot \left(\Delta\mathbf{L}_k + \Delta\mathbf{L}_k^{\mathrm{T}}\right) \cdot \sqrt{\Delta\mathbf{R}_k}^{\mathrm{T}} \tag{5.32}$$

where the square root of the incremental rotation is implicitly defined as:

$$\Delta\mathbf{R}_k = \sqrt{\Delta\mathbf{R}_k} \cdot \sqrt{\Delta\mathbf{R}_k} \tag{5.33}$$

Finally, by using the incremental strain and the incremental rotation, the numerical counterpart of Equation (5.29) can be constructed:

$$\boldsymbol{\sigma}_{k+1} = \Delta\boldsymbol{\sigma}_k(\Delta\mathbf{D}_k) + \Delta\mathbf{R}_k \cdot \boldsymbol{\sigma}_k \cdot \Delta\mathbf{R}_k^{\mathrm{T}} \tag{5.34}$$

The stress update as given above is accurate and simple. A drawback, however, is that the virtual power as given in Equation (5.19) is not consistent with this algorithm. In practice, quadratic convergence in the Newton–Raphson procedure will not be obtained. Luckily, if the increments are not too large, the continuum tangent will be close enough to the consistent tangent in order to give good convergence. A consistent tangent incorporating a full linearisation of the stress update algorithm as given above does not seem to be available in literature.

5.3.3 System Matrices

The derivations for the virtual work were performed in tensor form and did not incorporate the approximation fields as given in Section 5.2.1 and 5.2.2. In this section these discretisation spaces will be used in order to recast the virtual work and the virtual power into the nodal force vector and the stiffness matrix respectively. The final equations are given in Voigt form in order to allow for easy implementation in a computer program. As in the preceding chapters, matrices $\mathbf{B}$ and $\mathbf{N}$ relate the nodal velocities to the approximation spaces $\mathbf{D}$ and $\mathbf{v}$ and the virtual approximation spaces $\delta\mathbf{D}$ and $\delta\mathbf{v}$ as follows:

$$\{\mathbf{v}\} = [\mathbf{N}]\{\dot{\mathbf{d}}\} \qquad \{\delta\mathbf{v}\} = [\mathbf{N}]\{\delta\dot{\mathbf{d}}\} \tag{5.35}$$
$$\{\mathbf{D}\} = [\mathbf{B}]\{\dot{\mathbf{d}}\} \qquad \{\delta\mathbf{D}\} = [\mathbf{B}]\{\delta\dot{\mathbf{d}}\} \tag{5.36}$$

Note that matrix $\mathbf{B}$ is constructed by using the derivatives at the current step ($\bar{\nabla}_{k+1}$).

Firstly, the internal and external force vector will be given in Voigt notation. Equation (5.12) is rewritten as follows:

$$\underbrace{\int_\Omega [\mathbf{B}]^{\mathrm{T}}\{\boldsymbol{\sigma}\}\,\mathrm{d}\Omega}_{\{\mathbf{F}_{\mathrm{int}}\}} - \underbrace{\int_\Omega [\mathbf{N}]^{\mathrm{T}}\{\mathbf{f}\}\,\mathrm{d}\Omega - \int_\Gamma [\mathbf{N}]^{\mathrm{T}}\{\mathbf{t}\}\,\mathrm{d}\Omega}_{\{\mathbf{F}_{\mathrm{ext}}\}} = \mathbf{0} \tag{5.37}$$

where $\mathbf{F}_{\mathrm{int}}$ is the internal force vector and $\mathbf{F}_{\mathrm{ext}}$ is the external force vector.

For each iteration, a tangent of the force vector, also known by the name stiffness matrix, is required in order to predict a new configuration. Instead of consistently linearising the force vector $\mathbf{F}_{\mathrm{int}}$, which would also require a linearisation of the stress update algorithm, the continuum tangent is used. This tangent can easily be found by considering the virtual power as was given in Equation (5.19). The stiffness matrix is most easily implemented in a computer code when it is expressed in Voigt form. The equation of virtual power, on the contrary, is expressed in tensor form. In Appendix D a derivation is given in which the virtual power is rewritten into the Voigt form. In the remainder of this section, the virtual power in Voigt form will be discretised by using the approximation spaces. Do note that it is not necessary to convert the equations from tensor to Voigt form. This conversion is only given since it allows for a simple implementation into a computer code.

The internal stiffness matrix follows from the internal virtual power in Voigt form as given in Equation (D.23) and the application of the discretisation spaces. This matrix is denoted by $\mathbf{K}_{\mathrm{int}}$ and is defined as:

$$[\mathbf{K}_{\mathrm{int}}] = \int_\Omega [\mathbf{B}]^{\mathrm{T}}[\mathbf{C}_{\mathrm{tot}}][\mathbf{B}] + [\mathbf{B}_{\mathrm{nl}}]^{\mathrm{T}}[\mathbf{T}][\mathbf{B}_{\mathrm{nl}}]\,\mathrm{d}\Omega \tag{5.38}$$

Stiffening effects related to the stress in the body are accounted for in matrix $\mathbf{T}$ as follows:

$$[\mathbf{T}] = \begin{bmatrix} \sigma_{11} & \sigma_{12} & 0 & 0 \\ \sigma_{21} & \sigma_{22} & 0 & 0 \\ 0 & 0 & \sigma_{11} & \sigma_{12} \\ 0 & 0 & \sigma_{21} & \sigma_{22} \end{bmatrix} \tag{5.39}$$

The tangent of the material response is denoted by $\mathbf{C}_{\mathrm{tot}}$ and contains the effects of straining, rotating, and the change of the volume on a point:

$$[\mathbf{C}_{\mathrm{tot}}] = [\mathbf{C}^{\mathrm{J}}] - [\mathbf{C}_{\mathrm{spin}}] + [\mathbf{C}_{\mathrm{vol}}] \tag{5.40}$$

where $\mathbf{C}^{\mathrm{J}}$ is the tangent of the stress-strain relation, $\mathbf{C}_{\mathrm{spin}}$ contains terms related to the spin, and $\mathbf{C}_{\mathrm{vol}}$ accounts for the volume change at the position of a point. The first matrix is defined by the stress-strain relation chosen. The latter two matrices are defined as:

$$[\mathbf{C}_{\mathrm{spin}}] = \begin{bmatrix} 2\sigma_{11} & 0 & \sigma_{12} \\ 0 & 2\sigma_{22} & \sigma_{12} \\ \sigma_{12} & \sigma_{12} & \frac{1}{2}(\sigma_{11}+\sigma_{22}) \end{bmatrix} \qquad [\mathbf{C}_{\mathrm{vol}}] = \begin{bmatrix} \sigma_{11} & \sigma_{11} & 0 \\ \sigma_{22} & \sigma_{22} & 0 \\ \sigma_{12} & \sigma_{12} & 0 \end{bmatrix} \tag{5.41}$$

5.3.4 Stabilisation Matrices

In this section an updated Lagrangian implementation of the stabilisation scheme as given in Chapter 4 is proposed. The goal of the scheme is to stabilise the method with limited computational cost. Note that index i to indicate a node is dropped for readability.

The total stiffness matrix $\mathbf{K}$ and force vector $\mathbf{F}$ including the stabilisation terms become:

$$[\mathbf{K}] = [\mathbf{K}_{\text{int}}] + [\mathbf{K}_{\text{stab}}] \qquad \{\mathbf{F}\} = \{\mathbf{F}_{\text{int}}\} + \{\mathbf{F}_{\text{stab}}\} \tag{5.42}$$

where $\mathbf{K}_{\text{stab}}$ is the stabilisation stiffness matrix which is defined as:

$$[\mathbf{K}_{\text{stab}}] = \kappa([\mathbf{K}_{\xi}] + [\mathbf{K}_{\eta}]) \tag{5.43}$$

Matrices $\mathbf{K}_{\xi}$ and $\mathbf{K}_{\eta}$ are implemented according to Chapter 4 as follows:

$$[\mathbf{K}_{\xi}] = [\mathbf{B}_{\xi}]^{\text{T}}[\mathbf{C}_{\text{stab}}][\mathbf{B}_{\xi}] \int_{\Omega} \xi^2 \, \mathrm{d}\Omega \tag{5.44}$$

$$[\mathbf{K}_{\eta}] = [\mathbf{B}_{\eta}]^{\text{T}}[\mathbf{C}_{\text{stab}}][\mathbf{B}_{\eta}] \int_{\Omega} \eta^2 \, \mathrm{d}\Omega \tag{5.45}$$

The goal now is to define a stabilisation matrix $\mathbf{C}_{\text{stab}}$ that stabilises the method without causing volumetric locking. Choosing the constitutive tangent as stabilisation tangent ($\mathbf{C}_{\text{stab}} = \mathbf{C}^{\text{J}}$) adds two additional volumetric constraints per node. A small test in incompressibility or a simple constraint count demonstrates that the resulting method is locking. Clearly, the volumetric part of $\mathbf{C}^{\text{J}}$ should be removed from $\mathbf{C}_{\text{stab}}$ or at least decreased to a size in which it is not over-constraining the system. The latter option is chosen in the current research: not the complete volumetric tangent is used, but only a scaled down part of it. The proposed stabilisation matrix is defined as:

$$[\mathbf{C}_{\text{stab}}] = [\mathbf{C}^{\text{J}}_{\text{dev}}] + \frac{\text{diag}[\mathbf{C}^{\text{J}}_{\text{dev}}]}{\text{diag}[\mathbf{C}^{\text{J}}_{\text{vol}}]}[\mathbf{C}^{\text{J}}_{\text{vol}}] \tag{5.46}$$

where operator diag[ ] sums the terms on the diagonal of a matrix and $\mathbf{C}^{\text{J}}_{\text{dev}}$ is defined according to Equation (4.42). The volumetric part is found as follows:

$$[\mathbf{C}^{\text{J}}_{\text{vol}}] = [\mathbf{C}^{\text{J}}] - [\mathbf{C}^{\text{J}}_{\text{dev}}] \tag{5.47}$$

The stabilisation force vector is computed per increment as follows:

$$\{\mathbf{F}_{\text{stab}}\} = [\mathbf{K}_{\text{stab}}]\{\Delta\mathbf{d}\} \tag{5.48}$$

As a result, the stabilisation is a simple linear computation within the increment.

5.3.5 Newton–Raphson

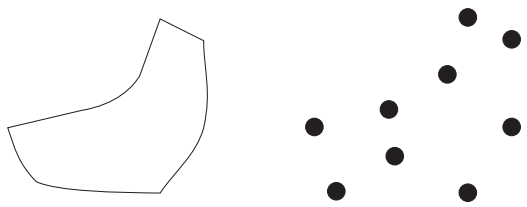
In the sections above, the force vector $\mathbf{F}$ and the accompanying continuum tangent were defined. The Newton–Raphson strategy uses a number of iterations, in order to find a set of displacement degrees of freedom for which the unbalance in the force vectors is of acceptable magnitude. Index k, which previously referred to the number of the increment, is dropped in order to improve the comprehensibility. In formula form, the iterative strategy within one increment is defined as:

$$\begin{aligned} \text{find } \{\Delta\mathbf{d}\} \text{ such that} \quad & \frac{\|\{\mathbf{F}\} - \{\mathbf{F}_{\text{ext}}\}\|}{\|\{\mathbf{F}\}\|} = e \leq e_{\text{tol}} \\ \text{by solving for every } m\text{:} \quad & [\mathbf{K}]\{\Delta\mathbf{d}\}^m = \{\mathbf{F}\} - \{\mathbf{F}_{\text{ext}}\} \\ \text{where:} \quad & \{\Delta\mathbf{d}\} = \sum_{m=1}^{N_{\text{iter}}} \{\Delta\mathbf{d}\}^m \end{aligned} \tag{5.49}$$

where m is the index of the current iteration, e_{tol} is a user-defined tolerance on the residual and N_{iter} is the number of iterations which is not known a priori. For each increment, the procedure as given above is repeated. If the relative residual e is smaller than e_{tol}, the scheme converged and the next increment is considered.

5.3.6 Triangles and Cells

For the shape functions, a Delaunay triangulation of the cloud of nodes is required. For the nodal averaging, a tessellation is used in order to nodally average the rate of deformation. For each time increment in the process, this triangulation and tessellation have to be computed. Computer algorithms for these purposes are widely available [27, 103], although they cannot be used straight away. Below, two problems associated with these algorithms will be addressed and a solution to them will be proposed. See Figure 5.3 for an example geometry to be analysed and a cloud of nodes which is used to discretise this geometry.



(a) model (b) cloud of nodes

Figure 5.3: A model and a cloud of nodes of that model.

Triangulations

The first aspect that requires attention is the Delaunay triangulation algorithm [40]. The triangulation of a cloud of nodes always gives a convex triangulation. Therefore, for convex bodies, triangulation algorithms can be used straight away. However, for practical problems in which concave boundary sections are present, the algorithm has to be modified in order to correctly represent these sections. Figure 5.4 shows the Delaunay triangulation of a cloud of nodes which clearly includes a part outside of the intended domain as shown in Figure 5.3(a).

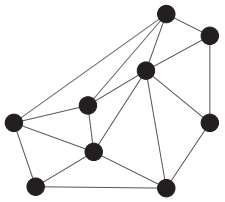


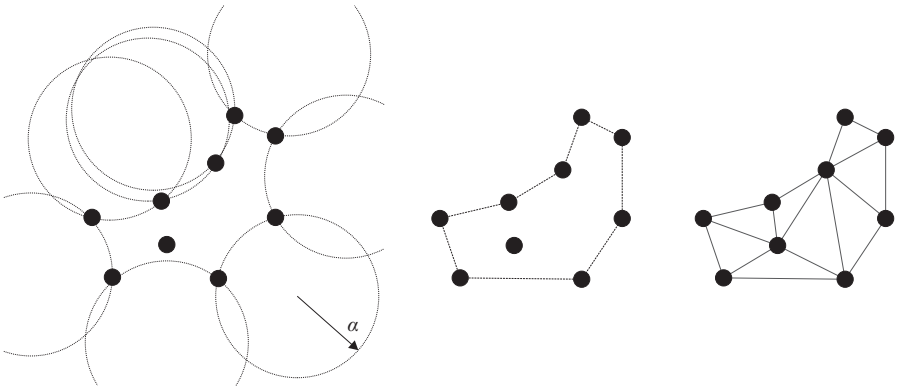
Figure 5.4: A Delaunay triangulation of the cloud of nodes.

A commonly used technique for defining a boundary on an arbitrary cloud of nodes is the method of α-shapes by Edelsbrunner *et al.* [48]. A ball with a radius of size α is rolled along the cloud of nodes and all nodes that touch the ball are on the boundary. The method has been introduced in the field of meshless methods by Cueto *et al.* [36]. Figure 5.5 gives an illustration of the method. The actual value of α has to be set by a user such that the intended contour is obtained. The α-shape criterion can easily be computed by using the Delaunay triangulation as given in Equation (3.18). If the radius of the circumscribed circle of a Delaunay triangle is larger than the value of α, this triangle is outside the domain. If $\mathbf{x}_i$, $\mathbf{x}_j$ and $\mathbf{x}_k$ are the three vertices of a Delaunay triangle and $\mathbf{x}_\mathrm{c}$ is the triangle its circumcentre, the body Ω is defined by the α-shape criterion as follows:

$$\begin{cases} V_{ijk} \in \Omega & \text{if} \quad d(\mathbf{x}_\mathrm{c}, \mathbf{x}_i) \leq \alpha \\ V_{ijk} \notin \Omega & \text{if} \quad d(\mathbf{x}_\mathrm{c}, \mathbf{x}_i) > \alpha \end{cases} \qquad \text{where:} \quad V_{ijk} = \mathrm{conv}\,(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \tag{5.50}$$

where conv contains all points in the convex hull of the three points in its argument list. All points in a Delaunay triangle that have a circumscribed circle with a radius larger than α are not in the body Ω. Triangles with a smaller radius are contained in the body. Hence this subdivision of internal an external points implicitly defines the boundary Γ. Figure 5.5(b) shows this boundary. Figure 5.5(c) shows the Delaunay triangulation that has been modified with the α shape criterion in order to represent the concave section properly.

For simulations in large deformations, the α shapes can be computed based on the Eulerian coordinates of the nodes, or on the Lagrangian coordinates. If the first



(a) circles with a radius of α (b) computed boundary (c) modified triangulation

Figure 5.5: An illustration of the α-shape criterion and the modified Delaunay triangulation.

strategy is chosen, nodes that were initially at the boundary Γ can be in the domain Ω at a later time step and vice versa. If the second method is used, all nodes on the boundary remain on the boundary and internal nodes remain internal. In Chapter 6, both two strategies will be used.

Tessellations

The Voronoi tessellation, as introduced by Voronoi [123], is a commonly used technique for subdividing a domain into a set of cells. In Section 3.2.5, a simplified tessellation technique was used in order to avoid difficulties related to the Voronoi tessellation. The cells were based on the centroids and midpoints of the Delaunay triangles.

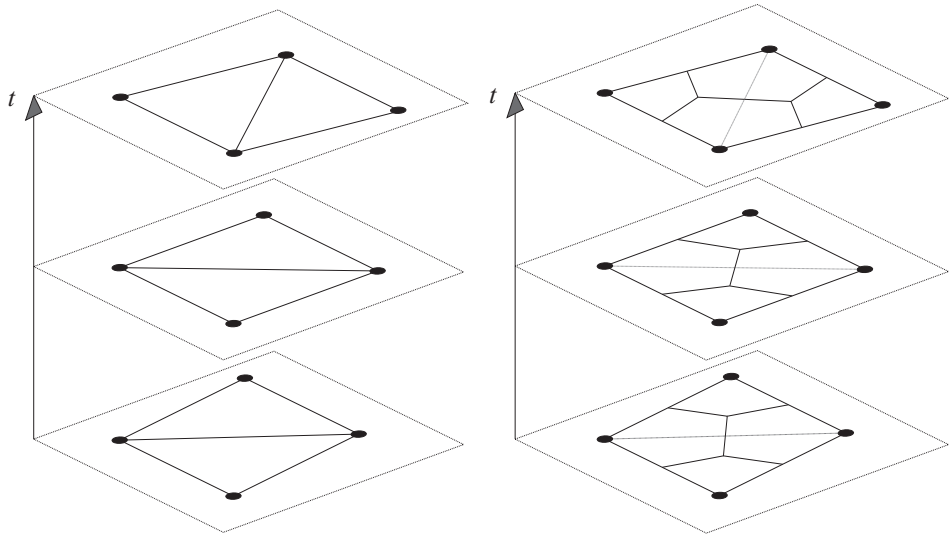
A downside of this technique is that if the nodes are positioned on a regular grid, the volume of the cells are unequally distributed as a result of the degenerate Delaunay triangulation. If nodes move such that the degenerate case is resolved, the volume of a cell accompanying a node can change drastically. See Figure 5.6 for an illustration on a degenerate Delaunay triangulation. Figure 5.7(a) shows a grid of nodes in simple shear such that a degenerate situation occurs. Figure 5.7(b) displays the centroid-based cells during the shearing. The volume of the cells changes if the degenerate case is 'passed'.

A Voronoi tessellation is independent of the degenerate cases of the Delaunay triangulation. Where the triangulation can be non-unique, the Voronoi tessellation is unique. However, constructing Voronoi cells at the boundary of a domain can be complicated. For nodes lying on the boundary of the domain, their accompanying cells can be undefined or have an infinite volume. A Voronoi tessellation including several undefined cells at the boundary is shown in Figure 5.8(a). In order to avoid

(a) an unique triangulation

(b) a degenerate triangulation; both triangulations are equally valid

Figure 5.6: A Delaunay triangulation of four nodes.



(a) Delaunay triangulation

(b) centroid based cells

Figure 5.7: An illustration of a Delaunay triangulation and the centroid-based cells during the pure shear of a 2x2 grid of nodes.

this issue, the Voronoi tessellation will be modified such that it always consists of properly defined cells. The modification consist of two steps which will be explained below.

Firstly, the vertices of a Voronoi cell will be modified. The vertices of the cells are defined by the circumcentres of the Delaunay triangulation. Equation (3.18) gives a definition of the circumcentre of a Delaunay triangle. The circumcentre of a Delaunay triangle can be located outside that triangle and thus also outside the domain. For this case a cell will overlap the boundary and trimming this cell on the boundary is a complicated task. To avoid this situation completely, the vertex of the cell for that triangle is not defined as the circumcentre, but as the midpoint of the triangle's side closest to the circumcentre. Two midpoints are added to the triangle's opposing facets such that the rest of the tessellation remains unaltered. As a result, no vertex point of a cell can be located outside the domain. Figure 5.8(b) shows this modification on the Voronoi diagram.

Secondly, cells that are on the boundary can be closed easily by adding the corresponding nodal location to the list of vertices of the cell. Figure 5.8(c) shows the modified tessellation with properly defined cells on the boundary. Figure 5.9 shows the alternating circumcentre-midpoint cells for the pure shear of the 2x2 grid of nodes.

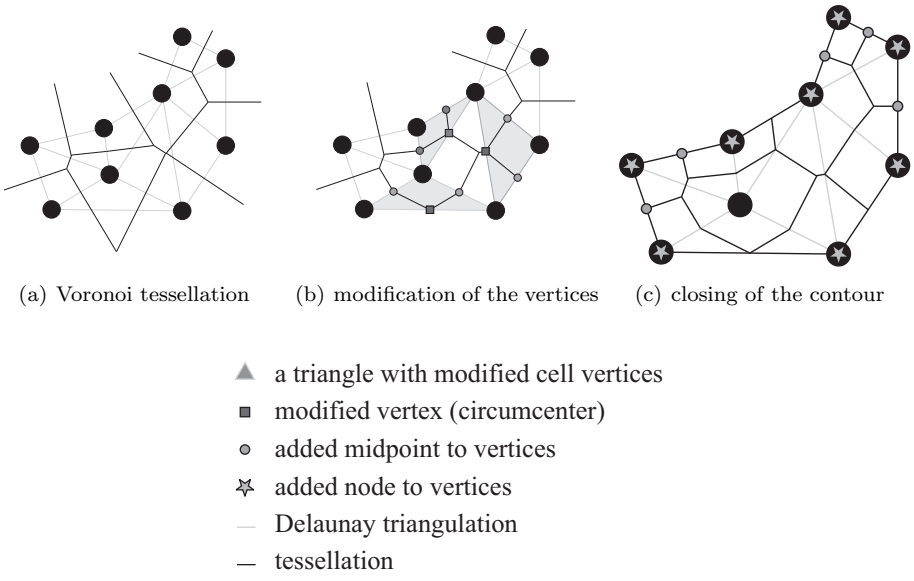


Figure 5.8: Modification of the Voronoi tessellation in order to correctly represent the boundary.

A downside of the tessellation algorithms as discussed above is that the volume assigned to the nodes is redistributed during the process. This redistribution will be discussed in brief below. Firstly a derivation is made for the case in which the

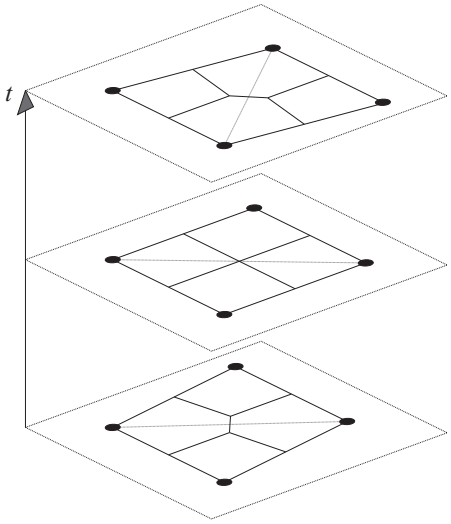


Figure 5.9: An illustration of the alternating circumcentre-midpoint cells.

volume of a cell is changing as a result of its Lagrangian motion. Thereafter the case is discussed in which the volume change is due to re-tessellation of the domain.

Assume that the volume of a cell is given by V and that the boundary of the cell is given by S. The volume of a cell can be computed by a contour integral along the boundary of a cell:

$$V = \int_S \mathbf{x} \cdot \mathbf{n} \, \mathrm{d}S \tag{5.51}$$

where $\mathbf{n}$ is the outward normal on the boundary of the cell. If the cell is moving according to the motion of the material, the contour integral as given above can be simplified. The contour, the coordinates and the outward normals are for this case a function of the Lagrangian coordinates, such that $S(\mathbf{x}(\mathbf{X}))$, $\mathbf{x}(\mathbf{X})$ and $\mathbf{n}(\mathbf{x}(\mathbf{X}))$. The volume of the cell can be simplified as follows:

$$V_{\mathbf{X}} = \int_{S(\mathbf{x}(\mathbf{X}))} \mathbf{x}(\mathbf{X}) \cdot \mathbf{n}(\mathbf{x}(\mathbf{X})) \, \mathrm{d}S \tag{5.52}$$

$$= \int_{V_0} \det(\mathbf{F}) \, \mathrm{d}V_0 \tag{5.53}$$

where symbol $V_{\mathbf{X}}$ is reserved for the case in which a cell follows a Lagrangian motion.

Secondly, the cells can be computed with the tessellation algorithm on an arbitrary time step. The input of the algorithm is the current location of the nodes and the output is a list with vertices of a cell. Consequently, the contour and the normal of a cell depend only on the Eulerian nodal coordinates. The dependency of the

contour and the normal on the nodal coordinates is expressed as $S(\mathbf{x}(\mathbf{x}_i))$ and $\mathbf{n}(\mathbf{x}(\mathbf{x}_i))$ respectively. The volume of a cell becomes:

$$V_{\mathbf{x}} = \int_{S(\mathbf{x}(\mathbf{x}_i))} \mathbf{x} \cdot \mathbf{n}(\mathbf{x}(\mathbf{x}_i)) \, \mathrm{d}S \tag{5.54}$$

The symbol $V_{\mathbf{x}}$ is used for the case in which the cell is determined only by the tessellation algorithm. This subdivision of the volume is purely algorithmic and does not incorporate the kinematics of the body on which it is used. The contour and the normal do not necessarily move in a Lagrangian motion. The circumcentre of a triangle for instance is not a Lagrangian point. The derivation given previously in Equation (5.53) can not be made. As a result, there can be a flux ΔV over the edges of a cell:

$$V_{\mathbf{X}} = V_{\mathbf{x}} + \Delta V \tag{5.55}$$

This flow term can be non-zero, depending on the displacements of nodes $\mathbf{x}_i$. For instance, if nodes move uniformly (a rigid body mode), the nodal flux ΔV is trivially zero. However, for non-uniform displacements this is not necessarily the case. Figure 5.10 illustrates the change of the geometry of a cell in time. For the current study, the nodal volume will be based on the tessellation algorithm after each converged increment.

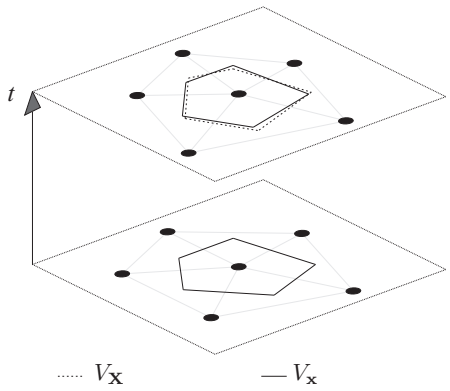


Figure 5.10: An illustration of the difference in the cell definition $V_{\mathbf{x}}$ or $V_{\mathbf{X}}$.

5.4 Numerical Examples

The goal of the ASFEM method is to solve forming processes in an efficient manner. However, before moving on to simulate these processes, several basic aspects of the method will be examined. For any numerical scheme used to simulate large deformations of solids, the following aspects need to be investigated in order to prove the scheme's physical validity:

- Since the rate of deformation is used as a measure of strain, the logarithmic strain should be obtained in uniaxial tension.
- There should be no straining in rigid rotation.
- Stresses should be rotated correctly in rotation.
- The tangent should give acceptable convergence for increments of reasonable size.

These four points will be addressed in the tests presented below. Firstly, in Section 5.4.1, a non-linear patch test will be performed in which the measure of strain and the convergence properties are examined. Secondly, the scheme is investigated in finite rotations by straining and rotating a tensile bar in Section 5.4.2. Finally, an illustrative example on the functioning of the method is given in Section 5.4.3. For all results presented in this section the stabilisation is on ($\kappa = 1.0$) unless stated otherwise. The material is assumed hypo-elastic and the 3d case is simplified to 2d by using the plane strain assumption.

5.4.1 Non-Linear Patch Test

Figure 5.11 shows the set-up for the patch test. The left-hand side of the geometry is supported. The right-hand side of the square workpiece is prescribed by a horizontal displacement. The sample is stretched by half its original length. As a result of this large deformation, the strain measure will make the problem non-linear and hence several iterations have to be performed. The accuracy of the strain approximation and the convergence characteristics are examined by running four simulations in which the the total load step will be divided over 1, 2, 4 and 8 increments respectively. The accuracy of the approximated strain will be compared with the exact strain tensor, which can be easily computed for this problem. To investigate the convergence, the trend of the residual e for subsequent iterations is examined. The tolerance to stop the iterations is set to $e_{\text{tol}} = 10^{-10}$.

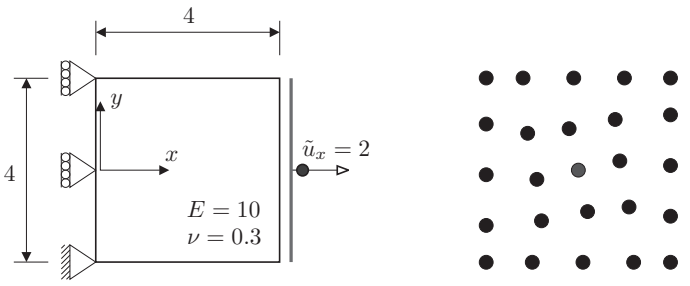
The exact logarithmic strain in x-direction of this problem is easily found because of the simple geometry and the loading condition. It can be computed by integrating the rate of deformation over the total simulation time as follows:

$$\varepsilon_{xx} = \int_0^t D_{xx}\,\mathrm{d}t = \int_0^t \frac{\tilde{v}_x}{L_0 + \tilde{u}_x}\,\mathrm{d}t = \ln\left(\frac{L_0 + \tilde{u}_x}{L_0}\right)$$

where ε_{xx} is the logarithmic strain and the initial length of the sample is given by L_0. Filling in the known quantities results in a logarithmic strain of $\varepsilon_{xx}^{\text{exact}} = 0.405465108$. The error between this exact strain and the approximated strain is defined as:

$$e_\varepsilon = \frac{\|\varepsilon_{xx} - \varepsilon_{xx}^{\text{exact}}\|}{\|\varepsilon_{xx}^{\text{exact}}\|} \times 100\% \tag{5.56}$$

Table 5.1 shows the error in percentages for the non-linear patch test. It can be seen that by subdividing the total load over multiple increments, the error in the strain



(a) model (b) set of nodes

Figure 5.11: The model and set of nodes for the non-linear patch test.

approximation is decreased and starts to approximate the exact strain. Moreover, even for a large loading step, the error in the strain is only around 1%. Table 5.2 gives the values of the stress σ_{xx} for five nodes of the grid. Irrespective of the non-uniform distribution of the nodes, the patch test is satisfied. The deviations between the values of the stress are of the same magnitude as the machine precision.

Table 5.1: The error in the logarithmic strain for the non-linear patch test in percentages.

N_{incr}	1	2	4	8
$e_\varepsilon[\%]$	1.33898	0.3514	0.08215	0.01482

Table 5.2: The stress σ_{xx} for five nodes of the grid.

node	position (x, y)	σ_{xx}
11	$(2.0, 0.0)$	4.4531231680510
12	$(2.2, 1.2)$	4.4531231680510
13	$(2.0, 2.0)$	4.4531231680509
14	$(1.8, 2.8)$	4.4531231680509
15	$(2.0, 4.0)$	4.4531231680509

The convergence behaviour of the method is examined in the following test. Similarly to the above, four simulations are performed, each with a different number of increments over which the total load is divided. The residual e and the logarithm of e will be plotted against the iteration number for one increment out of each simulation. Figure 5.12 shows the results. It can be seen in Figure 5.12(a) that

the tolerance on the residual e_{tol} is reached more quickly by increasing the number of increments. However, Figure 5.12(b) shows that, irrespective of the number of increments, quadratic convergence is not obtained. Nonetheless, it can be stated that the algorithm converges well for a limited set of increments.

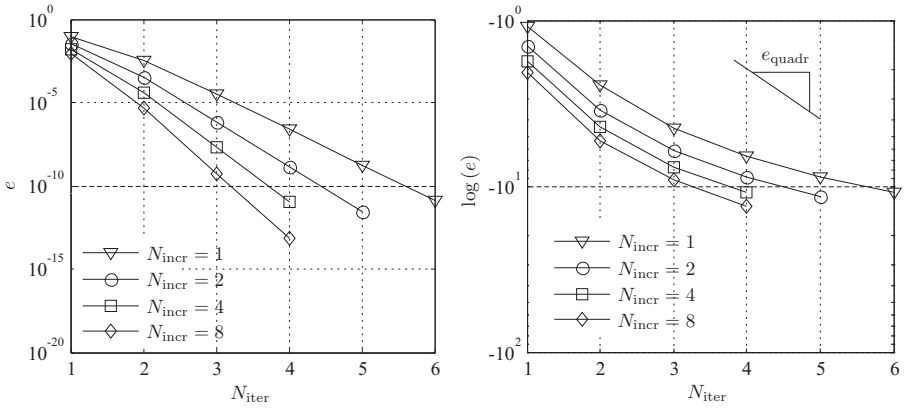
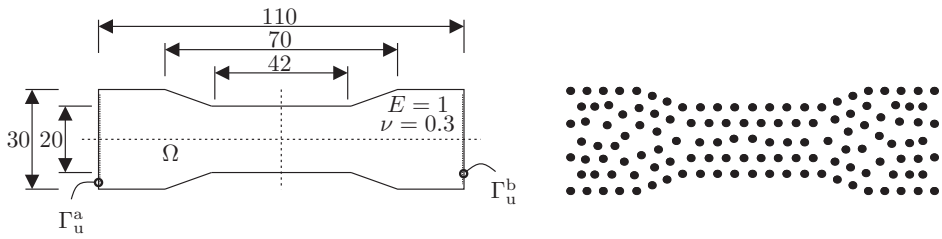


Figure 5.12: Convergence plots of the non-linear patch test for various number of increments. Line e_{quadr} indicates the slope of optimal quadratic convergence.

5.4.2 Rotation of a Tensile Bar

In this test several aspects of the code relating to rotations are examined. Firstly, the assumed rate of deformation as defined in Equation (5.11) should be zero or at least reasonably close to zero in rigid rotations. Secondly, the Cauchy stress tensor should be objective in rigid rotations. In this example it will be demonstrated that these two conditions are met. Figure 5.13 shows the tensile bar problem. Figure 5.13(a) and 5.13(b) show the geometry and the node set respectively. The Young's modulus and the Poisson's ratio for the elastic material are $E = 1$ and $\nu = 0.3$ respectively. The tolerance for the simulations is set to $e_{\text{tol}} = 10^{-6}$.

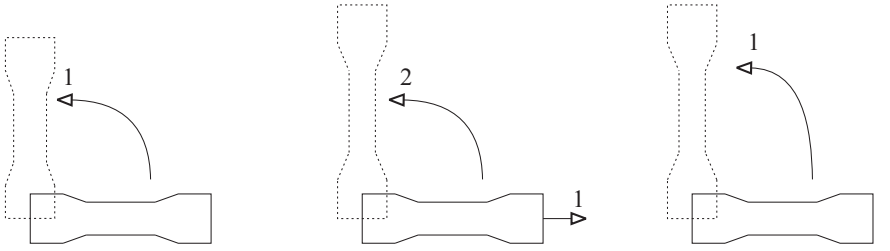


(a) geometry (b) cloud of nodes

Figure 5.13: The model of the tensile bar.

Three simulations will be performed on the tensile bar, each with a different loading path. These different loading stages are shown in Figure 5.14. Firstly, the tensile bar will be rotated 90 degrees by prescribing the left-hand side of the bar ($\Gamma_{\text{u}}^{\text{a}}$) only. This test is performed to check whether there is any nonphysical straining in rotation. Secondly, a test will be performed in which the bar is first stretched and then rotated. The displacements at both the left-hand side and the right-hand side of the bar are prescribed. Stresses that are introduced in the body during stretching should be rotated correctly to the final configuration. Finally, the bar will be stretched and rotated simultaneously. The simultaneous stretching and rotating of the bar should give the same final result as the previous test in which these two deformations are applied sequentially. For all three simulations the prescribed rotation is 90 degrees counterclockwise. For the latter two tests, the bar is stretched by 16.5 length units in total. The rotational load is applied over 5 increments. The stretching load is applied in 3 increments. The combined stretching and rotation load, as used for the second test, is distributed over 5 increments.

The results for the three simulations are given in Figure 5.15. The components of the stress tensor are plotted in the global coordinate axis. The contour plots are given for the first and third simulation after the first step (Figure 5.15(a) and Figure 5.15(c)), and for the second simulation after the first and the second loading stage (Figure 5.15(b)). It can be seen that for test one, in the case of a pure rotation



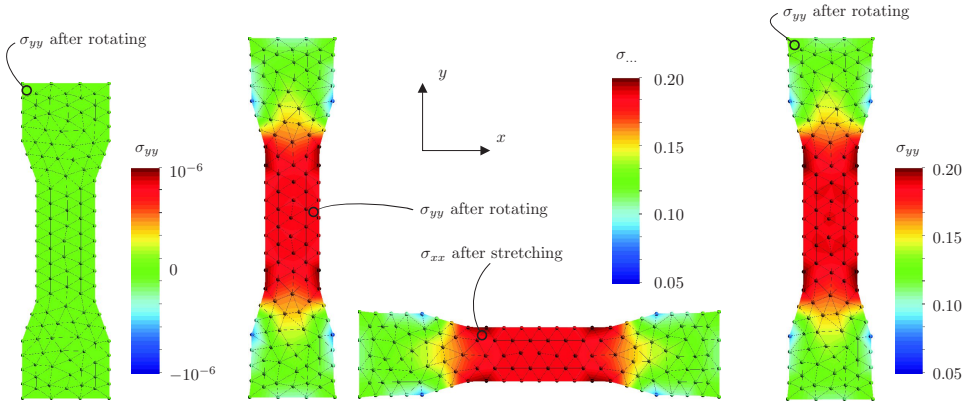
(a) load case 1: rotating without stretching

(b) load case 2: stretching before rotating

(c) load case 3: stretching and simultaneous rotating

Figure 5.14: Three load cases for the tensile bar problem.

without any prescribed stretching, the body remains free of stress. Moreover, the geometry of the sample is equal in rotated and un-rotated configuration. The contour plots of the second and third test show that, independent of the order of the loading steps, the results are equal. It can be concluded that the stress and strain measures are accurately approximated in rotations and stretching, independent of the order in which the loading steps are applied.



(a) load case 1

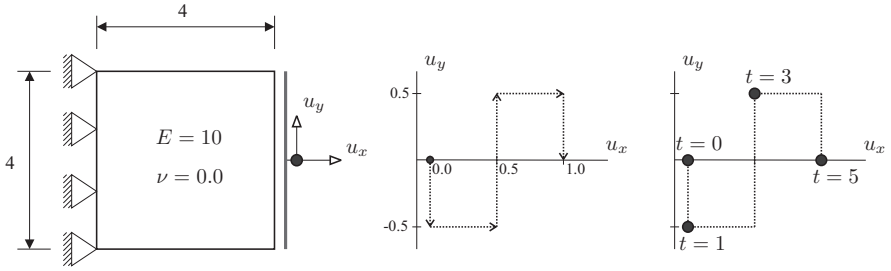
(b) load case 2

(c) load case 3

Figure 5.15: Contour plots of the stress for the three load cases.

5.4.3 Non-Proportional Loading

The current test will give an illustration of the Delaunay triangulation algorithm that is running for every increment in the ASFEM method. Figure 5.16(a) shows the model. The left-hand side of the block is clamped. The right-hand side of the square is prescribed according to the displacement pattern given in Figure 5.16(b). The problem is solved in 30 increments divided over the total loading path.



(a) workpiece model (b) non-proportional loading path

Figure 5.16: The non-proportional loading of a square block.

The computation of the material response, which for finite elements is carried out per integration point, is carried out per node for the nodal smoothed FEM. Since nodal locations are Lagrangian, no mapping algorithm is used for the simulation. Figure 5.17 shows a contour plot of the σ_{xx} stress for the same four deformed states. Although the triangulation altered throughout the simulation, the spatial distribution of the stress is reasonably smooth. Do note that because of the path dependency of the strain measure, the shear stress, for instance, is not equal to zero for this loading path.

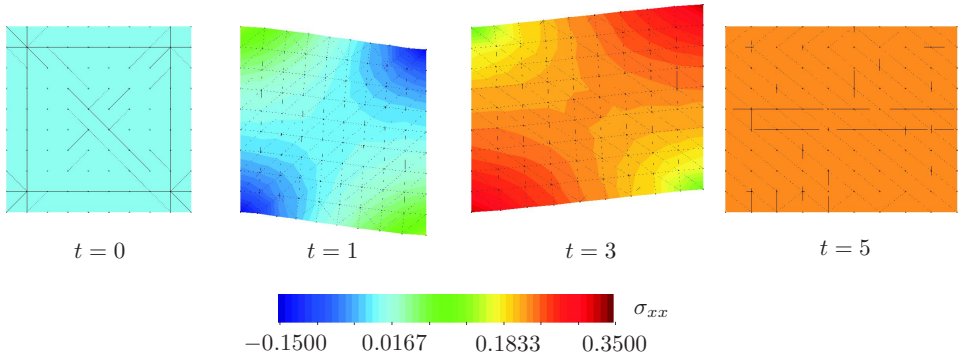


Figure 5.17: Contour plots of the stress σ_{xx} for four deformed states of the square block.

How the triangulation altered can be seen in Figure 5.18. The triangulation is depicted on the deformed states of the workpiece for 4 frames out of the total simulation, as indicated in Figure 5.16(b). In the figure the current coordinates of the nodes and the current triangulation are plotted. It can be seen that during deformation, the triangulation alters. The triangulation changes according to the deformation pattern such that it is optimal in the sense that aspect ratios of the triangles are minimal.

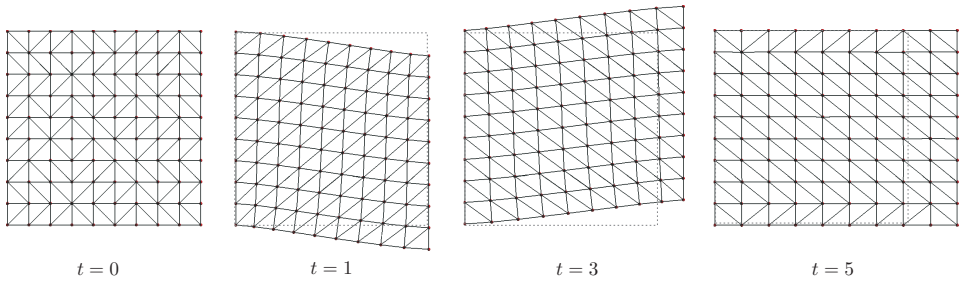


Figure 5.18: A plot of the deformed configuration and the Delaunay triangulation for four deformed states of the square block.

5.5 Closure

In this chapter, a smoothed finite element method with adaptive shape functions was introduced and validated. Two of the main constituents of the method, namely the shape function and the strain smoothing were presented for large deformations. Incremental approximations were introduced in order to avoid issues related to convergence in a Newton–Raphson process. Since the shape function is defined upon a triangulation and the strain smoothing is defined upon a tessellation, these geometrical concepts were elaborated. The Delaunay triangulation and Voronoi tessellation were used as starting point and were modified in order to correctly represent the boundary of the domain.

In several examples, the performance in geometrical non-linearity was investigated. Various combinations of stretching and rotating of a tensile bar were analysed in order to verify the correctness of the proposed method. The convergence of the method was shown to be good though not quadratic. A final example showed the alternating triangulation during non-proportional loading of a square workpiece. Large deformations were simulated and the re-triangulation of the domain took place without mapping the material model data.

6

Applications

6.1 Introduction

In this chapter, two forming processes will be simulated with the ASFEM method proposed in the previous chapter. Finite element simulations of the same two processes will be used as a reference. The goal of the two applications presented in this chapter is to examine the performance of the ASFEM method.

The first forming process that will be simulated is the forging of a circular rod. A history-dependent material model is chosen to model the constitutive behaviour. The second forming process that is simulated with the ASFEM method is the extrusion of an aluminium profile. This process should serve as an ideal benchmark for the newly developed method because of the extreme deformations that take place. Both simulations are in 2D and are simplified by using a plane strain assumption and by assuming the deformation to be isothermal. Since the simulations will involve the plastic flow of material, an plasticity algorithm is required. For the research as presented in this chapter, an implicit J2 radial return plasticity algorithm will be used in combination with a von Mises yield surface. The formulations of this algorithm can be found in Simo and Hughes [111], among others. For both simulations, the mesh will be revised on every time step. In order to describe contact between the deformable body and the tools, a simple penalty formulation is chosen. The tools are modelled by rigid frictionless analytical bodies.

For the FEM computations, the in-house developed finite element computer code DiekA [41] is used. This code is well-suited for simulating large deformation problems in combination with implicit time integration. For the ASFEM computations, a newly developed code, named CruncH is used. This code incorporates the formulations presented in Chapter 5. CruncH is mainly based on the open-source finite element code FeaTure [49] and uses the Qhull [103] library for its triangulation algorithm. All results presented in this chapter are post-processed in GiD [54].

In the post-processing stage, the contour plots of the finite element simulation are constructed as follows. Firstly, the integration point values of, for instance, the stress

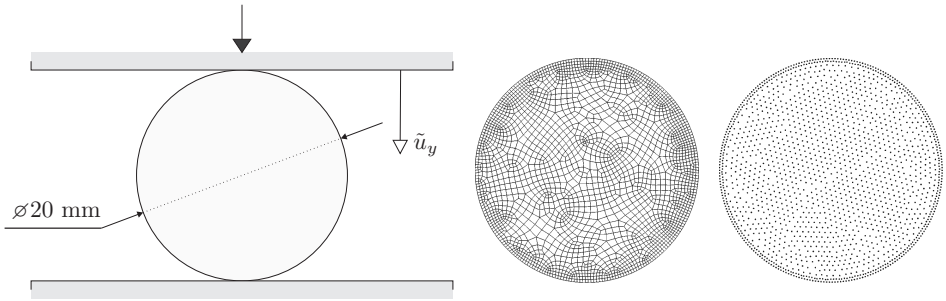
or strain are extrapolated to the nodes. Secondly, these values are averaged for each node. This 'nodal averaging' is a commonly used technique in order to obtain a smooth field from integration point data which is usually less smooth. The formulation can be found in most books on finite element analysis; for instance in Cook *et al.* [35]. The contour plots of the ASFEM method are constructed by plotting the nodal quantities directly on the nodes and interpolating the values over the triangles. There is no smoothing operation in the post-processing stage. This difference should be borne in mind when comparing FEM and ASFEM results.

The extrusion application presented in this chapter has been analysed previously with the method of Smooth Particle Hydrodynamics (SPH) and the Element-Free Galerkin (EFG) method by Quak *et al.* [106]. For preliminary results of the application of ASFEM and ALE finite elements on friction stir welding, the reader is referred to van der Stelt *et al.* [121]. An application of the ASFEM method on the indentation of a square workpiece was performed by Quak and van den Boogaard [104].

6.2 Forging of a Circular Rod

Model

In this application, a circular rod will be compressed between two flat rigid tools. The rod is compressed until its final height is 50% of its original diameter of 20 mm. Figure 6.1 shows the geometry of the setup and the two models used to simulate the forging. The bottom tool will be fixed in space. The tool at the top will be lowered by 10 mm. The friction between the tools and the deformable body is neglected and the simulation is assumed isothermal.



(a) geometry (b) FEM model (c) ASFEM model

Figure 6.1: The forging model. The FEM model consists of 2392 elements and 2518 nodes. The ASFEM model starts with 1783 nodes

The finite element simulation is performed with linear four-node quadrilateral finite elements by using DIEKA. The quad4 elements employ a mean dilatation formulation

in order to counter volumetric locking, and are used in a Lagrangian formulation without re-meshing or ALE strategies. The mesh used for the simulation is depicted in Figure 6.1(b). For the ASFEM method, a nodal grid serves as the starting point for the simulation. This grid is depicted in Figure 6.1(c). Based on this node set, the contour is detected by the α-shape criterion at the first increment, and is maintained on all increments thereafter.

For the constitutive relation, an elasto-plastic material model is used with a von Mises yield surface and a Ludwik–Nadai hardening curve. This hardening curve is expressed as:

$$\sigma_{\mathrm{y}} = C\,(\varepsilon_0 + \varepsilon_{\mathrm{eq.}})^n \tag{6.1}$$

where σ_{y} is the flow stress and ε_0, C and n are parameters following from an experiment. For this application, an hypothetical low alloyed deformation steel is chosen of which the parameters are given in Table 6.1. The initial flow stress of the material is 200 MPa.

Table 6.1: Constitutive parameters for the forging simulation.

elastic properties	E	210 000	MPa
	ν	0.28	-
plastic properties	C	600	MPa
	ε_0	$2.137 \cdot 10^{-4}$	-
	n	0.13	-

Results

Firstly, the deformation patterns during the simulation are examined. Contour plots of the equivalent plastic strain rate for both methods will be compared at 25%, 37.5% and 50% compression of the rod. In these contour plots, the location and the rate at which the material is deforming plastically can be examined. Figure 6.2 shows six contour plots for the two methods on the three stages of compression. It can be seen that depending on the amount of compression, a number of shear bands appear. The FEM and ASFEM simulation show very good agreement. The patterns and also the magnitude of the equivalent plastic strain rate correspond well. The contour plot of the finite element model shows two shear bands in perpendicular setup at 25% compression. For the ASFEM simulation these bands are also visible, though they have already started splitting in order to arrive at the four-band configuration depicted in Figure 6.2(d). Figures 6.2(e) and 6.2(f) shows the equivalent plastic strain rate at the final stage of the process. A configuration with 6 shear bands is obtained. The bands of the ASFEM simulation appear to be sharper and more detailed when compared to the FEM bands which are more diffuse. This effect can be caused by the severe distortion of the finite element mesh, or the nodal averaging in the post-processing stage. However, there is good agreement in general. For more details on inhomogeneous deformation in compression tests, Rietman [109] can be read.

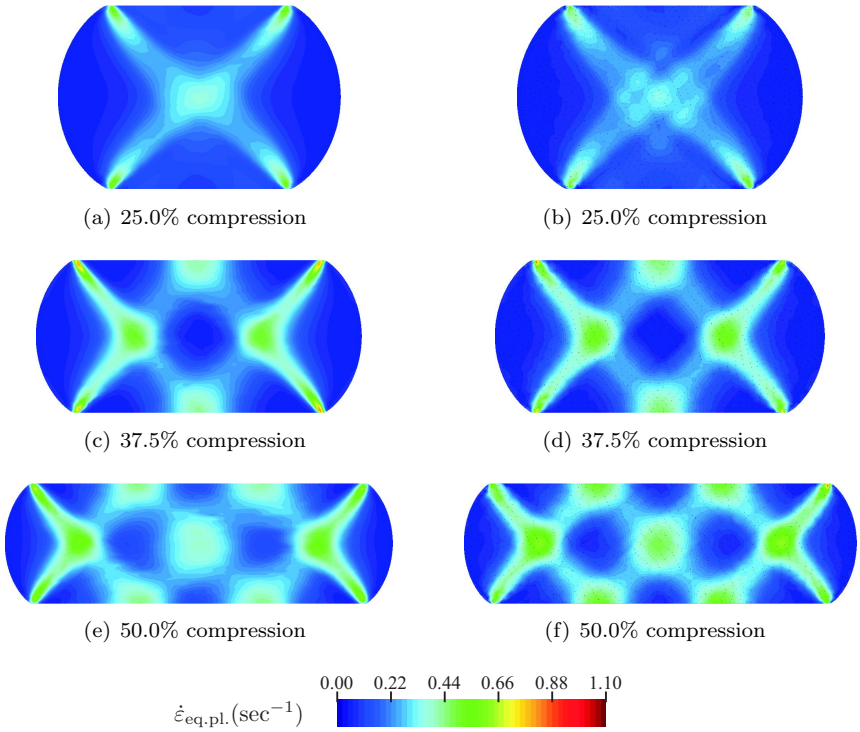


Figure 6.2: Contour plots of the equivalent plastic strain rate for the two methods for three steps in the process. On the left-hand side the FEM results. On the right-hand side the ASFEM results

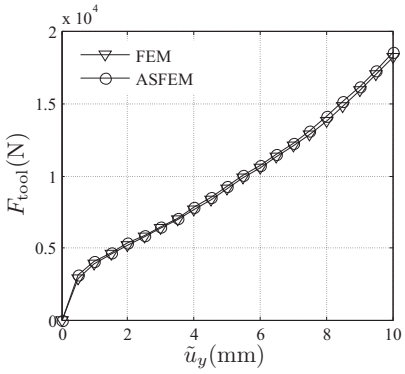
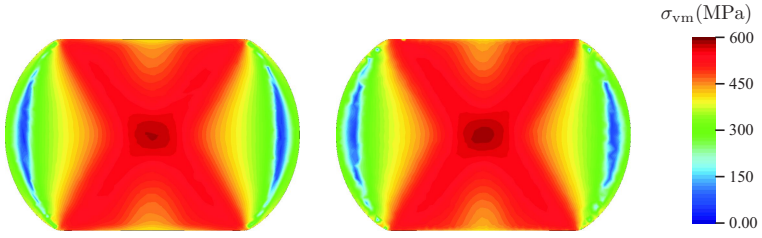


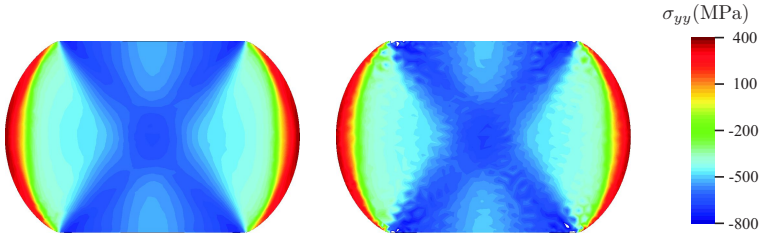
Figure 6.3: The tool force plotted versus its vertical displacement.

Secondly, the force required to compress the rod is examined. Especially for this application, in which a nearly incompressible material model is used, the force should not be nonphysically high as a result of volumetric locking. Figure 6.3 shows a graph of the force of the top tool versus its displacement. The curves of both methods correspond well.



(a) FEM (b) ASFEM

Figure 6.4: The von Mises stress for the two methods at 25% compression.

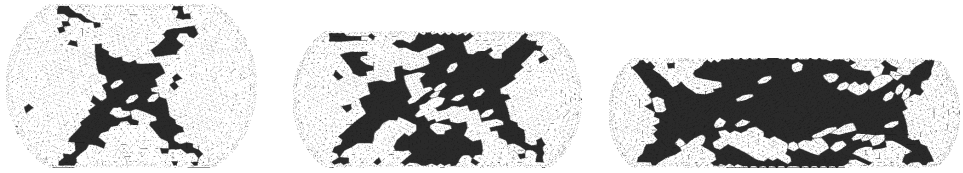


(a) FEM (b) ASFEM

Figure 6.5: The σ_{yy} stress for the two methods at 25% compression.

Thirdly, the quality of the stress prediction is examined. Generally, linear elements do not give an accurate stress prediction in the integration points. Therefore typically post-processing steps are employed in order to 'smoothen' these fields. For the finite element simulation this is a nodal averaging scheme. For the ASFEM simulation there are no post-processing operations. Figure 6.4 shows the von Mises stress for both simulations at the intermediate stage of 25% compression. The von Mises stresses compare well. The FEM field appears to be smoother, though as mentioned this can be a result of post-processing. Note that close to the free surface at the left and right-hand side of the compressed rod, a close to zero von Mises stress is predicted as a result of the outward bending of these regions. Whereas the material has deformed plastically almost everywhere, at these regions, the material remains elastic.

Figure 6.5 shows a plot of the σ_{yy} stress at the same configuration. Clearly, the stress prediction with the ASFEM method appears to be more rough, whereas the finite element results are smooth. The reason for the less smooth results for the



(a) 25.0% compression (b) 37.5% compression (c) 50.0% compression

Figure 6.6: An illustration of the change of connectivity between the steps depicted. Grey indicates an area where the triangulation altered.

ASFEM method is as follows. In Chapters 3 and 4 studies on the stability of nodally averaged methods were presented. The weak forms related to the deviatoric part and the volumetric part of the deformation were investigated. In order to stabilise the weak form related to the deviatoric part of the deformation, a stabilisation was proposed in Chapter 4. The volumetric part, which involves the pressure prediction, was shown to be unstable in the inf-sup test. Since the stress tensor depends both on the deviatoric stress and the pressure, an oscillatory field is retrieved, as can be seen in Figure 6.5(b). Do note, however, that quad4 plane strain finite elements also contain spurious pressure modes which can be very similar to the ones observed for the ASFEM method. However, this checkerboard mode is effectively erased by the post-processing step which averages the element values to nodal values. Based on the visualised results, the stress field appears to be smooth, though in reality this is not necessarily the case.

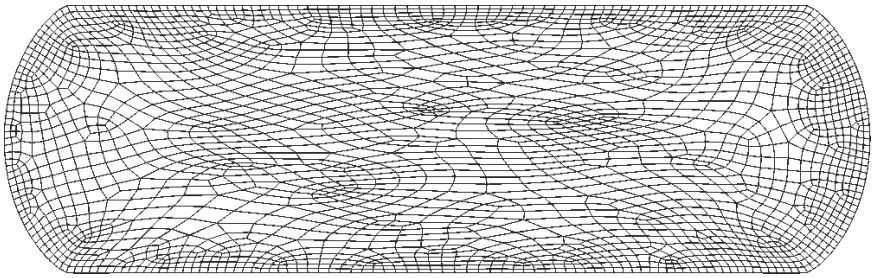
For the finite element simulation, one single mesh was used throughout the simulation. For the ASFEM method on the other hand, a new mesh was constructed for every increment of the simulation such that it was optimal for the cloud of nodes of that specific increment. In this paragraph, the change of the mesh will be investigated as follows. Between the configuration at the start of the simulation, and the configuration at 25% compression, the nodal connectivity has changed. Triangles that changed the nodal connectivity will be marked and the result will be plotted by means of a contour plot. Figure 6.6(a) shows the difference in triangulation between the 0% and 25% configuration in grey. Figures 6.6(b) and 6.6(c) depict the change in connectivity between the configuration at 25% and 37.5% compression and 37.5% and 50.0% compression respectively. In general it can be stated that in regions of large deformations, most triangles have been redefined. Figure 6.2(b), for example, shows the equivalent plastic strain rate at the configuration at 25% compression. Although this rate is determined over a single increment, whereas Figure 6.6(a) shows the differences over the configuration between 0% and 25%, it can be seen that the patterns observed in the two figures are comparable.

Figure 6.7 shows the mesh at the final step for the finite element and the adaptive smoothed finite element simulation. It is clear that the finite element mesh contains zones of badly shaped elements. The triangles of the ASFEM method appear to be reasonably shaped although a more quantitative comparison would be useful.

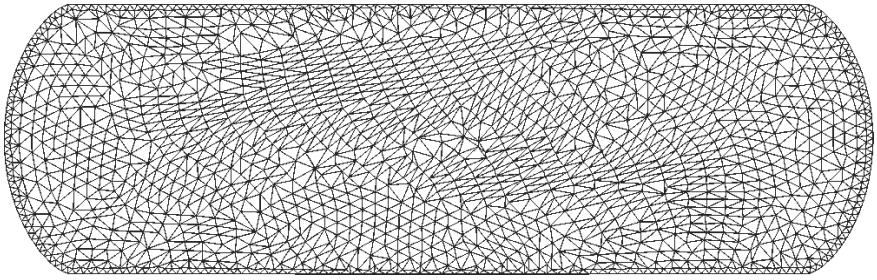
Therefore, an error criterion will computed for each triangle or quadrilateral in order to compare the quality of the two meshes. This criterion is defined similarly as in the GiD user manual [54] as follows:

$$e_{\text{tri}} = 1 - \frac{4 \cdot \sqrt{3} \cdot V_{\text{tri}}}{s_1^2 + s_2^2 + s_3^2} \qquad e_{\text{quad}} = 1 - \max_{i=1..4} \left\{ \frac{2 \cdot \|\mathbf{s}_{\text{l}}^i \times \mathbf{s}_{\text{r}}^i\|}{\|\mathbf{s}_{\text{l}}^i\|^2 + \|\mathbf{s}_{\text{r}}^i\|^2} \right\} \tag{6.2}$$

where the error criterion for a triangle and quadrilateral are given by e_{tr} and e_{quad} respectively, V_{tri} is the volume of the triangle and s_i is a side of the triangle. Vector $\mathbf{s}_{\text{l}}^i$ and $\mathbf{s}_{\text{r}}^i$ are vectors along the sides of the quadrilateral for node i. See Figure 6.8 for an illustration of the symbols used. For each shape, this criterion is computed. If the criterion is 0, the triangle or quadrilateral is optimally shaped. For a triangle this implies that it is an equilateral triangle. A quadrilateral is in this case a square. In the case e is 1, the shape is a sliver. For values between zero and one, the error measures can not be compared quantitatively. Therefore a qualitative comparison based on histograms is used.

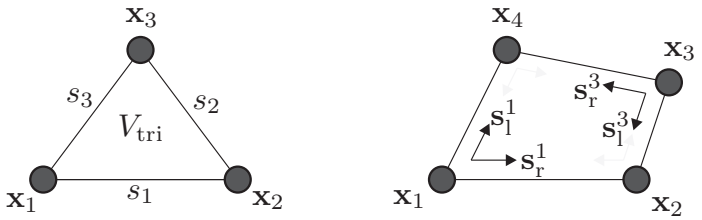


(a) FEM mesh



(b) ASFEM mesh

Figure 6.7: The mesh of the two methods at 50.0% compression.

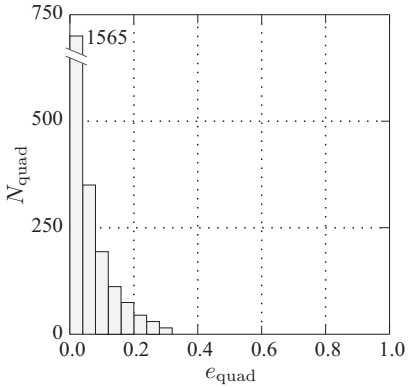


(a) triangle criterion (b) quadrilateral criterion

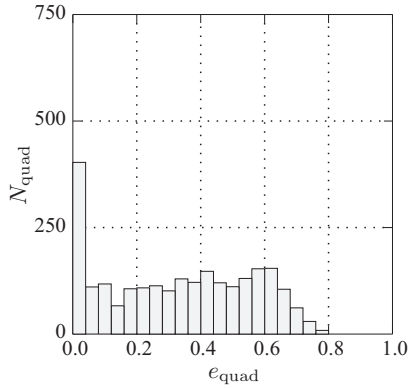
Figure 6.8: An illustration of the symbols used to compute the shape criterion.

Figure 6.9 shows the histograms for the FEM computation and the ASFEM computation for the initial configuration and the final configuration. As an illustration, a histogram is added for the case in which the ASFEM simulation is performed without using the re-triangulation on each time step. On the horizontal axis of the histogram the shape criterion is plotted with intervals of 0.04. The vertical axis gives the number of shapes within an interval. Firstly it can be seen that the initial meshes of the FEM and ASFEM simulation, as shown in Figures 6.9(a) and 6.9(c) respectively, are of good quality. Most shapes have a criterion of 0 and a few shapes have a higher value, although not higher that 0.35.

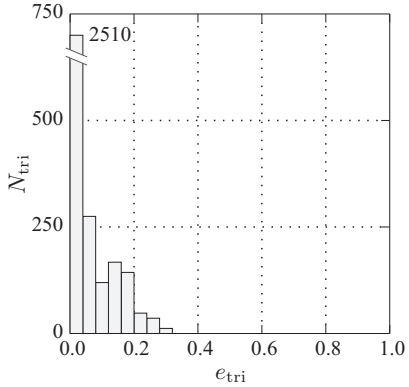
After the total deformation has been applied, it can be seen in Figure 6.9(b) that the shape of the finite elements is far from optimal. Shapes are present with a value of 0.8. The ASFEM results are given in Figure 6.9(d). Although the shape criterion used for quadrilaterals is different than the triangle criterion, it is possible to state that more triangles are located close to the optimal value of 0 in comparison with the FEM computation. If no re-triangulation is used for the ASFEM method, Figure 6.9(e) is obtained. Clearly, the quality of the mesh of this configuration is worse than for the simulation in which there were re-triangulations. Overall it can be stated that the ASFEM method uses the best shaped mesh.



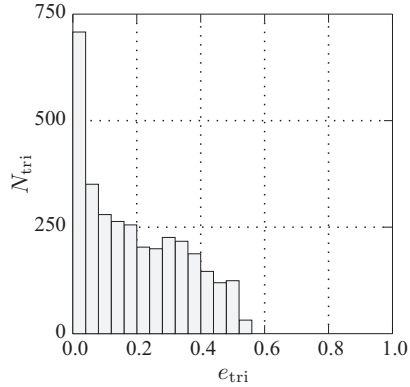
(a) FEM quadrilaterals at 0% compression



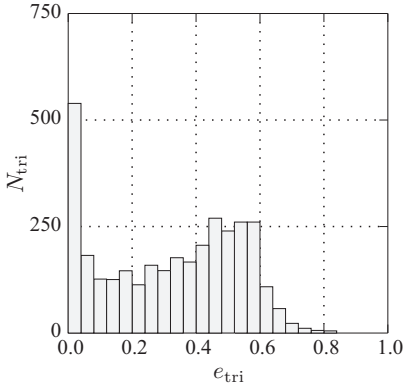
(b) FEM quadrilaterals at 50% compression



(c) ASFEM triangles at 0% compression



(d) ASFEM triangles at 50% compression



(e) ASFEM triangles at 50% compression (no re-triangulation took place during the simulation)

Figure 6.9: Histograms of the shape quality of the triangles and quadrilaterals.

6.3 Extrusion of Aluminium

The extrusion process is commonly used to produce aluminium profiles with arbitrary cross-sections. During the forming process, the deformations are generally of such an order that finite elements are not preferred in a Lagrangian formulation. Elements based on a Eulerian or ALE formulation are more successful since the elements are completely or partly decoupled from the deforming material. Mesh distortion problems are avoided completely or at least reduced. Examples of the modeling of the extrusion process with finite elements based on a Eulerian or ALE formulation can be found for instance in Lof [90], Koopman [72] and Assaad [11]. In this section, the ASFEM method will be compared with a Eulerian finite element simulation in a simplified extrusion problem.

Model

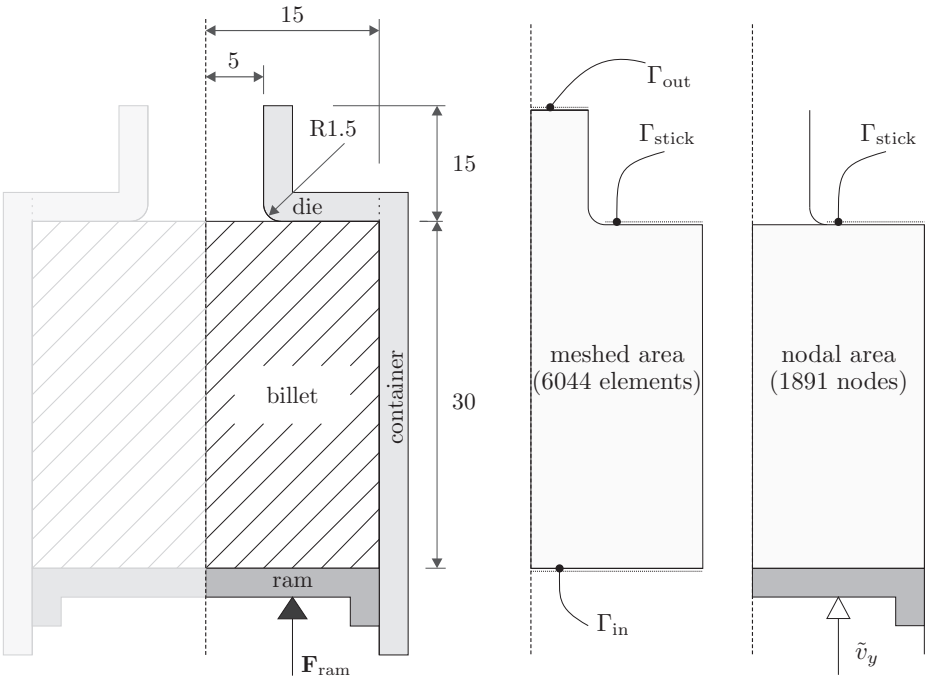
Figure 6.10 shows the model of the extrusion process. An aluminium billet with a width of 30 mm is reduced to a profile of 10 mm in width. A plane strain assumption applies to the model. On the dashed line, symmetry boundary conditions are enforced. The tools are modelled frictionless except for the top surface denoted by Γ_{stick}. Here a full stick boundary condition is applied. Material in the top right corner of the container is known to be fixed in its motion. The boundary condition on Γ_{stick} ensures the nearly zero flow in this corner.

Figure 6.10(b) shows the analysis domain for the finite element simulation. The finite element mesh is fixed in space and the material will move through this mesh. On the surface of the billet that is in contact with the container, boundary conditions are applied according to the description as given above. Along the bottom of the billet (the surface denoted by Γ_{in}), an inflow velocity of 1 mm/s is prescribed. At Γ_{out} there is a free outflow of material. The total simulation is divided into 200 time steps of 0.005 second, resulting in a total simulation time of 1 second. During this simulation time, the steady state solution is obtained and further simulation is superfluous.

The analysis domain for the ASFEM simulation is depicted in Figure 6.10(c). The ram will be moved upwards with a prescribed velocity of 1 mm/s. For the ASFEM model, the method of α-shapes is used for each increment of the simulation. Whereas for Section 6.2 the boundary was preserved on all time steps, this approach is not adequate as will be demonstrated later on in the results section. For the ASFEM computation, 400 time steps of 0.015 seconds each, are used. The resulting total simulation time is 6 seconds. Within this simulation time, a significant amount of the billet will have been extruded.

For the constitutive behaviour an elasto-viscoplastic material model is used with a Sellars–Tegart flow rule. This law is given by:

$$\sigma_{\text{y}} = \frac{1}{\alpha} \sinh^{-1} \left(\left(\frac{\dot{\varepsilon}_{\text{eq.}}}{A} \exp \frac{Q}{RT} \right)^{\frac{1}{n}} \right) \tag{6.3}$$



(a) geometry (b) finite element model (c) ASFEM model

Figure 6.10: Illustration of the geometry and models used to simulate the extrusion problem. All dimensions are given in millimetres.

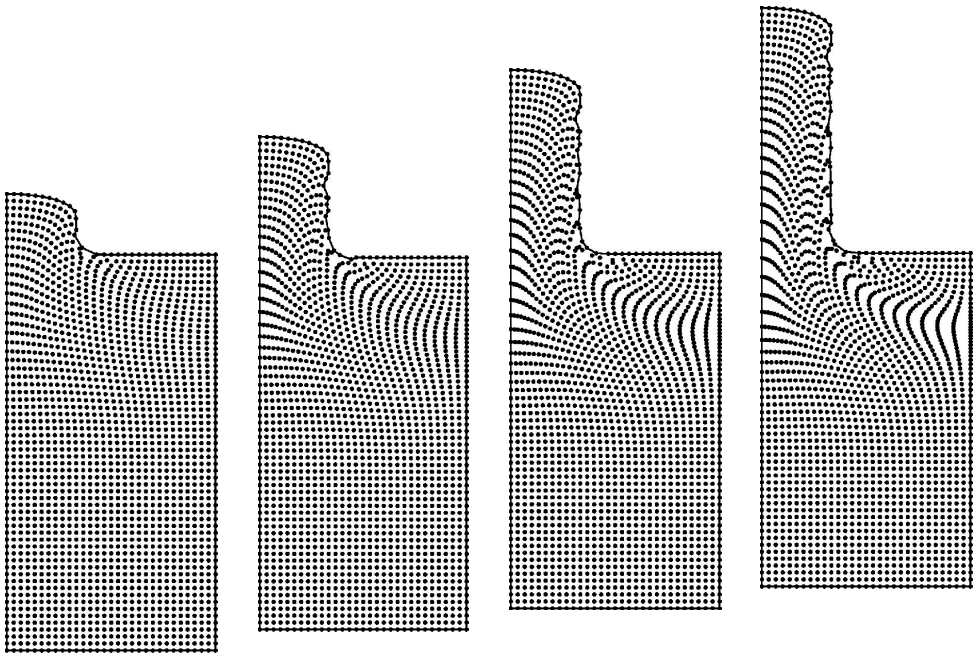
Table 6.2: Constitutive parameters for the extrusion simulation.

elastic properties	E	40 000	MPa
	ν	0.3	-
plastic properties	α	0.052	1/MPa
	Q	153 000	J/mol
	R	8.314	J/mol K
	T	700	K
	A	$2.39 \cdot 10^8$	1/sec
	n	2.976	-

where α, A, Q, R and n are variables following from an experiment, T is the absolute temperature and $\dot{\varepsilon}_{\text{eq.}}$ is the equivalent strain rate. The parameters for the aluminium billet are given in Table 6.2 and are taken from Assaad [11]. The temperature will be fixed throughout the process at 700 K.

Results

The deformed geometry, the boundary description for the ASFEM method, the velocity field, the ram force and the mass conservation will be investigated in this section. For the finite element simulation, the steady state situation is used for comparison. This steady state is reached after approximately one second of simulation time. The ASFEM simulation on the contrary, is a transient simulation. A specific time step out of the total simulation has to be chosen for the comparison. Therefore, the configuration at six seconds is used unless stated otherwise. Note that, looking for instance at contour plots of the equivalent plastic strain rate from a spatial Eulerian viewpoint, a steady state appears to be reached after around two seconds of simulation time.

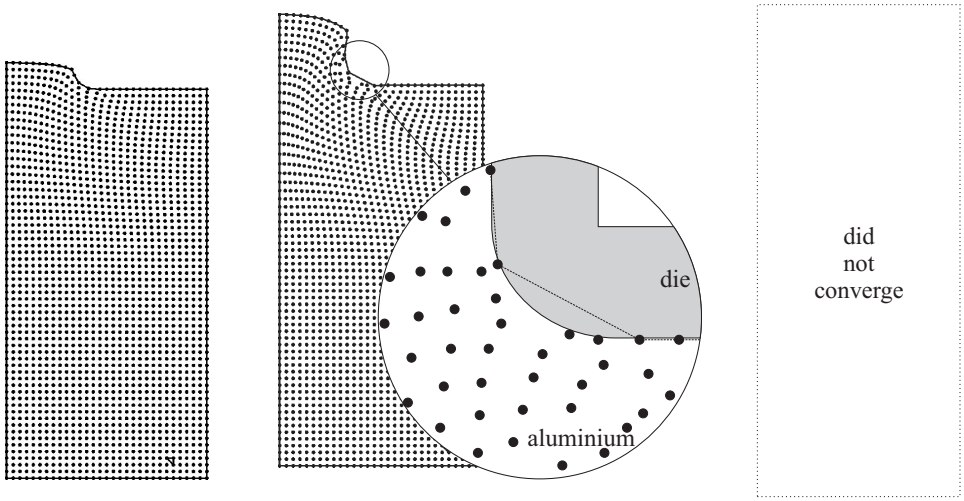


(a) $t = 1.5$ (b) $t = 3.0$ (c) $t = 4.5$ (d) $t = 6.0$

Figure 6.11: The deformed shapes of the billet for four time steps using the method of α-shapes for every increment.

Firstly the deformed geometry of the ASFEM simulation will be examined. Figure 6.11 shows the node sets for four stages out of the extrusion process. Several aspects can be observed from the four figures. The dead metal zone in the top right corner is clearly visible. At the line of symmetry, the aluminium is stretched vertically considerably. A distinct band forms diagonally from the container wall to the die radius. In this shear band, nodes are moving towards the die radius and eventually end up on the right-hand side edge of the exiting profile. The α-shape criterion, which retrieves the boundary for each time frame, marks these nodes as boundary nodes. Hence new boundary can form or disappear as a result of this criterion. This is also the reason that the right-hand side of the extruded profile is not completely straight, but appears somewhat 'ragged'.

In continuum solid mechanics, it is common to keep internal nodes internal and boundary nodes on the boundary. However, if this strategy is adopted and the boundary is not updated during the simulation, problems will occur. Figure 6.12 shows a simulation in which nodes on the boundary are kept on the boundary and internal nodes remain internal. All nodes are subjected to contact conditions with the container wall. After a set of steps, as a result of the boundary strategy, material is excessively being stretched over the die radius, as can be seen in Figure 6.12(b). Two neighbouring nodes, of which one is at the dead metal zone and the other is already exiting the die, will cause this situation. Finally the Newton–Raphson algorithm failed to converge. The method of α-shapes used on every increment of the simulation is clearly preferred for the extrusion process.



(a) $t = 0.75$ (b) $t = 2.25$. The dashed line indicates the boundary of the aluminium. (c) $t = 3.0$

Figure 6.12: The deformed shapes of the billet for four time steps using the method of α-shapes on the first increment only.

Secondly, the ram force is examined. Figure 6.13 shows the ram force plotted versus the simulation time. The simulation time for the ASFEM computation is 6 seconds. The FEM simulation reaches a steady state after 1 second. The ram force remains constant thereafter. The ASFEM simulation approaches the FEM simulation accurately. Note that if a method is suffering from volumetric locking, the ram force would be severely overestimated, which is clearly not the case here.

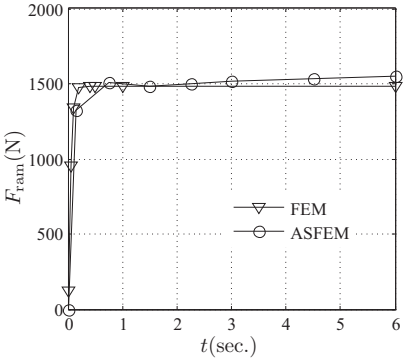


Figure 6.13: The ram force plotted for the time of the simulation

The material used in the extrusion simulation is nearly incompressible because of the high modulus of elasticity in comparison to the low flow stress. Using a method that does not suffer from volumetric locking is essential for obtaining correct forces and stresses. A downside of using a method that does not lock can be that the volumetric constraint is not sufficiently enforced. Especially for non-polynomial meshless approximations, this can be a concern and volume might not be conserved during a simulation. In this section the volume conservation is examined for both simulations. For each method, an error criterion is introduced. For the FEM simulation the error in volume conservation e_{vol} is computed as follows:

$$e_{\text{vol}} = \frac{f_{\text{out}} - f_{\text{in}}}{f_{\text{in}}} \times 100\% \qquad \text{where:} \quad f_{\text{in}} = \int_{\Gamma_{\text{in}}} \mathbf{v} \cdot \mathbf{n} \, \mathrm{d}\Gamma \qquad (6.4)$$

$$\text{and:} \quad f_{\text{out}} = \int_{\Gamma_{\text{out}}} \mathbf{v} \cdot \mathbf{n} \, \mathrm{d}\Gamma$$

where $\mathbf{n}$ is the outward normal on the boundary Γ. For the ASFEM method, the error is computed as follows:

$$e_{\text{vol}} = \frac{\Omega - \Omega_0}{\Omega_0} \times 100\% \qquad (6.5)$$

where Ω_0 is the initial volume and Ω is the volume at the end of the simulation ($t = 6$).

Table 6.3 gives the error in the volume conservation in percentages. It can be seen that the volume is conserved excellently for both methods. For the finite element simulation an insignificant amount of material is gained whilst for the ASFEM method

a negligible amount is lost. The latter method redetermines the boundary (hence implicitly the volume of the body) for each increment by means of the α-shape criterion, however no significant error is observed in the volume conservation.

Table 6.3: The error in the volume conservation for the extrusion process in percentages.

	FEM	ASFEM
$e_{\text{vol}}(\%)$	0.0178	-0.0867

Finally, contour plots between the two simulations will be compared. Figure 6.14 gives the contour plots of the magnitude of the velocity for both methods. Figure 6.14(a) gives the FEM results and 6.14(b) gives the ASFEM results. No significant differences in the velocity fields are observed. A contour plot of the equivalent plastic strain rate is given in Figure 6.15. Overall, the contour plots correspond well. The FEM results show a slightly higher rate at the die radius, which can be a result of the dense FEM mesh at the die radius. Besides this difference, it can be stated that although the two methods employ very different formulations, good agreement is obtained.

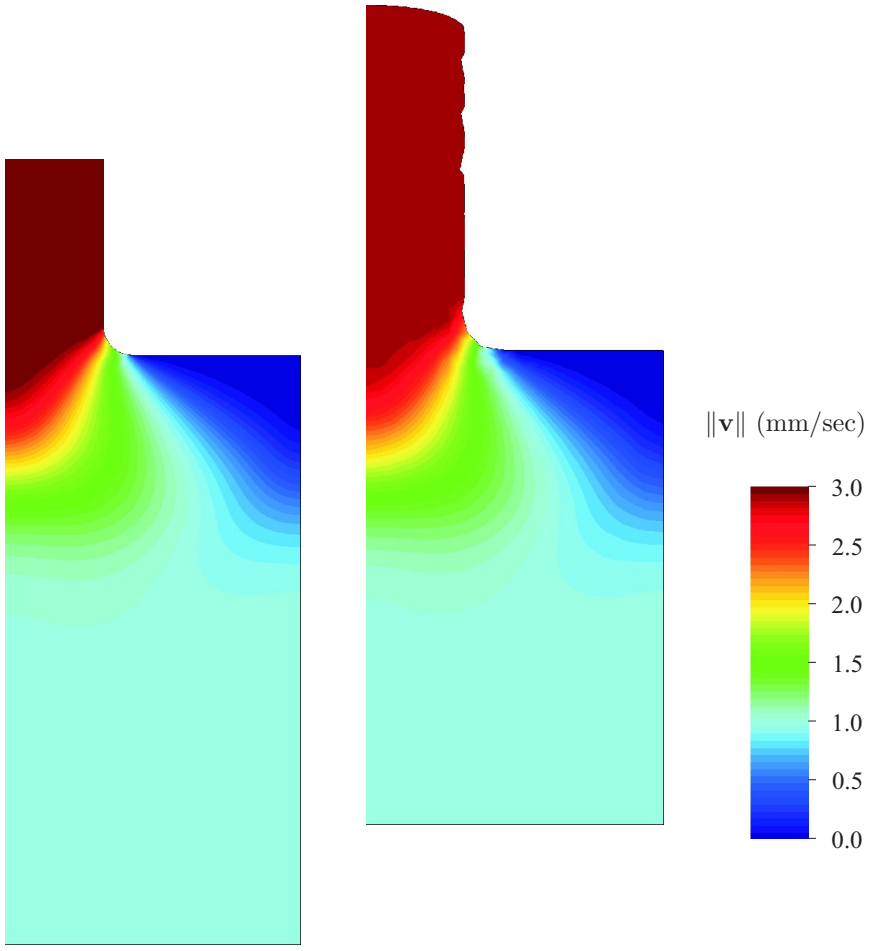
6.4 Conclusions

In this chapter, the performance of the ASFEM method was examined on two forming processes.

Firstly, the ASFEM method was successfully applied on the forging of a circular rod. The mesh was revised for every increment at the areas where most deformation took place. No mapping of the material data was required. It was demonstrated that the mesh of the ASFEM method is of better quality than the FEM mesh. Having no re-triangulations for the ASFEM simulation results in mesh of lesser quality. Finally it was shown that the results of the two methods are in very close agreement.

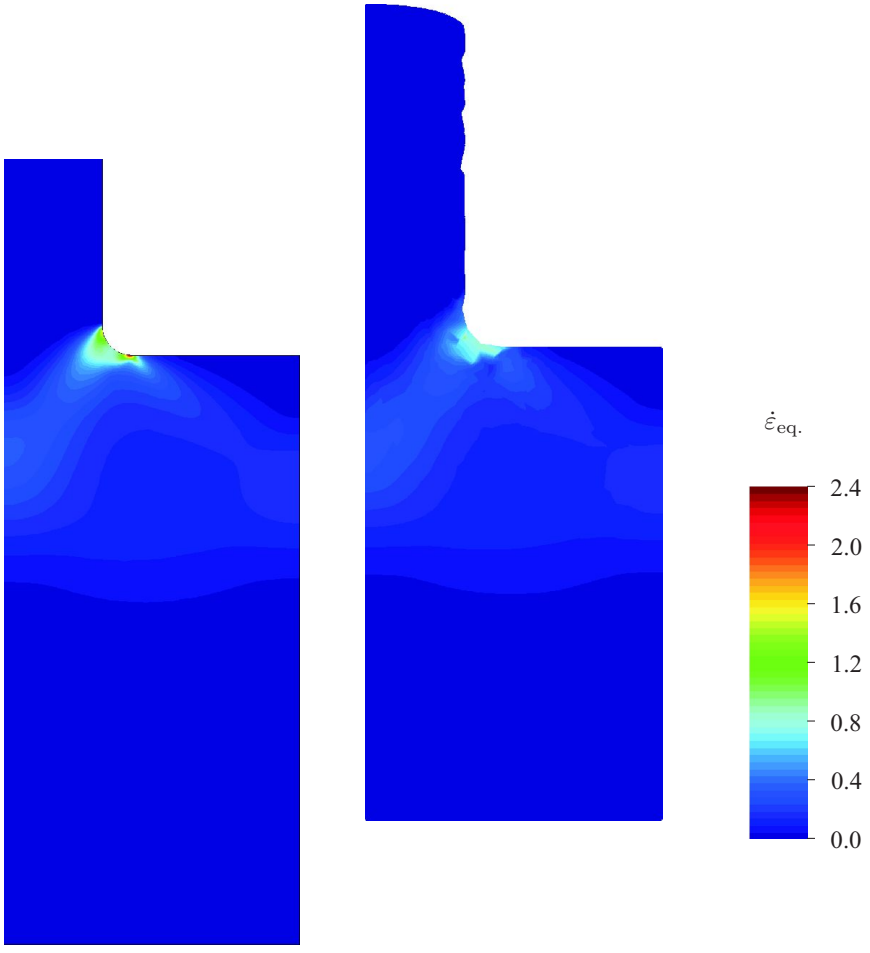
Secondly, the extrusion of a profile was simulated with the ASFEM method in a Lagrangian formulation. The results were compared with a finite element simulation based on a Eulerian formulation. The deformation patterns and the ram force compare well. It was shown that the volume is conserved to a good extent. The method of α-shapes proved to be essential in order to simulate the process.

It can be concluded that results made with the proposed method are in general comparing well to reference solutions made with finite elements. Hence, it can be expected that the results of the ASFEM method can be trusted, when simulating comparable forming processes.



(a) Eulerian FEM (b) ASFEM

Figure 6.14: A contour plot of the magnitude of the velocity for the two simulations



(a) Eulerian FEM (b) ASFEM

Figure 6.15: A contour plot of the equivalent strain rate for the two simulations

7

Conclusions

In this thesis, a nodal-based method is developed which is aimed at simulating large deformations as occur in forming process. Preceding this development, a comparative study on the performance of meshless and nodal-based methods was performed in order to select the most promising method in the field. In this comparative study, two diffuse meshless shape functions, a linear triangle interpolation and two numerical integration schemes were compared. Based on this research, it can be concluded that:

- diffuse meshless shape functions, such as moving least squares or local maximum entropy, are more accurate than linear triangle interpolation but are computationally expensive. Brief remarks about this are made in literature, however this thesis presents quantitative results on this issue and shows that the difference in computational efficiency can be in the order of up to fifteen.
- regardless of the shape function, using a Gaussian integration scheme in incompressibility results either in volumetric locking or in instabilities. A nodal integration scheme, on the other hand, gives locking-free results and gives good accuracy on highly irregular nodal grids. A drawback is that the pressure field cannot be guaranteed to be stable.
- based on the conclusion above, it can be said that selecting a good numerical integration scheme is more important than selecting a shape function.
- the linear triangle interpolation in combination with the nodal integration scheme gives a simple, efficient and reasonably accurate numerical scheme.

Taking the last conclusion as given above into consideration, the linear triangle interpolation in combination with a nodal integration scheme was chosen for further development. A formulation for large deformations was proposed, of which the key two aspects are a cloud of nodes following a Lagrangian description of motion, and a triangulation algorithm which re-triangulates the domain for each increment. The resulting method was named the adaptive smoothed finite element method (ASFEM).

The analysis of a forging and extrusion process showed good agreement between the ASFEM simulations and reference solutions made with finite elements. In the forging example, it was shown that the triangulation used for the ASFEM simulation is of better quality than that of the finite element simulation. For the extrusion problem, the ASFEM results compared well to a reference solution made with Eulerian finite elements. To conclude, it can be stated that with the proposed method, large deformations can be simulated in a Lagrangian formulation without running into mesh distortion problems.

8

Recommendations

In Chapter 3 an investigation on the stability of the pressure prediction for methods using a nodal averaging strategy was presented. It was shown that for nearly incompressible materials, the pressure prediction contains a nonphysical oscillatory mode. No modifications of the formulation have been introduced in order to counter this deficiency. Numerical methods that do not suffer from locking and give a stable pressure prediction require a specific combination of pressure and displacement fields. For finite elements, for instance, it is known which combinations work and which do not. For meshless and other nodal-based methods, this is to some extent an open topic. Apparently, the discontinuous pressure field as resulting from the nodal averaging strategy leave the pressure field with too much freedom. Research into different types of pressure fields (for instance continuous fields) is required in order to find the combination that gives proper pressure predictions.

A formulation for large deformation of the linear triangle interpolation in combination with a nodal averaging scheme was given in Chapter 5. One of the key aspects of the resulting method is the Delaunay triangulation algorithm on which it relies to great extent. In the current study, the triangulation is constructed on every increment of the simulation, even if there is no need for a new triangulation. Luckily, for the current study the computational cost of the triangulation algorithm is found to be small with respect to the total computation time. However, in 3D and for larger node sets, it is known that the computational effort required for the triangulation algorithm becomes larger as a result of its complexity. Therefore, it is interesting to investigate the option in which only at certain time steps and at certain regions the triangulation is revised. With 'triangle-flipping' algorithms, for instance, it is possible to adapt the triangulation locally. Clearly, the equivalent plastic strain rate gives a good indication of the locations which require an updated triangulation (see Section 6.2).

In Chapter 6, the method of α-shapes was used to determine the boundary of the domain for the extrusion process. The method performed well in this application. The volume of the domain was preserved accurately and a reasonable description of the boundary was obtained. There is, however, no physical theorem supporting

the method since it is based purely on geometrical concepts. For the extrusion simulation, results indicate that the boundary computed with the α-shape criterion consists of nodes that were initially in the interior of the billet. This gives rise to the question of whether this 'boundary creation' is observed in practice as well. A detailed investigation should give an answer to this question.

The simulation of the extrusion problem showed especially around the die corner large deformations locally, while in the rest of the billet the deformations were relatively small. Clearly, the regular grid of nodes with which the simulation is started, is not the most suitable arrangement for accurately approximating this highly irregular distribution of deformation. Ideally, more nodes are required around the die radius and everywhere besides this region, the nodal distribution should be coarse. An interesting aspect of the developed method is that nodes can be inserted and removed relatively easily, when compared to finite elements for instance. If one detects the region subject to large deformations, additional nodes can be inserted into that region, and eventually be removed afterwards. Data concerning the material state of a newly added node can be initialised by simply using the data of the neighbouring nodes. Mapping algorithms as required for re-meshing in the finite element method are not required. Within the project as presented in this thesis, a study on the topic of adaptivity and the ASFEM method has been performed by M. Dekker [39]. In this study a detection algorithm (where to refine), an insertion algorithm (how to refine) and a projection algorithm (how to initialize data for new nodes) have been implemented and tested thereafter on academical problems. Clearly, due to the nodal-based formulation of the method, the refinement procedure is remarkably simple and is worth further investigation and development in order to use it in the simulation of industrial forming processes.

A

Selected Topics on Tensors

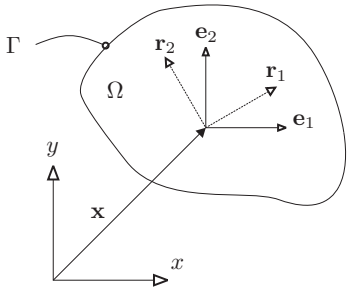


Figure A.1: The two sets of base vector for a specific point $\mathbf{x}$.

Assume a body in space as shown in Figure A.1. For a specific point $\mathbf{x}$, two sets of orthogonal base vectors are defined. Base vectors $\{\mathbf{e}_1, \mathbf{e}_2\}$, are aligned with the coordinates axis x and y respectively. Base vectors $\{\mathbf{r}_1, \mathbf{r}_2\}$ are a rotated set of vectors $\{\mathbf{e}_1, \mathbf{e}_2\}$. Tensor quantities expressed in set $\{\mathbf{r}_1, \mathbf{r}_2\}$ are denoted with the breve accent ($\breve{\cdot}$). The derivations given in this appendix are given for the 2D case.

Voigt Notation

Throughout this thesis, tensors are frequently expressed in Voigt form. Equations appearing in finite element analysis are more comprehendible in this form, and allow for the direct implementation in a computer code. First and second order tensors in Voigt form are given in curly brackets { } and fourth order tensors are given in straight brackets [ ]. Below is a listing of the Voigt forms and the equivalent tensor form of some commonly used tensors. The symbol $\Longleftrightarrow$ is used to indicate this equivalence.

The difference between these two ways of notation is as follows:

$$\mathbf{v} = v_i \mathbf{e}_i = \left\{ \begin{array}{c} v_1 \\ v_2 \end{array} \right\} \qquad \Longleftrightarrow \qquad \{\mathbf{v}\} = \left\{ \begin{array}{c} v_1 \\ v_2 \end{array} \right\} \tag{A.1}$$

$$\boldsymbol{\sigma} = \sigma_{ij} \mathbf{e}_i \mathbf{e}_j = \left[\begin{array}{cc} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{array} \right] \qquad \Longleftrightarrow \qquad \{\boldsymbol{\sigma}\} = \left\{ \begin{array}{c} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{array} \right\} \tag{A.2}$$

$$\boldsymbol{\varepsilon} = \varepsilon_{ij} \mathbf{e}_i \mathbf{e}_j = \left[\begin{array}{cc} \varepsilon_{11} & \varepsilon_{12} \\ \varepsilon_{21} & \varepsilon_{22} \end{array} \right] \qquad \Longleftrightarrow \qquad \{\boldsymbol{\varepsilon}\} = \left\{ \begin{array}{c} \varepsilon_{11} \\ \varepsilon_{22} \\ 2\varepsilon_{12} \end{array} \right\} \tag{A.3}$$

$$\mathbf{L} = L_{ij} \mathbf{e}_i \mathbf{e}_j = \left[\begin{array}{cc} L_{11} & L_{12} \\ L_{21} & L_{22} \end{array} \right] \qquad \Longleftrightarrow \qquad \{\mathbf{L}\} = \left\{ \begin{array}{c} L_{11} \\ L_{12} \\ L_{21} \\ L_{22} \end{array} \right\} \tag{A.4}$$

$$\mathbf{C} = C_{ijkl} \mathbf{e}_i \mathbf{e}_j \mathbf{e}_k \mathbf{e}_l = \ldots \qquad \Longleftrightarrow \qquad [\mathbf{C}] = \left[\begin{array}{ccc} C_{1111} & C_{1122} & C_{1112} \\ C_{2211} & C_{2222} & C_{2212} \\ C_{1211} & C_{1222} & C_{1212} \end{array} \right] \tag{A.5}$$

where subscripts 1 and 2 correspond to the x and y directions along the $\mathbf{e}_1$ and $\mathbf{e}_2$ base vectors respectively. Note that the constitutive tensor $\mathbf{C}$ is rewritten in Voigt form using the plane strain assumption as is frequently used throughout this thesis. For plane stress, the terms contained in the Voigt form are different.

Note that if the nodal velocity tensor $\mathbf{d}$ is expressed in curly brackets, the vector of all nodal velocities is meant. The equivalence is as follows:

$$\mathbf{d}_i = d_j^i \mathbf{e}_j = \left\{ \begin{array}{c} d_1^i \\ d_2^i \end{array} \right\} \qquad \text{for } i = 1 \ldots N_{\text{nod}} \qquad \Longleftrightarrow \qquad \{\mathbf{d}\} = \left\{ \begin{array}{c} \mathbf{d}_1 \\ \mathbf{d}_2 \\ \mathbf{d}_3 \\ \vdots \\ \mathbf{d}_{N_{\text{nod}}} \end{array} \right\} \tag{A.6}$$

where N_{nod} is the number of nodes in the domain, i is the index related to node i and d_1^i is the nodal degree of freedom along the x direction ($\mathbf{e}_1$) of node i. The force vectors are defined similarly:

$$\mathbf{F}_i = \left\{ \begin{array}{c} F_x^i \\ F_y^i \end{array} \right\} \qquad \text{for } i = 1 \ldots N_{\text{nod}} \qquad \Longleftrightarrow \qquad \{\mathbf{F}\} = \left\{ \begin{array}{c} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \mathbf{F}_3 \\ \vdots \\ \mathbf{F}_{N_{\text{nod}}} \end{array} \right\} \tag{A.7}$$

Please note that the symbol $\mathbf{F}$ refers here to the force vector and not to the deformation gradient.

Spatial Gradients

The pre-gradient is defined as:

$$\overrightarrow{\nabla} \ldots = \mathbf{e}_j \frac{\partial}{\partial x_j} \ldots \tag{A.8}$$

taking the pre-gradient for instance towards the velocity field gives:

$$\overrightarrow{\nabla}\mathbf{v} = \mathbf{e}_j \frac{\partial}{\partial x_j} v_i \mathbf{e}_i \tag{A.9}$$

$$= v_{i,j} \mathbf{e}_j \mathbf{e}_i \tag{A.10}$$

The post-gradient is defined as:

$$\ldots \overleftarrow{\nabla} = \frac{\partial}{\partial x_j} \ldots \mathbf{e}_j \tag{A.11}$$

applying this post-gradient on the velocity field gives:

$$\mathbf{v}\overleftarrow{\nabla} = \frac{\partial}{\partial x_j} v_i \mathbf{e}_i \mathbf{e}_j \tag{A.12}$$

$$= v_{i,j} \mathbf{e}_i \mathbf{e}_j \tag{A.13}$$

which is the transpose of the taking the pre-gradient:

$$\overrightarrow{\nabla}\mathbf{v} = (\mathbf{v}\overleftarrow{\nabla})^{\mathrm{T}} \tag{A.14}$$

In this thesis also a gradient is used without a prefix. This gradient is simply equal to the pre-gradient defined as:

$$\nabla\mathbf{v} = \overrightarrow{\nabla}\mathbf{v} \tag{A.15}$$

Hence the rate of deformation can be expressed as:

$$\mathbf{D} = \frac{1}{2}(\overrightarrow{\nabla}\mathbf{v} + \mathbf{v}\overleftarrow{\nabla}) \qquad \text{or} \qquad \mathbf{D} = \frac{1}{2}(\nabla\mathbf{v} + (\nabla\mathbf{v})^{\mathrm{T}}) \tag{A.16}$$

The equilibrium equation in tensor form, as used many times in this thesis, can be reduced to a more familiar form as follows:

$$\boldsymbol{\sigma} \cdot \overleftarrow{\nabla} + \mathbf{f} = \frac{\partial}{\partial x_k} \sigma_{ij} \mathbf{e}_i \mathbf{e}_j \cdot \mathbf{e}_k + f_i \mathbf{e}_i \tag{A.17}$$

$$= \sigma_{ij,j} \mathbf{e}_i + f_i \mathbf{e}_i \tag{A.18}$$

Rotating Tensors

The values of any tensor are expressed by the directions of the base vectors. These vectors can be chosen freely, and expressions can be constructed in which sets of base

vectors are related. The rotation between two sets of base vectors, for instance, is given as:

$$\mathbf{r}_i = R_{ij}\mathbf{e}_j \tag{A.19}$$

Rotating a first order tensor to a different set of base vectors is as follows. The velocity vector is used as an example.

$$\mathbf{v} = v_i\mathbf{e}_i = \breve{v}_j\mathbf{r}_j \tag{A.20}$$
$$= \breve{v}_j R_{ji}\mathbf{e}_i \tag{A.21}$$

Simplifying the result and writing it in Voigt form gives:

$$v_i = R_{ji}\breve{v}_j \qquad \Longleftrightarrow \qquad \{\mathbf{v}\} = [\mathbf{R}]^{\mathrm{T}}\{\breve{\mathbf{v}}\} \tag{A.22}$$

where matrix $[\mathbf{R}]$ rotates the first order tensor in Voigt form:

$$[\mathbf{R}] = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix} \tag{A.23}$$

Rotating a second order tensor, in this example the stress tensor, is as follows:

$$\boldsymbol{\sigma} = \sigma_{ij}\mathbf{e}_i\mathbf{e}_j = \breve{\sigma}_{kl}\mathbf{r}_k\mathbf{r}_l \tag{A.24}$$
$$= \breve{\sigma}_{kl}R_{ki}\mathbf{e}_i R_{lj}\mathbf{e}_j \tag{A.25}$$

giving:

$$\sigma_{ij} = R_{ki}\breve{\sigma}_{kl}R_{lj} \qquad \Longleftrightarrow \qquad \{\boldsymbol{\sigma}\} = [\mathbf{P}]\{\breve{\boldsymbol{\sigma}}\} \tag{A.26}$$

where matrix $[\mathbf{P}]$ rotates the second order stress tensor in Voigt form:

$$[\mathbf{P}] = \begin{bmatrix} R_{11}^2 & R_{21}^2 & 2R_{11}R_{21} \\ R_{12}^2 & R_{22}^2 & 2R_{22}R_{12} \\ R_{11}R_{12} & R_{21}R_{22} & R_{11}R_{22} + R_{12}R_{21} \end{bmatrix} \tag{A.27}$$

B

Some Kernel Functions

The most frequently used kernel, window or weighting functions will be given in this appendix. Firstly, the kernel functions are given for the 1D case. Thereafter, two methods are presented in order to extend these functions for use in 2D or 3D.

Table B.1 gives the name and the formulation of some commonly used kernel functions.

Table B.1: Some commonly used kernel functions.

name	formulation
exponential kernel	$\omega_1(s) = \begin{cases} \exp(-(\frac{s}{\alpha})^2) & \text{for } 0 \leq s \leq 1 \\ 0 & \text{for } s > 1 \end{cases}$
quadratic spline	$\omega_1(s) = \begin{cases} 1 - s^2 & \text{for } 0 \leq s \leq 1 \\ 0 & \text{for } s > 1 \end{cases}$
cubic spline	$\omega_1(s) = \begin{cases} \frac{2}{3} - 4\,s^2 + 4s^3 & \text{for } s \leqslant \frac{1}{2} \\ \frac{4}{3} - 4\,s + 4\,s^2 - \frac{4}{3}s^3 & \text{for } \frac{1}{2} < s \leqslant 1 \\ 0 & \text{for } s > 1 \end{cases}$
quartic spline	$\omega_1(s) = \begin{cases} 1 - 6s^2 + 8s^3 - 3s^4 & \text{for } 0 \leq s \leq 1 \\ 0 & \text{for } s > 1 \end{cases}$
quintic spline	$\omega_1(s) = \begin{cases} (1-s)^5 - 16(\frac{1}{2} - s)^5 & \text{for } 0 \leq s \leq \frac{1}{2} \\ (1-s)^5 & \text{for } \frac{1}{2} < s \leq 1 \\ 0 & \text{for } s > 1 \end{cases}$

The first technique in order to use one of the kernel functions as given above in 2 or 3D is to use a a circular footprint. The dimensionless coordinate s will be defined as:

$$s = \frac{d(\mathbf{x}, \boldsymbol{\xi})}{\beta} \tag{B.1}$$

where $d(\mathbf{x}, \boldsymbol{\xi})$ is the Euclidean distance between two points:

$$d(\mathbf{x}, \boldsymbol{\xi}) = \|\mathbf{x} - \boldsymbol{\xi}\| \tag{B.2}$$

The absolute size of the footprint β is found by considering:

$$\beta = \gamma \cdot h \tag{B.3}$$

where parameter h is an average value of the nodal spacing and γ is a relative size of the footprint. See Figure 2.7 for an illustration on parameters β, γ and h. The kernel function in any dimension can be found using Equation (B.1) to evaluate ω_1:

$$\omega(\mathbf{x} - \boldsymbol{\xi}, \gamma) = \omega_1(s) \tag{B.4}$$

$$\text{where} \quad s = \frac{d(\mathbf{x}, \boldsymbol{\xi})}{\gamma \cdot h} \tag{B.5}$$

The second technique presented in this appendix is to use a square footprint. Instead of using the Euclidean distance, two dimensionless coordinates s_x and $_y$ will be computed. The kernel in 2D becomes a multiplication of the two splines as a function of these two coordinates:

$$\omega(\mathbf{x} - \boldsymbol{\xi}, \gamma) = \omega_1(s_x) \cdot \omega_1(s_y) \tag{B.6}$$

$$\text{where} \quad s_x = \frac{d(x, \xi)}{\gamma \cdot h} \quad \text{and} \quad s_y = \frac{d(y, \eta)}{\gamma \cdot h} \tag{B.7}$$

Whereas the previous technique results in a shape function with a circular footprint, the footprint made with this technique has the shape of a square.

C

Enriched Nodal Averaging Implementation Details

This appendix gives aspects regarding the implementation of the enriched nodal averaging scheme. Equations (4.31), (4.32) and (4.33) need to be numerically evaluated. All three equations consist of a volume integral over arbitrarily shaped polygons. In order to compute these integrals, they will be rewritten as contour integrals. The contour integral can easily be evaluated by numerical integration over the boundary. See Figure C.1 for an illustration on the indices used in the following derivations. Note that the index i previously used to indicate a node is dropped for reasons of clarity.

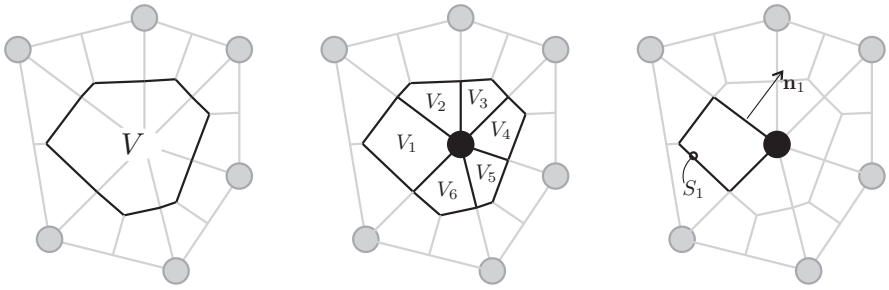


Figure C.1: An illustration of the used indices for implementing the enriched cell strains.

First of all the origin of the location vector $\boldsymbol{\xi}$ for each cell has to be found. The centre of gravity of a cell is found by:

$$\mathbf{x}_\mathrm{c} = \frac{1}{V} \int_V \mathbf{x} \, \mathrm{d}V \tag{C.1}$$

where V is the volume of the cell belonging to a node.

Secondly, Equation (4.31) will be worked out numerically. The shape function used in Chapter 4 is a linear triangle interpolation. The gradient of this shape function is constant within a triangle and discontinuous over the boundaries of the triangles. Using this knowledge of the displacement field, Equation (4.31) can be simplified as follows:

$$[\mathbf{B}_0] = \frac{1}{\int_V 1 \, \mathrm{d}V} \int_V [\mathbf{B}(\boldsymbol{\xi})] \, \mathrm{d}V \tag{C.2}$$

$$= \frac{1}{V} \sum_{j=1}^{N_{\text{tri}}} [\mathbf{B}]_j V_j \tag{C.3}$$

where N_{tri} is the number of triangles attached to the node under consideration, $\mathbf{B}_j$ is the strain matrix of triangle j attached to this node, and V_j is the volume of subcell j. The volume of this subcell can be found by converting the volume integral to a boundary integral:

$$V_j = \int_{S_j} \xi \cdot n_j^x \, \mathrm{d}S \tag{C.4}$$

where S_j is the boundary of subcell V_j and n_j^x is the normal on this boundary in the direction of x. The total volume of the cell can be computed by considering:

$$V = \sum_{j=1}^{N_{\text{tri}}} V_j \tag{C.5}$$

Thirdly, evaluating Equation (4.32) will prove to be somewhat more laborious. The equation will be divided in parts:

$$[\mathbf{B}_\xi] = \underbrace{\frac{1}{\int_V \xi^2 \, \mathrm{d}V}}_{a} \underbrace{\int_V [\mathbf{B}(\boldsymbol{\xi})] \xi \, \mathrm{d}V}_{b} \tag{C.6}$$

The first term in the denominator of part a can be computed as:

$$\int_V \xi^2 \, \mathrm{d}V = \sum_{j=1}^{N_{\text{tri}}} \int_{V_j} \xi^2 \, \mathrm{d}V \tag{C.7}$$

$$= \sum_{j=1}^{N_{\text{tri}}} \int_{S_j} \frac{1}{3} \xi^3 \cdot n_j^x \, \mathrm{d}S \tag{C.8}$$

where the divergence theorem has been used to derive Equation (C.8) from Equation (C.7). The second term, part b, can be computed as follows:

$$\int_V [\mathbf{B}(\boldsymbol{\xi})] \xi \, \mathrm{d}V = \sum_{j=1}^{N_{\text{tri}}} [\mathbf{B}]_j \int_{S_j} \frac{1}{2} \xi^2 \cdot n_j^x \, \mathrm{d}S \tag{C.9}$$

The last term of this equation is the area moment of subcell V_j.

To conclude, all terms appearing in the scheme can be computed with simple boundary integrals on the boundary of the subcells V_j. A two-point integration rule is sufficient to compute the terms exactly. For computational speed a lower order rule can be used.

D

Derivations for Large Deformations Accompanying Chapter 5

The Gradient towards the Midpoint Configuration

The stress update algorithm as given in Section 5.3.2 uses spatial gradients taken towards an intermediate or 'midpoint' configuration in order to give an accurate prediction of the strain and the rotation. In this section, the midpoint gradient shown in Equation (5.30) will be derived. To enhance readability, indices related to the increment under consideration will be dropped. Instead, only an initial and final set of coordinates are considered, given by $\mathbf{X}$ and $\mathbf{x}$ respectively.

These two sets of coordinates are related as follows:

$$\mathbf{x} = \mathbf{X} + \mathbf{u} \tag{D.1}$$

where $\mathbf{u}$ is the displacement. The coordinates at the intermediate configuration are given by $\mathcal{X}$, and are exactly halfway the step, hence the name 'midpoint' rule:

$$\mathcal{X} = \mathbf{X} + \frac{1}{2}\mathbf{u} \tag{D.2}$$

The spatial gradient of for instance a vector field $\mathbf{u}$ towards this intermediate set of coordinates is given by:

$$\frac{\partial \mathbf{u}}{\partial \mathcal{X}} = \underbrace{\frac{\partial \mathbf{u}}{\partial \mathbf{X}}}_{a} \cdot \underbrace{\frac{\partial \mathbf{X}}{\partial \mathcal{X}}}_{b} \tag{D.3}$$

Term b can be found by considering:

$$\frac{\partial \mathcal{X}}{\partial \mathbf{X}} = \frac{\partial \mathbf{X} + \frac{1}{2}\mathbf{u}}{\partial \mathbf{X}} \tag{D.4}$$

$$= \mathbf{1} + \frac{1}{2}\frac{\partial \mathbf{u}}{\partial \mathbf{X}} \tag{D.5}$$

$$= \frac{1}{2}(\mathbf{F} + \mathbf{1}) \tag{D.6}$$

Term a is simply found by the following relation:

$$\frac{\partial \mathbf{u}}{\partial \mathbf{X}} = \mathbf{F} - \mathbf{1} \tag{D.7}$$

Substitution of Equations (D.6) and (D.7) into (D.3), yields:

$$\frac{\partial \mathbf{u}}{\partial \mathcal{X}} = 2(\mathbf{F} - \mathbf{1}) \cdot (\mathbf{F} + \mathbf{1})^{-1} \tag{D.8}$$

System Matrices

In this section, Equation (5.38) will be derived from Equation (5.19). The only difference between the two equations, besides the fact that the latter already incorporates the approximation spaces, is the fact that the former is in tensor form and the latter is in Voigt form.

Firstly, Equation (5.19) is restated below in order to divide it into separate terms:

$$\delta \dot{W}_{\text{int}} = \int_\Omega \underbrace{(\delta \mathbf{L}^{\mathrm{T}} \cdot \mathbf{L}) : \boldsymbol{\sigma}}_{a} \, \mathrm{d}\Omega - 2 \int_\Omega \underbrace{(\delta \mathbf{D} \cdot \mathbf{D}) : \boldsymbol{\sigma}}_{b} \, \mathrm{d}\Omega + \int_\Omega \underbrace{\delta \mathbf{D} : \mathbf{C}^{\mathrm{J}} : \mathbf{D}}_{c} \, \mathrm{d}\Omega + \int_\Omega \underbrace{\delta \mathbf{D} : \boldsymbol{\sigma} \, \mathrm{tr}(\mathbf{D})}_{d} \, \mathrm{d}\Omega \tag{D.9}$$

Choosing the same base vectors for all tensors appearing in part a, and using the property that the contraction of two bases vectors results in the Kronecker delta ($\delta_{ij} = \mathbf{e}_i \mathbf{e}_j$), gives the following derivation:

$$(\delta \mathbf{L}^{\mathrm{T}} \cdot \mathbf{L}) : \boldsymbol{\sigma} = (\delta L_{ji} \mathbf{e}_i \mathbf{e}_j \cdot L_{lk} \mathbf{e}_l \mathbf{e}_k) : \sigma_{mn} \mathbf{e}_m \mathbf{e}_n \tag{D.10}$$

$$= \delta L_{ji} \sigma_{ik} L_{jk} \tag{D.11}$$

the equivalent Voigt form of the equation above becomes:

$$(\delta \mathbf{L}^{\mathrm{T}} \cdot \mathbf{L}) : \boldsymbol{\sigma} \qquad \Longleftrightarrow \qquad \{\delta \mathbf{L}\}^{\mathrm{T}} [\mathbf{T}] \{\mathbf{L}\} \tag{D.12}$$

where $\{\mathbf{L}\}$ is defined according to Equation (A.4) and $[\mathbf{T}]$ is given by:

$$[\mathbf{T}] = \begin{bmatrix} \sigma_{11} & \sigma_{12} & 0 & 0 \\ \sigma_{21} & \sigma_{22} & 0 & 0 \\ 0 & 0 & \sigma_{11} & \sigma_{12} \\ 0 & 0 & \sigma_{21} & \sigma_{22} \end{bmatrix} \tag{D.13}$$

The Voigt form of part b can be derived using the same procedure as used above. The result, including the coefficient in front of the integral, is given by:

$$2(\delta\mathbf{D}\cdot\mathbf{D}):\boldsymbol{\sigma} \qquad \Longleftrightarrow \qquad \{\delta\mathbf{D}\}^{\mathrm{T}}[\mathbf{C}_{\mathrm{spin}}]\{\mathbf{D}\} \tag{D.14}$$

where tensor $\mathbf{D}$ in Voigt form is defined in a similar way as the linear strain tensor in Equation (A.3):

$$\{\mathbf{D}\} = \left\{ \begin{array}{ccc} D_{11} & D_{22} & 2D_{12} \end{array} \right\}^{\mathrm{T}} \tag{D.15}$$

and matrix $\mathbf{C}_{\mathrm{spin}}$ is defined as:

$$[\mathbf{C}_{\mathrm{spin}}] = \left[\begin{array}{ccc} 2\sigma_{11} & 0 & \sigma_{12} \\ 0 & 2\sigma_{22} & \sigma_{12} \\ \sigma_{12} & \sigma_{12} & \frac{1}{2}(\sigma_{11}+\sigma_{22}) \end{array} \right] \tag{D.16}$$

Part c is rewritten in Voigt form as:

$$\delta\mathbf{D}:\mathbf{C}^{\mathrm{J}}:\mathbf{D} \qquad \Longleftrightarrow \qquad \{\delta\mathbf{D}\}^{\mathrm{T}}[\mathbf{C}^{\mathrm{J}}]\{\mathbf{D}\} \tag{D.17}$$

where $\mathbf{C}^{\mathrm{J}}$, for instance for linear elasticity under plane strain assumption, is defined as:

$$[\mathbf{C}^{\mathrm{J}}] = \frac{E}{(1+\nu)(1-2\nu)} \left[\begin{array}{ccc} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 1-2\nu \end{array} \right] \tag{D.18}$$

Part d is written in Voigt form as follows:

$$\delta\mathbf{D}:\boldsymbol{\sigma}\mathrm{tr}(\mathbf{D}) \qquad \Longleftrightarrow \qquad \{\delta\mathbf{D}\}^{\mathrm{T}}[\mathbf{C}_{\mathrm{vol}}]\{\mathbf{D}\} \tag{D.19}$$

where:

$$[\mathbf{C}_{\mathrm{vol}}] = \left[\begin{array}{ccc} \sigma_{11} & \sigma_{11} & 0 \\ \sigma_{22} & \sigma_{22} & 0 \\ \sigma_{12} & \sigma_{12} & 0 \end{array} \right] \tag{D.20}$$

The internal virtual power can now be rewritten:

$$\begin{aligned} \delta\dot{W}_{\mathrm{int}} = \int_{\Omega} \underbrace{\{\delta\mathbf{L}\}^{\mathrm{T}}[\mathbf{T}]\{\mathbf{L}\}}_{a}\,\mathrm{d}\Omega - \int_{\Omega} \underbrace{\{\delta\mathbf{D}\}^{\mathrm{T}}[\mathbf{C}_{\mathrm{spin}}]\{\mathbf{D}\}}_{b}\,\mathrm{d}\Omega + \\ \int_{\Omega} \underbrace{\{\delta\mathbf{D}\}^{\mathrm{T}}[\mathbf{C}^{\mathrm{J}}]\{\mathbf{D}\}}_{c}\,\mathrm{d}\Omega + \int_{\Omega} \underbrace{\{\delta\mathbf{D}\}^{\mathrm{T}}[\mathbf{C}_{\mathrm{vol}}]\{\mathbf{D}\}}_{d}\,\mathrm{d}\Omega \end{aligned} \tag{D.21}$$

By combining the $\mathbf{C}$-matrices as follows:

$$[\mathbf{C}_{\mathrm{tot}}] = [\mathbf{C}_{\mathrm{J}}] - [\mathbf{C}_{\mathrm{spin}}] + [\mathbf{C}_{\mathrm{vol}}] \tag{D.22}$$

the rate of internal work can be simplified accordingly:

$$\delta\dot{W}_{\mathrm{int}} = \int_{\Omega} \{\delta\mathbf{D}\}^{\mathrm{T}}[\mathbf{C}_{\mathrm{tot}}]\{\mathbf{D}\}\,\mathrm{d}\Omega + \int_{\Omega} \{\delta\mathbf{L}\}^{\mathrm{T}}[\mathbf{T}]\{\mathbf{L}\}\,\mathrm{d}\Omega \tag{D.23}$$

Bibliography

[1] I. Alfaro, D. Bel, E. Cueto, M. Doblaré, and F. Chinesta. Three-dimensional simulation of aluminium extrusion by the α-shape based natural element method. *Computer Methods in Applied Mechanics and Engineering*, 195:4269–4286, 2006.

[2] I. Alfaro, L. Fratini, F. Chinesta, and F. Micari. Meshless simulation of friction stir welding. In *NUMIFORM 2007 Materials Processing and Design: Modeling, Simulation and Application*, pages 203–208. American Institute of Physics, 2007.

[3] I. Alfaro, D. González, D. Bel, E. Cueto, M. Doblaré, and F. Chinesta. Recent advances in the meshless simulation of aluminium extrusion and other related forming processes. *Archives of Computational Methods in Engineering*, 11:307–384, 2004.

[4] I. Alfaro, J. Yvonnet, F. Chinesto, and E. Cueto. A study on the performance of natural neighbour-based Galerkin methods. *International Journal for Numerical Methods in Engineering*, 71:1436–1465, 2007.

[5] I. Alfaro, J. Yvonnet, E. Cueto, F. Chinesta, and M. Doblaré. Meshless methods with application to metal forming. *Computer Methods in Applied Mechanics and Engineering*, 195:6661–6675, 2006.

[6] N.R. Aluru. A point collocation method based on reproducing kernel approximations. *International Journal for Numerical Methods in Engineering*, 47:1083–1121, 2000.

[7] N.R. Aluru and G. Li. Finite cloud method: a true meshless technique based on a fixed reproducing kernel approximation. *International Journal for Numerical Methods in Engineering*, 50:2373–2410, 2001.

[8] F.M. Andrade Pires, E.A. de Souza Neto, and J.L. de la Cuesta Padilla. An assessment of the average nodal volume formulation for the analysis of nearly incompressible solids under finite strains. *Communications in Numerical Methods in Engineering*, 20:569–583, 2004.

[9] M. Arroyo and M. Ortiz. Local maximum-entropy approximation schemes: a seamless bridge between finite elements and meshfree methods. *International Journal for Numerical Methods in Engineering*, 65:2167–2202, 2006.

[10] H. Askes, R. de Borst, and O. Heeres. Conditions for locking-free elasto-plastic analysis in the element-free Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 173:99–109, 1999.

[11] W.A. Assaad. *Aluminium extrusion with a deformable die.* PhD thesis, University of Twente, 2010.

[12] S.N. Atluri and T. Zhu. A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Computational Mechanics*, 22:117–127, 1998.

[13] I. Babuška, U. Banerjee, and J.E. Osborn. Meshless and generalized finite element methods: A survey of some major results. In M. Griebel and M.A. Schweitzer, editors, *Meshfree Methods in Partial Differential Equations*, volume 26 of *Lecture notes in Computational science and engineering*, pages 1–20. Springer, 2002.

[14] K.J. Bathe. *Finite Element Procedures.* Klaus-Jürgen Bathe, 2006.

[15] S. Beissel and T. Belytschko. Nodal integration of the element-free Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 139:49–74, 1996.

[16] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: An overview and recent developments. *Computer Methods in Applied Mechanics and Engineering*, 139:3–47, 1996.

[17] T. Belytschko, W.K. Liu, and B. Moran. *Nonlinear Finite Elements for Continua and Structures.* John Wiley & Sons, 2008.

[18] T. Belytschko, Y.Y. Lu, and L. Gu. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering*, 37:229–256, 1994.

[19] T. Belytschko, Y.Y. Lu, and L. Gu. Crack propagation by element-free galerkin methods. *Engineering Fracture Mechanics*, 51:295–315, 1995.

[20] J. Bonet and A.J. Burton. A simple average nodal pressure tetrahedral element for incompressible and nearly incompressible dynamic explicit applications. *Communications in Numerical Methods in Engineering*, 14:437–449, 1998.

[21] J. Bonet and S. Kulasegaram. Correction and stabilization of smooth particle hydrodynamics methods with applications in metal forming simulations. *International Journal for Numerical Methods in Engineering*, 47:1189–1214, 2000.

[22] J. Bonet, S. Kulasegaram, M.X. Rodriguez-Paz, and M. Profit. Variational formulation for the smooth particle hydrodynamics (SPH) simulation of fluid and solid problems. *Computer Methods in Applied Mechanics and Engineering*, 193:1245–1256, 2004.

[23] J. Bonet, H. Marriott, and O. Hassan. Stability and comparison of different linear tetrahedral formulations for nearly incompressible explicit dynamic applications. *International Journal for Numerical Methods in Engineering*, 50:119–133, 2001.

[24] S.P.A. Bordas, T. Rabczuk, N.X. Hung, V.P. Nguyen, S. Natarajan, T. Bog, D.M. Quan, and N.V. Hiep. Strain smoothing in FEM and XFEM. *Computers and Structures*, 88:1419–1443, 2010.

[25] J. Braun and M. Sambridge. A numerical method for solving partial differential equations on highly irregular evolving grids. *Nature*, 376:655–660, 1995.

[26] M. Broccardo, M. Micheloni, and P. Krysl. Assumed-deformation gradient finite elements with nodal integration for nearly incompressible large deformation analysis. *International Journal for Numerical Methods in Engineering*, 78:1113–1134, 2009.

[27] CGAL. Computational Geometry Algorithms Library. *http://www.cgal.org/*.

[28] D. Chapelle and K.J. Bathe. The inf-sup test. *Computers and Structures*, 47:537–545, 1993.

[29] J.S. Chen, C. Pan, C.M.O.L. Roque, and H.P. Wang. A Lagrangian reproducing kernel particle method for metal forming analysis. *Computational Mechanics*, 22:289–307, 1998.

[30] J.S. Chen, C. Pan, C.T. Wu, and W.K. Liu. Reproducing kernel particle methods for large deformation analysis of non-linear structures. *Computer Methods in Applied Mechanics and Engineering*, 139:195–227, 1996.

[31] J.S. Chen, C.M.O.L. Roque, C. Pan, and S.T. Button. Analysis of metal forming process based on meshless method. *Journal of Materials Processing Technology*, 80-81:642–646, 1998.

[32] J.S. Chen and H.P. Wang. New boundary condition treatments in meshfree computation of contact problems. *Computer Methods in Applied Mechanics and Engineering*, 187:441–468, 2000.

[33] J.S. Chen, C.T. Wu, S. Yoon, and Y. You. A stabilized conforming nodal integration for Galerkin mesh-free methods. *International Journal for Numerical Methods in Engineering*, 50:435–466, 2001.

[34] J.S. Chen, S. Yoon, and C.T. Wu. Non-linear version of stabilized conforming nodal integration for Galerkin mesh-free methods. *International Journal for Numerical Methods in Engineering*, 53:2587–2615, 2002.

[35] R.D. Cook, D.S. Malkus, M.E. Plesha, and R.J. Witt. *Concepts and applications of finite element analysis.* John Wiley & Sons, 4 edition, 2002.

[36] E. Cueto, M. Doblaré, and L. Gracia. Imposing essential boundary conditions in the natural element method by means of density-scaled α-shapes. *International Journal for Numerical Methods in Engineering*, 49:519–546, 2000.

[37] E. Cueto, N. Sukumar, B. Calvo, J. Cegoñino, and M. Doblaré. Overview and recent advances in natural neighbour Galerkin methods. *Archives of Computational Methods in Engineering*, 10:307–384, 2003.

[38] S. De and K.J. Bathe. The method of finite spheres. *Computational Mechanics*, 25:329–345, 2000.

[39] M. Dekker. Study and implementation of a refinement procedure for an adaptive smoothed finite element method (CTW.11/TM-5655). Master's thesis, University of Twente, 2011.

[40] B. Delaunay. Sur la sphère vide. A la memoire de Georges Voronoi. *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7:793–800, 1934.

[41] DiekA. A finite element code for forming simulations. http://www.dieka.org.

[42] C.R. Dohrmann, M.W. Heinstein, J. Jung, S.W. Key, and W.R. Witkowski. Node-based uniform strain elements for three-node triangular and four-node tetrahedral meshes. *International Journal for Numerical Methods in Engineering*, 47:1549–1568, 2000.

[43] J. Dolbow and T. Belytschko. Numerical integration of the Galerkin weak form in meshfree methods. *Computational Mechanics*, 23:219–230, 1999.

[44] J. Dolbow and T. Belytschko. Volumetric locking in the element free Galerkin method. *International Journal for Numerical Methods in Engineering*, 46:925–942, 1999.

[45] C.A. Duarte. A review of some meshless methods to solve partial differential equations. Technical report, TICAM - Texas Institute for Computational and Applied Mathematics, 1995.

[46] C.A. Duarte and J.T. Oden. An h-p adaptive method using clouds. *Computer Methods in Applied Mechanics and Engineering*, 139:237–262, 1996.

[47] C.T. Dyka and R.P. Ingel. Adressing tension instability in SPH methods. Technical report, Naval Research Laboratory, Washington, 1994.

[48] H. Edelsbrunner, D.G. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, IT-29:551–559, 1983.

[49] FeaTure. A finite element toolbox in c++. http://www.tm.ctw.utwente.nl.

[50] S. Fernández-Méndez and A. Huerta. Imposing essential boundary conditions in mesh-free methods. *Computer Methods in Applied Mechanics and Engineering*, 193:1257–1275, 2004.

[51] L. Filice, I. Alfaro, F. Gagliardi, E. Cueto, F. Micari, and F. Chinesta. A preliminary comparison between finite element and meshless simulations of extrusion. *Journal of Materials Processing Technology*, 209:3039–3049, 2009.

[52] J.A. García, Ll. Gascón, E. Cueto, I. Ordeig, and F. Chinesta. Meshless methods with application to resin transfer molding simulation. *Computer Methods in Applied Mechanics and Engineering*, 198:2700–2709, 2009.

[53] M.W. Gee, C.R. Dohrmann, S.W. Key, and W.A. Wall. A uniform nodal strain tetrahedron with isochoric stabilization. *International Journal for Numerical Methods in Engineering*, 78:429–443, 2009.

[54] GiD. CIMNE. http://www.gidhome.com.

[55] D. González, E. Cueto, and M. Doblaré. A higher order method based on local maximum entropy approximation. *International Journal for Numerical Methods in Engineering*, 83:741–764, 2010.

[56] D. González, E. Cueto, M.A. Martínez, and M. Doblaré. Numerical integration in natural neighbour Galerkin methods. *International Journal for Numerical Methods in Engineering*, 60:2077–2104, 2004.

[57] J. Gosz and W.K. Liu. Admissable approximations for essential boundary conditions in the reproducing kernel particle method. *Computational Mechanics*, 19:120–135, 1996.

[58] C.F. Guedes and J.M.A. César de Sá. Coupling finite element and meshless methods to deal with contact and friction in forging processes. In J.M.A. César de Sá and A.D. Santos, editors, *NUMIFORM 2007 Materials Processing and Design: Modeling, Simulation and Application*, pages 1507–1511. American Institute of Physics, 2007.

[59] C.F. Guedes and J.M.A. César de Sá. A proposal to deal wit contact and friction by blending meshfree and finite element methods in forming processes. *International Journal of Material Forming*, 1:177–188, 2008.

[60] F.C. Günther and W.K. Liu. Implementation of boundary conditions for meshless methods. *Computer Methods in Applied Mechanics and Engineering*, 163:205–230, 1998.

[61] Y.M. Guo and K. Nakanishi. A backward extrusion analysis by the rigid-plastic integralles-meshless method. *Journal of Materials Processing Technology*, 140:19–24, 2003.

[62] D.L. Hicks, J.W. Swegle, and S.W. Attaway. Conservative smoothing stabilizes discrete-numerical instabilities in SPH material dynamics computations. *Applied Mathematics and Computation*, 85:209–226, 1997.

[63] W. Hu, L.G. Yao, and Z.Z. Hua. Parallel point interpolation method for three-dimensional metal forming simulations. *Engineering Analysis with Boundary Elements*, 31:326–342, 2007.

[64] A. Huerta, T. Belytschko, S. Fernández-Méndez, and T. Rabczuk. Chapter 10: Meshfree methods. In E. Stein, R. de Borst, and T.J.R. Hughes, editors, *Encyclopedia of Computational Mechanics*, volume 1: Fundamentals, pages 279–309. John Wiley & Sons, 2005.

[65] A. Huerta, Y. Vidal, and P. Villon. Pseudo-divergence-free element free Galerkin method for incompressible fluid flow. *Computer Methods in Applied Mechanics and Engineering*, 193:1119–1136, 2004.

[66] J. Huétink. *On the simulation of thermo-mechanical forming processes.* PhD thesis, University of Twente, 1992.

[67] T.J.R. Hughes and J. Winget. Finite rotation effects in numerical integration of rate constitutive equations arising in large-deformation analysis. *International Journal for Numerical Methods in Engineering*, 15:18621867, 1980.

[68] N.X. Hung, S.P.A. Bordas, and N.D. Hung. Addressing volumetric locking and instabilities by selective integration in smoothed finite elements. *Communications in Numerical Methods in Engineering*, 25:19–34, 2009.

[69] S.R. Idelsohn, M. Mier-Torrecilla, and E. Oñate. Multi-fluid flows with the particle finite element method. *Computer Methods in Applied Mechanics and Engineering*, 198:2750–2767, 2009.

[70] S.R. Idelsohn and E. Oñate. To mesh or not to mesh. That is the question. *Computer Methods in Applied Mechanics and Engineering*, 195:4681–4696, 2006.

[71] S.R. Idelsohn, E. Oñate, N. Calvo, and F. Del Pin. The meshless finite element method. *International Journal for Numerical Methods in Engineering*, 58:893–912, 2003.

[72] A.J. Koopman. *Analysis tools for the design of aluminium extrusion dies.* PhD thesis, University of Twente, 2009.

[73] P. Krysl and T. Belytschko. Analysis of thin shells by the element-free Galerkin method. *International Journal of Solids and Structures*, 33:3057–3080, 1996.

[74] P. Krysl and B. Zhu. Locking-free continuum displacement finite elements with nodal integration. *International Journal for Numerical Methods in Engineering*, 76:1020–1043, 2008.

[75] K.C. Kwon, S.H. Park, B.N. Jiang, and S.K. Youn. The least-squares meshfree method for solving linear elastic problems. *Computational Mechanics*, 30:196–211, 2003.

[76] K.C. Kwon, S.H. Park, and S.K. Youn. The least-squares meshfree method for elasto-plasticity and its application to metal forming analysis. *International Journal for Numerical Methods in Engineering*, 64:751–788, 2005.

[77] K.C. Kwon, S.H. Park, and S.K. Youn. The support integration scheme in the least-squares mesh-free method. *Finite Elements in Analysis and Design*, 43:127–144, 2006.

[78] K.C. Kwon and S.K. Youn. The least-squares meshfree method for rigid-plasticity with frictional contact. *International Journal of Solids and Structures*, 43:7450–7481, 2006.

[79] G. Li and T. Belytschko. Element-free Galerkin method for contact problems in metal forming analysis. *Engineering Computations*, 18:62–78, 2001.

[80] S. Li and W.K. Liu. Meshfree and particle methods and their applications. *Applied Mechanics Reviews*, 55:1–34, 2002.

[81] S. Li and W.K. Liu. *Meshfree Particle Methods*. Springer, 2004.

[82] G.R. Liu. *Mesh free methods*. CRC Press, 2003.

[83] G.R. Liu and Y.T. Gu. A point interpolation method for two-dimensional solids. *International Journal for Numerical Methods in Engineering*, 50:937–951, 2001.

[84] G.R. Liu, T.T. Nguyen, K.Y. Dai, and K.Y. Lam. Theoretical aspects of the smoothed finite element method (SFEM). *International Journal for Numerical Methods in Engineering*, 71:902–930, 2007.

[85] T. Liu, G.R. Nguyen-Thoi, H. Nguyen-Xuan, and K.Y. Lam. A node-based smoothed finite element method (NS-FEM) for upper bound solutions to solid mechanics problems. *Computers and Structures*, 87:14–26, 2009.

[86] W.K. Liu, Y. Chen, S. Jun, J.S. Chen, T. Belytschko, C. Pan, R.A. Uras, and C.T. Chang. Overview and applications of the reproducing kernel particle methods. *Archives of Computational Methods in Engineering*, 3:3–80, 1996.

[87] W.K. Liu, W. Han, H. Lu, S. Li, and J. Cao. Reproducing kernel element method. Part i: Theoretical formulation. *Computer Methods in Applied Mechanics and Engineering*, 193:933–951, 2004.

[88] W.K. Liu, S. Jun, S. Li, J. Adee, and T. Belytschko. Reproducing kernel particle methods for structural dynamics. *International Journal for Numerical Methods in Engineering*, 38:1655–1679, 1995.

[89] W.K. Liu, S. Jun, and Y.F. Zhang. Reproducing kernel particle methods. *International Journal for Numerical Methods in Fluids*, 20:1081–1106, 1995.

[90] J. Lof. *Developments in finite element simulations of aluminium extrusion*. PhD thesis, University of Twente, 2000.

[91] L.B. Lucy. A numerical aproach to the testing of the fission hypothesis. *The Astronomical Journal*, 82:1013–1024, 1977.

[92] J.J. Monaghan. An introduction to SPH. *Computer Physics Communications*, 48:89–96, 1988.

[93] J.J. Monaghan. SPH without a tensile instability. *Journal of Computational Physics*, 159:290–311, 2000.

[94] J.P. Morris. A study of the stability properties of smooth particle hydrodynamics. *Publications Astronomical Society of Australia*, 13:97–102, 1996.

[95] B. Nayroles, G. Touzot, and P. Villon. Generalizing the finite element method: Diffuse approximation and diffuse elements. *Computational Mechanics*, 10:307–318, 1992.

[96] T. Nguyen-Thoi, H.C. Vu-Do, T. Rabczuk, and H. Nguyen-Xuan. 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*, pages 3005–3027, 2010.

[97] H. Nguyen-Xuan, S. Bordas, and H. Nguyen-Dang. Smooth finite element methods: Convergence, accuracy and properties. *International Journal for Numerical Methods in Engineering*, 74:175–208, 2008.

[98] A. Ortiz, M.A. Puso, and N. Sukumar. Maximum-entropy meshfree method for compressible and near-incompressible elasticity. *Computer Methods in Applied Mechanics and Engineering*, 199:1859–1871, 2010.

[99] A. Ortiz, M.A. Puso, and N. Sukumar. Maximum-entropy meshfree method for incompressible media problems. *Finite Elements in Analysis and Design*, 47:572–585, 2011.

[100] T. Pannachet and H. Askes. Some observations on the enforcement of constraint equations in the EFG method. *Communications in Numerical Methods in Engineering*, 16:819–830, 2000.

[101] M.A. Puso, J.S. Chen, E. Zywicz, and W. Elmer. Meshfree and finite element nodal integration methods. *International Journal for Numerical Methods in Engineering*, 74:416–446, 2008.

[102] M.A. Puso and J. Solberg. A stabilized nodally integrated tetrahedral. *International Journal for Numerical Methods in Engineering*, 67:841–867, 2006.

[103] Qhull. Quick-hull. *http://www.qhull.org/*.

[104] W. Quak and A.H. van den Boogaard. Adaptive smoothed finite elements for history dependent materials. In *ESAFORM 2011, Belfast, North Ireland.*, 2011.

[105] W. Quak, A.H. van den Boogaard, D. González, and E. Cueto. A comparative study on the performance of meshless approximations and their integration. *Computational Mechanics*, 48:121–137, 2011.

[106] W. Quak, A.H. van den Boogaard, and J. Huétink. Meshless methods and forming processes. *International Journal of Material Forming*, 2 Supplement 1:585–588, 2009.

[107] D.P. Recio, R.M. Natal Jorge, and L.M.S. Dinis. Locking and hourglass phenomena in an element-free Galerkin context: the B-bar method with stabilization and an enhanced strain method. *International Journal for Numerical Methods in Engineering*, 68:1329–1357, 2006.

[108] D.P. Recio, R.M. Natal Jorge, and L.M.S. Dinis. On the use of element-free Galerkin method for problems involving incompressibility. *Engineering Analysis with Boundary Elements*, 31:103–115, 2007.

[109] A.D. Rietman. *Numerical analysis of inhomogeneous deformation in plane strain compression*. PhD thesis, University of Twente, 1999.

[110] X. Shangwu, W.K. Liu, J. Cao, J.M.C. Rodrigues, and P.A.F. Martins. On the utilization of the reproducing kernel particle method for the numerical simulation of plane strain rolling. *International Journal of Machine Tools & Manufacture*, 43:89–102, 2003.

[111] J.C. Simo and T.J.R. Hughes. *Computational Inelasticity*, volume 7 of *Interdisciplinary Applied Mathematics*. Springer, 1998.

[112] N. Sukumar. Construction of polygonal interpolants: a maximum entropy approach. *International Journal for Numerical Methods in Engineering*, 61:2159–2181, 2004.

[113] N. Sukumar, B. Moran, and T. Belytschko. The natural element method in solid mechanics. *International Journal for Numerical Methods in Engineering*, 43:839–887, 1998.

[114] N. Sukumar, B. Moran, A.Y. Semenov, and V.V. Belikov. Natural neighbour Galerkin methods. *International Journal for Numerical Methods in Engineering*, 50:1–27, 2001.

[115] N. Sukumar and R.W. Wright. Overview and construction of meshfree basis functions: From moving least squares to entropy approximants. *International Journal for Numerical Methods in Engineering*, 70:181–205, 2007.

[116] D. Sulsky, Z. Chen, and H.L. Schreyer. A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118:179–196, 1994.

[117] J.W. Swegle, D.L. Hicks, and S.W. Attaway. Smoothed particle hydrodynamics stability analysis. *Journal of Computational Physics*, 116:123–134, 1995.

[118] K.Y. Sze, J.S. Chen, N. Sheng, and X.H. Liu. Stabilized conforming nodal integration: exactness and variational justification. *Finite Elements in Analysis and Design*, 41:147–171, 2004.

[119] S. Timoshenko and J.N. Goodier. *Theory of Elasticity.* McGraw, 1951.

[120] L. Traversoni. Natural neighbour finite elements. *International Conference on Hydraulic Engineering Software Hydrosoft Proceedings*, 2:291–297, 1994.

[121] A.A. van der Stelt, T.C. Bor, H.J.M. Geijselaers, W. Quak, R. Akkerman, and J. Huétink. Comparison of ALE finite element method and adaptive smoothed finite element method for the numerical simulation of friction stir welding. In *ESAFORM 2011, Belfast, North Ireland*, 2011.

[122] Y. Vidal, P. Villon, and A. Huerta. Locking in the incompressible limit: pseudo-divergence-free element free Galerkin. *Communications in Numerical Methods in Engineering*, 19:725–735, 2003.

[123] G.M. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. *Journal f ur die Reine und Angewandte Mathematik*, 133:97–187, 1907.

[124] H. Wang, L. Guangyao, X. Han, and Z.Z. Hua. Development of parallel 3D RKPM meshless bulk forming simulation system. *Advances in Engineering Software*, 38:87–101, 2007.

[125] J.G. Wang and G.R. Liu. A point interpolation meshless method based on radial basis functions. *International Journal for Numerical Methods in Engineering*, 54:1623–1648, 2002.

[126] S. Xiong, W.K. Liu, J. Cao, C.S. Li, J.M.C. Rodrigues, and P.A.F. Martins. Simulation of bulk metal forming processes using the reproducing kernel particle method. *Computers and Structures*, 83:574–587, 2005.

[127] S. Xiong, J.M.C. Rodrigues, and P.A.F. Martins. Application of the element free Galerkin method to the simulation of plate strain rolling. *European Journal of Mechanics and Solids*, 23:77–93, 2004.

[128] S. Xiong, J.M.C. Rodrigues, and P.A.F. Martins. On the background cells during the analysis of bulk forming processes by the reproducing kernel particle method. *Computational Mechanics*, 40:223–247, 2007.

[129] J.W. Yoo, B. Moran, and J.S. Chen. Stabilized conforming nodal integration in the natural-element method. *International Journal for Numerical Methods in Engineering*, 60:861–890, 2004.

[130] S. Yoon and J.S. Chen. Accelerated meshfree method for metal forming simulation. *Finite Elements in Analysis and Design*, 38:937–948, 2002.

[131] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method*, volume 1: The basis. Butterworth-Heinemann, 5 edition, 2000.

[132] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method*, volume 2: Solid and structural mechanics. Butterworth-Heinemann, 2005.

Nawoord

Het onderzoek zoals gepresenteerd in dit proefschrift is uitgevoerd van 2007 tot 2011 als project op de Universiteit Twente binnen de vakgroep Technische Mechanica van de faculteit Construerende Technische Wetenschappen. Het project is een onderdeel geweest van het programma van het onderzoeksinstituut M2i onder nummer MC1.07291. In dit nawoord wil ik van de gelegenheid gebruik maken om alle mensen die direct of indirect hebben bijgedragen aan dit project te bedanken.

Ten eerste wil ik mijn promotor Han Huétink en assistent promotor Ton van den Boogaard bedanken voor hun motiverende begeleiding en voor het gekregen vertrouwen gedurende deze vier jaar. Deze zaken hebben zonder meer bijgedragen aan het eindresultaat en hebben daarnaast van het promotieproject een leerzame ervaring gemaakt voor mij. Daarnaast wil ik ook Bert Geijselaers bedanken voor de begeleiding in de periode dat Ton op de ETH Zürich verbleef, en misschien nog meer voor de vele daarop volgende creatieve discussies.

Ten tweede wil ik de medewerkers bedanken die mij hebben geholpen gedurende mijn promotie. In het bijzonder wil ik de namen noemen van de twee secretaresses Debbie Zimmerman en Tanja Gerrits. Om dit nawoord toch enigszins bondig te houden, zal ik de waslijst van alle zaken waarmee zij mij hebben geholpen achterwege laten. Nico van Vliet en Herman van Corbach bedankt voor de hulp betreffende ICT zaken en het beheren van CRUNCH. Ook Axel Lok natuurlijk bedankt. Als je PC een Trojaans paard opdoet in de laatste maanden van je promotie en er staat binnen en dag of twee een compleet nieuwe machine dan is dat een vermelding waard. Mijn dank aan Semih Perdahcıoğlu voor de vele discussies over de mechanica van grote vervormingen. Deze discussies hebben voor mij absoluut geholpen om deze lastige materie snel onder de knie te krijgen. Ook de bijdrage van Rene ten Thije voor de hulp bij onder meer de implementatie wil ik niet ongenoemd laten. Han Huétink, Ton van den Boogaard, Arnoud van der Stelt en Wouter Grouve, mijn dank voor jullie correcties op het manuscript. Matthijs Dekker en Jeroen Brouwers, bedankt voor jullie afstudeeronderzoek hetgeen heeft plaatsgevonden binnen het promotieproject.

De igual forma, me gustaría agradecer a toda la gente del Grupo GEMM en Zaragoza España por mi estancia de investigación en dicho centro. Recuerdo esos meses con una sonrisa por la calidez y sinceridad de su gente, en un magnífica ciudad. En especial, quiero dar las gracias al Elías Cueto y David González por la supervisión, y por las discusiones en el tema métodos sin malla. Muchas de las ideas y conceptos utilizados

en el desarrollo de esta tesis se originaron durante ese período.

Mijn promotietijd is een tijd waar ik met veel plezier aan zal terugdenken en de collega's van de groepen Technische Mechanica en Productie Technologie hebben hier voor een groot deel aan bijgedragen. Naast de werkgerelateerde zaken, heb ik toch ook zeker de activiteiten buiten werktijd met veel plezier beleefd en wil ik hiervoor mijn collega's bedanken. De ESAFORM2011 conferentie met daaropvolgend een roadtrip door Ierland zal ik niet snel vergeten. Of de waterski-sessies in het Rutbeek, de mountainbike trips in het Teutoburgerwald, de pokeravonden, enzovoort, enzovoort....

Voor een ieder die ik vergeten ben te noemen, hierbij mijn oprechte excuses. Verder wens ik diegenen van de groep die nog bezig zijn met hun promotie een succesvolle afronding toe en de rest,... gewoon het beste!

Wouter Quak
Enschede, september 2011

UNIVERSITY OF TWENTE.

group of applied mechanics